STAT 631 Risk

Jack Cunningham (jgavc@tamu.edu)

11/8/24

```
load("Risk.RData")
```

1)

We use the profile likelihood method to fit the multivariate t distribution.

```
library(MASS)
library(mnormt)

df = seq(1,8, 0.01)
loglik_p = c()
for(i in 1:length(df)){
  fit = cov.trob(y4 , nu = df[i])
  loglik_p[i] = sum(dmt(y4, mean = fit$center, S = fit$cov, df = df[i], log = T))
}

nu = df[which.max(loglik_p)]
cat("The MLE of degrees of freedom:", paste(nu))
```

```
The MLE of degrees of freedom: 4.45
```

Our estimates are:

```
est = cov.trob(y4, nu = nu, cor = T)
names(est)
```

```
[1] "cov"    "center" "n.obs"  "cor"    "call"   "iter"
```

The MLE for the mean vector is:

```r
est$center
```

```
      CPB       CVS         K        PG 
0.1905486 0.3139376 0.2375338 0.2751555 
```

The MLE of the scale matrix Lambda is:

```r
est$cov
```

```
         CPB      CVS        K       PG
CPB 5.378719 1.750477 2.680701 1.633501
CVS 1.750477 6.628764 1.721214 1.710757
K   2.680701 1.721214 3.677831 1.633538
PG  1.633501 1.710757 1.633538 3.232732
```

The MLE of Covariance is:

```r
est$cov*nu/(nu - 2)
```

```
         CPB       CVS        K       PG
CPB 9.769510  3.179437 4.869028 2.966971
CVS 3.179437 12.040000 3.126288 3.107294
K   4.869028  3.126288 6.680142 2.967039
PG  2.966971  3.107294 2.967039 5.871696
```

b)

The tangency portfolio allowing short selling has the following explicit solution:

$$w_t = \frac{\Sigma^{-1}\mu_{ex}}{1^T\mu_{ex}}$$

Where Σ is the covariance matrix:

```r
y.4.S = est$cov*nu/(nu - 2)
y.4.S
```

```
          CPB       CVS        K       PG
CPB 9.769510  3.179437 4.869028 2.966971
CVS 3.179437 12.040000 3.126288 3.107294
K   4.869028  3.126288 6.680142 2.967039
PG  2.966971  3.107294 2.967039 5.871696
```

And mu.ex is the excess return, taking the MLE of the mean vector and subtracting by the risk free rate:

```
mu.f = 3.5/52
m.ex = est$center - mu.f
ones = rep(1,4)
```

We can now calculate w_t:

```
IS = solve(y.4.S)
aT = as.numeric((t(ones)%*%IS%*%m.ex))
w4.T = 1/aT*(IS%*%m.ex)
mu4.T = as.numeric(t(w4.T)%*%est$center)
s4.T = sqrt(as.numeric(t(w4.T)%*%y.4.S%*%w4.T))
cat("Tangency Portfolio for y4:"); w4.T[,1];
```

```
Tangency Portfolio for y4:

       CPB        CVS          K         PG
-0.1158143  0.2711458  0.2758663  0.5688021
```

```
cat("Portfolio return is:", mu4.T, "\t with risk", s4.T)
```

```
Portfolio return is: 0.2850912   with risk 2.209045
```

c)

If we choose to have 20% of assets on the risk free asset and 80% on risky assets we choose a point on the tangency line.

```
w4.c = c(w4.T*.8,.2)
names(w4.c) = c(syb4, "RF")
w4.c
```

```
       CPB        CVS          K         PG         RF
-0.0926514  0.2169166  0.2206931  0.4550417  0.2000000
```

Then we can compute the expected return and risk easily, through:

$$\mu_p = \mu_f + w(\hat{\mu_T} - \mu_f), \hat{\sigma}_p = w\hat{\sigma}_T$$

```
mu.4.c = .8*mu4.T + .2*mu.f
S.4.c = .8*s4.T
cat("Portfolio for y4, 20% in RF:"); w4.c;
```

```
Portfolio for y4, 20% in RF:

       CPB        CVS          K         PG         RF
-0.0926514  0.2169166  0.2206931  0.4550417  0.2000000
```

```
cat("Portfolio return is:", mu.4.c, "\t with risk", S.4.c)
```

```
Portfolio return is: 0.2415345   with risk 1.767236
```

We use the following to find the distribution of the portfolio:

$$w^T Y \sim t_\nu(w^T \mu, w^T \Lambda w)$$

We need to take the risk from our last step, square it and transition it to the scale parameter:

```
scale.c = sqrt(S.4.c^2 * (nu - 2)/nu)
```

So the distribution of the portfolio is: $t_{\hat{\nu}}(.2415, 1.311287)$ where $\hat{\nu} = 4.45$.

d)

Using this distribution we can calculate the one-week VaR and ES with S = 50000.

```
S = 50000
alpha = c(.05,0.01)
q.t = qt(alpha, df = nu); VaR.t = -S*(mu.4.c + scale.c*q.t);
ES.t = S*(-mu.4.c + scale.c*dt(q.t, nu)/alpha*(nu+q.t^2)/(nu-1))
VaR.t = VaR.t/100; ES.t = ES.t/100
output <- rbind(alpha, VaR.t, ES.t)
output
```

```
          [,1]     [,2]
alpha    0.050    0.010
VaR.t 1237.325 2204.151
ES.t  1870.834 3032.390
```

2)

First we fit a t-distribution for each return series:

```r
n = dim(y8)[1];N = dim(y8)[2]
nu = c(); mu = c(); lambda = c()
for(i in 1:N){
  est = fitdistr(y8[, i], "t")$est
  nu[i] = est["df"]
  mu[i] = est["m"]
  lambda[i] = est["s"]
}
```

```
Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced
```

```
Warning in log(s): NaNs produced

Warning in log(s): NaNs produced
```

```
Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced
```

```
Warning in log(s): NaNs produced

Warning in log(s): NaNs produced
```

```
Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in log(s): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in log(s): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

Warning in log(s): NaNs produced

Warning in log(s): NaNs produced
```

```r
stat = cbind(nu, mu, lambda)
colnames(stat) = c("DF", "Mu", "Lambda")
rownames(stat) = syb8
stat
```

```
           DF        Mu   Lambda
AMZN 5.857536 0.4242009 3.428941
KO   3.875997 0.3136914 1.750172
NKE  4.625711 0.2389277 2.764757
PFE  4.297298 0.2632197 2.238336
TSLA 6.189865 0.7540555 6.153071
UNH  3.874175 0.5163419 2.446218
URI  3.512471 0.5951330 4.501945
V    3.550944 0.3784664 2.143774
```

```
alpha = 0.05
S = 50000
q.t = qt(alpha,df = nu); VaR.t = -S*(mu +lambda*q.t);
ES.t = S*(-mu + lambda*dt(q.t,nu)/alpha*(nu + q.t^2)/(nu-1))
VaR.t = VaR.t/100;ES.t = ES.t/100
stat_2 <- cbind(VaR.t, ES.t)
colnames(stat_2) <- c("VaR.t 0.05", "ES.t 0.05")
rownames(stat_2) <- syb8
stat_2
```

```
     VaR.t 0.05 ES.t 0.05
AMZN   3133.993  4470.886
KO     1725.984  2694.577
NKE    2716.666  4007.210
PFE    2207.640  3324.576
TSLA   5568.555  7883.577
UNH    2373.833  3728.315
URI    4699.083  7483.118
V      2181.477  3490.190
```

b)

We use the explicit solution below again:

$$w_t = \frac{\Sigma^{-1}\mu_{ex}}{1^T\mu_{ex}}$$

The means and covariance matrix of y8 are below:

```
y.mu = apply(y8,2,mean);y.mu
```

```
     AMZN        KO       NKE       PFE      TSLA       UNH       URI         V
0.4127509 0.1462473 0.2440613 0.2279685 0.8167512 0.4594456 0.4036380 0.3907812
```

```
y.S = var(y8); y.S
```

```
          AMZN       KO       NKE      PFE      TSLA      UNH       URI
AMZN 17.683202 2.944266  6.135668 3.203864 11.789976  4.070166  9.415175
KO    2.944266 6.669218  4.117950 3.093324  4.961146  4.300771  6.263668
NKE   6.135668 4.117950 13.426979 3.229095 11.152946  5.169045 11.387796
PFE   3.203864 3.093324  3.229095 8.814662  4.198274  4.712427  6.892252
```

```
TSLA 11.789976 4.961146 11.152946 4.198274 55.580284  7.630142 17.354382
UNH   4.070166 4.300771  5.169045 4.712427  7.630142 12.373849  9.676835
URI   9.415175 6.263668 11.387796 6.892252 17.354382  9.676835 41.942499
V     6.261562 4.187155  5.939110 3.568897  7.630532  5.042125  9.798622
            V
AMZN 6.261562
KO   4.187155
NKE  5.939110
PFE  3.568897
TSLA 7.630532
UNH  5.042125
URI  9.798622
V    9.816714
```

We create m.ex by subtracting the risk free rate, which is 3.5/52 in this case. We then we the explicit solution for w_t from before:

```r
m.ex = y.mu - mu.f
ones <- rep(1,N)
IS = solve(y.S)
aT = as.numeric((t(ones)%*%IS%*%m.ex))
w8.T = 1/aT*(IS%*%m.ex)[,1]
w8.T
```

```
       AMZN          KO         NKE         PFE        TSLA         UNH
 0.19114775 -0.68625472 -0.29800916  0.01445672  0.29305575  0.81957899
        URI           V
-0.12001816  0.78604281
```

c)

```r
alpha = 0.05
S = 500000

Rho = cor(y8, method = "s")
w = w8.T*VaR.t
VaR = sqrt(t(w)%*%Rho%*%w)
w = w8.T*ES.t
ES = sqrt(t(w)%*%Rho%*%w)
cat("5% VaR and ES of the tangency portfolio:"); c(VaR = VaR, ES = ES)
```

```
5% VaR and ES of the tangency portfolio:
```

```
      VaR       ES 
3252.052 4957.396 
```

3)

a)

```
S = 50000
alpha = c(0.05, 0.01)
n = length(y1)
```

The Nonparametric estimate of VaR uses the sample quantile $\hat{q}(\alpha)$:

$$\hat{\text{VaR}}^{\text{np}}(\alpha) = -S\hat{q}(\hat{\alpha})$$

And the Nonparametric estimate of Expected Shortfall is:

$$\hat{\text{ES}}^{\text{np}}(\alpha) = -S\frac{\sum_{i=1}^{n} R_i 1\{R_i < \hat{q}\}}{\sum_{i=1}^{n} 1\{R_i < \hat{q}(\alpha)\}}$$

```
q = quantile(y1, alpha)
VaR.np = -S*q; ES.np = c(-S*mean(y1[y1 < q[1]]),-S*mean(y1[y1 < q[2]]))
VaR.np = VaR.np/100; ES.np = ES.np/100
stats <- rbind(q,VaR.np,ES.np)
colnames(stats) <- c("5%","1%")
stats
```

```
                 5%          1%
q         -2.461283   -4.660405
VaR.np 1230.641348 2330.202253
ES.np  1981.443896 3467.810324
```

b)

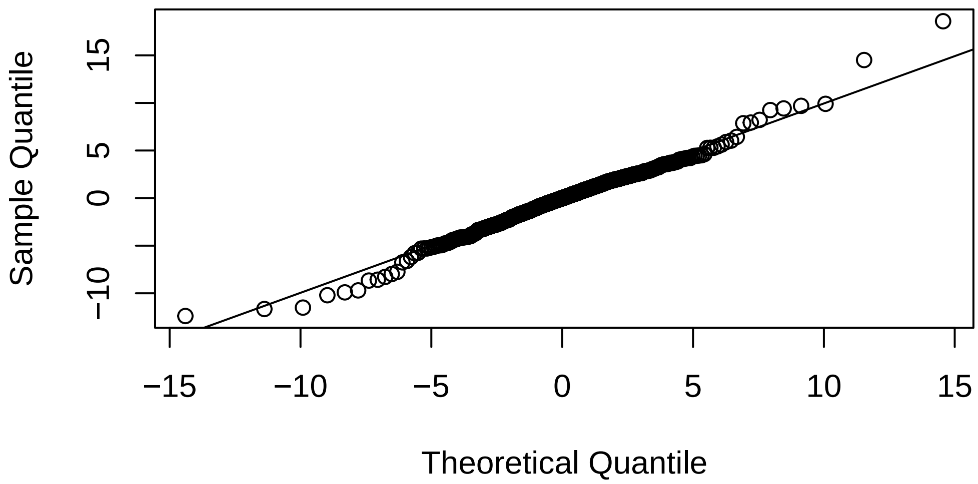
```
fit.t <- fitdistr(y1, "t"); fit.t$est;
```

```
         m          s         df 
0.07823946 0.98515612 3.00596300 
```

```
mu = fit.t$est["m"];lambda = fit.t$est["s"];nu = fit.t$est["df"];

q.t = qt(alpha, df = nu); VaR.t = -S*(mu + lambda*q.t);
ES.t = S*(-mu + lambda*dt(q.t,nu)/alpha * (nu + q.t^2)/(nu-1));
VaR.t = VaR.t/100; ES.t = ES.t/100



stats_4 <- rbind(mu + lambda*q.t, VaR.t, ES.t)
colnames(stats_4) <- c("5%", "1%")
stats_4
```

```
               5%          1%
        -2.238317   -4.387974
VaR.t 1119.158589 2193.987161
ES.t  1866.189649 3401.928369
```

```
q_grid = (1:n) / (n +1)
theoretical_quantile <- mu + lambda*qt(q_grid, df = nu)
qqplot(x = theoretical_quantile, y = as.numeric(y1),
       xlab = "Theoretical Quantile", ylab = "Sample Quantile")
abline(lm(quantile(y1, c(0.25,.75))~
            c(mu + lambda*qt(.25, df = nu), mu + lambda*qt(.75, df = nu))))
```



The fit is decent but struggles in the tail area.

c)

The Semi-parametric estimates use:

$$\text{VaR}(\alpha) = \text{VaR}(\alpha_0)(\frac{\alpha_0}{\alpha})^{1/\alpha}$$

We need to estimate the tail index, we use the Hill estimator which necessitates:

$$\log(-R_{(k)}) \approx \frac{1}{a}\log(\frac{A}{a}) - \frac{1}{a}\log(\frac{k}{n})$$

We choose from the candidate bandwidths $m = n^s, s = 0.5, 0.55, ..., 0.75, 0.8$

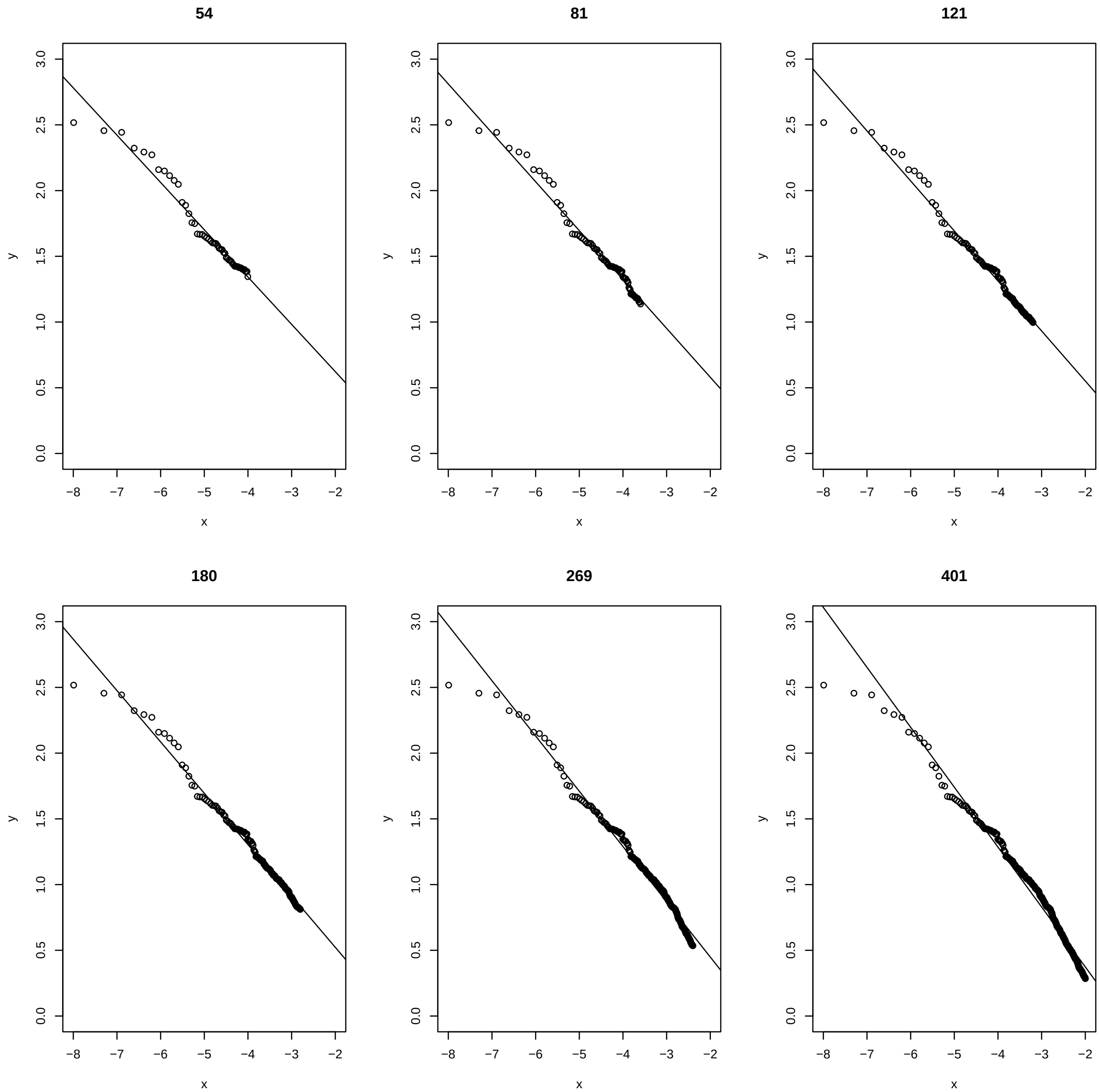
```
y_sort = sort(as.numeric(y1))
s = seq(0.5,0.8,0.05); s
```

```
[1] 0.50 0.55 0.60 0.65 0.70 0.75 0.80
```

```
m = round(n^s); names(m) = paste0("n^",s);m
```

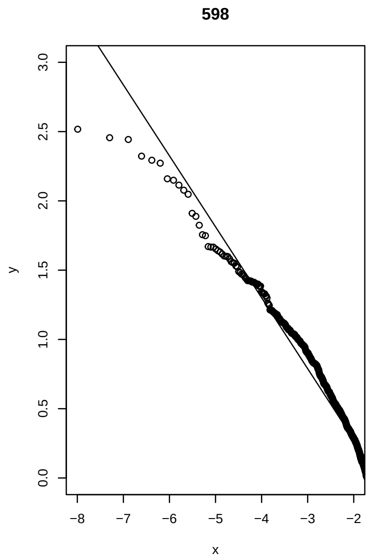
```
 n^0.5 n^0.55  n^0.6 n^0.65  n^0.7 n^0.75  n^0.8
    54     81    121    180    269    401    598
```

```
par(mfrow = c(2,3))
out = matrix(nrow = 3, ncol = length(m))
rownames(out) = c("slope","se","sig.e")
colnames(out) = paste("m", m, sep = " = ")

for(i in 1:length(m)){
  x = log((1:m[i])/n);y = log(-y_sort[1:m[i]])
  lse = lm(y~x)
  plot(x,y, main = m[i], xlim = c(-8,-2),ylim = c(0, 3))
  abline(lsfit(x,y)$coef)
  out[,i] = c(coef(lse)[2], sqrt(vcov(lse)[2,2]),0.01)
}
```



```
out = rbind(out, ahat = -1/out["slope", ]);
round(out,5)
```

```
        m = 54   m = 81  m = 121  m = 180  m = 269  m = 401  m = 598
slope -0.35977 -0.37182 -0.38056 -0.39054 -0.42041 -0.45576 -0.51198
se     0.00992  0.00681  0.00458  0.00324  0.00339  0.00335  0.00408
sig.e  0.01000  0.01000  0.01000  0.01000  0.01000  0.01000  0.01000
ahat   2.77952  2.68944  2.62771  2.56054  2.37862  2.19416  1.95319
```


598
y
x
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−8
−7
−6
−5
−4
−3
−2