1 Modell

The model we are talking about is the same as in the previous and presumably the next few talks and consists of a fixed and a random part:

$$y_i = \alpha + \beta X_i + b_i Z_i + \epsilon_i, \qquad i = 1, \ldots, N, \tag{1}$$

If we put all the groups i together (every group having n_i observations, with a total of N_T, m fixed and k random effects) and write one single model, we end up with:

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{N_T \times 1} = \underbrace{\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix}}_{N_T \times m} \underbrace{\begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix}}_{m \times 1} + \underbrace{\begin{bmatrix} Z_1 & 0 & 0 & 0 \\ 0 & Z_2 & 0 & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \ldots & Z_N \end{bmatrix}}_{N_T \times Nk} \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}}_{Nk \times 1} + \underbrace{\begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{bmatrix}}_{N_T \times 1} \tag{2}$$

We assume that the errors and the random effects are normally distributed, so we can calculate the covariance matrix of all random effects (error plus the b_i), defining $D := D_* \sigma^{-2}$ and knowing that thus $cov(b) = \sigma^2(I \times D)$:

$$V = \sigma^2 \begin{bmatrix} I_{n_1} + Z_1 D Z_1^T & 0 & 0 & 0 \\ 0 & I_{n_2} + Z_2 D Z_2^T & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & I_{n_N} + Z_N D Z_N^T \end{bmatrix} \tag{3}$$

2 Log-Likelihood function

From the density of the multivariate normal distribution we conclude

$$l(\theta) = -\frac{1}{2}\left(N_T ln \sigma^2 + \sum_{i=1}^{N} [ln|I + Z_i D Z_i^T| + \sigma^{-2}(y_i - X_i\beta)^T (I + Z_i D Z_i^T)^{-}1(y_i - X_i\beta)] \right) \tag{4}$$

This is what we want to optimise over $\beta \sigma^2 vec(D)$. However this involves taking determinants and inverses of large matrices. We can Improve the function.

3 Dealing with computational problems

There are (at least) three ways in which we can simplify this expression:

- **Dimension reduction:** As we see, $l(\theta)$ includes both $|V_i| and V^{-1}$. As V is of the dimension $n_i \times n_i$ the computation might take a while (As the algorithms are iterative, this is a problem). But we can get a formula only containing $k \times k$ matrices that have to be inverted.

- **Prfoile-likelihood:** We can substitute β and σ^2 as they can be expressed in terms of D. Thus we have to optimise over a less dimensional parameter space.
- **Inverse D:** We can even find a Form totally avoiding matrix inverses, which facilitates computation even more.

4 The three algorithms

All algorithms that we present here have an iterative approach.

The NR-Algorithm, as well as the FS-Algorithm have the form:

$$u_{s+1} = u_s + \lambda_s H_s^{-1} g_s \tag{5}$$

Where s denotes the iteration index, H_s a positive definite matrix and g the gratient of the Function at the point u_s. λ is in $(0, 1]$ such that $F(u_{s+1}) > F(u_s)$. (E.g. one first takes $\lambda = 1$ and if the likelihood improves one keeps the result, otherwise one takes $\lambda = 1/2, 1/4, 1/8$ etc until one has an improvement, which eventually will be the case as analysis tells us.) The Newton-Raphson algorithm takes the negative Hessian for H but this is not always positive definite, which is required for the algorithm to work properly). The Fisher-Scoring algorithm takes the expected negative Hessian, which is always positive definite and therefore it is more commonly used. Also an algorithm using some steps of FS first and then go over to NR is possible.

We can formulate the NR-Algorithm as:

$$\begin{bmatrix} \beta_{s+1} \\ \sigma^2_{s+1} \\ vec(D_{s+1}) \end{bmatrix} = \begin{bmatrix} \beta_s \\ \sigma^2_s \\ vec(D_s) \end{bmatrix} + \lambda_s \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{12}^T & H_{22} & H_{23} \\ H_{13}^T & H_{23}^T & H_{33} \end{bmatrix}_s^{-1} \begin{bmatrix} \frac{\partial l}{\partial \beta} \\ \frac{\partial l}{\partial \sigma^2} \\ vec(\frac{\partial l}{\partial D}) \end{bmatrix}_s \tag{6}$$

where the entries of H are blocks of complicated entries, making up the symmetric hessian matrix of $l(\theta)$

Last there is the EM-Algorithm consisting of an **E**stimation and an **M**aximation step. First the expectation of the joint log likelihood is calculated (fore some θ_t and then the maximiser of that function is taken to be the seed for the next calculation of the expectation of the joint likelihood. This process is then iterated until the changes become small enough.