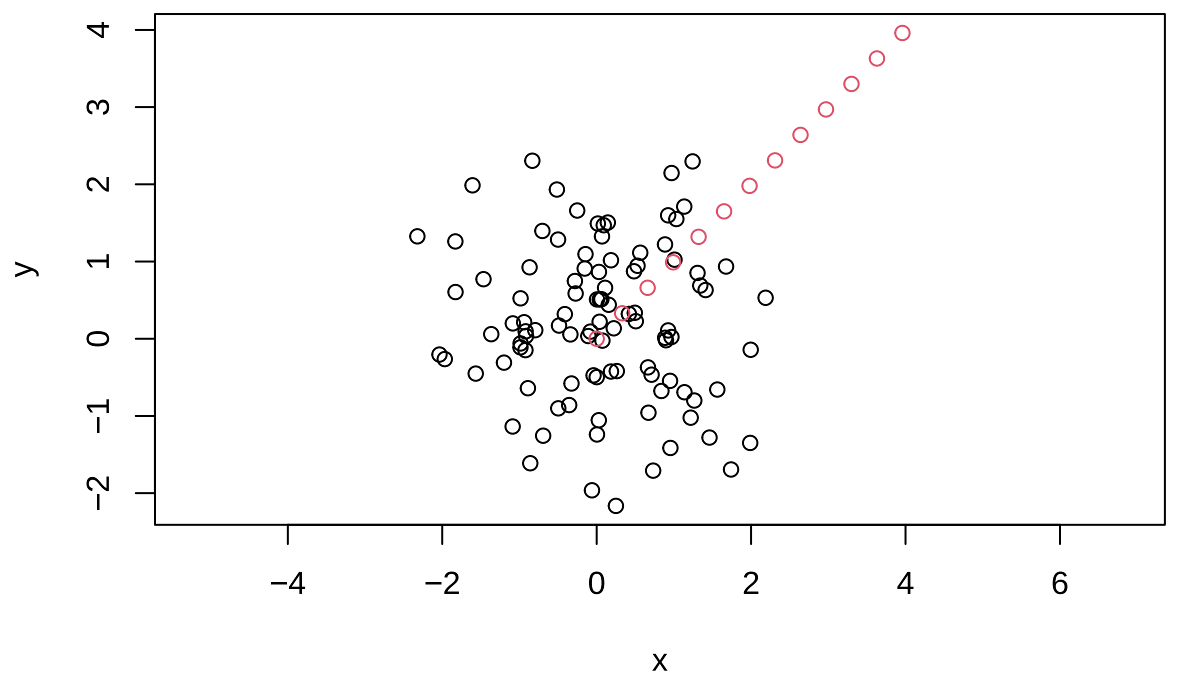
DQF Unsupervised

2022-11-10

Test Dataset

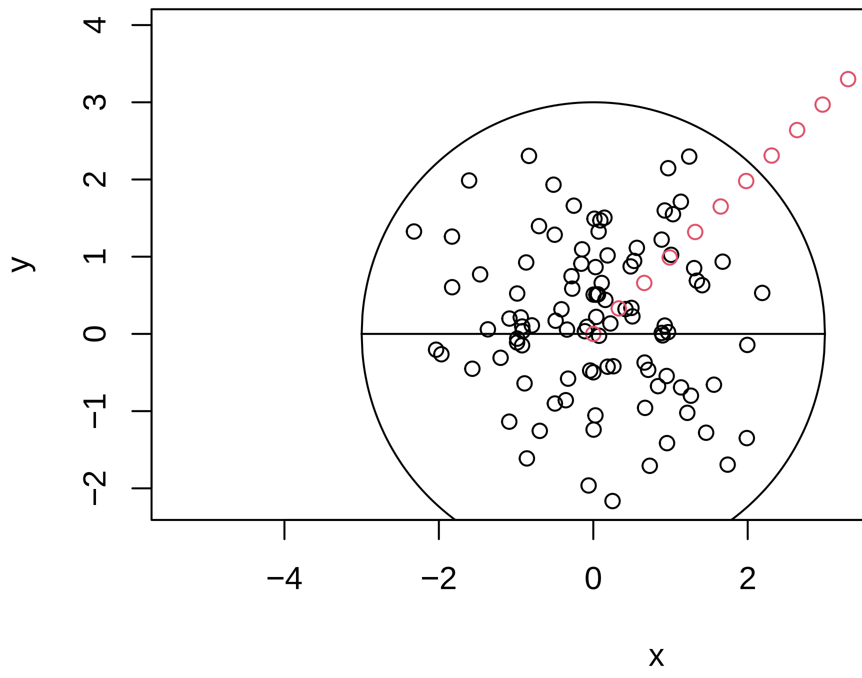
```
set.seed(47)

x <- rnorm(100)
y <- rnorm(100)
data1 <- cbind(x,y)
x.anomaly1 <- seq(0,4,.33)
y.anomaly1 <- seq(0,4,.33)
data1.anomaly <- cbind(x.anomaly1,y.anomaly1)
labels1 <- c(rep(1,length(data1[,1])),rep(2,length(data1.anomaly[,1])))
data1 <- rbind(data1,data1.anomaly)

plot(data1,col=labels1,asp=1)
```



Functions and tests

```
draw.circle <- function(center.x,center.y,radius){
  x <- center.x
  y <- center.y
  r <- radius

  circle.x <- seq(x-r,x+r,.01)
  upper.circle.y <- y+sqrt(r^2-(circle.x-x)^2)
  lower.circle.y <- y-sqrt(r^2-(circle.x-x)^2)

  upper.circle <- cbind(circle.x,upper.circle.y)
  lower.circle <- cbind(circle.x,lower.circle.y)
  circle <- rbind(upper.circle,lower.circle)

  lines(circle)
}
```

```
plot(data1,col=labels1,asp=1)
draw.circle(0,0,3)
```



`draw.circle` draws a circle on existing plot

```
require(dqfAnomaly)
```

`dqf.outlier` Gabe Chandler's DQF Anomaly Github

```
## Loading required package: dqfAnomaly
```

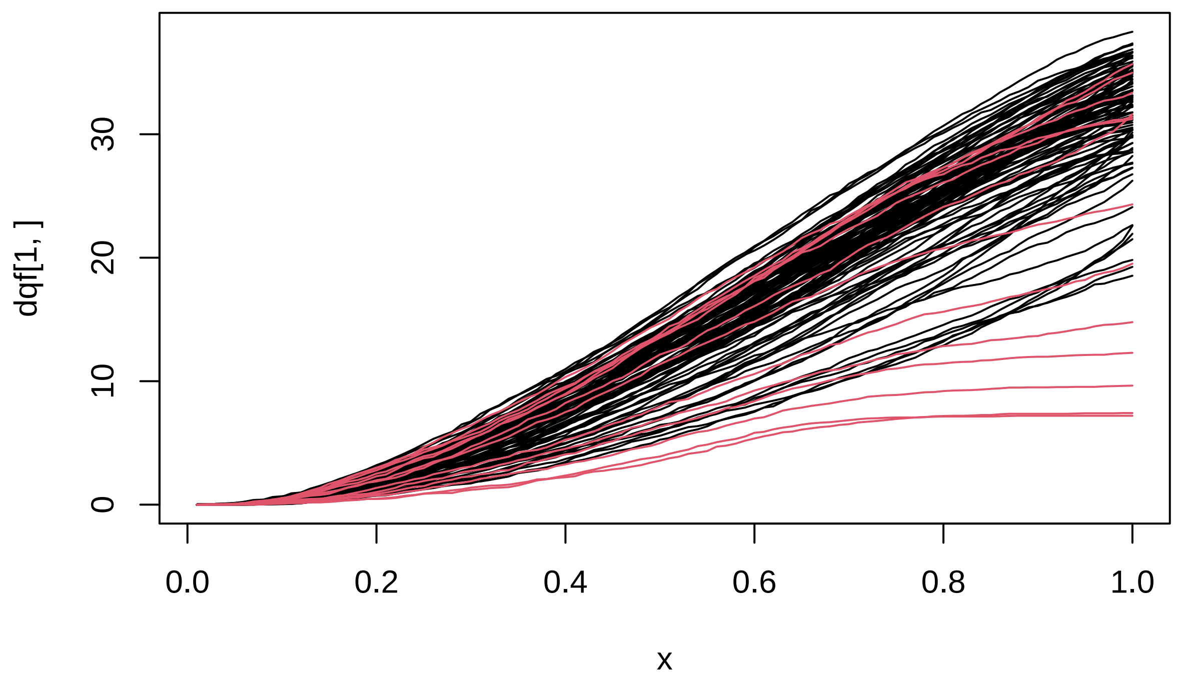
```r
dqfs1 <- dqf.outlier(data1)
```

```r
dqf1.1 <- dqfs1$dqf1
dqf1.2 <- dqfs1$dqf2
dqf1.3 <- dqfs1$dqf3
```

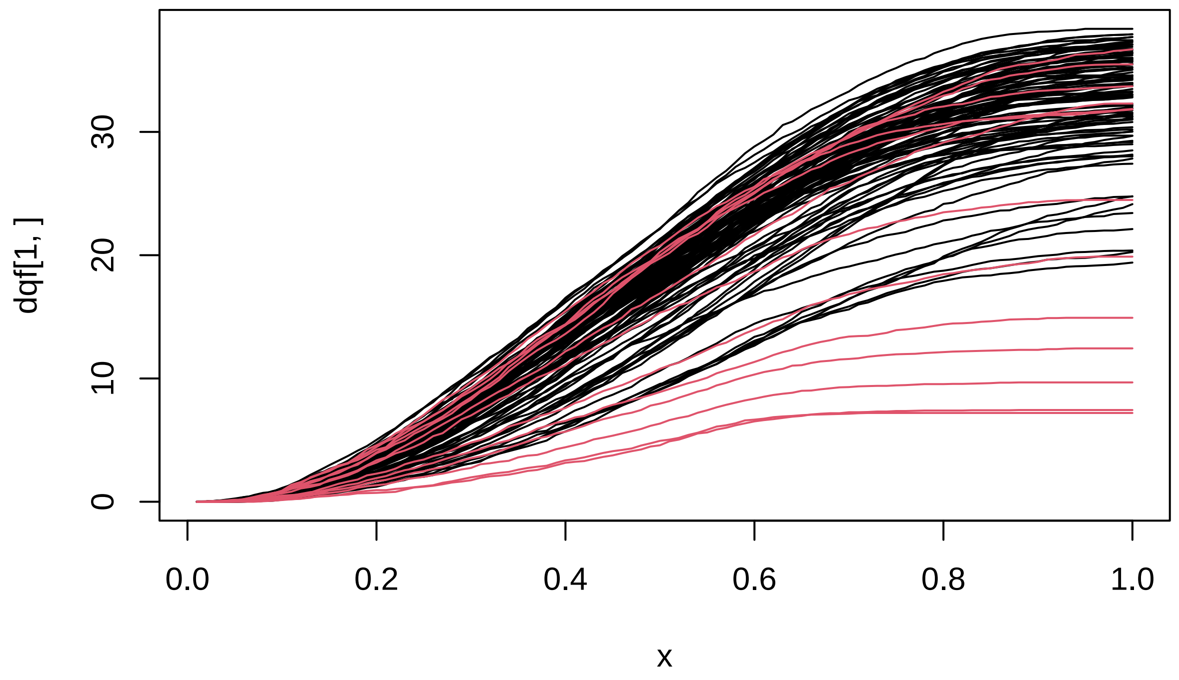
`plot.dqf`: Given a data frame of dqfs, where rows and columns are functions' x and y values respectively, plot them.

```r
plot.dqf <- function(dqf,labels){
  x <- seq(.01,1,.01)

  n.functions <- length(dqf[,1])

  plot(x,dqf[1,],t='l',ylim=c(0,max(dqf)))
  for(i in 2:n.functions){
    lines(x,dqf[i,],col=labels[i])
  }
}
```

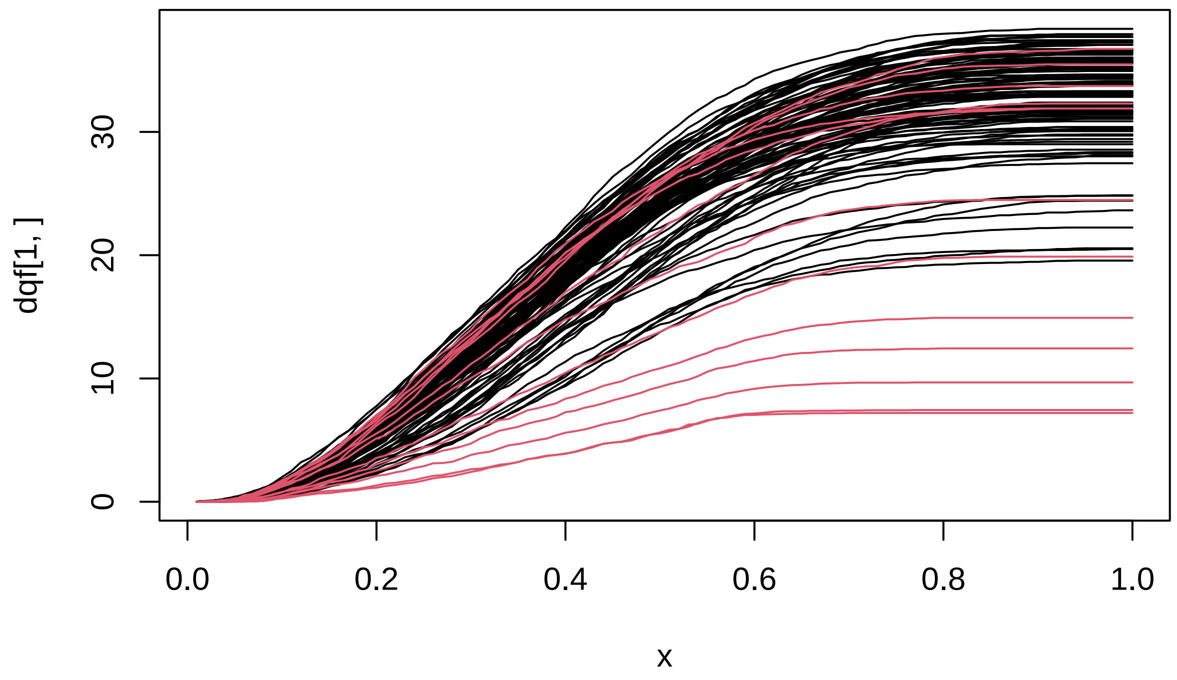
```r
plot.dqf(dqf1.1,labels1)
```



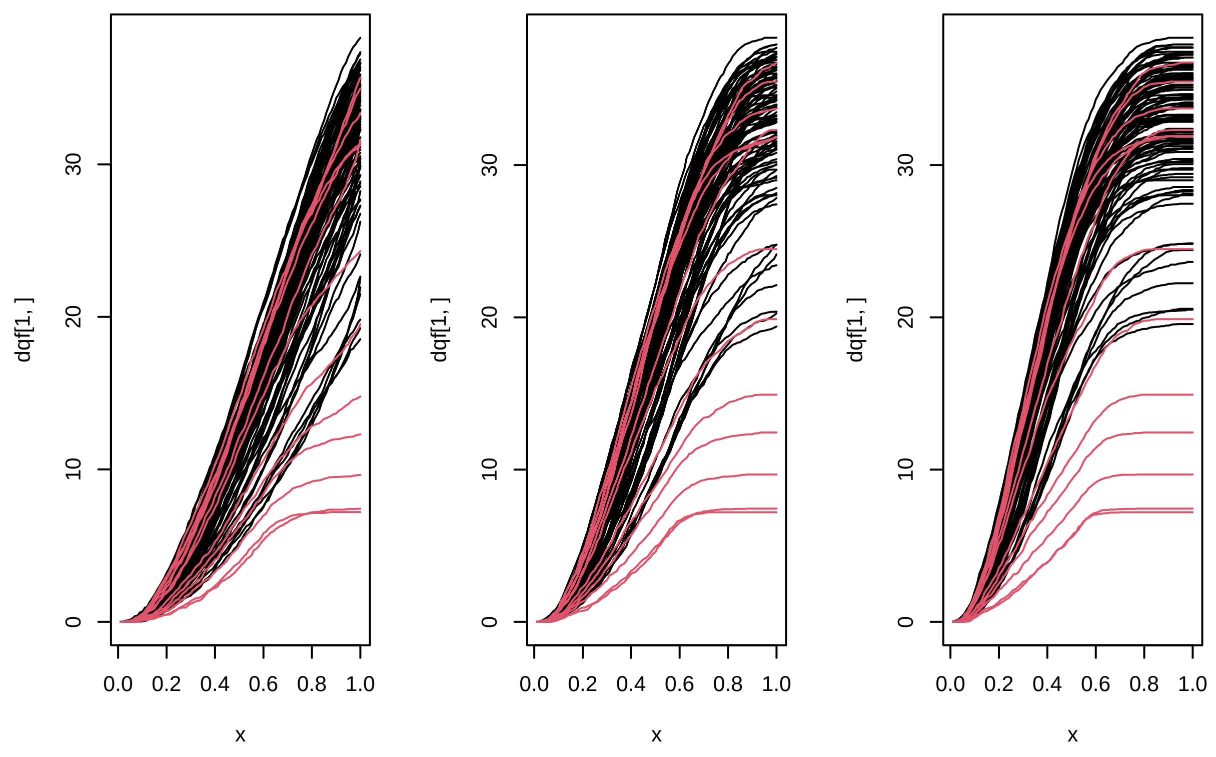
```r
plot.dqf(dqf1.2,labels1)
```



```
plot.dqf(dqf1.3,labels1)
```



```
par(mfrow=c(1,3))
plot.dqf(dqf1.1,labels1)
plot.dqf(dqf1.2,labels1)
plot.dqf(dqf1.3,labels1)
```

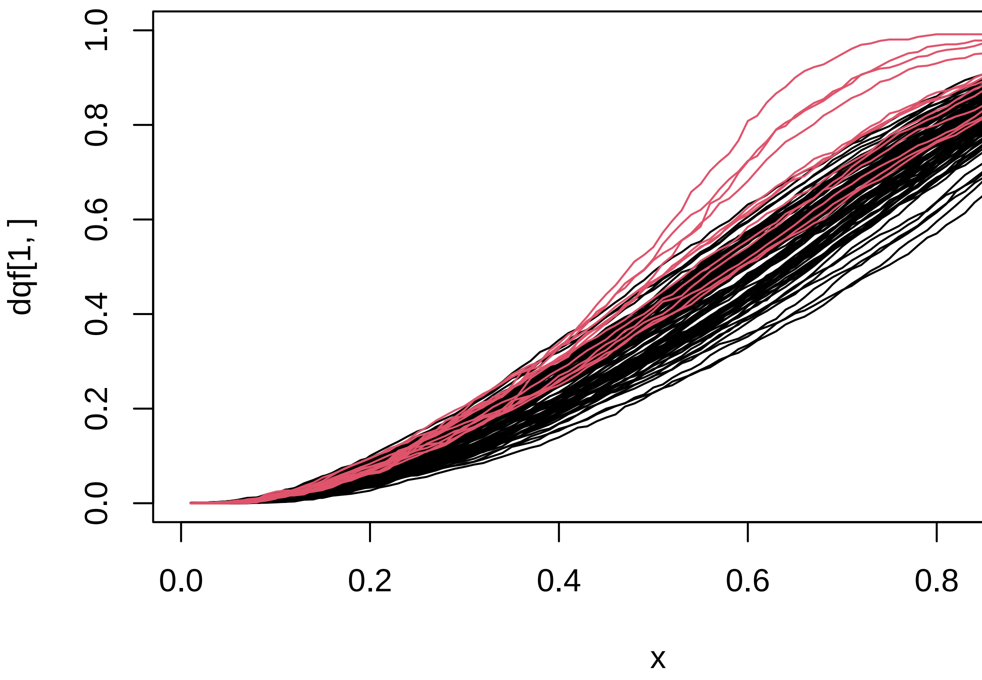


Scale functions (function specific)

```
scale.dqf.max <- function(dqf){
  n.functions <- length(dqf[,1])
  ret <- dqf

  for(i in 1:n.functions){
    func.max <- max(dqf[i,])
    for(j in 1:length(dqf[i,])){
      ret[i,j] <- dqf[i,j]/func.max
    }
  }

  return(ret)
}
```

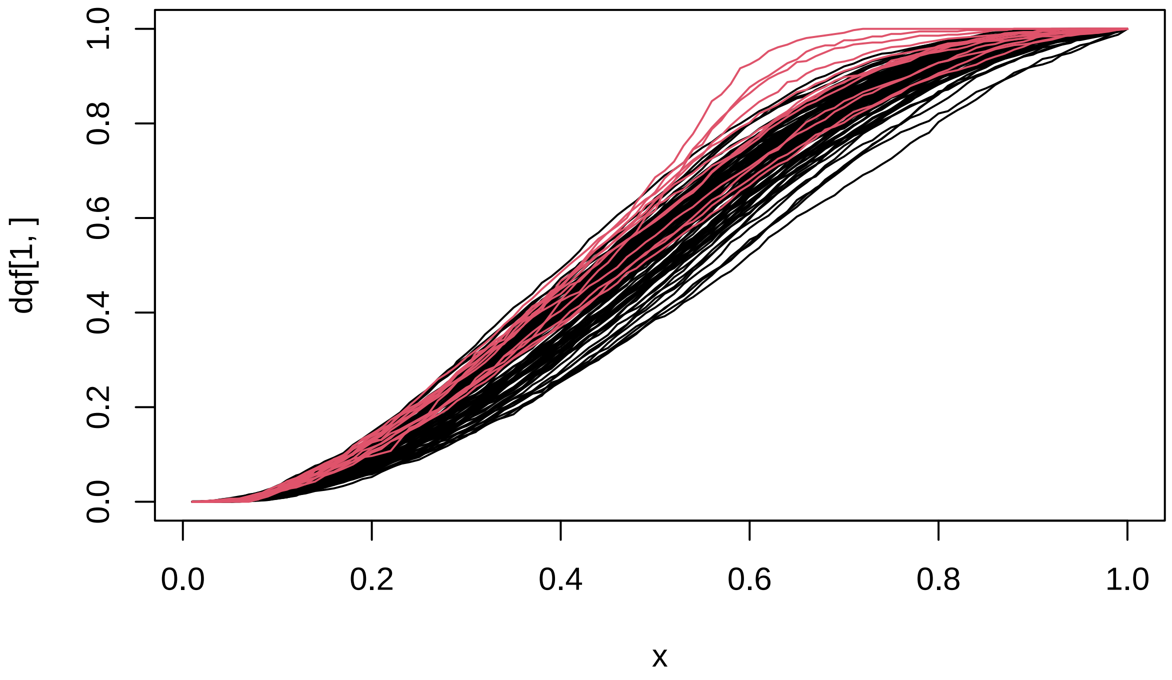
```
plot.dqf(scale.dqf.max(dqf1.1),labels1)
```

`scale.dqf.max`: Given dqfs, where rows and columns are functions' x and y values, scale each

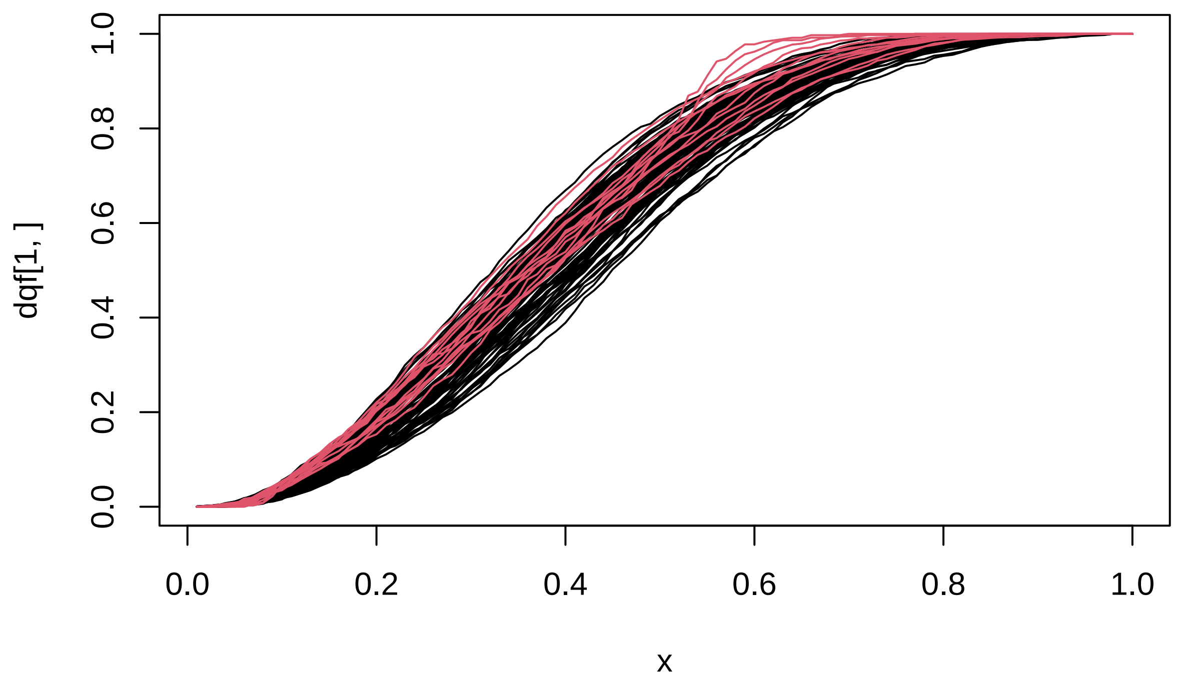


function to have min=0, max=1.

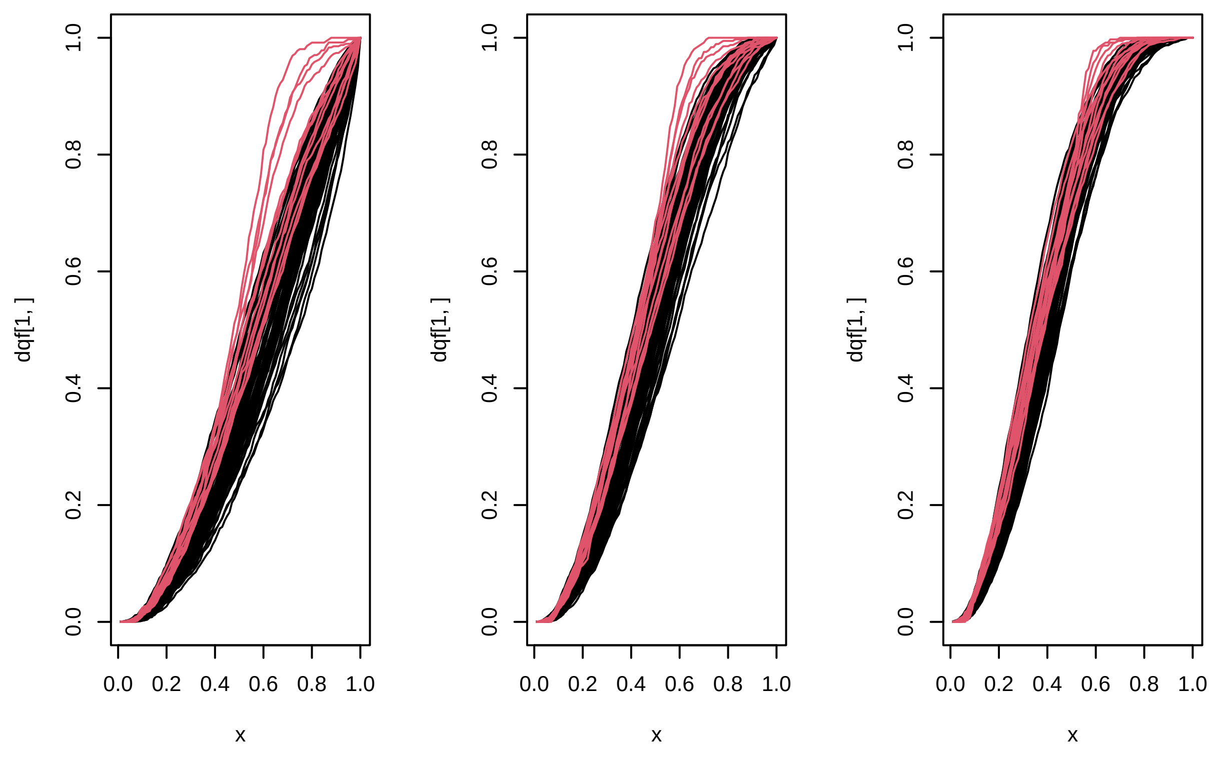
```
plot.dqf(scale.dqf.max(dqf1.2),labels1)
```



```
plot.dqf(scale.dqf.max(dqf1.3),labels1)
```



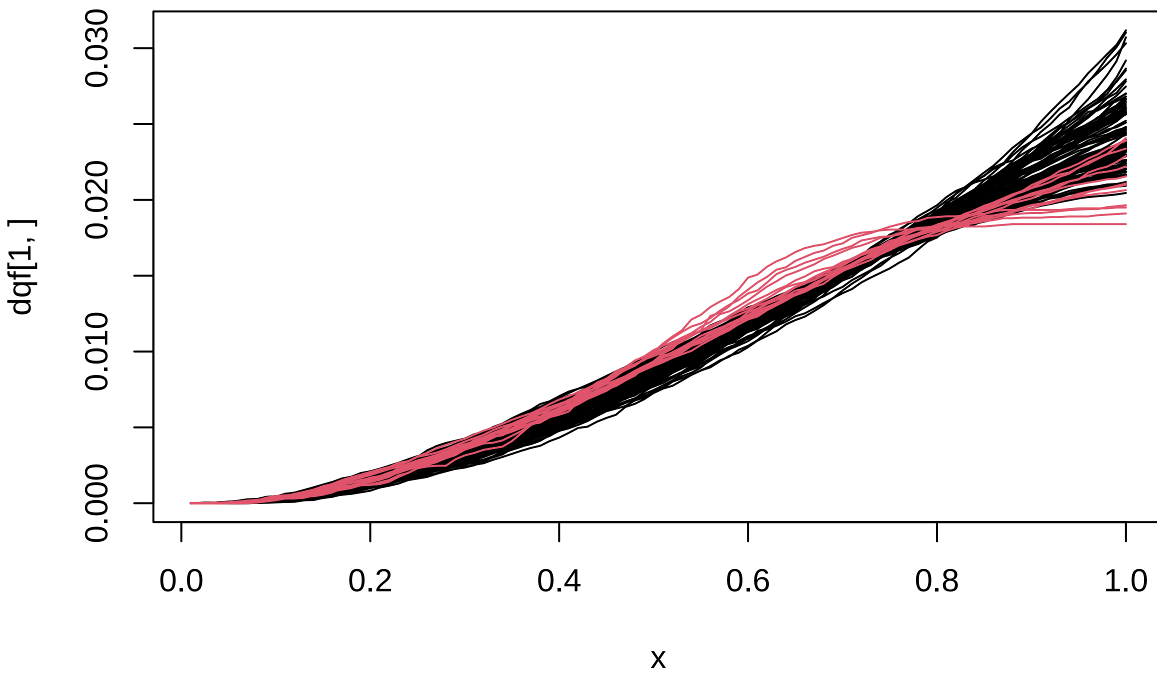
```
par(mfrow=c(1,3))
plot.dqf(scale.dqf.max(dqf1.1),labels1)
plot.dqf(scale.dqf.max(dqf1.2),labels1)
plot.dqf(scale.dqf.max(dqf1.3),labels1)
```



```
scale.dqf.sum <- function(dqf){
  n.functions <- length(dqf[,1])
  ret <- dqf

  for(i in 1:n.functions){
    func.sum <- sum(dqf[i,])
    for(j in 1:length(dqf[i,])){
      ret[i,j] <- dqf[i,j]/func.sum
    }
  }

  return(ret)
}
```

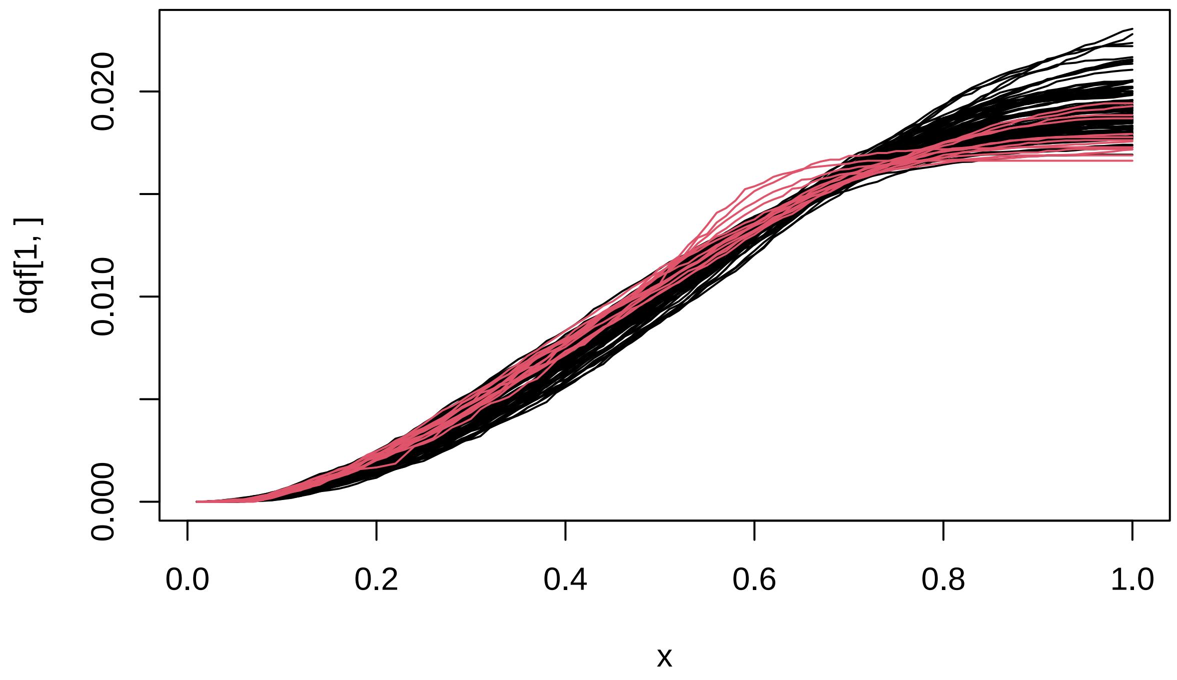
```
plot.dqf(scale.dqf.sum(dqf1.1),labels1)
```

`scale.dqf.sum`: Given dqfs, where rows and columns are functions' x and y values, scale each

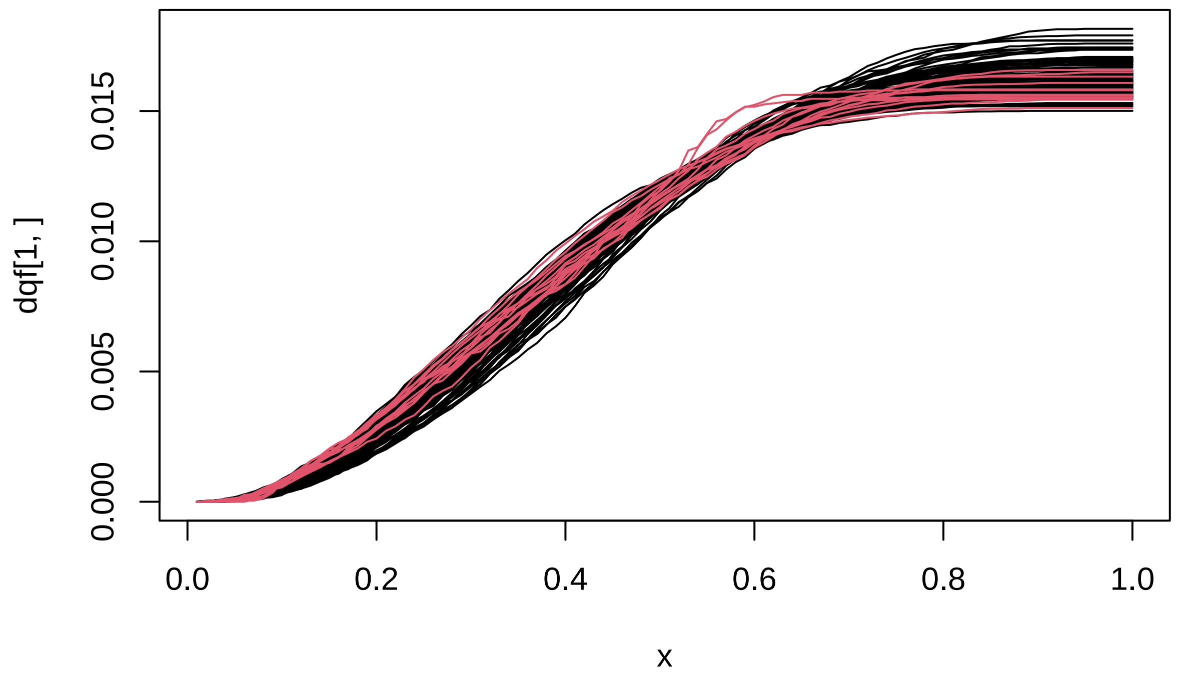


function to its sum.

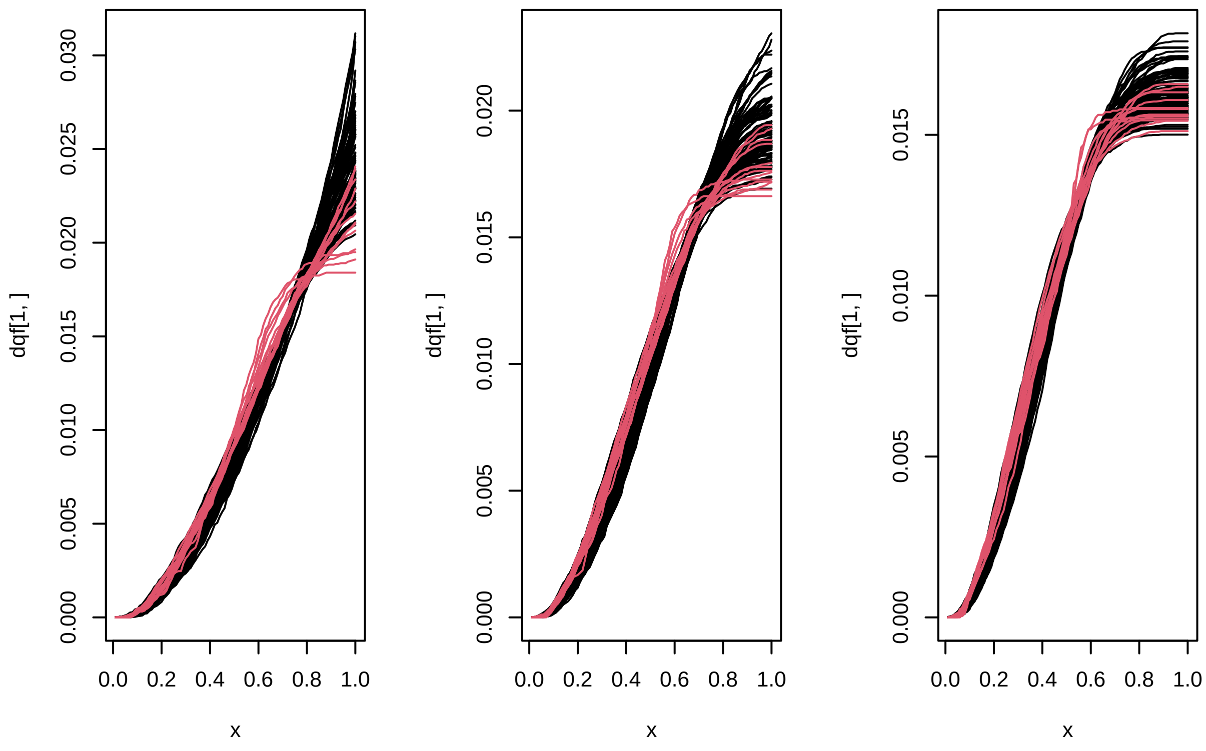
```
plot.dqf(scale.dqf.sum(dqf1.2),labels1)
```



```r
plot.dqf(scale.dqf.sum(dqf1.3),labels1)
```



```
par(mfrow=c(1,3))
plot.dqf(scale.dqf.sum(dqf1.1),labels1)
plot.dqf(scale.dqf.sum(dqf1.2),labels1)
plot.dqf(scale.dqf.sum(dqf1.3),labels1)
```

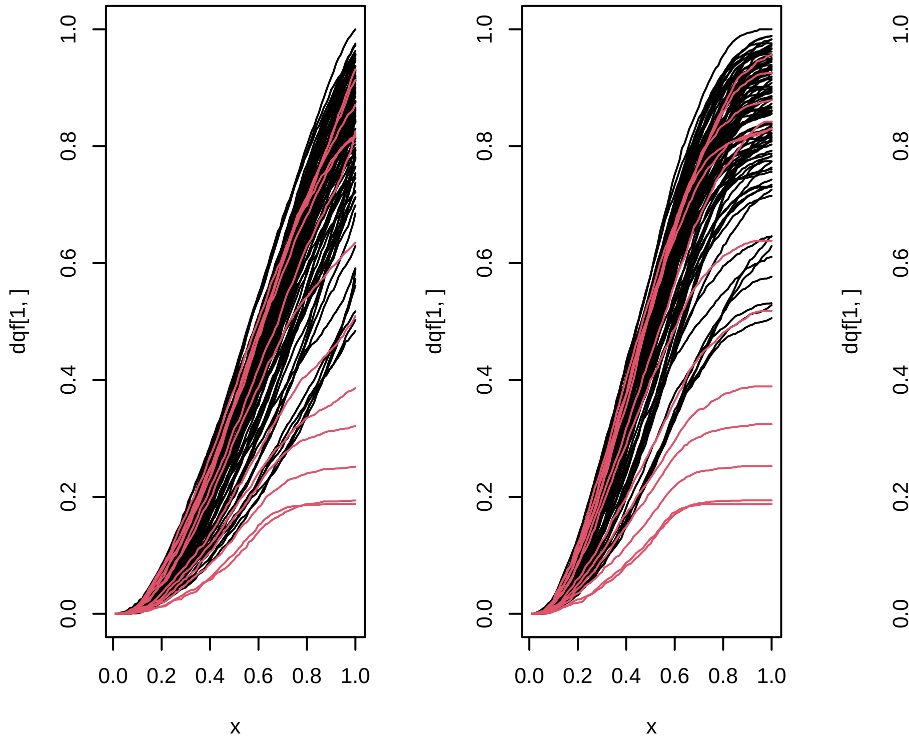


Scale functions (global)

```
scale.dqf.globalmax <- function(dqf){
  n.functions <- length(dqf[,1])
  globalmax <- max(dqf)

  ret <- dqf

  for(i in 1:n.functions){
    for(j in 1:length(dqf[i,])){
      ret[i,j] <- dqf[i,j]/globalmax
    }
  }

  return(ret)
}
```

```
par(mfrow=c(1,3))
plot.dqf(scale.dqf.globalmax(dqf1.1),labels1)
plot.dqf(scale.dqf.globalmax(dqf1.2),labels1)
plot.dqf(scale.dqf.globalmax(dqf1.3),labels1)
```

`scale.dqf.globalmax`: Given dqfs, where rows and columns are functions' x and y values, scale

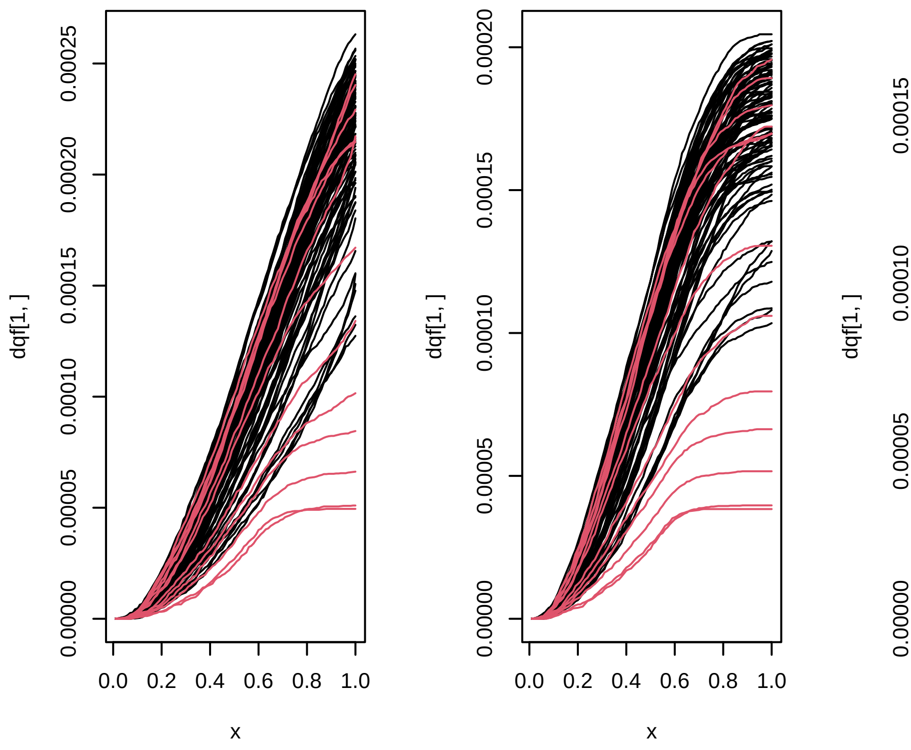


each function to relative to global max.

```
scale.dqf.globalsum <- function(dqf){
  n.functions <- length(dqf[,1])
  globalsum <- sum(dqf)

  ret <- dqf

  for(i in 1:n.functions){

    for(j in 1:length(dqf[i,])){
      ret[i,j] <- dqf[i,j]/globalsum
    }
  }

  return(ret)
}
```

```
par(mfrow=c(1,3))
plot.dqf(scale.dqf.globalsum(dqf1.1),labels1)
plot.dqf(scale.dqf.globalsum(dqf1.2),labels1)
plot.dqf(scale.dqf.globalsum(dqf1.3),labels1)
```

`scale.dqf.globalsum`: Given dqfs, where rows and columns are functions' x and y values, scale



each function to relative to global sum.

Summary DQF (means and standard deviations)

```r
dqf.mean <- function(dqf){
  means <- c()

  for(i in 1:length(dqf[1,])){
    means <- c(means, mean(dqf[,i]))
  }

  return(means)
}
```

```r
dqf.sd <- function(dqf){
  sds <- c()

  for(i in 1:length(dqf[1,])){
    sds <- c(sds, sd(dqf[,i]))
  }

  return(sds)
}
```

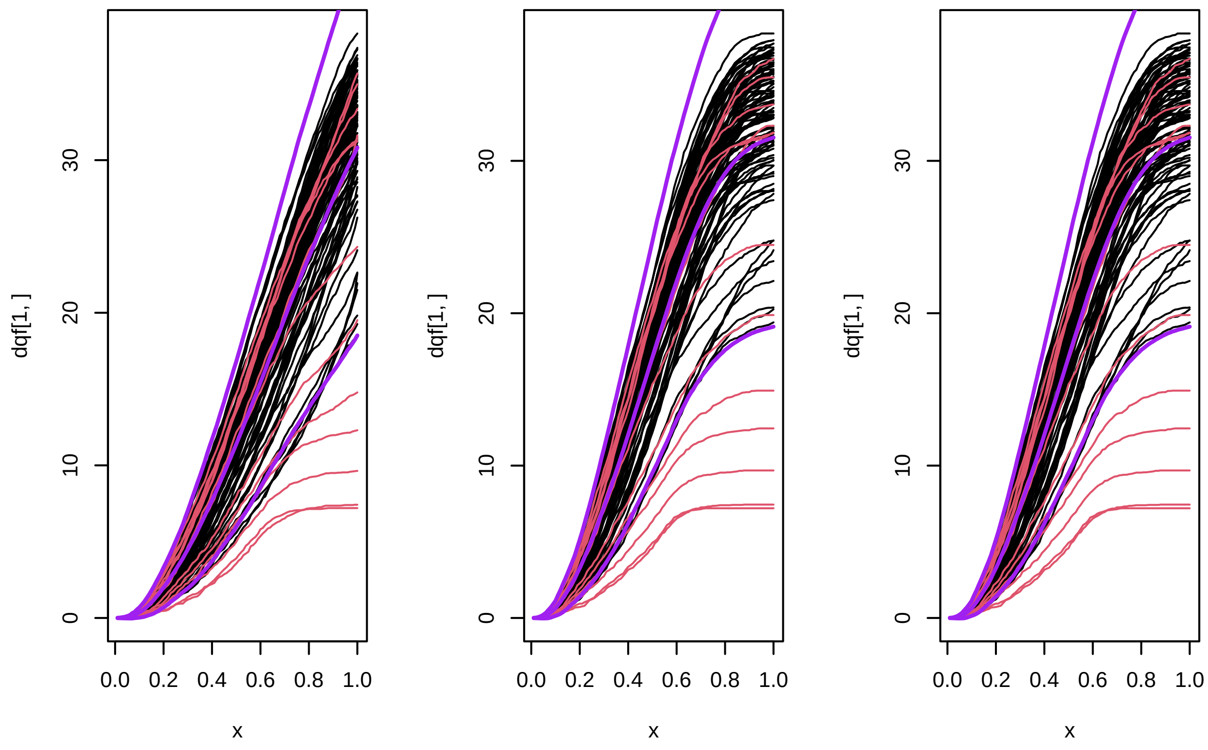
```r
dqf.upperbound <- function(mean,sd,n.sd=2){
  return(mean+n.sd*sd)
}
```

```
dqf.lowerbound <- function(mean,sd,n.sd=2){
  return(mean-n.sd*sd)
}
```

```
draw.mean.bounds <- function(mean, sd, n.sd=2){
  x <- seq(.01,1,.01)
  lines(x, mean, col='purple',lwd=2.0)
  lines(x,dqf.lowerbound(mean,sd),col='purple',lwd=2.0)
  lines(x,dqf.upperbound(mean,sd),col='purple',lwd=2.0)
}
```

test

```
par(mfrow = c(1,3))

plot.dqf(dqf1.1,labels1)
draw.mean.bounds(dqf.mean(dqf1.1),dqf.sd(dqf1.1),2)
plot.dqf(dqf1.2,labels1)
draw.mean.bounds(dqf.mean(dqf1.2),dqf.sd(dqf1.2),2)
plot.dqf(dqf1.2,labels1)
draw.mean.bounds(dqf.mean(dqf1.2),dqf.sd(dqf1.2),2)
```



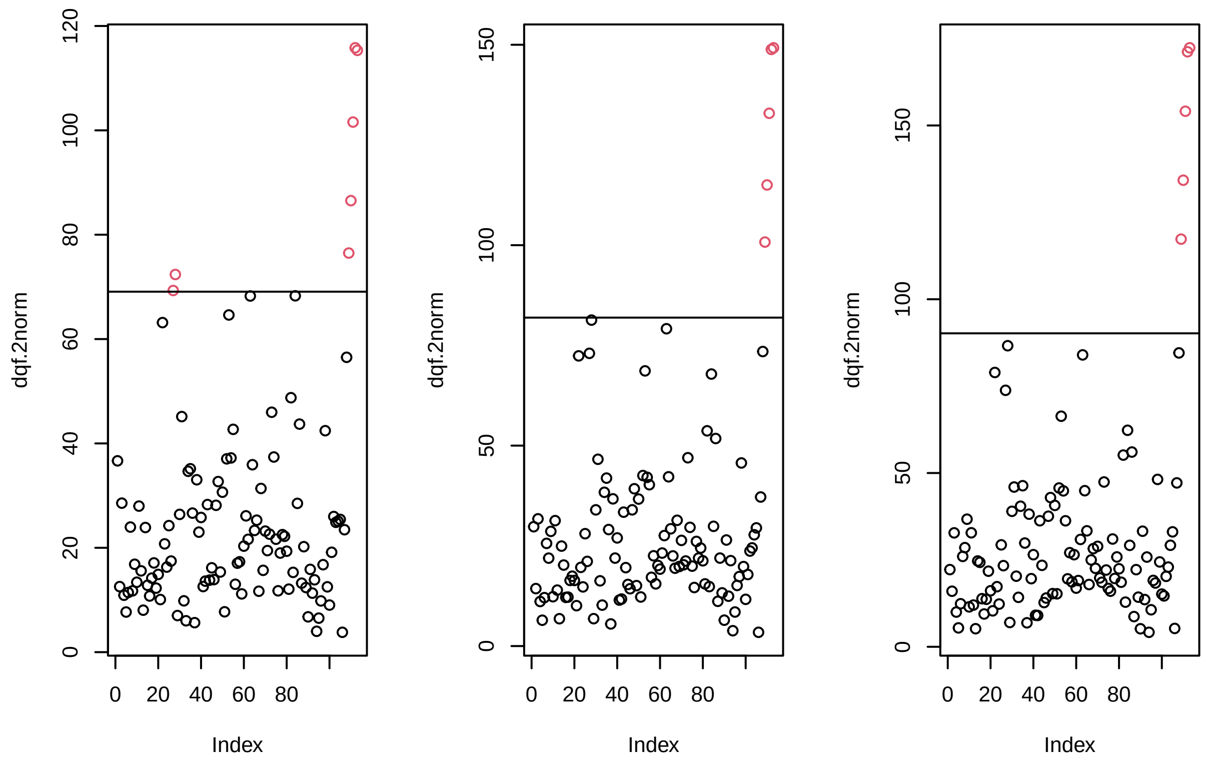
Function norms and operations

```
func.2norm <- function(dqf1,dqf2){
  sqrt(sum((dqf1-dqf2)^2))
}
```

```
dqf.2norm <- function(dqf){
  dqf.2norm <- c()

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  for(i in 1:length(dqf[,1])){
    norm <- func.2norm(mean,dqf[i,])
    dqf.2norm <- c(dqf.2norm, norm)
  }

  return(dqf.2norm)
}
```

```
plot.dqf.2norm <- function(dqf, n.sd=2){
  dqf.2norm <- c()
  labels <- c()

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  bound.dqf.2norm <- func.2norm(dqf.upperbound(mean,sd,n.sd),mean)

  for(i in 1:length(dqf[,1])){
    norm <- func.2norm(mean,dqf[i,])
    dqf.2norm <- c(dqf.2norm, norm)
    if(norm < bound.dqf.2norm) labels[i] <- 1
    else labels[i] <- 2
  }

  plot(dqf.2norm,col=labels)
  abline(h=bound.dqf.2norm)
}
```

```
par(mfrow=c(1,3))
plot.dqf.2norm(dqf1.1)
plot.dqf.2norm(dqf1.2)
plot.dqf.2norm(dqf1.3)
```

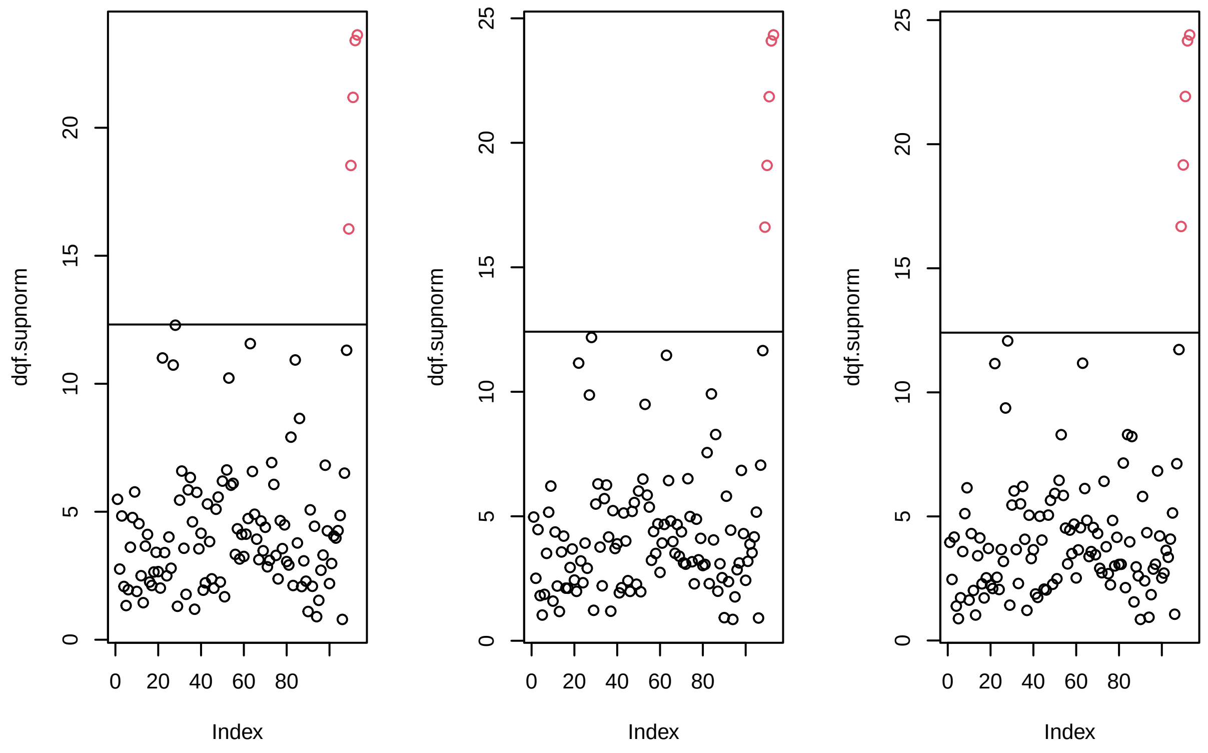


```r
func.supnorm <- function(dqf1,dqf2){
  sqrt(max((dqf1-dqf2)^2))
}
```

```r
dqf.supnorm <- function(dqf){
  dqf.supnorm <- c()

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  for(i in 1:length(dqf[,1])){
    norm <- func.supnorm(mean,dqf[i,])
    dqf.2norm <- c(dqf.supnorm, norm)
  }

  return(dqf.supnorm)
}
```

```r
plot.dqf.supnorm <- function(dqf, n.sd=2){
  dqf.supnorm <- c()
  labels <- c()

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  bound.dqf.supnorm <- func.supnorm(dqf.upperbound(mean,sd,n.sd),mean)
  for(i in 1:length(dqf[,1])){
    norm <- func.supnorm(mean,dqf[i,])
    dqf.supnorm <- c(dqf.supnorm, norm)
    if(norm < bound.dqf.supnorm) labels[i] <- 1
    else labels[i] <- 2
  }

  plot(dqf.supnorm,col=labels)
  abline(h=bound.dqf.supnorm)
}
```

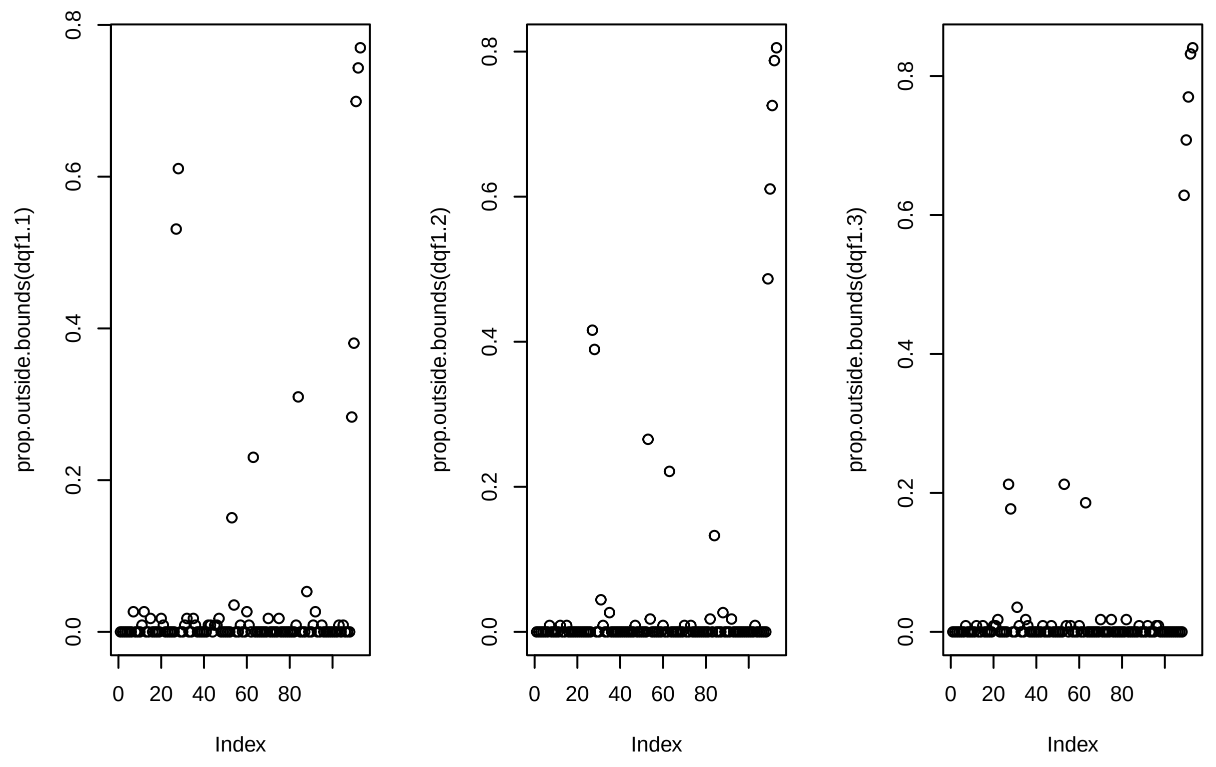
```
par(mfrow=c(1,3))
plot.dqf.supnorm(dqf1.1)
plot.dqf.supnorm(dqf1.2)
plot.dqf.supnorm(dqf1.3)
```



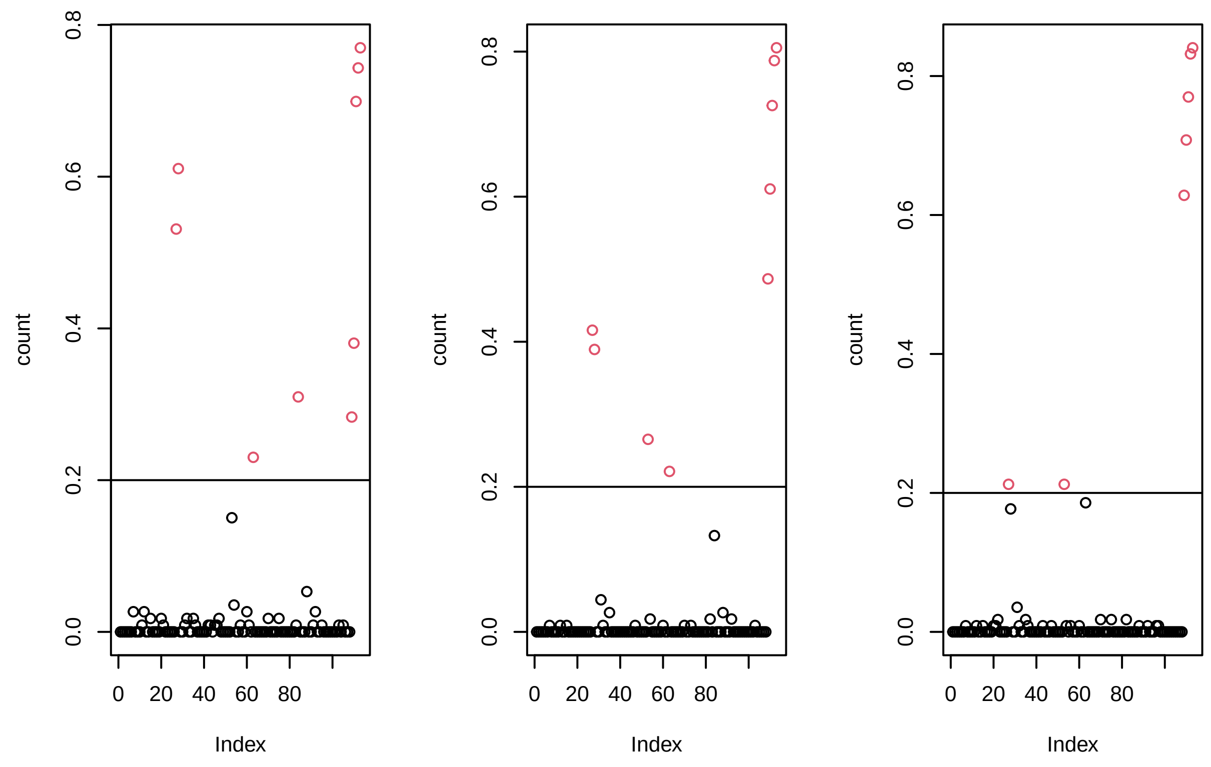
```
prop.outside.bounds <- function(dqf, n.sd=2){
  n <- length(dqf[,1])
  count <- rep(0,n)

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  lower.bound <- dqf.lowerbound(mean,sd,n.sd)
  upper.bound <- dqf.upperbound(mean,sd,n.sd)
  for(i in 1:n){
    for(j in 1:length(dqf[i,])){
      if(dqf[i,j] < lower.bound[j] | dqf[i,j] > upper.bound[j]){
        count[i] <- count[i]+1
      }
    }
  }

  return(count/n)

}
```

```
plot.prop.outside.bounds <- function(dqf,threshold=.2,n.sd=2){
  n <- length(dqf[,1])
  count <- rep(0,n)
  labels <- rep(1,n)

  mean <- dqf.mean(dqf)
  sd <- dqf.sd(dqf)

  lower.bound <- dqf.lowerbound(mean,sd,n.sd)
  upper.bound <- dqf.upperbound(mean,sd,n.sd)

  for(i in 1:n){
    for(j in 1:length(dqf[i,])){
      if(dqf[i,j] < lower.bound[j] | dqf[i,j] > upper.bound[j])count[i] <- count[i]+1
    }
    count[i] <- count[i]/n
    if(count[i] > threshold) labels[i] <- 2
  }

  plot(count,col=labels)
  abline(h=.2)

}
```

```
par(mfrow=c(1,3))
plot(prop.outside.bounds(dqf1.1))
plot(prop.outside.bounds(dqf1.2))
plot(prop.outside.bounds(dqf1.3))
```



```
par(mfrow=c(1,3))
plot.prop.outside.bounds(dqf1.1)
plot.prop.outside.bounds(dqf1.2)
plot.prop.outside.bounds(dqf1.3)
```

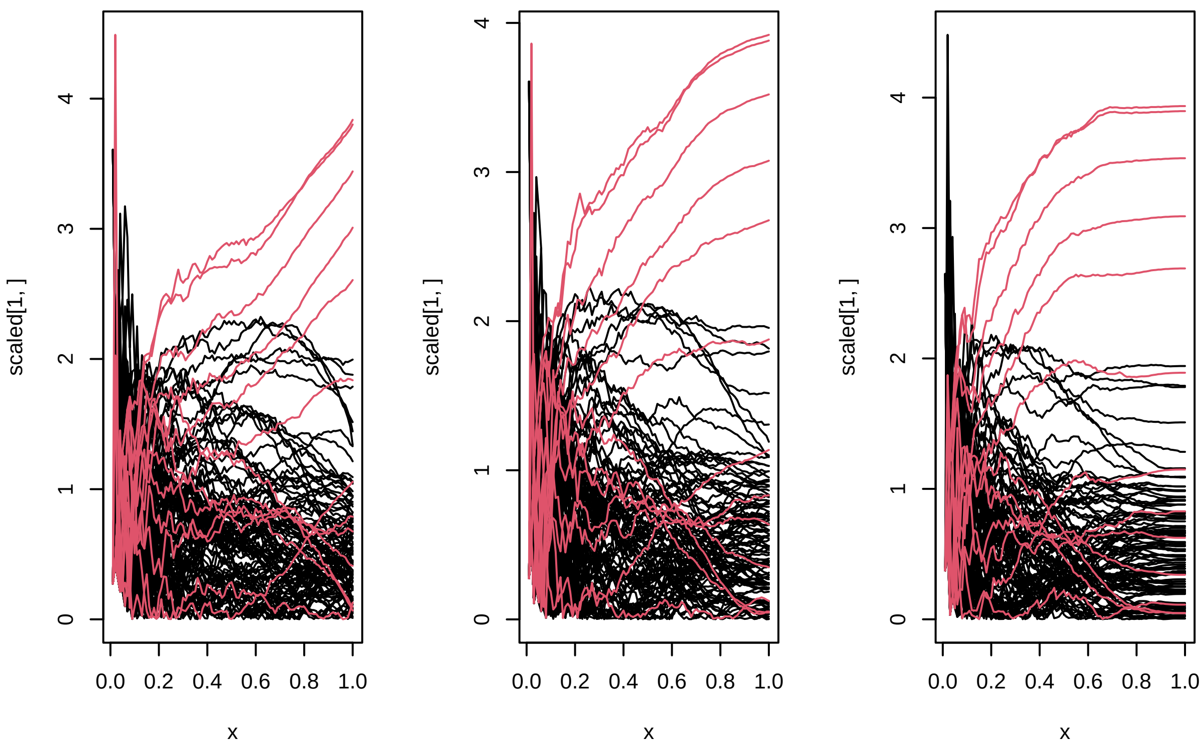


```
dqf.zscore <- function(dqf){
  return(abs(scale(dqf)))
}
```

```
dqf.mean.zscore <- function(dqf){
  return(rowSums(dqf.zscore(dqf))/length(dqf[,1]))
}
```

```
plot.zscore.dqf <- function(dqf,labels=NULL){
  scaled <- abs(scale(dqf))

  if(is.null(labels)) labels <- rep(1,length(dqf[,1]))

  x <- seq(.01,1,.01)
  plot(x,scaled[1,],t='l',col=labels[1],ylim=c(min(scaled),max(scaled)))
  for(i in 2:length(scaled[,1])){
    lines(x,scaled[i,],col=labels[i])
  }
}
```

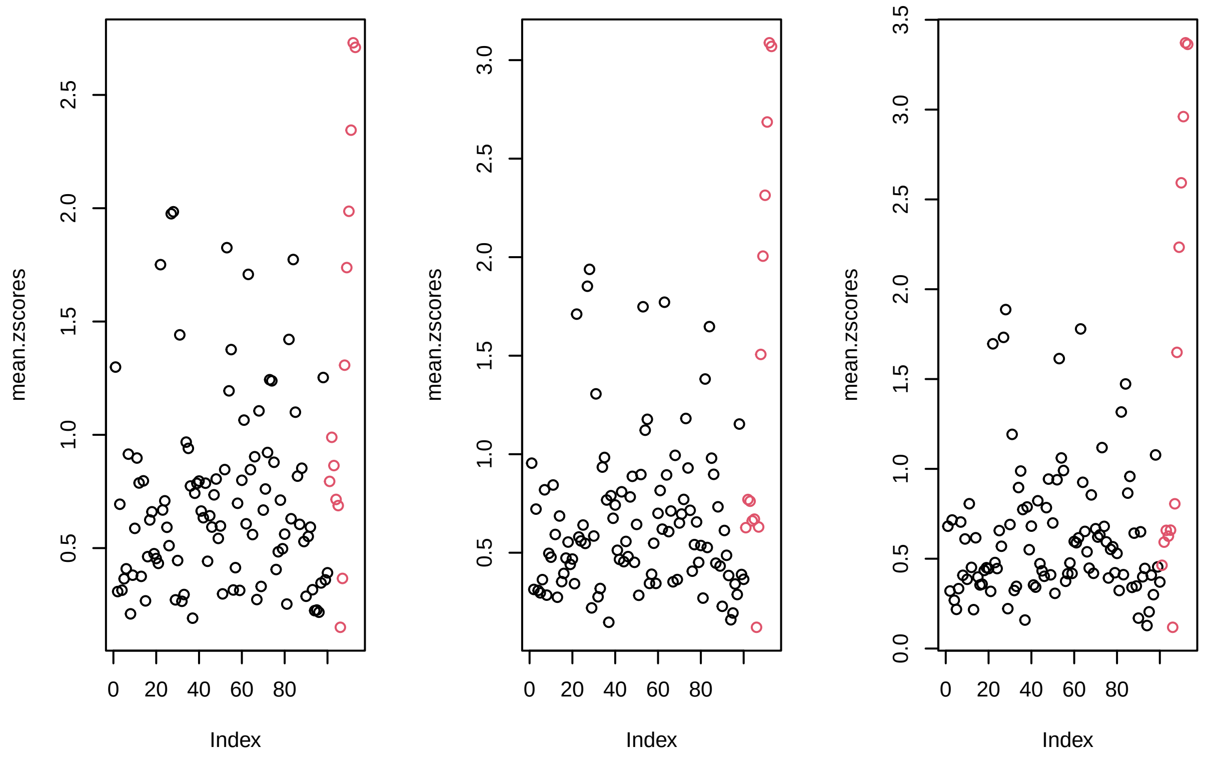
```
par(mfrow=c(1,3))
plot.zscore.dqf(dqf1.1,labels1)
plot.zscore.dqf(dqf1.2,labels1)
plot.zscore.dqf(dqf1.3,labels1)
```



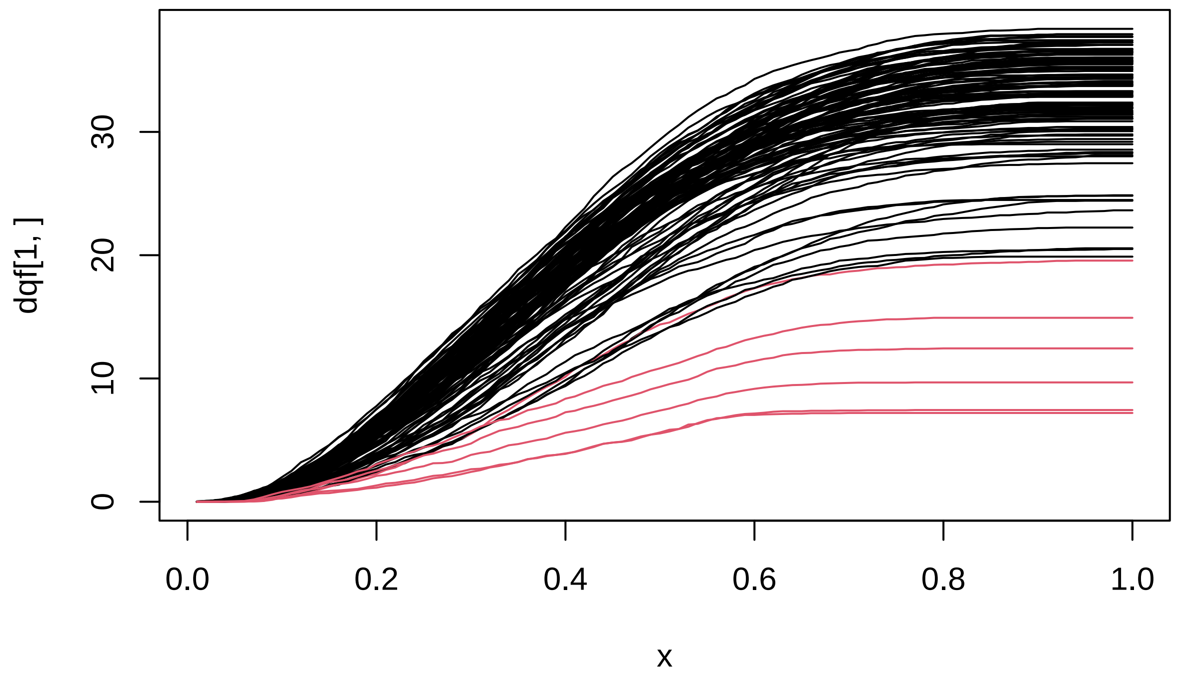
z-score

```
plot.mean.zscores <- function(dqf,labels=NULL){
  mean.zscores <- rowSums(dqf.zscore(dqf))/length(dqf[1,])
  if(is.null(labels)) labels <- rep(1,length(dqf[,1]))

  plot(mean.zscores,col=labels)
}
```

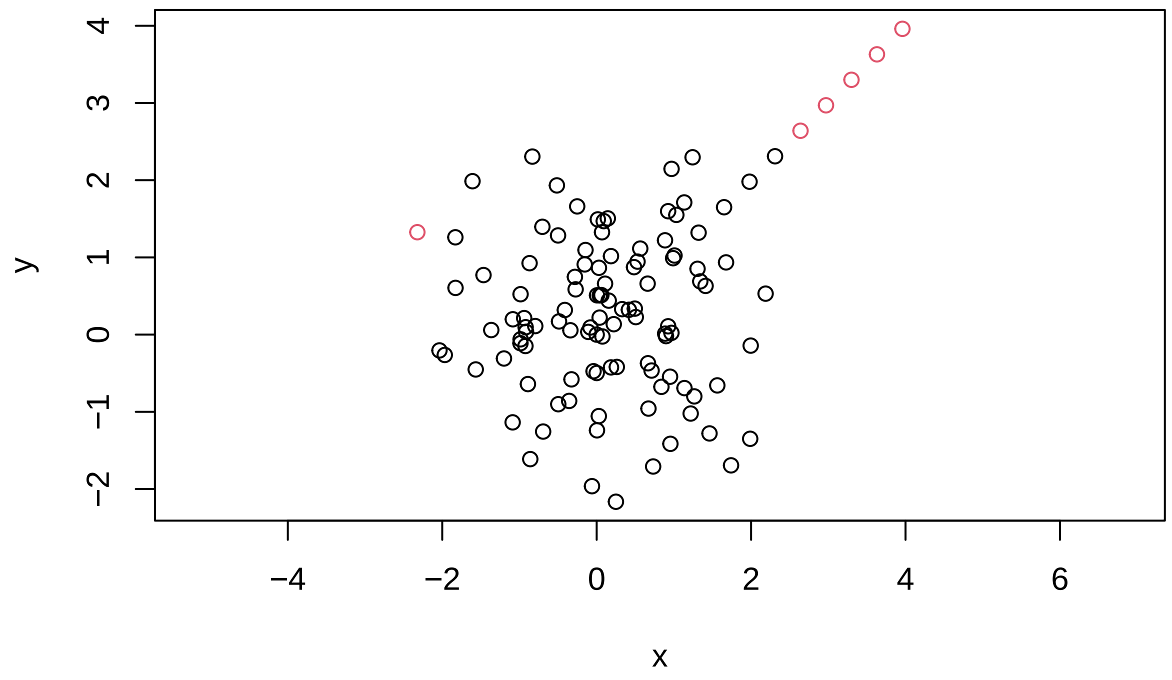
```
par(mfrow=c(1,3))
plot.mean.zscores(dqf1.1,labels1)
plot.mean.zscores(dqf1.2,labels1)
plot.mean.zscores(dqf1.3,labels1)
```



```
l <- rep(1,length(dqf1.1))
l[which(dqf.mean.zscore(dqf1.3) > 1.6)] <- 2
plot.dqf(dqf1.3,l)
```



```
plot(data1,col=l,asp=1)
```



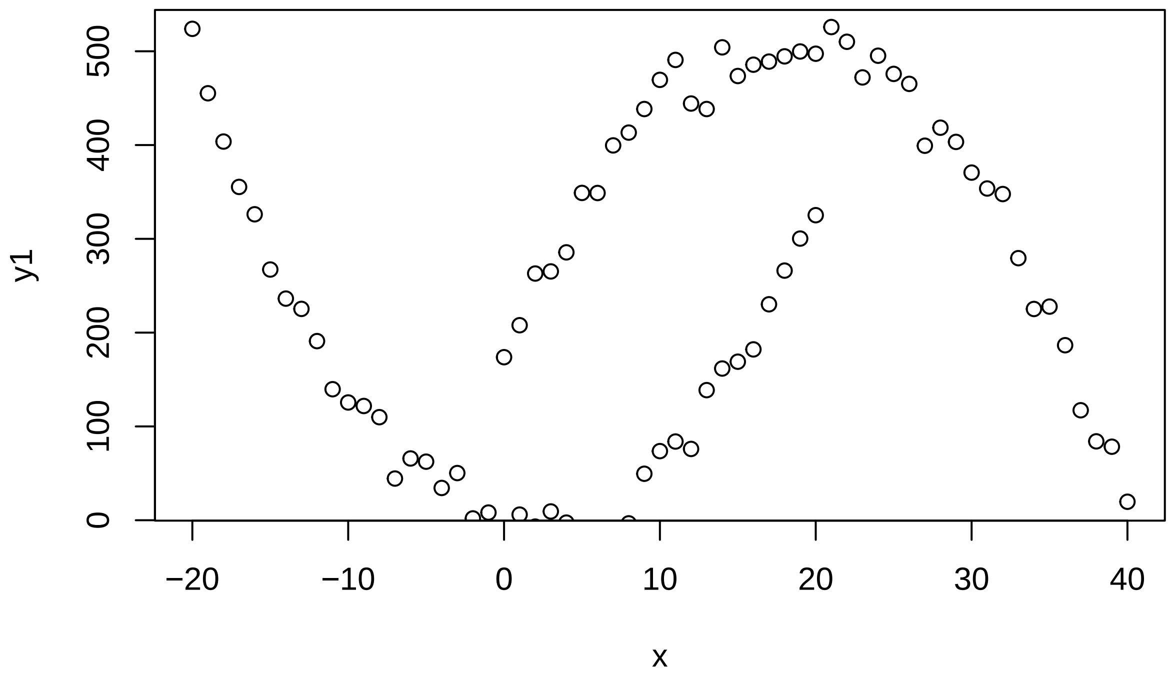
computation - store q__ij - average over different subsets/groups

data - half moon - circles

- others
- fuzzy 1,2-manifold in high dimensions

literature - using anomaly detection for unsupervised clustering - clustering for anomaly detection

```
set.seed(47)
x <- seq(-20,20,1)
x1 <- x+20
y1 <- (x-2)^2 + rnorm(length(x),0,20)
y2 <- -(x+2)^2 + 500 + rnorm(length(x),0,20)

data1 <- cbind(x,y1)
data1 <- rbind(data1,cbind(x1[1],y2[1]))
data2 <- cbind(x1,y2)

plot(x,y1,xlim=c(-20,40),ylim=c(min(y2),max(y1)))
points(x1,y2)
```



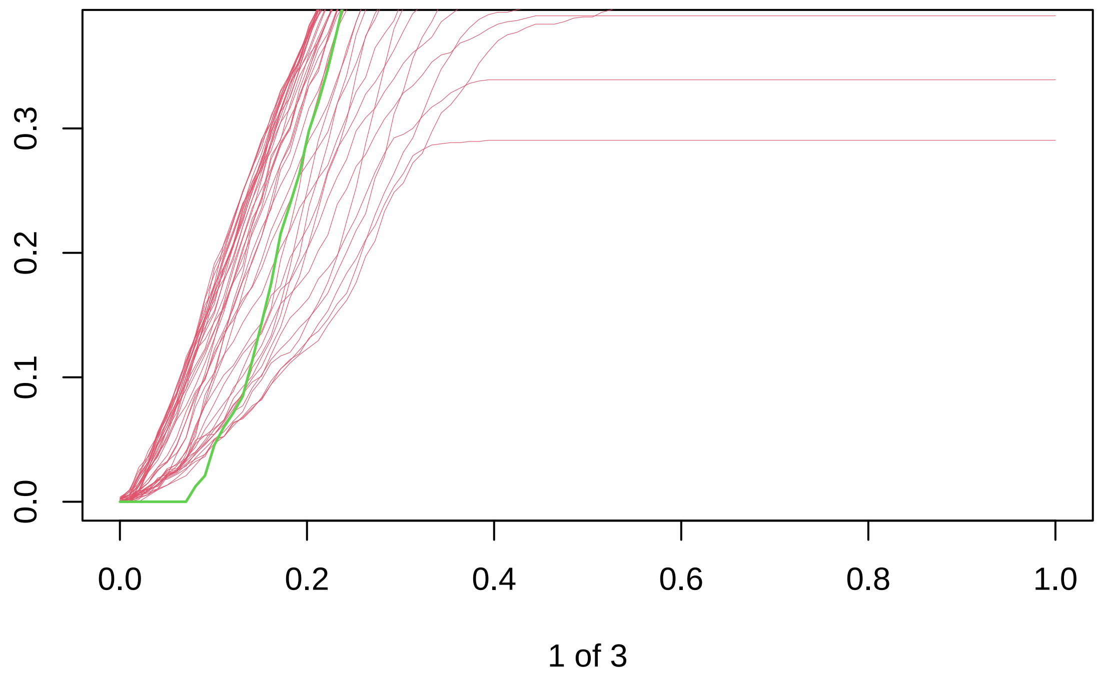
```
dqfs2 <- dqf.outlier(data1,g.scale=5)
```

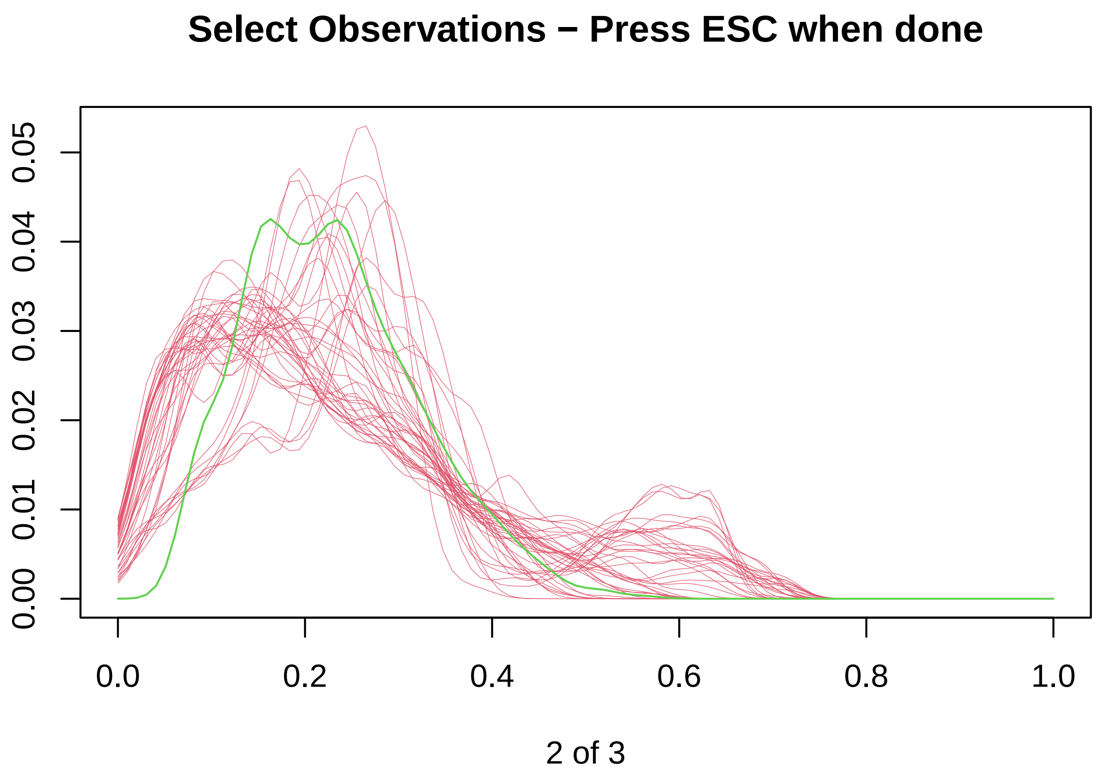
```
dqfs3 <- dqf.outlier(data2,g.scale=5)
```

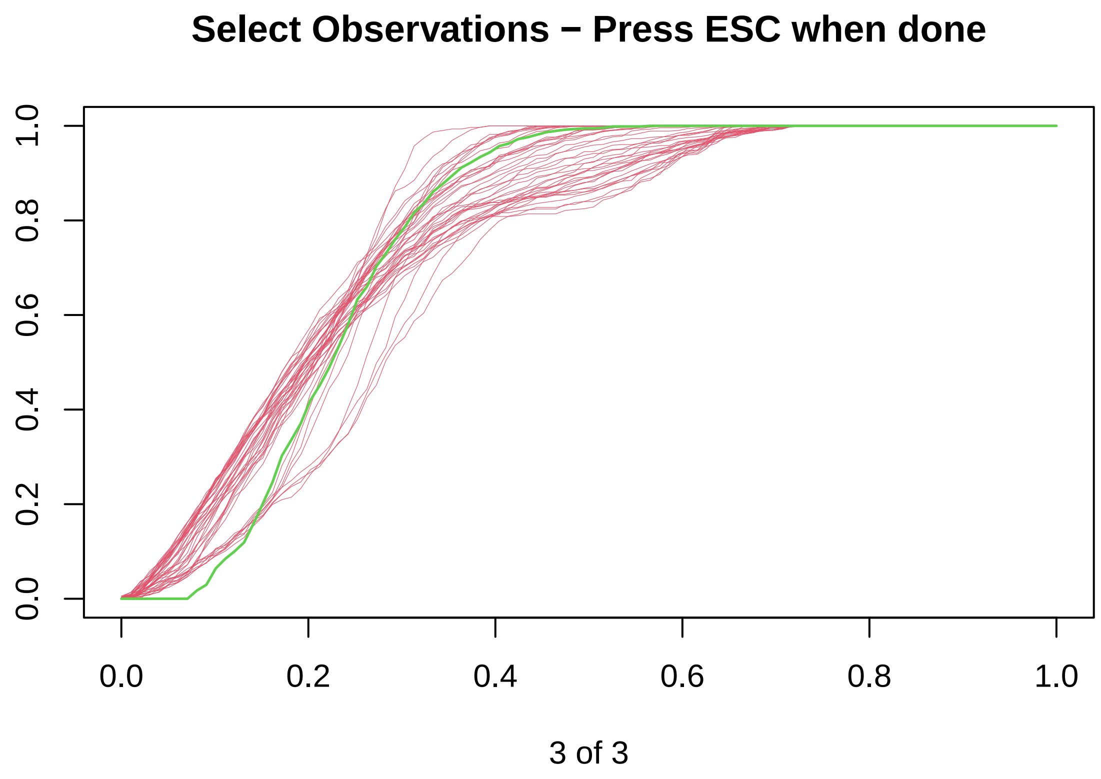
```
dqfs2.1 <- dqfs2$dqf1
dqfs2.2 <- dqfs2$dqf2
dqfs2.3 <- dqfs2$dqf3

dqfs3.1 <- dqfs3$dqf1
dqfs3.2 <- dqfs3$dqf2
dqfs3.3 <- dqfs3$dqf3
```

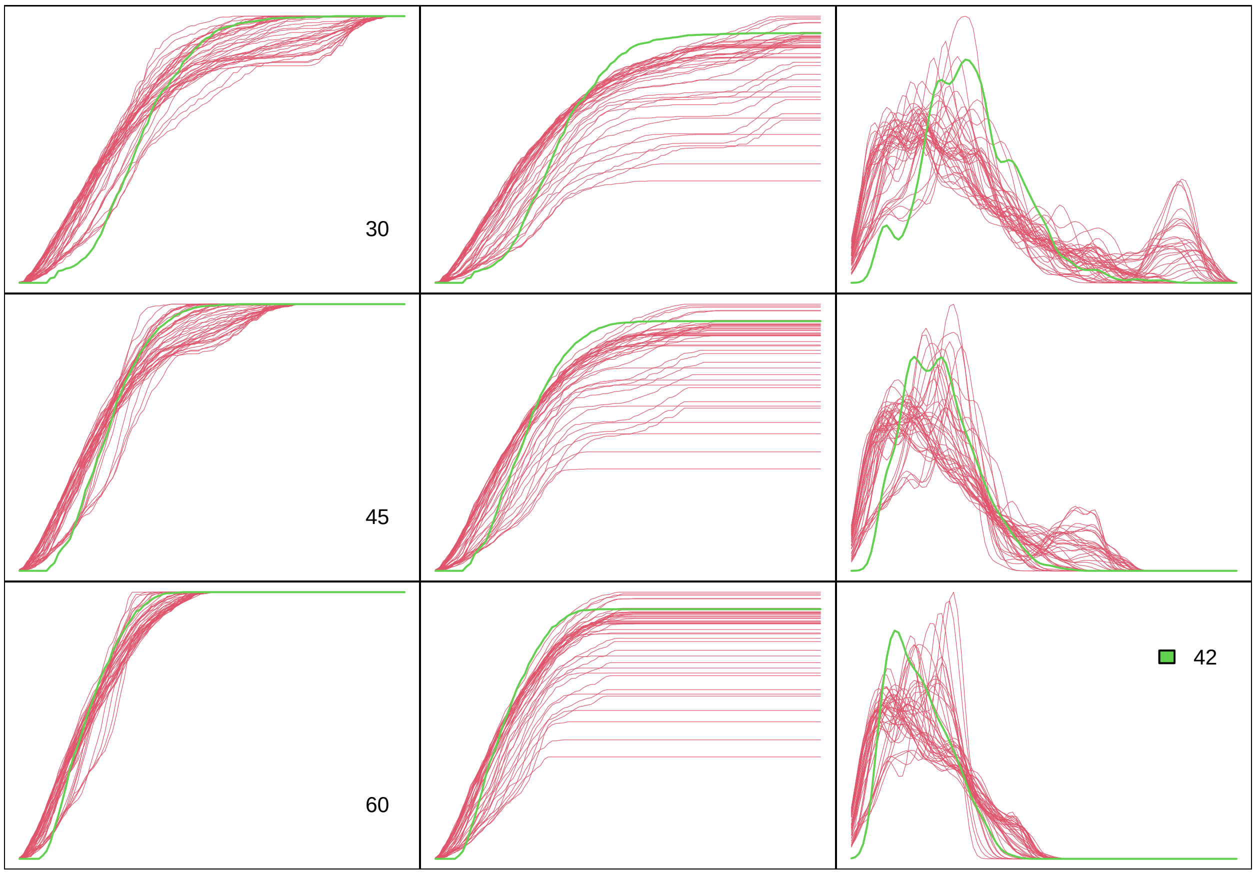
```
dqf.explore(dqfs2,42)
```

Select Observations − Press ESC when done



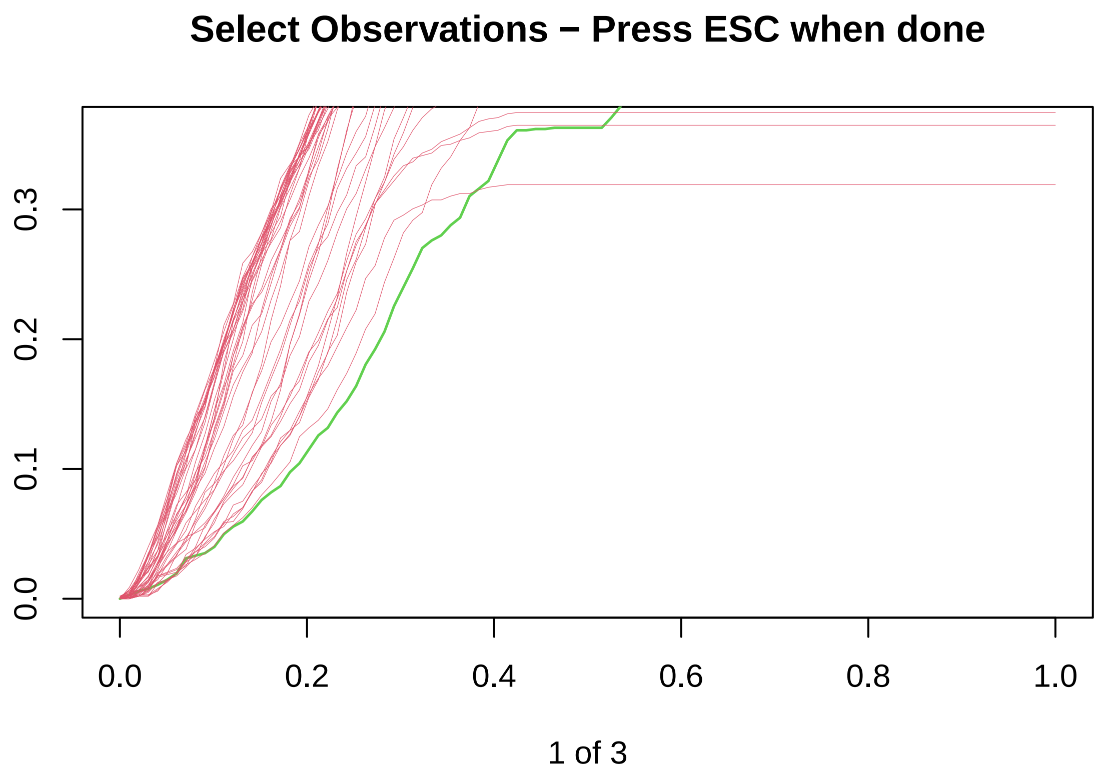

Select Observations − Press ESC when done
0.00
0.01
0.02
0.03
0.04
0.05
0.0
0.2
0.4
0.6
0.8
1.0
2 of 3

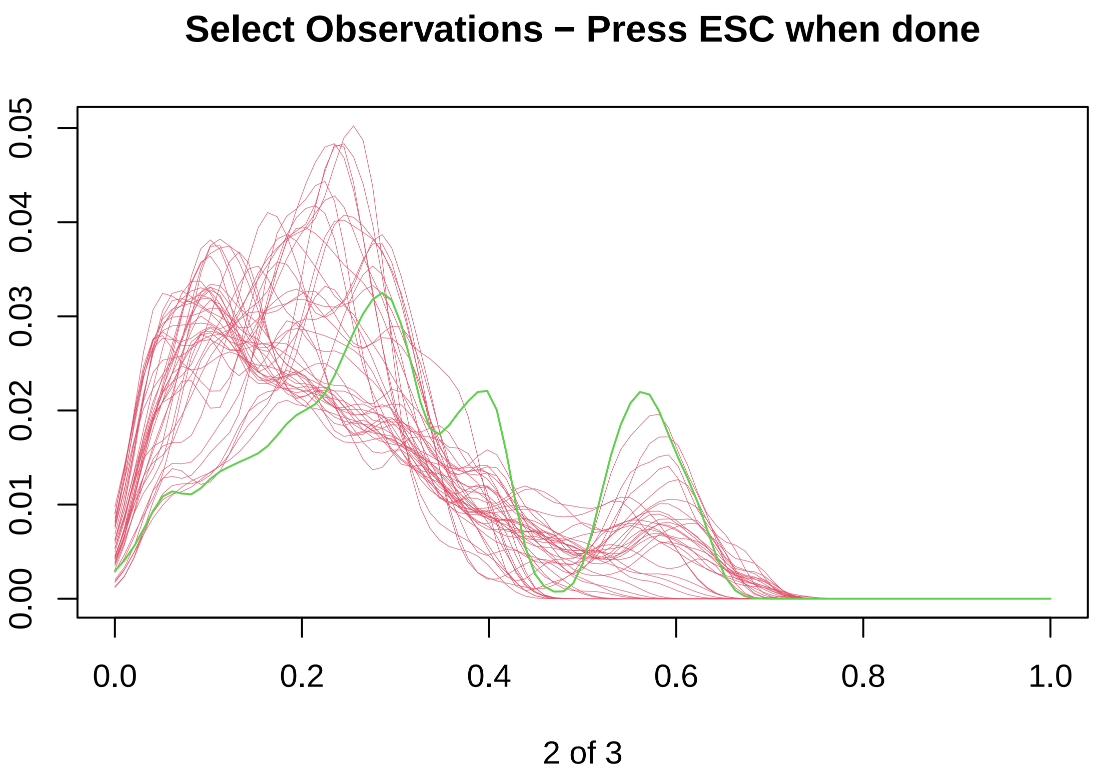

Select Observations − Press ESC when done
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
3 of 3

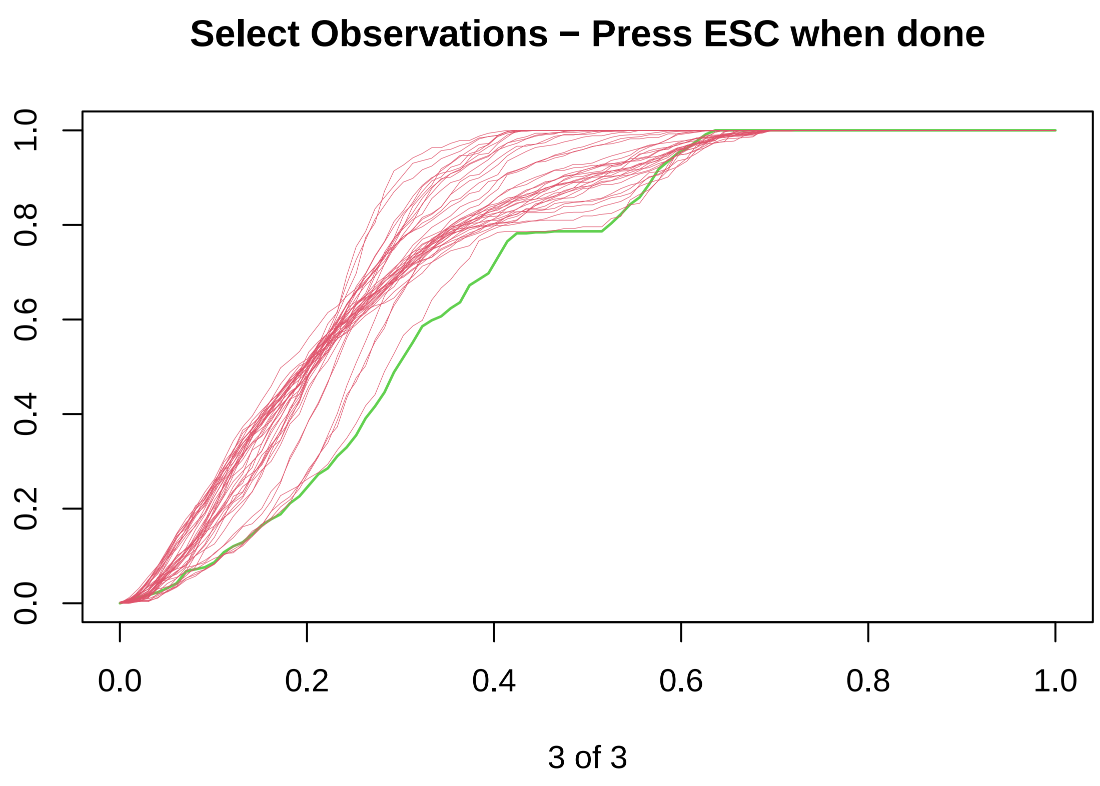


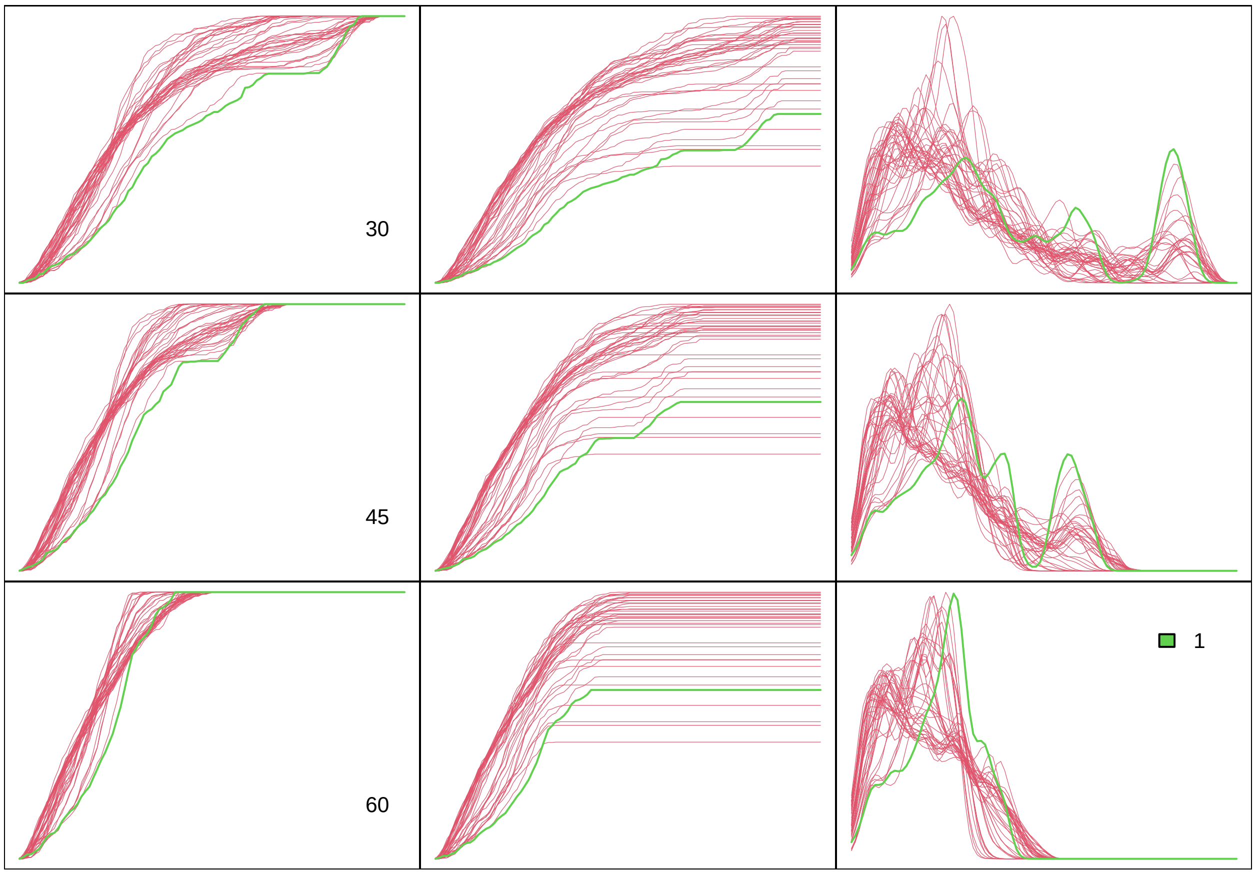
```
## [1] 42
```

```
dqf.explore(dqfs3,1)
```


Select Observations − Press ESC when done
0.0
0.1
0.2
0.3
0.0
0.2
0.4
0.6
0.8
1.0
1 of 3


Select Observations − Press ESC when done
0.00
0.01
0.02
0.03
0.04
0.05
0.0
0.2
0.4
0.6
0.8
1.0
2 of 3


Select Observations − Press ESC when done
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
3 of 3



```
## [1] 1
```