MA 589 Project 2 Solutions

Luis Carvalho

1

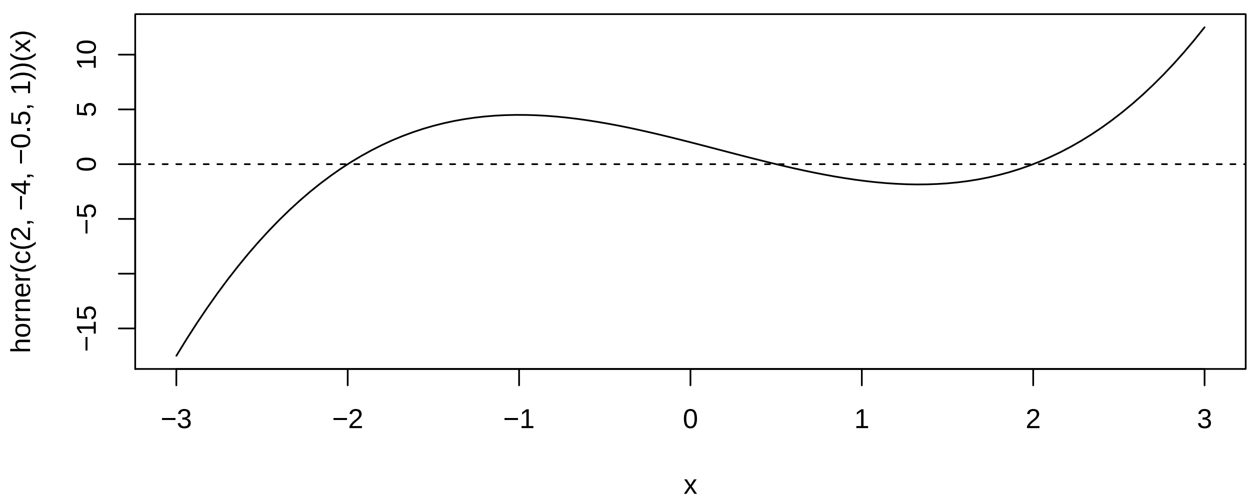
(a)

To evaluate `c(3, -2, 1)`, that is, $p(x) = 3 - 2x + x^2$, the scheme does $p(x) = 3 + x(-2 + 1 \cdot (x))$. Thus, the whole idea behind Horner's scheme is to explore a recurrent formulation:

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = a_0 + x(a_1 + x(a_2 + \cdots x(a_{n-1} + x \cdot (a_n)) \cdots)).$$

(b)

```
horner <- function (coef)
  function (x) Reduce(function(v, s) s * x + v, coef, 0, right = TRUE)
curve(horner(c(2, -4, -.5, 1))(x), from = -3, to = 3)
abline(h = 0, lty = 2)
```



(c)

```
# Derivative of polynomial `coef`
pderiv <- function (coef) (coef * seq(0, length(coef) - 1))[-1]

# Newton's method for polynomial `coef`, starting at `start`
proot <- function (coef, start = 0, max.it = 25, epsilon = 1e-8) {
  p <- horner(coef)
  dp <- horner(pderiv(coef))
  x <- start; px <- p(x)
  for (it in 1:max.it) {
    x.new <- x - px / dp(x)
```

```r
    if (!is.finite(x.new)) stop("Reached critical point at x = ", x)
    px.new <- p(x.new)
    if (abs(px - px.new) < epsilon) break
    x <- x.new; px <- px.new
  }
  x
}

coef <- c(2, -4, -.5, 1)
proot(coef, -1.5)
```

```
[1] -2
```

```r
proot(coef, 0)
```

```
[1] 0.5
```

```r
proot(coef, 1.5)
```

```
[1] 2
```

```r
proot(coef, -1) # -1 is local maximum, p'(-1) = 0
```

```
Error in proot(coef, -1): Reached critical point at x = -1
```

(d)

```r
legendre <- local({
  L <- list(c(1), c(0, 1)) # L_0, L_1
  function (n) {
    if (n + 1 <= length(L)) return(L[[n + 1]]) # in cache?
    message("computing Le_", n)
    un <- (n - 1) / n
    Ln <- (1 + un) * c(0, legendre(n - 1)) - un * c(legendre(n - 2), 0, 0)
    L[[n + 1]] <<- Ln # memoize
    Ln
  }
})

# Test
legendre(4)
```

```
computing Le_4

computing Le_3

computing Le_2

[1]  0.375  0.000 -3.750  0.000  4.375
```

```r
legendre(7)
```

```
computing Le_7

computing Le_6

computing Le_5
```

```
[1]  0.0000  -2.1875   0.0000  19.6875   0.0000 -43.3125   0.0000  26.8125
```

```r
legendre(6)
```

```
[1]  -0.3125   0.0000   6.5625   0.0000 -19.6875   0.0000  14.4375
```

2

(a)

```r
x <- seq(-3, 3, by = 0.5)
X <- outer(x, 0:4, `^`) # X = vandermonde(x, 4)
y <- c(-17, -7, .5, 3, 5, 4, 2.5, -.5, -2, -2, .5, 5, 12)
(T <- crossprod(cbind(X, y))) # tableau
```

```
                                                          y
   13.000   0.000   45.500    0.0000   284.3750    4.0000
    0.000  45.500    0.000  284.3750     0.0000  100.2500
   45.500   0.000  284.375    0.0000  2099.0938  -47.3750
    0.000 284.375    0.000 2099.0938     0.0000  946.0625
  284.375   0.000 2099.094    0.0000 16739.0234 -458.8438
y   4.000 100.250  -47.375  946.0625  -458.8438  572.0000
```

(b)

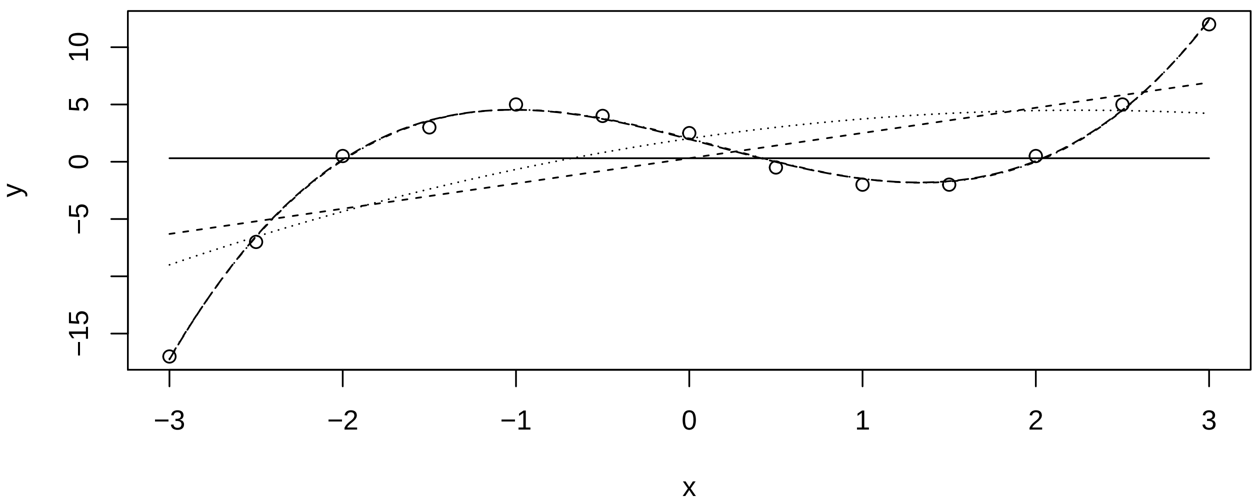
```r
SWEEP <- function (A, k) {
  D <- A[k, k]; A[k,] <- A[k,] / D
  for (i in 1:nrow(A)) {
    if (i != k) {
      B <- A[i,k]; A[i,] <- A[i,] - B * A[k,]
      A[i, k] <- - B / D
    }
  }
  A[k, k] <- 1 / D
  A
}
# check
near <- function (x, y, tol = 1e-9) abs(x - y) <= tol * max(abs(x), abs(y))
sweep_check <- TRUE
for (k in 1:5)
  sweep_check <- sweep_check & all(near(SWEEP(SWEEP(T, 1), 1), T))
sweep_check # SWEEP(SWEEP(T, k)) == T for k = 1,..., 5?
```

```
[1] TRUE
```

(c)

```r
plot(x, y) # plot data
a <- seq(-3, 3, length.out = 100)
T <- crossprod(cbind(X, y)) # tableau
```

```r
p <- ncol(T) - 1 # number of covariates (each monomial)
for (i in 1:p) { # for each covariate
  T <- SWEEP(T, i) # include i-th covariate, x^(i - 1), in the model
  lines(a, horner(T[1:i, p + 1])(a), lty = i) # plot fit
  print(c(i, T[p + 1, p + 1])) # report RSS
}
```

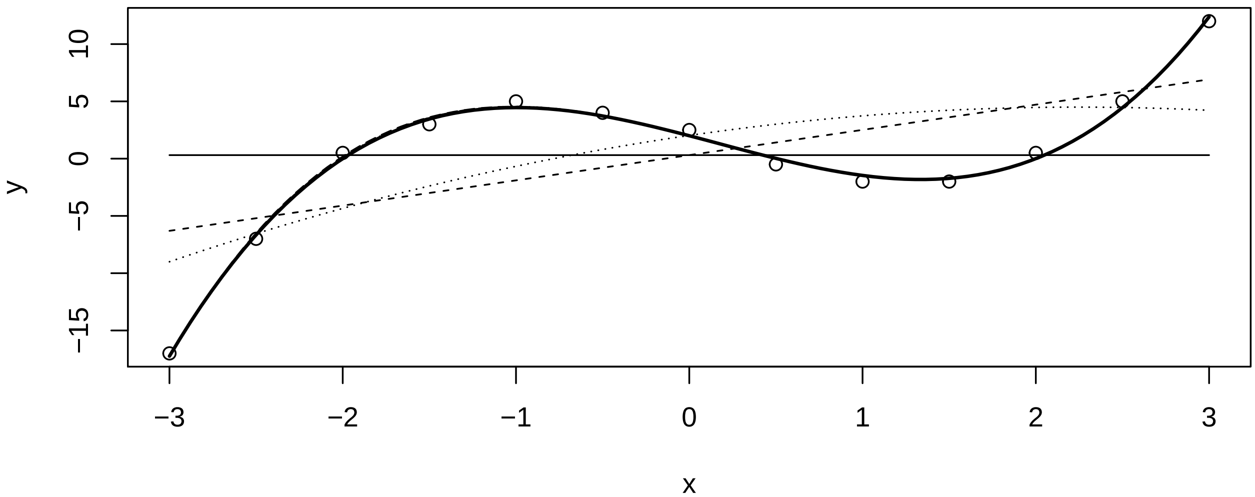



```
[1]   1.0000 570.7692
[1]   2.0000 349.8887
[1]   3.0000 319.7837
[1] 4.000000 2.517982
[1] 5.000000 2.486895
```

(d)

```r
# extended tableau
L <- rbind(c(1, -2, 4, -8, 16), c(1, 2, 4, 8, 16))
pl <- p + nrow(L)
T <- crossprod(cbind(X, matrix(0, nrow(X), nrow(L)), y))
T[(p + 1):pl, 1:p] <- L; T[1:p, (p + 1):pl] <- t(L)

plot(x, y)
for (i in 1:p) {
  T <- SWEEP(T, i)
  lines(a, horner(T[1:i, pl + 1])(a), lty = i)
}
rss <- T[pl + 1, pl + 1]
for (i in ((p + 1):pl)) T <- SWEEP(T, i)
beta.hat <- T[1:p, pl + 1]
lines(a, horner(beta.hat)(a), lwd = 2)
```




```r
(rss0 <- T[pl + 1, pl + 1])
```

```
[1] 2.587958
```

```r
near_zero <- function (x, tol = 1e-9) abs(x) <= tol
all(near_zero(L %*% beta.hat)) # L * beta.hat == 0?
```

```
[1] TRUE
```

```r
# RSS (from unconstrained fit) is close to RSS0 (from constrained)
# formal comparison via F-test leads to failure to reject:
(F <- (rss0 - rss) / nrow(L) / (rss / (length(y) - p)))
```

```
[1] 0.1625518
```

```r
(pval_F <- pf(F, nrow(L), length(y) - p, lower.tail = F))
```

```
[1] 0.85271
```

3

(a)

Straightforward from the ADJUST operations followed by in-place SWEEP assignments in x-column.

(b)

The xx-block of Σ after SWEEPing the yy-block is the Schur complement of the block, $\Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^\top$. But then, after a new SWEEP on the xx-block, the matrix becomes Σ^{-1} since all rows have been swept and the xx-block gets inverted, so $(\Sigma^{-1})_{xx} = (\Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^\top)^{-1}$ and the Woodbury identity follows from $(\Sigma^{-1})_{xx}$ in the first series of SWEEPs.

(c)

$$\begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} & \mu_x - \mathbf{x} \\ \Sigma_{xy}^\top & \Sigma_{yy} & \mu_y \end{bmatrix} \xleftrightarrow[\text{xx-block}]{SWEEP} \begin{bmatrix} \Sigma_{xx}^{-1} & \Sigma_{xx}^{-1}\Sigma_{xy} & \Sigma_{xx}^{-1}(\mu_x - \mathbf{x}) \\ -\Sigma_{xy}^\top\Sigma_{xx}^{-1} & \Sigma_{yy} - \Sigma_{xy}^\top\Sigma_{xx}^{-1}\Sigma_{xy} & \mu_y - \Sigma_{xy}^\top\Sigma_{xx}^{-1}(\mu_x - \mathbf{x}) \end{bmatrix}$$