Multivariate DL Code in R

This code comes with the usual disclaimer:

"Users of these routines are cautioned that, while due care has been taken and they are believed accurate, they have not been rigorously tested and their use and results are solely the responsibilities of the user".

The function for performing the analysis is called "multi_dl". This calls the function "just_do_two" which is also provided below.

The function "multi_dl"

This function requires a matrix of study estimates (ys) where each row corresponds to a particular study and each column corresponds to a particular treatment effect. It also requires a matrix of *within-study* variances and covariances (varscovs) where each row corresponds to a particular study; each row reads in the "upper triangle" (including the diagonal) of the within-study variance/covariance matrix of the study in question, so that all entries of the matrix "varscovs" are either a variance or a covariance (hence it's name).

For the bivariate example below for example, each row of "varscovs" provides the within-study variance of the first treatment effect, the within-study covariance, and the within-study variance of the second treatment effect.

Missing Estimates of treatment effect

Missing estimates of treatment effect are handled by replacing them with a typical value (such as the mean of those which are observed) and with the very large corresponding within-study variance of 10^12. The covariances of these missing values with observed values should be entered into varscovs as 0.

Note that the function "just_do_two" contains a check to see if either variance is 10^12 and ensures that if this is the case then these studies do not contribute to the calculation of *a* (equation 10 in the paper). 10^12 was used as this means that the missing values contribute negligibly and much larger values than 10^12 resulted in numerical instability when inverting matrices. If examples contain missing values, users are advised to try smaller values for the corresponding within-study variances (10^10 for example) and amend the "check" line in just_do_two, and check that the numerical results are stable.

R-CODE: Cut and paste the commands below into R for “multi_dl” to produce some illustrative results:

```
ys<-matrix(c(1.139434283, 1.446918983, 1.704748092, 0.470003629, 0.85566611,
1.440361582, 0.186585956, 1.504077397, 1.540445041, 1.665007764, 3.218875825,
1.299282984, 0.661398482, 3.283414346, 4.919980926, 1.386294361, 3.218875825,
2.197224577, 2.268683541, -1.145132304),ncol=2)
varscovs<-matrix(c(0.164999999811686, 0.308823529160345, 0.0738636362264944,
0.162499999665475, 0.0838235293119684, 0.137426900570286, 0.0938352427206998,
0.203703703772108, 0.40476190440518, 0.169884169901165, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1.04000000057403, 0.424242423879069, 0.0947580646552244,
0.34583333355377, 1.00729926944171, 0.208333333709766, 2.07999999946466,
0.555555554810304, 0.367816092120049, 0.188311688602409),ncol=3)

multi_dl<-
function (ys, varscovs)
{
n<-nrow(ys)
dim<-ncol(ys)
cols_need<-1:dim
for(i in 2:dim)
cols_need[i]<-cols_need[i-1]+dim+2-i
all<-1:(dim*(dim+1)/2)
others<-setdiff(all, cols_need)
vars<-matrix(ncol=dim, nrow=n)
for(i in 1: dim)
vars[,i]<-varscovs[,cols_need[i]]
covs<-matrix(ncol=(length(all)-dim), nrow=n)
for(i in 1:length(all)-dim)
covs[,i]<-varscovs[,others[i]]
mat<-matrix(0, nrow=dim, ncol=dim)
maty<-matrix(0, nrow=dim, ncol=1)
est_trun<-matrix(0, nrow=dim, ncol=dim)
est_mat<-matrix(0, nrow=dim, ncol=dim)
count<-1
for(i in 1:(dim-1))
for(j in i:(dim-1))
{
est_mat[i,(j+1)]<-just_do_two(ys[,i], ys[,(j+1)], vars[,i], vars[,(j+1)], covs=covs[,count])
count<-count+1
}
est_mat<-est_mat+t(est_mat)
for(i in 1:dim)
{
est_mat[i,i]<-just_do_two(ys[,i], ys[,i], vars[,i],vars[,i],vars[,i])
}
```

```
eig<-eigen(est_mat)
print("The eigenvalues of the untruncated estimated variance matrix are:",
quote=FALSE, max.levels=0)
print(eig$values)
for(i in 1:dim)
est_trun<-est_trun+max(0, eig$values[i])*eig$vectors[,i]%*%t(eig$vectors[,i])
for(k in 1:n)
{
count<-1
{
within<-matrix(0,nrow=dim, ncol=dim)
for(i in 1:dim)
for(j in i:dim)
{
within[i,j]<-varscovs[k,count]
count<-count+1
}
within<-within+t(within)-diag(diag(within))
mat1<-within+est_trun
mat1<-solve(mat1)
mat<-mat+mat1
y<-matrix(ys[k,],nrow=dim,ncol=1)
mat2<-mat1%*%y
maty<-maty+mat2
}
}
cov<-solve(mat)
beta_hat<-cov%*%maty
ses<-matrix(nrow=dim, ncol=1)
print("The estimated between study variance matrix is:", quote=FALSE)
print(est_trun)
for(i in 1:dim)
ses[i,1]<-cov[i,i]^0.5
print("The estimates are:", quote=FALSE)
print(c(beta_hat))
print("With standard errors:", quote=FALSE)
c(ses)
}

just_do_two<-
function (ys1, ys2,vars1, vars2, covs)
{
cross_ses<-(vars1*vars2)^0.5
mean1a<-sum(ys1/cross_ses)/sum(1/cross_ses)
mean2a<-sum(ys2/cross_ses)/sum(1/cross_ses)
Q<-sum((ys1-mean1a)*(ys2-mean2a)/cross_ses)
```

```
rhos<-covs/cross_ses
check<-(1-1*(vars1==10^12 | vars2==10^12))
rhos<-rhos*check
bit1<-sum(rhos)
bit2<-sum(rhos/cross_ses)/sum(1/cross_ses)
aa<-bit1-bit2
bit5<-sum(1/cross_ses)
bit6<-sum(1/(cross_ses^2))/sum(1/cross_ses)
bb<-bit5-bit6
est<-(Q-aa)/bb
est
}
multi_dl(ys, varscovs)
```