Semi-parametric density analysis

Andreas Kryger Jensen

11 February, 2018

Load data

```
rm(list=ls())
library(parallel)
library(gamlss.dist)
library(DEoptim)
library(mvmeta)
library(vioplot)

load(file="pigData.RData")

timeOriginal <- dat$time
time <- seq(0, 1, length.out = length(timeOriginal))
meas <- dat$meas

d <- t(dat$d)[, meas == 1 | meas == 2 | meas == 4]
id <- dat$pig[meas == 1 | meas == 2 | meas == 4]
meas <- meas[meas == 1 | meas == 2 | meas == 4]

rm(dat1, dat2, dat)

normalize <- function(t, d) {
  dF <- approxfun(t, d)
  int <- integrate(dF, min(t), max(t))$value
  d / int
}

dNorm <- sapply(1:ncol(d), function(k) normalize(time, d[,k]))
```

Define functions

```
trapz <- function (x, y) {
  n = length(y)
  sum(diff(x) * (y[-n] + y[-1]) / 2)
}

action <- function (time, d, w) {
  dF <- approxfun(time, d)
  wF <- approxfun(time, w)

  splineW <- smooth.spline(time, w, all.knots=TRUE)
  dwF <- predict(splineW, time, deriv=1)$y

  dF(wF(time)) * dwF
}

intervalScale <- function(d, min, max) {
  (max - min) / (max(d) - min(d)) * (d - max(d)) + max
}
```

```r
densityST5 <- function(time, par) {
  mu <- par[1]
  sigma <- par[2]
  nu <- par[3]
  tau <- par[4]

  d <- dST5(time, mu = mu, sigma = sigma, nu = nu, tau = tau)
  int <- trapz(time, d)

  d/int
}

densityNorm <- function(time, par) {
  mu <- par[1]
  sigma <- par[2]

  d <- dnorm(time, mean = mu, sd = sigma)
  int <- trapz(time, d)

  d/int
}

estParST5 <- function(time, d) {
  optFunc <- function(par) {
    acos(trapz(time, sqrt(d) * sqrt(densityST5(time, par))))^2
  }

  opt <- DEoptim(optFunc,
                 lower = c(mu=0, sigma=0.001, nu=0.01, tau=0.01),
                 upper = c(mu=1, sigma=20, nu=10, tau=10),
                 control = DEoptim.control(itermax=2000, trace=1, steptol=50))

  opt$optim$bestmem
}

estParNorm <- function(time, d) {
  optFunc <- function(par) {
    acos(trapz(time, sqrt(d) * sqrt(densityNorm(time, par))))^2
  }

  opt <- DEoptim(optFunc,
                 lower = c(mu=0, sigma=0.001),
                 upper = c(mu=1, sigma=20),
                 control = DEoptim.control(itermax=2000, trace=1, steptol=50))

  opt$optim$bestmem
}

getResidual <- function(time, obs, param) {
  inverse = function (f, lower = 0, upper = 1) {
    function (y) uniroot((function (x) f(x) - y), lower = lower, upper = upper)[1]$root
  }

  G <- Vectorize(function(u) integrate(approxfun(time, param), 0, u)$value)
  F <- Vectorize(function(u) integrate(approxfun(time, obs), 0, u)$value)
  Ginv <- Vectorize(inverse(G))

  w <- sapply(1:(length(time) - 10), function(i) Ginv(F(time[i])))
  w <- c(w, seq(max(w), 1, length.out=11)[-1])
```

```
  w
}

residualMagnitude <- function(time, w) {
  wF <- smooth.spline(time, w)
  wDeriv <- predict(wF, time, deriv=1)$y
  wDeriv[wDeriv < 0] <- 0
  acos(trapz (time, sqrt(wDeriv)))^2
}

std_mlm <- function(model) {
  Rinv <- with(model$qr, backsolve(qr, diag(rank)))
  std_unscaled <- sqrt(rowSums(Rinv ^ 2)[order(model$qr$pivot)])
  sigma <- sqrt(colSums(model$residuals ^ 2) / model$df.residual)
  "dimnames<-"(outer(std_unscaled, sigma), list = dimnames(model$coefficients))
}

confint.mlm <- function (model, level = 0.95) {
  beta <- coef(model)
  se <- std_mlm (model)
  alpha <- qt((1 - level) / 2, df = model$df.residual)
  list(lower = beta + alpha * se, upper = beta - alpha * se)
}
```

Analysis - Skew t-distribution Type 5

```
set.seed(12345)
paramEstST5 <- do.call("rbind", mclapply(1:ncol(dNorm), function(i) {
  estParST5(time, dNorm[,i])
}, mc.cores = 8, mc.preschedule = FALSE))

#Get parametric fits
dParamST5 <- apply(paramEstST5, 1, function(p) densityST5(time, p))

#Obtain residual functions
residualST5 <- do.call("cbind", mclapply(1:ncol(dNorm), function(k) {
  getResidual(time, dNorm[,k], dParamST5[,k])
}, mc.cores=8))

#Calculate residual magnitues
residualMagST5 <- apply(residualST5, 2, function(r) residualMagnitude(time, r))
```

Test equality of parameters

```
summary(lm(paramEstST5 ~ factor(meas)))
```

```
## Response mu :
##
## Call:
## lm(formula = mu ~ factor(meas))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.063620 -0.013283 -0.002223  0.015239  0.058392
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.166887   0.005783  28.857   <2e-16 ***
## factor(meas)2 0.007319   0.008179   0.895    0.375
## factor(meas)4 0.003782   0.008179   0.462    0.646
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02454 on 51 degrees of freedom
## Multiple R-squared:  0.01547,    Adjusted R-squared:  -0.02314
## F-statistic: 0.4006 on 2 and 51 DF,  p-value: 0.672
##
##
## Response sigma :
##
## Call:
## lm(formula = sigma ~ factor(meas))
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0184902 -0.0045994 -0.0009139  0.0055144  0.0150197
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.048746   0.001816  26.840  < 2e-16 ***
## factor(meas)2 0.012479   0.002568   4.858 1.17e-05 ***
## factor(meas)4 0.015885   0.002568   6.185 1.05e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007705 on 51 degrees of freedom
## Multiple R-squared:  0.454,  Adjusted R-squared:  0.4326
## F-statistic: 21.21 on 2 and 51 DF,  p-value: 1.986e-07
##
##
## Response nu :
##
## Call:
## lm(formula = nu ~ factor(meas))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21704 -0.09234 -0.02739  0.03450  0.85426
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.770230   0.039824  19.341   <2e-16 ***
## factor(meas)2 -0.002943   0.056320  -0.052   0.9585
## factor(meas)4  0.132492   0.056320   2.352   0.0225 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.169 on 51 degrees of freedom
## Multiple R-squared:  0.1289, Adjusted R-squared:  0.09473
## F-statistic: 3.773 on 2 and 51 DF,  p-value: 0.02963
##
##
## Response tau :
##
## Call:
## lm(formula = tau ~ factor(meas))
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32381 -0.11431 -0.01664  0.07068  0.33592
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.78925    0.03641  21.676  < 2e-16 ***
## factor(meas)2 -0.15898    0.05149  -3.087 0.003262 **
## factor(meas)4 -0.20762    0.05149  -4.032 0.000185 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1545 on 51 degrees of freedom
## Multiple R-squared:  0.2586, Adjusted R-squared:  0.2295
## F-statistic: 8.894 on 2 and 51 DF,  p-value: 0.000486
```

```
summary(lm(paramEstST5 ~ relevel(factor(meas), "2")))
```

```
## Response mu :
##
## Call:
## lm(formula = mu ~ relevel(factor(meas), "2"))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.063620 -0.013283 -0.002223  0.015239  0.058392
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  0.174207   0.005783  30.122   <2e-16 ***
## relevel(factor(meas), "2")1 -0.007319   0.008179  -0.895    0.375
## relevel(factor(meas), "2")4 -0.003538   0.008179  -0.433    0.667
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02454 on 51 degrees of freedom
## Multiple R-squared:  0.01547,    Adjusted R-squared:  -0.02314
## F-statistic: 0.4006 on 2 and 51 DF,  p-value: 0.672
##
##
## Response sigma :
##
## Call:
## lm(formula = sigma ~ relevel(factor(meas), "2"))
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0184902 -0.0045994 -0.0009139  0.0055144  0.0150197
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  0.061225   0.001816  33.711  < 2e-16 ***
## relevel(factor(meas), "2")1 -0.012479   0.002568  -4.858 1.17e-05 ***
## relevel(factor(meas), "2")4  0.003407   0.002568   1.326    0.191
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007705 on 51 degrees of freedom
## Multiple R-squared:  0.454,  Adjusted R-squared:  0.4326
## F-statistic: 21.21 on 2 and 51 DF,  p-value: 1.986e-07
```

```
##
##
## Response nu :
##
## Call:
## lm(formula = nu ~ relevel(factor(meas), "2"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21704 -0.09234 -0.02739  0.03450  0.85426
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  0.767288   0.039824  19.267   <2e-16 ***
## relevel(factor(meas), "2")1 0.002943   0.056320   0.052   0.9585
## relevel(factor(meas), "2")4 0.135434   0.056320   2.405   0.0199 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.169 on 51 degrees of freedom
## Multiple R-squared:  0.1289, Adjusted R-squared:  0.09473
## F-statistic: 3.773 on 2 and 51 DF,  p-value: 0.02963
##
##
## Response tau :
##
## Call:
## lm(formula = tau ~ relevel(factor(meas), "2"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32381 -0.11431 -0.01664  0.07068  0.33592
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.63027    0.03641  17.309  < 2e-16 ***
## relevel(factor(meas), "2")1  0.15898    0.05149   3.087  0.00326 **
## relevel(factor(meas), "2")4 -0.04864    0.05149  -0.945  0.34935
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1545 on 51 degrees of freedom
## Multiple R-squared:  0.2586, Adjusted R-squared:  0.2295
## F-statistic: 8.894 on 2 and 51 DF,  p-value: 0.000486
```

```
anova(lm(paramEstST5 ~ factor(meas)), lm(paramEstST5 ~ 1))
```

```
## Analysis of Variance Table
##
## Model 1: paramEstST5 ~ factor(meas)
## Model 2: paramEstST5 ~ 1
##   Res.Df Df  Gen.var.  Pillai approx F num Df den Df    Pr(>F)
## 1     51    0.0017331
## 2     53  2 0.0020913 0.63667   5.7207      8     98 5.686e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
round(lm(paramEstST5 ~ factor(meas) - 1)$coefficients, 3)
```

```
##                  mu sigma    nu   tau
## factor(meas)1 0.167 0.049 0.770 0.789
```

```
## factor(meas)2 0.174 0.061 0.767 0.630
## factor(meas)4 0.171 0.065 0.903 0.582
```

```
round(confint.mlm(lm(paramEstST5 ~ factor(meas) - 1))$lower, 3)
```

```
##                  mu sigma    nu   tau
## factor(meas)1 0.155 0.045 0.690 0.716
## factor(meas)2 0.163 0.058 0.687 0.557
## factor(meas)4 0.159 0.061 0.823 0.509
```

```
round(confint.mlm(lm(paramEstST5 ~ factor(meas) - 1))$upper, 3)
```

```
##                  mu sigma    nu   tau
## factor(meas)1 0.178 0.052 0.850 0.862
## factor(meas)2 0.186 0.065 0.847 0.703
## factor(meas)4 0.182 0.068 0.983 0.655
```

Test residual magnitudes

```
mean(residualMagST5)
```

```
## [1] 0.01257472
```

```
summary(lm(residualMagST5 ~ factor(meas)))
```

```
##
## Call:
## lm(formula = residualMagST5 ~ factor(meas))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.009645 -0.004201 -0.001623  0.001549  0.021622
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.012682   0.001590   7.976 1.58e-10 ***
## factor(meas)2 -0.001572   0.002248  -0.699    0.488
## factor(meas)4  0.001252   0.002248   0.557    0.580
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.006745 on 51 degrees of freedom
## Multiple R-squared:  0.03013,    Adjusted R-squared:  -0.0079
## F-statistic: 0.7923 on 2 and 51 DF,  p-value: 0.4583
```

Analysis - Normal distribution

```
set.seed(12345)
paramEstNormal <- do.call("rbind", mclapply(1:ncol(dNorm), function(i) {
  estParNorm(time, dNorm[,i])
}, mc.cores = 8, mc.preschedule = FALSE))

#Get parametric fits
dParamNormal <- apply(paramEstNormal, 1, function(p) densityNorm(time, p))

#Obtain residual functions
residualNormal <- do.call("cbind", mclapply(1:ncol(dNorm), function(k) {
  getResidual(time, dNorm[,k], dParamNormal[,k])
}, mc.cores=8))
```

```r
#Calculate residual magnitues
residualMagNormal <- apply(residualNormal, 2, function(r) residualMagnitude(time, r))
```

Test residual magnitudes

```r
mean(residualMagNormal)
```

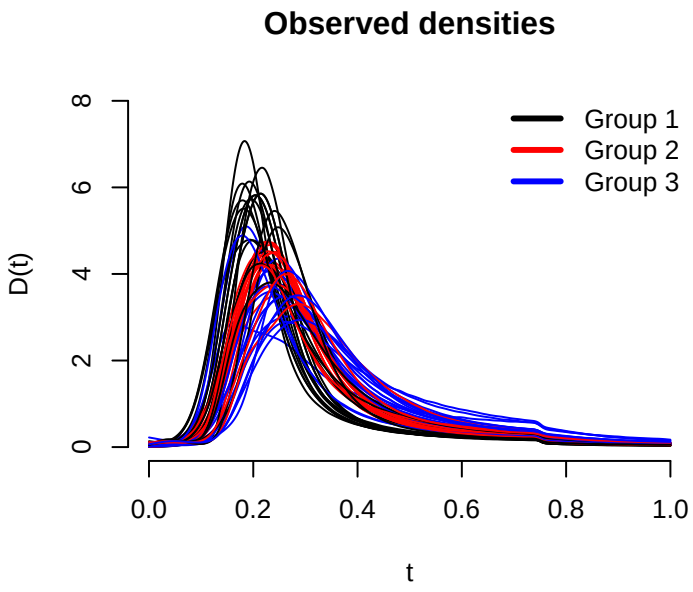
```
## [1] 0.3423053
```

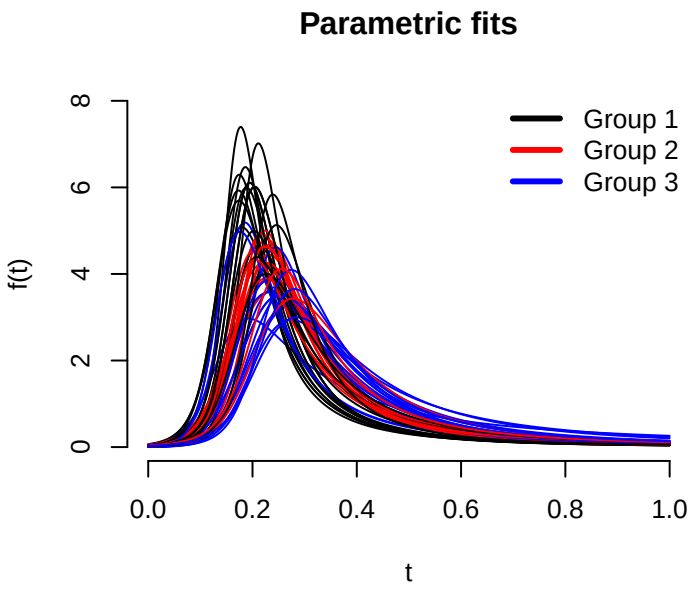
```r
summary(lm(residualMagNormal ~ factor(meas)))
```

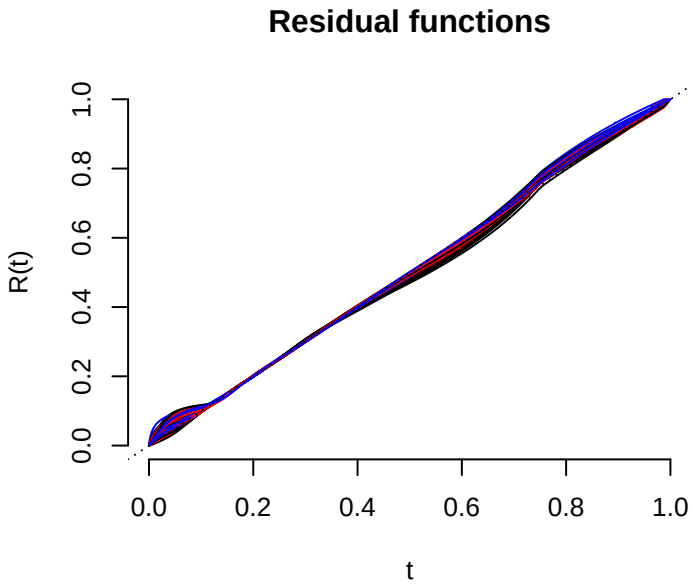
```
##
## Call:
## lm(formula = residualMagNormal ~ factor(meas))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.278326 -0.067757  0.003323  0.076797  0.181799
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.50428    0.02472  20.399  < 2e-16 ***
## factor(meas)2 -0.18646    0.03496  -5.333 2.22e-06 ***
## factor(meas)4 -0.29947    0.03496  -8.566 1.92e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1049 on 51 degrees of freedom
## Multiple R-squared:  0.5947, Adjusted R-squared:  0.5788
## F-statistic: 37.42 on 2 and 51 DF,  p-value: 9.938e-11
```

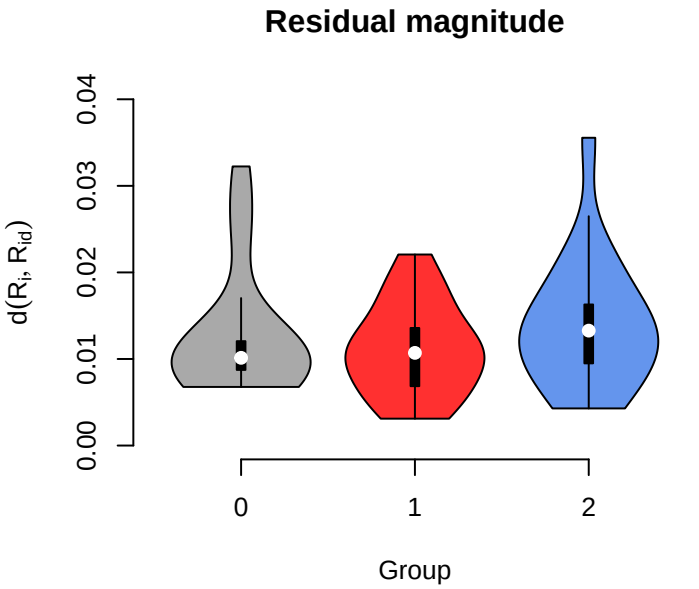
Results plot - Skew t-distribution Type 5

```r
par(mfrow=c(2,2), bty="n")
matplot(time, dNorm, type="l", lty=1, col=meas, xlab="t", ylab="D(t)", ylim=c(0,8))
title("Observed densities")
legend("topright", c("Group 1", "Group 2", "Group 3"), col=c(1,2,4), bty="n", lwd=3)

matplot(time, dParamST5, type="l", lty=1, col=meas, xlab="t", ylab="f(t)", ylim=c(0,8))
title("Parametric fits")
legend("topright", c("Group 1", "Group 2", "Group 3"), col=c(1,2,4), bty="n", lwd=3)

matplot(time, residualST5, type="l", lty=1, col=meas, xlab="t", ylab="R(t)")
title("Residual functions")
abline(0, 1, lty=3)

plot(0, 0, type="n", xlim=c(-0.5,2.5), ylim=c(0 ,0.04), bty="n", xaxt="n",
     xlab="Group", ylab=expression(d(R[i], R[id])))
axis(1, at=0:2)
vioplot(residualMagST5[meas == 1], col="darkgray", at = 0, add=TRUE)
vioplot(residualMagST5[meas == 2], col="firebrick1", at = 1, add=TRUE)
vioplot(residualMagST5[meas == 4], col="cornflowerblue", at = 2, add=TRUE)
title("Residual magnitude")
```

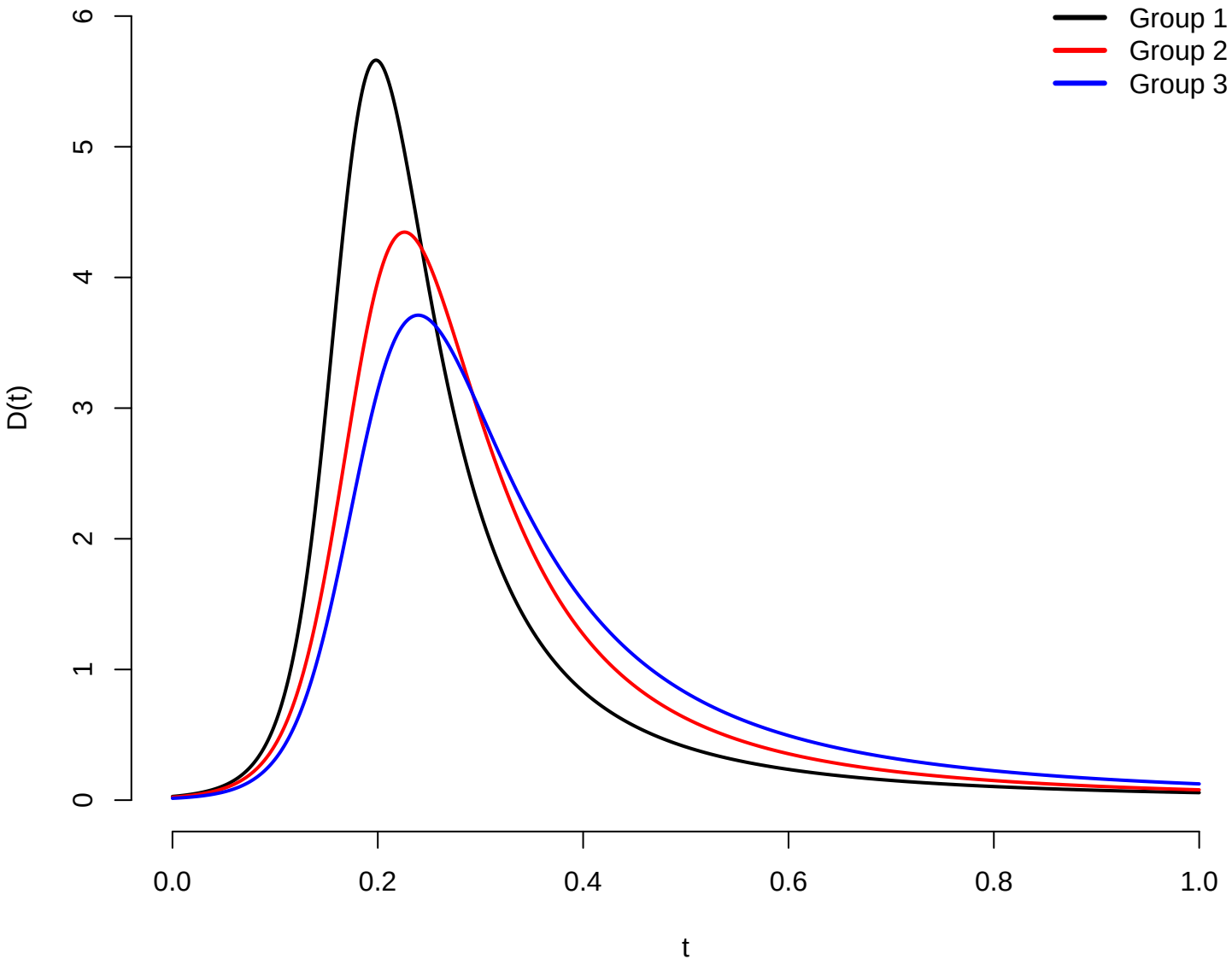






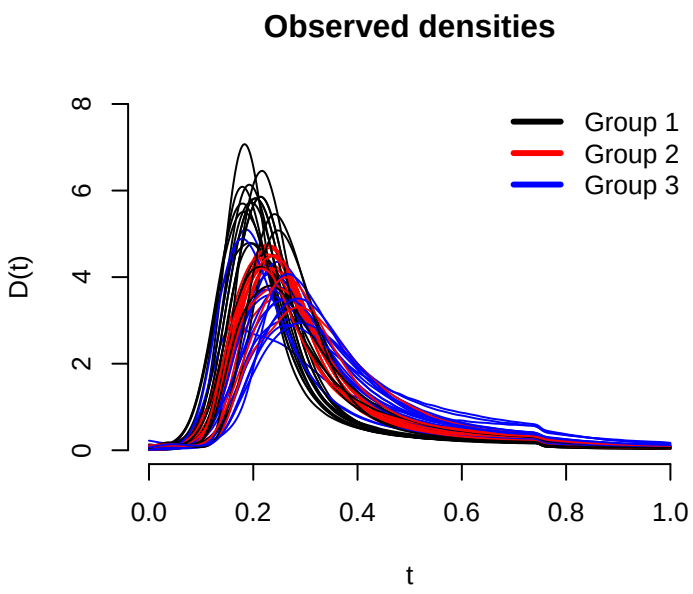


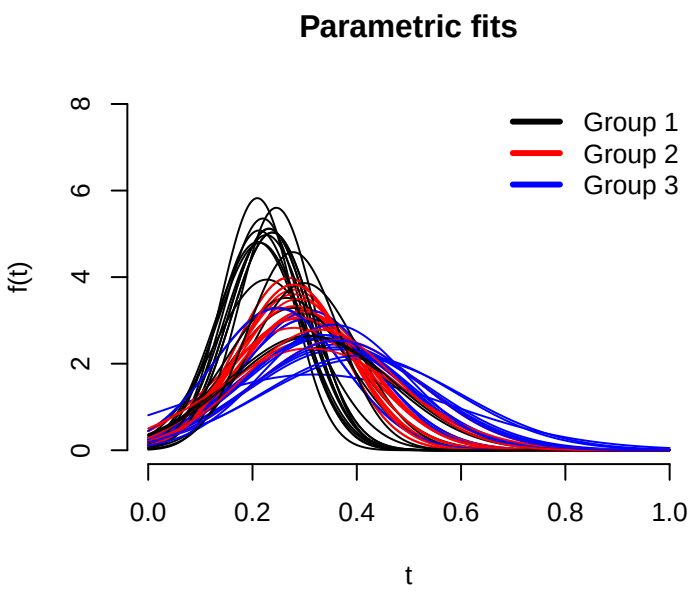

```
mST5dat <- data.frame(id = id, meas = factor(meas),
                      mu = paramEstST5[,1], sigma = paramEstST5[,2],
                      nu = paramEstST5[,3], tau = paramEstST5[,4])
mST5 <- lm(cbind(mu, sigma, nu, tau) ~ meas - 1, data=mST5dat)
plot(time, densityST5(time, predict(mST5, data.frame(meas = "1"))), lwd=2, type="l",
     xlab="t", ylab="D(t)", bty="n", ylim=c(0,6))
lines(time, densityST5(time, predict(mST5, data.frame(meas = "2"))), lwd=2, col=2)
lines(time, densityST5(time, predict(mST5, data.frame(meas = "4"))), lwd=2, col=4)
legend("topright", c("Group 1", "Group 2", "Group 3"), col=c(1,2,4), bty="n", lwd=3)
```

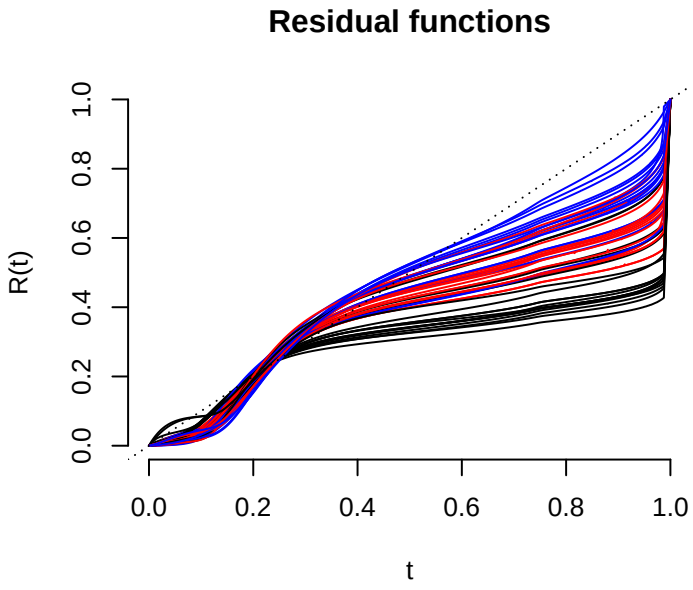


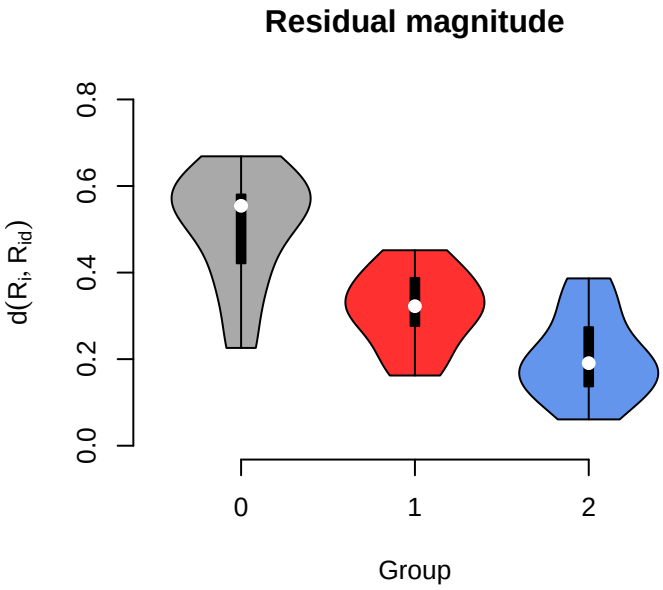
Results plot - Normal distribution

```
par(mfrow=c(2,2), bty="n")
matplot(time, dNorm, type="l", lty=1, col=meas, xlab="t", ylab="D(t)", ylim=c(0,8))
title("Observed densities")
legend("topright", c("Group 1", "Group 2", "Group 3"), col=c(1,2,4), bty="n", lwd=3)

matplot(time, dParamNormal, type="l", lty=1, col=meas, xlab="t", ylab="f(t)", ylim=c(0,8))
title("Parametric fits")
legend("topright", c("Group 1", "Group 2", "Group 3"), col=c(1,2,4), bty="n", lwd=3)

matplot(time, residualNormal, type="l", lty=1, col=meas, xlab="t", ylab="R(t)")
title("Residual functions")
abline(0, 1, lty=3)

plot(0, 0, type="n", xlim=c(-0.5,2.5), ylim=c(0, 0.8), bty="n", xaxt="n",
     xlab="Group", ylab=expression(d(R[i], R[id])))
axis(1, at=0:2)
vioplot(residualMagNormal[meas == 1], col="darkgray", at = 0, add=TRUE)
vioplot(residualMagNormal[meas == 2], col="firebrick1", at = 1, add=TRUE)
vioplot(residualMagNormal[meas == 4], col="cornflowerblue", at = 2, add=TRUE)
title("Residual magnitude")
```










```
save.image("semiParametricAnalysis.RData")
```