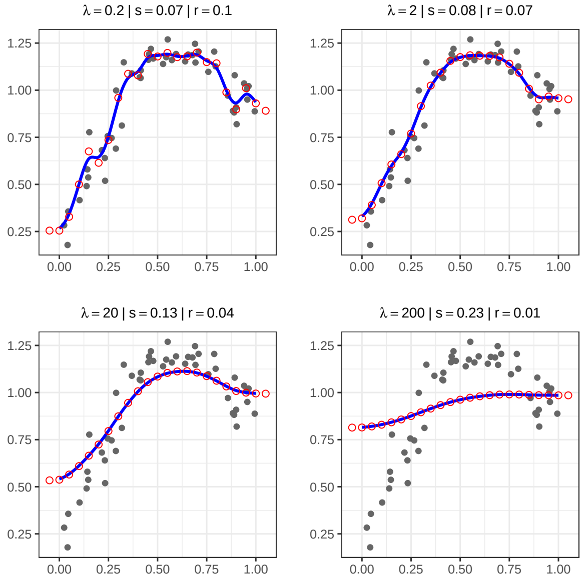
First order difference penalty in action with various tuning



The first order penalty in action for various values of λ. Also shown are the standard deviations of the residuals (s) and the roughness measure (r). Cubic B-splines, 20 segments. `R` code in `f-d1pen.R`

```
# First order difference penalty in action with various tuning
# A graph in the book ’Practical Smoothing. The Joys of P-splines’
# Paul Eilers and Brian Marx, 2019

library(ggplot2)
library(gridExtra)
library(JOPS)

# Simulate data
m = 50
set.seed(123)
x = runif(m)
y = sin(2.5 * x) + rnorm(m) * 0.1 + 0.2

# Make basis and penalty
nu = 200
u = seq(0, 1, length = nu)
nseg = 20
Bu = bbase(u, nseg = nseg)
B = bbase(x, nseg = nseg)
nb = ncol(B)
knots = ((1:nb) - 2) / nseg
n = ncol(B)
D = diff(diag(n), diff = 1)
P = t(D) %*% D
BtB = t(B) %*% B
Bty = t(B) %*% y

# Compute coefficients
A = Mu = NULL
lambdas = c(0.1, 1, 10, 100) * 2
for (lambda in lambdas) {
  a = solve(BtB + lambda * P, Bty)
  A = cbind(A, a)
}
```

```
Z = Bu %*% A
Mu = cbind(Mu, B %*% A)

# Generate the plots
plts = list()
for (j in 1:4) {
  # Compute roughness
  aj = c(A[, j])
  da = diff(aj)
  r = sqrt(sum(da ˆ 2) / (n - 1))
  r = round(r, 2)
  s = sqrt(mean((y - Mu[, j])ˆ2))
  s = round(s, 2)

  # Scaled basis
  Bsc = B %*% diag(aj)
  # Create data frames for ggplot
  Data = data.frame(x, y)
  Zf = data.frame(x = u, y = Z[, j], id  = as.factor(1))
  Bf = data.frame(x = rep(u, n), y = as.vector(Bsc),
                  id = as.factor(rep(1:n, each = nu)))
  Bf$y[Bf$y < 0.0001] = NA
  Bf = na.omit(Bf)
  titl = bquote(lambda==.(lambdas[j])˜ "|"˜s==.(s)˜"|"˜r==.(r))

  Fa = data.frame(x = knots, y = aj, id = as.factor(1))

  # Build the graphs
  plt1 = ggplot(Data, aes(x = x, y = y), ylim = c(0, 1.5)) +
    geom_point(color = ’grey40’) +
    geom_line(data = Zf, aes(x, y), color = "blue", size = 1)  +
    geom_point(data = Fa, color = "red", size = 2, shape = 1) +
    xlab("") + ylab("") +
    ggtitle(titl) +
    JOPS_theme() +
    theme(plot.title = element_text(size = rel(0.9)))

  # Add to list of plots
  plts[[j]] = plt1
}

# Plot and save
grid.arrange(grobs = plts, nrow = 2, ncol = 2)
```