

Do you want to do Matrix Calculations with Base SAS Software without SAS/IML? No Worries.

Sarfaraz Sayyed, Accenture Services Pvt. Ltd., Mumbai, INDIA
Binoy Varghese, Cybrid, Inc., Harrisburg, PA

ABSTRACT

Matrix calculations are required in various circumstances e.g. while solving a set of equations, for statistical analysis, etc. Matrix calculations can be done using Proc IML, however this procedure is not available in Base SAS as Proc IML is part of SAS/IML software.

This paper will discuss how to do various types of matrix calculations using Base SAS software.

INTRODUCTION

For some statistical computations, we require matrix calculations. Matrix calculations can be performed using procedures in Interactive Matrix Language (SAS/IML) software. However there are no built in procedures or macros within Base SAS which would enable us to perform matrix calculations.

This paper presents an in depth discussion about developing macros for matrix calculations.

The topics discussed include transposing a matrix, addition/subtraction of matrices, multiplication of two matrices, finding the trace, determinant and inverse of a matrix.

Requirements:

You need to have the matrix in the form of a SAS data set e.g.

Temp.sas7bdat =

A	B	C
1	2	3
4	5	6
7	8	9

= Matrix A

Temp_.sas7bdat =

D	E	F
11	12	13
14	15	16
17	18	19

= Matrix B

Temp_1.sas7bdat =

D	E	F
1	2	3
4	8	7
6	3	2

= Matrix C

Inv.sas7bdat =

A	B	C	D	E
1	2	4	5	1
4	5	6	6	2
7	8	9	8	3
3	7	9	5	4
5	4	6	8	9

Here we have 3 3x3 matrices and a 5x5 matrix in the form of a SAS dataset.

TRANSPOSING A MATRIX:

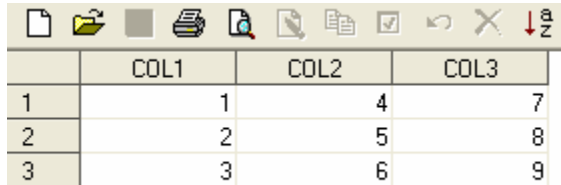
Let us start with the simplest one.

SAS CODE:

```
PROC TRANSPOSE DATA=temp OUT=temp1 (DROP=_name_) ;  
RUN;
```

```
PROC PRINT DATA=temp1 LABEL;  
RUN;
```

OUTPUT DATA SET TEMP1:



	COL1	COL2	COL3
1	1	4	7
2	2	5	8
3	3	6	9

ADDITION/SUBTRACTION OF THE MATRIX:

For this operation, consider matrix A (Temp.sas7bdat) and matrix B (Temp_.sas7bdat).

The idea used is adding/subtracting first column of first dataset by first column of second dataset, second column of first dataset by second column of second dataset and so on.

SAS CODE:

```
%MACRO addsub(libname1 = work  
              ,libname2 = work  
              ,in_data1 =  
              ,in_data2 =  
              ,out_data =  
              ,operator = %STR(+));  
  
  PROC SQL NOPRINT;  
    SELECT nobs INTO :obs1 FROM sashelp.vtable  
      WHERE libname="%upcase(&libname1.)" AND memname="%upcase(&in_data1.)";  
    SELECT nobs INTO :obs2 FROM sashelp.vtable  
      WHERE libname="%upcase(&libname2.)" AND memname="%upcase(&in_data2.)";  
  QUIT;  
  
  %IF &obs1. = &obs2. %THEN %DO;  
  
    %DO i = 1 %TO 2;  
      PROC SQL NOPRINT;  
        SELECT name INTO :mem&i. separated BY ' ' FROM sashelp.vcolumn  
          WHERE libname="%upcase(&libname&i.)" AND  
            memname="%upcase(&in_data&i.)";  
  
        SELECT count(distinct name) INTO :cntmem&i. FROM sashelp.vcolumn  
          WHERE libname="%upcase(&libname&i.)" AND  
            memname="%upcase(&in_data&i.)";  
      QUIT;
```

```

DATA _&i.;
  SET &&in_data&i.;
  ind=_n_;
  %DO j = 1 %TO &&cntmem&i.;
    IF %SCAN(&&mem&i., &j.)=. THEN %SCAN(&&mem&i., &j.)=0;
  %END;
RUN;
%END;

DATA &out_data.(KEEP=%DO k = 1 %TO &cntmem1.;
  col&k.
  %END;);
MERGE %DO l = 1 %TO 2;
  _&l.
  %END;;
BY ind;
%DO m = 1 %TO &cntmem1.;
  col&m.=0;
  col&m.=col&m. %DO n = 1 %TO 2;
    &operator. %SCAN(&&mem&n., &m.)
  %END;;
%END;
RUN;

PROC DATASETS LIBRARY=work MTYPE=data;
  DELETE _1 _2;
QUIT;

PROC PRINT DATA=&out_data. LABEL;
RUN;

%END;
%ELSE %DO;
  %PUT NOTE: The number of rows in the two matrices is not same, please check the
    dimensions and try again;
%END;

%MEND addsub;

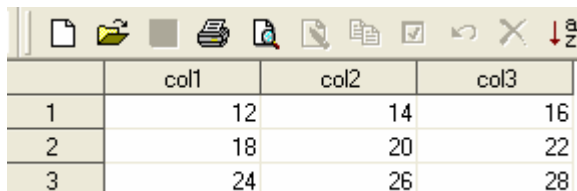
%addsub(in_data1 = temp
, in_data2 = temp_
, out_data = outd );

```

The case considered here is of addition i.e. Matrix A + Matrix B and the same will follow for subtraction when the macro variable 'operator' is assigned the value '%str(-)'.

Note: The default value of operator is set to addition. So, if the operator value is not passed while calling the macro, it will add the two matrices by default.

OUTPUT DATA SET OUTD :



	col1	col2	col3
1	12	14	16
2	18	20	22
3	24	26	28

FINDING THE DETERMINANT OF A MATRIX:

For this operation, take into consideration matrix C (Temp_1.sas7bdat).

SAS CODE:

```
%MACRO det(libname = work
            ,in_data =
            ,out_data = );

PROC SQL NOPRINT;
    SELECT nobs INTO :obs1 FROM sashelp.vtable
        WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";
    SELECT count (distinct name) INTO :cntcol FROM sashelp.vcolumn
        WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";
    SELECT name INTO :varnam separated BY ' ' FROM sashelp.vcolumn
        WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";
QUIT;

%IF &obs1. LE 1 OR &cntcol. LE 1 OR &obs1. NE &cntcol. %THEN %DO;
    %PUT NOTE: Check the dimensions of the matrix;
    %GOTO detend;
%END;

DATA _&in_data.;
    SET &in_data.;
    RENAME
    %DO i = 1 %TO &obs1.;
        %SCAN(&varnam., &i.)=col&i.
    %END;;
RUN;

PROC SQL NOPRINT;
    SELECT name INTO :varnam1 separated BY ' ' FROM sashelp.vcolumn
        WHERE libname="WORK" AND memname="%upcase(_&in_data.)";
QUIT;

DATA permute(keep=%DO i = 1 %TO &obs1.; r&i. %END;);
    ARRAY r [&obs1.] $3 (%DO i = 1 %TO &obs1.; "&i." %END;);
    n=dim(r);
    nfact=fact(n);
    DO i=1 TO nfact;
        CALL allperm(i, of r[*]);
        OUTPUT;
    END;
RUN;

PROC SORT DATA=permute;
    BY %DO i = 1 %TO &obs1.; r&i. %END;;
RUN;

DATA _null;
    SET permute end=eof;
    LENGTH temp $1000;

    sign1=(-1)**(_n_+1);
    %IF &obs1. > 2 %THEN %DO;

        %DO i = 2 %TO %EVAL(&obs1. - 1);
            m&i.=int(_n_/perm(&i.));
```

```

        IF mod(_n_,perm(&i.)) = 0 THEN m&i. = m&i. - 1;
        IF mod(&i.,2)=0 THEN DO;
            m&i._temp=int(_n_/perm(&i.+1));
            IF mod(_n_,perm(&i.+1)) = 0 THEN m&i._temp = m&i._temp - 1;
            m&i.=m&i. + m&i._temp;
        END;
        ELSE DO;
            m&i._temp=0;
        END;

        sign&i. = (-1)**(m&i.);
    %END;
%END;
sign = %DO i = 1 %TO %EVAL(&obs1.-1);
    sign&i. *
%END; 1;

%DO i = 1 %TO &obs1.;
    r&i.=compress("x&i."||r&i.);
%END;
temp=compress(put(sign,best.)||'*'||r1 %DO i = 2 %TO &obs1.;
    ||"*"||r&i.
    %END;);
CALL symput('_'||compress(put(_n_, best.)),temp);
RUN;

DATA split(KEEP= %DO i = 1 %TO &obs1.;
    %DO j = 1 %TO &obs1.;
        x&i.&j.
    %END;
    %END;);
SET _&in_data. end=eof;
%DO i = 1 %TO &obs1.;
    %DO j = 1 %TO &obs1.;
        RETAIN x&i.&j. 0;
        IF _n_ = &i. THEN x&i.&j.=col&j.;
    %END;
%END;
IF eof;
RUN;

PROC SQL NOPRINT;
    CREATE TABLE split1 AS SELECT * FROM _null, split;
QUIT;

DATA &out_data.(keep=det);
    SET split1 end=eof;
    RETAIN det 0;
    %DO i = 1 %TO %SYSFUNC(perm(&obs1.));
        IF _n_ = &i. THEN deter=&&&i.;
    %END;
    det=det + deter;
    IF eof;
RUN;

PROC PRINT DATA=&out_data. LABEL;
    TITLE;
RUN;

PROC DATASETS LIBRARY=work MTYPE=data;
    DELETE permute _&in_data. split: _null;
QUIT;

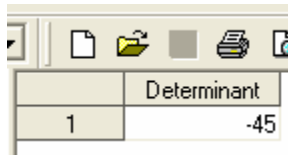
```

```
%detend:
```

```
%MEND det;
```

```
%det(libname = work  
      ,in_data = temp_1  
      ,out_data = temp2 );
```

OUTPUT DATA SET TEMP2:



	Determinant
1	-45

FINDING THE TRACE OF A MATRIX:

For this operation, we will use the matrix A (Temp.sas7bdat).

Here, the idea used is to take all the diagonal elements in a new column and add them, since trace is the sum of the diagonal elements.

SAS CODE :

```
%MACRO trace(libname = work  
              ,in_data =  
              ,out_data = );  
  
  PROC SQL NOPRINT;  
    SELECT count(distinct name) INTO :obs1 FROM sashelp.vcolumn  
      WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";  
    SELECT nobs INTO :numobs FROM sashelp.vtable  
      WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";  
    SELECT name INTO :varnam separated BY ' ' FROM sashelp.vcolumn  
      WHERE libname="%upcase(&libname.)" AND memname="%upcase(&in_data.)";  
  QUIT;  
  
  %IF &obs1. LE 1 OR &numobs. LE 1 OR &obs1. NE &numobs. %THEN %DO;  
    %PUT NOTE: Check the dimensions of the matrix;  
    %GOTO trcend;  
  %END;  
  
  DATA _&in_data.;  
    SET &in_data.;  
    RENAME  
    %DO i = 1 %TO &obs1.;  
      %SCAN(&varnam., &i.) = col&i.  
    %END;;  
  RUN;  
  
  DATA &out_data(KEEP=trace);  
    SET _&in_data end=eof;  
    RETAIN trace 0;  
    %DO i = 1 %TO &obs1.;  
      IF _n_ = &i. THEN temp=col&i.;  
    %END;  
    Trace=Trace+temp;  
    IF eof;  
    LABEL trace="Trace";
```

```

RUN;

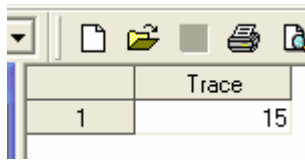
PROC DATASETS LIBRARY=work MTYPE=data;
    DELETE permute _&in_data.;
QUIT;

PROC PRINT DATA=&out_data. LABEL;
RUN;
%trcend:

%MEND trace;
%trace(libname = work
      ,in_data = temp
      ,out_data = tempo);

```

OUTPUT DATA SET TEMPO :



Trace	
1	15

PRODUCT OF TWO MATRICES:

For this operation, consider matrices A and B.

The approach used is transposing the second dataset and then taking the product of each row of the first dataset with each row of the other dataset.

SAS CODE:

```

%MACRO mat_product(libname1 = work
                  ,libname2 = work
                  ,in_data1 =
                  ,in_data2 =
                  ,out_data =
                  );

PROC SQL NOPRINT;
    SELECT nobs INTO :obs1 FROM sashelp.vtable
        WHERE libname="%upcase(&libname1.)" AND
              memname="%upcase(&in_data1.)";
    SELECT nobs INTO :obs2 FROM sashelp.vtable
        WHERE libname="%upcase(&libname2.)" AND
              memname="%upcase(&in_data2.)";
    SELECT count(distinct name) INTO :obscol1 FROM sashelp.vcolumn
        WHERE libname="%upcase(&libname1.)" AND
              memname="%upcase(&in_data1.)";
    SELECT count(distinct name) INTO :obscol2 FROM sashelp.vcolumn
        WHERE libname="%upcase(&libname2.)" AND
              memname="%upcase(&in_data2.)";
QUIT;

%IF &obs2. = &obscol1. %THEN %DO;

    PROC TRANSPOSE DATA=&libname2..&in_data2. OUT=_2 (DROP=_name_);
    RUN;

    DATA _1;
        SET &libname1..&in_data1.;

```

```

RUN;

%DO i = 1 %TO 2;
  PROC SQL NOPRINT;
    SELECT name INTO :mem&i. separated BY ' ' FROM sashelp.vcolumn
      WHERE libname="WORK" AND memname="%upcase(_&i.)";
    SELECT count(distinct name) INTO :cntmem&i. FROM sashelp.vcolumn
      WHERE libname="WORK" AND memname="%upcase(_&i.)";
  QUIT;

  DATA _&i.;
    SET _&i.;
    ind=_n_;
    %DO j = 1 %TO &&cntmem&i.;
      IF %SCAN(&&mem&i.,&j.)=. THEN %SCAN(&&mem&i.,&j.)=0;
      RENAME %SCAN(&&mem&i.,&j.) = col&i.&j.;
    %END;
  RUN;
%END;

%DO i = 1 %TO &obscol2.;
  PROC SQL NOPRINT;
    CREATE TABLE __&i. AS SELECT _1.ind,
      0 %DO j = 1 %TO &obscol1.;
      + (col1&j. * col2&j.)
      %END; AS r&i.
    FROM _1, (SELECT * FROM _2 WHERE ind=&i.) AS A;
  QUIT;

%END;

PROC SQL NOPRINT;
  CREATE TABLE &out_data. AS SELECT r1 %DO i = 2 %TO &obscol2.; ,r&i.
      %END;
    FROM __1(KEEP=ind r1) %DO i = 2 %TO &obscol2.;
      LEFT JOIN __&i.(KEEP=ind r&i.) ON __1.ind = __&i..ind
      %END;;
  QUIT;

  DATA &out_data. (DROP=i);
    SET &out_data.;
    ARRAY row{&obscol2.} r1 - r%CMPPRES(&obscol2.);
    DO i = 1 TO &obscol2.;
      IF row{i} < 0.00000000001 THEN row{i} = 0;
    END;
  RUN;

PROC DATASETS LIBRARY=work MTYPE=DATA;
  DELETE _1 _2 %DO i = 1 %TO &obscol2. ;
    __&i.
    %END;;
  QUIT;

PROC PRINT DATA=&out_data. LABEL;
  TITLE "The Product of matrix &in_data1. with &in_data2.";
  RUN;

%END;
%ELSE %DO;
  %PUT NOTE: The number of rows in the first matrix is not same as the
    number of columns in the other matrix, please check the
    dimensions and try again;

```



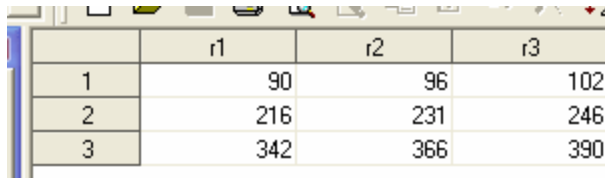
```

%END;

%MEND mat_product;
%mat_product(in_data1 = temp
             ,in_data2 = temp_
             ,out_data = outprd );

```

OUTPUT DATA SET OUTPRD:



	r1	r2	r3
1	90	96	102
2	216	231	246
3	342	366	390

INVERSE OF A MATRIX:

For this operation, we will be using the 5x5 matrix.

Here first we get the cofactor matrix then get the adjoint matrix i.e transpose of the cofactor matrix and then divide each value with the determinant of the original matrix.

SAS CODE:

```

%MACRO inverse(libnm_in = work
               ,datain   =
               ,libnm_ot = work
               ,dataout   =
               ,cleanup   = Y);

PROC SQL NOPRINT;
  SELECT count (DISTINCT name) INTO :colno FROM dictionary.columns
    WHERE libname="%upcase(&libnm_in.)" AND memname="%upcase(&datain.)";
  SELECT DISTINCT name INTO :colnmns separated BY ' ' FROM
    dictionary.columns
    WHERE libname="%upcase(&libnm_in.)" AND memname="%upcase(&datain.)";
  SELECT nobs INTO :rowno FROM dictionary.tables
    WHERE libname="%upcase(&libnm_in.)" AND memname="%upcase(&datain.)";
QUIT;
%IF &colno. LE 1 OR &rowno. LE 1 OR &colno. NE &rowno. %THEN %DO;
  %PUT Check the dimensions of the matrix;
  %GOTO invend;
%END;

%det(libname = &libnm_in.
     ,in_data = &datain.
     ,out_data = det1); /*Getting the Determinant of the original matrix*/

DATA _null_;
  SET det1;
  CALL symput("matchk",compress(put(det,best.)));
RUN;

%IF &matchk. = 0 %THEN %DO;
  %PUT Inverse does not exist;
  %GOTO invend;
%END;

%DO col = 1 %TO &colno.;

```

```

%DO row = 1 %TO &rowno.;
  DATA _cof&row.&col.;
    SET &libnm_in..&datain.;
    DROP %SCAN(&colms,&col.);
    IF _n_ ^= &row.;
  RUN;
  %det(libname = work
    ,in_data = _cof&row.&col.
    ,out_data = _cofac&row.&col.);

  DATA _null_;
    SET _cofac&row.&col.;
    CALL symput("cofac&row.&col.", compress(put(det, best.)));
  RUN;
%END;
%END;

DATA invert(drop=i j);
  ARRAY dumvar{&colno.} cofac1 - cofac&cmpres(&colno.);
  DO j = 1 TO &rowno.;
    DO i = 1 TO &colno.;
      dumvar{i}=.;
    END;
  OUTPUT;
END;
RUN;

DATA invert1;
  SET invert;
  %DO colm = 1 %TO &colno.;
    %DO rowm = 1 %TO &rowno.;
      IF _n_ = &rowm. THEN cofac&colm. = ((-1)**%EVAL(&rowm.+&colm.))
        *(&cofac&rowm.&colm.);
    %END;
  %END;
RUN;

PROC TRANSPOSE DATA=invert1 OUT=invert2(DROP=_name_);
RUN;

PROC SQL NOPRINT;
  CREATE TABLE &libnm_ot..&dataout. AS SELECT
    col1/det AS cofactor1
    %DO p = 2 %TO &colno.;
      , col&p./det AS cofactor&p.
    %END; FROM invert2, det1;
QUIT;

%IF &cleanup. = Y %THEN %DO;
  PROC DATASETS MTYPE=data LIBRARY=work;
    DELETE _cof: invert invert1 invert2 det1;
  QUIT;
%END;
%inwend:

%MEND inverse;
%inverse(datain = inv
  ,dataout = inv_out);

```

OUTPUT DATA SET INV_OUT:

	cofactor1	cofactor2	cofactor3	cofactor4	cofactor5
1	0.512195122	-2.073170732	1.2926829268	-0.170731707	0.0487804878
2	-1.52195122	4.1317073171	-2.126829268	0.1073170732	-0.087804878
3	1.2487804878	-3.492682927	1.7707317073	0.1170731707	-0.004878049
4	-0.248780488	1.4926829268	-0.770731707	-0.117073171	0.0048780488
5	-0.219512195	0.3170731707	-0.268292683	0.0731707317	0.1219512195

/*To check the surety of the Inverse we multiply the original matrix with its inverse and the expected output is an Identity matrix;*/

```
%mat_product(in_data1 = inv
              ,in_data2 = inv_out
              ,out_data = inv_inv_out );
```

OUTPUT DATA SET INV INV_OUT :

	r1	r2	r3	r4	r5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	1	0	0
4	0	0	0	1	0
5	0	0	0	0	1

CONCLUSION

This paper proposes a solution for some basic matrix calculations in the absence of SAS/IML.

The algorithm proposed here in case of calculating the inverse is for full rank matrix. In case of non full rank matrix the code will output a note in the log and quit. Also for calculating the determinant and trace of a matrix the above code requires square matrix failing which will lead to the generation of a note in the log and termination of the macro.

ACKNOWLEDGMENTS

We thank Dr. Sundar Ramamoorthy, Vice President, Mumbai Pharma Operations, Accenture Services Pvt. Ltd. and all our colleagues for guiding this work and carefully reviewing the paper with comments and suggestions.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Sarfaraz Sayyed
Accenture Services Pvt. Ltd.,
Tower 1, 1st Floor, Logitech Park,
Andheri-Kurla Road, Andheri (E) – 400059
Mumbai, India
Mobile: +91-9867893738
E- mail: s.sayyed@accenture.com
: sarfaraz.sayyed@gmail.com

Binoy Varghese
Cybrid, Inc
4807 Jonestown Road, Suite 253
Harrisburg, PA 17109
Email: mailme@binoyvarghese.com
Website: www.clinicalsasprogramming.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.