APPENDIX A

This appendix contains computer programs (in Pascal language) used during the different tests.

A.1 Program for Inverse Distance Weighting

```
**********************************************************
This program is used to predict heights using an inverse
distance weighting method, the main inputs are the
number of data points(nd), number of check points(chk),
and the power for the distance(gama) and the three
dimensional coordinates of the data and check points
respectively, arranged in a text file called'ju1',the
outputs are the predicted heights of all points, in
addition, the max. error and the root mean square error
will be printed in an output file called 'ju1r'.
**********************************************************
program wm;
type
mat=array[1..20,1..20]of real;
vec=array[1..16] of real;
var
i,j,nd,chk:integer;
sum1,sum2,sum3,rms,tot,err,gama:real;
d:mat;
ju1,ju1r:text;
e,n,h,z,w,res:vec;
begin
assign(ju1,'ju1.text');
reset(ju1);
assign(ju1r,'ju1r.text');
rewrite(ju1r);
{writeln('input nd, chk');}
readln(ju1,nd,chk,gama);
for i:=1 to (nd+chk) do
begin
{writeln('input x,y,z');}
readln(ju1,e[i],n[i],h[i]);
end;
sum3:=0.0;tot:=0;
for i:= nd+1 to nd+chk do
begin
sum1:=0.0;sum2:=0.0;
for j := 1 to nd do
begin
```

```
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
w[j]:=1/(exp(gama*ln(d[i,j])));
sum1:=sum1+(h[j]*w[j]);
sum2:=sum2+w[j];
end;
z[i]:=sum1/sum2;
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;
err:=tot/chk;
rms:=sqrt(sum3/(chk-1));
writeln(ju1r,'err=',err:6:2,' ','rms=',rms:6:2);
close(ju1r);
end.
```

A.2 Program for Surface Fitting

```
**********************************************************
This program is used to predict heights using the least
squares surface fitting method, the main inputs are the
number of data points(nd), the number of check
points(chk), the three dimensional coordinates of the
data and check points and an option (op=1-4 for
equations (5.1)-(5.4) respectively) arranged in a text
file called'me3', the outputs are the predicted heights
of all points, in addition, the max. error and the root
mean square error will be printed in an output file
called 'mer'.
**********************************************************

program ee;
type
mat=array[1..25,1..25]of real;
vec=array[1..25] of real;
var
a,b,at,ata,atb,ataatb:mat;
e,n,h,z,res:vec;
nd,i,j,k,chk,op:integer;
sum,sum3,tot,rms,err:real;
me3:text;
begin
assign(me3,'me3.text');
reset(me3);
{assign(mer,'mer.text');
rewrite(mer);}
```

```
ata[i,j]:=0; atb[i,j]:= 0 ;ataatb[i,j]:=0;
{writeln('input nd,(nd*(nd-1))/2');}
readln(me3,nd,chk);
for i:=1 to nd+chk do
begin
{writeln('input e[i],n[i],h[i]');}
readln(me3,e[i],n[i],h[i]);
end;
writeln('input option');
readln(op);
sum3:=0;tot:=0;
if(op=1)then
begin
for i:= 1 to nd do
begin
a[i,1]:=1;
a[i,2]:=e[i];
b[i,1]:=h[i];
end;
for i:=1 to 2 do
for j:=1 to nd do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 2 do
for j:=1 to 2 do
for k:=1 to nd do
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
for i:=1 to 2  do
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 2 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 2 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 2 do
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
```

```
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to nd do
    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to 2 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
for i:= 1 to 2 do
begin
for j:=1 to 1 do
begin
writeln(ataatb[i,j]:6:3,' ');
end;end;
for i:=nd+1 to nd+chk do
begin
z[i]:=ataatb[1,1]+(ataatb[2,1]*e[i]);
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;end;
if(op=2)then
begin
for i:= 1 to nd do
begin
a[i,1]:=1;
a[i,2]:=n[i];
b[i,1]:=h[i];
end;
for i:=1 to 2 do
for j:=1 to nd do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 2 do
for j:=1 to 2 do
for k:=1 to nd do
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
for i:=1 to 2  do
```

```
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 2 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 2 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 2 do
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to nd do

    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to 2 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
for i:= 1 to 2 do
begin
for j:=1 to 1 do
begin
writeln(ataatb[i,j]:6:3,' ');

end;end;
for i:=nd+1 to nd+chk do
begin
z[i]:=ataatb[1,1]+(ataatb[2,1]*n[i]);
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;end;
if(op=3)then
begin
for i:= 1 to nd do
begin
a[i,1]:=1;
```

```
a[i,2]:=e[i];
a[i,3]:=n[i];

b[i,1]:=h[i];
end;
for i:=1 to 3 do
for j:=1 to nd do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 3 do
for j:=1 to 3 do
for k:=1 to nd do
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
for i:=1 to 3  do
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 3 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 3 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 3 do
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
for i:=1 to 3 do
for j:=1 to 1 do
for k:=1 to nd do
    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 3 do
for j:=1 to 1 do
for k:=1 to 3 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
for i:= 1 to 3 do
begin
```

```
for j:=1 to 1 do
begin
writeln(ataatb[i,j]:6:3,' ');
end;end;
for i:=nd+1 to nd+chk do
begin
z[i]:=ataatb[1,1]+(ataatb[2,1]*e[i])+(ataatb[3,1]*n[i]);
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;end;
if(op=4)then
begin
for i:= 1 to nd do
begin
a[i,1]:=1;
a[i,2]:=e[i];
a[i,3]:=n[i];
a[i,4]:=e[i]*n[i];
b[i,1]:=h[i];
end;
for i:=1 to 4 do
for j:=1 to nd do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 4 do
for j:=1 to 4 do
for k:=1 to nd do
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
for i:=1 to 4  do
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 4 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 4 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 4 do
```

```
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
for i:=1 to 4 do
for j:=1 to 1 do
for k:=1 to nd do
    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 4 do
for j:=1 to 1 do
for k:=1 to 4 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
for i:= 1 to 4 do
begin
for j:=1 to 1 do
begin
writeln(ataatb[i,j]:6:3,' ');
end;end;
for i:=nd+1 to nd+chk do
begin
z[i]:=ataatb[1,1]+(ataatb[2,1]*e[i])+(ataatb[3,1]*n[i])+
(ataatb[4,1]*e[i]*n[i]);
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;end;
err:=tot/chk;
rms:=sqrt(sum3/(chk-1));
writeln('err=',err:6:2,' ','rms=',rms:6:2);
{close(mer);}
readln;readln;
end.
```

A.3 Program for least squares prediction (linear)

```
**********************************************************
This program is used to predict heights linear least
squares prediction, the main inputs are the number of
data points(nd), q(=(nd(nd-1))/2),number of check
points(chk), and the three dimensional coordinates of
the data and check points respectively, arranged in a
data file called'me3',the outputs are the predicted
heights of all points, in addition, the max. error and
```

```
the root mean square error will be printed in an output
file called 'mer'.
**********************************************************
program ee;
type
mat=array[1..25,1..25]of real;
vec=array[1..25] of real;
var
d,a,b,at,ata,atb,ataatb,c,u,cu,p,pcu:mat;
e,n,h,z,res:vec;
nd,i,j,k,q,chk:integer;
sum,di,sum3,tot,rms,err:real;
me3:text;
begin
assign(me3,'me3.text');
reset(me3);
{assign(mer,'mer.text');
rewrite(mer);}
ata[i,j]:=0; atb[i,j]:= 0
;ataatb[i,j]:=0;cu[i,j]:=0;pcu[i,j]:=0;
{writeln('input nd,(nd*(nd-1))/2');}
readln(me3,nd,q,chk);
for i:=1 to nd+chk do
begin
{writeln('input e[i],n[i],h[i]');}
readln(me3,e[i],n[i],h[i]);
end;
k:=0;
for i:= 1 to nd-1 do
begin
for j:=i+1 to nd do
begin
k:=k+1;
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
a[k,1]:=1;
a[k,2]:=d[i,j];
b[k,1]:=h[i]*h[j];
end;end;
for i:=1 to 2 do
for j:=1 to q do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 2 do
for j:=1 to 2 do
for k:=1 to q do
```

```
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
 k:=0;
for i:=1 to 2  do
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 2 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 2 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 2 do
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to q do
    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to 2 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
{for i:= 1 to 2 do
begin
for j:=1 to 1 do
begin
writeln(mer,ataatb[i,j]:6:3,' ');
end;end;}
for i:=1 to nd do
begin
for j:=1 to nd do
begin
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
c[i,j]:=ataatb[1,1]+ataatb[2,1]*d[i,j];
c[j,i]:=ataatb[1,1]+ataatb[2,1]*d[i,j];
if(i=j)then
```

```
begin
c[i,j]:=ataatb[1,1];
end;end;
u[i,1]:=h[i];
end;
for i:= 1 to nd do
begin
for j:=1 to nd do
begin
write(c[i,j]:6:3,' ');
end;writeln;
end;
for i:=1 to nd  do
  begin
sum:=c[i,i];
c[i,i]:=1;
for j:=1 to nd do
c[i,j]:=(c[i,j])/(sum);
for k:=1 to nd do
  begin
if(k<>i) then
  begin
sum:=c[k,i];
c[k,i]:=0;
for j:=1 to nd do
c[k,j]:=(c[k,j])-sum*c[i,j];
end; end; end;
for i:=1 to nd do
for j:=1 to 1 do
for k:=1 to nd do
    begin
cu[i,j]:=cu[i,j]+c[i,k]*u[k,j];
end;
sum3:=0.0;tot:=0;
for i:= nd+1 to nd+chk do
begin
for j := 1 to nd do
begin
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
p[i,j]:=ataatb[1,1]+(ataatb[2,1]*d[i,j]);
end;end;
for i:=nd+1 to nd+chk do
begin
pcu[i,1]:=0;
for j:=1 to nd do
begin
```

```
pcu[i,1]:=pcu[i,1]+p[i,j]*cu[j,1];
end;
z[i]:=pcu[i,1];
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;
err:=tot/chk;
rms:=sqrt(sum3/(chk-1));
writeln('err=',err:6:2,' ','rms=',rms:6:2);
readln;readln;
end.
```

A.4 Program for least squares prediction (exponential)

```
**********************************************************
This program is used to predict heights least squares
prediction, the main inputs are the number of data
points(nd), q(=(nd(nd-1))/2),number of check
points(chk), and the three dimensional coordinates of
the data and check points respectively, arranged in a
data file called'ju1',the outputs are the predicted
heights of all points, in addition, the max. error and
the root mean square error will be printed in an output
file called 'ju1r'.
**********************************************************
program ee;
type
mat=array[1..25,1..25]of real;
vec=array[1..25] of real;
var
d,a,b,at,ata,atb,ataatb,c,u,cu,p,pcu:mat;
e,n,h,z,res:vec;
nd,i,j,k,q,chk:integer;
sum,sum1,avg,sum3,tot,rms,err,pro:real;
me3:text;
begin
assign(me3,'me3.text');
reset(me3);
{assign(mer,'mer.text');
rewrite(mer);}
ata[i,j]:=0; atb[i,j]:= 0
;ataatb[i,j]:=0;cu[i,j]:=0;pcu[i,j]:=0;
{writeln('input nd,(nd*(nd-1))/2');}
readln(me3,nd,q,chk);
```

```
for i:=1 to nd+chk do
begin
{writeln('input e[i],n[i],h[i]');}
readln(me3,e[i],n[i],h[i]);
end;
k:=0;
for i:= 1 to nd-1 do
begin
for j:=i+1 to nd do
begin
k:=k+1;
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
a[k,1]:=1;
a[k,2]:=-1*d[i,j];
b[k,1]:=1*(ln(h[j]*h[i]));
end;end;
for i:=1 to 2 do
for j:=1 to q do
begin
At[i,j]:=A[j,i];
end;
for i:=1 to 2 do
for j:=1 to 2 do
for k:=1 to q do
  begin
ata[i,j]:=ata[i,j]+at[i,k]*a[k,j];
 end;
 {invers matrix}
 k:=0;
for i:=1 to 2  do
  begin
sum:=ata[i,i];
ata[i,i]:=1;
for j:=1 to 2 do
ata[i,j]:=(ata[i,j])/(sum);
for k:=1 to 2 do
  begin
if(k<>i) then
  begin
sum:=ata[k,i];
ata[k,i]:=0;
for j:=1 to 2 do
ata[k,j]:=(ata[k,j])-sum*ata[i,j];
end; end; end;
for i:=1 to 2 do
for j:=1 to 1 do
```

```
for k:=1 to q do
    begin
atb[i,j]:=atb[i,j]+at[i,k]*b[k,j];
     end;
for i:=1 to 2 do
for j:=1 to 1 do
for k:=1 to 2 do
    begin
ataatb[i,j]:=ataatb[i,j]+ata[i,k]*atb[k,j];
end;
ataatb[1,1]:=exp(ataatb[1,1]);
for i:= 1 to 2 do
begin
for j:=1 to 1 do
begin
writeln(ataatb[i,j]:6:3,' ');
end;end;
for i:=1 to nd do
begin
for j:=1 to nd do
begin
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
c[i,j]:=(ataatb[1,1])*exp(-1*ataatb[2,1]*d[i,j]);
c[j,i]:=(ataatb[1,1])*exp(-1*ataatb[2,1]*d[i,j]);
if(i=j)then
begin
c[i,j]:=ataatb[1,1];
end;end;
u[i,1]:=h[i];
end;
for i:=1 to nd  do
  begin
sum:=c[i,i];
c[i,i]:=1;
for j:=1 to nd do
c[i,j]:=(c[i,j])/(sum);
for k:=1 to nd do
  begin
if(k<>i) then
  begin
sum:=c[k,i];
c[k,i]:=0;
for j:=1 to nd do
c[k,j]:=(c[k,j])-sum*c[i,j];
end; end; end;
for i:= 1 to nd do
```

```
begin
for j:=1 to nd do
begin
write(c[i,j]:6:3,' ');
end;writeln;
end;
for i:=1 to nd do
for j:=1 to 1 do
for k:=1 to nd do
    begin
cu[i,j]:=cu[i,j]+c[i,k]*u[k,j];
end;
sum3:=0.0;tot:=0;
for i:= nd+1 to nd+chk do
begin
for j := 1 to nd do
begin
d[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(n[i]-n[j]));
p[i,j]:=(ataatb[1,1])*exp(-1*ataatb[2,1]*d[i,j]);
end;end;
for i:=nd+1 to nd+chk do
begin
pcu[i,1]:=0;
for j:=1 to nd do
begin
pcu[i,1]:=pcu[i,1]+p[i,j]*cu[j,1];
end;
z[i]:=pcu[i,1];
res[i]:=h[i]-z[i];
sum3:=sum3+sqr(res[i]);
tot:=tot+sqrt(sqr(res[i]));
end;
err:=tot/chk;
rms:=sqrt(sum3/(chk-1));
writeln('err=',err:6:2,' ','rms=',rms:6:2);
readln;readln;
end.
```

A.5 Program for criterion matrix of a levelling network

```
*********************************************************
This program computes the st. error of height diff. in a
levelling network, the following inputs were needed:
```

```
 no of observations(m), total no of stations in the
network(n), the no of unknowns(un), the cords of the
points and the stations of the survey lines arranged in
a text file saved
 as sim1.text,and returns the dist. and the st.error of
the height diff. }
*******************************************************
program cri;
type
vec=array[1..34]of real;
mat=array[1..34,1..34]of real;
var
sim1,sim1r:text;
i,m,h,l,j,n,un,k:integer;

cnt1,cnt2,cnt3,cnt4,cnt5,cnt6,cnt7,cnt8,cnt9,cnt10,cnt11
,cnt12,cnt13,cnt14:integer;
tot1,tot2,tot3,tot4,tot5,tot6,tot7,tot8,tot9,tot10,tot11
,tot12,tot13,tot14:real;
r1,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11,r12,r13,r14:real;
sum:real;
a,d,seg,w,at,atw,atwa,segdh,dis:mat;
e,no:vec;
begin
assign(sim1,'sim1.text');
reset(sim1);
assign(sim1r,'sim1r.text');
rewrite(sim1r);
at[i,j]:=0.0;
atw[i,j]:=0.0;
atwa[i,j]:=0.0;
cnt1:=0;cnt2:=0;cnt3:=0;cnt4:=0;cnt5:=0;cnt6:=0;cnt7:=0;
cnt8:=0;
cnt9:=0;cnt10:=0;cnt11:=0;cnt12:=0;cnt13:=0;cnt14:=0;
tot1:=0;tot2:=0;tot3:=0;tot4:=0;tot5:=0;tot6:=0;tot7:=0;
tot8:=0;
tot9:=0;tot10:=0;tot11:=0;tot12:=0;tot13:=0;tot14:=0;
{writeln('input obs,nstn,un');}
readln(sim1,m,n,un);
for i:= 1 to n do
begin
{writeln('input e,no');}
readln(sim1,e[i],no[i]);
end;
for i:=1 to m do
begin
```

```
{writeln('input h,l');}
readln(sim1,h,l);
a[i,h]:=1;
a[i,l]:=-1;
d[h,l]:=sqrt(sqr(e[h]-e[l])+sqr(no[h]-no[l]));
seg[h,l]:=0.01*(sqrt(d[h,l]));
w[i,i]:=1/(sqr(seg[h,l]));
end;
for i:= 1 to m do
    for j:= 1 to n-1 do
begin
a[i,j]:=a[i,j+1];
end;
for i:= 1 to un do
for j:= 1 to m do
begin
at[i,j]:=a[j,i];
end;
for i:= 1 to un do
for j := 1 to m do
for k:= 1 to m do
begin
atw[i,j]:=atw[i,j]+at[i,k]*w[k,j];
end;
for i:= 1 to un do
for j := 1 to un do
for k:= 1 to m do
begin
atwa[i,j]:=atwa[i,j]+atw[i,k]*a[k,j];
end;
writeln(sim1r,'****************************************
******');
writeln(sim1r,'line','   ','d[i,j]','   ','segdh(mm)');
writeln(sim1r,'****************************************
******');
{invers matrix}
FOR i:=1 to un  do
  begin
SUM:=AtWA[i,i];
AtWA[i,i]:=1;
FOR j:=1 to un do
AtWA[i,j]:=(AtWA[i,j])/(SUM);
FOR k:=1 to un do
  begin
IF(k<>i) THEN
  begin
```

```
SUM:=AtWA[k,i];
AtWA[k,i]:=0;
FOR j:=1 to un do
AtWA[k,j]:=(AtWA[k,j])-SUM*AtWA[i,j];
end; end; end;
{WRITELN('ATWAINV=');
FOR i:=1 to un do
  begin
FOR j:=1 to un do
WRITE('   ',AtWA[i,j]:8:6,'  ');
WRITELN; end;
{writeln('*** se of ht diff ***');}
for i:= 1 to un do
begin
e[i]:=e[i+1];
no[i]:=no[i+1];
end;
for i:= 1 to un-1 do
begin
for j:= i+1 to un do
begin
segdh[i,j]:=atwa[i,i]+atwa[j,j]-2*atwa[i,j];
dis[i,j]:=sqrt(sqr(e[i]-e[j])+sqr(no[i]-no[j]));
   {writeln(sim1r,i,'-',j,'   ',dis[i,j]:7:3,'
',segdh[i,j]:10:3);}
if(dis[i,j]>0)and(dis[i,j]<10.001)then
begin
cnt1:=cnt1+1;
tot1:=tot1+segdh[i,j];
end;
if(dis[i,j]>10.002)and(dis[i,j]<20.001)then
begin
cnt2:=cnt2+1;
tot2:=tot2+segdh[i,j];
end;
if(dis[i,j]>20.002)and(dis[i,j]<30.001)then
begin
cnt3:=cnt3+1;
tot3:=tot3+segdh[i,j];
end;
if(dis[i,j]>30.002)and(dis[i,j]<40.001)then
begin
cnt4:=cnt4+1;
tot4:=tot4+segdh[i,j];
end;
if(dis[i,j]>40.002)and(dis[i,j]<50.001)then
```

```
begin
cnt5:=cnt5+1;
tot5:=tot5+segdh[i,j];
end;
if(dis[i,j]>50.002)and(dis[i,j]<60.001)then
begin
cnt6:=cnt6+1;
tot6:=tot6+segdh[i,j];
end;
if(dis[i,j]>60.002)and(dis[i,j]<70.001)then
begin
cnt7:=cnt7+1;
tot7:=tot7+segdh[i,j];
end;
if(dis[i,j]>70.002)and(dis[i,j]<80.001)then
begin
cnt8:=cnt8+1;
tot8:=tot8+segdh[i,j];
end;
if(dis[i,j]>80.002)and(dis[i,j]<90.001)then
begin
cnt9:=cnt9+1;
tot9:=tot9+segdh[i,j];
end;
if(dis[i,j]>90.002)and(dis[i,j]<100.001)then
begin
cnt10:=cnt10+1;
tot10:=tot10+segdh[i,j];
end;
if(dis[i,j]>100.002)and(dis[i,j]<110.001)then
begin
cnt11:=cnt11+1;
tot11:=tot11+segdh[i,j];
end;
if(dis[i,j]>110.002)and(dis[i,j]<120.001)then
begin
cnt12:=cnt12+1;
tot12:=tot12+segdh[i,j];
end;
if(dis[i,j]>120.002)and(dis[i,j]<130.001)then
begin
cnt13:=cnt13+1;
tot13:=tot13+segdh[i,j];
end;
if(dis[i,j]>130.002)and(dis[i,j]<140.001)then
begin
```

```
cnt14:=cnt14+1;
tot14:=tot14+segdh[i,j];
end;

end;end;
r1:=tot1/cnt1;writeln('5','   ',r1:6:4);
r2:=tot2/cnt2;writeln('15','   ',r2:6:4);
r3:=tot3/cnt3;writeln('25','   ',r3:6:4);
r4:=tot4/cnt4;writeln('35','   ',r4:6:4);
r5:=tot5/cnt5;writeln('45','   ',r5:6:4);
r6:=tot6/cnt6;writeln('55','   ',r6:6:4);
r7:=tot7/cnt7;writeln('65','   ',r7:6:4);
r8:=tot8/cnt8;writeln('75','   ',r8:6:4);
r9:=tot9/cnt9;writeln('85','   ',r9:6:4);
r10:=tot10/cnt10;writeln('95','   ',r10:6:4);
{r11:=tot11/cnt11;writeln('105','   ',r11:6:4);}
r12:=tot12/cnt12;writeln('115','   ',r12:6:4);
r13:=tot13/cnt13;writeln('125','   ',r13:6:4);
r14:=tot14/cnt14;writeln('135','   ',r14:6:4);
close(sim1r);
readln;readln;
end.
```

A.6 Program for criterion matrix of a two dimensional network

```
*******************************************************
This program computes the st. error of coord. diff. in a
a 2-d network, the following inputs were needed:
 no of obs(m), total no of stations in the network(n),
the no of unknowns(un), the cords of the points and the
stations of the survey lines arranged in a text file
saved
 as sim1.text,and returns the dist. and the st.error of
the coords diff. }
*******************************************************

program net2;
const
t=206265;
type
vec=array[1..16]of real;
mat=array[1..32,1..32]of real;
var
sim3,sim3r,sim31r:text;
i,m,h,l,j,n,un,k,code,b,e,mid,s,q,g:integer;
```

```
sum:real;
a,d,seg,w,at,atw,atwa,segdh :mat;
de,dn:mat;
ea,no:vec;
begin
assign(sim3,'sim3.text');
reset(sim3);
assign(sim3r,'sim3r.text');
rewrite(sim3r);
assign(sim31r,'sim31r.text');
rewrite(sim31r);
at[i,j]:=0.0;
atw[i,j]:=0.0;
atwa[i,j]:=0.0;
{writeln('input obs,nstn,un');}
readln(sim3,m,n,un);
for i:= 1 to n do
begin
{writeln('input e,no');}
readln(sim3,ea[i],no[i]);
end;
for i:=1 to m do
begin
{writeln('input code');}
readln(sim3,code);
if(code=1)then
begin
{writeln('input b,e,seg');}
readln(sim3,b,e,seg[b,e]);
de[b,e]:=ea[b]-ea[e];dn[b,e]:=no[b]-no[e];

d[b,e]:=sqrt(sqr(de[b,e])+sqr(dn[b,e]));
a[i,b*2-1]:=de[b,e]/d[b,e];
a[i,b*2]:=dn[b,e]/d[b,e];
a[i,e*2-1]:=-1*a[i,b*2-1];
a[i,e*2]:=-1*a[i,b*2];
w[i,i]:=1/sqr(seg[b,e]) ;
end;
if(code=2)then
begin
{writeln('input b,e,seg');}
readln(sim3,b,e,seg[b,e]);
de[b,e]:=ea[e]-ea[b];dn[b,e]:=no[e]-no[b];
d[b,e]:=sqrt(sqr(de[b,e])+sqr(dn[b,e]));
a[i,b*2-1]:=-1*t*dn[b,e]/sqr(d[b,e]);
a[i,b*2]:=t*de[b,e]/sqr(d[b,e]);
```

```
a[i,e*2-1]:=t*dn[b,e]/sqr(d[b,e]);
a[i,e*2]:=-1*t*de[b,e]/sqr(d[b,e]);
w[i,i]:=1/sqr(seg[b,e]) ;
end;
if(code=3)then
begin
{writeln('input b,e,mid,seg');}
readln(sim3,b,e,mid,seg[b,mid]);
de[mid,e]:=ea[mid]-ea[e];dn[mid,e]:=no[mid]-no[e];
d[mid,e]:=sqrt(sqr(de[mid,e])+sqr(dn[mid,e]));
de[mid,b]:=ea[mid]-ea[b];dn[mid,b]:=no[mid]-no[b];
d[mid,b]:=sqrt(sqr(de[mid,b])+sqr(dn[mid,b]));
a[i,b*2-1]:=t*dn[b,e]/sqr(d[e,b]);
a[i,b*2]:=-1*t*de[b,e]/sqr(d[b,e]);
a[i,e*2-1]:=-1*t*dn[mid,e]/sqr(d[mid,e]);
a[i,e*2]:=t*de[mid,e]/sqr(d[mid,e]);
a[i,mid*2-1]:=(dn[mid,e]/sqr(d[mid,e])-
dn[mid,b]/sqr(d[mid,b]))*t;
a[i,mid*2]:=(de[mid,e]/sqr(d[mid,e])-
de[mid,b]/sqr(d[mid,b]))*t;
w[i,i]:=1/sqr(seg[b,mid]) ;
end;end;
for i:= 1 to m do
begin
    for j:= 1 to un do
begin
a[i,j]:=a[i,j+4];
end; end;
for i:= 1 to un do
for j:= 1 to m do
begin
at[i,j]:=a[j,i];
end;
for i:= 1 to un do
for j := 1 to m do
for k:= 1 to m do
begin
atw[i,j]:=atw[i,j]+at[i,k]*w[k,j];
end;
for i:= 1 to un do
for j := 1 to un do
for k:= 1 to m do
begin
atwa[i,j]:=atwa[i,j]+atw[i,k]*a[k,j];
end;
```

```
{writeln(sim3r,'****************************************
*******');
writeln(sim3r,'line','   ','d[i,j]','   ','segdE(m2)');
writeln(sim3r,'****************************************
******');
writeln(sim31r,'****************************************
*******');
writeln(sim31r,'line','   ','d[i,j]','   ','segdN(m2)');
writeln(sim31r,'****************************************
*******');}
{invers matrix}
FOR i:=1 to un  do
  begin
SUM:=AtWA[i,i];
AtWA[i,i]:=1;
FOR j:=1 to un do
AtWA[i,j]:=(AtWA[i,j])/(SUM);
FOR k:=1 to un do
  begin
IF(k<>i) THEN
  begin
SUM:=AtWA[k,i];
AtWA[k,i]:=0;
FOR j:=1 to un do
AtWA[k,j]:=(AtWA[k,j])-SUM*AtWA[i,j];
end; end; end;
{for i:= 1 to un do
begin
for j:=1 to un do
begin
write(sim3r,atwa[i,j]:7:3);
end;writeln(sim3r);end;}

s:= n-2 ;
for i:= 1 to n-2 do
begin
ea[i]:=ea[i+2];
no[i]:=no[i+2];
end;
for i:= 1 to un-2 do
begin
if(i mod 2 =1)then
begin
q:=trunc((i+1)/2);
for j:= q to s-1 do
begin
```

```
k:=2*j+1;
segdh[i,k]:=atwa[i,i]+atwa[k,k]- 2*(atwa[i,k]);
{writeln(atwa[i,i]:5:3,atwa[k,k]:5:3,atwa[i,k]:5:3);
readln;readln;}
g:=trunc((k+1)/2);
d[q,g]:=sqrt(sqr(ea[q]-ea[g])+sqr(no[q]-no[g]));
writeln(sim3r,q,'-',g,'   ',d[q,g]:7:3,'
',segdh[i,k]:6:3);
end;end;
if(i mod 2 =0)then
begin
q:=trunc((i/2)+1);
for j:= q to s do
begin
k:= 2*j;
segdh[i,k]:=atwa[i,i]+atwa[k,k]-2*(atwa[i,k]);
g:=trunc((i/2));
d[g,j]:=sqrt(sqr(ea[g]-ea[j])+sqr(no[g]-no[j]));
writeln(sim31r,{g,'-',j,'   ',}d[g,j]:7:3,'
',segdh[i,k]:6:3);
end;end;
end;
close(sim3r);
close(sim31r);
end.
```

A.7 Program to compute Khatri-Rao product

```
This program is used to compute rao product of a matrix,
the main inputs are the number of observations(n),
number of unknowns(m) and r which equals (trunk(m*(m-
1))/2)arranged in a text file (kh), and the elements of
the matrix. The product will be found in a text file
called(rr).
*****************************************************

program rao;
type
mat=array[1..28,1..28]of real;
var
kh,rr:text;
a,c:mat;
i,j,k,m,n,q,r:integer;
begin
assign(kh,'kh.text');
reset(kh);
assign(rr,'rr.text');
```

```
rewrite(rr);
{writeln('input obs,un,r');}
readln(kh,n,m,r);
{r:=trunc(m*(m-1))/2;}
q:=1;
             {input of matrix a'}
for i:=1 to n do
begin
for j:= 1 to m do
begin
{writeln('input a[',i,',',j,']');}
readln(kh,a[i,j]);
end;end;
             {comp. of the product}
for i:=1 to m do
begin
for k:=i to m do
begin
for j:=1 to n do
begin
c[q,j]:=a[j,i]*a[j,k];
end;
q:=q+1;
end;end;
 {out put}
for i:= 1 to r do
begin
for j:=1 to n do
begin
write(rr,' ',c[i,j]:3:1);
end;
writeln(rr);
end;
readln;readln;
end.
```