URHO A. RAUHALA
*Geodetic Services, Inc.*
*Indialantic, FL 32903*

# Introduction to Array Algebra*

The elementary principles of array algebra are presented, a small array multiplication is detailed, and a FORTRAN program is devised.

## WHY BOTHER WITH ARRAY ALGEBRA?

MOST MATHEMATICAL SCIENCES deal with the linear algebra and to a greater extent the linear problems treat multidimensional data. For example, the advanced measuring technology with satellites and other computerized instruments produces a flood of digital data related to a space which has at least four local coordinates $x$, $y$, $z$, $t$. Yet, the tools of linear algebra have been centered in solving for a linear system $\underset{mn}{\boldsymbol{A}}\ \underset{n,1}{\boldsymbol{X}} = \underset{m,1}{\boldsymbol{L}} - \underset{m,1}{\boldsymbol{V}}$, where the parameters $\boldsymbol{X}$, $\boldsymbol{V}$ and the observed values $\boldsymbol{L}$ are only one-dimensional vectors.

Array algebra is a new powerful mathematical tool extending the linear algebra to deal with the multidimensional data. The above matrix equation is extended to an $i$-dimensional

ABSTRACT: *Array algebra is a generalization of the vector, matrix, and tensor algebras extending the so-called fast transform technology of information and computer sciences. It forms the fast multilinear algebra for handling gridded data, although some of its fast characteristics can also be utilized in processing monolinear and non-gridded data. Array algebra makes a rigorous solution of millions of parameters computationally feasible, often for the very first time. The way to these generalized concepts can be paved by an educational introduction to the elementary principles of array algebra. Such an introduction is the scope of this presentation. It is based on a collection of the author's lecture notes on array algebra since the late 60's to graduate students at the Royal Institute of Technology in Stockholm and representatives of some U.S. government and research organizations. A small array multiplication is detailed and a FORTRAN program devised for computing a more general consistent "array transform." The connection of this special array multiplication to conventional fast transforms and signal processing is outlined. The generality of array algebra is demonstrated through generalized monolinear operators called loop inverses.*

array equation where $\underset{n_1 n_2 \cdots n_i}{\boldsymbol{X}}$, $\underset{m_1 m_2 \cdots m_i}{\boldsymbol{L}}$, $\underset{m_1 m_2 \cdots m_i}{\boldsymbol{V}}$ are $i$-dimensional arrays associated with $i$ partial design matrices $\underset{m_1 n_1}{\boldsymbol{A}_1}$, $\underset{m_2 n_2}{\boldsymbol{A}_2}$, $\cdots$ $\underset{m_k n_k}{\boldsymbol{A}_k}$ $\cdots$, $\underset{m_i n_i}{\boldsymbol{A}_i}$. In two dimensions the array equation can be expressed as $\underset{m_1 n_1}{\boldsymbol{A}_1}\ \underset{n_1 n_2}{\boldsymbol{X}}\ \underset{n_2 m_2}{\boldsymbol{A}_2^T} = \underset{m_1 m_2}{\boldsymbol{L}} - \underset{m_1 m_2}{\boldsymbol{V}}$, but in higher dimensions the notational system of matrix and tensor calculus would fail. Therefore, an important part of array algebra consists of the symbols and grammatical rules for expressing the multilinear operations.

The reward of using array algebra is related to the significant computational and storage space savings. The number of scalar arithmetical operations of a non-sparse array solution is proportional to the first power of the number, $N$, of the parameters in contrast to the third

* Presented at the ASP/ACSM Annual Convention, 19-23 March 1979, Washington, DC.



PHOTOGRAMMETRIC ENGINEERING AND REMOTE SENSING,
Vol. 46, No. 2, February 1980, pp. 177-192.

power of the conventional linear algebra. The storage space requirement for a solution of the non-sparse array equation is $N$ locations in contrast to $N^2$ locations of the conventional case.

Array algebra can be characterized as a generalized field of the so-called fast transform technology that caused a rethinking of several sciences in the 60's and 70's. There is a vast number of problems, technologies, and sciences where array algebra principles can be applied, either directly or after a rethinking. Because of the generality of array algebra, the use of the conventional fast transform technology would fail in many of these array algebra applications.

Not all multidimensional problems can be directly expressed by array algebra, not even after some modifications. The observed values (real or fictitious) have to form an array or a complete grid. Also, the math model has to have separable variables or design matrices. This requirement is identical to the technique of successive one-dimensional modeling, one variable direction at a time. Often these theoretical restrictions of array algebra can be released by a smart problem designer such that many real world problems can be modified and solved in the new approximated form with sufficient accuracy for practical purposes—and who would care about an exact solution of millions of parameters if it is not computationally feasible.

### AN ILLUSTRATIVE EXAMPLE

Assume some function values, say, temperatures

$$\underset{2,2,2}{\boldsymbol{L}_0} = \left\{ (l_0)_{j_1 j_2 j_3} \right\}^{j_1 = 1,2}_{j_2 = 1,2}_{j_3 = 1,2} \tag{1}$$

be measured at the corners of a rectangular room with sides $a$, $b$, $c$. The problem is defined to interpolate these values into the corners of a smaller concentric room of sides $sa$, $sb$, $sc$, where $s$ is a scale factor such that $0 < s < 1$.

Because the array $\underset{2,2,2}{\boldsymbol{L}_0}$ contains only two measured values in each coordinate direction of the space variables $z$, $y$, $x$, the interpolation function has to be restricted now to the tri-linear trapezoidal interpolation. The functional model therefore contains the variables $[1, z]$, $[1, y]$, $[1, x]$ in each "dimensionwise" interpolation of values $\boldsymbol{L}_0$ located at the intersections of the coordinates $z = z_{01}, z_{02}$; $y = y_{01}, y_{02}$; $x = x_{01}, x_{02}$. The trapezoidal interpolation coefficients of any variable, $u$, can be derived by

$$\begin{aligned} \underset{1,2}{\boldsymbol{k}}(u) &= [1, u] \begin{bmatrix} 1 & u_{01} \\ 1 & u_{02} \end{bmatrix}^{-1} \\ &= \underset{1,2}{\boldsymbol{a}}(u) \; \underset{2,2}{\mathbf{A}_0^{-1}} . \end{aligned} \tag{2}$$

Now the function values $\underset{2,1}{\boldsymbol{L}_0}$ from locations $u = u_{01}, u_{02}$ can be interpolated into values $\underset{2,1}{\boldsymbol{L}}$ at locations $u = u_1, u_2$ by

$$\begin{aligned} \underset{2,1}{\boldsymbol{L}} &= \begin{bmatrix} 1 & u_1 \\ 1 & u_2 \end{bmatrix} \begin{bmatrix} 1 & u_{01} \\ 1 & u_{02} \end{bmatrix}^{-1} \boldsymbol{L}_0 \\ &= \underset{2,2}{\boldsymbol{A}} \; \underset{2,2}{\boldsymbol{A}_0^{-1}} \; \underset{2,1}{\boldsymbol{L}_0} \\ \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}_{2,1} &= \underset{2,2}{\boldsymbol{K}} \; \underset{2,1}{\boldsymbol{L}_0}, \; \boldsymbol{K} = \boldsymbol{A}\boldsymbol{A}_0^{-1}, \underset{2,1}{\boldsymbol{L}_0} = \begin{bmatrix} l_{01} \\ l_{02} \end{bmatrix} . \end{aligned} \tag{3}$$

In the present example the coordinate system is centered by choosing

$$\begin{aligned} u_{01} &= -d/2 \quad & u_1 &= -sd/2 \\ u_{02} &= +d/2 \quad & u_2 &= +sd/2 \end{aligned}$$

$$\underset{2,2}{\boldsymbol{K}} = \begin{bmatrix} 1 & -sd/2 \\ 1 & sd/2 \end{bmatrix} \begin{bmatrix} 1 & -d/2 \\ 1 & d/2 \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} t_1 & t_2 \\ t_2 & t_1 \end{bmatrix}$$

$$t_1 = (1 + s)/2$$
$$t_2 = (1 - s)/2,$$

where $d$ can take the place of any of $a, b, c$. Thus, each coordinate direction of the example happens to have the interpolator $\boldsymbol{K}_1 = \boldsymbol{K}_2 = \boldsymbol{K}_3 = \boldsymbol{K}$ yielding

$$l_1 = t_1 l_{01} + t_2 l_{02}$$
$$l_2 = t_2 l_{01} + t_1 l_{02}. \tag{4}$$

The following three steps will yield the interpolated values $\underset{2,2,2}{\boldsymbol{L}}$ at coordinates $z = -sa/2, sa/2$; $y = -sb/2; sb/2$, $x = -sc/2, sc/2$ from the measured values $\underset{2,2,2}{\boldsymbol{L}_0}$ at coordinates $z = -a/2, a/2$; $y = -b/2, b/2$; $x = -c/2, c/2$:

Step 1:

Interpolations are performed along all columns of $\underset{2,2,2}{\boldsymbol{L}_0}$ by the summation

$$m_{r_1 j_2 j_3} = \sum_{j_1=1}^{2} (k_1)_{r_1 j_1} (l_0)_{j_1 j_2 j_3}, \quad \begin{matrix} r_1 = 1,2 \\ j_2 = 1,2 \\ j_3 = 1,2 \end{matrix} \tag{5a}$$

$$\boldsymbol{K}_1 = \begin{bmatrix} 1 & -sa/2 \\ 1 & sa/2 \end{bmatrix} \quad \begin{bmatrix} 1 & -a/2 \\ 1 & a/2 \end{bmatrix}^{-1}$$

$$= \boldsymbol{A}_1 \boldsymbol{A}_{01}^{-1} = \boldsymbol{K} = \begin{bmatrix} t_1 & t_2 \\ t_2 & t_1 \end{bmatrix}$$

$$\begin{aligned} m_{111} &= t_1 (l_0)_{111} + t_2 (l_0)_{211} & m_{121} &= t_1 (l_0)_{121} + t_2 (l_0)_{221} \quad \text{front} \\ m_{211} &= t_2 (l_0)_{111} + t_1 (l_0)_{211} & m_{221} &= t_2 (l_0)_{121} + t_1 (l_0)_{221} \quad \text{wall} \\ m_{112} &= t_1 (l_0)_{112} + t_2 (l_0)_{212} & m_{122} &= t_1 (l_0)_{122} + t_2 (l_0)_{222} \quad \text{back} \\ m_{212} &= t_2 (l_0)_{112} + t_1 (l_0)_{212} & m_{222} &= t_2 (l_0)_{122} + t_1 (l_0)_{222}. \quad \text{wall} \end{aligned} \tag{5a}$$

Thus, array $\underset{2,2,2}{\boldsymbol{L}_0}$ is replaced by the new array $\underset{2,2,2}{\boldsymbol{M}}$, i.e., the same storage locations can be utilized for both arrays. In practice an auxiliary vector $\underset{n_1,1}{\boldsymbol{Y}}$ with $n_1$ elements is required for the intermediate storing $\underset{n_1,1}{\boldsymbol{Y}} = \underset{n_1 n_1}{\boldsymbol{K}_1} \underset{n_1,1}{\boldsymbol{L}_{0_{j_2 j_3}}}$ before replacing the entire column $\underset{n_1,1}{\boldsymbol{L}_{0_{j_2 j_3}}}$ by $\underset{n_1,1}{\boldsymbol{Y}}$. The number of scalar multiplications of this step consists of the $n_2 n_3$ repeated matrix by column multiplications $\left( \underset{n_1 n_1}{\boldsymbol{K}_1} \underset{n_1,1}{\boldsymbol{L}_0} \right)_{j_2 j_3}$ requiring $n_1^2$ operations (scalar additions and multiplications) each or totally $n_2 n_3 n_1^2 = 16$ operations. Array $\boldsymbol{M}$ contains the interpolated values at the coordinate intersections of $z = -sa/2, sa/2$; $y = -b/2, b/2$; $x = -c/2, c/2$. Thus, $N = n_1 n_2 n_3$ values were interpolated using only $n_2 n_3 n_1^2$ operations or $n_1$ operations per point. Formation of the interpolation matrix $\underset{n_1 n_1}{\boldsymbol{K}_1} = \underset{n_1 n_1}{\boldsymbol{A}_1} \underset{n_1 n_1}{\boldsymbol{A}_{01}^{-1}}$ would require in the order of only $n_1^3$ operations which is an order of magnitude less than the above number $n_2 n_3 n_1^2$ of the scalar operations of the summation $\sum_{j_1=1}^{n_1} (k_1)_{r_1 j_1} (l_0)_{j_1 j_2 j_3}$.

Step 2:

Interpolations are now performed along the rows of array $\underset{2,2,2}{\boldsymbol{M}}$ to yield a new array $\underset{2,2,2}{\boldsymbol{N}}$ at the coordinate intersections $z = -sa/2, sa/2$; $y = -sb/2, sb/2$; $x = -c/2, c/2$ through the summation $n_{r_1 r_2 j_3} = \sum_{j_2=1}^{n_2} (k_2)_{r_2 j_2} m_{r_1 j_2 j_3}$ by

$$\begin{aligned} n_{111} &= t_1 m_{111} + t_2 m_{121} & n_{121} &= t_2 m_{111} + t_1 m_{121} \quad \text{front} \\ n_{211} &= t_1 m_{211} + t_2 m_{221} & n_{221} &= t_2 m_{211} + t_1 m_{221} \quad \text{wall} \\ n_{112} &= t_1 m_{112} + t_2 m_{122} & n_{122} &= t_2 m_{112} + t_1 m_{122} \quad \text{back} \\ n_{212} &= t_1 m_{212} + t_2 m_{222} & n_{222} &= t_2 m_{212} + t_1 m_{222}. \quad \text{wall} \end{aligned} \tag{5b}$$

Again the same "replacement" storage space can be utilized for both arrays $\boldsymbol{M}, \boldsymbol{N}$. The summation requires $n_1 n_3$ repeated row by matrix multiplications $\underset{1,n_2}{(\boldsymbol{M})}\ \underset{r_1 j_3}{\ }\ \underset{n_2 n_2}{\boldsymbol{K}_2^{\mathrm{T}}}$, $\begin{matrix} r_1 = 1,2,\cdots n_1 \\ j_3 = 1,2,\cdots n_3 \end{matrix}$ or

totally $n_1 n_3 n_2^2$ operations to yield $N = n_1 n_2 n_3$ new interpolated values. By denoting the "front walls" of $\underset{n_1 n_2 n_3}{\boldsymbol{L}_0}$, $\underset{n_1 n_2 n_3}{\boldsymbol{N}}$ with $\underset{n_1 n_2}{\boldsymbol{L}_{01}}$, $\underset{n_1 n_2}{\boldsymbol{N}_1}$ and the "back walls" with $\underset{n_1 n_2}{\boldsymbol{L}_{02}}$, $\underset{n_1 n_2}{\boldsymbol{N}_2}$ the steps 1-2 can be combined into the matrix expressions

$$\begin{aligned} \underset{m_1 m_2}{\boldsymbol{N}_1} &= \underset{m_1 n_1}{\boldsymbol{K}_1}\ \underset{n_1 n_2}{\boldsymbol{L}_{01}}\ \underset{n_2 m_2}{\boldsymbol{K}_2^{\mathrm{T}}} \qquad \begin{matrix} m_1 = n_1 = 2 \\ m_2 = n_2 = 2 \\ n_3 = 2 \end{matrix} \\ \underset{m_1 m_2}{\boldsymbol{N}_2} &= \underset{m_1 n_1}{\boldsymbol{K}_1}\ \underset{n_1 n_2}{\boldsymbol{L}_{02}}\ \underset{n_2 m_2}{\boldsymbol{K}_2^{\mathrm{T}}} . \end{aligned} \tag{5ab}$$

Step 3:

The one-dimensional interpolations performed now along the third ("depth row") dimension of array $\underset{n_1 n_2 n_3}{\boldsymbol{N}}$ will yield the final desired array $\boldsymbol{L}$ at locations $z = -sa/2, sa/2$; $y = -sb/2, sb/2$; $x = -sc/2, sc/2$ by the summation

$$l_{r_1 r_2 r_3} = \sum_{j_3=1}^{n_3} (k_3)_{r_3 j_3}\, n_{r_1 r_2 j_3} \tag{5c}$$

$$= \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \sum_{j_3=1}^{n_3} (k_1)_{r_1 j_1}\, (k_2)_{r_2 j_2}\, (k_3)_{r_3 j_3}\, (l_0)_{j_1 j_2 j_3}$$

$$\begin{aligned} l_{111} &= t_1 n_{111} + t_2 n_{112} & l_{121} &= t_1 n_{121} + t_2 n_{122} & &\text{new} \\ l_{211} &= t_1 n_{211} + t_2 n_{212} & l_{221} &= t_1 n_{221} + t_2 n_{222} & &\text{front} \\ & & & & &\text{wall} \\ l_{112} &= t_2 n_{111} + t_1 n_{112} & l_{122} &= t_2 n_{121} + t_1 n_{122} & &\text{new} \\ l_{212} &= t_2 n_{211} + t_1 n_{212} & l_{222} &= t_2 n_{221} + t_1 n_{222} . & &\text{back} \\ & & & & &\text{wall} \end{aligned} \tag{5c}$$

Again the same storage locations can be utilized and this final step requires $n_1 n_2 n_3^2 = n_3 N$ operations.

NOTATIONAL SYSTEM

The fundamental notational convention of array calculus expresses the summation

$$\sum_{j_k=1}^{n_k} a_{r_k j_k}\, x_{j_1 j_2 \cdots j_k \cdots j_i} \qquad \begin{matrix} j_1 = 1,2,3,\cdots n_1 \\ j_2 = 1,2,3,\cdots n_2 \\ \vdots \\ r_k = 1,2,3,\cdots m_k \\ \vdots \\ j_i = 1,2,3,\cdots n_i \end{matrix} \tag{6a}$$

by the so-called R-matrix or array multiplication

$$\underset{m_k n_k}{\boldsymbol{A}^k}\ \underset{n_1 n_2 \cdots n_k \cdots n_i}{\boldsymbol{X}} = \underset{n_1 n_2 \cdots m_k \cdots n_i}{\boldsymbol{L}} \tag{6b}$$

analogous to the notational system of matrix calculus. The superscript of matrix $\boldsymbol{A}$ now indicates whether $\boldsymbol{A}$ is a left, right, "back", etc., side matrix, i.e., it identifies the subscript $j_k$ of array $\boldsymbol{X}$ and the column index of $\boldsymbol{A}$ in regard to which subscript the summation is to be performed. Thus, for example, the matrix multiplications of Equation 5ab can be combined in the short expression

$$\underset{m_1 m_2 n_3}{\boldsymbol{N}} = \underset{m_1 n_1}{\boldsymbol{K}_1^1}\ \underset{m_2 n_2}{\boldsymbol{K}_2^2}\ \underset{n_1 n_2 n_3}{\boldsymbol{L}_0} \tag{5ab}'$$

$$= \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} (k_1)_{r_1 j_1}\, (k_2)_{r_2 j_2}\, (l_0)_{j_1 j_2 j_3} .$$

The final array $\underset{m_1 m_2 m_3}{\boldsymbol{L}} = \boldsymbol{K}_3^3 \boldsymbol{N}$ is expressed

$$\underset{m_1 m_2 m_3}{\boldsymbol{L}} = \underset{m_1 n_1}{\boldsymbol{K}_1^1}\ \underset{m_2 n_2}{\boldsymbol{K}_2^2}\ \underset{m_3 n_3}{\boldsymbol{K}_3^3}\ \underset{n_1 n_2 n_3}{\boldsymbol{L}_0}$$

$$= \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \sum_{j_3=1}^{n_3} (k_1)_{r_1 j_1} (k_2)_{r_2 j_2} (k_3)_{r_3 j_3} (l_0)_{j_1 j_2 j_3}, \tag{5abc}$$

where usually $m_1 \neq n_1, m_2 \neq n_2, m_3 \neq n_3$. In the above example $m_1 = n_1, m_2 = n_2, m_3 = n_3$ and the total number of operations for performing the triple summation (Equation 5 abc) becomes in this special case

$$\begin{aligned} op &= n_2 n_3 n_1^2 + n_1 n_3 n_2^2 + n_1 n_2 n_3^2 \\ &= (n_1 + n_2 + n_3)\, N \\ N &= n_1 n_2 n_3. \end{aligned} \tag{7}$$

In the additional special case of $n_1 = n_2 = n_3 = n$, $N = n^3$

$$\begin{aligned} op &= 3\, n\, N \\ &= \log_n N\ n\ N. \end{aligned} \tag{8}$$

It will now be shown that the above interpolations implicitely contain a rigorous linear solution of $N$ modeling parameters $\underset{n_1 n_2 n_3}{\boldsymbol{X}}$ : In the one-dimensional case $\underset{m,1}{\boldsymbol{L}} = \underset{mn}{\boldsymbol{A}}\ \underset{nn}{\boldsymbol{A}_0^{-1}}\ \underset{n,1}{\boldsymbol{L}_0}$ the multiplication $\boldsymbol{A}_0^{-1} \boldsymbol{L}_0$ solves for the "transform" coefficients $\boldsymbol{X}$, which then can be evaluated at points to be interpolated by $\underset{m,1}{\boldsymbol{L}} = \underset{m,n}{\boldsymbol{A}}\ \underset{n,1}{\boldsymbol{X}}$. Similarly the function

$$F(z,y,x) = \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \sum_{j_3=1}^{n_3} z^{j_1-1}\, y^{j_2-1}\, x^{j_3-1}\ \ (X)_{j_1 j_2 j_3} \tag{9}$$

can be fitted to values $\underset{n_1 n_2 n_3}{\boldsymbol{L}_0}$ by solving for

$$\underset{n_1 n_2 n_3}{\boldsymbol{X}} = (\underset{n_1 n_1}{\boldsymbol{A}_{01}^{-1}})^1\, (\underset{n_2 n_2}{\boldsymbol{A}_{02}^{-1}})^2\, (\underset{n_3 n_3}{\boldsymbol{A}_{03}^{-1}})^3\ \underset{n_1 n_2 n_3}{\boldsymbol{L}_0}\ . \tag{10}$$

Compared to the tri-linear interpolation of Equation 5 abc, only the matrices $\boldsymbol{K}_1 = \boldsymbol{A}_1 \boldsymbol{A}_{01}^{-1}$, $\boldsymbol{K}_2 = \boldsymbol{A}_2 \boldsymbol{A}_{02}^{-1}, \boldsymbol{K}_3 = \boldsymbol{A}_3 \boldsymbol{A}_{03}^{-1}$ are replaced by the small inverses $\boldsymbol{A}_{01}^{-1}, \boldsymbol{A}_{02}^{-1}, \boldsymbol{A}_{03}^{-1}$. The inversions require approximately only $n_1^3 + n_2^3 + n_3^3$ operations, which can be neglected compared to the $(n_1 + n_2 + n_3) n_1 n_2 n_3$ operations of the R-matrix multiplications.

It can be shown that mathematically the array solution is identical to the conventional solution where $\boldsymbol{X}, \boldsymbol{L}_0$ are treated as long column vectors by stacking the columns of the arrays one on the other similarly to the internal treatment of arrays in a computer. Notice that for example

$$\boldsymbol{A}_{01}^{-1} \boldsymbol{L}_0 \boldsymbol{A}_{02}^{-1\mathrm{T}} = \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} (a_{01}^{-1})_{r_1 j_1} (a_{02}^{-1})_{r_2 j_2} (l_0)_{j_1 j_2} \tag{11a}$$

equals the "long-hand" expression

$$\left\{ (a_{02}^{-1})_{r_2 j_2}\ \underset{n_1 n_1}{\boldsymbol{A}_{01}^{-1}} \right\} \underset{N,1}{\boldsymbol{L}_0^{\mathrm{E}1,2}} \tag{11b}$$

$$\underset{n_1 n_2, 1}{\boldsymbol{L}_0^{\mathrm{E}1,2}} = \left[ (\underset{n_1,1}{\boldsymbol{L}_0})_1^{\mathrm{T}},\ (\underset{n_1,1}{\boldsymbol{L}_0})_2^{\mathrm{T}},\ \cdots\ (\underset{n_1,1}{\boldsymbol{L}_0})_{j_2}^{\mathrm{T}}\ \cdots\ (\underset{n_1,1}{\boldsymbol{L}_0})_{n_2}^{\mathrm{T}} \right]^{\mathrm{T}}.$$

The following section will detail a computer program for the array multiplications such that the reader can numerically verify the identity of the array solution with the conventional one.

COMPUTATIONAL SOLUTIONS

The consistent system

$$\underset{n_1 n_1}{\boldsymbol{A}_{01}^1}\ \underset{n_2 n_2}{\boldsymbol{A}_{02}^2}\ \underset{n_3 n_3}{\boldsymbol{A}_{03}^3}\ \underset{n_1 n_2 n_3}{\boldsymbol{X}} = \underset{n_1 n_2 n_3}{\boldsymbol{L}_0} \longleftrightarrow \underset{N,N}{\boldsymbol{A}_0}\ \underset{N,1}{\boldsymbol{X}^{\mathrm{E}1,2,3}} = \underset{N,1}{\boldsymbol{L}_0^{\mathrm{E}1,2,3}} \tag{12}$$

is to be solved by utilizing the special structure of matrix $\underset{N,N}{\boldsymbol{A}_0}$ in the fashion of array calculus. The long columns of the extracted and rearranged columns of arrays $\underset{n_1n_2n_3}{\boldsymbol{X}}$ , $\underset{n_1n_2n_3}{\boldsymbol{L}_0}$ are denoted $\underset{N,1}{\boldsymbol{X}^{E1,2,3}}$ , $\underset{N,1}{\boldsymbol{L}_0^{E1,2,3}}$ . According to the summation

$$(l_0)_{r_1r_2r_3} = \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \sum_{j_3=1}^{n_3} (a_{01})_{r_1j_1} (a_{02})_{r_2j_2} (a_{03})_{r_3j_3} (X)_{j_1j_2j_3} \tag{13}$$

the large "conventional" design matrix $\underset{N,N}{\boldsymbol{A}_0}$ has the special structure

$$\underset{N,N}{\boldsymbol{A}_0} = \left\{ (a_{01})_{r_1j_1} (a_{02})_{r_2j_2} (a_{03})_{r_3j_3} \right\} \tag{14}$$

$$= \underset{n_1n_1}{\boldsymbol{A}_{01}} \otimes \underset{n_2n_2}{\boldsymbol{A}_{02}} \otimes \underset{n_3n_3}{\boldsymbol{A}_{03}} .$$

The symbol $\otimes$ denotes a tensor or Kronecker product having the property

$$\underset{N,N}{\boldsymbol{A}_0^{-1}} = \underset{n_1n_1}{\boldsymbol{A}_{01}^{-1}} \otimes \underset{n_2n_2}{\boldsymbol{A}_{02}^{-1}} \otimes \underset{n_3n_3}{\boldsymbol{A}_{03}^{-1}} . \tag{15}$$

Thus the inversion of $\underset{N,N}{\boldsymbol{A}_0^{-1}}$ is replaced by inversions of three small matrices requiring only $n_1^3 + n_2^3 + n_3^3$ operations in contrast to $N^3 = n_1^3 n_2^3 n_3^3$ operations of the conventional matrix inversion. Construction of $\boldsymbol{A}_0^{-1}$ from the small inverses and the subsequent matrix multiplication $\underset{N,N}{\boldsymbol{A}_0^{-1}}$ $\underset{N,1}{\boldsymbol{L}_0^{E1,2,3}}$ would require $N^2$ operations, each. This "Kronecker solution" (Greville, 1961) thereby requires in the order of $N$ times less operations than the conventional case.

A FORTRAN program will now be outlined for performing consistent "replacement" R-matrix multiplications, i.e., the dimensions of the input and output arrays remain the same and the small left, right and "back-side" matrices $\underset{n_1n_1}{\boldsymbol{A}_{01}}$ , $\underset{n_2n_2}{\boldsymbol{A}_{02}}$ , $\underset{n_3n_3}{\boldsymbol{A}_{03}}$ are square. These matrices are to be coded as arrays A1, A2, A3. The algorithm can be used for array solutions or evaluations of solutions depending on the mathematical content of the matrices A1, A2, A3. In order to solve for

$$\underset{n_1n_2n_3}{\boldsymbol{X}} = (\underset{n_1n_1}{\boldsymbol{A}_{01}^{-1}})^1 (\underset{n_2n_2}{\boldsymbol{A}_{02}^{-1}})^2 (\underset{n_3n_3}{\boldsymbol{A}_{03}^{-1}})^3 \underset{n_1n_2n_3}{\boldsymbol{L}_0} \tag{16a}$$

the matrices A1, A2, A3 represent the inverses $\boldsymbol{A}_{01}^{-1}, \boldsymbol{A}_{02}^{-1}, \boldsymbol{A}_{03}^{-1}$. The dimensions $n_1, n_2, n_3$ will be coded as N1, N2, N3. The algorithm for performing the fast array solution

$$X_{j_1j_2j_3} = \sum_{r_1=1}^{n_1} \sum_{r_2=1}^{n_2} \sum_{r_3=1}^{n_3} (a_{01}^{-1})_{j_1r_1} (a_{02}^{-1})_{j_2r_2} (a_{03}^{-1})_{j_3r_3} (l_0)_{r_1r_2r_3} \tag{16b}$$

can be outlined as follows:

```
      DIMENSION XL (N1, N2, N3), A(NMAX, NMAX), Y(NMAX)
C     NMAX IS MAX(N1, N2, N3)
      READ N1, N2, N3, XL(I, J, K), I = 1, N1, J = 1, N2, K = 1, N3
C     XL IS INPUT ARRAY (LO WHEN SOLVING FOR X AND X WHEN
C     EVALUATING L)
      READ A (I, J), I = 1, N1, J = 1, N1
C     A CONTAINS MATRIX A1
      DO 1 I = 1, N2
      DO 1 J = 1, N3
    1 CALL RMULT(A,N1, XL(1, I, J), 1)
      READ A(I, J), I = 1, N2, J = 1, N2
C     A CONTAINS MATRIX A2
      DO 2 I = 1, N1
      DO 2 J = 1, N3
    2 CALL RMULT (A, N2, XL(I, 1, J), N1)
C
      READ A (I, J), I = 1, N3, J = 1, N3
```

```
C       A CONTAINS MATRIX A3
        N1N2 = N1 * N2
        DO 3 I = 1, N1
        DO 3 J = 1, N2
    3   CALL RMULT (A, N3, XL(I, J, 1), N1N2)
        WRITE XL(I, J, K), I = 1, N1, J = 1, N2, K = 1, N3
        STOP
        END
C
        SUBROUTINE RMULT (A, NK, X, INC)
        DIMENSION A(1), X(1), Y(1)
        K = 0
        DO 1 I = 1, NK
        K = K + 1
        S = 0.
        KX = -INC + 1
        KK = K - NK
        DO 2 J = 1, NK
        KX = KX + INC
        KK = KK + NK
    2   S = S + A(KK) * X(KX)
    1   Y (I) = S
        KKX = -INC + 1
        DO 3 I = 1, NK
        KKX = KKX + INC
    3   X(KKX) = Y(I)
        RETURN
        END
```

The storage space allocation is approximately $N = n_1 n_2 n_3$ elements and the number of arithmetical operations is $(n_1 + n_2 + n_3)N$. Thus, the array solution reduces the number of both arithmetical operations and storage elements from $N^2$ of the Kronecker solution to the magnitude $N$. It is concluded that already the non-sparse array equations can be solved extremely efficiently using array calculus. The same statement applies for the usage stage of the solution: In the present example the interpolations through the "transform domain" coefficients $\boldsymbol{X}$ are performed by applying the above program for A1 = $\mathbf{A}_1$, A2 = $\mathbf{A}_2$, A3 = $\mathbf{A}_3$ and inputting $\underset{n_1 n_2 n_3}{\boldsymbol{X}}$ as array XL. This array will then be replaced by a new array $\boldsymbol{L}$ expressed as

$$\underset{n_1 n_2 n_3}{\boldsymbol{L}} = \underset{n_1 n_1}{\boldsymbol{A}_1^1}\ \underset{n_2 n_2}{\boldsymbol{A}_2^2}\ \underset{n_3 n_3}{\boldsymbol{A}_3^3}\ \underset{n_1 n_2 n_3}{\boldsymbol{X}} \tag{17a}$$

$$= \boldsymbol{A}_1^1 \boldsymbol{A}_2^2 \boldsymbol{A}_3^3 (\boldsymbol{A}_{01}^{-1})^1 (\boldsymbol{A}_{02}^{-1})^2 (\boldsymbol{A}_{03}^{-1})^3 \boldsymbol{L}_0 \tag{17b}$$

$$= (\boldsymbol{A}_1 \boldsymbol{A}_{01}^{-1})^1 (\boldsymbol{A}_2 \boldsymbol{A}_{02}^{-1})^2 (\boldsymbol{A}_3 \boldsymbol{A}_{03}^{-1})^3 \boldsymbol{L}_0 \tag{17c}$$
$$= \boldsymbol{K}_1^1 \boldsymbol{K}_2^2 \boldsymbol{K}_3^3 \boldsymbol{L}_0.$$

According to the last formula, Equation 17c, the values $\boldsymbol{L}$ can be directly interpolated without the intermediate step of first solving for $\boldsymbol{X}$ by inputting XL = $\boldsymbol{L}_0$, A1 = $\boldsymbol{K}_1$, A2 = $\boldsymbol{K}_2$, A3 = $\boldsymbol{K}_3$. In practical applications with large values of $n_1$, $n_2$, $n_3$, the "interpolation matrices" $K_1$, $K_2$, $K_3$ exhibit sparse structures resulting in further significant savings both in the number of arithmetical operations and storage locations. The fastest of such consistent solution algorithms require $kN$ scalar additions, where $1 < k < 20$. The number of storage locations is $k\ n_{\min}$, where $n_{\min} = \min(n_1, n_2)$ of a two-dimensional array $\underset{n_1 n_2}{\boldsymbol{L}_0}$.

Significant further acceleration of the computation time is achieved by tailoring parallel processing into the R-matrix multiplications. Using parallel array processing the $n_2 n_3$ repeated matrix by vector multiplications in, for example,

$$\underset{m_1 n_1}{\boldsymbol{K}^1}\ \underset{n_1 n_2 n_3}{\boldsymbol{L}_0} = \left( \sum_{j_2=1}^{n_1} k_{r_1 j_1} (l_0)_{j_1} \right)_{j_2 j_3 \; j_2 = 1,2,3,\cdots n_2 \; j_3 = 1,2,3,\cdots n_3} \tag{18}$$

are computed simultaneously. Therefore a single multiplication $\underset{n_k n_k}{\boldsymbol{K}^k}\ \underset{n_1 n_2 \cdots n_k \cdots n_i}{\boldsymbol{L}_0}$ only requires $n_k^2$ parallel operations or totally $n_1^2 + n_2^2 + \cdots n_i^2$ parallel operations are needed for a consistent solution of $N = n_1 n_2 \cdots n_i$ parameters. The fast banded cases require $k(n_1 + n_2 + \cdots + n_i)$ parallel operations standing in a high contrast to the $n_1^3 n_2^3 \cdots n_k^3 \cdots n_i^3$ sequential operations of the conventional linear solution. Finally the computational solution can be speeded, say 100-fold, by performing the operations in a tailored hardware algorithm. Thus, for example, if $n_1 = n_2 = 1000$, $N = 10^6$ or one million parameters can be solved in $(n_1 + n_2)$ 10 $\mu$s $\simeq$ 0.02 seconds if one parallel operation requires $^{10}/_k$ $\mu$s. This time is less than the 1/30 sec. picture rate of a TV-system. By assuming that an image, constantly received by an eye, contains in the order of $10^6$ gray values it is tempting to compare the performance of the eye-brain system to the above outlined array algebra computer processing.

## Generality of Array Algebra

The above elementary introduction of array calculus only dealt with the consistent special case which is not completely new for the so called "fast transform" technology (Good, 1958; Cooley and Tukey, 1965; Rivard, 1977). The distinguishing feature of array algebra allows the expression and general solutions of the multi-linear equations

$$\underset{m_1 n_1}{\boldsymbol{A}_1^1}\ \underset{m_2 n_2}{\boldsymbol{A}_2^2} \cdots \underset{m_i n_i}{\boldsymbol{A}_i^i}\ \underset{n_1 n_2 \cdots n_i}{\boldsymbol{X}} = \underset{m_1 m_2 \cdots m_i}{\boldsymbol{L}} - \underset{m_1 m_2 \cdots m_i}{\boldsymbol{V}} . \tag{19}$$

In array algebra $n_1, n_2, \cdots, n_i, m_1, m_2, \cdots, m_i$ can be completely arbitrary numbers and there are no restrictions to the structure or ranks of matrices $\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_i$. The problem area is extended from the inflexible consistent Fourier, Haar, Hadamard, etc. transforms to the more typical problems of linear algebra where the problem maker has free hands in the design of parameters $\underset{n_1 n_2 \cdots n_i}{\boldsymbol{X}}$ and the functional model resulting in matrices $\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_i$. Further, the observed values $\underset{m_1 m_2 \cdots m_i}{\boldsymbol{L}}$ need not form an *evenly* distributed grid with unit a priori weights as in the conventional fast transforms.

The first practical applications of array algebra have yielded solutions which most often cannot be solved using the conventional transforms nor even the theoretical array algebra. However, certain "cheating" a la Gordian knot has opened ways for utilization of array algebra. Such modified solutions do not yield exact solutions to the original (often conventional) problem definition, but for many real world problems it sufficies to have the modified and "nearly conventional" solutions—and who cares about an exact rigorous solution if it cannot be computationally realized. Such applications as volumetric computations of photogrammetrically measured liquid natural gas carriers, array correlation and feature extraction, fast solutions for the fundamental problems in photogrammetry, physical and geometric geodesy, fast multidimensional finite elements solutions, digital terrain modeling, TV-tracking, and fast image processing only show a part of the diversified field of array algebra applications. Most of these new fast solutions are so general that the use of conventional fast transforms would fail.

## Array Algebra FFT

The conventional fast transforms are centered around the monolinear fast discrete Fourier transform, FFT, which can be characterized as a "reverse array calculus" and will be demonstrated next.

The very special structure of the Fourier transform matrix

$$\underset{N,N}{\boldsymbol{A}^{-1}} = \frac{1}{N} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots \\ 1 & w & w^2 & w^3 & \cdots \\ 1 & w^2 & w^4 & w^6 & \cdots \\ 1 & w^3 & w^6 & w^9 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix} \qquad w = e^{i2\pi/N} \tag{20}$$

to yield the complex transform coefficients $\underset{N,1}{\boldsymbol{X}} = \underset{N,N}{\boldsymbol{A}^{-1}}\ \underset{N,1}{\boldsymbol{L}}$ allows some form of arraying of $\underset{N,1}{\boldsymbol{L}}$. One way of doing the arraying, (Rauhala, 1976), is to split $\underset{N,1}{\boldsymbol{X}}$ into the even term column $\underset{N/2,1}{\boldsymbol{X}}$ and odd term column $\underset{N/2,1}{\boldsymbol{X}_2}$, and to split $\underset{N,1}{\boldsymbol{L}}$ into $\underset{N/2,1}{\boldsymbol{L}_1}$, $\underset{N/2,1}{\boldsymbol{L}_2}$ by

$$\begin{aligned} \underset{N/2,1}{\boldsymbol{L}_1} &= [l_0, l_1, l_2 \cdots l_{N/2-1}]^{\mathrm{T}} \\ \underset{N/2,1}{\boldsymbol{L}_2} &= [l_{N/2}, l_{N/2+1}, l_{N/2+2}, \cdots l_{N-1}]^{\mathrm{T}}. \end{aligned} \tag{21}$$

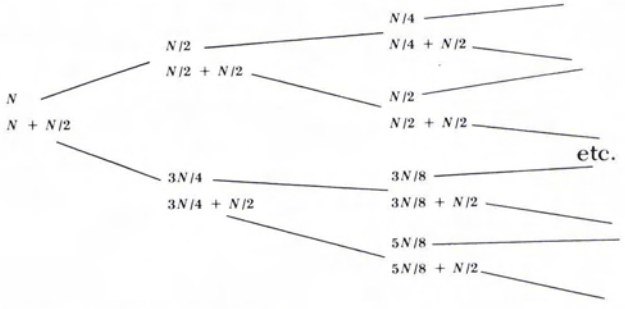
Here N is assumed to be a power of 2 for simplicity. Now the multiplication $\underset{N,N}{\boldsymbol{A}^{-1}}\ \underset{N,1}{\boldsymbol{L}}$ can be equivalently performed as

$$\begin{aligned} [\underset{N/2,2}{\boldsymbol{Y}_1}, \underset{N/2,2}{\boldsymbol{Y}_2}] &= \underset{N/2,2}{\boldsymbol{L}}\ \underset{2,2}{\boldsymbol{B}^{\mathrm{T}}} \qquad \underset{N/2,2}{\boldsymbol{L}} = [\boldsymbol{L}_1, \boldsymbol{L}_2] \\ \underset{N/2,1}{\boldsymbol{X}_1} &= \underset{N/2,\,N/2}{\boldsymbol{C}_1}\ \underset{N/2,1}{\boldsymbol{Y}_1} \\ \underset{N/2,1}{\boldsymbol{X}_2} &= \underset{N/2,\,N/2}{\boldsymbol{C}_2}\ \underset{N/2,1}{\boldsymbol{Y}_2} \end{aligned} \tag{22}$$

Similar splitting can be used for the multiplication $\boldsymbol{C}_1 \boldsymbol{Y}_1$, $\boldsymbol{C}_2 \boldsymbol{Y}_2$. Therefore, only the post-multiplications by the small matrices $\underset{2,2}{\boldsymbol{B}}$ are performed at each stage until at last of the $\log_2 N$ steps, only the premultiplications by matrices $\underset{2,2}{\boldsymbol{C}}$ need be performed to yield the final coefficients (usually in reversed binary ordering). Matrices $\underset{2,2}{\boldsymbol{B}}$ have the special structure

$$\boldsymbol{B} = \begin{bmatrix} 1 & w^k \\ 1 & -w^k \end{bmatrix} \tag{23}$$

and the rule of finding the power, $k$, follows the simple branching and halving pattern




(24)

The following simple complex numbers

$$\begin{aligned} w^N &= -w^{N/2} = 1 + 0i \\ w^{N/4} &= 0 + i \\ w^{N/8} &= \frac{1}{\sqrt{2}}(1 + i) \end{aligned} \tag{25}$$

occur in the first few splitting stages where the columns to be multiplied still are long. Therefore, it pays off to bypass (precompute) the complex multiplications for $w^N$, $w^{N/2}$, $w^{N/4}$ and to reduce the four scalar multiplications of a general complex multiplication into only two scalar multiplications (with factor $1/\sqrt{2}$) for $w^{N/8}$. Therefore, these savings become proportionally large for small values of $N$ (Rauhala, 1976, p. 80). These prederived algorithms can then be utilized for large values of $N$ by factorizing the one-dimensional transform into some bi-linear forms (Silverman, 1977). Table 1 is an example which, for $N = 16$, demonstrates the one-dimensional algorithm of array algebra FFT.

TABLE 1.

| | | | | |
|---|---|---|---|---|
| $l_0$ | $t_0 = l_0 + l_1$ | $m_0 = t_0 + t_1$ | $s_0 = m_0 + m_1$ | $X_0 = s_0 + s_1$ |
| $l_2$ | $t_2 = l_2 + l_3$ | $m_2 = t_2 + t_3$ | $s_1 = m_2 + m_3$ | $X_8 = s_0 - s_1$ |
| $l_4$ | $t_4 = l_4 + l_5$ | $m_1 = t_4 + t_5$ | $s_2 = m_0 - m_1$ | $X_4 = s_2 + is_3$ |
| $l_6$ | $t_6 = l_6 + l_7$ | $m_3 = t_6 + t_7$ | $s_3 = m_2 - m_3$ | $X_{12} = \bar{X}_4$ |
| $l_8$ | $t_1 = l_8 + l_9$ | $m_4 = t_0 - t_1$ | $s_4 = m_4 + im_5$ | $X_2 = s_4 + w^2 s_5$ |
| $l_{10}$ | $t_3 = l_{10} + l_{11}$ | $m_6 = t_2 - t_3$ | $s_5 = m_6 + im_7$ | $X_{10} = s_4 - w^2 s_5$ |
| $l_{12}$ | $t_5 = l_{12} + l_{13}$ | $m_5 = t_4 - t_5$ | $s_6 = \bar{s}_4$ | $X_6 = \bar{X}_{10}$ |
| $l_{14}$ | $t_7 = l_{14} + l_{15}$ | $m_7 = t_6 - t_7$ | $s_7 = \bar{s}_5$ | $X_{14} = \bar{X}_2$ |
| $l_1$ | $t_8 = l_0 - l_1$ | $m_8 = t_8 + it_9$ | $s_8 = m_8 + w^2 m_9$ | $X_1 = s_8 + ws_9$ |
| $l_3$ | $t_{10} = l_2 - l_3$ | $m_{10} = t_{10} + it_{11}$ | $s_9 = m_{10} + w^2 m_{11}$ | $X_9 = s_8 - ws_9$ |
| $l_5$ | $t_{12} = l_4 - l_5$ | $m_9 = t_{12} + it_{13}$ | $s_{10} = m_8 - w^2 m_9$ | $X_5 = s_{10} + w^5 s_{11}$ |
| $l_7$ | $t_{14} = l_6 - l_7$ | $m_{11} = t_{14} + it_{15}$ | $s_{11} = m_{10} - w^2 m_{11}$ | $X_{13} = s_{10} - w^5 s_{11}$ |
| $l_9$ | $t_9 = l_8 - l_9$ | $m_{12} = \bar{m}_8$ | $s_{12} = \bar{s}_{10}$ | $X_3 = \bar{X}_{13}$ |
| $l_{11}$ | $t_{11} = l_{10} - l_{11}$ | $m_{14} = \bar{m}_9$ | $s_{13} = \bar{s}_{11}$ | $X_{11} = \bar{X}_5$ |
| $l_{13}$ | $t_{13} = l_{12} - l_{13}$ | $m_{13} = \bar{m}_{10}$ | $s_{14} = \bar{s}_8$ | $X_7 = \bar{X}_9$ |
| $l_{15}$ | $t_{15} = l_{14} - l_{15}$ | $m_{15} = \bar{m}_{11}$ | $s_{15} = \bar{s}_9$ | $X_{15} = \bar{X}_1$ |
| | | | | total |
| mult | 0 | 0 | 4 | 10 14 |
| add | 16 | 8 | 16 | 20 60 |

In the example the complex multiplication with $w^{N/8} = w^2$ is counted to require two scalar multiplications and additions. The total number ntot of scalar multiplications, required for typical small $N$ FFT's, are shown in Table 2.

TABLE 2.

| $N$ | 8 | 16 | 32 | 64 | 128 |
|---|---|---|---|---|---|
| ntot | 2 | 14 | 54 | 163 | 454 |

The multidimensional array FFT can be performed in analogy to the computer program of Section 1.3. The subroutine RMULT and its calling statements have to be replaced to perform the above type of prederived one dimensional FFT's.

### GENERAL MONOLINEAR OPERATIONS

The FFT is restricted to evenly distributed and homogeneous observed values $\boldsymbol{L}$ in $\boldsymbol{X} = \boldsymbol{A}^{-1}\boldsymbol{L}$ and usually to the factorization $N = 2^i$. Array algebra is based on the general monolinear estimation theory which is then successively used in higher dimensions through the computational rules of array calculus. Thus, the rigorous mathematical concept "algebra" separates array algebra apart from the purely computational and grammatical rules of array calculus and the conventional fast transforms.

Array algebra is essentially linked to the general concepts (unbiasness, minimum variances) of mathematical statistics. Therefore, the monolinear starting point of array algebra is centered in solving the inconsistent system

$$\underset{mn}{\boldsymbol{A}}\ \underset{n,1}{\boldsymbol{X}} \neq \underset{m,1}{\boldsymbol{L}}\ ,\ \text{rank}\ (\underset{mn}{\boldsymbol{A}}) \leq n \tag{26a}$$

under the Gauss-Markov model

$$\mathrm{E}(\boldsymbol{L}) = \boldsymbol{A}\,\boldsymbol{X}, \tag{26b}$$

where E denotes the expectation operator. The classical linear algebra, started in the field of the adjustment calculus of mathematical surveying sciences, developed recipies for the full rank least squares solution

$$\hat{\boldsymbol{X}} = \boldsymbol{A}^{l}\boldsymbol{L} \longleftrightarrow \| \boldsymbol{L} - \boldsymbol{A}\boldsymbol{X} \|^2 = \min.$$
$$\boldsymbol{A}^{l} = (\underset{n\,n}{\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}})^{-1}\ \underset{nm}{\boldsymbol{A}^{\mathrm{T}}}, \text{ rank } (\underset{mn}{\boldsymbol{A}}) = n. \tag{27}$$

The following discussion will outline some generalized linear operators for solving the above system. For a more detailed presentation the reader is referred to Rauhala (1974, 1976, 1978b).

The theory of generalized matrix inverses (Rao and Mitra, 1971) extended the least-squares solution to

$$\hat{\boldsymbol{X}} = \boldsymbol{G}\boldsymbol{L} + (\boldsymbol{I} - \boldsymbol{G}\boldsymbol{A})\boldsymbol{U}, \tag{28}$$

where $\boldsymbol{U}$ can be arbitrary. The general least-squares operator $\boldsymbol{G}$ fulfills the only condition

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\,\boldsymbol{G} = \boldsymbol{A}^{\mathrm{T}} \tag{29a}$$

which can be converted into the two conditions

$$\left.\begin{aligned} \boldsymbol{A}\,\boldsymbol{G}\,\boldsymbol{A} = \boldsymbol{A} \quad &\longleftrightarrow \boldsymbol{G} \in \boldsymbol{A}^{g} \\ (\boldsymbol{A}\,\boldsymbol{G})^{\mathrm{T}} = \boldsymbol{A}\,\boldsymbol{G} &\longleftrightarrow \boldsymbol{G} \in \boldsymbol{A}_{l} \end{aligned}\right\} \boldsymbol{G} \in \boldsymbol{A}_{l}^{g}. \tag{29b}$$

The explicit expression of $\boldsymbol{A}_{l}^{g}$ is (Rauhala, 1976, p. 93),

$$\begin{aligned} \boldsymbol{A}_{l}^{g} &= \boldsymbol{A}_{lr}^{g} + (\boldsymbol{I} - \boldsymbol{A}_{lr}^{g}\boldsymbol{A})\,\boldsymbol{U}_2 \\ \boldsymbol{A}_{lr}^{g} &= (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{g}\,\boldsymbol{A}^{\mathrm{T}}, \end{aligned} \tag{29c}$$

where $\boldsymbol{U}_2$ can be arbitrary. The subscript "$r$" denotes the reflexivity property that if $\boldsymbol{G}\,\boldsymbol{A}\,\boldsymbol{G} = \boldsymbol{G}$ then $\boldsymbol{G} \in \boldsymbol{A}_r$.

In analogy to the theory of least squares the norm $\| \hat{\boldsymbol{X}} \|^2$ can be minimized yielding for the inconsistent system (Rauhala, 1976, p. 96),

$$\hat{\boldsymbol{X}} = \boldsymbol{A}_{mr}\boldsymbol{L} \longleftrightarrow \begin{aligned} &\boldsymbol{A}_{mr}\,\boldsymbol{A}\,\boldsymbol{A}_{mr} = \boldsymbol{A}_{mr} \\ &(\boldsymbol{A}_{mr}\,\boldsymbol{A})^{\mathrm{T}} = \boldsymbol{A}_{mr}\,\boldsymbol{A}. \end{aligned} \tag{30}$$

One of the main findings of the theory of generalized inverses is the realization that if $r(\underset{mn}{\boldsymbol{A}}) < n$, or precisely in the non-full rank cases of general inverses, the parameters $\boldsymbol{X}$ are not unbiasedly estimable (Rao and Mitra, 1971; Bossler, 1973; Grafarend and Schaffrin, 1974; Rauhala, 1974, 1975, 1976, 1978b). This fact has been overlooked and misinterpreted in one of the few surveying textbooks on this subject (Bjerhammar, 1973) as discussed in more detail in Rauhala (1976, 1978b, 1979).

For the case rank $(\underset{mn}{\boldsymbol{A}}) < n$ the minimization of the bias yields the estimator (Rauhala, 1976, p. 100, 1978b, p. 45)

$$\hat{\boldsymbol{X}} = \boldsymbol{A}_{m}^{g}\boldsymbol{L} + \boldsymbol{U}^{\mathrm{T}}(\boldsymbol{I} - \boldsymbol{A}\,\boldsymbol{A}_{m}^{g})\,\boldsymbol{L}. \tag{31}$$

The general minimum variance biased estimator was found to be (Rauhala, 1976, p. 100)

$$\hat{\boldsymbol{X}} = \boldsymbol{A}_{lr}\boldsymbol{L}. \tag{32}$$

The left side loop inverses usually satisfy these two conditions (Rauhala, 1974), i.e., the general operator to yield minimum variances need not necessarily satisfy the g-inverse condition. The estimate

$$\begin{aligned} \hat{\boldsymbol{X}} &= \boldsymbol{A}_{lrm}^{q}\boldsymbol{L} \\ &= \boldsymbol{A} + \boldsymbol{L} \end{aligned} \tag{33}$$

has all of the above properties, i.e., least squares, minimum variance, minimum norm, minimum bias. For the full-rank case this solution yields the zero bias Gaussian least-squares solution. The remainder of this section will describe the estimation technique of loop inverses where the biased estimates are bypassed through a simple parameter transformation to unbiasedly estimable "problem parameters" $\underset{p,1}{\boldsymbol{L}_0} = \underset{pn}{\boldsymbol{A}_0}\ \underset{n,1}{\boldsymbol{X}}$.

The idea of array calculus and algebra started from an estimation technique called loop inverses. The idea is already reflected in the first example of this paper where the parameters

$\underset{n,1}{\boldsymbol{X}} = \underset{nn}{\boldsymbol{A}_0^{-1}}\ \underset{n,1}{\boldsymbol{L}_0}$ in $\underset{m,1}{\boldsymbol{L}} \neq \underset{mn}{\boldsymbol{A}}\ \underset{nn}{\boldsymbol{A}_0^{-1}}\ \underset{n,1}{\boldsymbol{L}_0}$ are exchanged into $\boldsymbol{L}_0$ to yield the linear system of equations in parameters $\boldsymbol{L}_0$ by $\underset{m,1}{\boldsymbol{L}} \neq \underset{mn}{\boldsymbol{K}}\ \underset{n,1}{\boldsymbol{L}_0}$, $\underset{mn}{\boldsymbol{K}} = \underset{mn}{\boldsymbol{A}}\ \underset{nn}{\boldsymbol{A}_0^{-1}}$. In typical modeling problems $\underset{m,1}{\boldsymbol{L}}$ can be interpreted as interpolations from the unknown fictitious observables $\underset{n,1}{\boldsymbol{L}_0} = \underset{nn}{\boldsymbol{A}_0}\ \underset{n,1}{\boldsymbol{X}}$. In higher dimensions it is intuitively simple to realize the grid structure requirement of $\boldsymbol{L}_0$, $\boldsymbol{L}$ through the "dimensionwise" one-dimensional interpolations and also the associated computational rules of array calculus can be intuitively derived and experimented. In the simplest monolinear case we choose $m > n$ and now parameters $\boldsymbol{L}_0$ can be solved from

$$\underset{mn}{\boldsymbol{K}}\ \underset{n,1}{\boldsymbol{L}_0} = \underset{m,1}{\boldsymbol{L}} - \underset{m,1}{\boldsymbol{V}}, \quad \underset{mn}{\boldsymbol{K}} = \underset{mn}{\boldsymbol{A}}\ \underset{nn}{\boldsymbol{A}_0^{-1}}, \tag{34a}$$

by

$$\underset{n,1}{\hat{\boldsymbol{L}}_0} = \underset{nm}{\boldsymbol{K}^t}\ \underset{m,1}{\boldsymbol{L}}. \tag{34b}$$

The least-squares estimate $\hat{\boldsymbol{L}}_0$ is the compacted and filtered representant of the interpolation function and called "elevation array" in digital terrain modeling. In the special cases of evenly distributed and homogeneous observations $\underset{m,1}{\boldsymbol{L}}$ the solution $\boldsymbol{K}^t\ \boldsymbol{L} = \underset{nm}{\boldsymbol{H}}\ \underset{m,1}{\boldsymbol{L}}$ boils down to the convolution integral of signal processing

$$\begin{aligned}(\hat{l}_0)_i &= \sum_{k=-b}^{b} h_k\, l_{i-k} \\ &= \text{IFT}\,(\text{FT}(h) * \text{FT}(L)).\end{aligned} \tag{35}$$

Here FT denotes the Fourier transform, IFT is its inverse transform and $*$ denotes the dot multiplications of the frequencies. Now the main portions of the "filter matrix" $\boldsymbol{H} = \boldsymbol{K}^t$ are circulant, i.e., the rows are identical with exception of some column shifts. The identical row coefficients conform the "impulse response" $h$ whose Fourier transform is called the "transfer function" of the system. It can be easily verified that the transform domain solution

$$\begin{aligned}\underbrace{\boldsymbol{A}\overbrace{\boldsymbol{A}_0^{-1}\boldsymbol{L}_0}^{\boldsymbol{X}}}_{\boldsymbol{K}} &= \boldsymbol{L} - \boldsymbol{V} \\ \hat{\boldsymbol{X}} &= \boldsymbol{A}^t\,\boldsymbol{L} \\ &= \boldsymbol{A}_0^{-1}\,\boldsymbol{K}^t\,\boldsymbol{L}\end{aligned}$$

yields exactly the least-squares estimate $\underset{n,1}{\hat{\boldsymbol{L}}_0}$ by $\underset{nn}{\boldsymbol{A}_0}\ \underset{n,1}{\hat{\boldsymbol{X}}}$, i.e.,

$$\underset{nm}{\boldsymbol{H}} = \boldsymbol{K}^t = (\boldsymbol{A}\,\boldsymbol{A}_0^{-1})^t = \underset{nn}{\boldsymbol{A}_0}\ \underset{nm}{\boldsymbol{A}^t}. \tag{37}$$

Published and non-published simulations of operators $\boldsymbol{K}, \boldsymbol{K}^t$ (Rauhala, 1972a, 1974, 1976, 1977, 1978a, 1978b; Rauhala and Gerig, 1976; Kratky, 1978) have opened new vistas for such typical signal processing problems as separable filter design, replacement of non-separable filters with separable ones, recursive array filters, fast interpolators, array Kalman filters, array algebra finite elements, integrators, derivators, correlators, detection and extraction of blunders, discontinuities, edges, features, patterns, etc. These new concepts can be combined with their relatives of the similar existing techniques speeding the applications of array algebra in several real world problems. The new data snooping philosophy of Rauhala (1977, p. 182) offers one example of these applications which often can be realized in real-time systems because of their computational speed and requirement for a fixed production pattern. A combined array calculus data snooping allows, for example, the "more than real-time" control of relative orientation of a photogrammetric plotter system while the rank of the design matrix is not yet full.

The above estimation technique of Rauhala (1972a) was generalized to singular systems and a whole family of new linear operators called loop inverses (Rauhala, 1974). For example, the replacement of $\underset{nn}{\boldsymbol{A}_0^{-1}}$ to yield the parameter transformation from the consistent relationship

$$\underset{p,1}{\boldsymbol{L}_0} = \underset{pn}{\boldsymbol{A}_0}\ \underset{n,1}{\boldsymbol{X}} \tag{38}$$

can be done through the operator

$$\underset{np}{\boldsymbol{A}_0^m} = \underset{np}{\boldsymbol{A}_0^{\mathrm{T}}}\ (\underset{pp}{\boldsymbol{A}_0\,\boldsymbol{A}_0^{\mathrm{T}}})^{-1}. \tag{39}$$

Now

$$\underset{mp}{\boldsymbol{K}} = \underset{mn}{\boldsymbol{A}}\ \underset{np}{\boldsymbol{A}_0^m}$$

$$\underset{p,1}{\hat{\boldsymbol{L}}_0} = \underset{pm}{\boldsymbol{K}^l}\ \underset{m,1}{\boldsymbol{L}} \tag{40}$$

$$\begin{aligned}\hat{\boldsymbol{X}} &= \boldsymbol{A}_0^m\,\hat{\boldsymbol{L}}_0 + (\boldsymbol{I} - \boldsymbol{A}_0^m\,\boldsymbol{A}_0)\boldsymbol{U}\\ &= \boldsymbol{A}^{lm}\,\boldsymbol{L} + (\boldsymbol{I} - \boldsymbol{A}^{lm}\,\boldsymbol{A})\boldsymbol{U},\end{aligned} \tag{41}$$

where $\underset{n,1}{\boldsymbol{U}}$ can be arbitrary and the $lm$-inverse exhibits a typical structure of loop inverses, namely,

$$\boldsymbol{A}^{lm} = \boldsymbol{A}_0^m\,(\boldsymbol{A}\,\boldsymbol{A}_0^m)^l. \tag{42}$$

If $p = \operatorname{rank}\ (\underset{mn}{\boldsymbol{A}})$ the lm-inverse creates the pseudo-inverse $\boldsymbol{A}^+$ as a special case but the operator

$$\begin{aligned}\boldsymbol{H} &= (\boldsymbol{A}\,\boldsymbol{A}_0^m)^l\\ &= \boldsymbol{A}_0\,\boldsymbol{A}^{lm}\end{aligned} \tag{43}$$

still yields the Gaussian least-squares solution for $\underset{p,1}{\hat{\boldsymbol{L}}_0} = \underset{pm}{\boldsymbol{H}}\ \underset{m,1}{\boldsymbol{L}} = \underset{pn}{\boldsymbol{A}_0}\ \underset{n,1}{\hat{\boldsymbol{X}}}$ .

It can be shown that the observables $\boldsymbol{L} = \boldsymbol{A}\,\boldsymbol{X}$ are always unbiasedly estimable (Rao and Mitra, 1971); therefore, $\underset{p,1}{\boldsymbol{L}_0}$ can be chosen as an independent subset of $\underset{m,1}{\boldsymbol{L}}$ such that $\underset{pn}{\boldsymbol{A}_0}$ forms a basis of the row space of $\boldsymbol{A} \ni \boldsymbol{A}_0$. Then, by a proper ordering of rows, matrix $\boldsymbol{A}$ partitions as

$$\underset{mn}{\boldsymbol{A}} = \begin{bmatrix}\underset{pn}{\boldsymbol{A}_0}\\ \underset{rn}{\boldsymbol{A}_E}\end{bmatrix},\ r = m - p. \tag{44}$$

The filter matrix now simplifies to

$$\boldsymbol{A}\,\boldsymbol{A}_0^m = \begin{bmatrix}\underset{pp}{\boldsymbol{I}}\\ \underset{rp}{\boldsymbol{K}_E}\end{bmatrix},\ \boldsymbol{K}_E = \underset{rn}{\boldsymbol{A}_E}\ \underset{np}{\boldsymbol{A}_0^m} \tag{45}$$

$$\underset{pm}{\boldsymbol{H}} = (\boldsymbol{A}\,\boldsymbol{A}_0^m)^l = (\boldsymbol{I} + \underset{pp}{\boldsymbol{K}_E^{\mathrm{T}}}\,\boldsymbol{K}_E^{\mathrm{T}})^{-1}\,[\underset{pm}{\boldsymbol{I}}, \boldsymbol{K}_E^{\mathrm{T}}].$$

The estimate $\underset{p,1}{\hat{\boldsymbol{L}}_0} = \underset{pm}{\boldsymbol{H}}\ \underset{m,1}{\boldsymbol{L}}$ usually "solves the problem", i.e., there is no need for computing the biased estimates, $p < n$,

$$\hat{\boldsymbol{X}} = \boldsymbol{A}_0^m\,\hat{\boldsymbol{L}}_0 + (\boldsymbol{I} - \boldsymbol{A}_0^m\,\boldsymbol{A}_0)\,\boldsymbol{U} \tag{41}$$

or this computation becomes simple.

The above estimation technique was designed to solve for bad-conditioned photogrammetric systems closely related to the problem of collocation (Moritz, 1972) and has been applied in self-calibrating block adjustments and free net adjustments (Rauhala, 1972a, 1974, 1975). Some features of the above discussed full-rank starting idea of loop inverses have been partially reproduced in Bjerhammar (1975), where the application in Wiener-Hopf related prediction was discussed. The general mono- and multilinear cases were presented in Rauhala (1974, pp. 112-126). The over-constrained case, $p < \operatorname{rank}\ (\underset{mn}{\boldsymbol{A}})$ , and multiloop in-

verses with or without the multilinear cases of array algebra have created unique linear operators previously not treated in the literature of linear algebra (as far as the author is aware).

ARRAY ALGEBRA

The general monolinear operators can be converted into the multilinear array operators through the use of array calculus. Depending on the generality of the monolinear operators, one can distinguish the following three categories for solving the array equation:

$$\underset{m_1 n_1}{\boldsymbol{A}_1^1}\ \underset{m_2 n_2}{\boldsymbol{A}_2^2} \cdot \cdot \cdot \underset{m_i n_i}{\boldsymbol{A}_i^i}\ \underset{n_1 n_2 \cdot \cdot \cdot n_i}{\boldsymbol{X}} = \underset{m_1 m_2 \cdot \cdot \cdot m_i}{\boldsymbol{L}} - \underset{m_1 m_2 \cdot \cdot \cdot m_i}{\boldsymbol{V}}. \tag{46}$$

*Conventional multilinear fast transforms.* The equation system is consistent such that

$$m_1 = n_1, m_2 = n_2, \cdot \cdot \cdot m_k = n_k, \cdot \cdot \cdot m_i = n_i \tag{47}$$

and usually $n_k$ is a power of 2. The full rank square matrices $\underset{n_k n_k}{\boldsymbol{A}_k}$, $k = 1, 2, \cdot \cdot \cdot i$, all have the very special structure of Equation 20 such that a single R-matrix multiplication $(\boldsymbol{A}_k^{-1})^k \boldsymbol{L}$ requires $n_1 n_2 \cdot \cdot \cdot n_{k-1} n_{k+1} \cdot \cdot \cdot n_i$ repeated one-dimensional conventional fast transforms.

*Gaussian array solution.* If all of the rectangular matrices $\underset{m_k n_k}{\boldsymbol{A}_k}$, $k = 1, 2, \cdot \cdot \cdot i$, have full ranks, the least-squares estimate $\hat{\boldsymbol{X}}$ becomes unique

$$\underset{n_1 n_2 \cdot \cdot \cdot n_i}{\hat{\boldsymbol{X}}} = \underset{n_1 m_1}{\boldsymbol{A}_1^{t1}}\ \underset{n_2 m_2}{\boldsymbol{A}_2^{t2}} \cdot \cdot \cdot \underset{n_i m_i}{\boldsymbol{A}^{ti}}\ \underset{m_1 m_2 \cdot \cdot \cdot m_i}{\boldsymbol{L}}. \tag{48}$$

Separable a priori weights $\boldsymbol{P}_k = (\sqrt{\boldsymbol{P}_k})^T \sqrt{\boldsymbol{P}_k}$ can be included in the solution by the premultiplications

$$\begin{aligned} \boldsymbol{A}_k &= \sqrt{\boldsymbol{P}_k}\, \boldsymbol{A}_k,\ k = 1, 2, 3, \cdot \cdot \cdot i \\ \boldsymbol{L} &= (\sqrt{\boldsymbol{P}_1})^1 (\sqrt{\boldsymbol{P}_2})^2 \cdot \cdot \cdot (\sqrt{\boldsymbol{P}_i})^i \boldsymbol{L}. \end{aligned} \tag{49}$$

*General array solution.* The existing knowledge of linear algebra can be included in the array solution

$$\begin{aligned} \underset{n_1 n_2 \cdot \cdot \cdot n_i}{\hat{\boldsymbol{X}}} = {} & \underset{n_1 m_1}{\boldsymbol{G}_1^1}\ \underset{n_2 m_2}{\boldsymbol{G}_2^2} \cdot \cdot \cdot \underset{n_i m_i}{\boldsymbol{G}_i^i}\ \underset{m_1 m_2 \cdot \cdot \cdot m_i}{\boldsymbol{L}} + \underset{n_1 n_2 \cdot \cdot \cdot n_i}{\boldsymbol{U}} \\ & - \underset{n_1 n_1}{(\boldsymbol{G}_1 \boldsymbol{A}_1)^1}\ \underset{n_2 n_2}{(\boldsymbol{G}_2 \boldsymbol{A}_2)^2} \cdot \cdot \cdot \underset{n_i n_i}{(\boldsymbol{G}_i \boldsymbol{A}_i)^i}\ \underset{n_1 n_2 \cdot \cdot \cdot n_i}{\boldsymbol{U}}. \end{aligned} \tag{50}$$

Any operator $\boldsymbol{G}_k$, $k = 1, 2, \cdot \cdot \cdot, i$, may represent any general operator of monolinear algebra. As shown in the theory of loop inverses (Rauhala, 1974, pp. 37-38), there exist no bounds to the generality of these operators. For example, a single third loop inverse like

$$\begin{aligned} \boldsymbol{A}^{mlm} &= \boldsymbol{A}_0^{lm} (\boldsymbol{A}\, \boldsymbol{A}_0^{lm})^m \\ &= \boldsymbol{A}_{00}^m (\boldsymbol{A}_0 \boldsymbol{A}_{00}^m)^l \left[\boldsymbol{A}\, \boldsymbol{A}_{00}^m (\boldsymbol{A}_0 \boldsymbol{A}_{00}^m)^l\right]^m \end{aligned} \tag{51}$$

represents an operator of a singular system with additional constraints. This expression boils down to Cayleyan matrix inversions because the $l$- and $m$-inverses only contain full rank matrices. Thus, the theory of loop inverses solves the problem of generalized matrix inverses without any computational use of the $g$-inverse $\boldsymbol{A}^g$, which has formed the starting point of the previous theories of general inverses.

The mathematical statistics of the general array solution should be developed, because the present concepts are more or less restricted to the monolinear case. On the other hand, the main multilinear applications favor the Gaussian least-squares array solution with a simple transition to the classical concepts (Rauhala, 1976, p. 111).

*Multi stage array solutions.* Some further generalized array equations deal with the array version of Kalman filtering or batch processing of array equations yielding the general case of constrained array equations (Rauhala, 1974, pp. 113-114; 1976, p. 79; 1977, p. 179). A two-dimensional special case of this problem is treated in Buchanan and Thomas (1968), and a further restricted special case for inclusion of one single constraint is discussed in Jancaitis and Magee (1977) and Snay (1978).

Recent application oriented research of array algebra has brought forth, in connection with a new correlation concept, a general system of array equations (Rauhala, 1977, p. 183). The

solutions of some special cases of array correlation may cause rethinking in the correlation technology.

## Conclusions

The first section, "Why Bother with Array Algebra," illustrated the elementary computational principles of array calculus, pointing out the physical relation of these "generalized fast transforms" to multilinear interpolations, matrix multiplications and inversions, solution of multilinear equations, and filtering. A simple FORTRAN program was devised for computing three-dimensional non-sparse array multiplications. Then the very fast sparse case, parallel processing, and array hardware were used to demonstrate the potential power of this technology.

In the next section, "Generality of Array Algebra," array algebra FFT was outlined by a successive arraying of the Fourier transform vectors. Then some general linear operators and loop inverses were introduced. The application of these general monolinear operators into array calculus created the general concept of array algebra. In conclusion, array algebra is indeed a generalized form of the conventional linear algebra and fast transforms, possessing great power and potential for use in many sciences, technologies, and specific problems.

## References


Bjerhammar, A., 1973. *Theory of Errors and Generalized Matrix Inverses*, Elsevier, Amsterdam.

———, 1975. Reflexive Prediction, *Vermessung, Photogrammetrie, Kulturtechnik*, 73: 3,4.

Blaha, G., 1977a. A Few Basic Principles and Techniques of Array Algebra. *Bull. Geod.*, 3rd issue.

———, 1977b. Least Squares Prediction and Filtering in Any Dimensions Using the Principles of Array Algebra. *Bull. Geod.*, 4th issue.

Bossler, J. D., 1973. A Note on the Meaning of Generalized Inverse Solutions in Geodesy, *JGR* 78:14.

Buchanan, J. E., and D. H. Thomas, 1968. On Least Squares Filtering of Two-Dimensional Data with a Special Structure, *Siam J. Numer. Anal.*, Vol. 5, No. 2.

Cooley, J., and J. Tukey, 1965. An Algorithm for Machine Computation of Complex Fourier Series, *Math. Comput.*, Vol. 19, pp. 297-301.

Good, I. J., 1958. The Interaction Algorithm and Practical Fourier Series, *J. Royal Stat. Soc.*, Ser. 13, 20, pp. 361-372.

Grafarend, E., and B. Schaffrin, 1974. Unbiased Free Net Adjustment, *Survey Review*, XXII, 171, pp. 200-218.

Greville, T. N. E., 1961. Note on Fitting of Functions of Several Independent Variables, *J. Soc. Indust. Appl. Math.*, Vol. 9, No. 1.

Jancaitis, J. R., and R. L. Magee, 1977. *Investigation of the Application of Array Algebra to Terrain Modeling*, USA-ETL, Ft. Belvoir, Va.

Kratky, V., 1976. Grid-Modified Polynomial Transformation of Satellite Imagery, *Remote Sensing Environ.*, 5, pp. 67-74.

———, 1978. DTM Interpolation with Gliding Vectors, *Proceedings of Digital Terrain Models (DTM) Symposium*, May 9-11, St. Louis, American Society of Photogrammetry, Falls Church, Va.

Moritz, H., 1972. *Advanced Least-Squares Methods*, Department of Geodetic Science, Report No. 175, The Ohio State University, Columbus, 1972.

Noble, B., 1966. *A Simple Method for the Solution of Badly-Conditioned Least-Squares Equations*, M R L Technical Summary Report #644, Mathematical Research Center, U.S. Army, The Univ. of Wisc.

Rao, C. R., S. K. Mitra, 1971. *Generalized Inverse of Matrices and Its Applications*, Wiley, New York.

Rauhala, U. A., 1972a. Calculus of Matrix Arrays and General Polynomial and Harmonic Interpolation by Least Squares With New Solutions in Photogrammetry and Geodesy, *Fotogrammetriska Meddelanden*, VI: 4, Department of Photogrammetry, Royal Institute of Technology, Stockholm.

———, 1972b. New Solutions For Fundamental Calibration and Triangulation Problems, *Svensk Lantmäteritidskrift*, No. 4, Stockholm.

———, 1974. Array Algebra With Applications in Photogrammetry and Geodesy, *Fotogrammetriska Meddelanden*, VI:6, Department of Photogrammetry, Royal Institute of Technology, Stockholm.

———, 1975. Calculation of Loop Inverses, *Fotogrammetriska Meddelanden*, 2:38, Department of Photogrammetry, Royal Institute of Technology, Stockholm.

———, 1976. A Review of Array Algebra, *Fotogrammetriska Meddelanden*, 2:38, Department of Photogrammetry, Royal Institute of Technology, Stockholm.

———, 1977. Array Algebra as General Base of Fast Transforms, *Proceedings of Symposium Image*

*Processing—Interactions with Photogrammetry and Remote Sensing*, held 3-5 October in Graz, Mitteilungen Der Geodätischen Institute Der Technischen Universität Graz, Folge 29, Graz.

_____, 1978a. Array Algebra DTM, *Proceeding of Digital Terrain Models (DTM) Symposium*, May 9-11, St. Louis, American Society of Photogrammetry, Falls Church, Virginia.

_____, 1978b. *Loop Inverses and Array Algebra as Geodetic Tools*, Paper presented at AGU Fall Meeting, San Francisco, CA., December 4-8, 1978.

_____, 1979. Intuitive Derivation of Loop Inverses and Array Algebra, *Bull. Geod.*, 4th issue (in press).

Rauhala, U. A., and S. Gerig, 1976. *Digital Terrain Data Compaction Using Array Algebra*, Report of DBA Systems, Inc., performed for USA-ETL, Fort Belvoir, Virginia, under contract No. DAAG53-76-C-0127.

Rivard, G. E., 1977. Direct Fast Fourier Transform of Bivariate Functions, *IEEE Trans. Acoust. Speech, Signal Processing*, Vol. ASSP-25, No. 3, pp. 250-252.

Silverman, H. F., 1977. An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA), *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-25, No. 2, pp. 152-165.

Snay, R. A., 1978. Applicability of Array Algebra, *Reviews of Geophysics and Space Physics*, Vol. 16, No. 3, pp. 459-464.

Woltring, H. J., 1976. *Measurement and Control of Human Movement*, Doctoral Thesis, Laboratory of Psychology, University of Nijmegen, Nijmegen, Netherlands.

(Received 1 February 1979; accepted 16 July 1979)

## Notice to Contributors

1. Manuscripts should be typed, double-spaced on 8½ × 11 or 8 × 10½ white bond, on *one* side only. References, footnotes, captions—everything should be double-spaced. Margins should be 1½ inches.
2. Ordinarily *two* copies of the manuscript and two sets of illustrations should be submitted where the second set of illustrations need not be prime quality; EXCEPT that *five* copies of papers on Remote Sensing and Photointerpretation are needed, all with prime quality illustrations to facilitate the review process.
3. Each article should include an abstract, which is a *digest* of the article. An abstract should be 100 to 150 words in length.
4. Tables should be designed to fit into a width no more than five inches.
5. Illustrations should not be more than twice the final size: *glossy* prints of photos should be submitted. Lettering should be neat, and designed for the reduction anticipated. Please include a separate list of captions.
6. Formulas should be expressed as simply as possible, keeping in mind the difficulties and limitations encountered in setting type.

## Journal Staff