**Assignment # 3 Fitting a Bivariate t-Distribution**
**Scott Morgan**

**Solve textbook problem 7.13.3 a), b), e), and f) on pages 180 (Ruppert). (40 pts)**

```
library(mnormt)
data(CRSPday, package = "Ecdat")
Y = CRSPday[ , c(5, 7)]
loglik = function(par)
{
  mu = par[1:2]
  A = matrix(c(par[3], par[4], 0, par[5]), nrow = 2, byrow = T)
  scale_matrix = t(A) %*% A
  df = par[6]
  -sum(log(dmt(Y, mean = mu, S = scale_matrix, df = df)))
}

A = chol(cov(Y)) #Can be unconstrained
start = as.vector(c(apply(Y, 2, mean),A[1, 1], A[1, 2], A[2, 2], 4))
fit_mvt = optim(start, loglik, method = "L-BFGS-B", #allows box constraints
                lower = c(-0.02, -0.02, -0.1, -0.1, -0.1, 2),
                upper = c(0.02, 0.02, 0.1, 0.1, 0.1, 15), hessian = T)
```

***(a)*** ***What does the code* A = chol(cov(Y)) *do?***

Per the instructions on page 180 of the text, when fitting a multivariate distribution, the covariance matrix needs to be positive definite. The code *A* = chol(cov(Y)) performs the Choleski decomposition on a positive-definite matrix (Hermitian). The function returns *A,* which is an upper triangular matrix. The code t(A) can be used to transpose the decomposed matrix into a lower triangular matrix to satisfy $A^TA=\lambda$. The matrix elements of A serve as an input for the unconstrained optimization problem.

***(b)*** ***Find $\theta$ ML, the MLE of $\theta$.***

The MLE of $\theta$, a scaled parameter, is provided below:

```
 fit_mvt$par
[1] 0.0003789236 0.0008317070 0.0126907363 0.0026859493 0.0051011198 4.2618395311
```

The estimated mean vector are the first two figures (0.0003789, 0.0008317) and the Cholesky factors for the scale matrix are the next three numbers in series (0.0126907, 0.0026859, 0.0051011).

Finally, the MLE of θ is the last number in the series (4.2618395).

The following code can also be used to round the MLEs:

```
cat("MLE =", round(fit_mvt$par, digits = 5)) #Rounded
#MLE = 0.00038 0.00083 0.01269 0.00269 0.0051 4.26184
```

***(e) Find the MLE of the covariance matrix of the returns.***

First, we calculate the covariance matrix of Z (or ⋋), which is also the scale matrix for X. Next, apply equation 7.17 to calculate the covariance matrix of the returns (or Σ). “t(A)” is the transpose of matrix A and t(A)%*%A is the matrix product. v is 4.26184 (par[6]) thus the requirement for v>2 is met.

```
#Covariance Matrix of X (Sigma)-------------------------------------------------------

#Setup covariance matrix, extract factors

par = fit_mvt$par
A = matrix(c(par[3:4],0,par[5]),nrow=2,byrow=TRUE)
z = t(A)%*%A #lambda
x = z * par[6]/(par[6]-2) #sigma
print(x)
#              [,1]         [,2]
# [1,] 3.034652e-04 6.422734e-05
# [2,] 6.422734e-05 6.262399e-05
```

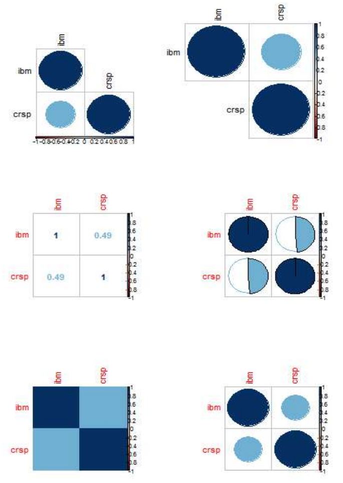
***(f) Find the MLE of $\rho$, the correlation between the two returns ($Y_1$ and $Y_2$).***

Below is the MLE of $\rho$ (Y1 and Y2). Since Σ is proportional to ⋋, X and Z have the same correlation coefficients. If the code par[6]/(par[6]-2) was left out of the equation from part (e), the correlation coefficient would be equal the MLE of $\rho$. This would be incorrect as it would be missing the scale factor.

```
#Equation A.30
corr_x<-x[1,2]/sqrt(x[1,1]*x[2,2])
print(corr_x)
#[1] 0.4659025
```

As an aside, *cor()* can be used with the *corrplot()* package to create great correlation visuals.

```
library(corrplot)
par(mfrow=c(3,2))
M1<-cor(Y)
corrplot(M1, method="circle", type="lower", order = "original", tl.col = "black")
corrplot(M1, type="upper", order = "hclust", tl.col = "black")
corrplot(M1, method="number")
corrplot(M1, method="pie")
corrplot(M1, method="color")
corrplot(M1, method="circle")
```


ibm
crsp