Stat 312/612 HW9.4.iv

Load the library and the diabetes data set: `library(lars); data(diabetes)`

```
## Loaded lars 1.2
```

Run lars, then collect the information needed for the pictures:

```
    X <- diabetes$x; y <- diabetes$y
    out <- lars(X,y,type="lasso")
    lambda <- c(out$lambda,0)   # note the zero
    coeffs <- data.frame(out$beta)
    ell1 <- rowSums(abs(coeffs))
    fit <- data.frame(lambda, RSS=out$RSS, ell1, coeffs)
# I prefer to think of lambda increasing when plotting:
    fit <- fit[order(fit$lambda),]
    is.coeff <- match(names(coeffs),names(fit)) # to avoid editing errors
    print(fit,digits=2) # rownames give LARS stage nuumber
```

```
##    lambda     RSS ell1   age  sex bmi map   tc ldl  hdl tch ltg glu
## 12    0.0 1263983 3460 -10.0 -240 520 324 -792 477  101 177 751  68
## 11    1.3 1264765 2863  -7.0 -237 521 322 -580 314    0 140 675  67
## 10    2.2 1264977 2802  -5.7 -234 523 320 -554 287    0 149 663  66
## 9     5.1 1269390 2196   0.0 -227 526 315 -237  34 -135 111 545  65
## 8     5.5 1270233 2116   0.0 -226 527 314 -195   0 -152 106 530  64
## 7    20.0 1275355 1915   0.0 -198 522 297 -104   0 -224   0 515  55
## 6    69.0 1308932 1537   0.0 -112 512 253    0    0 -196   0 452  12
## 5    88.8 1324118 1441   0.0  -75 511 234    0    0 -170   0 451   0
## 4   130.1 1365734 1251   0.0    0 506 191    0    0 -114   0 440   0
## 3   316.1 1527165  889   0.0    0 435  79    0    0    0   0 375   0
## 2   452.9 1700369  664   0.0    0 362   0    0    0    0   0 302   0
## 1   889.3 2510465   60   0.0    0  60   0    0    0    0   0   0   0
## 0   949.4 2621009    0   0.0    0   0   0    0    0    0   0   0   0
```

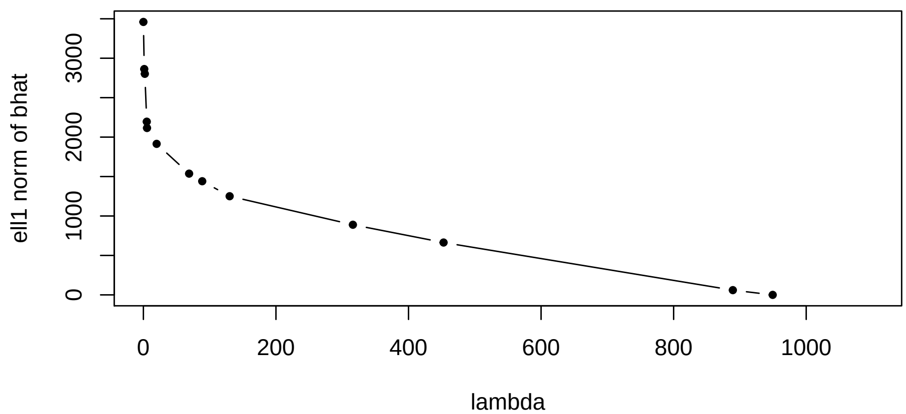
Remark. Isn't it interesting that the ℓ_1 penalty has only a small effect on RSS for λ near zero.

Remember that $\widehat{b}(\lambda)$ denotes the vector of coefficients that minimizes

$$G_\lambda(b) = \tfrac{1}{2}Q(b) + \lambda\|b\|_1 = \tfrac{1}{2}\|y - Xb\|_2^2 + \lambda\sum\nolimits_j |b_j|.$$

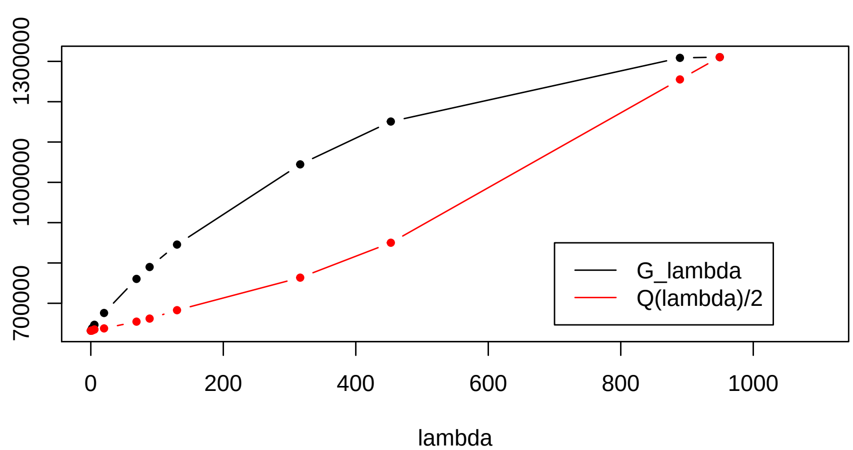
According to Lasso.pdf, $\lambda \mapsto \widehat{b}(\lambda)$ should be piecewise linear and continuous, with $\widehat{b}(\lambda) = 0$ for large enough λ. For the diabetes data set, large enough means $\lambda \geq 949.4$. The function $\ell(\lambda) = \|\widehat{b}(\lambda)\|_1$ is also piecewise linear and decreasing, taking the value zero for $\lambda \geq 949.4$:

```
plot(fit$lam,fit$ell1,type="b",pch=20,xlim=c(0,1100),
    ylab = "ell1 norm of bhat", xlab="lambda")
```



The values of $q(\lambda) = Q(\widehat{b}(\lambda))$ and $m(\lambda) = G_\lambda(\widehat{b}(\lambda)) = \frac{1}{2}q(\lambda) + \ell(\lambda)$ are easy to recover at the values of λ that are important to LARS:

```
plot(fit$lam,fit$lam*fit$ell1 + fit$RSS/2,
    type="b",pch=20,xlim=c(0,1100),
    ylab = "", xlab="lambda" )
points(fit$lam,fit$RSS/2,type="b",pch=20,col="red")
legend(700, 850000,
    leg = c("G_lambda","Q(lambda)/2"),lty=c(1,1),
    col= c("black","red"))
```



The last picture is misleading because it suggests that both $q(\lambda)$ and $m(\lambda)$ are piecewise linear, which they are not. It takes a lot more effort to get the actual (quadratic) behaviour between the values of λ that are important to LARS.

The vector of residuals, $R(\lambda) = y - X\widehat{b}(\lambda)$, should also be piecewise linear and continuous, with

$$q(\lambda) = Q\left(\widehat{b}(\lambda)\right) = \|R(\lambda)\|_2^2$$

an increasing function of λ that stays constant for $\lambda \geq 949.4$.

Remark. The LARS procedure implicitly fits an intercept term for the diabetes data. All the predictors are centered to have zero means. To reproduce the residual sums of squares in `out$RSS` I needed to center the response y as well.

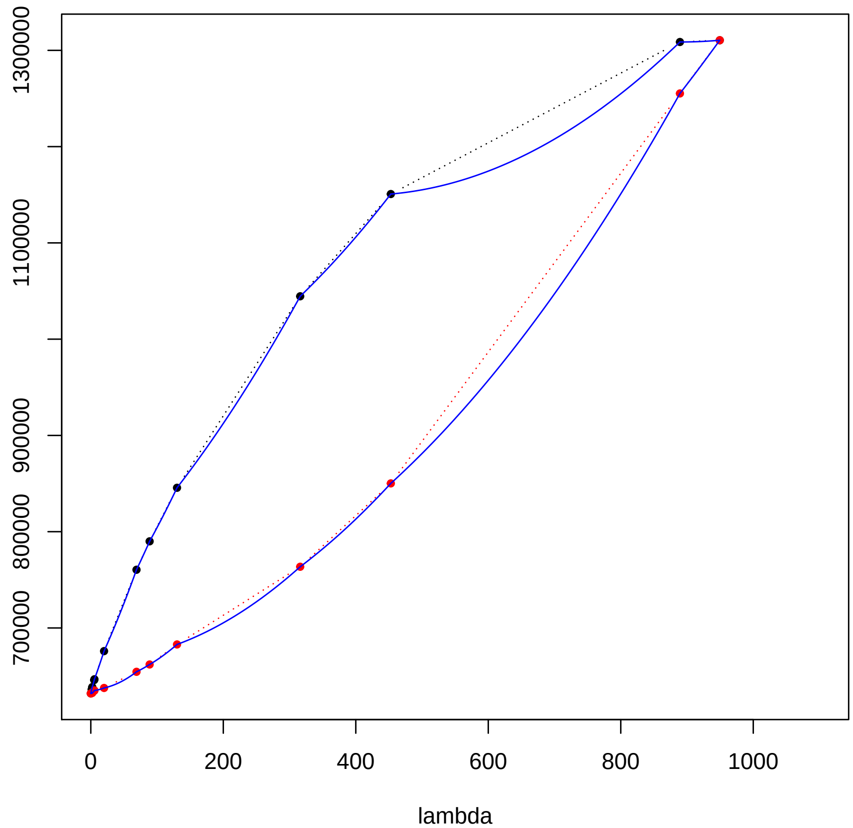
```
Resid <- data.frame(y - mean(y) - X %*% t(fit[,is.coeff]))
```

Note that `max(abs( colSums(Resid^2)-fit$RSS))`

Let $0 = \lambda_0 < \lambda_1 < \cdots < \lambda_k$ denote the values in the `fit$lambda`. (For the diabetes data, $k = 12$.) How does $q(\lambda)$ behave in each of the the intervals $\lambda_i < \lambda < \lambda_{i+1}$? Write R_i for $R(\lambda_i)$ and D_i for $R_{i+1} - R_i$. Also define $\Delta_i = \lambda_{i+1} - \lambda_i$. Then $R(\lambda_i + t\Delta_i) = R_i + tD_i$ for $0 \leq t \leq 1$, so that

$$q(\lambda_i + t\Delta_i) = \|R_i\|_2^2 + 2t\langle R_i, D_i\rangle + t^2\|D_i\|_2^2 \qquad \text{for } 0 \leq t \leq 1.$$

```
# Repeat the previous plot.
plot(fit$lam,fit$lam*fit$ell1 + fit$RSS/2,
    type="b",pch=20,xlim=c(0,1100),
    ylab = "", xlab="lambda",lty=3 )
points(fit$lam,fit$RSS/2,type="b",pch=20,col="red",lty=3)
# Then add quadratic parts.
tgrid <- seq(0,1,by=0.01)
for (ss in 1:12){
    RR <- Resid[,ss]
    DD <- Resid[,ss+1] - Resid[ ,ss]
    Del <- fit$lambda[ss+1] - fit$lambda[ss]
    innerp <- sum(RR*DD)  # the inner product term
    lam.grid <- Del * tgrid
    QQ <- fit$RSS[ss] + 2* tgrid*innerp + tgrid^2 * sum(DD^2)
    LL <- (1-tgrid)*fit$ell1[ss]*fit$lambda[ss] +
        tgrid*fit$ell1[ss+1]*fit$lambda[ss+1]
    points(fit$lambda[ss] +  lam.grid, QQ/2,
        col = "blue",type="l")
    points(fit$lambda[ss] +  lam.grid, LL + QQ/2 ,
        col = "blue",type="l")
}
```



It seems hardly worth the extra effort.