```
Needs[“MultivariateStatistics`”];
w = .20;
cond = 1;
replications = 10000;
Vnor = {0,1,2, 3,4,-1}; (* 0 = normal; 1 is g = 0, h =.142; 2 is g = 0, h = 0.225, 3 is g = 0.76, h =
-0.098; 4 is g = 0.225, h = 0.225; -1 is mixed normal*)
Vdelta = {0, 0.2, 0.5, 0.8, 1.5};
Vn = {50, 100, 300};
Vpn = {0.25, 0.5, 0.75};
VVR = {0, 1, 2}; (* Variance ratio*)

r = Table[
0, {Length[Vnor]*Length[Vdelta]*Length[Vn]*Length[Vpn]*
Length[VVR]}];
d = Table[0, {replications}];
drstar = Table[0, {replications}];
dr = Table[0, {replications}];
A = Table[0, {replications}];
Rpb = Table[0, {replications}];
CL = Table[0, {replications}];
cd = Table[0, {replications}];
norpos = 1;
(*Simulation starts*)
While[norpos <= Length[Vnor],
nor = Vnor[[norpos]];
If[nor == 1, g = 0; h = 0.142];
If[nor == 2, g = 0; h = 0.225];
If[nor == 3, g = 0.76; h = -0.098];
If[nor == 4, g = 0.225; h = 0.225];
deltapos = 1;
While[deltapos <= Length[Vdelta],
delta = Vdelta[[deltapos]];
vrpos = 1;
While[vrpos <= Length[VVR],
VR = VVR[[vrpos]];
(* 0 is 1:1 variance; 1 is 1:4 variance’ 2 is 4:1 variance*)

If[VR == 0, v = 1];
If[VR == 1, v = 4];
If[VR == 2, v = 0.25];
npos = 1;
While[npos <= Length[Vn],
n = Vn[[npos]];
npnpos = 1;
While[npnpos <= Length[Vpn],
prop = Vpn[[npnpos]];
```

```
n1 = Round[n*prop];
n2 = n – n1;
rep = 1;
While[rep <= replications,
data1 = RandomReal[NormalDistribution[0, 1], n1];
data2 = RandomReal[NormalDistribution[0, 1], n2];
(*non-normal data*)
If[nor > 0,
If[g == 0, data1 = data1*E^(h*data1^2/2);
data2 = data2*E^(h*data2^2/2);];
If[g != 0, data1 = ((E^(g*data1) – 1)/g)*E^(h*data1^2/2);
data2 = ((E^(g*data2) – 1)/g)*E^(h*data2^2/2);];
];
(* mixed normal data*)
If[nor == -1,
y1 = RandomReal[UniformDistribution[{0, 1}], n1];
y2 = RandomReal[UniformDistribution[{0, 1}], n2];
Table[
If[y1[[i]] > 0.9, data1[[i]] = data1[[i]]*10], {i, n1}];
Table[
If[y2[[i]] > 0.9, data2[[i]] = data2[[i]]*10], {i, n2}];
];
mdelta =
delta*(Sqrt[((n1 – 1) + (n2 – 1)*v^2)/(n1 + n2 – 2)]);
data2 = data2*v + mdelta;
data1 = Sort[data1];
data2 = Sort[data2];
(* robust d*)
g1 = Floor[w*n1];
g2 = Floor[w*n2];
wdata1 = data1;
wdata2 = data2;
Table[wdata1[[i]] = data1[[g1 + 1]], {i, 1, g1}];
Table[wdata1[[i]] = data1[[n1 – g1]], {i, n1 – g1, n1}];
Table[wdata2[[i]] = data2[[g2 + 1]], {i, 1, g2}];
Table[wdata2[[i]] = data2[[n2 – g2]], {i, n2 – g2, n2}];
tdata1 = Table[0, {n1 – 2*g1}];
tdata2 = Table[0, {n2 – 2*g2}];
Table[tdata1[[i]] = data1[[i + g1]], {i, 1, n1 – 2*g1}];
Table[tdata2[[i]] = data2[[i + g2]], {i, 1, n2 – 2*g2}];
tdrstar =
N[((Mean[tdata2] – Mean[tdata1])/
Sqrt[((n1 – 1)*Variance[wdata1] + (n2 – 1)*
Variance[wdata2])/(n1 + n2 – 2)])];
tdr = N[tdrstar*.642];
(* Cohen's d*)
```

```
td = N[(Mean[data2] – Mean[data1])/
Sqrt[((n1 – 1)*Variance[data1] + (n2 – 1)*
Variance[data2])/(n1 + n2 – 2)]];
(* point biserial correlation*)
p1 = n1/(n1 + n2);
p2 = n2/(n1 + n2);
tRpb =
N[(Mean[data2] – Mean[data1])/
Sqrt[(p1*Variance[data1] + p2*Variance[data2])/(p1*
p2) + (Mean[data2] – Mean[data1])^2]];
(* parametric CL effect size*)

tCL = N[CDF[NormalDistribution [0, 1],
N[(Mean[data2] – Mean[data1])/
Sqrt[((n1 – 1)*Variance[data1] + (n2 – 1)*
Variance[data2])/(n1 + n2 – 2)]]]];
(* non-parametric A*)
count = Table[0, {n2}, {n1}];
Table[
If[data2[[i]] > data1[[j]], count[[i, j]] = 1,
If[data2[[i]] == data1[[j]], count[[i, j]] = 0.5]], {i,
n2}, {j, n1}];
tA = N[Total[Total[count]]/(n1*n2)];
(*d converted from A*)
p1 = N[n1/n];
p2 = N[n2/n];
tcd =
InverseCDF[NormalDistribution[0, 1], tA]/
Sqrt[(p1*Variance[data1] +
p2*Variance[data2])/(Variance[data1] +
Variance[data2])];
drstar[[rep]] = tdrstar;
dr[[rep]] = tdr;
d[[rep]] = td;
Rpb[[rep]] = tRpb;
CL[[rep]] = tCL;
A[[rep]] = tA;
cd[[rep]] = tcd;
p1 = N[n1/(n1 + n2)];
p2 = N[n2/(n1 + n2)];
rep++];
TrueRpb = N[delta/Sqrt[delta^2 + 4]];
TrueCL = N[CDF[NormalDistribution [0, 1], N[delta/Sqrt[2]]]];
(* percentage bias*)
If[delta > 0,
dB = N[(Mean[d] – delta)/delta];
```

```
drsB = N[(Mean[drstar] – delta)/delta];
drB = N[(Mean[dr] – delta)/delta];
RpbB = N[(Mean[Rpb] – TrueRpb)/TrueRpb];
AB = N[(Mean[A] – TrueCL)/TrueCL];
CLB = N[(Mean[CL] – TrueCL)/TrueCL];
cdB = N[(Mean[cd] – delta)/delta];
];
If[delta == 0,
dB = N[(Mean[d] – delta)];
drsB = N[(Mean[drstar] – delta)];
drB = N[(Mean[dr] – delta)];
RpbB = N[(Mean[Rpb] – TrueRpb)];
AB = N[(Mean[A] – TrueCL)/TrueCL];
CLB = N[(Mean[CL] – TrueCL)/TrueCL];
cdB = N[(Mean[cd] – delta)];
];
Print["Cond = ", cond, ", pro = ", prop, ", N = ", n, ", VR = ",
v, ", d = ", delta, ", nor = ", nor];
Print[Mean[d], ", ", Mean[dr], ", ", Mean[drstar], ", ",
Mean[Rpb], ", ", Mean[A], ", ", Mean[CL], ", ", Mean[cd], ", ",
delta, ", ", TrueRpb, ", ", TrueCL, ", ", dB, ", ", drB, ", ",
drsB, ", ", RpbB, ", ", AB, ", ", CLB, ", ", cdB];
Print[
"_____"\
];
r[[cond]] = {Mean[d], Mean[dr], Mean[drstar], Mean[Rpb],
Mean[A], Mean[CL], Mean[cd], delta, TrueRpb, TrueCL, dB, drB,
drsB, RpbB, AB, CLB, cdB};
cond++;
npnpos++];
npos++];
vrpos++];
deltapos++];
norpos++];
Export["results.csv", r];
```