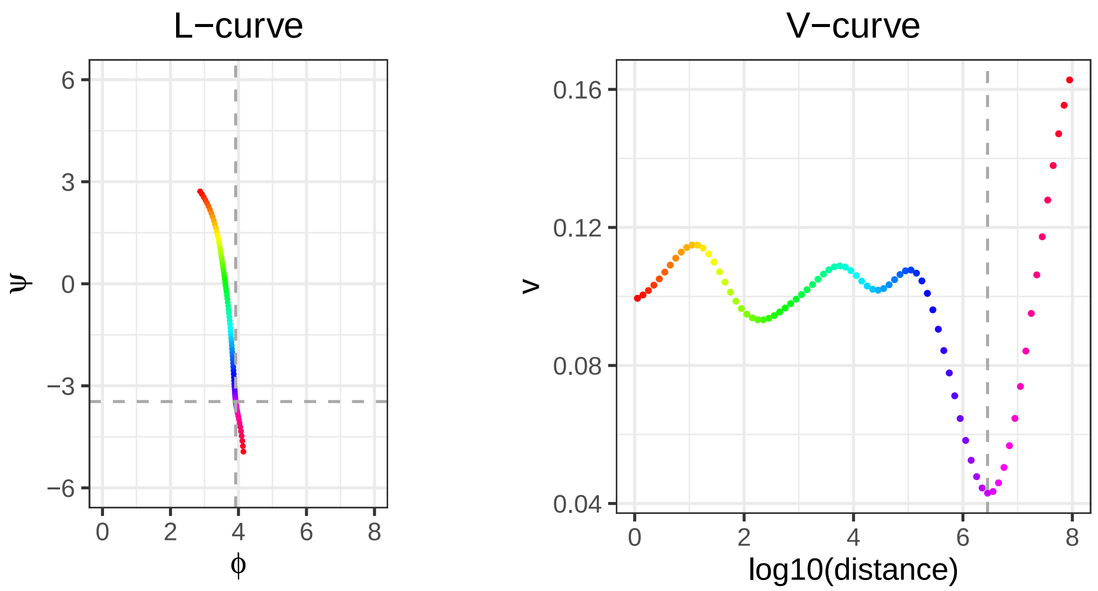
# Plot L- and V-curves for optimal tuning (Wood surface data)



L-curve and V-curve for the wood surface data. R code in `f-L-and-V-wood.R`

```
# Plot L- and V-curves for optimal tuning (Wood surface data)
# A graph in the book 'Practical Smoothing. The Joys of P-splines'
# Paul Eilers and Brian Marx, 2019

library(stringr)
library(spam)
library(gridExtra)
library(JOPS)
library(ggplot2)

# Get data
y = Woodsurf$y

m = length(y)
x = 1:m

# Prepare for smoothing
E = diag(m)
D = diff(E, diff = 2)

# Do series of lambdas
fits = pens = NULL
lambdas = 10^seq(0, 8, by = 0.1)
for (lambda in lambdas) {
    z = solve(E + lambda * t(D) %*% D, y)
    fit = log10(sum((y - z)^2))
    pen = log10(sum((D %*% z)^2))
    fits = c(fits, fit)
    pens = c(pens, pen)
}

# Compute V-curve
dfits = diff(fits)
dpens = diff(pens)
v = sqrt(dfits^2 + dpens^2)
nla = length(lambdas)
midla = sqrt(lambdas[-1] * lambdas[-nla])
```

```
k = which.min(v)

# Do final smooth
lambda = midla[k]
z = solve(E + lambda * t(D) %*% D, y)
fitopt = log10(sum((y - z)^2))
penopt = log10(sum((D %*% z)^2))

# Make plots
graphics.off()
par(mfrow = c(1, 2), mar = c(3.5, 3.5, 2, 1), mgp = c(1.6, 0.8, 0))

#-----
lamopt=log10(lambda)
log10lam=log10(midla)
F1 = data.frame(x=fits, y=pens)
F2 = data.frame(y=v, x=log10lam)
plt1 = ggplot(F1, aes(x , y)) +
  geom_point(size=0.3, color = rainbow(nla)) +
  ggtitle('L-curve') +
  geom_vline(xintercept=fitopt, lty=2, color = 'darkgrey')+
  geom_hline(yintercept=penopt, lty=2, color = 'darkgrey') +
  xlab(expression(phi))+
  ylab(expression(psi))+
  coord_fixed(ratio = 1, xlim=c(0,8), ylim=c(-6,6))+
  JOPS_theme()

plt2 = ggplot(F2, aes(x , y)) +
  geom_point(size=0.5, , color = rainbow(nla - 1)) +
  ggtitle('V-curve') +
  geom_vline(xintercept=lamopt, lty=2, color = 'darkgrey')+
  #geom_hline(yintercept=penopt) +
  xlab('log10(distance)')+
  ylab('v')+
  JOPS_theme()


grid.arrange(plt1, plt2, ncol = 2, nrow = 1)
```