Mixed-effects models in R

Andrew Robinson

Department of Mathematics & Statistics
University of Melbourne

Outline

Data - Height/Diameter from Stage (1963)

A brief synopsis:

- 66 trees, purposively selected in
- 9 national forests around northern and central Idaho, representing
- 5 habitat types.
- Trees split: decadal measurements of
  - height
  - dbhib

Scatterplot

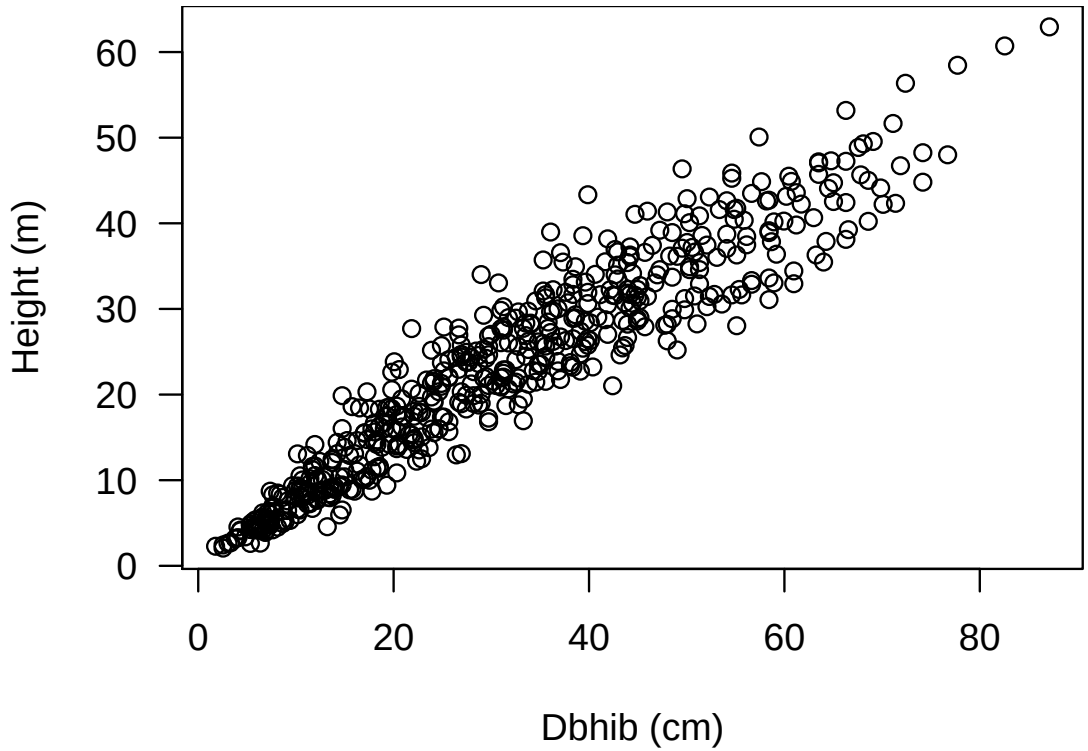
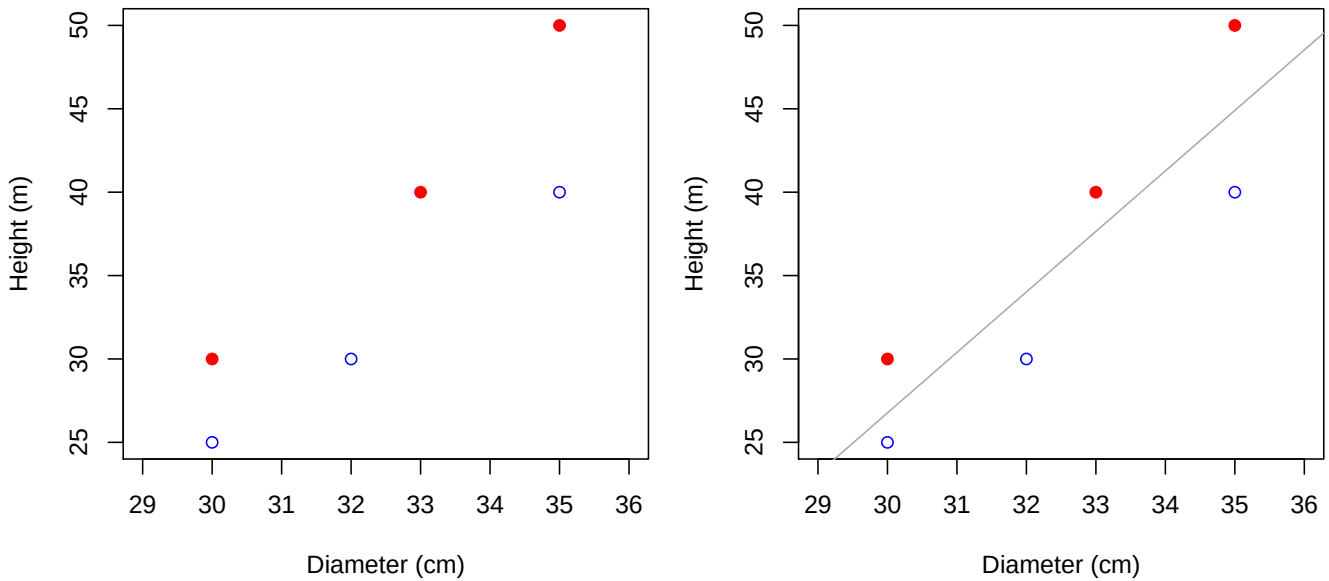


Figure: Al Stage's Grand Fir stem analysis data: height (m) against diameter inside bark (cm). These were dominant and co-dominant trees.

A simpler perspective

Construct a height-diameter relationship using two randomly selected trees in a forest, given that we have measured each three times.

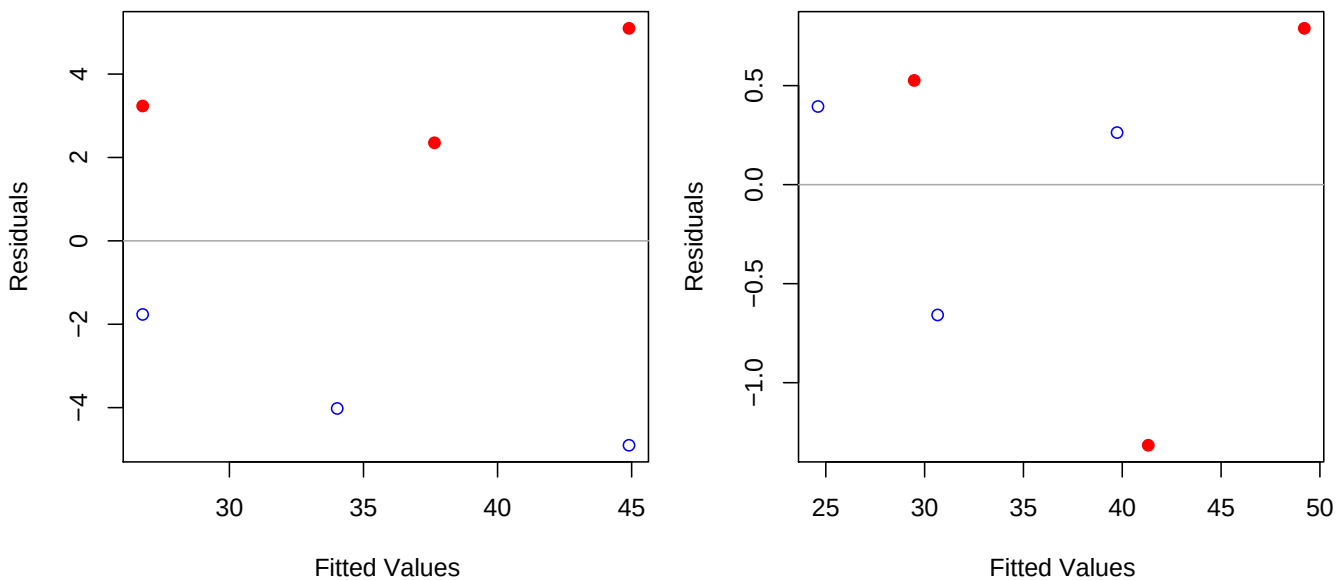
Growing conditions are quite different for the trees, leading to a systematic difference between the height-diameter relationships.



A simpler perspective

If we fit a simple regression to the trees then we obtain a flawed residual/fitted value plot (left).

If we fit a simple regression to the trees with an intercept for each plot then we obtain a reasonable residual/fitted value plot (right).



Dilemma! Do we use a useless good model or a useful bad model?

Model for getting it wrong in R

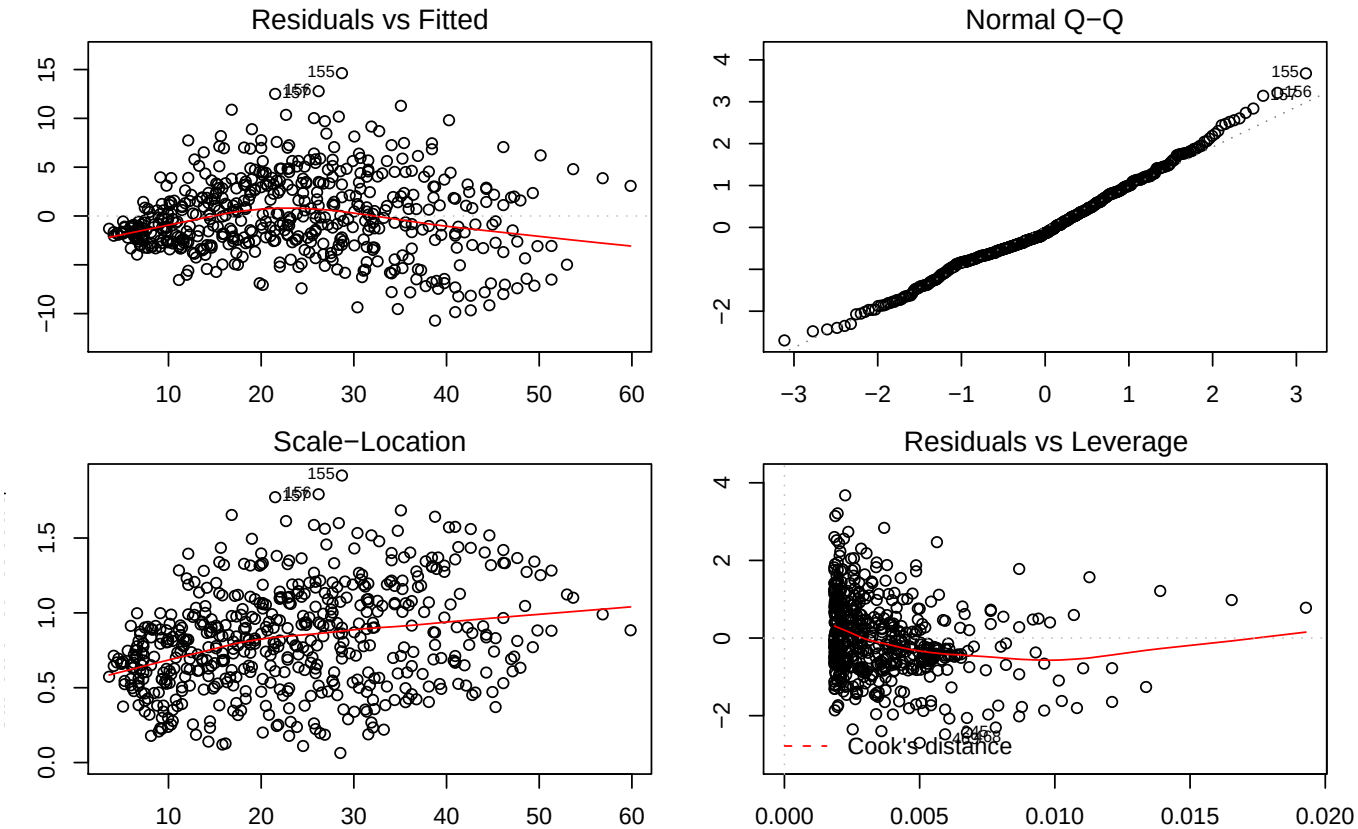
$$h_i = \beta_0 + \beta_1 \times d_i + \epsilon_i \tag{1}$$

Regression assumptions.

- True relationship is linear.
- $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- ϵ_i independent.

Diagnostics for getting it wrong in R

```
> hd.lm <- lm(height.m ~ dbhib.cm, data = stage)
> plot(hd.lm)
```



Decomposition 1

Note that the model specification implies that:

$$\hat{\epsilon}_{ij} = y_{ij} - \hat{y}_{ij} \tag{2}$$

and

- The true relationship is linear.
- $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- The ϵ_i are independent.

Clearly not true. This is our problem.

Decomposition 2

What if we could make:

$$y_{ij} - \hat{y}_{ij} = \hat{b}_i + \hat{\epsilon}_{ij} \tag{3}$$

So

$$\hat{\epsilon}_{ij} = y_{ij} - \hat{y}_{ij} - \hat{b}_i \tag{4}$$

Then we merely need to assume that:

- The true relationship is linear.
- $b_i \sim \mathcal{N}(0, \sigma_b^2)$
- $\epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$
- The residuals ϵ_{ij} are independent.

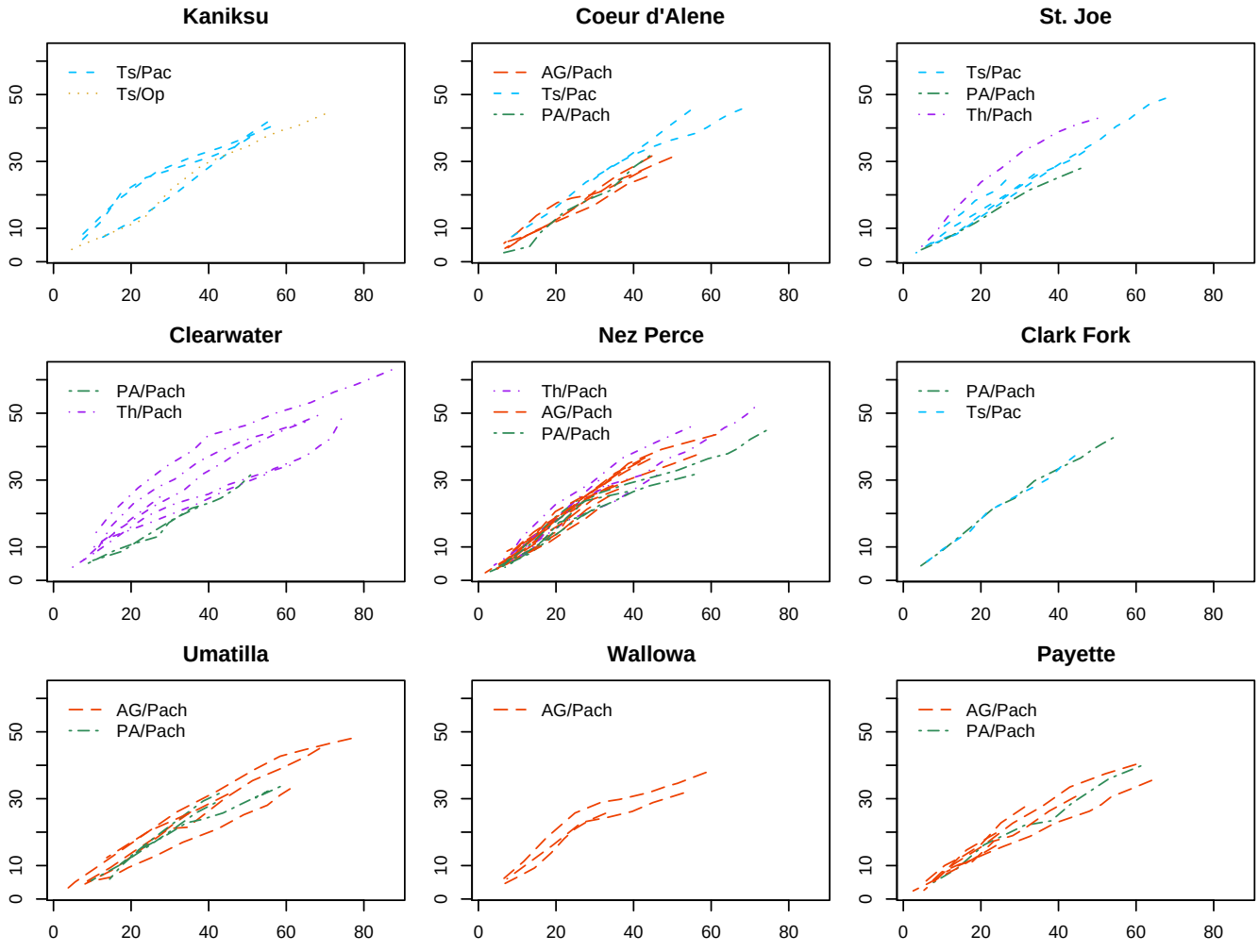
Much more tenable!

Decomposition 3

The assumptions are satisfied because the systematic differences between the plots, which previously produced correlation, are now accounted for by the new random effects.

However, when the time comes to use the model for prediction, we do not need to know the plot identity, as the fixed effects do not require it.

Another look



Model for getting it less wrong in R

$$h_{it} = \beta_0 + b_{0i} + \beta_1 \times d_{it} + \epsilon_{it} \tag{5}$$

Regression assumptions.

- True relationship is linear.
- $b_{0i} \sim \mathcal{N}(0, \sigma^2_{b_0})$, independent.
- $\epsilon_{it} \sim \mathcal{N}(0, \sigma^2)$, independent.

Temporary Vocabulary.

- ϵ_{it} are the innermost (level $k = 1$) residuals.
- $b_{0i} + \epsilon_{it}$ are the outermost (level 0) residuals.

(General Model Statement)

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Zb} + \boldsymbol{\epsilon}$$

$$\begin{aligned} \mathbf{b} &\sim \mathcal{N}(\mathbf{0}, \mathbf{D}) \\ \boldsymbol{\epsilon} &\sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \end{aligned}$$

Design Matrices

- $\mathbf{X}$ allocates the fixed effects.
- $\mathbf{Z}$ allocates the random effects.

Covariance Matrices

- $\mathbf{D}$ describes the random effects covariance.
- $\mathbf{R}$ describes the residuals covariance.

Assumptions for getting it less wrong in R

Now, the key assumptions that we're making are that:

1. the model structure is correctly specified
2. the tree random effects are normally distributed,
3. the tree random effects are homoskedastic.
4. the innermost residuals are normally distributed,
5. the innermost residuals are homoskedastic within and across the tree random effects.
6. the innermost residuals are independent within the groups.

Software

Two primary tools to fit this model:

- `lme (nlme)`
  - Stable
  - Well-documented
  - Many helper functions
- `lmer (lme4)`
  - Crossed random effects
  - Optimized for large datasets
  - MCMC for inference
  - GLMM

although numerous other tools will do it too.

Fitting a model

Popular arguments.

```
lme(fixed,                  # y ~ the fixed effects
    random,                 # ~ covariates | groups
    data,                   # name the dataframe
    na.action,              # what to do with missing values
    subset,                 # including only some of the data
    weights,                # heteroskedasticity within groups
    correlation,            # correlation within groups
    method,                 # ReML or ML
    control,                # underneath the bonnet
    keep.data = TRUE)
```

Fit.

```
> require(nlme)
> stage.g <- groupedData(height.m ~ dbhib.cm | Tree.ID,
+                        data = stage)
> hd.lme.1 <- lme(height.m ~ dbhib.cm,
+                 random = ~ 1 | Tree.ID,
+                 data = stage.g)
```

```
> require(lme4)
> hd.lmer.1 <- lmer(height.m ~ dbhib.cm + (1|Tree.ID),
+                   data = stage)
```

Helper functions

- `lme()`

```
> resid(hd.lme.1, type = "p", level = 1)
> fitted(hd.lme.1, level = 0)
> ranef(hd.lme.1, standard = FALSE)
```

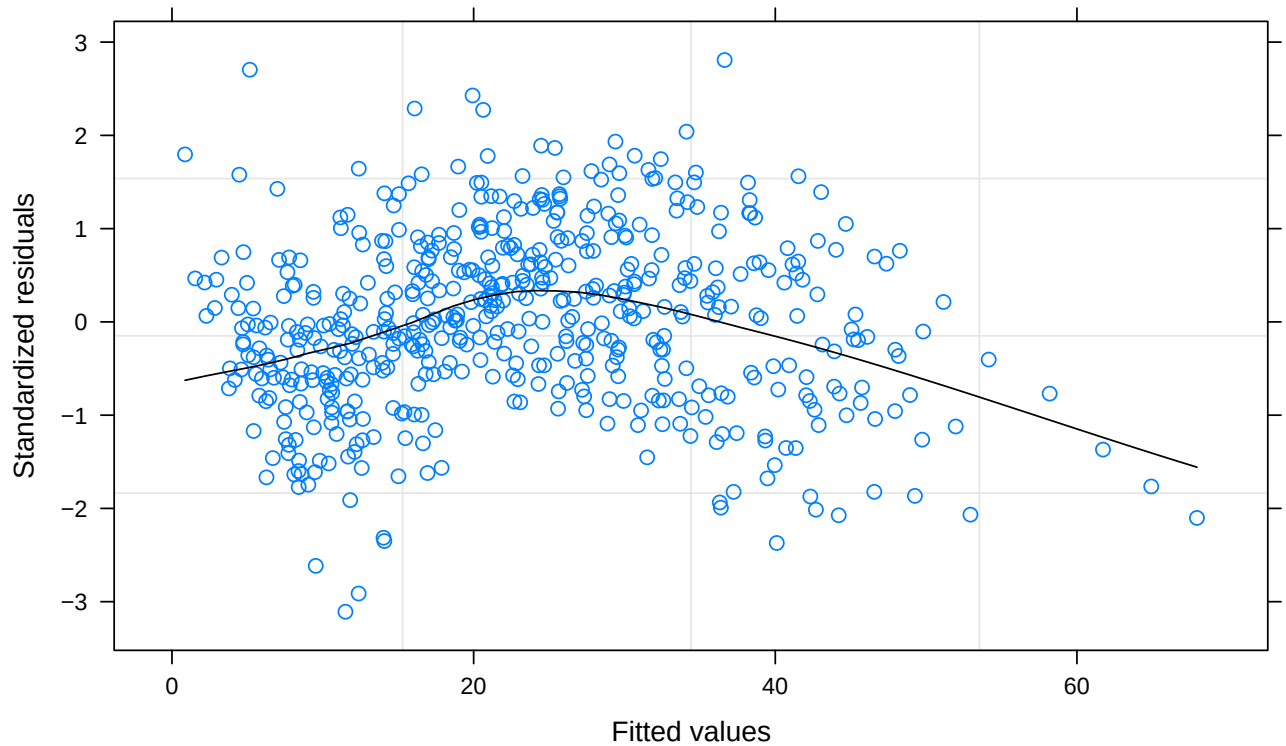
- `lmer()`

```
> resid(hd.lmer.1)     # level = k, not standardized
> fitted(hd.lmer.1)    # level = k
> ranef(hd.lmer.1)     # Not standardized
```

lme() Linearity, constant variance

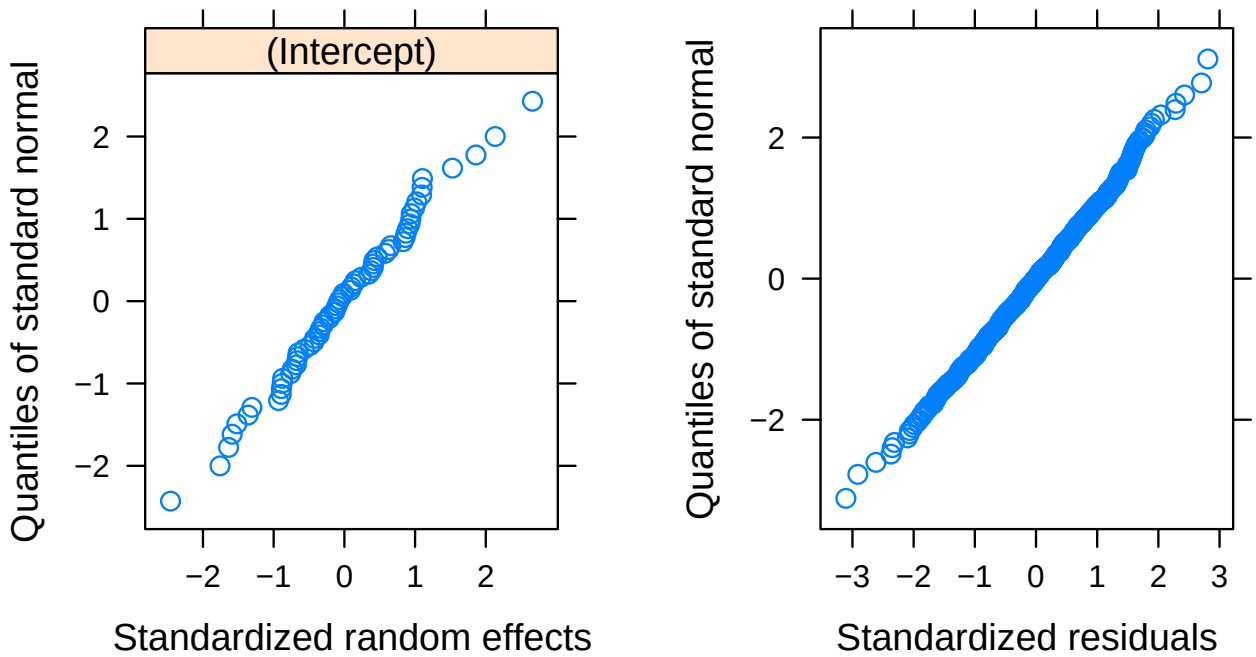
```
> trellis.par.set(plot.line = list(col = "black"))

> plot(hd.lme.1,
+      resid(., type = "p") ~ fitted(.),
+      type = c("p", "smooth"))
```



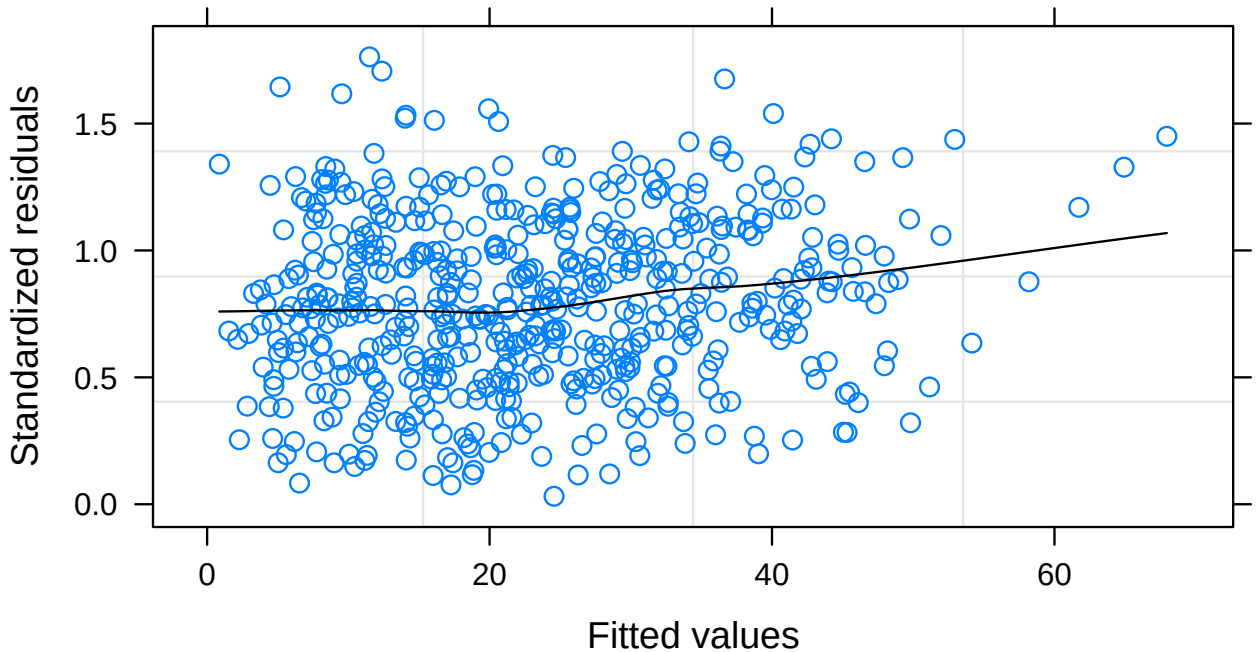
lme() Normality (and homoskedasticity)

```
> qqnorm(hd.lme.1, ~ ranef(., standard = TRUE))
> qqnorm(hd.lme.1, ~ resid(., type = "p"))
```



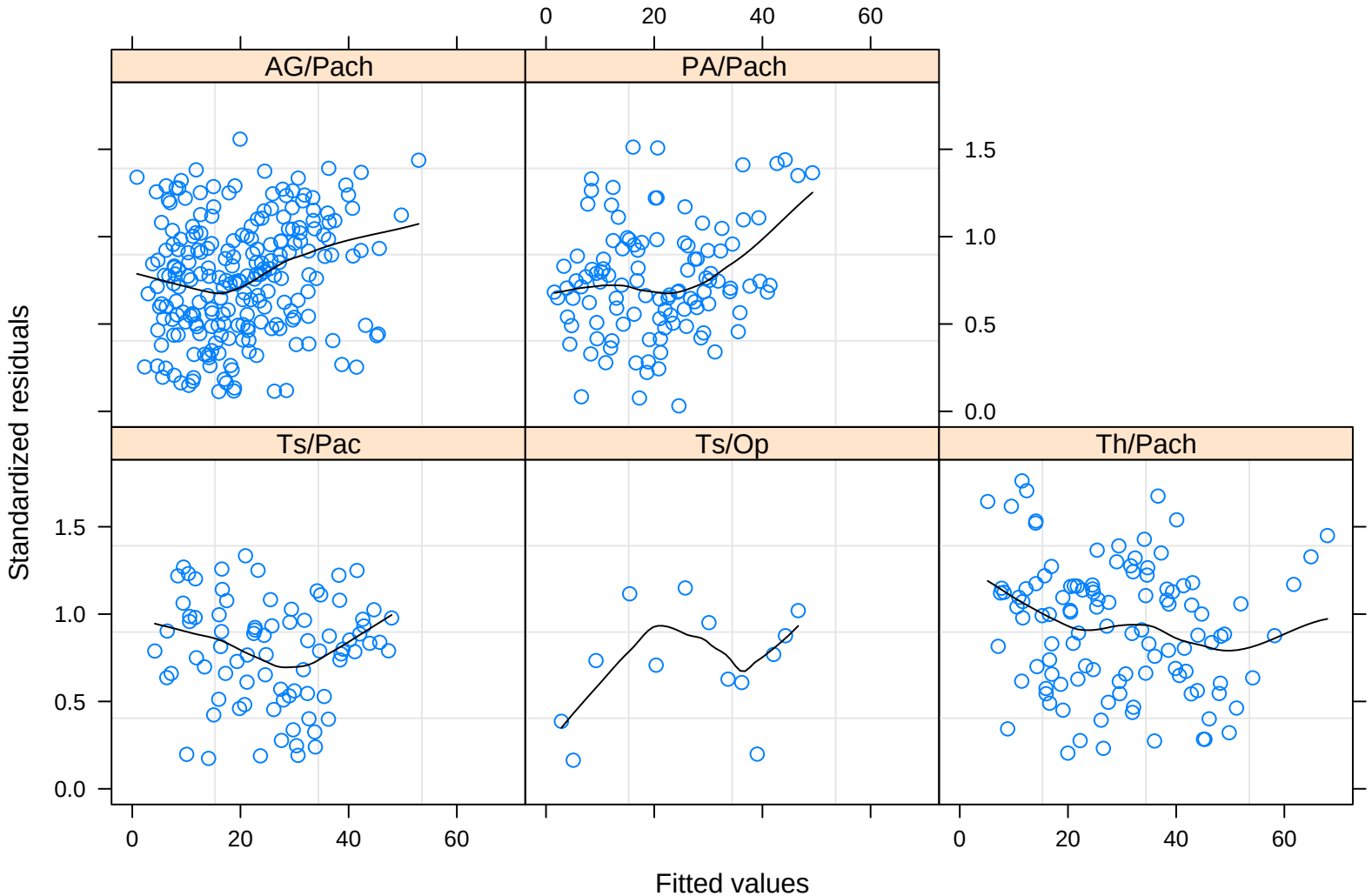
lme() Constant Variance.

```
> plot(hd.lme.1,
+      sqrt(abs(resid(., type = "p"))) ~ fitted(.),
+      type = c("p", "smooth"))
```



lme() Diagnosing Variance Structures

```
> plot(hd.lme.1,
+      sqrt(abs(resid(., type = "p"))) ~ fitted(.) | HabType.ID,
+      type = c("p", "smooth"))
```



`lme()` Within-subject heteroskedasticity

In the one-level case, we model as follows:

$$y_{ij} = \beta_0 + b_i + \beta_1 \times x_{ij} + \epsilon_{ij} \tag{6}$$

Regression assumptions.

- True relationship is linear.
- $b_i \sim$ i.i.d $\mathcal{N}(0, \sigma_b^2)$
- $\epsilon_{ij} \sim$ i.i.d $\mathcal{N}(0, \text{Var}(\epsilon_{ij}|b_i))$
- $\text{Var}(\epsilon_{ij}|b_i) = \sigma^2 g^2(\mu_{ij}, \mathbf{v_{ij}}, \boldsymbol{\delta})$

`lme()` Variance Structures: `weights =`

Arguments:

- `value =` $\boldsymbol{\delta_0}$, `form =` $\sim$ `cov | stratum`, `fixed =` $\boldsymbol{\delta}$.

Function types:

- `varFixed`: known variance relation (still estimate σ^2).
- `varIdent`: different constant variances within each stratum
- `varPower`: power of nominated covariate(s); default *fitted*.

$$g = |\mathbf{v_{ij}}|^{\delta}$$

- `varExp`: exponential of covariate(s); default *fitted*.

$$g = \exp(\boldsymbol{\delta}\mathbf{v_{ij}})$$

- `varConstPower`: constant plus power; default *fitted*.

$$g = \delta_1 + |\mathbf{v_{ij}}|^{\delta_2}$$

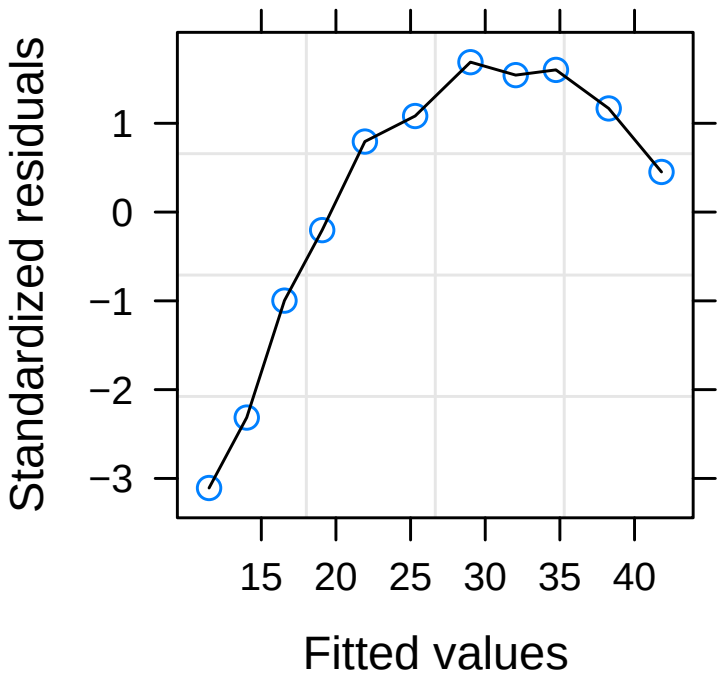
- `varComb`: combination by *product*.

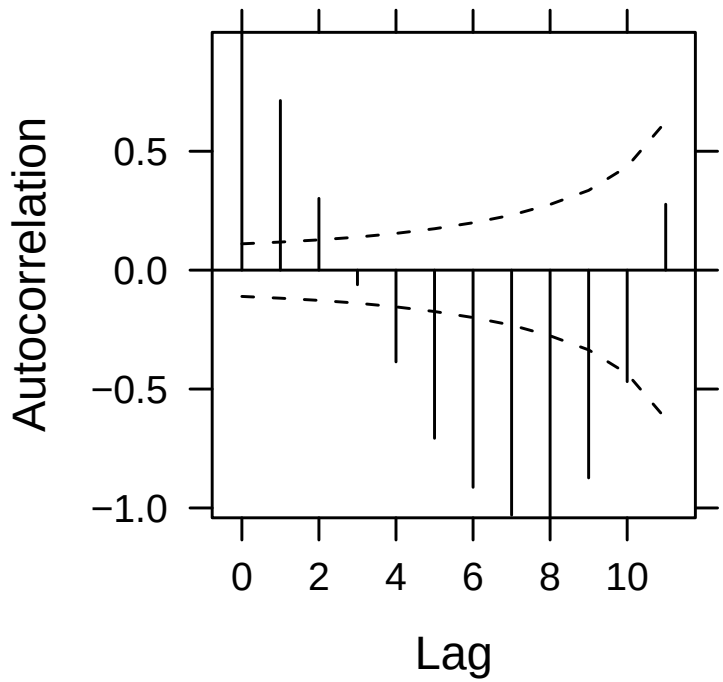
lme() Testing Variance Structures

```
anova(model_1, model_2)
```

lme() Conditional independence

```
> plot(hd.lme.1,
+      resid(., type = "p") ~ fitted(.),
+      subset = stage$Tree.ID=="77", type = "b")
> plot(ACF(hd.lme.1, resType = "n"), alpha = 0.01)
```





`lme()` Modelling within-subject autocorrelations

Spatial or temporal (or both)

- By definition, residuals are no longer independent.
- Correlation of two residuals is modelled as some smooth function of their relative position.
- $corr(\epsilon_{ij}, \epsilon_{ij'}) = h(|\epsilon_{ij}, \epsilon_{ij'}|, \vec{\rho})$

NB: Now we use `resType = "n"` to assess corrections to the model.

`lme()` Temporal Correlation: `correlation =`

Arguments:

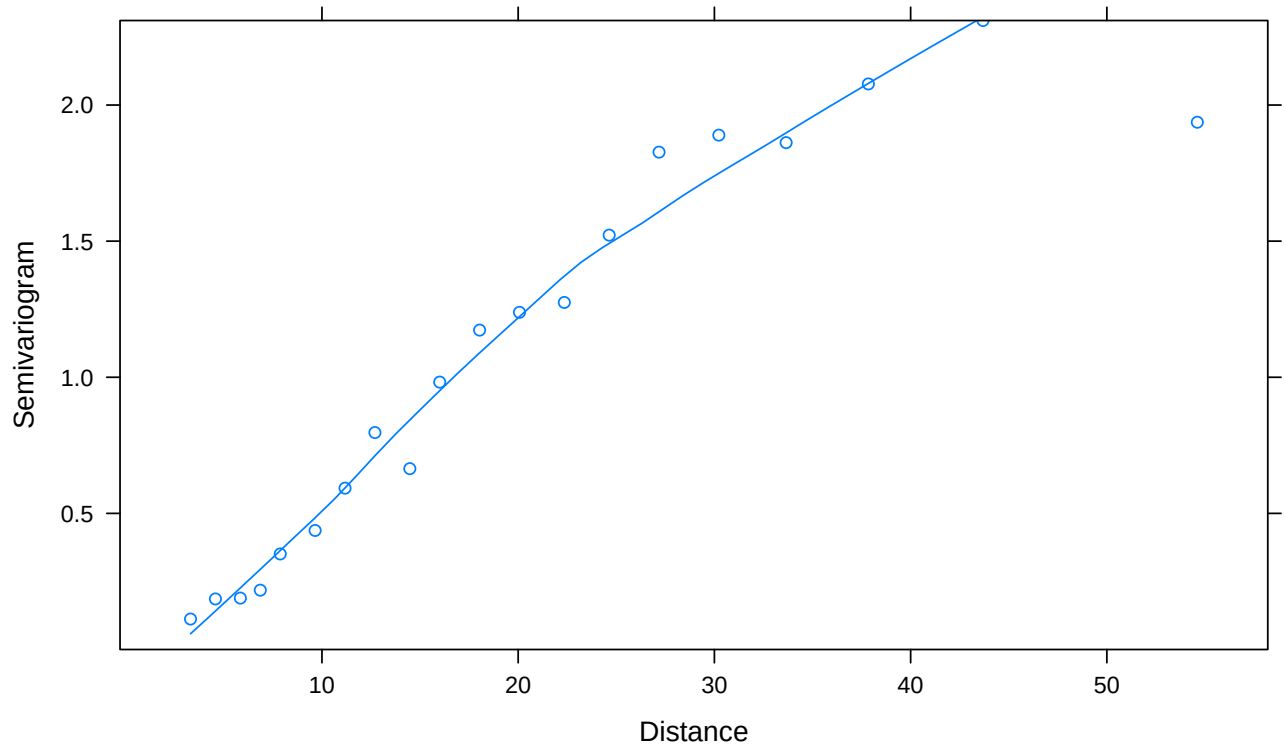
- `value =` $\boldsymbol{\delta_0}$, `form =` $\sim$ `cov | stratum`, `fixed = FALSE`.

Function types:

- `corCompSymm()`: a compound-symmetric matrix; $h(|\epsilon_{ij}, \epsilon_{ij'}|, \rho) = \rho$
- `corSymm()`: a general positive definite matrix; $h(|\epsilon_{ij}, \epsilon_{ij'}|, \rho) = \rho_k$
- `corAR1()`: autoregressive with order 1; $h(|\epsilon_{ij}, \epsilon_{ij'}|, \rho) = \rho^k, k = 0, 1, \ldots$
- `corCAR1()`: continuous autoregressive with order 1; $h(|\epsilon_{ij}, \epsilon_{ij'}|, \rho) = \rho^k, k \geq 0$
- `corARMA(p = a, q = b)`: autoregressive moving average, any order (a, b).

lme() Spatial Correlation Structures

```
> plot(Variogram(hd.lme.1, form = ~dbhib.cm | Tree.ID))
```



`lme()` Spatial Correlation Structures

These functions were intended for 2-d correlation but are more general, and can be used for 1-d correlation as well.

The general form of the semivariogram is:

$$\begin{aligned} \gamma[d(\epsilon_x, \epsilon_y), \lambda] &= \frac{1}{2}\mathsf{Var}(\epsilon_x - \epsilon_y) \\ &= \frac{1}{2}\mathsf{E}(\epsilon_x - \epsilon_y)^2 \end{aligned}$$

`lme()` Spatial Correlation Structures

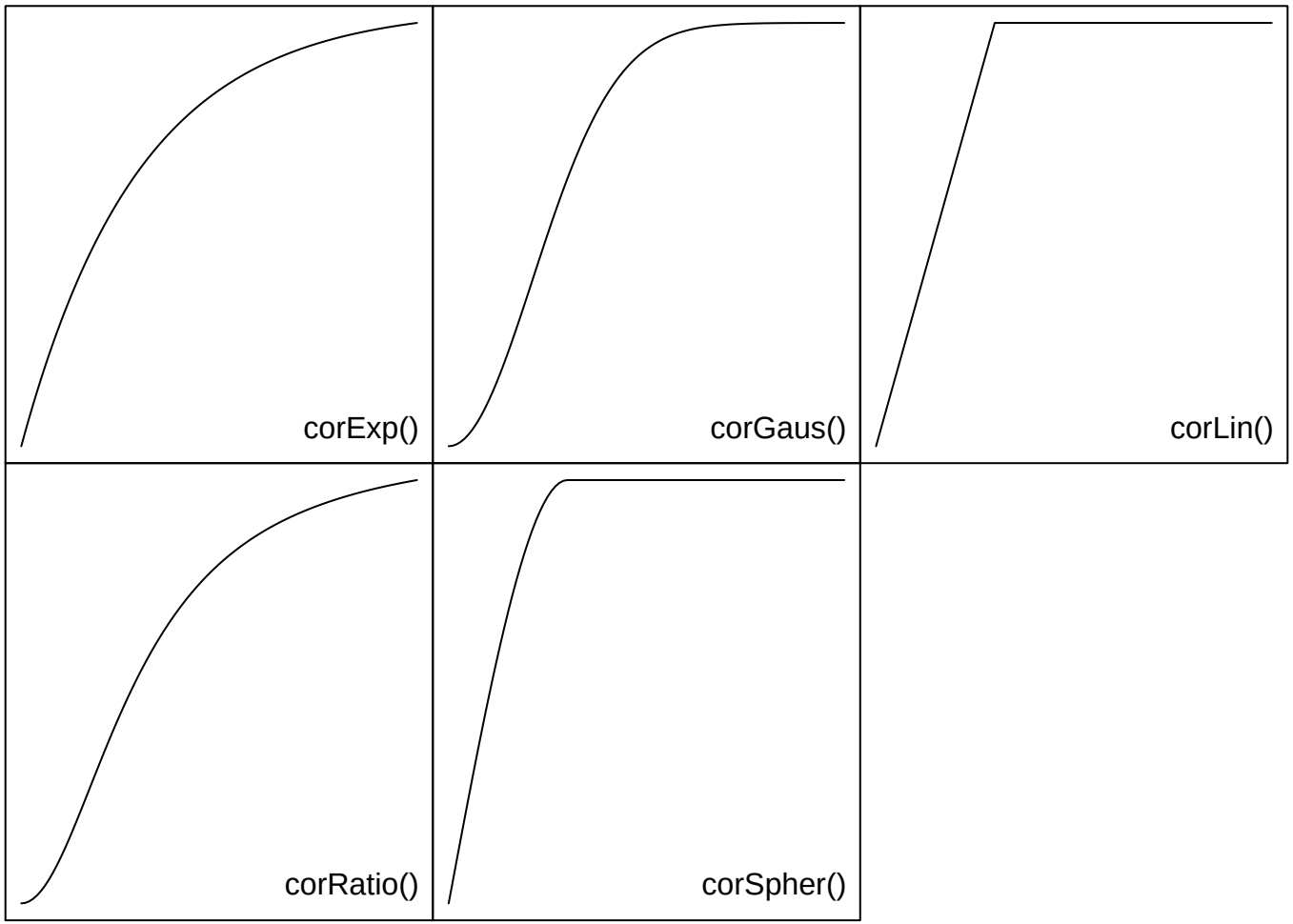
Arguments:

- `value` = $\boldsymbol{\delta_0}$, `form` = $\sim$ `cov | stratum`, `nugget = FALSE`, `metric = "euclidean"`, `fixed = FALSE`.

The supported semivariogram functions are:

1. `corExp`: exponential semivariogram;
   $\gamma(s, \rho) = 1 - \exp(-s/\rho)$
2. `corGaus`: Gaussian semivariogram;
   $\gamma(s, \rho) = 1 - \exp(-(s/\rho)^2)$
3. `corLin`: linear semivariogram;
   $\gamma(s, \rho) = 1 - (1 - s/\rho) \times I(s < \rho)$
4. `corRatio`: rational quadratic semivariogram;
   $\gamma(s, \rho) = (s/\rho)^2/(1 + (s/\rho)^2)$
5. `corSpher`: spherical semivariogram;
   $\gamma(s, \rho) = 1 - (1 - 1.5s/\rho + 0.5(s/\rho)^3) \times I(s < \rho)$

lme() Spatial Correlation Structures



Variations on a Theme from Stage

- `lme()`

```
> hd.lme.2 <- update(hd.lme.1,
+                    random = ~ dbhib.cm | Tree.ID)

> hd.lme.3 <- update(hd.lme.1,
+                  random = ~ dbhib.cm - 1 | Tree.ID)
```

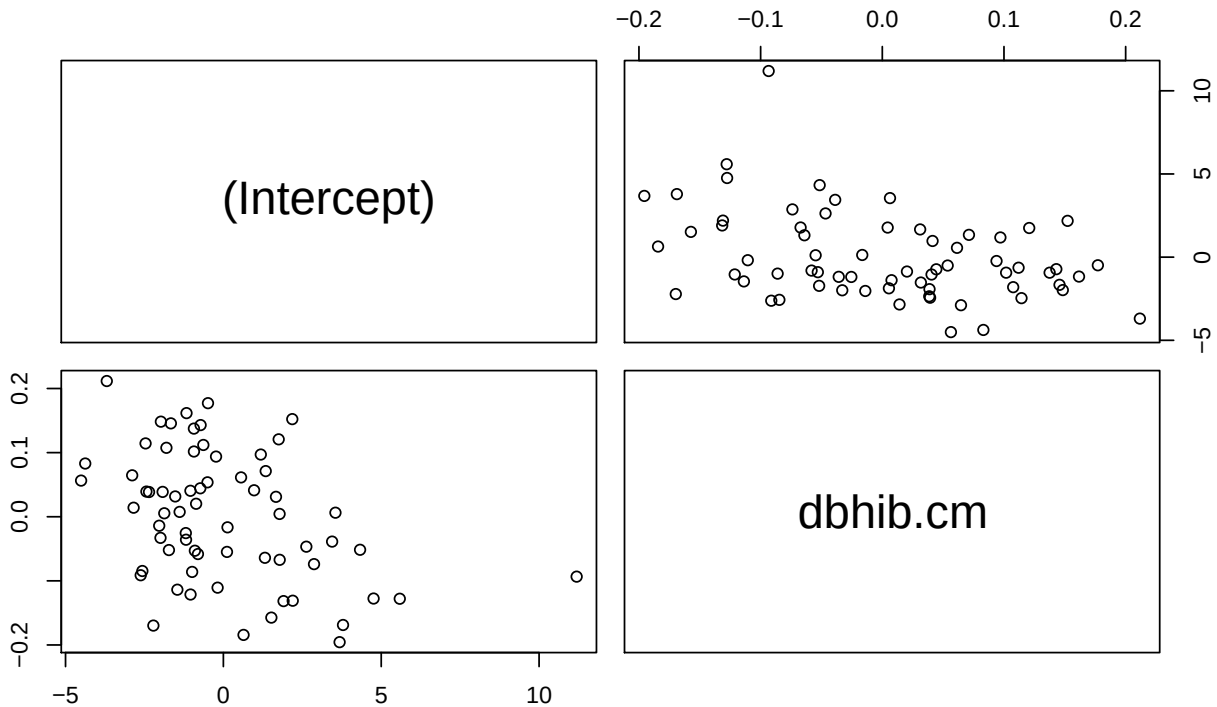
- `lmer()`

```
> hd.lmer.2 <-
+   update(hd.lmer.1,
+          formula = height.m ~ dbhib.cm +
+                    (dbhib.cm | Tree.ID))

> hd.lmer.3 <-
+   update(hd.lmer.1,
+          formula = height.m ~ dbhib.cm +
+                    (dbhib.cm - 1 | Tree.ID))
```

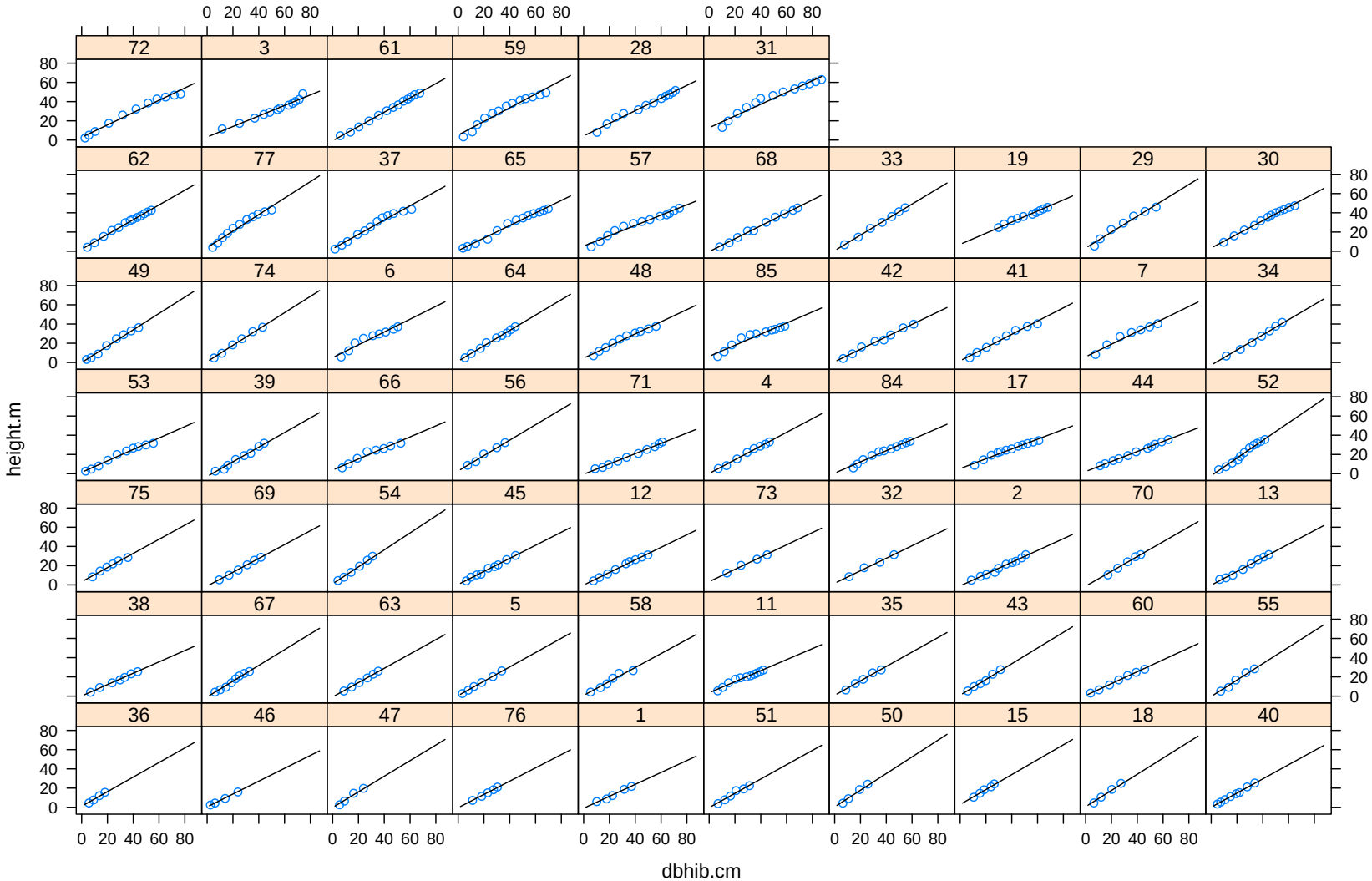
lme() More than one random effect per subject.

```
> pairs(ranef(hd.lme.2))
```



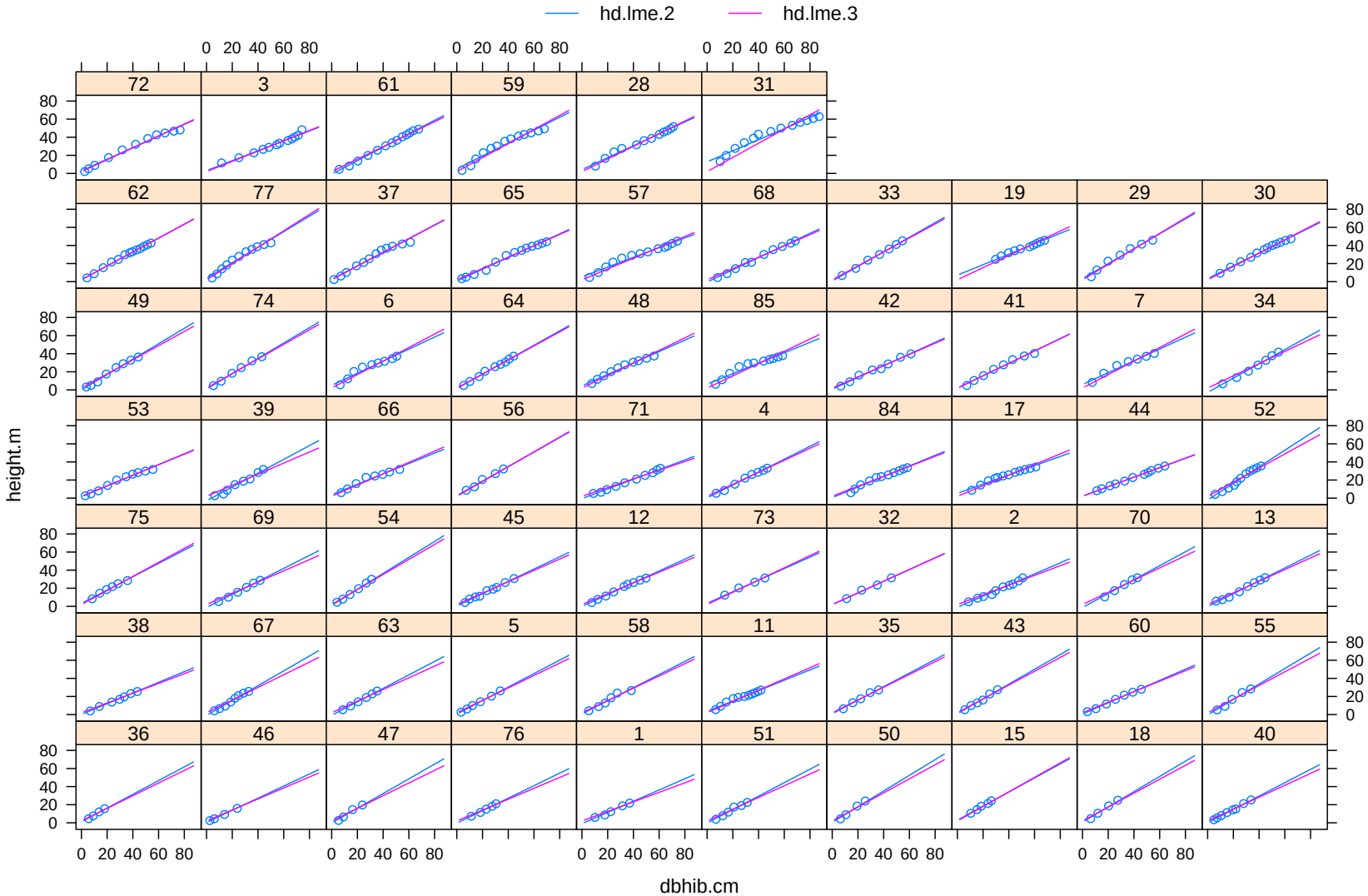
lme() Snapshot.

```
> plot(augPred(hd.lme.2))
```



lme() Double Snapshot.

```
> plot(comparePred(hd.lme.2, hd.lme.3))
```



`lme()` The *other* correlation.

The following structures are supported for the covariance matrices of random effects.

- `pdBlocked`: a block-diagonal matrix, comprising other types.
- `pdCompSymm`: a compound-symmetry matrix
- `pdIdent`: some multiple of the identity matrix
- `pdSymm`: the default, a general positive definite matrix

lme() Specifying the control parameters

I often use ...

```
control = lmeControl(
              maxIter = 5000,                 # Of course
              msMaxIter = 5000,               #      it helps
              niterEM = 500,                  #      to be
              msMaxEval = 500,                #      patient!
              msVerbose = TRUE,               # Loquacity ...
              opt = "optim",                  # Optional, and
              optimMethod = "Nelder-Mead" #    often useful
              )
```

Changing the optimizer can make the difference between convergence and failure.

> help(lmeControl)

Estimating Random Effects

Point Estimates

```
> summary(hd.lme.1)

...

Random effects:
 Formula: ~1 | Tree.ID
        (Intercept) Residual
StdDev:    2.976158 2.386359

> VarCorr(hd.lme.1)

Tree.ID = pdLogChol(1)
            Variance StdDev
(Intercept) 8.857514 2.976158
Residual    5.694708 2.386359
```

Estimating Random Effects

Interval Estimates

```
> intervals(hd.lme.1)

...

 Random Effects:
  Level: Tree.ID
                       lower     est.    upper
sd((Intercept)) 2.474999 2.976158 3.578796

 Within-group standard error:
   lower     est.    upper
2.239627 2.386359 2.542704
```

Inference for Random Effects

Fit two models: one *including* the random effect(s) under question, and the other *excluding* it/them. Compare the models using the `anova()` function, which performs a likelihood ratio test (among other things).

```
> anova(hd.lme.1, hd.lme.2)
```

```
         Model df      AIC      BIC    logLik   Test  L.Ratio p-value
hd.lme.1     1  4 2664.745 2681.911 -1328.372
hd.lme.2     2  6 2463.740 2489.489 -1225.870 1 vs 2 205.0049  <.0001
```

Estimating Fixed Effects

Point Estimates

```
> summary(hd.lme.1)

...

Fixed effects: height.m ~ dbhib.cm
                Value Std.Error  DF  t-value p-value
(Intercept) 2.2896107 0.4268314 475   5.3642       0
dbhib.cm    0.6628527 0.0065173 475 101.7070       0
 Correlation:
         (Intr)
dbhib.cm -0.445
```

Estimating Fixed Effects

Interval Estimates

```
> intervals(hd.lme.1)

...

 Fixed effects:
                lower       est.    upper
(Intercept) 1.4508994 2.2896107 3.128322
dbhib.cm    0.6500465 0.6628527 0.675659
```

Inference for Fixed Effects

Be Alert, not Alarmed.

The degrees of freedom and therefore the tests that they index should be viewed as approximate.

Use the `anova()` function to construct sequential conditional F tests.

```
> anova(hd.lme.1)
```

	numDF	denDF	F-value	p-value
(Intercept)	1	475	3198.518	<.0001
dbhib.cm	1	475	10344.325	<.0001

Inference for Fixed Effects

If the term order is inappropriate, then refit (`update()`!) the model, replacing the expression of the fixed effects with:

```
terms(y ~ x1 * x2 + x3, keep.order=TRUE)
```

The *t*-tests reported in the various coefficient tables are marginal. The `anova()` function will perform marginal tests also, by calling the `type = "marginal"` argument.

lme() Alert or Alarmed?

The authors have provided a simulation function that permits the checking of various large-sample assumptions for any given model. Denote a null model and an alternative model. Data are generated from the null model and then fit to both models, and the LRT computed.

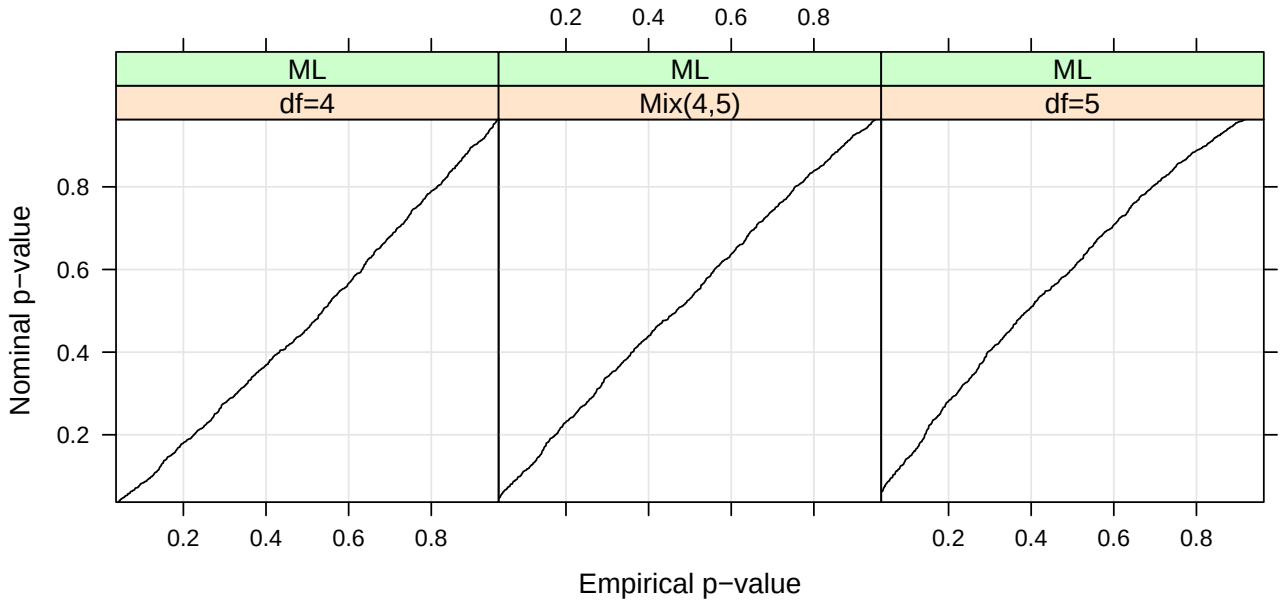
A similar tool might become available for lmer at some point.

```
> hd.lme.4 <- update(hd.lme.1,
+                    fixed = height.m ~ dbhib.cm + HabType.ID)

> simulate.hd <- simulate.lme(hd.lme.1, hd.lme.4,
+                             method = "ML",
+                             nsim=1001, seed=2115153)
```

lme() Alert or Alarmed?

```
> plot(simulate.hd, df = c(4, 5))
```



Other relationships between random effects

The consideration of only nested random effects amounts to a substantial constraint on model fitting.

In reality, effects can be any structure. For example, ...

Stage Data

The 66 trees (t) were measured at multiple ages (a) in 9 forests (f) and 5 habitat types (h). Forests are crossed with habitat types, and trees are nested within both.

$$h_{atfh} = \beta_0 + b_f + b_h + b_{t(fh)} + \beta_1 \times d_{at(fh)} + \epsilon_{atfh} \qquad (7)$$

Regression assumptions.

- True relationship is linear.
- $b_f \sim$ iid $\mathcal{N}(0, \sigma^2_{b_f})$
- $b_h \sim$ iid $\mathcal{N}(0, \sigma^2_{b_h})$
- $b_{t(fh)} \sim$ iid $\mathcal{N}(0, \sigma^2_{b_t})$
- $\epsilon_{atfh} \sim$ iid $\mathcal{N}(0, \sigma^2)$

NB: a 1-d hierarchical structure of residuals no longer makes sense ...

New Strategy

```
> hd.lmer.0 <-
+   lmer(height.m ~ dbhib.cm + (1|Tree.ID), data=stage)
```

```
> hd.lmer.1 <-
+   lmer(height.m ~ dbhib.cm + (1|Forest.ID) +
+         (1|HabType.ID) + (1|Tree.ID), data=stage)
```

New Strategy

```
> str(ranef(hd.lmer.1, postVar = TRUE))

Formal class 'ranef.lmer' [package "lme4"] with 1 slots
  ..@ .Data:List of 3
  .. ..$ :'data.frame': 66 obs. of  1 variable:
  .. .. ..$ (Intercept): num [1:66] -2.09 -2.68 -6.10 -1.74 -1.41 ...
  .. .. ..- attr(*, "postVar")= num [1, 1, 1:66] 1.56 1.22 1.04 1.33 1.47 ...
  .. ..$ :'data.frame': 9 obs. of  1 variable:
  .. .. ..$ (Intercept): num [1:9]  0.0579 -0.3830  0.0811 -0.3708  0.8248 ...
  .. .. ..- attr(*, "postVar")= num [1, 1, 1:9] 0.528 0.434 0.475 0.464 0.327 .
  .. ..$ :'data.frame': 5 obs. of  1 variable:
  .. .. ..$ (Intercept): num [1:5]  1.174 -0.647  1.791 -0.563 -1.755
  .. .. ..- attr(*, "postVar")= num [1, 1, 1:5] 1.158 2.180 1.110 0.952 1.034

> str(residuals(hd.lmer.1))

 num [1:542] -0.902 -0.249 -0.687 -1.164  1.138 ...
```

New Strategy

```
> anova(hd.lmer.0, hd.lmer.1)

Data: stage
Models:
hd.lmer.0: height.m ~ dbhib.cm + (1 | Tree.ID)
hd.lmer.1: height.m ~ dbhib.cm + (1 | Forest.ID) + (1 | HabType.ID) + (1 |
hd.lmer.0:     Tree.ID)
           Df     AIC     BIC  logLik  Chisq Chi Df Pr(>Chisq)
hd.lmer.0   3  2654.4  2667.3 -1324.2
hd.lmer.1   5  2650.4  2671.9 -1320.2 8.0161      2    0.01817 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What information is available?

```
> slotNames(summary(hd.lmer.1))

 [1] "isG"        "methTitle" "logLik"    "ngrps"     "sigma"     "coefs"
 [7] "vcov"       "REmat"     "AICtab"    "flist"     "Zt"        "X"
[13] "y"          "wts"       "wrkres"    "cnames"    "nc"        "Gp"
[19] "XtX"        "ZtZ"       "ZtX"       "Zty"       "Xty"       "Omega"
[25] "L"          "RZX"       "RXX"       "rZy"       "rXy"       "devComp"
[31] "deviance"   "fixef"     "ranef"     "RZXinv"    "bVar"      "gradComp"
[37] "status"     "frame"     "call"      "terms"
```

Variance estimates

```
> summary(hd.lmer.1)@REmat

 Groups        Name           Variance  Std.Dev.
 "Tree.ID"     "(Intercept)" "6.38991" "2.52783"
 "Forest.ID"   "(Intercept)" "0.65699" "0.81055"
 "HabType.ID" "(Intercept)" "2.96455" "1.72179"
 "Residual"    ""            "5.69223" "2.38584"
```

Fixed effects: something's missing

```
> summary(hd.lmer.1)@coefs

             Estimate  Std. Error    t value
(Intercept) 2.325739 0.963098970   2.414849
dbhib.cm    0.662185 0.006508289 101.744860

> anova(hd.lmer.1)

Analysis of Variance Table
          Df Sum Sq Mean Sq
dbhib.cm   1  58926   58926
```

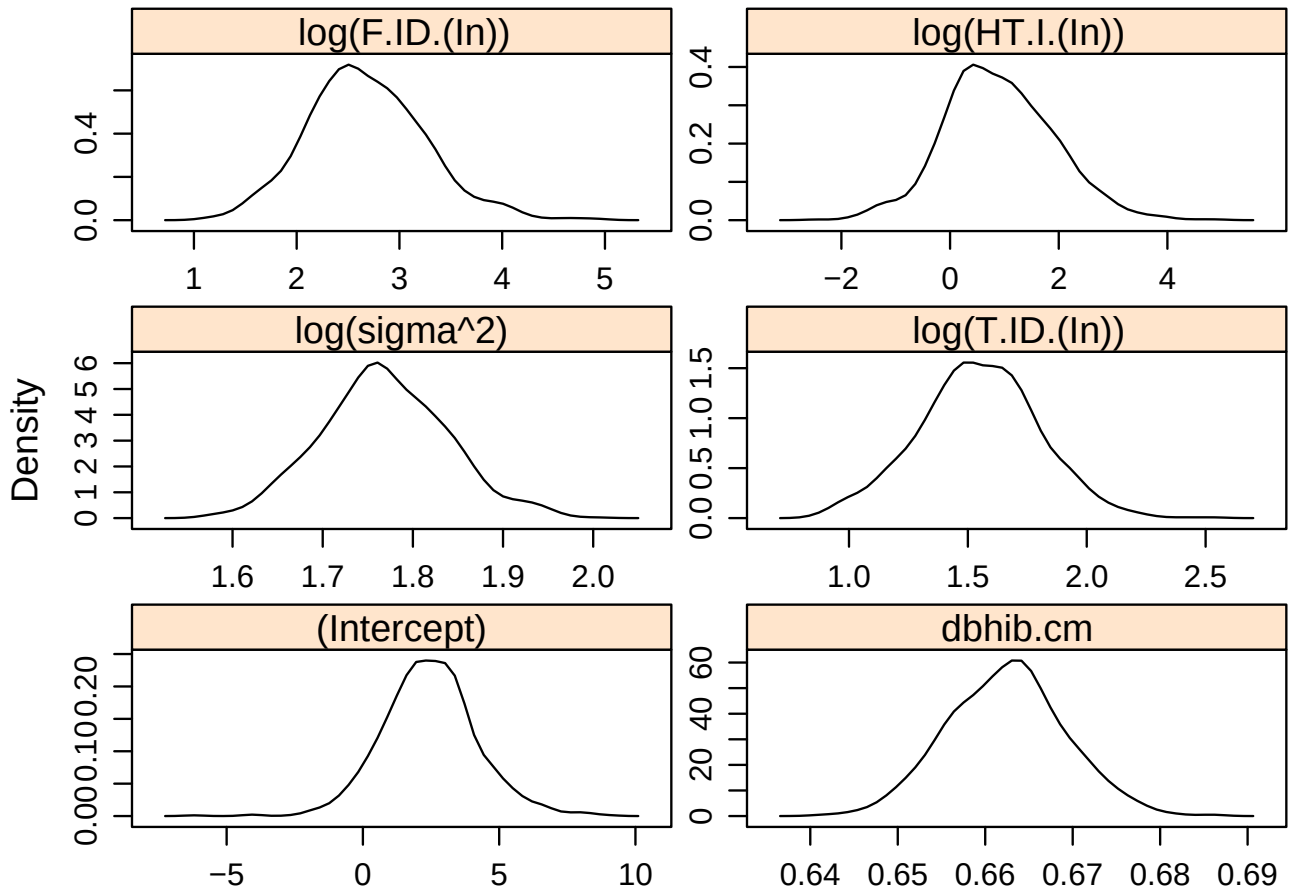
So, `lmer()` provides no *p*-values for fixed effects!

1. Degrees of freedom are unknown, and approximations are poor.
2. The tests ignore the extra uncertainty from estimating random effects (they condition on the random effects.)
3. We don't know the underlying distribution of the parameter estimates.

```
> require(coda)
> mcmc.hd <- mcmcsamp(hd.lmer.1, n = 1001)
> densityplot(mcmc.hd, plot.points = FALSE)
```

Density Plot

```
> print(densityplot(mcmc.hd, plot.points = FALSE))
```



New Strategy

```
> HPDinterval(mcmc.hd)

                       lower     upper
(Intercept)     -1.0057929 5.7980535
dbhib.cm         0.6489785 0.6750611
log(sigma^2)     1.6243177 1.9027143
log(T.ID.(In))   1.0089994 2.0111319
log(F.ID.(In))   1.4614669 3.7438338
log(HT.I.(In)) -0.9911411 2.9967855
attr(,"Probability")
[1] 0.95005
```

New Strategy

```
> require(languageR)
> pvals.fnc(hd.lmer.1, nsim = 1001)
$fixed
             Estimate MCMCmean HPD95lower HPD95upper  pMCMC Pr(>|t|)
(Intercept)    2.3257   2.2781    -0.9768     5.8582 0.1578   0.0161
dbhib.cm       0.6622   0.6626     0.6507     0.6765 0.0010   0.0000

$random
           MCMCmean HPD95lower HPD95upper
sigma         2.428     2.2645      2.603
T.ID.(In)     2.117     1.6956      2.719
F.ID.(In)     3.845     2.2808      6.704
HT.I.(In)     1.553     0.5937      4.169
```

Designed Experiments: Case study.

Oats growth data: the help file in R says:

> *The treatment structure used in the experiment was a 3×4 full factorial, with three varieties of oats and four concentrations of nitrogen. The experimental units were arranged into six blocks, each with three whole-plots subdivided into four subplots. The varieties of oats were assigned randomly to the whole-plots and the concentrations of nitrogen to the subplots. All four concentrations of nitrogen were used on each whole-plot.*

Get the data and examine it.

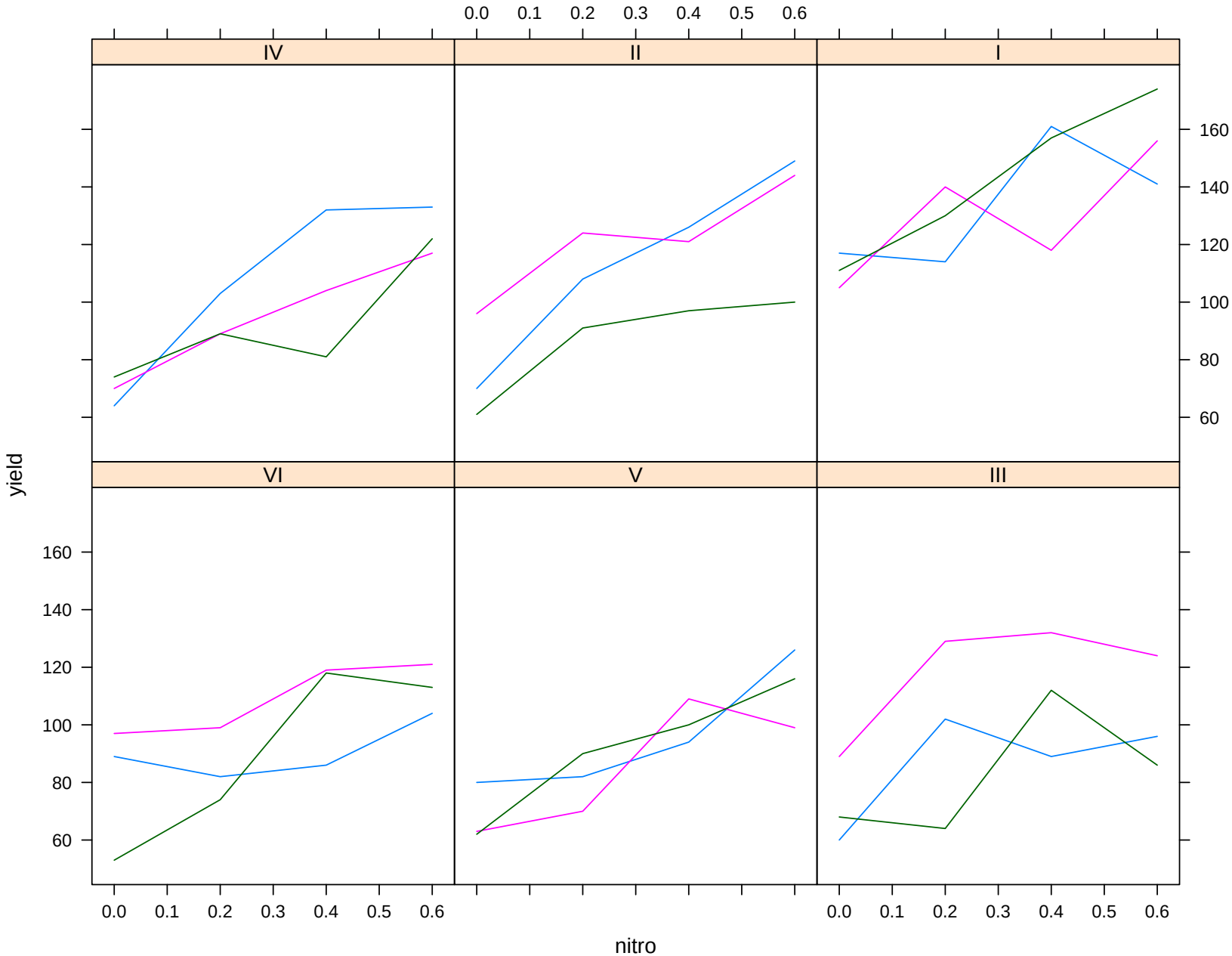
```
> data(Oats)
> str(Oats)

Classes 'nfnGroupedData', 'nfGroupedData', 'groupedData' and 'data.frame': 72 obs.
 $ Block  : Ord.factor w/ 6 levels "VI"<"V"<"III"<..: 6 6 6 6 6 6 6 6 6 6 ...
 $ Variety: Factor w/ 3 levels "Golden Rain",..: 3 3 3 3 1 1 1 1 2 2 ...
 $ nitro  : num  0 0.2 0.4 0.6 0 0.2 0.4 0.6 0 0.2 ...
 $ yield  : num  111 130 157 174 117 114 161 141 105 140 ...
 - attr(*, "formula")=Class 'formula' length 3 yield ~ nitro | Block
  .. ..- attr(*, ".Environment")=<R_GlobalEnv>
 - attr(*, "labels")=List of 2
  ..$ y: chr "Yield"
  ..$ x: chr "Nitrogen concentration"
 - attr(*, "units")=List of 2
  ..$ y: chr "(bushels/acre)"
  ..$ x: chr "(cwt/acre)"
 - attr(*, "inner")=Class 'formula' length 2 ~Variety
  .. ..- attr(*, ".Environment")=<R_GlobalEnv>
```

Represent the data somehow.

```
> require(lattice)

> xyplot(yield ~ nitro | Block,
+        group = Variety,
+        type = "l",
+        data = Oats)
```


IV
II
I
VI
V
III
yield
nitro
0.0
0.1
0.2
0.3
0.4
0.5
0.6
60
80
100
120
140
160

Effect an analysis

```
> Oats$nitro <- factor(Oats$nitro)
> oats.1 <- lme(yield ~ nitro * Variety,
+                 random = ~ 1 | Block / Variety,
+                 data = Oats)
```

Diagnostics

Scatterplot of residuals:

```
> scatter.smooth(fitted(oats.1), resid(oats.1))
> abline(h = 0, col = "tomato2")
```

qq-plot of residuals:

```
> qqnorm(resid(oats.1))
> qqline(resid(oats.1), col = "maroon4")
```

Variance-checking plot:

```
> scatter.smooth(fitted(oats.1), sqrt(abs(resid(oats.1))))
```

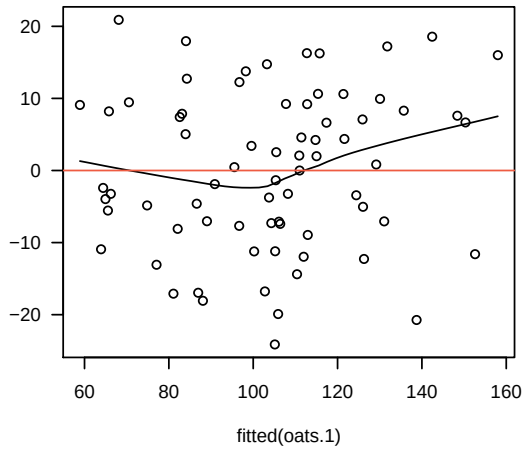
qq-plot of standardized block random effects:

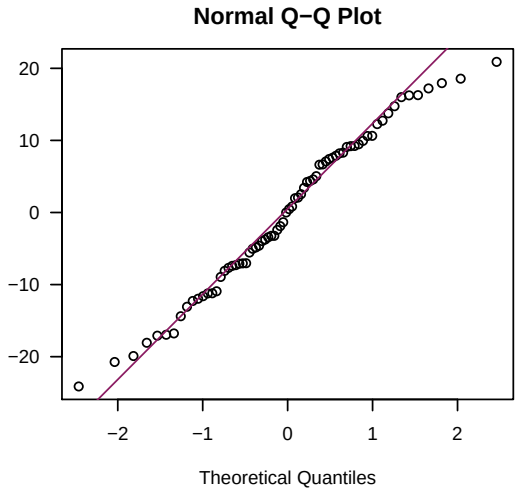
```
> qqnorm(ranef(oats.1, standard = TRUE)[[1]][, 1])
> qqline(ranef(oats.1, standard = TRUE)[[1]][, 1], col = "steelblue4")
```

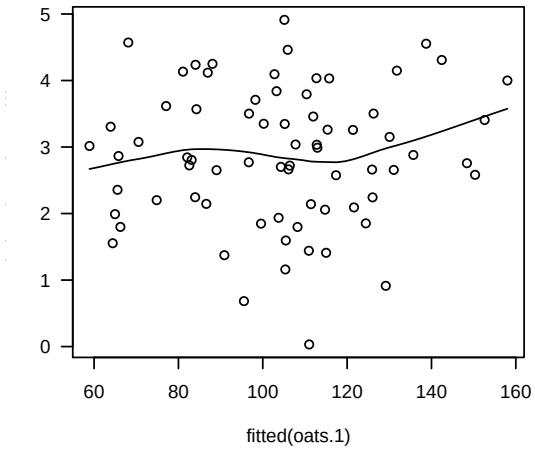
qq-plot of standardized variety within block random effects:

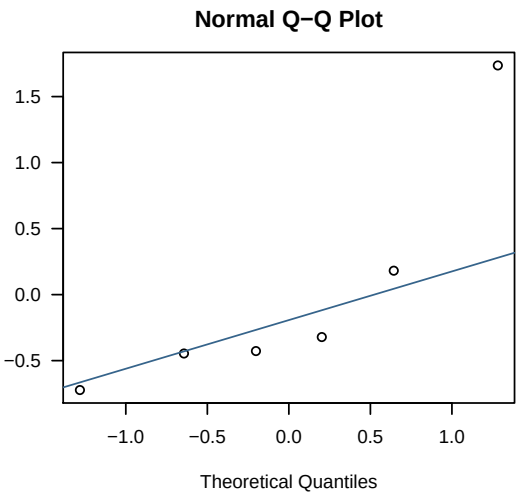
```
> qqnorm(ranef(oats.1, standard = TRUE)[[2]][, 1])
> qqline(ranef(oats.1, standard = TRUE)[[2]][, 1], col = "violetred3")
```

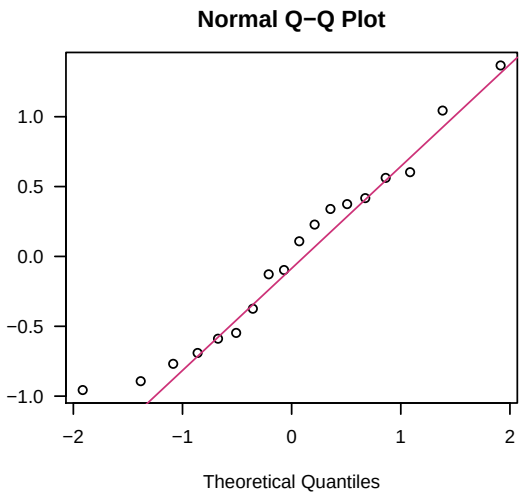
Check assumptions











Check assumptions

One slightly odd block ...

```
> ranef(oats.1)[[1]]
```

```
    (Intercept)
VI    -6.259679
V    -10.582911
III   -6.529881
IV    -4.706018
II     2.656986
I     25.421504
```

```
> anova(oats.1)
```

	numDF	denDF	F-value	p-value
(Intercept)	1	45	245.14333	<.0001
nitro	3	45	37.68561	<.0001
Variety	2	10	1.48534	0.2724
nitro:Variety	6	45	0.30282	0.9322

```
> summary(oats.2 <- update(oats.1, fixed = yield ~ nitro))
Linear mixed-effects model fit by REML
 Data: Oats
       AIC      BIC    logLik
  596.2187 611.7553 -291.1094

Random effects:
 Formula: ~1 | Block
        (Intercept)
StdDev:    14.50596

 Formula: ~1 | Variety %in% Block
        (Intercept) Residual
StdDev:    11.03868 12.74987

Fixed effects: yield ~ nitro
               Value Std.Error DF   t-value p-value
(Intercept) 79.38889  7.132398 51 11.130742       0
nitro0.2    19.50000  4.249955 51  4.588283       0
nitro0.4    34.83333  4.249955 51  8.196164       0
nitro0.6    44.00000  4.249955 51 10.353050       0
 Correlation:
         (Intr) ntr0.2 ntr0.4
nitro0.2 -0.298
nitro0.4 -0.298  0.500
nitro0.6 -0.298  0.500  0.500

Standardized Within-Group Residuals:
```

```
> summary(oats.3 <- update(oats.1, fixed = yield ~ ordered(nitro)))
Linear mixed-effects model fit by REML
 Data: Oats
      AIC      BIC    logLik
  597.605 613.1416 -291.8025

Random effects:
 Formula: ~1 | Block
        (Intercept)
StdDev:    14.50596

 Formula: ~1 | Variety %in% Block
        (Intercept) Residual
StdDev:    11.03866 12.74987

Fixed effects: yield ~ ordered(nitro)
                     Value Std.Error DF   t-value p-value
(Intercept)      103.97222  6.640618 51 15.657009  0.0000
ordered(nitro).L  32.94473  3.005173 51 10.962675  0.0000
ordered(nitro).Q  -5.16667  3.005173 51 -1.719258  0.0916
ordered(nitro).C  -0.44721  3.005173 51 -0.148815  0.8823
 Correlation:
                  (Intr) or().L or().Q
ordered(nitro).L 0
ordered(nitro).Q 0      0
ordered(nitro).C 0      0      0

Standardized Within-Group Residuals:
```

```
> summary(aov(yield ~ nitro * Variety + Error(Block/Variety),
+             data = Oats))

Error: Block
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals  5 15875.3  3175.1

Error: Block:Variety
          Df Sum Sq Mean Sq F value Pr(>F)
Variety    2 1786.4   893.2  1.4853 0.2724
Residuals 10 6013.3   601.3

Error: Within
              Df  Sum Sq Mean Sq F value    Pr(>F)
nitro          3 20020.5  6673.5 37.6856 2.458e-12 ***
nitro:Variety  6   321.7    53.6  0.3028    0.9322
Residuals     45  7968.7   177.1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The roles differ

For the design,

- fixed effects represent *themselves*;
- random effects represent *a population*.

Within the model,

- fixed effects *explain* variation;
- random effects *organize* unexplained variation.

Random effects are effects that common sense says will explain variation, but you don't want to have to know them in order to be able to apply the model.

Utility

Mixed effects models also allow the estimation of useful quantities.

- Variance components.
- Intra-class correlation.

Modelling is much more involved

Add a new dimension to your flow chart!

A Candidate Modelling Strategy

The modeling strategy depends on the modeler's intention.

1. Fit baseline model.
   - 1.1 Include the meaningful fixed effects.
   - 1.2 Include the design random effects.
2. Check the assumption diagnostics.
3. Add or modify random components until diagnostics are satisfied.
   - 3.1 a heteroskedastic variance structure (several candidates)
   - 3.2 a correlation structure (several candidates)
   - 3.3 extra random effects (e.g. random slopes)
4. Consider adding more fixed effects.
5. Re-examine the diagnostics, add/modify random effects, etc.

Questions?

Hierarchical Linear Models

Assumptions

Random Effects

Fixed Effects

Further Developments

Designed Experiments

Wrap-up

$$\mathbf{Y} = \mathbf{X}\beta + \mathbf{Zb} + \epsilon$$

$$\text{Var}\left(\mathbf{Y} \mid \mathbf{X}, \mathbf{Z}, \beta, \mathbf{b}\right) = \mathbf{R}$$

$$\text{Var}\left(\mathbf{Y} \mid \mathbf{X}, \beta\right) = \mathbf{ZDZ}' + \mathbf{R} = \mathbf{V}$$

Log Likelihood

$$\mathcal{L}(\beta, \mathbf{V} \mid \mathbf{Y}, \mathbf{X}) = -\frac{1}{2}\ln(|\mathbf{V}|) - \frac{n}{2}\ln(2\pi) - \frac{1}{2}(\mathbf{Y} - \mathbf{X}\beta)' \mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\beta)$$

Profile β out

$$\hat{\beta} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{Y}$$

β is gone!

$$\mathcal{L}(\beta, \mathbf{V} \mid \mathbf{Y}, \mathbf{X}) = f(\mathbf{V} \mid \mathbf{Y}, \mathbf{X}, \mathbf{Z})$$

Estimate $\hat{V}$ by maximization and then $\hat{\beta}$ by substitution.

ReML

NB: Maximum likelihood estimators of covariance parameters are usually negatively biased.

Briefly, ReML involves applying ML, but replacing

- $\mathbf{Y}$ with $\mathbf{KY}$;
- $\mathbf{X}$ with $\mathbf{0}$;
- $\mathbf{Z}$ with $\mathbf{K}'\mathbf{Z}$; and
- $\mathbf{V}$ with $\mathbf{K}'\mathbf{VK}$

where $\mathbf{K}$ is *any* $\mathbf{K}$ such that $\mathbf{K}'\mathbf{X} = 0$.

e.g. $\mathbf{K}' = \mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$

- $\mathbf{Y}$ becomes $\mathbf{Y} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$;
- $\mathbf{X}$ becomes 0;
- $\mathbf{Z}$ becomes $\mathbf{Z} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{XZ}$; and
- $\mathbf{V}$ becomes $(\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')\mathbf{V}(\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')'$

Multiple Comparisons

It is often necessary to obtain simultaneous estimates of fixed-effects parameters.

The `estimable (gmodels)` function allows for joint testing of arbitrary linear contrasts, but not joint interval estimation.

Joint estimation and size-corrected testing requires the `glht (multcomp)` function.

Estimating Contrasts of Fixed Effects

Use the `estimable()` function from the third-party package called `gmodels` to obtain arbitrary linear contrasts of the model parameter estimates. *E.g.* we get a prediction of the height for a 30 inch tree by:

```
> require(gmodels)
> estimable(hd.lme.1, c("(Intercept)" = 1, dbhib.cm = 30))
```

You'll get a warning. Don't take it personally.

```
help(estimable)
```