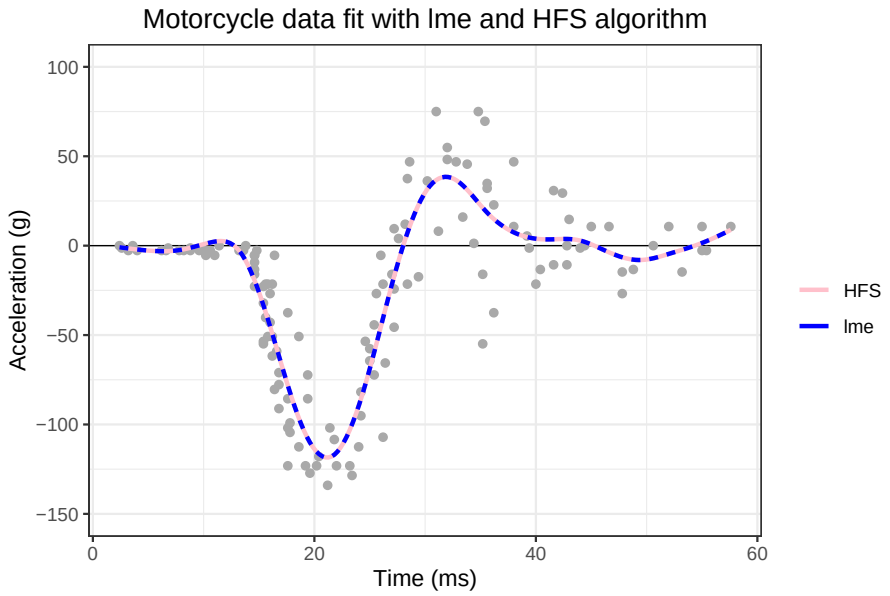
P-spline fit using mixed model and HFS algorithm (Motorcycle data)



Automatic smoothing of the motorcycle data with a mixed model using the `lme` function (solid blue line) and with the HFS algorithm (broken red line), both with $\lambda \approx 0.01$. R code in `f-mot-lme-mix.R`

```
# P-spline fit using mixed model and HFS algorithm (Motorcycle data)
# A graph in the book 'Practical Smoothing. The Joys of P-splines'
# Paul Eilers and Brian Marx, 2019

library(JOPS)
library(nlme)
library(MASS)
library(ggplot2)

makeZ <- function(x, xl = min(x), xr = max(x), nseg = 10, bdeg = 3) {
  # Construct basis for P-spline random effects
  B <- bbase(x, xl, xr, nseg = nseg, bdeg = bdeg)
  D <- diff(diag(ncol(B)), diff = 2)
  Q <- solve(D %*% t(D), D)
  Z <- B %*% t(Q)
  Z
}

# Get the data
data(mcycle)
x = mcycle$times
y = mcycle$accel
grp = 0 * x + 1   # A 'grouping variable', needed for lme()

# Set P-spline parameters
xmin = min(x)
xmax = max(x)
nseg = 20
bdeg = 3
Z = makeZ(x, xl = xmin, xr = xmax, nseg = nseg, bdeg = bdeg)

# Mixed model fitting with lme
lm1 <- lme(y ~ x, random = list(grp = pdIdent(~ Z - 1)))
cfix = lm1$coefficients$fixed
cran = c(lm1$coefficients$random$grp)
```

```
# Compute fit on fine grid
xg = seq(xmin, xmax, length = 200)
Zg = makeZ(xg, xl = xmin, xr = xmax, nseg = nseg, bdeg = bdeg)
fitg = cfix[1] + cfix[2] * xg + Zg %*% cran

# Compute basis matrix and inner products
B = bbase(x, xl = xmin, xr = xmax, bdeg = bdeg, nseg = nseg)
n = ncol(B)
D = diff(diag(n), diff = 2)
P = t(D) %*% D
BtB = t(B) %*% B
Bty = t(B) %*% y

# Fit with Harville-Fellner-Schall (HFS) algorithm
lambda = 1
m = length(y)
d = 2
for (it in 1:20) {
  G = BtB + lambda * P
  a = solve(G, Bty)
  mu = B %*% a
  r = y - mu
  H = solve(G, BtB)
  ed = sum(diag(H))
  sig2 = sum(r ^ 2) / (m - ed)
  tau2 = sum((D %*% a) ^ 2) / (ed - d)
  lanew = sig2 / tau2
  dla = (lanew - lambda) / lambda
  if (abs(dla) < 1e-8) break
  lambda = lanew
  cat(it, ed, dla, "\n")
}

# Compute grid on fine grid
Bg = bbase(xg, xl = xmin, xr = xmax, bdeg = 3, nseg = nseg)
yg = Bg %*% a

#titl = bquote("HFS algorithm:" ~ lambda == .(round(lambda, 2)))

# Make the plot
F1 = data.frame(x, y)
id = as.factor(rep(c("HFS", "lme"), each = length(xg)))
F2 = data.frame(x = c(xg, xg), y = c(c(yg), c(fitg)), id = id)
plt2 = ggplot() +
  geom_hline(yintercept = 0, size = 0.3) +
  geom_point(data = F1, aes(x = x, y = y), size = 1.5, color = "darkgray") +
  geom_line(data = F2, aes(x = x, y = y, col = id, linetype = id), size = 1) +
  xlab("Time (ms)") + ylab("Acceleration (g)") +
  ylim(c(-150, 100)) +
  ggtitle("Motorcycle data fit with lme and HFS algorithm") +
  labs(color = '') +
  guides(linetype = F) +
  scale_color_manual(values = c('pink', 'blue')) +
  JOPS_theme()

print(plt2)
```