SAS/IML™ Versus IBM®/APL as a Scientific Programming Language

S. D. Breytenbach, Department of Computer Services, University of SOUTH AFRICA
G. Erens, Department of Quantitative Management, University of SOUTH AFRICA

ABSTRACT

The relative merits of SAS/IML and IBM/APL as scientific programming languages are discussed and illustrated using a number of algorithms from current scientific literature. It is argued that SAS/IML is sufficiently powerful for most scientific programming problems. Moreover it is easier for both the SAS user and the uninitiated to acquire the necessary level of expertise than is the case with IBM/APL.

INTRODUCTION

This paper was written with a specific audience in mind, SUGI 13 being a user's conference.

Many users request IBM/APL. The authors, having used SAS/IML previously, launched an investigation into IBM/APL in order to evaluate the relative merits of IBM/APL and SAS/IML. The SAS/IML programming was done by the authors and the IBM/APL programming was done by Ms. Illza Nell, an IBM/APL consultant from INFO*PLUS.

SAS/IML AND IBM/APL AS SCIENTIFIC PROGRAMMING LANGUAGES

Both SAS/IML and IBM/APL are matrix oriented languages, geared for the use of people who think in terms of matrix algebra. In the end, the problem with writing either of these languages is not the programming language but the user's inability to think in terms of matrix algebra.

The applications for SAS/IML and IBM/APL are similar. The same scientific problems can be solved with both SAS/IML and IBM/APL.

IBM/APL	SAS/IML code
`A ← 1 2 r 1 2 3 4`	`A = {1 2, 3 4};`
`A ← 12 34 + 56 78`	`A = {12 34} + {56 78};`
`B ← A + C`	`B = A + C;`

Two problems from the field of operations research are used to illustrate the point. The problems are:

- the Predator-Prey differential Equations and
- an Acyclic Algorithm for the shortest path through a Network.

The Predator-Prey Differential Equations

The problem statement follows:

The populations of predators p(t) and prey q(t) at time *t* are related by the following set of coupled differential equations. The constants *k* and *a* determine the relative decay/growth of the populations.

$$\frac{dp}{dt} = -k_1 p + a_1 qp$$

$$\frac{dq}{dt} = -k_2 p + a_2 qp$$

The difference equations in matrix form are:

$$P(t + t) = [K + A \# P(t)]P(t) \qquad \ldots(1)$$

where

$$P(t) = \begin{bmatrix} p(t) \\ q(t) \end{bmatrix}$$

$$K = I(2) - \begin{bmatrix} k1 & 0 \\ 0 & -k2 \end{bmatrix} \# \delta t$$

$$A = \begin{bmatrix} 0 & a1 \\ -a2 & 0 \end{bmatrix} \# \delta t$$

and # is the Hadamard/Schur product

$$\text{ie} \quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} \# \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} ae & be \\ cf & df \end{bmatrix}$$

Program equation (1) as part of an iterative cycle.

After each iteration set P(t) = P(t + δt)

Execute the program for:

k1 = k2 = 1000 δt = 0.0001
Start with p(0) = 510 and q(0) = 990 and do 300 iterations, printing out the values of p(t) and q(t) at every tenth iteration.

The SAS/IML program

```
START;
* COEFFICIENTS K1 & K2;
KC = {1000 0,
      0 -1000};
* COEFFICIENTS A1 & A2;
AC = {0  1,
     -2 0};
* INITIAL VALUES OF P0 & Q0;
P0 = {510, 990};
* NUMBER OF ITERATIONS & STEPLENGTH;
NI = 300; DT = 0.0001;
K = I(2) - KC # DT;
A = AC # DT;
PT = P0;
*** PREDATOR PREY ITERATION ***;
DO I = I TO NI;
PD = (K + A # PT) * PT;
IF I + 10 * INT(I/10) THEN DO;
   S = (I//PD)';
   PRINT S; END;
END;
PT = PD;
END;
FINISH;
```

The IBM/APL program

```
⍝ THE PREDATOR - PREY PROGRAM
⍝ SET MATRICES AND CONSTANTS
⍝ COEFFICIENTS K1 AND K2
KC ← 2 2 ⍴ 1000 0 0 ¯1000
⍝ COEFFICIENTS A1 AND A2
AC ← 2 2 ⍴ 0 1 ¯2 0
⍝ INITIAL VALUES OF P0 AND Q0
P00 ← 2 1 ⍴ 510 990
P0 ← ⍉ 2 2 ⍴ P 00
⍝ NUMBER OF ITERATIONS AND STEPLENGTH
NI ← 10
DT ← 0.0001
I2 ← 2 2 ⍴ 1 0 0 1
K ← I2 - KC x DT
A ← AC x DT
PT ← P0
I ← 0
⍝ THE PREDATOR PREY ITERATION
ITER: → (NI < 1 ← I + 1) / 0
P00 ← (K + A x PT) +.x P00
→ (0 ≠ 10 ⍳ I) / INC
⍝ DISPLAY EVERY TENTH ITERATION
I,[1]P00
INC:PT ← ⍉ 2 2 ⍴ P 0 0
→ ITER
```

An Acyclic Algorithm for the Shortest Path through a Network

The problem statement follows:

The network is defined by the nodes, integers I and J (with $1 \leq I < J \leq N$ = dimension of network) which are connected and the distance DN(I,J) between the two nodes.

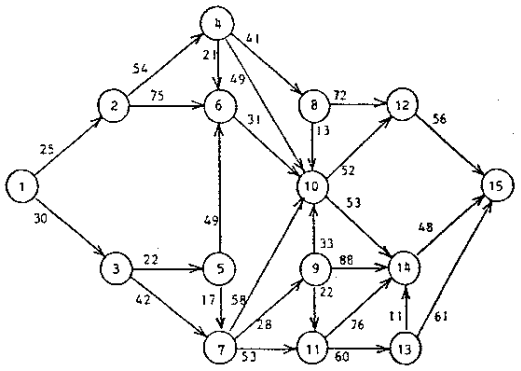
N = 'dimension' of network
DI = 'infinite distance'
Declare NxN matrix DN and vectors SD, HP, and NS of length N.
Set all elements of SD to DI.
Set the first elements of NS and DS to 0.

Read the network data (nodes that are connected and their connecting distance) into the matrix DN.

Iterate for J = 2 to N:
- HP = DN(,J) + SD
- SD(J) = Minimum element of HP
- NS(J) = Index of minimum element of HP

Print SD and NS

The Network



The SAS/IML program

```
START;
* SET DN TO DI, THE INFINITE DISTANCE;
DN = J(N,N,DI);
* MATRIX DR CONTAINS THE NETWORK DATA;
NR = NROW(DR);
DO K = 1 TO NR;
   I = DR(|K,1|); J = DR(|K,2|);
   DN(|I,J|) = DR(|K,3|);
END;
* SET SD TO DI;
SD = J(1,N,DI);
* SET NS TO 0;
NS = J(1,N,0);
* THE FIRST ELEMENT OF SD IS 0;
SD(|1,1|) = 0;
*** EXECUTE ACYCLIC ALGORITHM ***;
DO J = 2 TO N;
   HP = DN(|,J|)' + SD;
   SD(|1,J|) = MIN(HP);
   NS(|1,J|) = HP(|,>:<|);
END;
SDNS = (SD//NS);
PRINT SDNS;
FINISH;
```

The IBM/APL program

```
∇ NETWORK DR
⍝ DR CONTAINS THE NETWORK DATA
N ← ⌈ / DR [;2]
⍝ SET DN TO DI, THE INFINITE DISTANCE
DN ← (N x N) ⍴ DI ← 9999
⍝ LOAD MATRIX DN
DRVI ← ((DR[;1] -1) x N) + DR[;2]
DN[DRVI] ← DR[;3]
DN ← (N,N) ⍴ DN
⍝ SET SD TO DI
SD ← N ⍴ DI
⍝ SET NS TO 0
NS ← N ⍴ 0
⍝ THE FIRST ELEMENT OF SD IS 0
SD[1] ← 0
J ← 1
⍝ EXECUTE ACYCLIC ALGORITHM
LP: → (N < J ← J + 1) / OLP
HP ← DN[;J] + SD
SD[J] ← L / HP
NS[J] ← HP ⍳ L / HP
→ LP
OLP: SDNS ← ⍕ SD,[0.5]NS
⍝ PRINT SDNS
SDNS
```

CHOOSING BETWEEN SAS/IML AND IBM/APL

Potential users often need to be advised as to the suitability of either SAS/IML and IBM/APL. How to go about this seems to be a bit of a psychological problem. Either the positive or the negative aspects of SAS/IML and IBM/APL can be compared.

In this paper the negative aspects of each of the languages are emphasized as users most often want to know why they should not use a particular language.

Arguments against using IBM/APL

- In order to use IBM/APL to it's full capabilities, a special keyboard and a graphical video display unit is necessary.
- For a new user, IBM/APL is nearly incomprehensible, as the notation is strange and the special keyboard used, makes the user feel very inept. IBM/APL uses the Greek alphabet and some made-up characters.
- The IBM/APL symbols are IBM/APL's biggest problems. IBM/APL also works from right to left in a rather back to front fashion. Once one knows the notation there is little difficulty. It takes a lot of perserverance, however, to get that far. One can of course use mnemonics instead of the IBM/APL symbols but, according to the experts, IBM/APL then loses much of it's charm.
- IBM/APL has few preprogrammed functions. In IBM/APL one needs to know exactly what one is programming and what the functions do.
- IBM/APL, used in an unintelligent or uninformed way, has been known to make a large computer come to a standstill. Good quality control is therefore needed for IBM/APL programs.
- The very intricate coding and programming makes IBM/APL programs nearly impossible to maintain. IBM/APL is also difficult to debug unless one is an expert.
- We had to use SAS/GRAPH® to get a hard copy of the IBM/APL programs.

Arguments against SAS/IML

- Compared to the IBM/APL editors and output, SAS/IML is cumbersome.
- The way SAS/IML handles vectors and scalars is inelegant.
- Some functions available in IBM/APL may well be included in SAS/IML.
- The printing facility under SAS/IML needs to be improved.
- A compiler will greatly improve run-time. The SAS® system, and as a result also SAS/IML, doesn't have one as yet.

SIMILARITIES BETWEEN SAS/IML AND IBM/APL

- Both are interactive matrix languages and are extremely suitable for prototyping.
- Workspace handling is available in both the languages and routines can be stored in similar fashion in both languages.
- Work methods are similar.
- Graphics are available.
- The potential user-base is the same.

POTENTIAL IBM/APL / SAS/IML USERS

- A novice needs to think carefully before an attempt is made to learn IBM/APL - SAS/IML is easier to learn and write, being English-like, and has many preprogrammed functions included. The rest of the SAS system is of course available and easily accessable.
- A real IBM/APL fanatic should be left alone. IBM/APL is a very useful language and very efficient in the hands of an expert. One could possibly persuade such a person to start using SAS® and then suggest using SAS/IML as part of a SAS program. The similarities between SAS/IML and IBM/APL mean that an existing IBM/APL user will be able to become comfortable with SAS/IML quite easily as a lot of the concepts are similar or can easily be reproduced.

CONCLUSIONS

Reasons why SAS/IML is the preferred language

- A graphics terminal might not necessarily be readily available.
- SAS/IML, being English-like, is easier to assimilate than IBM/APL. The keyboard and notation that one needs to learn in order to use IBM/APL is daunting. Particularly for

someone who already knows some the SAS system, SAS/IML is the obvious choice.

- SAS/IML has ready-made functions available, e.g. the Hadamard/Schur product as used in the problems solved.
- IBM/APL is a matrix language, no more but also no less.
- The most important aspect of SAS/IML is that the rest of the SAS system is available to the user. As a result the user is not forced to program in matrix notation in SAS/IML, as is the situation in IBM/APL. The user can get out of SAS/IML as soon as the need for a matrix notation is satisfied and go and do whatever else needs to be done in a procedure.

Thanks to the following people who willingly gave their time and expertise:

Ms. I. Nell, INFO*PLUS, Johannesburg
Mr. Calitz, Department of Public Works, Pretoria
Drs. Schniedovitz and Bardau, CSIR, Pretoria

Names and address of authors:
Stephanie D. Breytenbach and Gordon Erens
University of South Africa,
P. O. Box 392 Pretoria
Republic of South Africa

Telex SA Code 9350068 (UNISA)
Fax: International +27+12+4293221
(automatic 24 hrs/day)
Telephone (012) 440-3111