Lauri Mehtätalo and Juha Lappi

Biometry for Forestry and Environmental Data: With Examples in R

Full Versions of the Web Examples

Contents

Foreword

This document includes full versions of the Web Examples of our textbook "Biometry for Forestry And Environmental Data: With Examples in R", which was published by Chapman Hall / CRC Press in 2020.

To keep our textbook within the page limitations agreed with the publisher, we had to leave out quite many of the examples of the original textbook manuscript from the published textbook. However, we think that those examples may be of use for many readers of the book, and therefore we decided to make this web extension to the book. This document is freely available from the textbook website at

`http://www.biombook.org`.

The website includes also the original R-codes for all examples of the textbook and of this document.

Throughout this document, we refer to the main textbook Mehtätalo and Lappi (2020) by acronym "ML2020". Numbering of the chapters and examples follows that of ML2020. Whenever the text refers to material (figures, equation, pages) of ML2020, the textbook acronym is explicitly mentioned. If the textbook is not mentioned, the reference is to this document.

Joensuu and Suonenjoki, June 2020

Lauri Mehtätalo and *Juha Lappi*

2

Random Variables

Example 2.5. A commonly used distribution function with tree diameter data is the *Weibull*(α, β) distribution. The two-parameter Weibull distribution function is (also a three parameter version is used by foresters)

$$F(x|\alpha,\beta) = 1 - \exp\left[-\left(\frac{x}{\beta}\right)^{\alpha}\right] \qquad x > 0, \alpha, \beta > 0\,, \tag{2.1}$$

where α and β are the shape and scale parameters, respectively.

The Weibull density is obtained by differentiating the *cdf* in (2.1) with respect to x

$$\begin{aligned} f(x|\alpha,\beta) &= F'(x|\alpha,\beta) && (2.2)\\ &= -\exp\left\{-\left(\frac{x}{\beta}\right)^{\alpha}\right\}\left\{-\alpha\left(\frac{x}{\beta}\right)^{\alpha-1}\frac{1}{\beta}\right\} && (2.3)\\ &= \frac{\alpha}{\beta^{\alpha}}x^{\alpha-1}e^{-(x/\beta)^{\alpha}}\,, && (2.4)\end{aligned}$$

The bottom graphs of Figure 2.1 on page 2 show the *cdf* and *pdf* of *Weibull*$(5, 15)$ distribution.

Example 2.6. The distribution of ALS return heights on a single tree crown. In airborne laser scanning (ALS), a forested landscape is explored from an aircraft through laser measurement techniques. A laser scanning device has been installed to an aircraft and it uses narrow laser pulses that measure distance from the aircraft to the forest canopy. If the flying altitude is low and the laser beam is very narrow so that the footprint on the canopy can be realistically modeled as a point, the measurements provided by the scanner are essentially observations of forest canopy height at points within the forested area. Alternatively, if the beam is wide and altitude is high and whole distribution of the returned energy is recorded, the scanner can provide a measurement of the full distribution of canopy heights within the footprint.

Motivated by ALS, and to illustrate how non-standard distributions could be derived, consider the distribution of crown height within a tree crown. Let us describe the radius of a tree crown between the height of maximal crown radius and total height h by an ellipse centered at (x_0h, y_0h), $x_0 < 1, y_0 \leq 0$, with the half-axes ah and bh. However, assume that the observable crown radius remains constant below the level of maximal crown width, even though the actual radius decreases. therefore, the crown radius y at height z $(0 \leq z \leq h)$ above the ground level could be described as

$$y(z) = \begin{cases} h\,(y_0 + b) & z \leq x_0 h \\ h\left(y_0 + b\sqrt{1 - \frac{\left(\frac{z}{h} - x_0\right)^2}{a^2}}\right) & x_0 h < z \leq h \end{cases}\,, \tag{2.5}$$

where x_0, y_0, a and b are parameters. However, we define $a = \sqrt{\frac{b^2(1-x_0)^2}{b^2 - y_0^2}}$, which ensures that $y(z, h)$ passes through point $(h, 0)$, the tree tip. The assumed function is illustrated in the top left subfigure of Figure 2.2 for a tree with $h = 20$ and shape parameters $x_0 = 0.2$, $y_0 = -0.1$, and $b = 0.25$. The three-dimensional crown results by rotating this function about the x-axis. The cross-sectional area of the crown at height z is then $A(z) = \pi y(z)^2$. Its maximum value $A_{max} = \pi h^2 (y_0 + b)^2$ is obtained when $z < x_0 h$.

Assume that ALS returns are modeled as the height of canopy surface at given points. For

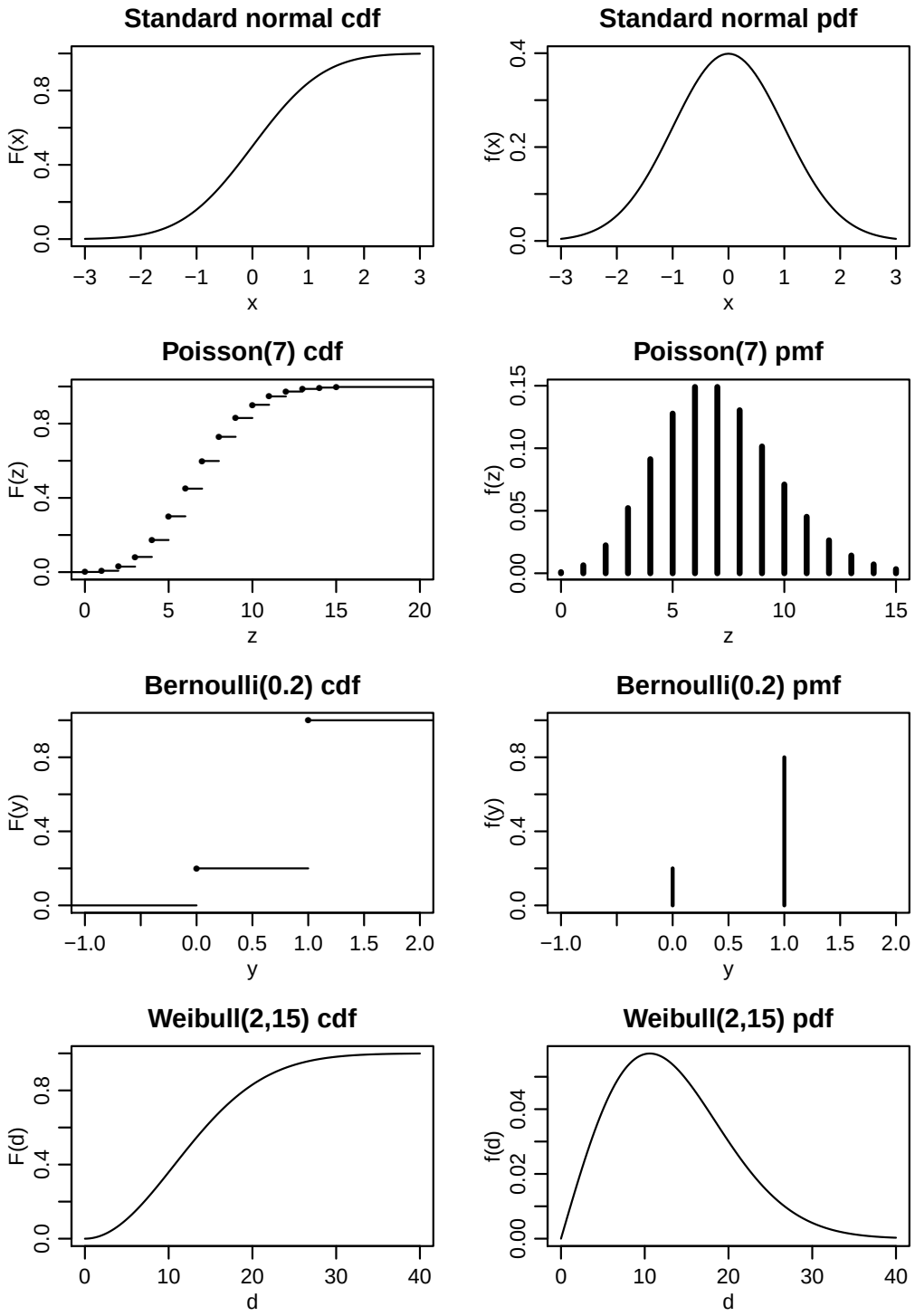


FIGURE 2.1 Examples of four different cumulative distribution functions (*cdf*) (on the left) and the corresponding probability mass function (*pmf*) or probability density functions (*pdf*) (on the right).

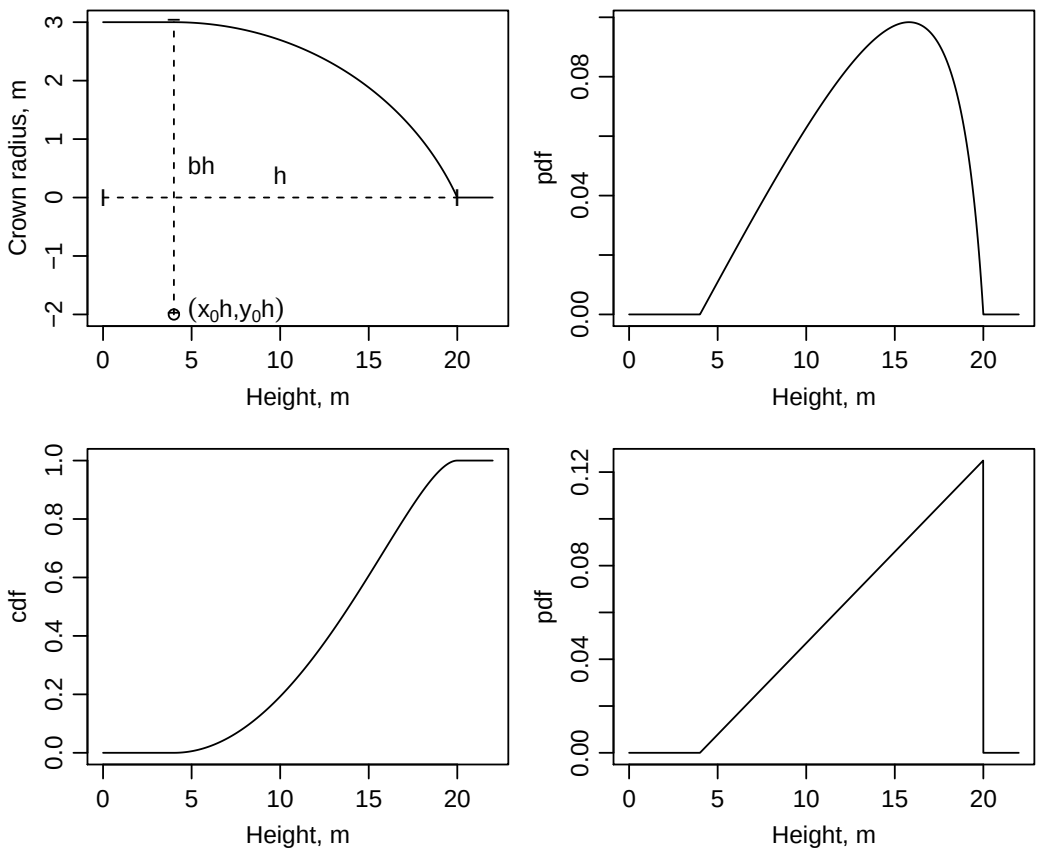


FIGURE 2.2 Illustration for example 2.6 using parameter values $h = 20$, $x_0 = 0.2$, $y_0 = -0.1$, and $b = 0.25$. The upper left subfigure shows the observable crown radius as a function of height. The cdf of crown height observations is shown in the lower left figure, and the corresponding density in the upper right figure. The lower right figure illustrates the density when $y_0 = 0$.

simplicity, assume that the tree crown is viewed directly from above. Let random variable Z describe the height of the observable tree crown at random location within a horizontal projection of the crown. The *cdf* of Z is the proportion of the cross-sectional area below z:

$$F_Z(z) = P(Z \le z) = \frac{A_{max} - A(z)}{A_{max}} = 1 - \frac{A(z)}{A_{max}}$$
$$= 1 - \frac{y_0^2 + 2y_0 b\sqrt{1 - \frac{\left(\frac{z}{h} - x_0\right)^2}{a^2}} + b^2\left(1 - \frac{\left(\frac{z}{h} - x_0\right)^2}{a^2}\right)}{(y_0 + b)^2}$$

The *pdf* results by differentiating with respect to z

$$f_Z(z) = \frac{1}{(y_0 + b)^2}\left[\frac{2b^2}{a^2 h}\left(\frac{z}{h} - x_0\right) + y_0 b\left(1 - \frac{\left(\frac{z}{h} - x_0\right)^2}{a^2}\right)^{-1/2}\frac{2}{a^2 h}\left(z/h - x_0\right)\right]$$

These functions are illustrated in the bottom-left and top-right graphs of Figure 2.2. An interesting result is obtained when $y_0 = 0$, i.e., when the center of ellipsoid is on the x-axis:

$$f_Z(z) = \frac{2}{a^2}\left(\frac{z}{h} - x_0\right),$$

which is the density of the triangular distribution illustrated in the lower-right plot of Figure 2.2. For extension of these ideas to a forest stand with several trees, see Mehtätalo *et al.* (2014a). To further model the penetration of the laser beam to canopies, one could integrate the model described in Example 2.25 to these models.

Parts of the R-scripts for Figure 2.2 are shown below

```
> radius.ellipse<-function(x,b,x0=0,y0=0,h) {
+                 x<-x
+                 a<-sqrt(b^2*(1-x0)^2/(b^2-y0^2))
+                 r<-rep(h*(y0+b),length(x))
+                 r[x>x0*h&x<=h]<-h*(y0+b*sqrt(1-(x[x>x0*h&x<=h]/h-x0)^2/a^2))
+                 r[x>h]<-0
+                 r
+                 }
>
> cdf.laser<-function(x,b,x0,y0,h) {
+     a<-sqrt(b^2*(1-x0)^2/(b^2-y0^2))
+     value<-rep(0,length(x))
+     value[x>=x0*h&x<h]<-1-
+          (y0^2+2*y0*b*sqrt(1-(x[x>=x0*h&x<h]/h-x0)^2/a^2)+
+          b^2*(1-(x[x>=x0*h&x<h]/h-x0)^2/a^2))/(y0+b)^2
+     value[x>=h]<-1
+     value
+     }
>
> pdf.laser<-function(x,b,x0,y0,h) {
+     a<-sqrt(b^2*(1-x0)^2/(b^2-y0^2))
+     value<-rep(0,length(x))
+     value[x>x0*h&x<h]<--1/(y0+b)^2*
+          (y0*b*(1-(x[x>x0*h&x<h]/h-x0)^2/a^2)^(-1/2)*
+          (-2/(a^2*h)*(x[x>x0*h&x<h]/h-x0))+
+          b^2*(-2/(a^2*h)*(x[x>x0*h&x<h]/h-x0)))
+     value
+     }
> x0<-0.2; y0<--0.1; b<-0.25; h<-20
> x<-seq(0,22,0.01)
> r<-radius.ellipse(x,b,x0,y0,h)
> plot(x, r, type="l",xlab="Height, m", ylab="Crown radius, m", ylim=h*c(-0.1,0.15))
> plot(x, cdf.laser(x,b,x0,y0,h), type="l",xlab="Height, m", ylab="Cumulative density")
> plot(x, pdf.laser(x,b,x0,y0,h), type="l",xlab="Height, m", ylab="Density")
> plot(x, pdf.laser(x,b,x0,0,h), type="l",xlab="Height, m", ylab="Density")
```

Example 2.8. Let X be tree diameter with the Weibull distribution function

$$F(x|\alpha,\beta) = 1 - \exp\left[-\left(\frac{x}{\beta}\right)^{\alpha}\right],$$

with values $\alpha = 5$ and $\beta = 15$ for the shape and scale parameters, respectively. Assume that tree height above the breast height, $Y = H - 1.3$, depends deterministically on tree diameter according to the power function

$$Y = g(X) = aX^b,$$

where the parameters are $a = 8$ and $b = 0.2$. Using these values for parameters, the assumed height - diameter (H-D) curve is an increasing function of tree diameter and the height for a zero-diameter tree is equal to the breast height. To find the distribution of Y, the inverse H-D curve is

$$g^{-1}(y) = \left(\frac{y}{a}\right)^{1/b}.$$

The distribution of tree height above breast height results by writing the inverse transformation into the *cdf* of diameter. We get (see Equation (2.7) on p. 14 of ML2020)

$$\begin{aligned} F(y|\alpha,\beta,a,b) &= 1 - \exp\left\{-\left[\frac{\left(\frac{y}{a}\right)^{1/b}}{\beta}\right]^{\alpha}\right\} \qquad (2.6)\\ &= 1 - \exp\left[-\left(\frac{y}{a\beta^b}\right)^{\alpha/b}\right] \\ &= F(y|\alpha/b, a\beta^b), \qquad (2.7) \end{aligned}$$

which shows that, assuming a Weibull distribution for tree diameter and a power equation as the H-D curve, the tree height above the breast height is also distributed according to Weibull distribution, with shape and scale parameters α/b and $a\beta^b$. Note that this is not a general result, but it holds only for these specific functions. In general, there is no result that the functional form of the distributions of tree height above the breast height and diameter would be the same. In addition, treating height as transformation implies that the height is a deterministic function of tree diameter, which is not true. For more discussion, see ML2020, Section 11.4.1.

The first derivative of the inverse H-D curve is

$$\frac{d}{dy}g^{-1}(y) = \frac{1}{ab}\left(\frac{y}{a}\right)^{1/b-1} .$$

The density of height can be obtained by using the previously solved $g^{-1}(y)$ in the Weibull *pdf* and multiplying it by the derivative function above (see Equation (2.8) on p. 14 of ML2020) Another possibility is to use the result from the previous example, i.e., use α/b and $a\beta^b$ directly instead of the original parameters in the Weibull density. Both approaches lead to the same final result.

Figure 2.3 illustrates the applied transformations and distributions of diameter and height above the breast height. The following code shows how the distribution of tree heights above the breast height can be found either by using the result of the special case of Example 2.8 and the general definitions of a transformed random variable. The two alternative ways are compared by plotting the two vectors on each other; if they are on the line $y = x$, they are equal. This is often an useful way to make a quick check of computations.

```
> x<-seq(0,25,0.10)
> a<-8
> b<-0.2
> alpha<-5
> beta<-15
> Fx<-pweibull(x,alpha,beta)
> fx<-dweibull(x,alpha,beta)

> # Plot the cdf and density of diameter distribution
> plot(x,Fx,xlab="x",ylab=expression(F[X](x)),type="l",main="cdf of diameter")
> plot(x,fx,xlab="x",ylab=expression(f[X](x)),type="l",main="pdf of diameter")

> # Define the power function
> powerf<-function(x,a,b){
+         a*x^b
+         }
> # The inverse of power function
> power.inv<-function(y,a,b){
+            (y/a)^(1/b)
+            }
> # The 1st derivative of the inverse power function
> dpower.inv<-function(y,a,b){
+             1/(a*b)*(y/a)^(1/b-1)
+             }

> # Plot the HD-curve
> plot(x,powerf(x,a,b),xlab="x",ylab="g(x)",type="l",main="HD-curve")
> y<-seq(8,17,0.1)
> Fy<-pweibull(y,alpha/b,a*beta^b)
> fy<-dweibull(y,alpha/b,a*beta^b)

> # Plot the cdf and density of height distribution
> plot(y,Fy,xlab="y, m",ylab=expression(F[Y](y)),type="l",main="cdf of height")
> plot(y,fy,xlab="y, m",ylab=expression(f[Y](y)),type="l",main="pdf of height")

> # Compute the cdf in another way without computations with pen and paper
> Fy2<-pweibull(power.inv(y,a,b),alpha,beta)
> fy2<-dweibull(power.inv(y,a,b),alpha,beta)*dpower.inv(y,a,b)

> # Check graphically that the two ways gave the same result
> plot(y,Fy2,xlab="y, m",ylab=expression(F[Y](y)),type="l",main="cdf2 of height")
> plot(y,fy2,xlab="y, m",ylab=expression(f[Y](y)),type="l",main="pdf2 of height")
> plot(Fy,Fy2,main=expression("Match between "*F[X]*({g^-1}(y))*" and "*F[Y](y)))
> abline(0,1)
```

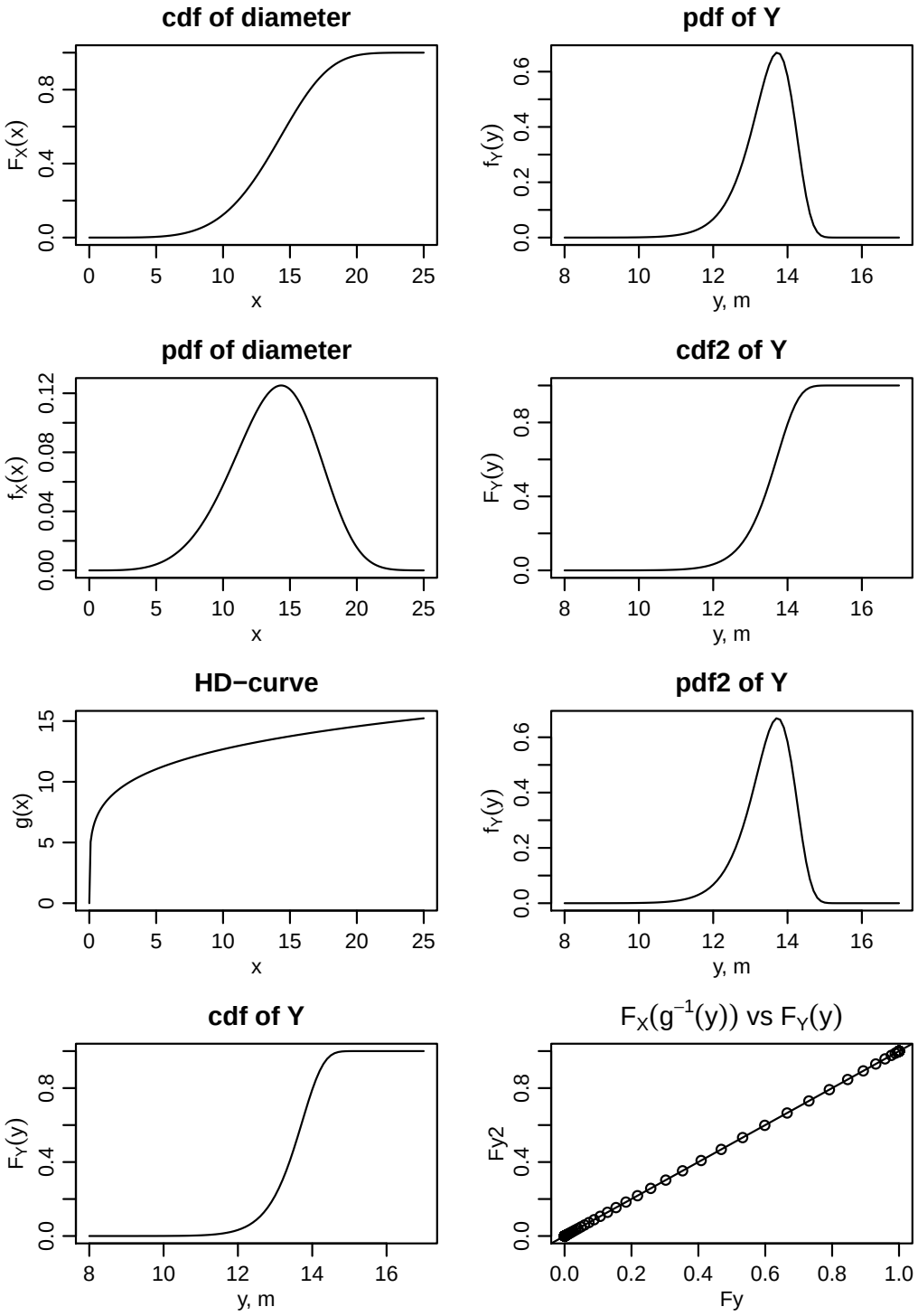


FIGURE 2.3 Illustration of the Example 2.8. The two equivalent versions of the pdf and cdf are computed diferently: the ones with texts "cdf" and "pdf" in the title are based on the $Weibull(\alpha/b, a\beta^b)$ distribution and the graphs with texts "cdf2" and "pdf2" in the title on the general results about transformed random variables. The lower-right panel illustrates a useful graphical comparison to check whether the two cdf's are identical (see the R-code).

3

Statistical Modeling, Estimation and Prediction

Example 3.9. Consider a simulated count data on the number of insects Y_i in a trap in a natural forest reserve. The true dependence of the expected number of insects in the trap on the decayed tree volume v_i in the surrounding area is

$$E(Y_i) = \mu_i = \exp(\beta_0 + \beta_1 v_i) ,$$

where the true values of parameters are $\beta_0 = -2$ and $\beta_1 = 0.05$. For illustration, we assume that the true model form is known, and data from 50 traps are used to estimate parameters β_0 and β_1. The solution is based on Maximum Likelihood under independence, by writing in Poisson likelihood (Equation 2.6 on p. 13 of ML2020) with the assumed relationship on the predictor and expected value.

The resulting ML-estimates of the parameters are $\hat{\beta}_0 = -2.25$ and $\hat{\beta}_1 = 0.053$, with variances $\text{var}(\hat{\beta}_0) = 0.117 = 0.343^2$ and $\text{var}(\hat{\beta}_1) = 0.0000187 = 0.0043^2$. The resulting estimated relationship is shown in Figure 3.1, together with the true known relationship and simulated data. For illustration, the estimation was implemented by using `stats::mle`. A more straightforward way would be to use function `glm` with `family=poisson(link="log")`.

```
> library(stats4)
> # values of the predictor variable
> V<-c(1,4,5,11,16,18,19,20,22,23,23,24,26,27,27,28,29,29,30,30,31,32,33,46,48,
+      50,51,51,53,54,55,61,62,62,62,64,65,67,68,69,76,77,81,81,83,84,86,91,92,99)
> # generate the data
> set.seed(1234)
> obs<-rpois(length(V),exp(-2+0.05*V))
> obs
 [1]  0  0  0  0  1  0  0  0  1  0  1  0  0  2  0  1  0  0  0  0  0  0  0  0  0
[26]  3  2  4  3  0  2  2  2  3  1  5  2  2 10  6  6  7  6  8  7  9 11 12 16 16
>
> # Define the negative log-likelihood
> nll<-function(b0,b1) -sum(log(dpois(obs,exp(b0+b1*V))),na.rm=TRUE)
>
> solution<-mle(minuslogl=nll,start=list(b0=0.5,b1=0),control=list(trace=TRUE))
>
> coef(solution)
         b0          b1
-2.25070964  0.05286306
> vcov(solution)
              b0            b1
b0  0.117475398 -1.442781e-03
b1 -0.001442781  1.877795e-05
```

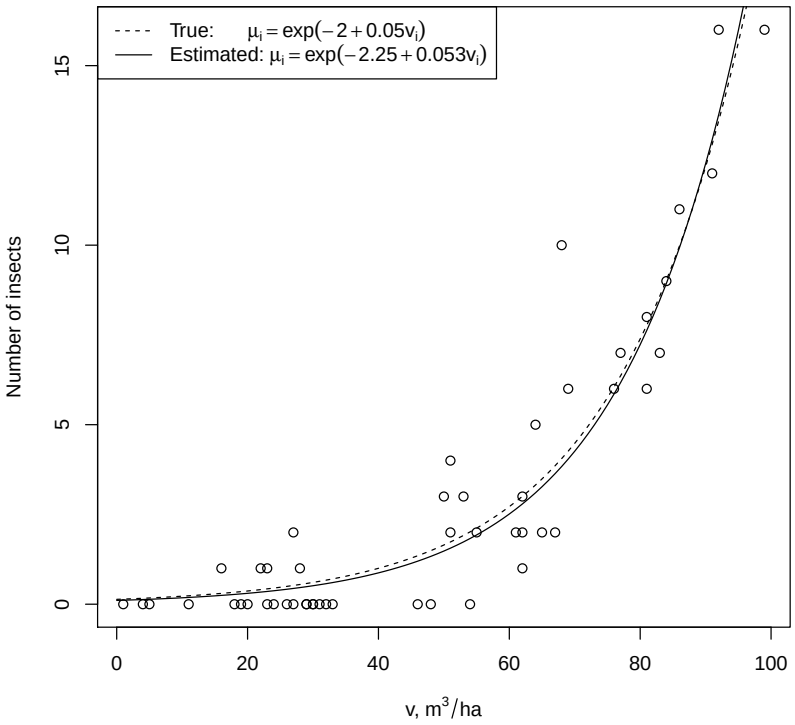


FIGURE 3.1 The observed number of insects in a trap as a function of decayed wood volume v_i in a simulated data (Example 3.9), and the true and estimated underlying relationship between μ_i and v_i.

4

Linear Model

Example 4.8. Using $\boldsymbol{X}$ and $\boldsymbol{y}$ as defined in Example 4.4. of the textbook, (the object `X` was defined before), we get

```
> y<-stumplift$Time
> n<-length(y)
> p<-dim(X)[2]
> betaOLS<-solve(t(X)%*%X)%*%t(X)%*%y
> ehat<-y-X%*%betaOLS
> sigma2OLS<-t(ehat)%*%ehat/(n-p)
> varbetaOLS<-as.numeric(sigma2OLS)*solve(t(X)%*%X)

> t(round(betaOLS,3))
                                  M2                  M3
[1,] -26.826 2.654 -0.014 116.547 -6.75 0.119 77.404 -5.02 0.095

> sqrt(sigma2OLS)
         [,1]
[1,] 24.79195

> round(sqrt(diag(varbetaOLS)),3)
                          M2                    M3
24.489  1.408  0.019 32.803  1.830  0.024 42.519  2.373  0.032

> round(cov2cor(varbetaOLS),3)
                             M2                      M3
    1.000 -0.986  0.954 -0.747  0.759 -0.751 -0.576  0.585 -0.568
   -0.986  1.000 -0.989  0.736 -0.769  0.778  0.568 -0.594  0.589
    0.954 -0.989  1.000 -0.713  0.761 -0.786 -0.550  0.587 -0.595
M2 -0.747  0.736 -0.713  1.000 -0.985  0.949  0.430 -0.437  0.424
    0.759 -0.769  0.761 -0.985  1.000 -0.988 -0.437  0.457 -0.453
   -0.751  0.778 -0.786  0.949 -0.988  1.000  0.432 -0.462  0.468
M3 -0.576  0.568 -0.550  0.430 -0.437  0.432  1.000 -0.988  0.956
    0.585 -0.594  0.587 -0.437  0.457 -0.462 -0.988  1.000 -0.989
   -0.568  0.589 -0.595  0.424 -0.453  0.468  0.956 -0.989  1.000
```

Example 4.10. Consider the multiple regression model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e}$$

with $\text{var}(\boldsymbol{e}) = \sigma^2\boldsymbol{I}$. The log-likelihood for a linear regression model

$$\boldsymbol{y} \sim N(\boldsymbol{X}\boldsymbol{\beta}, \boldsymbol{V})$$

was given in Equation (4.30) on p. 95 of ML2020. For model

$$\boldsymbol{K}'\boldsymbol{y} \sim N(\boldsymbol{0}, \boldsymbol{K}'\boldsymbol{V}\boldsymbol{K})\,,$$

the log-restricted likelihood is defined correspondingly as

$$l = -\frac{n-p}{2}\ln 2\pi - \frac{1}{2}\ln\left|\boldsymbol{K}'\boldsymbol{V}\boldsymbol{K}\right| - \frac{1}{2}\left(\boldsymbol{K}'\boldsymbol{y} - \boldsymbol{0}\right)'\left(\boldsymbol{K}'\boldsymbol{V}\boldsymbol{K}\right)^{-1}\left(\boldsymbol{K}'\boldsymbol{y} - \boldsymbol{0}\right)\,.$$

Using $\boldsymbol{V} = \sigma^2\boldsymbol{I}$ yields

$$l = -\frac{n}{2}\ln 2\pi - \frac{1}{2}\ln\left(\sigma^{2(n-p)}\left|\boldsymbol{K}'\boldsymbol{K}\right|\right) - \frac{1}{2\sigma^2}\boldsymbol{y}'\boldsymbol{K}\left(\boldsymbol{K}'\boldsymbol{K}\right)^{-1}\boldsymbol{K}'\boldsymbol{y}\,.$$

To estimate σ^2, we differentiate the log-likelihood with respect to σ^2

$$\frac{\partial l}{\partial \sigma^2} = -\frac{n-p}{2\sigma^2} + \frac{1}{2\sigma^4}\boldsymbol{y}'\boldsymbol{K}\left(\boldsymbol{K}'\boldsymbol{K}\right)^{-1}\boldsymbol{K}'\boldsymbol{y}\,.$$

Setting this to zero and solving for σ^2 gives the REML estimator

$$\widehat{\sigma}^2_{REML} = \frac{1}{n-p}\boldsymbol{y}'\boldsymbol{K}\left(\boldsymbol{K}'\boldsymbol{K}\right)^{-1}\boldsymbol{K}'\boldsymbol{y}\,. \tag{4.1}$$

The following R-code computes the REML estimator for the above-specified model using a dataset of 5 observations. We use packages `Matrix`, `MASS` and `nlme` in this example.

```
> library(Matrix)
> library(MASS)
> library(nlme)
```

First define the vector of response, $\boldsymbol{y}$, and the model matrix, $\boldsymbol{X}$.

```
> y<-c(10.3,10.3,10.2,16.4,18.8)
> x1<-c(8.3,8.6,8.8,10.5,10.7)
> x2<-c(70,65,63,72,81)
> X<-cbind(1,x1,x2)
```

Formulate 4 different $\boldsymbol{c}$-vectors of length 5. Using these vectors in Equation (4.31) on p. 96 of ML2020 provides candidate columns for matrix $\boldsymbol{K}$, which are stacked into matrix $\boldsymbol{K}$ including now 5 columns.

```
> c1<-c(1,0,0,0,0)
> c2<-c(0,1,0,0,0)
> c3<-c(0,0,1,0,0)
> c4<-c(0,0,0,1,0)
> K<-t(rbind(c1,c2,c3,c4)%*%(diag(rep(1,5))-X%*%ginv(X)))
```

Check that $\boldsymbol{K}$ satisfies the condition $\boldsymbol{K}'\boldsymbol{X} = \boldsymbol{0}$.

```
> t(K)%*%X
                              x1            x2
c1  8.881784e-16 -1.199041e-14  2.486900e-14
c2  0.000000e+00 -1.942890e-14 -3.641532e-14
c3  8.881784e-16 -1.110223e-14  3.019807e-14
c4 -6.661338e-16 -2.486900e-14 -8.171241e-14
```

This seems to be a null matrix (when rounded to the 14th decimal). Next, check the rank of $\boldsymbol{K}$. Based on the result discussed above, it should not be higher than $n - p = 5 - 3 = 2$.

```
> rankMatrix(K,tol=1e-10)
[1] 2
```

The rank is 2, meaning that if we appropriately select 2 columns from the matrix, the other rows are linear combinations of the two selected columns. Let us select two different matrices: one including first two columns (`K1`) and another including columns 3 and 4 (`K2`). They both have rank 2 (code not shown), and they both lead to the same estimate of $\widehat{\sigma}^2 = 0.135$.

```
> K1<-K[,1:2]
> K2<-K[,3:4]
> sigma1<-sqrt(t(y)%*%K1%*%solve(t(K1)%*%K1)%*%t(K1)%*%y/2)
> sigma1
          [,1]
[1,] 0.1347088
> sigma2<-sqrt(t(y)%*%K2%*%solve(t(K2)%*%K2)%*%t(K2)%*%y/2)
> sigma2
          [,1]
[1,] 0.1347088
```

The estimate equals to the one produced by function `gls` of package `nlme`.

```
> gls(y~x1+x2,method="REML")
 < . . . >
  Log-restricted-likelihood: -2.634508
 < . . . >
Degrees of freedom: 5 total; 2 residual
Residual standard error: 0.1347088
```

However, our two alternative $\boldsymbol{K}$-matrices and the fit using `gls` all provide different value for the restricted log-likelihood at the solution.

```
> logLik<-function(K,n=5,p=3){
+         -n/2*log(2*pi)-1/2*log(sigma2^(2*(n-p))*det(t(K)%*%K))-
+         1/(2*sigma2^2)*t(y)%*%K%*%solve(t(K)%*%K)%*%t(K)%*%y
+         }
> logLik(K1)
           [,1]
[1,] -0.3821908
> logLik(K2)
          [,1]
[1,] -0.507354
```

5

Linear Mixed-Effects Models

Example 5.9. Consider the model with random intercept with the following variance function

$$\text{var}(\epsilon_{ij}) = \sigma^2 |\widetilde{y}_{ij}|^{2\delta} , \tag{5.1}$$

where $\widetilde{y}_{ij}$ is the *group-level* prediction from the model. This structure is commonly applicable for processes where the predictions are nonzero.

With that model, the variance-covariance structure of $\boldsymbol{y}_i$ is

$$\begin{aligned}
\text{var}\,(\boldsymbol{y}_i) &= \boldsymbol{Z}_i \boldsymbol{D}_* \boldsymbol{Z}_i' + \boldsymbol{R}_i \\
&= \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \sigma_b^2 \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix} + \sigma^2 \begin{bmatrix} |\widetilde{y}_{i1}|^{2\delta} & 0 & \dots & 0 \\ 0 & |\widetilde{y}_{i2}|^{2\delta} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & |\widetilde{y}_{in_i}|^{2\delta} \end{bmatrix} \\
&= \begin{bmatrix} \sigma_b^2 + \sigma^2 |\widetilde{y}_{i1}|^{2\delta} & \sigma_b^2 & \dots & \sigma_b^2 \\ \sigma_b^2 & \sigma_b^2 + \sigma^2 |\widetilde{y}_{i2}|^{2\delta} & \dots & \sigma_b^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_b^2 & \sigma_b^2 & \dots & \sigma_b^2 + \sigma^2 |\widetilde{y}_{in_i}|^{2\delta} \end{bmatrix} .
\end{aligned}$$

Example 5.12. The model with random intercept and slope:

$$y_{ij} = \beta_1 + \beta_2 x_{ij} + b_i^{(1)} + b_i^{(2)} x_{ij} + \epsilon_{ij} ,$$

for the data of Table 5.2 (same as Table 5.2 on p. 149 of ML2020) is specified by defining

$$\boldsymbol{Z} = \begin{pmatrix} 1 & 10 & 0 & 0 & 0 & 0 \\ 1 & 13 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 11 & 0 & 0 \\ 0 & 0 & 1 & 14 & 0 & 0 \\ 0 & 0 & 1 & 15 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 12 \end{pmatrix} , \boldsymbol{b} = \begin{pmatrix} b_1^{(1)} \\ b_1^{(2)} \\ b_2^{(1)} \\ b_2^{(2)} \\ b_3^{(1)} \\ b_3^{(2)} \end{pmatrix} ,$$

$$\text{var}(\boldsymbol{b}) = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & 0 & 0 & 0 & 0 \\ \sigma_{12} & \sigma_2^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_1^2 & \sigma_{12} & 0 & 0 \\ 0 & 0 & \sigma_{12} & \sigma_2^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_1^2 & \sigma_{12} \\ 0 & 0 & 0 & 0 & \sigma_{12} & \sigma_2^2 \end{pmatrix}$$

Group	y	x	Group	y	x	Group	y	x
1	2	10	2	2	11	2	5	15
1	4	13	2	4	14	3	4	12

TABLE 5.2 A small grouped data.

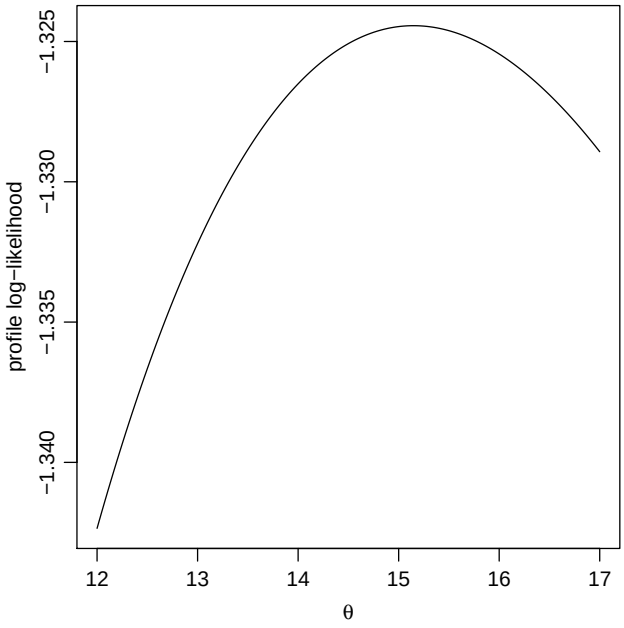


FIGURE 5.1 The profile log likelihood as a function of $\theta = \sigma_b^2/\sigma^2$ in Example 5.15.

where $\text{var}(b_i^{(1)}) = \sigma_1^2$, $\text{var}(b_i^{(2)}) = \sigma_2^2$ and $\text{cov}(b_i^{(1)}, b_i^{(2)}) = \sigma_{12}$. The other matrices are as specified in Example 5.11 of ML2020.

Example 5.15. Consider the model

$$y_{ij} = \beta_1 + \beta_2 x_{ij} + b_i + \epsilon_{ij}$$

in the dataset of Table 5.2 (p. 13).

```
> dat<-data.frame(group=c(1,1,2,2,2,3),
+                     y=c(2,4,2,4,5,4),
+                     x=c(10,13,11,14,15,12))
```

To profile out $\boldsymbol{\beta}$ and σ^2 from the log-likelihood, we implement the estimators of $\boldsymbol{\beta}$ and σ^2 as R-functions. These functions are called from within the function that defines the profile log-likelihood. The only remaining parameter in matrix $\boldsymbol{V}$ is the ratio of within-group variance and error variance, therefore $\boldsymbol{\theta}$ is scalar $\theta = \sigma_b^2/\sigma^2$ (see Example 5.1 on p. 149 of ML2020).

```
> dat<-data.frame(group=c(1,1,2,2,2,3),
+                     y=c(2,4,2,4,5,4),
+                     x=c(10,13,11,14,15,12))
>
> betagls<-function(X,V,y){
+          solve(t(X)%*%solve(V)%*%X)%*%t(X)%*%solve(V)%*%y
+ }
>
> # A function to profile out sigma^2 from the likelihood in a model with
> # single random effect and constant variance
> sigma2<-function(y,X,beta,V){
+         t(y-X%*%beta)%*%solve(V)%*%(y-X%*%beta)/length(y)
+ }
>
> loglik<-function(theta,y,X,group){
+         N<-length(y)
+         M<-length(unique(group))
+         D<-diag(rep(theta,M))
+         Z<-matrix(0,ncol=M,nrow=N)
```

```
+         for (i in 1:M){
+              Z[group==i,i]<-1
+         }
+         R<-diag(rep(1,length(group)))
+         V<-Z%*%D%*%t(Z)+R
+         betavalue<-betagls(X,V,y)
+         sigma2value<-sigma2(y,X,betavalue,V)
+         ll<--N/2*log(2*pi)-N/2*log(sigma2value)-
+              1/2*log(det(V))-
+              1/(2*sigma2value)*t(y-X%*%betavalue)%*%solve(V)%*%(y-X%*%betavalue)
+         list(betahat=betavalue,sigma=sqrt(sigma2value),logLik=ll)
+ }
>
> theta<-seq(12,17,0.01)
> y<-dat$y
> X<-cbind(1,dat$x)
> group<-dat$group
> l<-sapply(theta,function(par) loglik(par,y,X,group)$logLik)
>
> #sratioMLE<-sb2ps[l==max(l)]
> #sratioMLE
>
> thetaMLE<-theta[l==max(l)]
> thetaMLE
[1] 15.15
```

A proper solution would be to differentiate the profile log-likelihood with respect to θ, set it to zero and solve it. However, for illustration purposes, Figure 5.1 shows the likelihood as a function of θ. The log-likelihood is maximized at $\theta \approx 15.15$. We use this estimate to compute the ML estimates of $\boldsymbol{\beta}$ and σ, and σ_b.

```
> sol<-loglik(thetaMLE,y,X,group)
> sol
$betahat
           [,1]
[1,] -5.1064731
[2,]  0.7047766

$sigma
          [,1]
[1,] 0.1304378

$logLik
          [,1]
[1,] -1.324437

> sb<-sqrt(thetaMLE*sol$sigma^2)
> sb
          [,1]
[1,] 0.5077031
```

The small differences in the estimates compared to Example 5.14 on p. 155 of Mehtätalo and Lappi (2020) result from rounding the $\theta = 15.15$ to the accuracy of two decimals only. Function `lme` used the value $(0.5076805/0.1304432)^2 = 15.1474$.

6

More About Linear Mixed-Effects Models

Example 6.2. Using the dataset shown in Table 6.1 (same as Table 6.1 on p. 184 of ML2020) and assuming random intercept and slope at both levels of grouping, the model for group 1 can be written as

$$\boldsymbol{y}_1 = \boldsymbol{X}_1\boldsymbol{\beta} + \boldsymbol{Z}_1\boldsymbol{b}_1 + \boldsymbol{\epsilon}_1$$

by defining

$$\boldsymbol{Z}_1 = \begin{bmatrix} 1 & 13 & 1 & 13 & 0 & 0 \\ 1 & 26 & 1 & 26 & 0 & 0 \\ 1 & 18 & 1 & 18 & 0 & 0 \\ 1 & 22 & 0 & 0 & 1 & 22 \\ 1 & 12 & 0 & 0 & 1 & 12 \end{bmatrix}, \boldsymbol{b}_1 = \begin{bmatrix} a_1^{(1)} \\ a_2^{(2)} \\ c_{11}^{(1)} \\ c_{11}^{(2)} \\ c_{12}^{(1)} \\ c_{12}^{(2)} \end{bmatrix},$$

$$\boldsymbol{X}_1 = \begin{bmatrix} 1 & 13 \\ 1 & 26 \\ 1 & 18 \\ 1 & 22 \\ 1 & 12 \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}.$$

The model for the entire data can be written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b} + \boldsymbol{\epsilon}$$

by using the group-specific model matrices, from Example 6.1 (p. 184 of ML2020) in

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_1 & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Z}_2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{Z}_3 \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_3 \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \boldsymbol{X}_3 \end{bmatrix}.$$

Example 6.6. Mehtätalo *et al.* (2014b) used linear mixed-effects models with crossed tree and year level random effects to estimate the pure effects of silvicultural thinning on tree-specific growth time series. The data are the same as in Examples 6.3 and 6.5 of ML2020, except that it also includes observations of ring basal area before the thinning, which happened after the growth period of year 1986. For the unthinned plot, the analysis uses for all years. For thinned plots, only observations for the years before thinning are used. Such a data allows prediction

Group (i)	Subgroup (j)	x	y	Group (i)	Subgroup (j)	x	y
1	1	13	19.4	2	2	28	36.3
1	1	26	32.1	2	3	18	25.3
1	1	18	21.4	3	1	16	31.0
1	2	22	31.8	3	1	15	30.3
1	2	12	20.1	3	2	19	29.9
2	1	27	33.9	3	2	24	34.4
2	2	12	22.6	3	3	25	35.8

TABLE 6.1 A small grouped data set with two nested levels of grouping.

of tree effects for all trees of the data and year effects for all calendar years of the data. Those are estimated using a mixed-effects model with crossed year and tree level random effects. The model is used to predict the hypothetical *RBA* for the trees of thinned plots without thinning also for years after thinning, from which the tree and year effects in undisturbed growth has been eliminated. The effect of thinning on RBA is then estimated as the difference between the observed RBA and the prediction. The model was fitted for logarithmic basal area growths, and the age trend was modeled using a restricted (natural) cubic spline with three knots (Harrell 2001), see Example 4.14 on p. 108 and Section 10.4.1 of ML2020.

Start by reading the data and computing the transformation for the spline.

```
> data(patti)
> # use only data of year 1982 ->
> patti<-patti[patti$Year>1982,]
> # Use 3 knots to model the age trend using a regression spline.
> # Compute the required spline component.
> te<-quantile(patti$CA,probs=c(0.1,0.5,0.9))
> patti$e1<-natspline(X=patti$CA,t=te,j=1)
```

Next, we formulate the model fitting dataset `patti2`, which includes only (1) observations for thinned plots before thinning and (2) observations of control plots for all years and fit the model with crossed tree and year effects.

```
> patti2<-patti[patti$SDClass==1|patti$Year<1987,]
> extmod<-lmer(log(RBA)~CA+e1+(1|Tree)+(1|Year),data=patti2)
```

The model is used to make the logarithmic predictions for dataset `patti`, which includes also observations after the year of thinning for the thinned plots. Back-transforming the logarithmic growths to original scale introduces transformation bias, which is adjusted using the two-point bias correction (see Section 10.2.3 of ML2020). The estimation errors of fixed effects, the prediction errors of random effects and their covariance are ignored in the bias correction; implementation of a correction including also these components is left as an exercise (See Section 5.4.2 of ML2020 for details about computing the required variances and covariances). The estimated thinning effects, obtained by subtracting the predicted RBA's from the observed RBA's, are shown in Figure 6.1 (right). These effects will later be modeled using nonlinear mixed-effects model in Chapter 7.

```
> patti$logpred<-predict(extmod,newdata=patti,
+                        re.form=~(1|Tree)+(1|Year))

> # Back-transformation with the two-point bias correction
> patti$trend<-(exp(patti$logpred+summary(extmod)$sigma)+
+               exp(patti$logpred-summary(extmod)$sigma))/2

> # Compute the thinning effects to the dataset.
> patti$ThEf<-patti$RBA-patti$trend
```

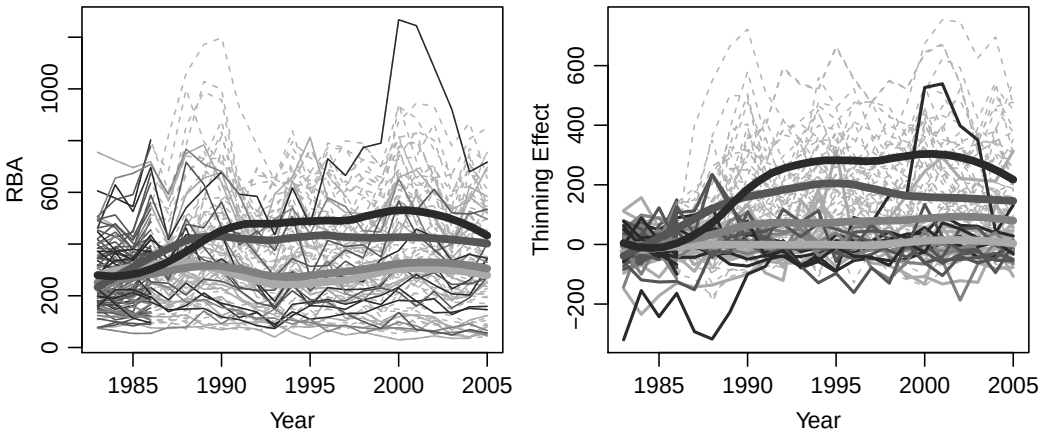


FIGURE 6.1 The ring basal areas for all trees over all years since 1982 (left) and the extracted thinning effects (right). The thin solid lines show the observations from thinned plots before thinning and control plots for the entire period. The thin dashed lines show observations of thinned plots after the thinning. The thick lines show the average trend, based on a lowess smoother, separately for each thinning treatment. The grayscale indicates the thinning treatment (black=heavy ... lightgray=control).

7

Nonlinear (Mixed-Effects) Models

Example 7.5. To demonstrate the Gauss-Newton algorithm (see p. 217 of ML2020), this example fits the model of Example 7.4 (p. 219 of ML2020) manually without using the function `nls`.

The partial derivatives of the response function with respect to ϕ_1 and ϕ_2 are

$$\begin{aligned}\frac{\partial f(T_i,\phi)}{\partial \phi_1} &= f(T_i,\phi)\\ \frac{\partial f(T_i,\phi)}{\partial \phi_2} &= \frac{-8T_i \exp(4-8T_i\exp(-\phi_2)+\phi_1-\phi_2)}{[1+\exp(4-8T_i\exp(-\phi_2))]^2}.\end{aligned}$$

The Jacobian $\boldsymbol{X}_{23\times 2}$ at each iteration is obtained by evaluating these derivatives for the observed values ($n = 23$) at the current estimate of $\boldsymbol{\phi}$. The following script implements the modified Gauss-Newton estimation of the parameters. The iteration is continued until the sum of squared change in parameter estimates gets smaller than 10^{-10}.

```
> data(thinning)
> # Reparameterized logistic function
> mylogis<-function(x,phi1,phi2){
+          ephi1<-exp(phi1)
+          ephi2<-exp(phi2)
+          ephi1/(1+exp(4-8*x/ephi2))
+          }

> # The Jacobian
> mylogisJ<-function(x,phi1,phi2){
+           temp<-8*x*exp(-phi2)
+           d1<-exp(phi1)/(1+exp(4-temp))
+           d2<--8*x*exp(4-temp+phi1-phi2)*(1+exp(4-temp))^{-2}
+           cbind(d1,d2)
+           }

> # Modified Gauss-Newton algorithm
> y<-thinning$ThEf
> x<-thinning$Year-1986
> phi<-0
> phinew<-c(log(320),log(5))
> i<-0
> while (sum((phi-phinew)^2)>1e-10){
+        phi<-phinew
+        i<-i+1
+        u<-y-mylogis(x,phi[1],phi[2])
+        RSS<-RSSnew<-sum(u^2)
+        cat("Iteration: ",i,exp(phi),RSS,"\n")
+        X<-mylogisJ(x,phi[1],phi[2])
+        psi<-coef(lm(u~X-1))
+        lambda<-2
+        while (RSSnew>=RSS){
+               lambda<-lambda/2
+               phinew<-phi+lambda*psi
+               unew<-y-mylogis(x,phinew[1],phinew[2])
+               RSSnew<-sum(unew^2)
+               }
+        }
Iteration:  1 320 5 70072.44
Iteration:  2 331.1573 4.789681 67144.78
Iteration:  3 330.4498 4.703177 67062.88
Iteration:  4 330.2351 4.666695 67048.41
Iteration:  5 330.1442 4.651719 67045.97
```

```
Iteration:  6 330.1068 4.645636 67045.57
Iteration:  7 330.0916 4.643176 67045.5
Iteration:  8 330.0855 4.642183 67045.49
Iteration:  9 330.083 4.641783 67045.49
Iteration:  10 330.082 4.641621 67045.49
Iteration:  11 330.0816 4.641556 67045.49

> sigma2<-RSSnew/(23-2)
> varbeta<-sigma2*solve(t(X)%*%X)
> phinew              # Parameter estimates
     Xd1      Xd2
5.799339 1.535044
> sqrt(sigma2)        # Residual standard error
[1] 56.50347
> sqrt(diag(varbeta)) # Standard errors of parameter estimates
        d1         d2
0.04371602 0.13834707
```

Example 7.7. Consider the model

$$y_{ij} = \phi_{ij}^{x_{ij}} + \epsilon_{ij}, \tag{7.1}$$

where $\phi_{ij} = \beta + b_i$, $b_i \sim N(0, \sigma_b^2)$, $\epsilon_{ij} \sim N(0, \sigma^2)$, the random effects b_i are mutually independent, residual errors ϵ_{ij} are mutually independent, and random effects are independent of residual errors. The use of this function might be justified for a process where y approaches asymptotically zero; in that case we should have $\phi_{ij} > 0$. To minimize the probability for a negative values of ϕ_{ij}, β should be much larger than $\text{sd}(b_{ij})$. Negative values could be completely prohibited through model formulation, for example, by using a gamma distribution for the random effects (with the minimum of $-\beta$), or reparameterizing the model as $y_{ij} = \exp(\theta_{ij} x_{ij}) + \epsilon_{ij}$ where $\theta_{ij} = \beta + b_i$ and b_i is normally distributed.

The R-script below generates a small balanced data of 25 observations (5 groups with 5 observations per group) from this model using parameter values $\beta = 0.5$, $\sigma_b^2 = 0.15^2$, $\sigma^2 = 0.05^2$ for values $\boldsymbol{x}_i = (0, 2, 4, 6, 8)'$ of the predictor in all groups. The simulated true values of the random effects are $\boldsymbol{b} = (0.0878, 0.1064, -0.0164, -0.0680, 0.0909)'$. Figure 7.4 shows the data.

```
> set.seed(12345)
> beta<-0.5
> b<-rnorm(5,0,0.15)
> phi<-beta+rep(b,each=5)
> x<-rep(seq(0,8,2),5)
> group<-rep(1:5,each=5)
> y<-phi^x+rnorm(25,0,0.05)
> nlmeData<-data.frame(x,y,group)
```

We can write the model in form

$$y_{ij} = f(x_{ij}, \boldsymbol{\phi}_{ij}) + \epsilon_{ij}$$

where

$$\boldsymbol{\phi}_{ij} = \boldsymbol{A}_{ij}\boldsymbol{\beta} + \boldsymbol{B}_{ij}\boldsymbol{b}_i$$

by defining

$$\boldsymbol{A}_{ij} = 1,\ \boldsymbol{\beta} = \beta,\ \boldsymbol{B}_{ij} = 1,\ \boldsymbol{b}_i = b_i.$$

The model has only one primary parameter and only one fixed second-order parameter, therefore all these matrices and vectors became scalars.

For group 1, the model can be written in the form of Equation (7.19) of ML2020 by defining

$$\boldsymbol{y}_1 = \begin{pmatrix} 0.909 \\ 0.377 \\ 0.106 \\ 0.027 \\ -0.032 \end{pmatrix},\ \boldsymbol{\phi}_1 = \begin{pmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{14} \\ \phi_{15} \end{pmatrix},\ \boldsymbol{\epsilon}_1 = \begin{pmatrix} \epsilon_{11} \\ \epsilon_{12} \\ \epsilon_{13} \\ \epsilon_{14} \\ \epsilon_{15} \end{pmatrix},$$

$$\boldsymbol{f}_1(\boldsymbol{x},\boldsymbol{\phi}) = \begin{pmatrix} \phi_{11}^0 \\ \phi_{12}^2 \\ \phi_{13}^4 \\ \phi_{14}^6 \\ \phi_{15}^8 \end{pmatrix}, \boldsymbol{x}_1 = \begin{pmatrix} 0 \\ 2 \\ 4 \\ 6 \\ 8 \end{pmatrix}, \boldsymbol{A}_1 = \boldsymbol{B}_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

For the entire data, the model can be written in form Equation (7.20) of ML2020 by defining $\boldsymbol{\beta} = \beta$,

$$\boldsymbol{y} = \begin{pmatrix} 0.909 \\ \vdots \\ -0.032 \\ 0.994 \\ \vdots \\ -0.019 \\ \vdots \\ 1.090 \\ \vdots \\ 0.007 \end{pmatrix}, \boldsymbol{\phi} = \begin{pmatrix} \phi_{11} \\ \vdots \\ \phi_{15} \\ \phi_{21} \\ \vdots \\ \phi_{25} \\ \vdots \\ \phi_{51} \\ \vdots \\ \phi_{55} \end{pmatrix}, \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_{11} \\ \vdots \\ \epsilon_{15} \\ \epsilon_{21} \\ \vdots \\ \epsilon_{25} \\ \vdots \\ \epsilon_{51} \\ \vdots \\ \epsilon_{55} \end{pmatrix}, \boldsymbol{f}(\boldsymbol{\phi},\boldsymbol{x}) = \begin{pmatrix} \phi_{11}^0 \\ \vdots \\ \phi_{15}^8 \\ \phi_{21}^0 \\ \vdots \\ \phi_{25}^8 \\ \vdots \\ \phi_{51}^0 \\ \vdots \\ \phi_{55}^8 \end{pmatrix},$$

$$\boldsymbol{x} = \begin{pmatrix} 0 \\ \vdots \\ 8 \\ 0 \\ \vdots \\ 8 \\ \vdots \\ 0 \\ \vdots \\ 8 \end{pmatrix}, \boldsymbol{A} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \boldsymbol{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{pmatrix}.$$

Now $\text{var}(\boldsymbol{\epsilon}) = \sigma^2 \boldsymbol{I}_{25}$ and $\text{var}(\boldsymbol{b}) = \boldsymbol{I}_5 \otimes \boldsymbol{D}_*$, where $\boldsymbol{D}_* = \sigma_b^2$.

Note that if model (7.1) were used in analysis of real data, it would be justified to parameterize it in terms of $\theta_{ij} = \ln(\phi_{ij})$ to ensure that ϕ_{ij} is always positive. One could then assume that ϕ_{ij} is a linear function of predictors and random effects. Of course, such model would not be the correct model for the data of Figure 7.4.

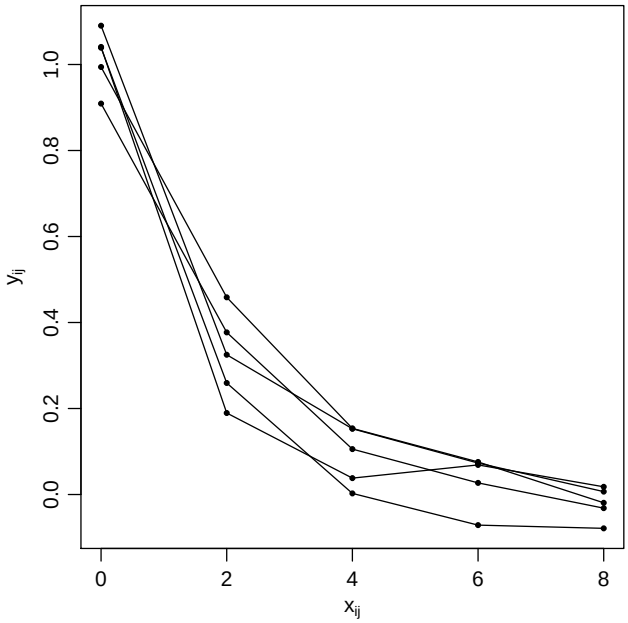
Example 7.9. Consider the simulated data from Example 7.7, shown in Figure 7.4. To compute matrices $\widehat{\boldsymbol{X}}_i$ and $\widehat{\boldsymbol{Z}}_i$, we need to differentiate the model function with respect to β and b. These become

$$\frac{\partial(\beta + b_i)^{x_{ij}}}{\partial \beta} = x_{ij}(\beta + b_i)^{x_{ij}-1}$$

$$\frac{\partial(\beta + b_i)^{x_{ij}}}{\partial b_i} = x_{ij}(\beta + b_i)^{x_{ij}-1}$$

In addition, we need starting values of the estimates for the first PNLS step. It is easy to notice that $f(1) = \phi^1 = \phi$. Therefore, ϕ can be interpreted as the mean of y_i at $x_{ij} = 1$. Our data does not include observed values of y_{ij} for $x_{ij} = 1$, but we approximate them by the means of y_{ij} for values 0 and 2 of x_{ij} for each group. Thereafter we compute the mean and variance of these and use them as the initial guesses of β and σ_b^2. To find the initial guess for σ^2, we fit a nonlinear fixed-effects model for each group separately and use the mean of the estimated residual variances of these models. For more general strategies for finding the initial guesses, see Lindstrom and Bates (1990). The script below implements the algorithm.

```
> # Starting values
```



x_{ij}	y_{ij}				
	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$
0	0.909	0.994	1.041	1.039	1.090
2	0.377	0.459	0.190	0.259	0.325
4	0.106	0.154	0.038	0.003	0.153
6	0.027	0.076	0.069	-0.071	0.073
8	-0.032	-0.019	0.018	-0.079	0.007

FIGURE 7.4 A simulated small data set that follows model (7.1) with $\beta = 0.5$, $\sigma_b^2 = 0.15^2$, $\sigma^2 = 0.05^2$, and $\boldsymbol{b}' = (0.0878, 0.1064, -0.0164, -0.0680, 0.0909)$.

```
> b0<-rep(0,5)
> groupmeans<-with(nlmeData[nlmeData$x<=2,],tapply(y,group,mean))
> beta0<-mean(groupmeans)
> D<-var(groupmeans)
> s2<-rep(NA,5)
> for (i in 1:5) {
+      s2[i]<-summary(nls(y~phi^x,start=list(phi=beta0),
+                         data=nlmeData[nlmeData$group==i,]))$sigma^2
+ }
> sigma2<-mean(s2)
> L<-chol(D/sigma2)
> Delta0<-diag(rep(solve(L),5))

> # Matrices A and B and the pseudo-data for the PNLS-step.
> fun<-function(x,phi) phi^x
> A<-matrix(1,ncol=1,nrow=25)
> Bi<-matrix(1,ncol=1,nrow=5)
> library(magic)
> B<-adiag(Bi,Bi,Bi,Bi,Bi)
> y<-nlmeData$y
> x<-nlmeData$x
> group<-nlmeData$group
> ypseudo<-c(y,rep(0,5))

> betaold<-0 # initialize to control the stopping criteria
> i<-0

> while ((beta0-betaold)^2>1e-16) {
+        i<-i+1
```

```
+        fn<-function(x,beta,b1,b2,b3,b4,b5) {
+            b<-c(b1,b2,b3,b4,b5)
+            phi<-A%*%beta+B%*%b
+            c(fun(x,phi),Delta0%*%b)
+            }
+
+        pnlsmod<-nls(ypseudo~fn(x,beta,b1,b2,b3,b4,b5),
+                     start=list(beta=beta0,b1=b0[1],b2=b0[2],b3=b0[3],b4=b0[4],b5=b0[5]))
+
+        betaold<-beta0
+        beta0<-coef(pnlsmod)[1]
+        b0<-coef(pnlsmod)[-1]
+        cat("PNLS-step",i,": beta:",beta0,"\n")
+        cat("PNLS-step",i,": b:",b0,"\n")
+        phi0<-A%*%beta0+B%*%b0
+        X1<-matrix(x*phi0^(x-1),ncol=1)
+        z<-x*phi0^(x-1)
+        Z1<-matrix(0,ncol=5,nrow=25)
+        for (j in 1:5) Z1[group==j,j]<-z[group==j]
+
+        w1<-y-fun(x,phi0)+X1%*%beta0+Z1%*%b0
+        lmemod<-lme(w1~X1-1,random=~z-1|group,method="ML")
+        L<-chol(getVarCov(lmemod)/lmemod$sigma^2)
+        Delta0<-diag(rep(solve(L),5))
+        beta0<-fixef(lmemod)
+        cat("LME-step",i,": sigmab^2 and sigma^2",sqrt(getVarCov(lmemod)), lmemod$sigma,"\n")
+ }

PNLS-step 1 : beta: 0.5611208
PNLS-step 1 : b: 0.02092517 0.06586918 -0.05736649 -0.05107781 0.02164995
LME-step 1 : sigmab^2 and sigma^2 0.06791626 0.05047221
 < part of the output was removed >
PNLS-step 5 : beta: 0.5581621
PNLS-step 5 : b: 0.02764707 0.07910986 -0.07360078 -0.06215649 0.02900033
LME-step 5 : sigmab^2 and sigma^2 0.06711578 0.05063212

> round(summary(pnlsmod)$parameters[1,],5)
  Estimate Std. Error    t value   Pr(>|t|)
   0.55816    0.03511   15.89650    0.00000
> VarCorr(lmemod)
group = pdLogChol(z - 1)
         Variance    StdDev
z        0.004504527 0.06711578
Residual 0.002563612 0.05063212
> coef(pnlsmod)[-1]
      b1.b1       b2.b2       b3.b3       b4.b4       b5.b5
 0.02764707  0.07910986 -0.07360078 -0.06215649  0.02900033
```

The algorithm converged after 5 iterations, with following parameter estimates: $\widehat{\beta} = 0.558$, $\widehat{\sigma}_b^2 = 0.0671^2$, and $\widehat{\sigma}^2 = 0.0506^2$. The predicted values of random effects were $\widetilde{\boldsymbol{b}}' = (0.0276, 0.0791, -0.0736, -0.0622, 0.0290)$.

9

Multivariate (Mixed-Effects) Models

Example 9.8. Consider the model system that was estimated in Example 9.6 on p. 303 of ML2020. Assume that three trees have been measured for DBH, HDB, and HCB, with the following measurements and values of the predictors.

```
> obs
  plot tree   DBH  HDB  HCB  hmax   h20   h30   h70   h80 a_hmean  a_veg a_h30 a_h70
2    1   30 24.60 0.75 12.3 20.87 13.74 14.94 18.42 18.84   14.44 0.8952 14.51 17.53
5    1   43 28.25 0.40 13.0 20.00 13.33 14.77 17.77 18.47   14.75 0.8533 14.89 18.52
7    1   18 25.40 2.30 14.1 21.50 13.43 15.58 18.63 19.50   14.51 0.9268 14.25 17.84
```

The aim is to predict the random effects of all five models of the system by using these data. We extract $\boldsymbol{\beta}$, $\boldsymbol{D}$, and $\boldsymbol{\Sigma}$ from the fitted model (Equation (9.11), p 301 of ML2020). Thereafter we construct the required matrices and vectors and predict the random effects. The constructed matrices are printed to show their structure. The predicted values of random effects and their prediction errors are then computed to objects `b` and `varbb`.

```
> D<-getVarCov(syssur)
> corepsilon<-corMatrix(syssur$modelStruct[2]$corStruct)[1][[1]]
> sdepsilon<-syssur$sigma*c(1,exp(coef(syssur$modelStruct[3]$varStruct)))
> Sigma<-diag(sdepsilon)%*%corepsilon%*%diag(sdepsilon)

> library(magic)
> ntrees<-3
> Z1234<-cbind(1,obs[,"hmax"])
> Z5<-cbind(1,obs[,"h20"])
> Zo<-adiag(Z1234,Z1234,Z5)

> C<-D[,-c(3,4,5,6)]
> Do<-D[-c(3,4,5,6),-c(3,4,5,6)]
> Ro<-Sigma[-c(2,3),-c(2,3)]%x%diag(ntrees)
> Xo<-adiag(cbind(1,obs$hmax,obs$a_hmean,sqrt(obs$h30)),
+           cbind(1,obs$hmax,obs$a_h70),
+           cbind(1,obs$h20,obs$a_hmean,log(obs$a_veg)))

> beta<-fixef(syssur)
> betao<-beta[-seq(5,11)]
> muo<-Xo%*%betao
> yo<-c(obs$DBH,sqrt(obs$HDB),obs$HCB)

> round(D,3)
Random effects variance covariance matrix
          conD  hmaxD   conH  hmaxH   conV  hmaxV conHDB hmaxHDB conHCB h20HCB
conD    20.978 -1.199 -1.794  0.098 -0.306  0.029  4.375  -0.222 -2.078 -0.086
hmaxD   -1.199  0.079  0.096 -0.005  0.026 -0.001 -0.263   0.015  0.122  0.003
conH    -1.794  0.096  0.520 -0.032  0.023  0.001 -0.413   0.014  0.098 -0.002
hmaxH    0.098 -0.005 -0.032  0.002  0.000  0.000  0.021  -0.001 -0.009  0.001
conV    -0.306  0.026  0.023  0.000  0.161 -0.009 -0.177   0.011 -0.030 -0.002
hmaxV    0.029 -0.001  0.001  0.000 -0.009  0.001  0.012  -0.001 -0.001  0.000
conHDB   4.375 -0.263 -0.413  0.021 -0.177  0.012  1.057  -0.060 -0.424 -0.012
hmaxHDB -0.222  0.015  0.014 -0.001  0.011 -0.001 -0.060   0.004  0.029  0.000
conHCB  -2.078  0.122  0.098 -0.009 -0.030 -0.001 -0.424   0.029  0.367 -0.001
h20HCB  -0.086  0.003 -0.002  0.001 -0.002  0.000 -0.012   0.000 -0.001  0.003
  Standard Deviations: 4.58 0.2811 0.7211 0.04472 0.4013 0.03162 1.028 0.06325 0.6058 0.05477
> Ro
         [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]    [,9]
 [1,]  7.7567  0.0000  0.0000 -0.1420  0.0000  0.0000 -0.8723  0.0000  0.0000
 [2,]  0.0000  7.7567  0.0000  0.0000 -0.1420  0.0000  0.0000 -0.8723  0.0000
 [3,]  0.0000  0.0000  7.7567  0.0000  0.0000 -0.1420  0.0000  0.0000 -0.8723
 [4,] -0.1420  0.0000  0.0000  0.1196  0.0000  0.0000  0.0279  0.0000  0.0000
 [5,]  0.0000 -0.1420  0.0000  0.0000  0.1196  0.0000  0.0000  0.0279  0.0000
```

```
 [6,]  0.0000  0.0000 -0.1420  0.0000  0.0000  0.1196  0.0000  0.0000  0.0279
 [7,] -0.8723  0.0000  0.0000  0.0279  0.0000  0.0000  1.1346  0.0000  0.0000
 [8,]  0.0000 -0.8723  0.0000  0.0000  0.0279  0.0000  0.0000  1.1346  0.0000
 [9,]  0.0000  0.0000 -0.8723  0.0000  0.0000  0.0279  0.0000  0.0000  1.1346
> round(C,3)
          conD  hmaxD conHDB hmaxHDB conHCB h2OHCB
conD    20.978 -1.199  4.375  -0.222 -2.078 -0.086
hmaxD   -1.199  0.079 -0.263   0.015  0.122  0.003
conH    -1.794  0.096 -0.413   0.014  0.098 -0.002
hmaxH    0.098 -0.005  0.021  -0.001 -0.009  0.001
conV    -0.306  0.026 -0.177   0.011 -0.030 -0.002
hmaxV    0.029 -0.001  0.012  -0.001 -0.001  0.000
conHDB   4.375 -0.263  1.057  -0.060 -0.424 -0.012
hmaxHDB -0.222  0.015 -0.060   0.004  0.029  0.000
conHCB  -2.078  0.122 -0.424   0.029  0.367 -0.001
h2OHCB  -0.086  0.003 -0.012   0.000 -0.001  0.003
> Zo
      [,1]  [,2] [,3]  [,4] [,5]  [,6]
 [1,]    1 20.87    0  0.00    0  0.00
 [2,]    1 20.00    0  0.00    0  0.00
 [3,]    1 21.50    0  0.00    0  0.00
 [4,]    0  0.00    1 20.87    0  0.00
 [5,]    0  0.00    1 20.00    0  0.00
 [6,]    0  0.00    1 21.50    0  0.00
 [7,]    0  0.00    0  0.00    1 13.74
 [8,]    0  0.00    0  0.00    1 13.33
 [9,]    0  0.00    0  0.00    1 13.43
> Xo
      [,1]  [,2]  [,3]  [,4] [,5]  [,6]  [,7] [,8]  [,9] [,10]    [,11]
 [1,]    1 20.87 14.44 3.865    0  0.00  0.00    0  0.00  0.00  0.00000
 [2,]    1 20.00 14.75 3.843    0  0.00  0.00    0  0.00  0.00  0.00000
 [3,]    1 21.50 14.51 3.947    0  0.00  0.00    0  0.00  0.00  0.00000
 [4,]    0  0.00  0.00 0.000    1 20.87 17.53    0  0.00  0.00  0.00000
 [5,]    0  0.00  0.00 0.000    1 20.00 18.52    0  0.00  0.00  0.00000
 [6,]    0  0.00  0.00 0.000    1 21.50 17.84    0  0.00  0.00  0.00000
 [7,]    0  0.00  0.00 0.000    0  0.00  0.00    1 13.74 14.44 -0.11073
 [8,]    0  0.00  0.00 0.000    0  0.00  0.00    1 13.33 14.75 -0.15863
 [9,]    0  0.00  0.00 0.000    0  0.00  0.00    1 13.43 14.51 -0.07604
> yo
[1] 24.6000 28.2500 25.4000  0.8660  0.6325  1.5166 12.3000 13.0000 14.1000
> t(muo)
      [,1]  [,2]  [,3]  [,4]  [,5] [,6]  [,7]  [,8]  [,9]
[1,] 27.25 25.24 28.47 1.385 1.453 1.43 12.21 11.97 12.09
> b<-C%*%t(Zo)%*%solve(Zo%*%Do%*%t(Zo)+Ro)%*%(yo-muo)
> varbb<-D-C%*%t(Zo)%*%solve(Zo%*%t(Do)%*%t(Zo)+Ro)%*%Zo%*%t(C)
> t(b)
       conD  hmaxD  conH    hmaxH     conV    hmaxV  conHDB    hmaxHDB conHCB  h2OHCB
[1,] -2.874 0.1077 0.245 -0.01338 -0.09844 0.002137 -0.3629 -0.0003529 0.1384 0.04219
> round(varbb,3)
Random effects variance covariance matrix
          conD  hmaxD   conH  hmaxH   conV  hmaxV conHDB hmaxHDB conHCB h2OHCB
conD     9.898 -0.515 -1.363  0.086 -0.364  0.032  2.262  -0.107 -0.975  0.011
hmaxD   -0.515  0.030  0.071 -0.004  0.020 -0.001 -0.119   0.006  0.048 -0.001
conH    -1.363  0.071  0.457 -0.029  0.036 -0.001 -0.368   0.017  0.102 -0.007
hmaxH    0.086 -0.004 -0.029  0.002 -0.001  0.000  0.021  -0.001 -0.011  0.001
conV    -0.364  0.020  0.036 -0.001  0.145 -0.009 -0.160   0.008 -0.041  0.001
hmaxV    0.032 -0.001 -0.001  0.000 -0.009  0.001  0.011  -0.001  0.000  0.000
conHDB   2.262 -0.119 -0.368  0.021 -0.160  0.011  0.594  -0.029 -0.165  0.002
hmaxHDB -0.107  0.006  0.017 -0.001  0.008 -0.001 -0.029   0.001  0.008  0.000
conHCB  -0.975  0.048  0.102 -0.011 -0.041  0.000 -0.165   0.008  0.205 -0.009
h2OHCB   0.011 -0.001 -0.007  0.001  0.001  0.000  0.002   0.000 -0.009  0.002
  Standard Deviations: 3.146 0.1732 0.676 0.04472 0.3808 0.03162 0.7707 0.03162 0.4528 0.04472
```

To illustrate the benefit from using the predicted random effects, Table 9.1 shows the observed values of all five response variables for 5 new trees of the same target stand. Predictions were made using fixed part only and using fixed and random parts, with the predicted random effects from vector `b`. In most cases, the predictions using fixed and random effects are closer to the observed values than the predictions based on fixed part only. The scripts for the prediction are shown below. The predictions of V and HDB were not adjusted for the back-transformation bias; an improved analysis should do this; see Section 10.2 and Example 5.19 (p. 163) of ML2020 and Example 6.6 on p. 17 of this document.

```
> new
```

Fixed effects only					Fixed part + random effects				
DBH	H	V	HDB	HCB	DBH	H	V	HDB	HCB
31.2	24.4	1.0	4.8	13.4	30.8	24.4	0.9	3.3	14.2
27.3	22.8	0.6	4.5	12.5	26.7	22.8	0.6	3.1	13.2
25.3	21.7	0.5	4.1	12.3	24.5	21.7	0.5	2.9	13.0
30.7	24.9	1.0	4.7	14.7	30.3	24.8	1.0	3.2	15.6
30.3	23.8	0.8	4.7	12.4	29.8	23.8	0.8	3.3	13.1

Field measurements				
DBH	H	V	HDB	HCB
26.9	24.1	0.63	0.4	15.9
28.3	22.4	0.65	1.7	13.2
21.2	21.2	0.34	0.8	12.4
29.8	24.2	0.77	4.0	14.0
32.4	23.9	0.90	0.4	13.1

TABLE 9.1 Predictions using the fixed part only, using the fixed part and the predicted random effects, and field-measured values of the five trees on the five response variables from the target stand. The random effects were predicted by using the observatoins of DBH, HDB, and HCB for three sample trees in Example 9.8.

```
   plot tree  DBH    H     V  HDB  HCB hmax  h20  h30  h70  h80 a_hmean a_veg a_h30 a_h70
10    1   29 26.9 24.1 0.627 0.40 15.9 23.0 15.6 16.9 19.8 20.3    14.9 0.906  15.6  18.8
11    1    8 28.2 22.4 0.645 1.70 13.2 21.3 13.9 14.7 18.7 19.2    15.5 0.901  15.6  18.6
12    1   46 21.1 21.2 0.345 0.75 12.4 20.1 13.7 15.5 17.8 18.4    14.9 0.905  15.1  18.1
13    1   14 29.8 24.2 0.769 4.00 14.0 23.0 17.5 18.3 21.1 21.5    15.5 0.935  15.8  18.7
14    1   45 32.4 23.9 0.899 0.40 13.1 22.7 13.8 14.9 18.5 19.3    15.1 0.906  15.7  18.9

> X<-adiag(cbind(1,new$hmax,new$a_hmean,sqrt(new$h30)),
+          cbind(1,new$hmax,new$h80),
+          cbind(1,new$hmax,log(new$a_h30),log(new$h70)),
+          cbind(1,new$hmax,new$a_h70),
+          cbind(1,new$h20,new$a_hmean,log(new$a_veg)))

> Z1234<-cbind(1,new[,"hmax"])
> Z5<-cbind(1,new[,"h20"])
> Z<-adiag(Z1234,Z1234,Z1234,Z1234,Z5)

> predf<-X%*%beta
> predfr<-predf+Z%*%b
```

The script below predicts also the residual errors for all responses. For DBH, HDB, and HCB, the predicted residuals are equal to the observed residuals $\boldsymbol{e}_o = \boldsymbol{y}_o - (\boldsymbol{X}_o\boldsymbol{\beta}_o + \boldsymbol{Z}_o\boldsymbol{b}_o)$, as shown below. For H and V they are also nonzero because the cross-model correlations of residual errors are nonzero.

```
> B<-Sigma[,-c(2,3)]%x%diag(ntrees)

> V1<-adiag(R,D)
> V12<-rbind(B,C%*%t(Zo))
> V2<-Zo%*%Do%*%t(Zo)+Ro
> h1hat<-V12%*%solve(V2)%*%(yo-muo)

> # The predicted residuals for DBH, H, V, HDB, and HCB
> matrix(h1hat[1:15],ncol=5)
      [,1]   [,2]   [,3]   [,4]   [,5]
[1,] -1.95 -0.197 -0.204 -0.146 -0.639
[2,]  3.78  0.133  0.402 -0.448  0.331
[3,] -2.43  0.140 -0.257  0.461  1.304
> # For observed responses, the residuals above equal to y-(xbeta+zb)
> t(yo-muo-Zo%*%b[-c(3:6)])
      [,1] [,2]  [,3]   [,4]   [,5]  [,6]   [,7]  [,8] [,9]
[1,] -1.95 3.78 -2.43 -0.146 -0.448 0.461 -0.639 0.331  1.3
```

10

Additional Topics on Regression

Example 10.1. Consider the OLS model of example 4.8. The sample mean and sample variance-covariance matrix of vector $\begin{pmatrix} y \\ \boldsymbol{x} \end{pmatrix}$ are computed below:

```
> data(stumplift)
> M2<-as.numeric(stumplift$Machine==2)
> M3<-as.numeric(stumplift$Machine==3)
> D<-stumplift$Diameter
> X<-cbind(D,M2,M3,D^2,M2*D,M3*D,M2*D^2,M3*D^2)
> y<-stumplift$Time
> yX<-cbind(y,X)

> yXbar<-apply(yX,2,mean)
> SigmaHat<-var(yX)
```

The script below shows that these matrix include all information that is needed to construct the model. BLUP leads to the same estimates of regression coefficients than OLS. The prediction variances are related through

$$\widehat{\sigma}^2_{BLP} = \frac{n}{n-1}\widehat{\sigma}^2_{ML} = \frac{n-p-1}{n-1}\widehat{\sigma}^2_{REML}\,,$$

where p is the number of predictors excluding the intercept.

```
> n<-dim(stumplift)[1]
> V12<-SigmaHat[1,-1]
> V1<-SigmaHat[1,1]
> V2<-SigmaHat[-1,-1]
> mu1<-yXbar[1]
> mu2<-yXbar[-1]

> # Intercept
> mu1-V12%*%solve(V2)%*%mu2
          [,1]
[1,] -26.82633
> # other regression coefficients
> V12%*%solve(V2)
           D       M2       M3
[1,] 2.653878 116.5469 77.40427 -0.01414773 -6.749991 -5.020219 0.1189522

[1,] 0.09462626

> sqrt(V1-V12%*%solve(V2)%*%V12)
         [,1]
[1,] 24.58621

> mod1ML<-gls(Time~Diameter+Machine*Diameter+Machine*I(Diameter^2),
+           data=stumplift,method="ML")
>
> sqrt(n/(n-1)*mod1$sigma^2)
[1] 24.58621
> coef(mod1ML)
          (Intercept)              Diameter              Machine2
          -26.82632510            2.65387823          116.54694066
             Machine3          I(Diameter^2)      Diameter:Machine2
           77.40426665           -0.01414773           -6.74999139
    Diameter:Machine3 Machine2:I(Diameter^2) Machine3:I(Diameter^2)
           -5.02021866            0.11895223            0.09462626
```

11

Modeling Tree Size

Example 11.4. Moment-based estimation of logit-logistic function parameters. We may not have expressions for the population moments available, but one may use general numerical integration. For example, finding computationally convenient expressions for the moments of the logit-logistic distribution (Equation 11.1, p. 338 of ML2020) might be rather time-consuming. However, that is not necessary, generally applicable numerical integration algorithms can be used instead. The following script was developed to implement the method of moments for all parameters of the log-logistic distribution.

```
> dmin<-min(d22)-1
> range<-max(d22)-min(d22)+2
> LLmom<-function(n,lxi,llambda,lsigma,mu) {
+        sigma<-exp(lsigma)/(1+exp(lsigma))
+        xi<-exp(lxi)
+        lambda<-exp(llambda)
+        f<-function(x) x^n*dll(x,mu,sigma,xi,lambda)
+        integrate(f,exp(lxi),exp(lxi)+exp(llambda))$value
+        }
>
> fnlist<-list(function(theta) {
+              print(theta);LLmom(1,theta[1],theta[2],theta[3],theta[4])-mean(d22)
+              },
+         function(theta) LLmom(2,theta[1],theta[2],theta[3],theta[4])-mean(d22^2),
+         function(theta) LLmom(3,theta[1],theta[2],theta[3],theta[4])-mean(d22^3),
+         function(theta) LLmom(4,theta[1],theta[2],theta[3],theta[4])-mean(d22^4))
>
> NRnum(c(log(dmin),log(range),0,0),fnlist)
[1] 1.740466 3.529297 0.000000 0.000000
...
[1] -446.250623    5.964379   -3.062894    1.816521
Error in solve.default(grad, -value) :
  Lapack routine dgesv: system is exactly singular: U[1,1] = 0
```

As seen in the output, we run to computational problems, where we are trying to invert a singular matrix. We do not make further efforts to find moment estimates for all parameters. Instead, we simplified the task by setting the minimum and maximum of the distribution to $\min(d) - 1$ and $\max(d) + 1$, respectively. Using these restrictions, the moment estimates of the remaining parameters are $\widehat{\sigma}_{MOM} \approx \frac{e^{0.298}}{1+e^{0.298}} \approx 0.574$ and $\widehat{\mu}_{MOM} \approx 0.589$. The *pdf* based on these parameter estimates is shown in Figure 11.2 using the gray solid line; see also Figure 11.2 on p. 339 of ML2020.

```
> fnlist<-list(function(theta) LLmom(1,log(dmin),log(range),theta[1],theta[2])-mean(d22),
+              function(theta) LLmom(2,log(dmin),log(range),theta[1],theta[2])-mean(d22^2))
>
> print(estMOMLL<-NRnum(c(0,0),fnlist))
$par
[1] 0.2980984 0.5891041

$value
[1] 6.274092e-12 2.429374e-09
```

The numerical approximation of the population moments using `integrate` worked nicely in this example. However, one should bear in mind the possibility of large approximation errors especially if the endpoints of integral are $\pm\infty$.

Example 11.12. We define a dominant tree as a tree that is among the 100 largest trees per ha. The dominant height is defined as the expected value of dominant tree heights. The diameter

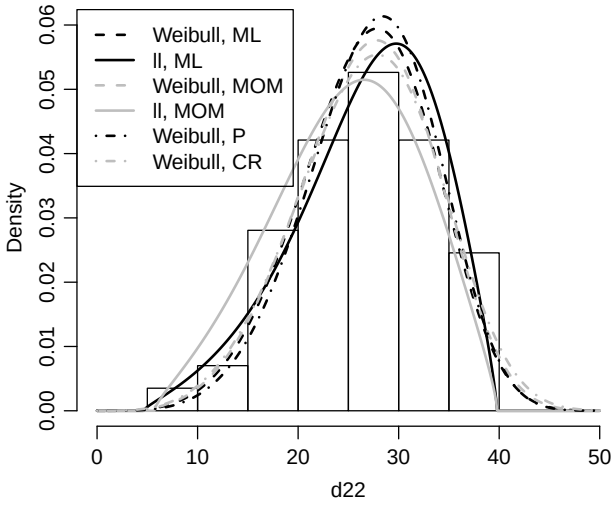


FIGURE 11.2 The histogram of tree diameters on plot 22 of dataset `spati` and the fitted logit-logistic and Weibull functions from Examples 11.2 – 11.7.

distribution of the dominant trees is obtained by left-truncating the diameter distribution at the quantile that corresponds the limit of dominant trees and rescaling so that the area under the *pdf* is unity. The height distribution of the dominant trees can be computed correspondingly by using the height limit of dominant trees.

Assuming that the stand density is N=500 stems/ha and the dominant trees are the 100 tallest trees per hectare, the height limit of dominant trees is the 80^{th} percentile of the height distribution: $H_{lim} = F_H^{-1}((N-100)/N)$. Computing this would require evaluations of $F_H(h) = \int_0^h f_H(h)$ and further computing its inverse. These all would be possible, e.g., by using function `integrate` and `lmfor::updown`, but a much easier way to find the limit is to determine the height that corresponds to the 80^{th} percentile of the diameter distribution as

$$H_{lim} = h(F_D^{-1}((N-100)/N))\,.$$

```
> N<-500
> domlim<-korfhd(qweibull((N-100)/N,alpha,beta))
> domlim
[1] 19.89572
```

The dominant height is the mean height of the dominant trees,

$$\begin{aligned} H_{dom} &= \mathrm{E}(H|H > H_{lim}) \\ &= \frac{N}{100}\int_{H_{lim}}^{\infty} u f_H(u)\mathrm{d}u \approx 20.45\mathrm{m}\,. \end{aligned}$$

Another option would be to use equation (2.10) (p. 15 of ML2020). For this purpose, we need to compute the diameter limit of dominant trees,

$$D_{lim} = F_D^{-1}((N-100)/N) \approx 16.90\mathrm{cm}\,.$$

The expected value of $h(d)$ over the diameter distribution of dominant trees yields

$$\begin{aligned} H_{dom} &= \mathrm{E}(h(D)|D > D_{lim}) \\ &= \frac{N}{100}\int_{D_{lim}}^{\infty} f_D(u)h(u)du \approx 20.45\mathrm{m}\,. \end{aligned}$$

The numerical values from both approximations are the same up to the 5^{th} decimal place.

```
> N/100*integrate(function(y) y*dhdist(y,alpha,beta,a,b),domlim,26.3)$value
[1] 20.44752

> Domlim<-qweibull((N-100)/N,alpha,beta)
> Domlim
[1] 16.89506

> integrate(function(x) korfhd(x)*dweibull(x,alpha,beta)*N/100,
+           qweibull((N-100)/N,alpha,beta),Inf)
20.44752 with absolute error < 3e-07
```

Example 11.13. The distribution of volume Let us consider tree volume as a transformation, which is obtained from tree diameter using the volume function of Laasasenaho (1982) for Scots pine,

$$v = g(d) = e^{-5.394+3.481\ln(2+1.25d)-0.0399d},$$

where, following the notations of Section 2.2.3 of ML2020, function g defines a transformation of tree diameter. The distribution of volume becomes

$$F_V(v) = F_D(g^{-1}(v))\,.$$

Unfortunately, the inverse transformation $g^{-1}(v)$ cannot be computed in a closed form. However, an easy way to compute the inverse transformation is to use the step halving algorithm of `lmfor::updown`. The step halving algorithm can be used to solve equations of form $f(x) = 0$ numerically. The user needs to give endpoints of an interval from which the root is searched for so that the sign of a continuous $f(x)$ changes between the endpoints. If the function is continuous and there is only one root at the interval, the algorithm will find it. The Newton-Raphson algorithm of `lmfor::NR` or `lmfor::NRnum` would be a faster option, implementation using that is left as an exercise. To compute the value of the inverse volume function for volume v, we solve $g(d) - v = 0$ for d to get the diameter that corresponds to volume v. This diameter is then substituted into the Weibull *cdf* to get the *cdf* of volume. Figure 11.3 shows the graphs of the inverse transformation and the distribution of volume. Making a graph of the density would have required the use of numerical differentiation (one could use `R`-function `numericDeriv` for this), but it is left as an exercise for those interested in doing it. The code below produces the graph of *cdf* shown in Figure 11.3. The applied volume function is available at `lmfor::predvol`.

```
> # diameter for a given volume, scalar argument v
> ginv.scal<-function(v) {
+            voldif<-function(d) predvol(species=1,d=d)-v
+            updown(0,80,voldif)
+            }
> # as above, for vector argument v
> ginv<-function(v) sapply(v,ginv.scal)
>
> # cdf of volume
> Fv<-function(x,alpha,beta) pweibull(ginv(x),alpha,beta)
> v<-seq(1,400,1)
> plot(v,Fv(v,alpha,beta),type="l",xlab="Volume, liters",ylab="Cumulative density")
```

Example 11.17. Fitting basal area weighted Weibull function to an angle count sample. We now assume that the basal-area weighted diameter distribution is of the Weibull form, and the data are collected using angle-count sampling. We fit the Weibull distribution to the angle count sample of Example 11.15 on p. 354 of ML2020, ignoring the fact that the Weibull form was assumed for the unweighted diameter distribution when the data were generated. The log-likelihood is similar to that of Example 11.16 on on p. 354 of ML2020:

$$l(\alpha, \beta | d_1, \ldots, d_n) = \sum \ln(f_D^g(d|\alpha, \beta))\,, \tag{11.1}$$

where the basal-area weighted density f_D^g is assumed to be of the Weibull form. The ML-fit gives the estimates $\widehat{\alpha} = 3.60$ and $\widehat{\beta} = 23.27$. We cannot compare these to the true values; the true values cannot be specified because a basal-area weighted Weibull distribution was not assumed when the data were generated.

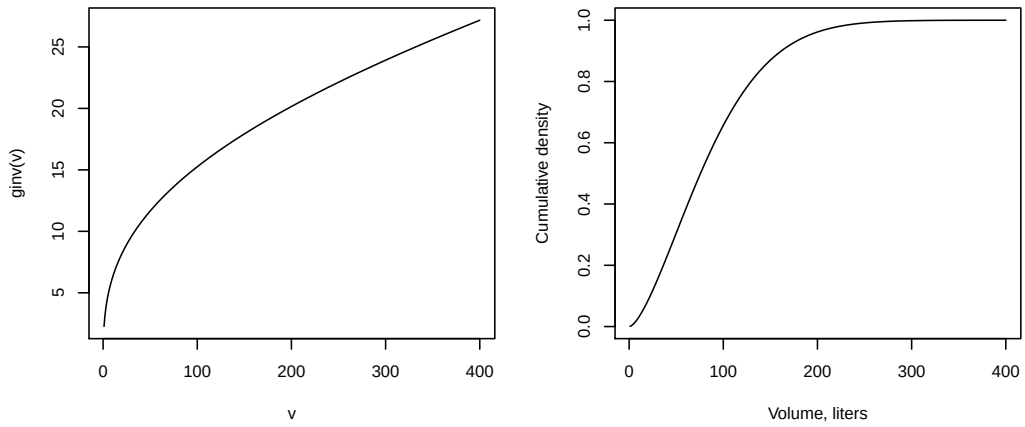


FIGURE 11.3 Tree diameter as a function of volume (left) and the resulting *cdf* of volume when diameter follows $Weibull(4, 15)$ distribution (right). See Example 11.13 for details.

```
> ll2<-function(shape,scale) {
+      -sum(log(dweibull(dsample,shape,scale)))
+ }
>
> print(est.ml2<-mle(ll2,start=list(shape=3,scale=20)))

Call:
mle(minuslogl = ll2, start = list(shape = 3, scale = 20))

Coefficients:
    shape     scale
 5.133115 24.426388
> vcov(est.ml2)
          shape    scale
shape 0.7116284 0.261183
scale 0.2611830 1.039362
```

Example 11.18. Another alternative for model fitting in the situation of Example 11.17 is the method of moments (see Section 11.2.2 of ML2020). The two first moments of the angle count sample are computed and set equal to the theoretical moments of the sampling distribution:

$$\int_0^\infty u f_D^g(u|\alpha,\beta)du - \frac{1}{n}\sum_{i=1}^n d_i = 0$$
$$\int_0^\infty u^2 f_D^g(u|\alpha,\beta)du - \frac{1}{n}\sum_{i=1}^n d_i^2 = 0\,.$$

The sample moments were 22.41 cm and 529.94 cm^2. Solution using `lmfor::NRnum` gave $\widehat{\alpha}_{MOM} = 3.739$ and $\widehat{\beta}_{MOM} = 21.36$.

```
> m1<-mean(dsample)
> m2<-mean(dsample^2)
> m1
[1] 22.41906
> m2
[1] 529.9396

> # A function that numerically computes the moments of the sampling distribution
> wmom2<-function(n,alpha,beta) {
+        f1<-function(x) x^n*dsampling(x,alpha,beta)
+        integrate(f1,0,qweibull(1-1e-10,alpha,beta))$value
+        }
```

```
> # Define the individual functions and put them into a list
> fn1<-function(parms) wmom2(1,parms[1],parms[2])-m1
> fn2<-function(parms) wmom2(2,parms[1],parms[2])-m2
> fnlist<-list(fn1,fn2)

> # Solve the parameters
> NRnum(c(3,20),fnlist)
$par
[1]  3.73890 21.35684

$value
[1] 6.449273e-09 3.786901e-07
```

Example 11.21. Overlapping crowns in aerial forest inventory. In an aerial forest inventory, tree size can be described by the crown projection at the ground. For simplicity, we call this projection the crown, its area the crown area, and the radius of a disc with that area the crown radius. Let us assume that the radii Z within a sample plot are independent identically distributed random variables with the *pdf* $f_Z(z)$. Furthermore, assume that tree crowns are discs centered at the tree locations, and a tree remains undetected in the inventory if the tree location falls within a crown of a larger tree. Otherwise, the tree is detected and the radius of the crown is determined correctly. Assume a complete spatial randomness with density λ trees per m^2 for a forest stand (recall Example 8.6, p. 266 of ML2020). This model is a special case of so-called Boolean model, which is discussed in more detail in the literature of stochastic geometry (Chiu *et al.* 2013).

Under the above-specified assumptions, the probability of a tree being detected, the *detectability*, depends on the crown radius according to (Mehtätalo 2006)

$$w(z) = \exp\left\{-\lambda\pi \int_z^\infty t^2 f_Z(t|\alpha,\beta)dt\right\}, \tag{11.2}$$

see the left panel of Figure 11.5 for illustration.

An interesting feature of the Boolean model is that the area fraction (the proportion of the area covered by the crowns, i.e. the theoretical canopy closure) can be expressed as

$$p = 1 - \exp\{-\lambda\pi\,\mathrm{E}(Z^2)\}$$

Solving it for λ gives

$$\lambda = -\frac{\ln(1-p)}{\pi\,\mathrm{E}(Z^2)} \tag{11.3}$$

Using this in (11.2) gives such expression for the detectability that does not include λ:

$$w(z) = (1-p)^{\frac{1}{\mathrm{E}(Z^2)}\int_z^\infty t^2 f_Z(t|\alpha,\beta)dt}.$$

Here p is the expected canopy closure, which can be replaced by its observed value in applications. The sampling distribution in aerial inventory is that of the detectable crown radii. In our model, it is

$$f_Z^w(z|\alpha,\beta) = \frac{w(z|\alpha,\beta)f_Z(z|\alpha,\beta)}{\int_0^\infty w(u|\alpha,\beta)f_Z(u|\alpha,\beta)\mathrm{d}u}. \tag{11.4}$$

Parameters α, β and λ can be estimated by fitting the above distribution to the observed sample of crown radii. An alternative estimator for λ is based on Equation (11.3).

For illustration, consider a one-hectare forest stand that follows the above-specified model, using stand density of 1000 trees per hectare (0.01 trees per m^2) and crown radii from $Weibull(3,2)$ distribution. A realization of such a forest stand is shown in Figure 11.4.

```
> # Diameters follow Weibull(shape,scale) distribution
> simulate.trees<-function(n,plot=TRUE,shape=3,scale=2) {
+  n<-rpois(1,n*(110^2/100^2)) # A 5 meter buffer
+  # generate crown radii and tree locations
+  radius<-rweibull(n,shape,scale)
+  xk<-runif(n,-5,105)
+  yk<-runif(n,-5,105)
+  data.frame(x=xk,y=yk,r=radius,r2=radius^2)
```

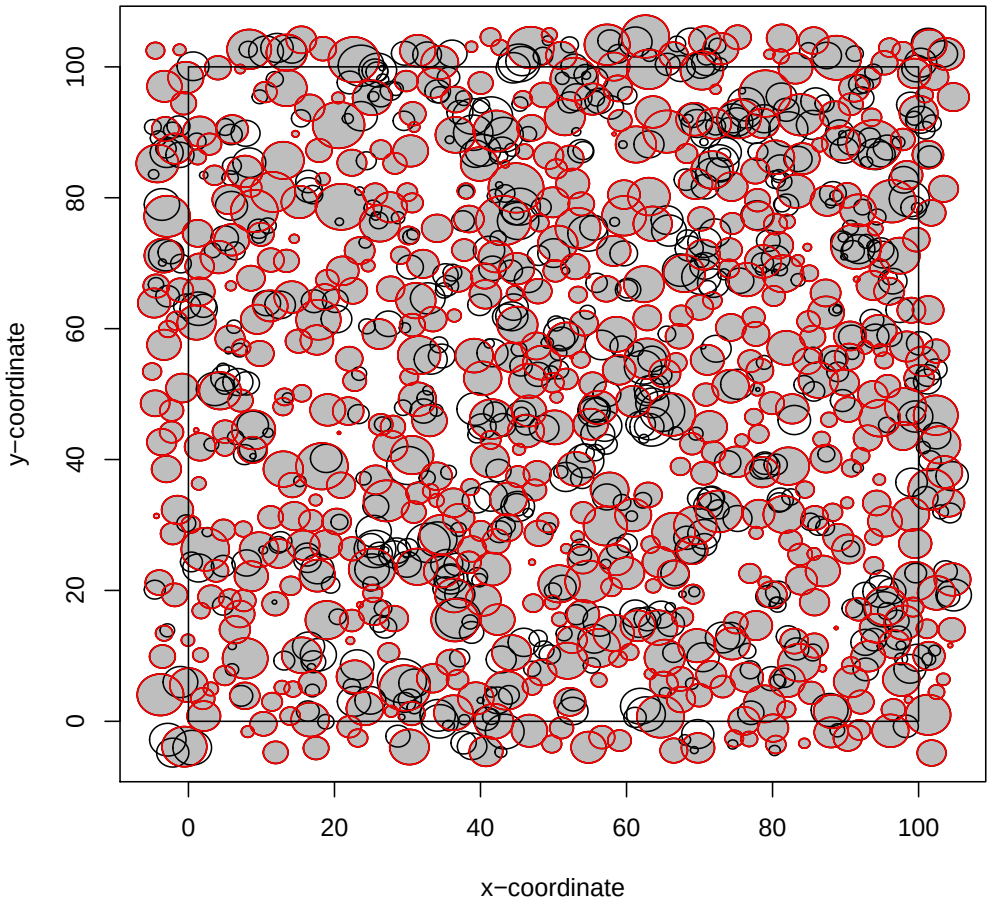


FIGURE 11.4 An illustration of the detectability condition in Example 11.21 in a one-hectare stand where stand density is 1000 trees per ha and crown radius follows the $Weibull(3, 2)$ distribution. The crowns with red borders and gray shading are detectable, whereas the crowns with black thin borders are not.

```
+ }
>
> # this function determines whether a tree in data "trees" is visible
> # and adds binary column ""
> detectable<-function(n=500,trees=simulate.trees(n)) {
+  dtable<-rep(NA,dim(trees)[1]) # binary vector, is tree detectable
+  for (i in 1:dim(trees)[1]) {
+  trees1<-trees[trees$r>trees$r[i],]
+  dtable[i]<-sum(sqrt((trees1$x-trees$x[i])^2+(trees1$y-trees$y[i])^2)<trees1$r)==0
+  }
+  cbind(trees,detectable=dtable)
+ }
>
> trees<-detectable(1000)
```

We compute the empirical canopy closure by placing a 400 by 400 point square grid over the area and computing the proportion of nodes covered by tree crowns. For this particular

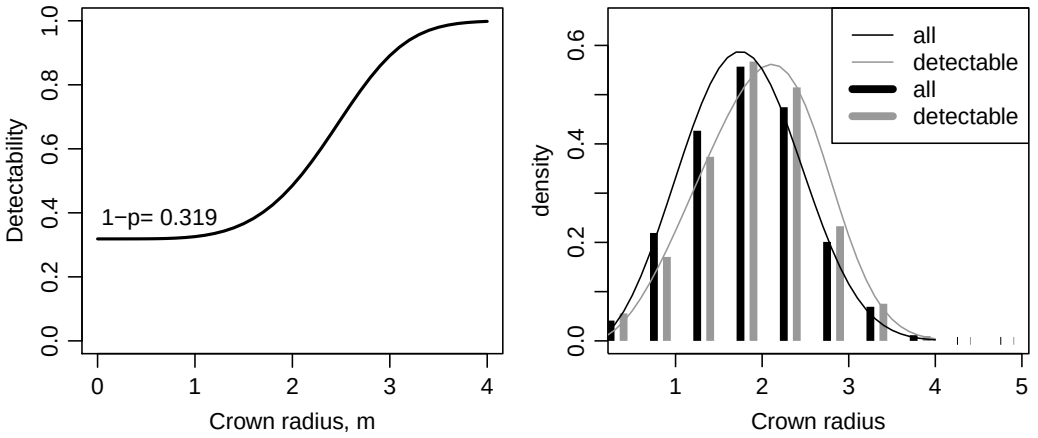


FIGURE 11.5 The plot on the left shows the probability of a tree being observed, $w(z)$ in the population illustrated in Figure 11.4. The histograms illustrate the distribution of crown radii for all trees and for detectable trees; the solid lines show the corresponding true pdf.

realization, we got $\widehat{p} = 0.681$, i.e. about 68% of the area is covered by tree crowns. The theoretical canopy closure in this model is 0.678.

```
> # Compute empirical canopy cover by exploring which proportion
> # of the nodes of rectangular grod are covered by tree crown discs.
> canopyCover<-function(trees,xlim=c(0,100),ylim=c(0,100),dens=c(400,400),plot=TRUE) {
+     stepx<-(xlim[2]-xlim[1])/(dens[1]+1)
+     stepy<-(ylim[2]-ylim[1])/(dens[2]+1)
+     xpt<-seq(xlim[1]+stepx/2,xlim[2],stepx)
+     ypt<-seq(ylim[1]+stepy/2,ylim[2],stepy)
+     grid<-matrix(0,length(ypt),length(xpt))
+     # Count how many tree crowns cover each point
+     for (x in (1:length(xpt))) {
+         for (y in (1:length(ypt))) {
+             grid[y,x]<-sum(sqrt((trees$x-xpt[x])^2+(trees$y-ypt[y])^2)<=trees$r)
+         }
+     }
+     grid[grid>=1]<-1
+     sum(grid)/length(grid)
+     }
>
> print(p<-canopyCover(trees))
[1] 0.6814137
>
> # theoretical caopy cover
> Ez2<-2^2*gamma(1+2/3)
> 1-exp(-0.1*pi*Ez2)
[1] 0.6783924
```

The left panel of Figure 11.5 illustrates the detectability as a function of crown radius for the assumed true model, and the right-hand graph illustrates the distribution of crown radii both for all trees and for the detectable ones. The distributions are clearly different, for example the expected value is clearly higher for the detectable trees than for all trees. In the simulated stand, the mean squared radius is 3.61 m^2 for all trees and 4.42 m^2 for detectable trees.

```
> # remove the buffer zone trees
> trees<-trees[trees$x>0&trees$x<100&trees$y>0&trees$y<100,]
> print(meanr2<-mean(trees$r^2))
[1] 3.607218
> print(meanr2Obs<-mean(trees$r[trees$detectable]^2))
[1] 4.420905
```

We could estimate the stand density by using Equation (11.3), but it requires knowledge

about the expected squared crown radius in the population of all trees. If the mean squared crown radius of the generated stand is used, we get the estimate 1009 trees per ha, which is very close to the true density of 1000 trees per ha. If the mean squared radius of detectable trees is used, we get much larger underestimation, 823 trees per ha. The additional underestimation results from the difference in the means of all trees and detectable trees.

```
> lambdaest<--log(1-p)/(pi*meanr2) # density, trees per m^2
> lambdaestWrong<--log(1-p)/(pi*meanr2Obs) # density, trees per m^2
> 10000*c(lambdaest,lambdaestWrong)
[1] 1009.3720  823.5927
```

In practice, the expected crown radius could be estimated by fitting the weighted density (Equation (11.4)) to the observed crown data and using the resulting parameter estimates to find the unweighted expected value Z. The functions below implement the weighted *pdf* (Equation (11.4)) in this situation; they were used to produce the gray lines in Figure 11.5.

```
> # detectability as a function of crown radius
> w<-function(r,shape,scale,p=0) {
+    Ex2<-scale^2*gamma(1+2/shape)
+    lambda<--log(1-p)/(pi*Ex2)
+    fun<-function(r0) exp(-lambda*pi*integrate(function(x) x^2*dweibull(x,shape=shape,
+         scale=scale),r0,Inf,rel.tol=10e-8)$value)
+    sapply(r,fun)
+ }
>
> # Censored pdf
> # parameters are weibull shape, scale and logit of the canopy cover
> cenpdf<-function(r,shape,scale,p) {
+     denom<-integrate(function(x) dweibull(x,shape,scale)*w(x,shape,scale,p),0,Inf,
+           rel.tol=10e-8)$value
+     dweibull(r,shape,scale)*w(r,shape,scale,p)/denom
+     }
```

ML fitting of the weighted distribution was also implemented by (1) estimating only the Weibull parameters and conditioning on the observed canopy closure $\widehat{p}$ and (2) by estimating also the p using ML. Essential difference between these two methods is that the former uses information about canopy closure and the size distribution of the detectable trees in estimation, whereas the latter does not utilize information about the canopy closure. The estimated stand densities using these approaches were 1025 and 2248 trees per ha, respectively. The large overestimation in the latter case is caused by severe overestimation of canopy closure parameter ($\widehat{p}_{ML} = 0.870$), of which the observed sample of detectable crown radii does not provide that much information.

```
> r<-trees$r[trees$detectable]
> pobs<-p
> # Use the empirical canopy closure as p
> ll1<-function(shape=3,scale=2) {
+     cat(shape,scale," ")
+     value<--sum(log(cenpdf(r,shape,scale,pobs)))
+     cat(value,"\n")
+     value
+     }
>
> est1<-try(mle(ll1,method="L-BFGS-B",lower=c(0.1,0.1)))
3 2  515.6248
< . . . >
2.933559 1.978251  515.3752
> 10000*(-log(1-pobs)/(pi*coef(est1)[2]^2*gamma(1+2/coef(est1)[1])))
   scale
1025.618
> # Estimate also p using ML
> ll2<-function(shape=3,scale=2,p=0.5) {
+  cat(shape,scale,p," ")
+  value<--sum(log(cenpdf(r,shape,scale,p)))
+  cat(value,"\n")
+  value
+     }
>
> est2<-try(mle(ll2,method="L-BFGS-B",lower=c(0.1,0.1,0.1)))
```

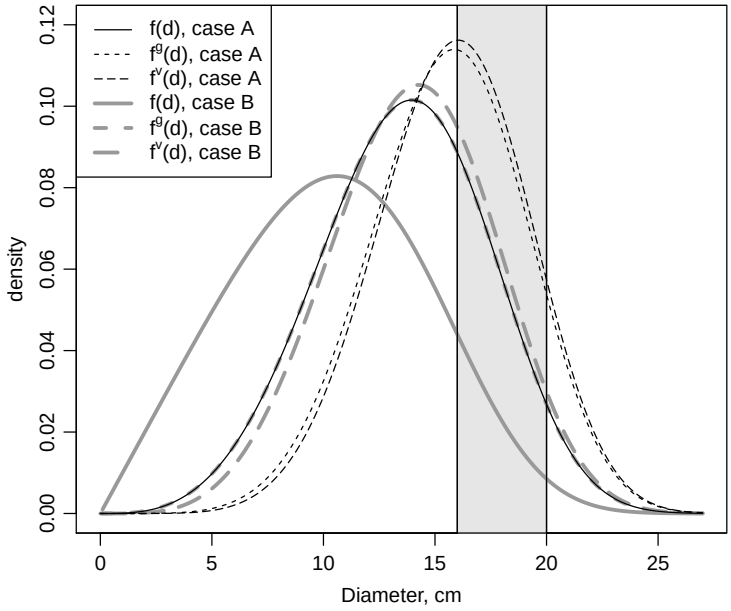


FIGURE 11.11 The unweighted, basal-area weighted, and volume-weighted densities of the diameter distribution when the unweighted distribution is $Weibull(4,15)$ (case A) and when the basal-area-weighted distribution is $Weibull(4,15)$ (case B). The gray shading illustrates the diameter class $(16, 20]$

```
3 2 0.5  519.0886
< . . . >
2.63801 1.779568 0.8704597  514.8365
> 10000*(-log(1-coef(est2)[3])/(pi*coef(est2)[2]^2*gamma(1+2/coef(est2)[1])))
       p
2247.573
```

This example was based on Mehtätalo (2006). The approach has also been extended to the situation where the assumption about complete spatial randomness is significantly relaxed and no parametric distribution is assumed for crown radii, see Kansanen *et al.* (2016) and function `lmfor::HTest` for details.

Example 11.25. Scaling basal-area-weighted diameter distribution by total volume. Consider a forest stand where the basal-area-weighted diameter distribution is of the Weibull form with parameters $\alpha = 4$ and $\beta = 15$. The volume function and H-D curve are the same as in Example 11.24 (on p. 362 of ML2020). The total volume per ha is available and used for scaling the basal-area weighted diameter distribution so that any the class densities based on the distribution give the total basal area per ha in the class, and the total volume of the growing stock equals to the given volume per ha.

The ratio of the total volume and basal area in the assumed diameter distribution is 8.89 m (i.e., if the tree stems were cylinders of equal height, they would be 8.89 meters tall), giving $\tau_{12} = 1/8.89$. The volume 200 m^3/ha therefore corresponds to basal area 22.50 m^2/ha. The diameter-class specific number of stems, basal area, and volume were $N_{(16,20]} = 216$ trees/ha, $G_{(16,20]} = 5.21$ m^2/ha, and $V_{(16,20]} = 50.56$ m^3/ha, which correspond to 9.6%, 23.2%, and 25.3% of the total N, G, and V, respectively. The stand density corresponding to the known volume is 2256 trees/ha. The gray lines in Figure 11.11 demonstrate the unweighted, basal-area weighted, and volume-weighted densities; see also Figure 11.11 on p. 363 of ML2020.

```
> print(VperG<-integrate(function(d) 40000/pi*vol2(d)/d^2*dweibull(d,4,15),
>              0,Inf)$value/1000)
[1] 8.890101
```

```
> tau12<-1/VperG
> print(G<-tau12*V)
[1] 22.49693
> print(Nclass<-G*integrate(function(d) 1/(pi*d^2/40000)*dweibull(d,4,15),16,20)$value)
[1] 215.8777
> print(Gclass<-G*(pweibull(20,4,15)-pweibull(16,4,15)))
[1] 5.21073
> print(Vclass<-G*integrate(function(d) vol2(d)/1000*1/(pi*d^2/40000)*dweibull(d,4,15),
>               16,20)$value)
[1] 50.55984
> print(N<-G*integrate(function(d) 1/(pi*d^2/40000)*dweibull(d,4,15),0,50)$value)
[1] 2256.451
> c(Nclass/N, Gclass/G, Vclass/V)
[1] 0.09567134 0.23161956 0.25279918
```

Example 11.30. Assume that F_D^g is a two-parameter Weibull distribution. Recover such values for α and β that yield $\widehat{V}$=200 m^3/ha, $\widehat{G}$=20 m^2/ha and, $\bar{h}$=16 m. The volume function and H-D curve are as specified in Example 11.24 on p. 362 of ML2020.

The parameter estimates should simultaneously fulfil

$$\frac{40000\widehat{G}}{\pi}\int_0^\infty f_D^g(u|\alpha,\beta)u^{-2}v(u,h(u))\mathrm{d}u - \widehat{V} = 0$$
$$\frac{\int_0^\infty f_D^g(u|\alpha,\beta)u^{-2}h(u)du}{\int_0^\infty f_D^g(u|\alpha,\beta)u^{-2}du} - \bar{h} = 0\,.$$

Solving the system using the bivariate Newton's method gave $\widehat{\alpha} = 2.95$ and $\widehat{\beta} = 29.8$.

```
> V<-200; G<-20; H<-16
> a<-25; b<--5
>
> # The height-weighted density, sums up to the total height/m^2
> hHw<-function(x,alpha,beta,a,b) {
+      hNw(x,alpha,beta)*korfhd(x,a,b)
+ }

> # The corresponding distribution function with numerical integration.
> HHw<-function(x,alpha,beta,a,b) {
+      integrate(f=function(u) hHw(u,alpha,beta,a,b),lower=0,upper=x)$value
+      }

> f1<-function(theta) {
+     val<-G*HVw(1000,theta[1],theta[2],a,b)/1000-V
+     cat(theta[1],theta[2],val,"\n")
+     val
+     }
> f2<-function(theta) {
+     val<-HHw(1000,theta[1],theta[2],a,b)/HNw(1000,theta[1],theta[2])-H
+     cat(theta[1],theta[2],val,"\n")
+     val
+     }

> fn<-list(f1,f2)
> NRnum(c(4,15),fn)
4 15 -22.19466
4 15 -0.4595083
 < part of the output has been removed >
2.950265 29.80601 0
2.950265 29.80601 7.105427e-15
$par
[1]  2.950265 29.806012

$value
[1] 0.000000e+00 7.105427e-15
```

Example 11.31. Simultaneous recovery of the H-D curve and diameter distribution (Mehtätalo *et al.* 2007). Assume that a forest inventory has been carried out by using ALS, and estimates of the number of stems ($\widehat{N}$), total volume ($\widehat{V}$), basal-area median diameter ($\widehat{DGM}$), and height of basal-area median tree ($\widehat{HGM}$) are available for a sample plot. In addition, a ground based survey has been conducted, including diameter and height measurements for all trees on

the circular, 9-meter radius plot. The ALS- and ground- based estimates for the characteristics are tabulated below.

Variable	ALS	Ground
N, trees per ha	473.41	747
DGM, cm	28.69	25.60
HGM, m	20.60	19.80
V, m^3 per ha	248.60	288.02

Let us assume that diameter follows the Weibull distribution and the stand-specific H-D curve is of form

$$h(d|A) = \max(1.4, \exp(A - By(d)))\,,$$

where

$$y(d) = \frac{(d+7)^c - (DGM+17)^c}{17^c - 37^c}\,,$$

and $c = 0.98 + 0.058DGM$. The shape parameter of the H-D curve was predicted on DGM using $B = 0.62 - 0.027DGM + 0.00094DGM^2$ (Mehtätalo 2005a). The same volume model was used as in Example Example 11.24 on p. 362 of ML2020.

We implement a recovery algorithm based on the following system of equations

$$\begin{aligned} g_V(\alpha,\beta,A|\widehat{N}) - \widehat{V} &= 0 \\ g_{DGM}(\alpha,\beta,A) - \widehat{DGM} &= 0 \\ g_{HGM}(\alpha,\beta,A) - \widehat{HGM} &= 0\,, \end{aligned}$$

where the functions g_V, g_{DGM}, and g_{HGM} give the standing volume, DGM, and HGM, respectively, as a function of Weibull parameters α and β and the asymptote parameter A of the H-D curve for known $\widehat{N}$. The functions are defined as follows:

$$\begin{aligned} g_V(\alpha,\beta,A|\widehat{N}) &= \widehat{N}\int_0^\infty f_D(u|\alpha,\beta)v(u,h(d|A))du \\ g_{DGM}(\alpha,\beta) &= F_D^{g,-1}(0.5|\alpha,\beta) \\ g_{HGM}(\alpha,\beta,A) - \widehat{HGM} &= h(g_{DGM}(\alpha,\beta)|A)\,, \end{aligned}$$

where $F_D^{g,-1}$ is the quantile function corresponding to the basal-area weighted diameter distribution and $f_D(u|\alpha,\beta)$ the assumed unweighted Weibull *pdf*. See equations 11.5 and 11.7 on p. 353 of ML2020 for the definition of $f_D^g(u)$.

A simple but computationally inefficient way for estimation is to minimize the sum of squared differences

$$\begin{aligned} \left(g_V(\alpha,\beta,A|\widehat{N}) - \widehat{V}\right)^2 &+ \left(g_{DGM}(\alpha,\beta,A) - \widehat{DGM}\right)^2 \\ &+ \left(g_{HGM}(\alpha,\beta,A) - \widehat{HGM}\right)^2 \end{aligned}$$

with respect to α, β, and A. The solution fulfils the recovery equations if the value of the function is 0 at the solution. If the objective function is nonzero at the optimum, the solution might be useful if the weights for the three components in a wise way; the above definition uses unit weights for each so that one-unit squared difference in squared volume, diameter, and height (in the applied units) are equally harmful. Function `recov2` implements this procedure. The function will find estimates even with quite poor starting values. Another alternative is the Newton's method implemented in `lmfor::NRnum`, which is used in function `recov` below. This is a quick algorithm, but requires good initial estimates. It converges to a solution only if all equations are fulfilled. Function `recov3` combines these two, doing the estimation using `recov` by using initial estimates found using `recov2`.

Using starting values $A = 3$, $\alpha = 5$, and $\beta = 15$, `recov` does not converge. However, function `recov2` finds estimates, but the value of the objective function is not exactly zero in the solution. Function `recov` converges if the estimates from `recov2` are given as the initial estimates, which confirms that a solution to the system exists but `recov2` just did not find it. The final estimates

are $\widehat{\alpha} = 4.872$, $\widehat{\beta} = 28.16$, and $\widehat{A} = 3.073$, which differ slightly from the initial estimates obtained using `recov2`. The estimates based on field data are $\widehat{\alpha} = 4.064$, $\widehat{\beta} = 24.603$, and $\widehat{A} = 3.047$.

Figure 11.13 shows the observed diameter distribution of the plot, as well as the tree heights. In addition, the recovered diameter distribution and H-D curve are shown based on ALS-estimates (solid) and field measurements (dashed). Because ALS overestimated DGM, the recovered distribution is located to the right from the true distribution. The H-D curves match well with each other and the true data. The shape of the diameter distribution is quite ok, because the ratio of N and V was quite well estimated, even though both these characteristics were underestimates. The distribution based on ground-based values fits well to the observed data.

```
> # Reparameterize tree diameter for the height model of Mehtatalo (2005)
> dexf<-function(x,DGM) {
+       lda<-7
+       c<-0.98227850+0.05753439*DGM
+       ((x+lda)^(-c)-(DGM+10+lda)^(-c))/((10+lda)^(-c)-(30+lda)^(-c))
+       }

> # Predict height for trees with given diameters x.
> # Uses a given scale parameter A for the H-D curve.
> # The shape parameter (B) is predicted using the model of Mehtatalo (2005)
> korfhd2<-function(x,A,DGM) {
+          B<-0.6156-0.02707*DGM+0.000935*DGM^2
+          pmax(1.4,exp(A-B*dexf(x,DGM)))
+          }
>
> # Predict volume for a given diameter and scale parameter of the height curve
> volf<-function(x,A,DGM) {
+       h<-korfhd2(x,A,DGM)
+       predvol(1,x,h,2)
+       }
>
> # Basal-area weighted distribution when unweighted is weibull.
> # Provides a scaled distribution that sums up to total basal area
> hG<-function(x,shape,scale,N) {
+     N*pi/40000*dweibull(x,shape,scale)*x^2
+     }
>
> # CDF corresponding to hG.
> # Scaled so that FG(Inf)=G
> HG<-function(x,shape,scale,N) {
+     sapply(x,function(y) integrate(function(u) hG(u,shape,scale,N),0,y)$value)
+     }
>
> # The median of hG,
> # i.e solution to HG=0.5*G
> DGMf<-function(shape,scale,N) {
+       G<-HG(qweibull(1-1e-10,shape,scale),shape,scale,N)
+       updown(l=qweibull(0.01,shape,scale),
+              u=qweibull(0.99,shape,scale),
+              fn=function(u) HG(u,shape,scale,N)-0.5*G,
+              crit=10)
+       }
>
> # The height of a DGM-tree, meters
> HGMf<-function(shape,scale,A,N,DGM) {
+       korfhd2(DGMf(shape,scale,N),A,DGM)
+       }
>
> # The distribution weighted by volume
> hV<-function(x,shape,scale,A,DGM,N) {
+     N/1000*dweibull(x,shape,scale)*volf(x,A,DGM)
+     }
>
> # The cdf corresponding to hV
> HV<-function(x,shape,scale,A,DGM,N) {
+     sapply(x,function(y) integrate(function(u) hV(u,shape,scale,A,DGM,N),0,y)$value)
+     }
>
> recov<-function(N,DGM,HGM,V,start=c(5,15,3)) {
+        theta<-log(start)
+        f1<-function(theta) HV(qweibull(1-1e-10,exp(theta[1]),exp(theta[2])),
+                               exp(theta[1]),exp(theta[2]),exp(theta[3]),DGM,N)-V
+        f2<-function(theta) DGMf(exp(theta[1]),exp(theta[2]),N)-DGM
```

```
+        f3<-function(theta) HGMf(exp(theta[1]),exp(theta[2]),exp(theta[3]),N,DGM)-HGM
+        fn<-list(f1,f2,f3)
+        exp(NRnum(theta,fn)$par)
+        }
>
> recov2<-function(N,DGM,HGM,V,start=c(5,15,3)) {
+         theta<-log(start)
+         f1<-function(theta) HV(qweibull(1-1e-10,exp(theta[1]),exp(theta[2])),
+                                exp(theta[1]),exp(theta[2]),exp(theta[3]),DGM,N)-V
+         f2<-function(theta) DGMf(exp(theta[1]),exp(theta[2]),N)-DGM
+         f3<-function(theta) HGMf(exp(theta[1]),exp(theta[2]),exp(theta[3]),N,DGM)-HGM
+         fn2<-function(theta) {
+              f1(theta)^2+f2(theta)^2+f3(theta)^2
+              }
+         est<-optim(log(start),fn2)
+         structure(exp(est$par),value=est$value)
+         }
>
> recov3<-function(N,DGM,HGM,V,start=c(5,DGM,3)) {
+         start2<-try(recov2(N,DGM,HGM,V,start=start))
+         if (class(start2)=="try-error") start2=start
+         result<-try(recov(N,DGM,HGM,V,start=start2))
+         if (class(result)=="try-error") result<-rep(NA,3)
+         result
+         }
>
> N<-473.41; DGM<-28.69; HGM<-20.60; V<-248.60
> plotArea<-pi*0.09^2
>
> d<-c(6.4,16.2,19.2,19.7,20.0,20.1,20.2,20.3,20.8,20.8,
+      21.5,21.6,23.0,25.6,25.9,29.6,30.6,31.1,35.6)
> h<-c(6.1,17.0,19.1,15.9,18.8,18.9,17.8,17.9,18.0,17.7,
+      18.3,17.8,18.4,19.8,20.8,20.3,20.0,19.7,20.2)
>
> Nt<-length(d)/plotArea
> cs<-cumsum(d^2)
> DGMt<-min(d[cs>max(cs)/2])
> HGMt<-h[d==min(d[cs>max(cs)/2]) ]
> Vt<-sum(predvol(1,d,h,model=2))/plotArea/1000
>
> recov(N,DGM,HGM,V)
Error in solve.default(grad, -value) :
  Lapack routine dgesv: system is exactly singular: U[1,1] = 0
> recov2(N,DGM,HGM,V)
[1]  4.863375 28.154893  3.073311
attr(,"value")
[1] 0.0001629323
> recov3(N,DGM,HGM,V)
[1]  4.871530 28.161714  3.072694
attr(,"value")
[1] 4.017425e-07
> recov3(Nt,DGMt,HGMt,Vt)
[1]  4.064351 24.603225  3.046871
attr(,"value")
[1] 5.320013e-07
```

Example 11.33. The calibration procedure of Mehtätalo (2005b) is illustrated by the percentile-based models published by Kangas and Maltamo (2000) for Norway spruce stands. The model is a seemingly unrelated system of linear models for 11 logarithmic percentiles of the basal-area weighted diameter distribution. Because the distribution is basal-area weighted, the observed quantiles also need to have the same weighting, i.e. they should be based on angle count sample plots, not on fixed area plots. The known basal-area weighted median diameter is used directly as the 50^{th} percentile. The model is described by percentiles that correspond to the following values of the basal-area weighted *cdf*:

```
> F<-c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,0.95,1)
```

For each percentile, the fixed part of the model is of form

$$\mathrm{E}(\ln \xi_i^{(j)}) = \beta_1 + \beta_2 \ln(T_i/G_i) + \beta_3 \ln G_i + \beta_4 \ln DGM_i + \beta_5 \ln T_i \,,$$

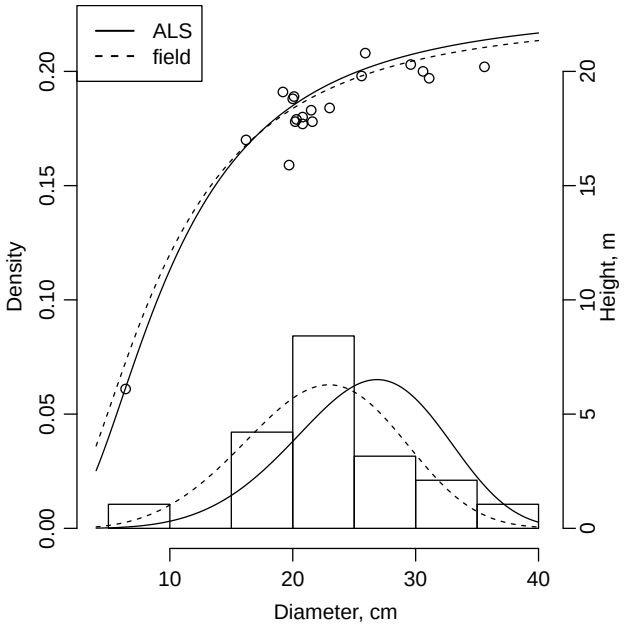


FIGURE 11.13 The histogram of field-measured diameters and observed heights, and the recovered Weibull *pdf* and H-D curve using ALS based predictions (solid) and field estimates (dashed) of N, V, DGM, and HGM in Web example 11.31.

where G_i, T_i and DGM_i are the basal area (m^2/ha), age (years) and basal-area median diameter (cm) of the target stand. The parameter estimates and the variance-covariance matrix of residual errors are in the objects `beta` and `D`.

```
> round(t(beta),3)
       [,1]   [,2]   [,3]   [,4]   [,5] [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12]
[1,] -0.356 -0.212 -0.167 -0.320 -0.132    0 0.177 0.324 0.477 0.777 0.900 1.382
[2,] -0.118 -0.074 -0.079 -0.038 -0.031    0 0.000 0.000 0.000 0.000 0.000 0.000
[3,] -0.126  0.000  0.000  0.000  0.000    0 0.000 0.000 0.000 0.000 0.000 0.000
[4,]  0.835  0.883  0.968  1.053  1.027    1 0.969 0.896 0.838 0.750 0.702 0.624
[5,]  0.000  0.000  0.000  0.000  0.000    0 0.000 0.035 0.060 0.079 0.101 0.083
> round(D,3)
        [,1]   [,2]   [,3]   [,4]   [,5] [,6]   [,7]   [,8]   [,9]  [,10]  [,11]  [,12]
 [1,]  0.162  0.050  0.022  0.011  0.007    0  0.000 -0.003 -0.005 -0.007 -0.007 -0.010
 [2,]  0.050  0.075  0.035  0.016  0.009    0 -0.003 -0.005 -0.007 -0.008 -0.008 -0.009
 [3,]  0.022  0.035  0.029  0.015  0.009    0 -0.003 -0.004 -0.006 -0.007 -0.007 -0.008
 [4,]  0.011  0.016  0.015  0.014  0.009    0 -0.002 -0.002 -0.004 -0.004 -0.005 -0.005
 [5,]  0.007  0.009  0.009  0.009  0.010    0 -0.001 -0.001 -0.002 -0.003 -0.004 -0.004
 [6,]  0.000  0.000  0.000  0.000  0.000    0  0.000  0.000  0.000  0.000  0.000  0.000
 [7,]  0.000 -0.003 -0.003 -0.002 -0.001    0  0.003  0.003  0.003  0.003  0.003  0.003
 [8,] -0.003 -0.005 -0.004 -0.002 -0.001    0  0.003  0.006  0.006  0.006  0.006  0.006
 [9,] -0.005 -0.007 -0.006 -0.004 -0.002    0  0.003  0.006  0.009  0.008  0.008  0.009
[10,] -0.007 -0.008 -0.007 -0.004 -0.003    0  0.003  0.006  0.008  0.011  0.011  0.011
[11,] -0.007 -0.008 -0.007 -0.005 -0.004    0  0.003  0.006  0.008  0.011  0.014  0.014
[12,] -0.010 -0.009 -0.008 -0.005 -0.004    0  0.003  0.006  0.009  0.011  0.014  0.025
```

Consider a forest stand with the following values of the predictors: $DGM = 20$, 60 and $G = 20$. The resulting predicted logarithmic percentiles are computed below to vector `xi`. The resulting percentile-based distribution of logarithmic diameters is illustrated in the top-left corner of Figure 11.14.

```
> DGM<-20;T<-60;G<-20
> x<-cbind(1,log(T/G),log(G),log(DGM),log(T))
> t(round(xi<-t(pred<-x%*%beta),3))
      [,1]  [,2]  [,3]  [,4] [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12]
[1,] 1.638 2.352 2.646 2.792 2.91 2.996 3.079 3.151 3.234 3.349 3.417 3.593
```

Assume then that a total of five trees were measured for diameter on two angle-count sample plots with $n_1 = 7$ and $n_2 = 9$ trees. On the first plot, the observed diameters were 10 cm for the smallest tree, 11 cm for the third smallest tree and, 27 cm for the second largest tree. On the second plot, the smallest and second smallest tree had the diameters 8 and 12 cm. Recalling that the models treat diameters in logarithmic scale, we had therefore the following observations $Y_{1:7} = \ln(10)$, $Y_{3:7} = \ln(11)$, and $Y_{6:7} = \ln(27)$. On the second plot, $Y_{1:9} = \ln(8)$ and $Y_{2:9} = \ln(12)$ were observed. The expected values of these quantiles are the expected values of the distribution presented in Equation 11.24 on p. 374 of ML2020. We get become

$$\boldsymbol{\mu}^* = (2.27, 2.83, 3.21, 2.19, 2.53)'$$

implying

$$\boldsymbol{p}^* = (0.089, 0.338, 0.769, 0.077, 0.161)'.$$

The sampling error associated with each quantile is the variance of the above-mentioned *pdf* 11.24, and the covariance between two observations from the same plot can be found based on the joint density 11.25 on p. 374 of ML2020. These computations have been implemented in function `lmfor::qtree.varcov`, which further calls functions `lmfor::qtree.moments` and `lmfor::qtree.exy`. The marginal distributions of $Y_{1:7}$ and $Y_{3:7}$ and their joint distribution are illustrated in the top-right and middle panel of Figure 11.14; the joint density was computed using `lmfor::qtree.jointdens`. Because we had trees from two plots, the variance-covariance matrix of sampling errors, var($\boldsymbol{\epsilon}$) (element `R` below), includes two blocks. The variances on the diagonal are the smaller the larger the number of trees on the plot is, e.g. $\text{var}(Y_{1:7}) = 0.14 > \text{var}(Y_{1:9}) = 0.13$. Within an individual sample plot, successive quantiles are more strongly correlated than the quantiles that are far from each other. For example, $\text{cor}(Y_{1:9}, Y_{2:9}) = 0.61$ on plot 2, whereas $\text{cor}(Y_{1:7}, Y_{6:7}) = 0.20$ on plot 1.

```
> obs<-data.frame(r=c(1,3,6,1,2),n=c(7,7,7,9,9),plot=c(1,1,1,2,2),d=c(10,11,27,8,12))
> print(qtrees<-qtree.varcov(obs,xi,F))
$obs
  r n plot  d       Ed        pEd
1 1 7    1 10 2.271087 0.08860375
2 3 7    1 11 2.836980 0.33807139
3 6 7    1 27 3.208331 0.76861726
4 1 9    2  8 2.189040 0.07711348
5 2 9    2 12 2.532421 0.16127920

$R
           [,1]       [,2]       [,3]       [,4]       [,5]
[1,] 0.14301084 0.04049206 0.01012148 0.00000000 0.00000000
[2,] 0.04049206 0.04528762 0.01263340 0.00000000 0.00000000
[3,] 0.01012148 0.01263340 0.01845891 0.00000000 0.00000000
[4,] 0.00000000 0.00000000 0.00000000 0.12807488 0.06553978
[5,] 0.00000000 0.00000000 0.00000000 0.06553978 0.08950899

> cov2cor(qtrees$R)
          [,1]      [,2]      [,3]      [,4]      [,5]
[1,] 1.0000000 0.5031478 0.1969957 0.0000000 0.0000000
[2,] 0.5031478 1.0000000 0.4369459 0.0000000 0.0000000
[3,] 0.1969957 0.4369459 1.0000000 0.0000000 0.0000000
[4,] 0.0000000 0.0000000 0.0000000 1.0000000 0.6121243
[5,] 0.0000000 0.0000000 0.0000000 0.6121243 1.0000000
```

The calibration procedure has been implemented below. Matrix $\boldsymbol{D}^*$ is a 5 by 5 square variance-covariance matrix of stand effects at the values $\boldsymbol{p}^*$ of the observed percentiles. Matrix $\boldsymbol{C} = \text{cov}(\boldsymbol{e}, \boldsymbol{e}^{*\prime})$ is a 5 by 12 matrix. These matrices are obtained from $\boldsymbol{D}$ through linear interpolation using `lmfor::interpolate.D`.

```
> obs<-qtrees$obs
> mustar<-obs$Ed
> ystar<-log(obs$d)
> R<-qtrees$R
> Dtayd<-interpolate.D(D,obs$pEd)
> round(Dstar<-Dtayd$D1,4)
       [,1]    [,2]    [,3]    [,4]    [,5]
[1,]  0.0845  0.0126 -0.0064  0.0817  0.0483
```

```
[2,]  0.0126  0.0125 -0.0028  0.0122  0.0129
[3,] -0.0064 -0.0028  0.0078 -0.0061 -0.0058
[4,]  0.0817  0.0122 -0.0061  0.0945  0.0464
[5,]  0.0483  0.0129 -0.0058  0.0464  0.0468
> round(C<-Dtayd$D2,4)
        [,1]    [,2]    [,3]    [,4]    [,5]
 [1,]  0.0628  0.0093 -0.0046  0.0756  0.0331
 [2,]  0.0718  0.0130 -0.0066  0.0690  0.0503
 [3,]  0.0335  0.0128 -0.0053  0.0320  0.0315
 [4,]  0.0151  0.0123 -0.0032  0.0145  0.0152
 [5,]  0.0086  0.0095 -0.0021  0.0083  0.0091
 [6,]  0.0000  0.0000  0.0000  0.0000  0.0000
 [7,] -0.0024 -0.0013  0.0030 -0.0020 -0.0027
 [8,] -0.0048 -0.0020  0.0059 -0.0046 -0.0044
 [9,] -0.0071 -0.0032  0.0078 -0.0069 -0.0065
[10,] -0.0080 -0.0040  0.0075 -0.0078 -0.0072
[11,] -0.0080 -0.0043  0.0076 -0.0078 -0.0075
[12,] -0.0091 -0.0047  0.0079 -0.0092 -0.0082
```

The predicted plot effects are computed below to vector `ehat`, and calibrated percentiles to `xi1`. The variance-covariance matrix of calibrated 0^{th}, 30^{th}, 70^{th} and 100^{th} percentiles are shown below together with corresponding elements from matrix $\boldsymbol{D}$. The differences between these variances quantify the gain obtained from the sample information of the quantile trees. The original and calibrated *cdf* and *pdf*'s are shown in the bottom panel of Figure 11.14; linear interpolation is used in the illustrations also in the back-transformed scale even though the calibration procedure is based on a percentile-based distribution with linear interpolation in the logarithmic scale. The calibrated distribution has clearly a bimodal shape, even though the original PPM models led to (practically) unimodal shape.

```
> ehat<-C%*%solve(Dstar+R)%*%(ystar-mustar)
> t(round(xi1<-xi+ehat,3))
      [,1]  [,2]  [,3] [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12]
[1,] 1.622 2.331 2.552 2.67 2.809 2.996 3.108 3.204 3.307 3.428   3.5  3.68
> D[c(1,4,8,12),c(1,4,8,12)]
             [,1]         [,2]         [,3]         [,4]
[1,]  0.161652909  0.010707222 -0.002739361 -0.009819052
[2,]  0.010707222  0.014182608 -0.002448765 -0.005410252
[3,] -0.002739361 -0.002448765  0.005924280  0.006247869
[4,] -0.009819052 -0.005410252  0.006247869  0.024965960
> (varxi1<-D-C%*%solve(Dstar+R)%*%t(C))[c(1,4,8,12),c(1,4,8,12)]
              [,1]          [,2]          [,3]         [,4]
[1,]  1.285930e-01  0.0046329856 -5.332619e-05 -0.005027418
[2,]  4.632986e-03  0.0095018613 -5.153176e-04 -0.002148230
[3,] -5.332619e-05 -0.0005153176  4.248145e-03  0.003808720
[4,] -5.027418e-03 -0.0021482303  3.808720e-03  0.021328241
```

In practice, it would be justified to iterate the procedure because the improved stand-level diameter distribution will lead to different values of $\boldsymbol{\mu}^*$ and $\boldsymbol{p}^*$; this is left as an exercise. Mehtätalo and Kangas (2005) developed the algorithm further to take into account the measurement errors of stand characteristics. Mehtätalo *et al.* (2006) analyzed the optimal choice of percentiles for measurements.

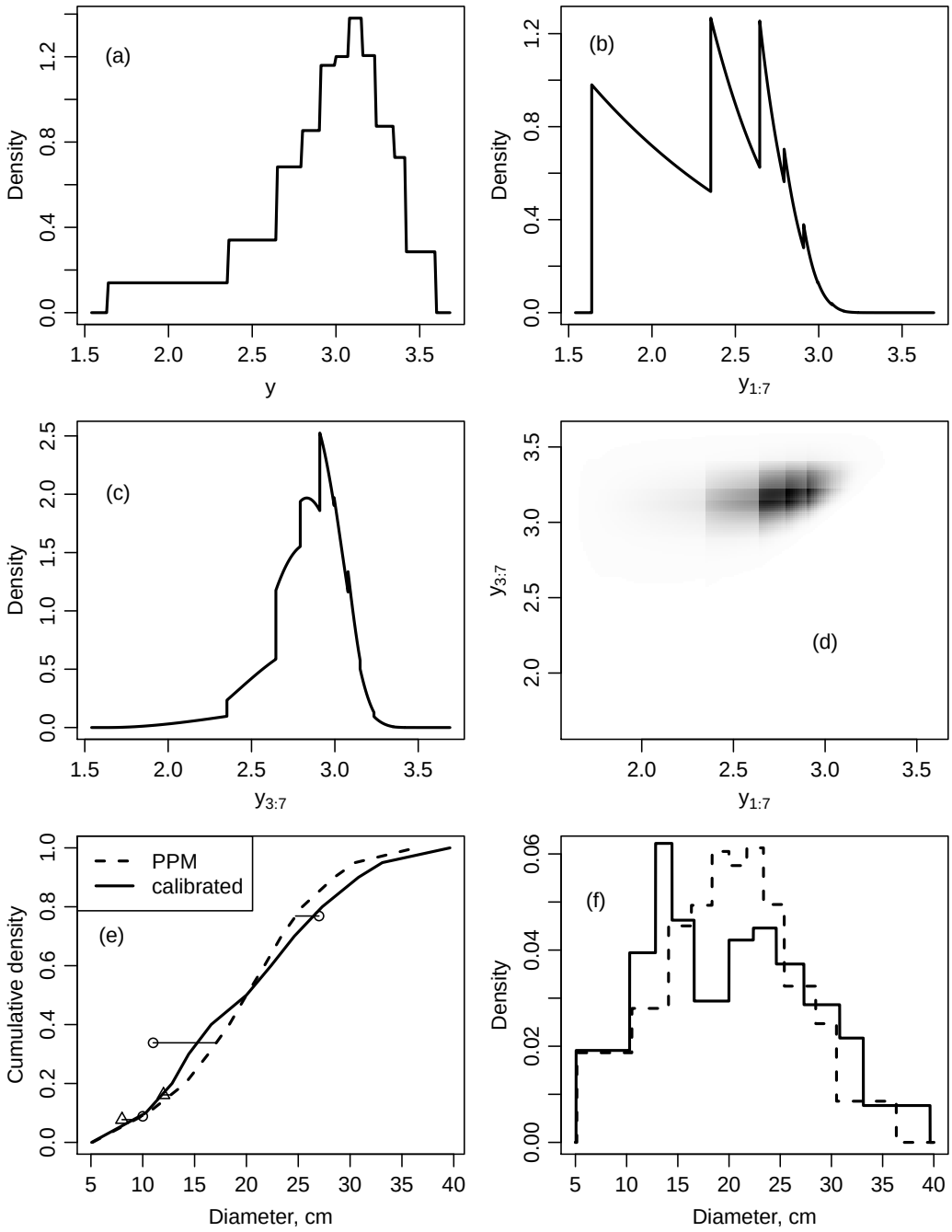


FIGURE 11.14 The percentile-based diameter distribution of logarithmic diameters, the corresponding marginal *pdf*'s of the minimum $y_{1:7}$ and third smallest trees $y_{3:7}$ in a sample of size 7, and the joint *pdf* of $y_{1:7}$ and $y_{3:7}$. The bottom graphs illustrate the percentile-based diameter distribution using back-transformed diameters based on the PPM model (solid) and the distribution calibrated by using the five quantile trees from two sample plots (dashed). The observed quantile trees are illustrated by the dots in the bottom left graph, with plot-specific symbols.

Bibliography

Chiu, S.N., Stoyan, D., Kendall, W.S., and Mecke, J., 2013. *Stochastic Geometry and Its Applications*, New York, USA: Wiley, 3rd ed.

Harrell, F.J., 2001. *Regression Modeling Strategies with Applications to Linear Models, Logistic Regression, and Survival Analysis*, New York, USA: Springer, 562 p.

Kangas, A. and Maltamo, M., 2000. Percentile-based basal area diameter distribution models for Scots pine, Norway spruce and birch species, *Silva Fennica*, 34 (4), 371–380.

Kansanen, K., Vauhkonen, J., Lähivaara, T., and Mehtätalo, L., 2016. Stand density estimators based on individual tree detection and stochastic geometry, *Canadian Journal of Forest Research*, 46 (11), 1359–1366.

Laasasenaho, J., 1982. Taper curve and volume functions for pine, spruce and birch, *Comm. Inst. For. Fenn.*, 108.

Lindstrom, M.J. and Bates, D.M., 1990. Nonlinear mixed effects models for repeated measures data, *Biometrics*, 46 (3), 673–687.

Mehtätalo, L., 2005a. Height-diameter models for scots pine and birch in Finland, *Silva Fennica*, 39 (1), 55–66.

Mehtätalo, L., 2005b. Localizing a predicted diameter distribution using sample information, *Forest Science*, 51 (4), 292–302.

Mehtätalo, L., 2006. Eliminating the effect of overlapping crowns from aerial inventory estimates, *Canadian Journal of Forest Research*, 36 (7), 1649–1660.

Mehtätalo, L. and Kangas, A., 2005. An approach to optimizing data collection in an inventory by compartments, *Canadian Journal of Forest Research*, 35 (1), 100–112.

Mehtätalo, L. and Lappi, J., 2020. *Biometry for Forestry and Environmental Data: with Examples in R*, Boca Raton: Chapman Hall / CRC Press, 1st ed., 411 p.

Mehtätalo, L., Maltamo, M., and Kangas, A., 2006. The use of quantile trees in predicting the diameter distribution of a stand, *Silva Fennica*, 40 (3), 501–516.

Mehtätalo, L., Maltamo, M., and Packalén, P., 2007. Recovering plot-specific diameter distribution and height-diameter curve using als-based stand characteristics, *IAPRSS*, 36/ Part 3/W52, 288–293.

Mehtätalo, L., Nyblom, J., and Virolainen, A., 2014a. A model-based approach for the recovery of forest attributes using als data., *in:* M. Maltamo, E. Næsset, and J. Vauhkonen, eds., *Forestry applications of airborne laser scanning- Concepts and Case Studies*, Springer, 193–211.

Mehtätalo, L., Peltola, H., Kilpeläinen, A., and Ikonen, V.P., 2014b. The response of basal area growth of scots pine to thinning: A longitudinal analysis of tree-specific series using a nonlinear mixed-effects model, *Forest Science*, 60 (4), 636 – 644.