A Grain Data Study with Orthogonal Polynomials

by Marek Rychlik

March 25, 2009

Data definition

The measured variate is grain yield, subject to several treatment levels of a fertilizer. In this experimental study, the treatment is a quantitative factor (fertilizer level). This is an **ordered** factor. R treats ordered factors differently from unordered factors: it automatically assigns contrasts, which are discrete orthogonal polynomials. Manually, these contrasts can be generated by calling `contr.poly`.

```
> t = 5; r = 3; N = r * t
```

```
> Yield = as.vector(matrix(c(12.2, 16.0, 18.6, 17.6, 18.0,   11.4,
  15.5, 20.2, 19.3, 16.4,   12.4, 16.5, 18.2, 17.1, 16.6), byrow =
  T, nrow = r, ncol = t))
```

```
> PlantDensity = gl(t, r, labels = seq(10, 50, 10), ordered = T)
```

```
> PlantDensityLevels = as.numeric(levels(PlantDensity))
```

```
> yield.data = data.frame(PlantDensity, Yield)
```

```
> yield.data
```

```
   PlantDensity Yield
1            10  12.2
2            10  11.4
3            10  12.4
4            20  16.0
5            20  15.5
6            20  16.5
7            30  18.6
8            30  20.2
9            30  18.2
10           40  17.6
11           40  19.3
12           40  17.1
13           50  18.0
14           50  16.4
15           50  16.6
```

```
>
```

A plot of the data

Setting up TEXMACS plotting parameters

```
> opts = options(); opts$texmacs$width=7.5;
  opts$texmacs$height=7.5; opts$texmacs$nox11=F; options(opts)
```

```
>
```

Setting up X Windows plotting parameters

```
> X11(pointsize=8, height=3, width=3)
```

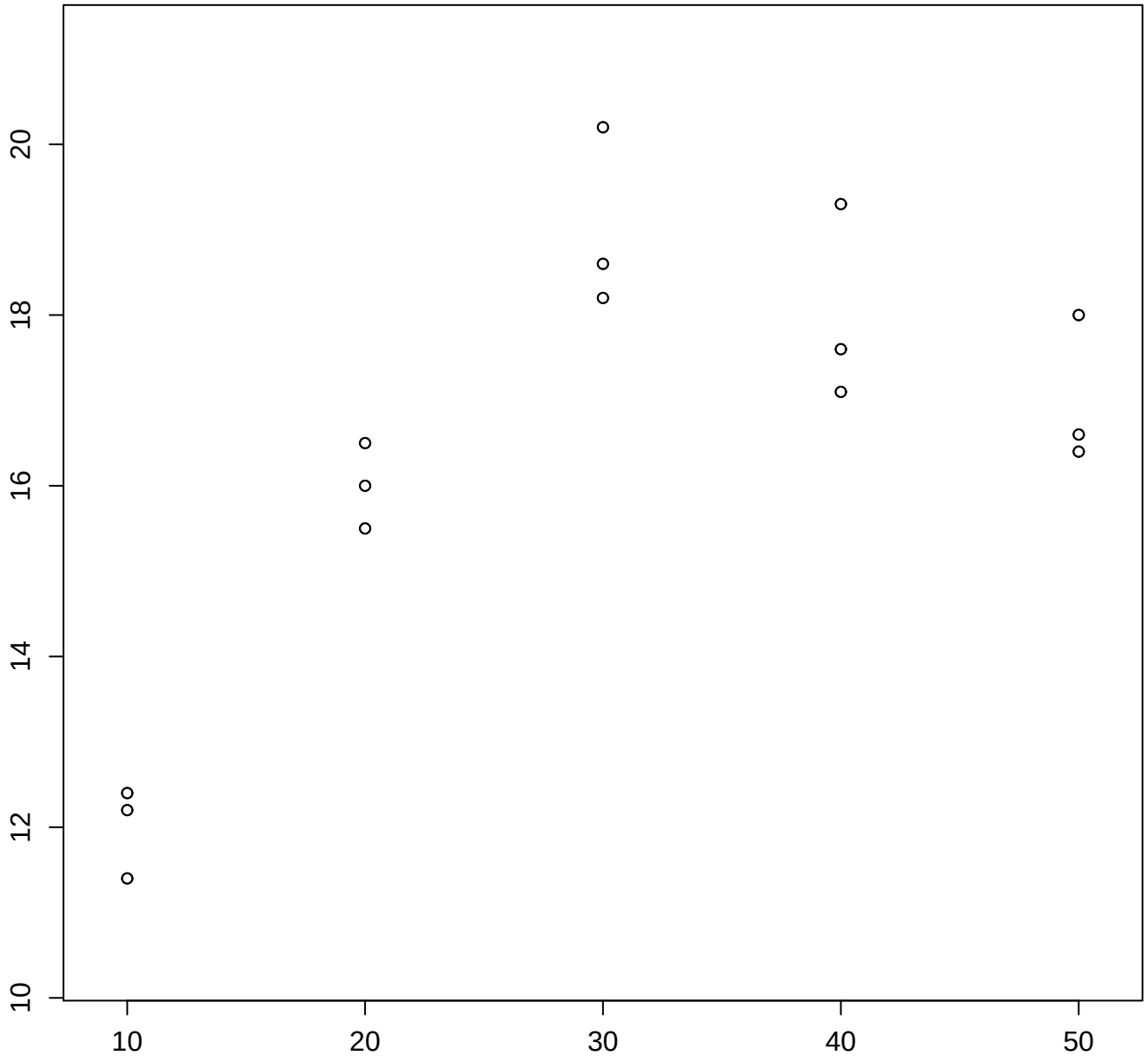
```
>
```

The plot

```
> xadd = 1; yadd = 1
```

```
> plot(numeric(0), numeric(0), xlim =
  range(PlantDensityLevels)+c(-1,1)*xadd, ylim = range(Yield)+c(-
  1,1)*yadd, type = "n", ann = FALSE)
```

```
> points(PlantDensityLevels[as.numeric(PlantDensity)], Yield); v()
```


10
12
14
16
18
20
10
20
30
40
50

```
>
```

Analysis of variance

This is standard:

```
> yield.aov = aov(Yield ~ PlantDensity, yield.data, projections =
  T, qr = T)
```

```
> summary(yield.aov)
```

```
             Df Sum Sq Mean Sq F value    Pr(>F)
PlantDensity  4 87.600  21.900  29.278 1.690e-05 ***
Residuals    10  7.480   0.748

---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05
âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
>
```

Fast sums of squares for contrasts and residuals

In this discussion, we utilize the function `proj` which does the heavy lifting of

decomposing experimental error and between-groups variation along the contrast directions. **All** sums of squares used in our analysis are the sums of squares of the columns of the matrix component of the data structure returned by `proj`.

```
> yield.proj = proj(yield.aov, onedf = T)
```

```
> SS = as.matrix(apply(yield.proj^2, 2, sum))
```

```
> (SSC = SS[2:t,])
```

```
PlantDensity.L PlantDensity.Q PlantDensity.C PlantDensity^4
          43.2           42.0            0.3            2.1
```

```
> (SST = sum(SSC))
```

```
[1] 87.6
```

```
> (SSE = SS[t+1,])
```

```
Residuals
```

```
7.48
```

```
>
```

F-statistic for contrasts and the omnibus

We utilize the explicit expressions for SSC and other sums of squares. We process all contrasts in parallel, taking advantage of processing of vector components in parallel, offered by the R arithmetic operator and functions.

```
> MSC = SSC
```

```
> MSE = SSE / (N - t)
```

```
> (F.contrasts = MSC / MSE)
```

```
PlantDensity.L PlantDensity.Q PlantDensity.C PlantDensity^4
    57.7540107     56.1497326      0.4010695      2.8074866
```

```
> MST = SST / (t - 1)
```

```
> (F.omnibus = MST / MSE)
```

```
Residuals
 29.27807
```

```
>
```

F-tests for each contrast - significance levels

We obtain all significance level simultaneously.

```
> (sig.contrasts = pf(F.contrasts, df1 = 1, df2 = N - t,
  lower.tail = F))
```

```
PlantDensity.L PlantDensity.Q PlantDensity.C PlantDensity^4
  1.840830e-05   2.078706e-05   5.407497e-01   1.247622e-01
```

```
> (sig.omnibus = pf(F.omnibus, df1 = t - 1, df2 = N - t,
  lower.tail = F))
```

```
   Residuals
1.689528e-05
```

```
>
```

Check with ordinary ANOVA

We can see that our results, within numerical precision, agree with the results of the omnibus F-statistic summarized by `aov` and `summary`.

```
> summary(yield.aov)
```

```
             Df Sum Sq Mean Sq F value    Pr(>F)
PlantDensity  4 87.600  21.900  29.278 1.690e-05 ***
Residuals    10  7.480   0.748

---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05
âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
>
```

Plots

The coefficients of the polynomial approximation are obtained by calling `coef` on the result of `aov`. The functions `poly` and `predict` are used for the following two tasks:

- To obtain the values of the orthogonal polynomials at the abscissas which are the levels of the factor
- To interpolate the values of the orthogonal polynomials for a much finer mesh, on which we would like to approximate the approximating polynomial

The evaluation of a polynomial expressed as a linear combination of the coefficients is performed using matrix product. More precisely, if the approximating polynomial is given by the formula

$$p(x) = \beta_0 + \sum_{i=1}^{t-1} \beta_i P_i(x)$$

with the underlying values of x to be x_1, x_2, ..., x_t, then the values $p(x_j)$, $j = 1$, $2, ..., x_t$ are given by

$$\begin{pmatrix} p(x_1) \\ p(x_2) \\ \vdots \\ p(x_t) \end{pmatrix} = \begin{pmatrix} P_1(x_1) & P_2(x_1) & ... & P_{t-1}(x_1) \\ P_1(x_2) & P_2(x_2) & ... & P_{t-1}(x_2) \\ \vdots & \vdots & & \vdots \\ P_1(x_t) & P_2(x_t) & ... & P_{t-1}(x_t) \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{t-1} \end{pmatrix} + \beta_0 \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

The $t \times t$ coefficient matrix in this formula is returned by the function `poly` when called with one argument (the vector (x_1, x_2, ..., x_t)). A call to `predict` returns a similar matrix, but with an arbitrary collection of points (not necessarily the underlying x_i's). This allows us to interpolate the approximating polynomial on a fine grid.

```
> co = coef(yield.aov);
```

```
> x0 = as.numeric(as.vector(PlantDensity)); x1 =
  as.numeric(levels(PlantDensity)); range.x = range(x1); range.y =
  range(Yield); step.x = 1
```

```
> x = seq(range.x[1], range.x[2], step.x)
```

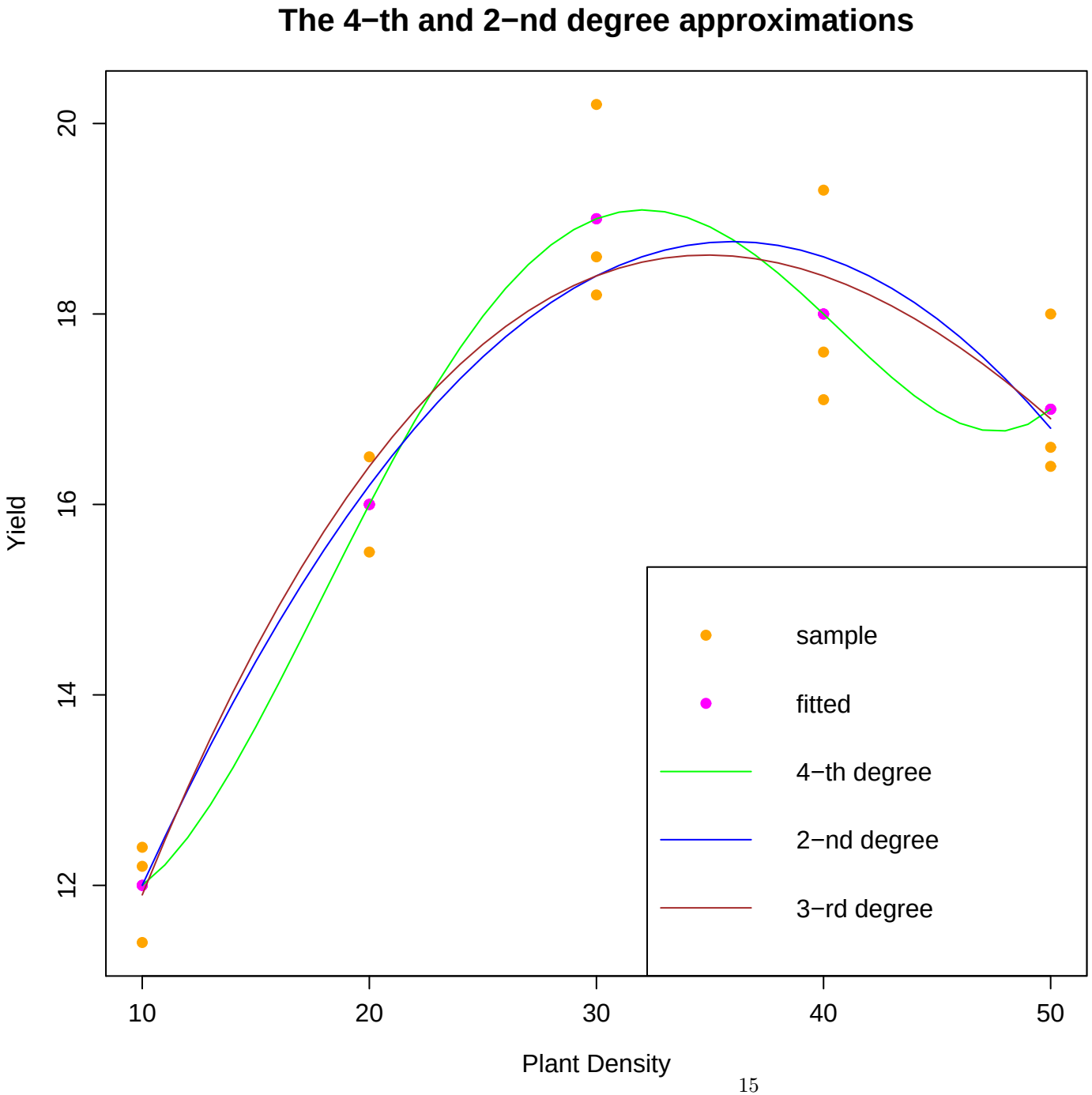
```
> poly.coefs = poly(x1, degree = t - 1)
```

```
> pred = predict(poly.coefs, x); y.full = cbind(1, pred) %*% co;
  y.quadratic = cbind(1, pred) %*% c(co[1:3], 0, 0); y.cubic =
  cbind(1, pred) %*% c(co[1:4], 0)
```

```
> plot(range.x, range.y, type = "n", xlab = "Plant Density", ylab
  = "Yield", main = "The 4-th and 2-nd degree approximations")
```

```
> points(x0, Yield, col = "orange", pch = 19); points(x0,
  fitted(yield.aov), type = "p", col = "magenta", pch = 19);
  points(x, y.full, type = "l", col = "green", pch = 24);
  points(x, y.quadratic, type = "l", col = "blue", pch = 25);
  points(x, y.cubic, type = "l", col = "brown", pch = 25)
```

```
> legend("bottomright",
  col=c("orange","magenta","green","blue","brown"), lty = c(-1,-
  1,1,1,1), pch = c(19,19,-1,-1,-1), c("sample", "fitted", "4-th
  degree", "2-nd degree","3-rd degree")); v()
```


The 4−th and 2−nd degree approximations
Yield
Plant Density
12
14
16
18
20
10
20
30
40
50
sample
fitted
4−th degree
2−nd degree
3−rd degree

```
>
```