SAS Macro for the Generation, Randomization and Analysis of MERC Designs for Pair-wise Comparison among gca Effects

Eldho Varghese, Cini Varghese, Seema Jaggi and Rajender Parsad
ICAR-Indian Agricultural Statistics Research Institute, New Delhi

In plant breeding programmes, Mating-Environmental (ME) designs are commonly used under one-way blocking setup to get an estimate for general and specific combining abilities (gca and sca) of inbred lines involved in the crosses. When a large number of crosses are to be compared, which lead to a large experimental area, it may be important to account for or eliminate the effects of fertility trends in the land in two directions. In such situations, designs with crosses arranged in a row-column (RC) set up can be more advantageously used. Though information on both gca and sca effects are important to the breeders, most of the statistical papers dealing with these designs assumes sca effects as negligible for reducing the complexity in mathematical derivations. Here, a set of macros have been given for generating three classes of efficient MERC designs for complete diallel cross (CDC) experiments which are variance balanced for estimating the contrasts pertaining to the gca effects free from sca effects when two cross classified sources of variations are present in the experimental material **for making all pair-wise comparison among gca effects**. These programs were developed using SAS IML in the form of macros so that user would be able to get the design by providing only number lines (t). Besides, SAS codes have been developed for the analysis of the data generated from such designs and the code is given in the section **SAS code for the analysis**.

MERC design for all pairwise comparison among gca effects

Series 1: The parameters are number of crosses (v) = number of rows (p) = number of columns (q) = number of replications (r) = $\frac{t(t-1)}{2}$, where t is the number of lines.

1. SAS Macro for generation of designs using Method 1 (Varghese et al., 2015)

```
%let t=5;/*number of lines*/
proc iml;
c_no=comb(&t,2);
n_no=&t;
k=1;
d0=j(c_no,3,0);
do i=1 to n_no-1;
do j=i+1 to n_no;
d0[k,1]=i;
d0[k,2]=j;
d0[k,3]=k;
k=k+1;
end;
end;
*print d0;
d1=j((c_no*c_no),3,0);
do l=1 to c_no;
do i=1 to c_no;
```

```
if i+(l-1)<(c_no+1) then do;
d1[((l-1)*c_no)+i, ]=d0[(l-1)+i, ];
end;
else do;
d1[((l-1)*c_no)+i, ]=d0[(i+(l-1))-c_no, ];
end;
end;
end;
*print d1;
k=1;
x=j((c_no*c_no),2,0);
do i=1 to c_no;
do j=1 to c_no;
x[k,1]=i;
x[k,2]=j;
k=k+1;
end;
end;
*print x;
xx=x||d1;
*print xx;
/*****************code for generating designs*******/
ww=j(c_no,c_no,0);
k=1;
do i=1 to c_no;
do j=1 to c_no;
ww[i,j]=d1[k,1];
k=k+1;
end;
end;
*print ww;
ww1=j(c_no,c_no,0);
k=1;
do i=1 to c_no;
do j=1 to c_no;
ww1[i,j]=d1[k,2];
k=k+1;
end;
end;
*print ww1;
ww_=char(ww,4,0);
ww1_=char(ww1,4,0);
www=j(nrow(ww),ncol(ww),' x');
MERC_Design=ww_+www+ww1_;
print MERC_Design;
/************************************************/
/*********Randmization*****************************/

/*****row-randomization*******/
r=j(1,nrow(ww),0);
call randgen(r,'uniform');
*print r;
```

```
rr=rank(r);
*print rr;
ra=j(nrow(ww),ncol(ww),0);
random_row=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
random_row[i,j]= MERC_Design[rr[i],j];
end;
end;
*print random_row;
/*******/

/******column-randomization****/
r=j(ncol(ww),1,0);
call randgen(r,'uniform');
*print r;
rr=rank(r);
*print rr;
Randomized_Layout=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
Randomized_Layout[i,j]= random_row[i,rr[j]];
end;
end;
print Randomized_Layout; /******/
```

SAS Output

MERC_Design										
	COL1	COL2	COL3	COL4	COL5	COL6	COL7	COL8	COL9	COL10
ROW1	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5
ROW2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2
ROW3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3
ROW4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4
ROW5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5
ROW6	2 x 4	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3
ROW7	2 x 5	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4
ROW8	3 x 4	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5
ROW9	3 x 5	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4
ROW10	4 x 5	1 x 2	1 x 3	1 x 4	1 x 5	2 x 3	2 x 4	2 x 5	3 x 4	3 x 5

Randomized_Layout										
	COL1	COL2	COL3	COL4	COL5	COL6	COL7	COL8	COL9	COL10
ROW1	2 x 3	2 x 5	1 x 2	4 x 5	2 x 4	3 x 5	1 x 3	3 x 4	1 x 4	1 x 5
ROW2	1 x 3	1 x 5	3 x 4	2 x 5	1 x 4	2 x 4	3 x 5	2 x 3	4 x 5	1 x 2
ROW3	3 x 5	1 x 2	2 x 3	1 x 5	4 x 5	1 x 4	2 x 4	1 x 3	2 x 5	3 x 4
ROW4	2 x 5	3 x 5	1 x 4	1 x 3	3 x 4	1 x 2	1 x 5	4 x 5	2 x 3	2 x 4
ROW5	4 x 5	1 x 3	2 x 4	2 x 3	1 x 2	1 x 5	2 x 5	1 x 4	3 x 4	3 x 5
ROW6	2 x 4	3 x 4	1 x 3	1 x 2	2 x 5	4 x 5	1 x 4	3 x 5	1 x 5	2 x 3
ROW7	3 x 4	4 x 5	1 x 5	1 x 4	3 x 5	1 x 3	2 x 3	1 x 2	2 x 4	2 x 5
ROW8	1 x 4	2 x 3	3 x 5	3 x 4	1 x 5	2 x 5	4 x 5	2 x 4	1 x 2	1 x 3
ROW9	1 x 2	1 x 4	2 x 5	2 x 4	1 x 3	2 x 3	3 x 4	1 x 5	3 x 5	4 x 5
ROW10	1 x 5	2 x 4	4 x 5	3 x 5	2 x 3	3 x 4	1 x 2	2 x 5	1 x 3	1 x 4

2. SAS Macro for generation of designs using Method 2 (Varghese et al., 2015)

Series 2: The parameters are $v = \frac{t(t-1)}{2}$, $p = \frac{(t-3)(t-2)}{2}$, $q = \frac{t(t-1)}{2}$ and $r = \frac{(t-3)(t-2)}{2}$, where t (number of lines) should be a prime number.

```
%let t=7;/*t should be prime*/
proc iml;
c_no=comb(&t,2);
n_no=&t;
k=1;
d0=j(n_no-2,((n_no-1)/2)*n_no,0);
do k=1 to (n_no-1)/2;
do i=1 to n_no-2;
do j=1 to n_no;
d0[i,j+((k-1)*n_no)]=mod((j+(i-1)+(i-1)*(k-1)),n_no);
if d0[i,j+((k-1)*n_no)]=0 then d0[i,j+((k-1)*n_no)]=n_no;
end;
end;
end;
*print d0[format=3.0];
d1=j((n_no-2)*(n_no-3)/2,(n_no-1)*n_no,0);
do j=1 to (n_no-1/2)*n_no;
l=1;
do i=1 to (n_no-2);
do q=1 to (n_no-2)-i;
d1[l,(2*j)-1]=d0[i,j];
d1[l,(2*j)]=d0[i+q,j];
l=l+1;
end;
end;
end;
*print d1;
d2=j(((n_no-2)*(n_no-3)/2)*c_no,2,0);
l=1;
do j=1 to ((n_no-1)*n_no)/2;
do i=1 to (n_no-2)*(n_no-3)/2;
d2[l,1]=d1[i,(2*j)-1];
d2[l,2]=d1[i,2*j];
l=l+1;
end;
end;
*print d2;
k=1;
d00=j(c_no,3,0);
do i=1 to n_no-1;
do j=i+1 to n_no;
d00[k,1]=i;
d00[k,2]=j;
d00[k,3]=k;
k=k+1;
end;
```

```
end;
*print d00;
d22=j(((n_no-2)*(n_no-3)/2)*c_no,1,0);
do i=1 to ((n_no-2)*(n_no-3)/2)*c_no;
do k=1 to c_no;
if (d2[i,1]=d00[k,1] & d2[i,2]=d00[k,2])|(d2[i,1]=d00[k,2] &
d2[i,2]=d00[k,1])
then d22[i,1]=d00[k,3];
end;
end;
*print d22;
x=j(((n_no-2)*(n_no-3)/2)*c_no,2,0);
k=1;
do j=1 to c_no;
do i=1 to ((n_no-2)*(n_no-3)/2);
x[k,1]=i;
x[k,2]=j;
k=k+1;
end;
end;
*print x;
xx=x||d2||d22;
*print xx;
/*****************code for generating designs*******/
ww=j(((n_no-2)*(n_no-3)/2),c_no,0);
k=1;
do j=1 to c_no;
do i=1 to ((n_no-2)*(n_no-3)/2);
ww[i,j]=d2[k,1];
k=k+1;
end;
end;
*print ww;
ww1=j(((n_no-2)*(n_no-3)/2),c_no,0);
k=1;
do j=1 to c_no;
do i=1 to ((n_no-2)*(n_no-3)/2);
ww1[i,j]=d2[k,2];
k=k+1;
end;
end;
*print ww1;
ww_=char(ww,4,0);
ww1_=char(ww1,4,0);
www=j(nrow(ww),ncol(ww),' x');
MERC_Design=ww_+www+ww1_;
print MERC_Design;
/************************************************/
/*********Randmization*****************************/

/*****row-randomization*******/
r=j(1,nrow(ww),0);
```

```
call randgen(r,'uniform');
*print r;
rr=rank(r);
*print rr;
ra=j(nrow(ww),ncol(ww),0);
random_row=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
random_row[i,j]= MERC_Design[rr[i],j];
end;
end;
*print random_row;
/*******/

/******column-randomization****/
r=j(ncol(ww),1,0);
call randgen(r,'uniform');
*print r;
rr=rank(r);
*print rr;
Randomized_Layout=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
Randomized_Layout[i,j]= random_row[i,rr[j]];
end;
end;
print Randomized_Layout;
/******/
```

SAS Output

MERC_design

	COL1	COL2	COL3	COL4	COL5	COL6	COL7	COL8	COL9	COL10	COL11	COL12	COL13	COL14	COL15	COL16	COL17	COL18	COL19	COL20	COL21
ROW1	1 x 2	2 x 3	3 x 4	4 x 5	5 x 6	6 x 7	7 x 1	1 x 3	2 x 4	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2	1 x 4	2 x 5	3 x 6	4 x 7	5 x 1	6 x 2	7 x 3
ROW2	1 x 3	2 x 4	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2	1 x 5	2 x 6	3 x 7	4 x 1	5 x 2	6 x 3	7 x 4	1 x 7	2 x 1	3 x 2	4 x 3	5 x 4	6 x 5	7 x 6
ROW3	1 x 4	2 x 5	3 x 6	4 x 7	5 x 1	6 x 2	7 x 3	1 x 7	2 x 1	3 x 2	4 x 3	5 x 4	6 x 5	7 x 6	1 x 3	2 x 4	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2
ROW4	1 x 5	2 x 6	3 x 7	4 x 1	5 x 2	6 x 3	7 x 4	1 x 2	2 x 3	3 x 4	4 x 5	5 x 6	6 x 7	7 x 1	1 x 6	2 x 7	3 x 1	4 x 2	5 x 3	6 x 4	7 x 5
ROW5	2 x 3	3 x 4	4 x 5	5 x 6	6 x 7	7 x 1	1 x 2	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2	1 x 3	2 x 4	4 x 7	5 x 1	6 x 2	7 x 3	1 x 4	2 x 5	3 x 6
ROW6	2 x 4	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2	1 x 3	3 x 7	4 x 1	5 x 2	6 x 3	7 x 4	1 x 5	2 x 6	4 x 3	5 x 4	6 x 5	7 x 6	1 x 7	2 x 1	3 x 2
ROW7	2 x 5	3 x 6	4 x 7	5 x 1	6 x 2	7 x 3	1 x 4	3 x 2	4 x 3	5 x 4	6 x 5	7 x 6	1 x 7	2 x 1	4 x 6	5 x 7	6 x 1	7 x 2	1 x 3	2 x 4	3 x 5
ROW8	3 x 4	4 x 5	5 x 6	6 x 7	7 x 1	1 x 2	2 x 3	5 x 7	6 x 1	7 x 2	1 x 3	2 x 4	3 x 5	4 x 6	7 x 3	1 x 4	2 x 5	3 x 6	4 x 7	5 x 1	6 x 2
ROW9	3 x 5	4 x 6	5 x 7	6 x 1	7 x 2	1 x 3	2 x 4	5 x 2	6 x 3	7 x 4	1 x 5	2 x 6	3 x 7	4 x 1	7 x 6	1 x 7	2 x 1	3 x 2	4 x 3	5 x 4	6 x 5
ROW10	4 x 5	5 x 6	6 x 7	7 x 1	1 x 2	2 x 3	3 x 4	7 x 2	1 x 3	2 x 4	3 x 5	4 x 6	5 x 7	6 x 1	3 x 6	4 x 7	5 x 1	6 x 2	7 x 3	1 x 4	2 x 5

Randomized_Layout

	COL1	COL2	COL3	COL4	COL5	COL6	COL7	COL8	COL9	COL10	COL11	COL12	COL13	COL14	COL15	COL16	COL17	COL18	COL19	COL20	COL21
ROW1	4 x 5	1 x 3	1 x 4	3 x 6	6 x 7	7 x 2	6 x 1	5 x 7	2 x 4	3 x 4	4 x 6	5 x 6	2 x 3	5 x 1	6 x 2	1 x 2	3 x 5	7 x 3	7 x 1	2 x 5	4 x 7
ROW2	1 x 5	2 x 3	6 x 4	1 x 6	3 x 7	1 x 2	7 x 1	6 x 7	3 x 4	7 x 4	5 x 6	2 x 6	6 x 3	3 x 1	4 x 2	5 x 2	4 x 5	5 x 3	4 x 1	7 x 5	2 x 7
ROW3	3 x 5	6 x 3	5 x 4	7 x 6	5 x 7	5 x 2	4 x 1	3 x 7	7 x 4	2 x 4	2 x 6	4 x 6	1 x 3	2 x 1	3 x 2	7 x 2	1 x 5	4 x 3	6 x 1	6 x 5	1 x 7
ROW4	2 x 4	4 x 1	2 x 1	4 x 3	4 x 6	3 x 7	2 x 6	1 x 5	5 x 2	1 x 3	7 x 4	3 x 5	7 x 2	6 x 5	7 x 6	6 x 1	6 x 3	1 x 7	5 x 7	3 x 2	5 x 4
ROW5	3 x 4	6 x 1	5 x 1	7 x 3	5 x 6	5 x 7	4 x 6	3 x 5	7 x 2	2 x 3	2 x 4	4 x 5	1 x 2	2 x 5	3 x 6	7 x 1	1 x 3	4 x 7	6 x 7	6 x 2	1 x 4
ROW6	2 x 3	4 x 6	2 x 5	4 x 7	4 x 5	3 x 5	2 x 4	1 x 3	5 x 7	1 x 2	7 x 2	3 x 4	7 x 1	6 x 2	7 x 3	6 x 7	6 x 1	1 x 4	5 x 6	3 x 6	5 x 1
ROW7	2 x 5	4 x 3	2 x 4	4 x 6	4 x 7	3 x 2	2 x 1	1 x 7	5 x 4	1 x 4	7 x 6	3 x 6	7 x 3	6 x 1	7 x 2	6 x 2	6 x 5	1 x 3	5 x 1	3 x 5	5 x 7
ROW8	1 x 2	2 x 4	6 x 2	1 x 4	3 x 4	1 x 3	7 x 2	6 x 1	3 x 5	7 x 1	5 x 7	2 x 3	6 x 7	3 x 6	4 x 7	5 x 6	4 x 6	5 x 1	4 x 5	7 x 3	2 x 5
ROW9	1 x 4	2 x 1	6 x 1	1 x 3	3 x 6	1 x 7	7 x 6	6 x 5	3 x 2	7 x 3	5 x 4	2 x 5	6 x 2	3 x 5	4 x 6	5 x 1	4 x 3	5 x 7	4 x 7	7 x 2	2 x 4
ROW10	1 x 3	2 x 6	6 x 5	1 x 7	3 x 5	1 x 5	7 x 4	6 x 3	3 x 7	7 x 2	5 x 2	2 x 4	6 x 1	3 x 2	4 x 3	5 x 7	4 x 1	5 x 4	4 x 6	7 x 6	2 x 1

3. SAS Macro for generation of designs using Method 3 (Varghese et al., 2015)

Series 3: The parameters are $v = \frac{t(t-1)}{2}$, $p = \frac{t(t-1)}{2}$, $q = t$ and $r = t$. where t (number of lines) should be a prime number.

```
%let t=4;/*when t is a power of 2*/
proc iml;
p=log(&t)/log(2);
*print p;
aa=j(p,1,0);
aa1=j(p,1,0);
do i=1 to p;
aa[i,1]=2**(i-1);
aa1[i,1]=2**(p-i);
end;
*print aa aa1;
aaa=j(p,(&t/2)-1,1);
do i=1 to (p-1);
do j=1 to (&t/2)-1;
if mod(j,aa[i,1])=0 then
aaa[i,j]=(aa[i,1]+1);
end;
end;
*print aaa;
aaaa=j(&t-1,(&t/2)-1,0);
l=1;
do i=1 to p;
do k=1 to aa1[i,1];
aaaa[l, ]=aaa[i, ];
l=l+1;
end;
end;
*print aaaa [format=3.0];
k=1;
a1=j(&t/2,&t,0);
do i= 1 to &t/2;
a1[i,1]=k;
a1[i,2]=2*i;
end;
*print a1[format=3.0];
a2=j((&t/2)-1,&t,0);
l=1;
do i=2 to p;
do j=1 to aa1[i,1];
a2[l,1]=k;
a2[l,2]=(2*(aa[i,1]*(j-1)))+aa[i,1]+1;
l=l+1;
end;
end;
*print a2 [format=3.0];
a=a1//a2;
do i=1 to (&t/2)-1;
```

```
do j=1 to 2;
a[ ,((2*i)+j)]=mod((a[ ,((2*i)+j)-2]+aaaa[ ,i]),&t);
end;
end;
*print a[format=3.0];
do i=1 to &t-1;
do j=1 to &t;
if a[i,j]=0 then a[i,j]=&t;
end;
end;
rsuma=a[ ,+];
*print a[format=3.0];
*print rsuma;
x_=j((&t/2)*(&t-1),&t,0);
do k=1 to &t/2;
do i=1 to (&t-1);
do j=1 to &t;
if j+(2*(k-1))>&t then
            do;
            x_[i+((&t-1)*(k-1)),j]=a[i,j+(2*(k-1))-&t];
            end;
            else do;
            x_[i+((&t-1)*(k-1)),j]=a[i,j+(2*(k-1))];
            end;
end;
end;
end;
x_1=j(nrow(x_)-1,ncol(x_),0);
do i=2 to nrow(x_);
x_1[i-1, ]=x_[i, ];
end;
x_1=x_1//x_[1, ];
x_=x_||x_1;
*print x_;
c_no=comb(&t,2);
n_no=&t;
d00=j((c_no*&t),2,0);
l=1;
do j=1 to &t;
do i=1 to c_no;
do k=1 to 2;
d00[l,k]=x_[i,(2*(j-1))+k];
end;
l=l+1;
end;
end;
*print d00;
k=1;
d0=j(c_no,3,0);
do i=1 to c_no;
d0[k,1]=d00[i,1];
d0[k,2]=d00[i,2];
```

```
d0[k,3]=i;
k=k+1;
end;
*print d0;
d01=j((c_no*&t/2),1,0);
l=1;
do j=1 to &t/2;
do i=1 to c_no;
if (&t-1)*(j-1)+i < c_no+1 then
                        do;
                        d01[l,1]=(&t-1)*(j-1)+i;
                        end;
                        else do;
                        d01[l,1]=(&t-1)*(j-1)+i-c_no;
                        end;
l=l+1;
end;
end;
*print d01;
d02=j((c_no*&t/2),1,0);
do i=1 to c_no*&t/2;
d02[i,1]=mod(d01[i, ]+1,(&t*(&t-1)/2));
if d02[i,1]=0 then d02[i,1]=(&t*(&t-1)/2);
end;
*print d02;
d1=d00||(d01//d02);
*print d1;
k=1;
x=j((c_no*&t),2,0);
do j=1 to &t;
do i=1 to c_no;
x[k,1]=i;
x[k,2]=j;
k=k+1;
end;
end;
*print x;
xx=x||d1;
*print xx;
/*****************code for generating designs*******/
ww=j(c_no,&t,0);
k=1;
do j=1 to &t;
do i=1 to c_no;
ww[i,j]=d1[k,1];
k=k+1;
end;
end;
*print ww;
ww1=j(c_no,&t,0);
k=1;
do j=1 to &t;
```

```
do i=1 to c_no;
ww1[i,j]=d1[k,2];
k=k+1;
end;
end;
*print ww1;
ww_=char(ww,4,0);
ww1_=char(ww1,4,0);
www=j(nrow(ww),ncol(ww),' x');
MERC_design=ww_+www+ww1_;
print MERC_design;
/*************************************************/
/*********Randmization*****************************/

/*****row-randomization*******/
r=j(1,nrow(ww),0);
call randgen(r,'uniform');
*print r;
rr=rank(r);
*print rr;
ra=j(nrow(ww),ncol(ww),0);
random_row=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
random_row[i,j]= MERC_design[rr[i],j];
end;
end;
*print random_row;
/*******/

/******column-randomization****/
r=j(ncol(ww),1,0);
call randgen(r,'uniform');
*print r;
rr=rank(r);
*print rr;
Randomized_Layout=char(ra,10,0);
do i=1 to nrow(ww);
do j=1 to ncol(ww);
Randomized_Layout[i,j]= random_row[i,rr[j]];
end;
end;
print Randomized_Layout;
/******/
```

SAS Output

MERC_design			
1 x 2	3 x 4	1 x 4	3 x 2
1 x 4	3 x 2	1 x 3	2 x 4
1 x 3	2 x 4	3 x 4	1 x 2
3 x 4	1 x 2	3 x 2	1 x 4
3 x 2	1 x 4	2 x 4	1 x 3
2 x 4	1 x 3	1 x 2	3 x 4

Randomized_Layout			
1 x 4	2 x 4	3 x 2	1 x 3
3 x 4	1 x 4	1 x 2	3 x 2
1 x 2	3 x 2	3 x 4	1 x 4
1 x 3	1 x 2	2 x 4	3 x 4
3 x 2	1 x 3	1 x 4	2 x 4
2 x 4	3 x 4	1 x 3	1 x 2

SAS code for the analysis

```
Data MERC;
input row column line1 line2 cross yld;
cards;
1     1     1     2     1     31
1     2     1     3     2     58
1     3     1     4     3     76
1     4     1     5     4     101
1     5     2     3     5     118
1     6     2     4     6     21
1     7     2     5     7     137
1     8     3     4     8     156
1     9     3     5     9     173
1     10    4     5     10    193
2     1     1     3     2     57
2     2     1     4     3     78
2     3     1     5     4     99
2     4     2     3     5     120
2     5     2     4     6     81
2     6     2     5     7     88
2     7     3     4     8     157
2     8     3     5     9     179
2     9     4     5     10    193
2     10    1     2     1     113
3     1     1     4     3     75
3     2     1     5     4     99
3     3     2     3     5     116
3     4     2     4     6     81
```

3	5	2	5	7	146
3	6	3	4	8	106
3	7	3	5	9	178
3	8	4	5	10	197
3	9	1	2	1	120
3	10	1	3	2	146
4	1	1	5	4	100
4	2	2	3	5	120
4	3	2	4	6	81
4	4	2	5	7	150
4	5	3	4	8	168
4	6	3	5	9	131
4	7	4	5	10	200
4	8	1	2	1	128
4	9	1	3	2	148
4	10	1	4	3	168
5	1	2	3	5	117
5	2	2	4	6	81
5	3	2	5	7	146
5	4	3	4	8	168
5	5	3	5	9	189
5	6	4	5	10	149
5	7	1	2	1	127
5	8	1	3	2	152
5	9	1	4	3	166
5	10	1	5	4	189
6	1	2	4	6	18
6	2	2	5	7	86
6	3	3	4	8	104
6	4	3	5	9	129
6	5	4	5	10	147
6	6	1	2	1	16
6	7	1	3	2	91
6	8	1	4	3	110
6	9	1	5	4	127
6	10	2	3	5	146
7	1	2	5	7	134
7	2	3	4	8	155
7	3	3	5	9	176
7	4	4	5	10	198
7	5	1	2	1	125
7	6	1	3	2	91
7	7	1	4	3	160
7	8	1	5	4	182
7	9	2	3	5	195
7	10	2	4	6	158
8	1	3	4	8	152
8	2	3	5	9	176
8	3	4	5	10	194
8	4	1	2	1	125
8	5	1	3	2	149
8	6	1	4	3	109

```
8     7     1     5     4     181
8     8     2     3     5     199
8     9     2     4     6     156
8     10    2     5     7     223
9     1     3     5     9     173
9     2     4     5     10    194
9     3     1     2     1     121
9     4     1     3     2     149
9     5     1     4     3     167
9     6     1     5     4     130
9     7     2     3     5     198
9     8     2     4     6     160
9     9     2     5     7     221
9     10    3     4     8     241
10    1     4     5     10    193
10    2     1     2     1     123
10    3     1     3     2     147
10    4     1     4     3     169
10    5     1     5     4     190
10    6     2     3     5     149
10    7     2     4     6     161
10    8     2     5     7     227
10    9     3     4     8     241
10    10    3     5     9     264
;
ods rtf file='anova.doc' startpage=no;
ods output overallanova=total_ss lsmeans=lsmeans;
proc glm data=MERC;
class row column cross;
model yld=row column cross/ss3;/*Type I sum of squares (ss1 instead of
ss3)should be used For designs where columns/rows are incomplete with
respect to crosses*/
lsmeans cross;
run;
ods output close;
proc iml;
use merc;
read all into xx;
*print xx;
row=xx[ ,1];
column=xx[ ,2];
cross=xx[ ,3]||xx[ ,4];
*print cross;
sca0=xx[ ,5];
*print sca0;
n_no=max(xx[ ,4]);
c_no=max(xx[ ,5]);
m=j(nrow(cross),1,1);
/*print cross;*/
x11=j(nrow(cross),max(cross),0);
k=1;
do i=1 to nrow(cross);
```

```
do j=1 to ncol(cross);
if cross[i,j]>0 then
x11[k,cross[i,j]]=x11[k,cross[i,j]]+1;
end;
k=k+1;
end;
*print x11;
sca=j(nrow(sca0),c_no,0);
k=1;
do i=1 to nrow(sca0);
if sca0[i, ]>0 then
sca[k,sca0[i, ]]=1;
k=k+1;
end;
*print sca[format=3.0];
rep=sca`*sca;
row_1=j(nrow(row),max(row),0);
k=1;
do i=1 to nrow(row);
if row[i, ]>0 then
row_1[k,row[i, ]]=1;
k=k+1;
end;
*print row_1;
col_1=j(nrow(column),max(column),0);
k=1;
do i=1 to nrow(column);
if column[i, ]>0 then
col_1[k,column[i, ]]=1;
k=k+1;
end;
*print col_1;
x1=sca;
x2=m||row_1||col_1;
c=(x1`*x1)-(x1`*x2)*ginv(x2`*x2)*(x2`*x1);
*print c[format=3.2];
q1=x11`*sca;
*print q1;
q=q1/rep[1,1];
*print q;
qq=q*q`;
*print qq;
inv_qq=inv(qq);
*print inv_qq;
qqq=inv_qq*q;
*print qqq;
jpv=j(n_no,c_no,1);
*print jpv;
jpv1=jpv*(.5/c_no);
*print jpv1;
h1=(inv_qq*q)-jpv1;
*h1=(inv_qq*q)-(((&n_no-1)/&c_no)*(inv_qq*jpv));
```

```
*print h1;
h2=i(c_no)-(q`*inv_qq*q);
*print h2;
rankh1=round(trace(ginv(h1)*h1));
*print rankh1;
rankh2=round(trace(ginv(h2)*h2));
*print rankh2;
h1h2=h1*h2`;
*print h1h2;
h1h1=h1*h1`;
h2h2=h2*h2`;
*print h1h1;
*print h2h2;
c11=h1*c*h1`;
c12=h1*c*h2`;
c21=h2*c*h1`;
c22=h2*c*h2`;
*print c11;
cross_t=j(max(xx[ ,5]),1,0);
do j=1 to max(xx[ ,5]);
do i= 1 to nrow(xx);
if xx[i,5]=j then cross_t[j,1]=cross_t[j,1]+xx[i,6];
end;
end;
*print cross t;
row_t=j(max(xx[ ,1]),1,0);
do j=1 to max(xx[ ,1]);
do i= 1 to nrow(xx);
if xx[i,1]=j then row_t[j,1]=row_t[j,1]+xx[i,6];
end;
end;
*print row_t;
col_t=j(max(xx[ ,2]),1,0);
do j=1 to max(xx[ ,2]);
do i= 1 to nrow(xx);
if xx[i,2]=j then col_t[j,1]=col_t[j,1]+xx[i,6];
end;
end;
*print col_t;
n1=sca`*row_1;
n2=sca`*col_1;
m=row_1`*col_1;
k1=row_1`*row_1;
k2=col_1`*col_1;
DF_gca=max(xx[ ,4])-1;/*Degrees of freedom for gca effects*/
DF_sca=max(xx[ ,5])-max(xx[ ,4]);/*Degrees of freedom for sca
effects*/
adj_cross=cross_t-(n1*inv(k1)*row_t)-(n2-n1*inv(k1)*m)*ginv(k2-
m`*inv(k1)*m)*(col_t-m`*inv(k1)*row_t);/*Adjusted cross totals*/
*print adj_cross;
SS_gca=adj_cross`*h1`*ginv(c11)*h1*adj_cross;/* sum of suares due to
gca*/
```

```
SS_sca=adj_cross`*h2`*ginv(c22)*h2*adj_cross;/* sum of suares due to
sca*/
*ss_cross=ss_gca+ss_sca;
*print ss_gca ss_sca;
MS_gca=ss_gca/df_gca;/* Mean square due to gca*/
MS_sca=ss_sca/df_sca;/* Mean square due to sca*/
*print ms_gca ms_sca;
use total_ss;
read all var{DF SS MS FValue ProbF} into y;
*print y;
FValue_gca=ms_gca/y[2,3];/*calculated value of F for testing gca
effects*/
FValue_sca=ms_sca/y[2,3];/*calculated value of F for testing sca
effects*/
*print fvalue_gca fvalue_sca;
FtabValue_gca=finv(1-.025,df_gca,y[2,1]);/*Table value of F for
testing gca effects*/
FtabValue_sca=finv(1-.025,df_sca,y[2,1]);/*Table value of F for
testing sca effects*/
*print FtabValue_gca FtabValue_sca;
ProbF_gca=1-probf(fvalue_gca,df_gca,y[2,1]);/* Significance of gca
effects*/
if probf_gca<.00001 then probf_gca='<.0001';
ProbF_sca=1-probf(fvalue_sca,df_sca,y[2,1]);/* Significance of sca
effects*/
if probf_sca<.00001 then probf_sca='<.0001';
*print probf_gca probf_sca;
print df_gca ss_gca ms_gca fvalue_gca FtabValue_gca probf_gca;
print df_sca ss_sca ms_sca fvalue_sca FtabValue_sca probf_sca;
use lsmeans;
read all into yy;
grand_mean=sum(xx[ ,6])/nrow(xx);
cross_est=yy-grand_mean;
*print cross_est;
gca_effects=h1*cross_est;
Line_number=j(nrow(gca_effects),1,0);
do i=1 to nrow(gca_effects);
Line_number[i, ]=i;
end;
gca_effects=Line_number||gca_effects;
print gca_effects;
zero=j(1,nrow(gca_effects)-2,0);
p1={1 -1};
p11=p1||zero;
var=(p11*ginv(c11)*p11`)*y[2,3];
Std_Error_Diff_gi_gj=sqrt(var);
print Std_Error_Diff_gi_gj;/*Standard Error of Difference of gca
effects*/
TValue=tinv(1-.025,y[2,1]);/*Table value of T for C.D. Values*/
Critical_Difference=std_error_diff_gi_gj*TValue;/*Critical differece
for pair-wise comparison*/
print Critical_Difference;
```

```
ods rtf close;
quit;
```

Output

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
row	10	1 2 3 4 5 6 7 8 9 10
column	10	1 2 3 4 5 6 7 8 9 10
cross	10	1 2 3 4 5 6 7 8 9 10

Number of Observations Read	100
Number of Observations Used	100

The GLM Procedure
Dependent Variable: yld

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	27	240034.6700	8890.1730	10975.5	<.0001
Error	72	58.3200	0.8100		
Corrected Total	99	240092.9900			

R-Square	Coeff Var	Root MSE	yld Mean
0.999757	0.633758	0.900000	142.0100

Source	DF	Type III SS	Mean Square	F Value	Pr > F
row	9	81172.89000	9019.21000	11134.8	<.0001
column	9	79584.09000	8842.67667	10916.9	<.0001
cross	9	79277.69000	8808.63222	10874.9	<.0001

Least Squares Means

cross	yld LSMEAN
1	102.900000
2	118.800000
3	127.800000

cross	yld LSMEAN
4	139.800000
5	147.800000
6	99.800000
7	155.800000
8	164.800000
9	176.800000
10	185.800000

DF_gca	SS_gca	MS_gca	FValue_gca	FtabValue_gca	ProbF_gca
4	66188.973	16547.243	20428.695	2.9693207	<.0001

DF_sca	SS_sca	MS_sca	FValue_sca	FtabValue_sca	ProbF_sca
5	13088.717	2617.7433	3231.7819	2.7483203	<.0001

gca_effects	
1	-26.24667
2	-20.58
3	13.386667
4	3.3866667
5	30.053333

Std_Error_Diff_gi_gj	Critical_Difference
0.697137	1.3897172

/**********/

Research Paper:

- Cini Varghese, **Eldho Varghese***, Seema Jaggi and Rajender Parsad. (2015). Row-column designs for diallel cross experiments with specific combining abilities. *Journal of the Indian Society of Agricultural Statistics*, **69 (2)**: 201-225.