```
################################
#   FLOWS IN PIPELINES         #
################################

#Data set of daily flows of natural gas in 3 pipelines.
#Here we will focus on the first two pipelines.

setwd("C:/Users/u0071962/Box Sync/Teaching/KULEUVEN/Statistics for Finance and Insurance/Slides 2017-2018/R
2017-2018/copulas")

library(MASS)
library(fCopulae)
library(copula)
library(sn)
dat=read.csv("FlowData.csv")
dat = dat/10000
n = nrow(dat)

#Use univariate skewed t-model for flows 1 and 2
x1=dat$Flow1
x1s = sort(x1)
x2=dat$Flow2
x2s = sort(x2)

fit1 = st.mple(matrix(1,n,1), y=x1, dp = c(mean(x1),sd(x1),0,10))
est1 = fit1$dp
u1 = pst(x1, dp=est1)

fit2 = st.mple(matrix(1,n,1), y=x2, dp = c(mean(x2),sd(x2),0,10))
est2 = fit2$dp
u2 = pst(x2, dp=est2)

#uniform-transformed flows
U.hat = cbind(u1,u2)

#histograms of both samples of uniform-transformed flows and their scatterplot.
par(mfrow=c(1,2), cex.axis=1.2, cex.lab=1.2, cex.main=1.2)
hist(u1, main="(a)", xlab=expression(hat(U)[1]), freq = FALSE)
hist(u2, main="(b)", xlab=expression(hat(U)[2]), freq = FALSE)
#some deviations from uniform distribution (might be due to random variation).
#maybe skewed t-model is not correct and one can try semi-parametric pseudo ML approach

#scatterplot of uniform-transformed flows
par(mfrow=c(1,1))
plot(u1, u2, main="(c)", xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]), mgp = c(2.5, 1, 0))

#two-dimensional KDE density contours
library(ks)
fhatU = kde(x=U.hat, H=Hscv(x=U.hat))
plot(fhatU, drawpoints=FALSE, drawlabels=FALSE, cont=seq(10, 80, 10),
     main="(d)", xlab=expression(hat(U)[1]), ylab=expression(hat(U)[2]), mgp = c(2.5, 1, 0))


#normal-transformed flows.
z1 = qnorm(u1)
z2 = qnorm(u2)
Z.hat = cbind(z1,z2)

#normal QQ plot
par(mfrow=c(1,2), cex.axis=1.2, cex.lab=1.2, cex.main=1.2)
qqnorm(z1, datax=T, main="(a)") ; qqline(z1)
qqnorm(z2, datax=T, main="(b)") ; qqline(z2)
#normal transformed flows have approximately linear normal quantile plots

#scatterplot and KDE density contours
plot(z1, z2, main="(c)", xlab = expression(hat(Z)[1]), ylab = expression(hat(Z)[2]), mgp = c(2.5, 1, 0))
fhatZ = kde(x=Z.hat, H=Hscv(x=Z.hat))
plot(fhatZ, drawpoints=FALSE, drawlabels=FALSE, cont=seq(10, 90, 10),
     main="(d)", xlab=expression(hat(Z)[1]), ylab=expression(hat(Z)[2]), mgp = c(2.5, 1, 0))
#scatterplot shows negative correlation ==> Gumbel will not fit well (also Clayton not in that region)


#assume that two flows have meta-gaussian distribution

#estimate correlation (3 different ways)
cor(cbind(x1,x2),method="spearman")
sin(pi/2*cor(cbind(x1,x2),method="kendall"))
cor(cbind(z1,z2),method="pearson")
```

```
#reasonably close agreement
options(digits=3)
cor.test(z1, z2, method="pearson")
#both rank correlations belong to 95% CI for Pearson correlation

omega = -0.371


#Parametric copulas were fit to uniform-transformed flows (Gumbel gives error since data shows negative
dependence)
options(digits=4)

Ct = fitCopula(copula=tCopula(dim = 2), data=U.hat, method="ml", start=c(omega, 10))
Ct@estimate

Cgauss = fitCopula(copula=normalCopula(dim = 2), data=U.hat, method="ml", start=c(omega))
Cgauss@estimate

Cfr = fitCopula(copula=frankCopula(1, dim=2), data=U.hat, method="ml")
Cfr@estimate

Ccl = fitCopula(copula=claytonCopula(1, dim=2), data=U.hat, method="ml")
Ccl@estimate

#AIC values
AIC(Ct)
loglikCopula(param=Ct@estimate, u=U.hat, copula=tCopula(dim = 2))
-2*.Last.value + 2*length(Ct@estimate)

AIC(Cgauss)
loglikCopula(param=Cgauss@estimate, u=U.hat, copula=normalCopula(dim = 2))
-2*.Last.value + 2*length(Cgauss@estimate)

AIC(Cfr)
loglikCopula(param=Cfr@estimate, u=U.hat, copula=frankCopula(dim = 2))
-2*.Last.value + 2*length(Cfr@estimate)

AIC(Ccl)
loglikCopula(param=Ccl@estimate, u=U.hat, copula=claytonCopula(dim = 2))
-2*.Last.value + 2*length(Ccl@estimate)

#Frank copula fits best
#tcopula similar to Gaussian copula due to large degrees of freedom


par(mfrow=c(2,3), mgp = c(2.5, 1, 0))
#uniform-transformed scatterplot
plot(u1, u2, main="Uniform-Transformed Data",
     xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))

Udex = (1:n)/(n+1)
#empirical copula
Cn = C.n(u = cbind(rep(Udex, n), rep(Udex, each=n)) , U.hat,
         offset=0, method="C")
EmpCop = expression(contour(Udex,Udex,matrix(Cn,n,n), col=2, add=T))
#
contour(normalCopula(param=0,dim=2), pCopula, main=expression(C[0]),
        xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))
#?`contour,Copula-method`
eval(EmpCop)
#
contour(tCopula(param=Ct@estimate[1], dim=2,
                df=round(Ct@estimate[2])),
        pCopula, main = expression(hat(C)[t]),
        xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))
eval(EmpCop)
#
contour(normalCopula(param=Cgauss@estimate[1], dim = 2),
        pCopula, main = expression(hat(C)[Gauss]),
        xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))
eval(EmpCop)
#
contour(frankCopula(param=Cfr@estimate[1], dim = 2),
        pCopula, main = expression(hat(C)[Fr]),
        xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))
eval(EmpCop)
#
contour(claytonCopula(param=Ccl@estimate[1], dim = 2),
        pCopula, main = expression(hat(C)[Cl]),
        xlab = expression(hat(U)[1]), ylab = expression(hat(U)[2]))
eval(EmpCop)



#  We could also use the flows transformed by their empirical cdfs (and repeat the analysis)
u1 = (rank(x1) - 0.5)/n
u2 = (rank(x2) - 0.5)/n
#This yields semiparametric pseudo-maximum likelihood estimates
#Results are very similar.
```