Lineær mikset modell for kompressor data med en applikasjon

Mads Herdahl

Master i fysikk og matematikk
Oppgaven levert: Juni 2008
Hovedveileder: Jo Eidsvik, MATH

Norges teknisk-naturvitenskapelige universitet
Institutt for matematiske fag

Oppgavetekst

Make a Linear Mixed Model for Compressor Data. And make an application to be used as a better alternative to todays practice, which is to manually read numbers from diagrams.

Oppgaven gitt: 17. januar 2008
Hovedveileder: Jo Eidsvik, MATH

LINEAR MIXED MODEL FOR COMPRESSOR HEAD AND FLOW DATA WITH AN APPLICATION

by

MADS HERDAHL

MASTER THESIS
Statistics

NTNU
Norwegian University of
Science and Technology

Department of Mathematics, NTNU

June 12th 2008

Abstract

StatoilHydro is the operator of the Åsgard oil and gas field outside of Trøndelag, Norway, where large compressors for injection, recompression and export of natural gas are installed. The facility transports and stores up to 36 millions Sm^3 of gas every day. If the compressors are not optimally operated, large values are lost.

This paper describes how to use linear mixed models to model the condition of the compressors. The focus has been on the 1- and 2- stage recompression compressors. Reference data from Dresser-Rand have been used to make the model. Head and flow data are the modelled, and the explanatory variables used are molweight, rotational speed and an efficiency indicator. The paper also shows how cross validation is used to give an indication of how future datapoints will fit the model. A graphical user interface has been developed to do estimation and plotting with various models.

Different models are tested and compared by likelihood methods. For a relatively simple model using three explanatory variables reasonable predictions are obtained. Results are not so good for very high rotational speeds and high molweights.

Contents

List of Figures

1 Introduction

StatoilHydro operates the Åsgård oil and gas field outside of Trøndelag, Norway. The field has installed several compressors for injection, recompression and transportation of gas. The compressors are critical components during transportation of the gas. They provide the mechanism to move large quantities of gas around a network to meet instantaneous demand and to replenish local storage, see Pearson and Henderson [9].

It is very important to monitor the performance of a compressor to ensure it runs optimally,

- Firstly, if it runs at a deteriorated state the customers will not get their demands fullfilled. This will of course prevent cash flow to an oil and gas company.

- When a compressor is not optimally run, it may cause surge and choke conditions. Surge and choke conditions will in some cases result in breakdown of the machinery, which again will lead to customers not getting their demands fullfilled and hence a great loss of income. When a compressor has broken down it is very costly to replace or repair.

- Since the 80s enviromental policies have become more and more important. A company of StatoilHydro's size is obliged to do its part to help keeping the environment as clean as possible. A reduction of the compressor's fuel usage in a year of 1% will prevent the production of 6 million tonnes of CO_2 per year, [9]. An optimal fuel usage will definitely let StatoilHydro play a role as a pioneer at environmental issues among similar companies.

Each compressor has a head and flow chart. Head is the mechanical work made by the compressor pump in $[\frac{\mathrm{J}}{\mathrm{kg}}]$. Flow is the volume of gas that flows through the pipe in $[\frac{\mathrm{m}^3}{\mathrm{h}}]$. Complete charts are either based on compressor rig test results, or alternatively the chart of a similar compressor can be suitably scaled. Compressor charts are important, since they are an integral part of predicting the performance of the compressor. Condition analysis is the term that describes controlling the compressors by observing different operating values. These values can be used directly or calculated to other parameters using equations of state. When the operating value is found it will be compared to an expected value. The main control parameters that are used are the head and flow values. Today, the procedure in condition analysis is to manually study data from the compressors with reference charts to spot possible anomalities. The reference data were measured when the compressor was new.

Sample charts from the compressors are very expensive to get, so few tests are made, and those test samples are mainly gotten from the most common conditions. It is therefore higly advisable to get a model that does not require operators to manually read charts to control the compressors. A model will in addition to avoid human error also give a more accurate condition analysis.

This thesis will describe how Dresser-Rand's reference data are used to make a Linear Mixed Model for the Head and Flow data at the Åsgård field for the 1-stage recompression compressors. Such a model might prevent the manual work of reading charts. By feeding the model with the measured operating values a prediction for head and flow values are given. Flow is the gas volume flow through the compressor, while head is defined as an integration of infinite many small isentropic part compressions through the real compression line, see Øverli [10] and Ambjørnsen [1].

The report is divided in 4 main parts. Chapter 2 gives a brief introduction to the thermodynamics behind the compressors. This chapter also contains a deeper presentation of the Åsgård dataset provided by Dresser-Rand. In Chapter 3 the statistical theory behind the Linear Mixed Model is presented. How the Linear Mixed Model is used on the Dresser-Rand dataset will also be described. The theory behind Cross Validation will be presented, and how this validation method is used on the Dresser-Rand dataset. There will also be a discussion around model choice. In chapter 4 the results from the choice for model will be presented and discussed along with results from cross validation.

2 Data of Polytropic Head and Volume

This chapter will give a brief introduction to the theory behind compressors, which is based on thermodynamic theory derived under ideal conditions. Further, the parameters head and flow are defined, which are the main control parameters to run a compressor optimally. The operating values should always be measured for thermodynamic condition analysis, and these values are used directly or calculated to other parameters, using equations of state. When the operating value is found, it will be compared to an expected value. The expected value is given by Dresser-Rand and is from when the compressor was new. Condition analysis involves monitoring these parameters and comparing them to the reference data.

Finally, the specific dataset of head and flow values from the Åsgård field that is used in this thesis is presented. The data are collected from a recompression process. Different cross plots are made as a preliminary data analysis and interpretations along with the natural explanatory variables.

Variables that are used in the data analysis are

- molweight, denoted m [kg],
- rotational speed, denoted n [rpm],
- efficiency, denoted P [W].

2.1 Head/Flow Diagram

Compressors are run under different pressures and temperatures, but at an installation the most important parameters to control are the parameters head and flow, H and Q. H is defined as the mechanical work made by the pump,

$$H = \int_{p_1}^{p_2} v \mathrm{d}p \qquad \left[\frac{\mathrm{J}}{\mathrm{kg}}\right]. \tag{1}$$

This is a compression from a pressure p_1 to a pressure p_2. See Øverli [10] for a derivation of the specific work made by a pump. H is a defined parameter that is not physically possible to measure directly. Q is intuitively the volume of gas that flows through a pipe per hour (m^3/h),

$$Q = \frac{\Delta P r^4 \pi}{8 L \eta} \qquad \left[\frac{\mathrm{m}^3}{\mathrm{h}}\right], \tag{2}$$

where ΔP is the pressure between the two ends of the tube, r is the radius of the tube, L is the length of the tube and η is the viscosity.

The highest efficiency of a compressor is somewhere between the surge and choke areas. But this point may vary from the different rotational speeds. At one speed the most efficient point may lie close to the surge area, while at another speed it may lie closer to the choke area, but in general the most efficient point is somewhere close to the middle.

Figure 1 shows a constructed example of a Q/H diagram for a compressor. Every curve represent a rotational speed, in this case the rotational speeds n_1 and n_2, where $n_1 > n_2$. The lines are based on what is called a basis line with a negative slope, indicating the radial speed component at the outlet increases with increasing flow. The Euler equation can be used to prove this. See Ambjørnsen [1] for a derivation.

The surge effect is a result of low flow through the compressor. The compressor might experience backflow, that may damage the compressor severely. Backflow means that the gas starts to flow the opposite way due to too small pressure produced by the compressor. Choke happens when the flow is too high, that may cause supersonic effects. Choke may also damage the compressor. The compressor will be run so that this is avoided in practice. Looking at figure 1 the optimal condition of a compressor would be to run close to the dotted line. This is where the efficiency is highest and the risk of machinery failure is lowest.

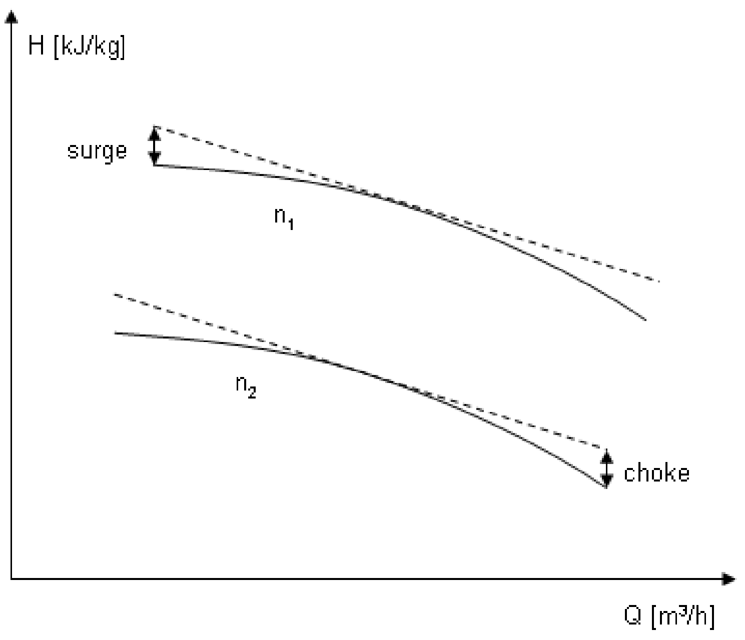


Figure 1: A constructed H/Q diagram for the rotational speeds n_1 and n_2.

2.2 Dresser-Rand Diagrams

As stated, the goal in this thesis is to propose a model, that can be used in condition analysis of the compressors. The model is based on the reference data of H and Q with various explanatory variables supplied by Dresser-Rand. Figure 2 shows all curves plotted like in Figure 1, with Q on the x-axis and H on the y-axis. The curves are from the recompression process with molweights 39.9 in blue, 44.3 in green and 47.5 in red. The different rotational speeds are 6033, 6895, 7757, 8619, 9050, where the curves for the rotational speed 6033 are clustered at the bottom left part of the diagram, and increasing rotational speed moves the cluster right and upwards, to the highest rotational speed, 9050, at the top right part of the diagram. The curves below the Q/H chart are the efficiency curves to corresponding Q/H curve, [4].

The dataset consists of 75 measurements for H and Q values, along with 75 values for the efficiency at each datapoint. These measurements are made for different running conditions as shown in the curves in Figure 2 and specified further here. There are as said three different molweights, and for each molweight there are five rotational speeds. For every pair of rotational speed and molweight, five measurements are collected. Hence, each curve in Figure 2 are made of five datapoints. In total this gives $5 \cdot 5 \cdot 3$ data values of each Q and H. The five measurements are collected by adjusting the in- and outlet pressures, thus naturally obtaining different Q and H values.

Under testing the end members of the curves are close to the surge and choke domains, while typically the middle values are the ones with the greatest efficiency, and thus sought after when running the compressors. This is however not always the case, sometimes the measurements with the highest efficiency lie close to the surge and choke domains, but in general the most efficient parts are close to the middle of the curves.

Dresser-Rand gives the H, Q and efficiency for each molweight in different diagrams. Ambjørnsen [1] provides an extensive presentation of how the data is aquired, and relates this to the compressor theory underlying the diagrams.

As in all data analysis, it is important to examine the data before starting the statistical modeling. Examining the different diagrams, there is a clear connection within each. Increasing rotational speeds indicate higher H and Q values, and increasing molweights increases at least flow, and in most cases also head. The curves seem almost parallell, but with different lengths.

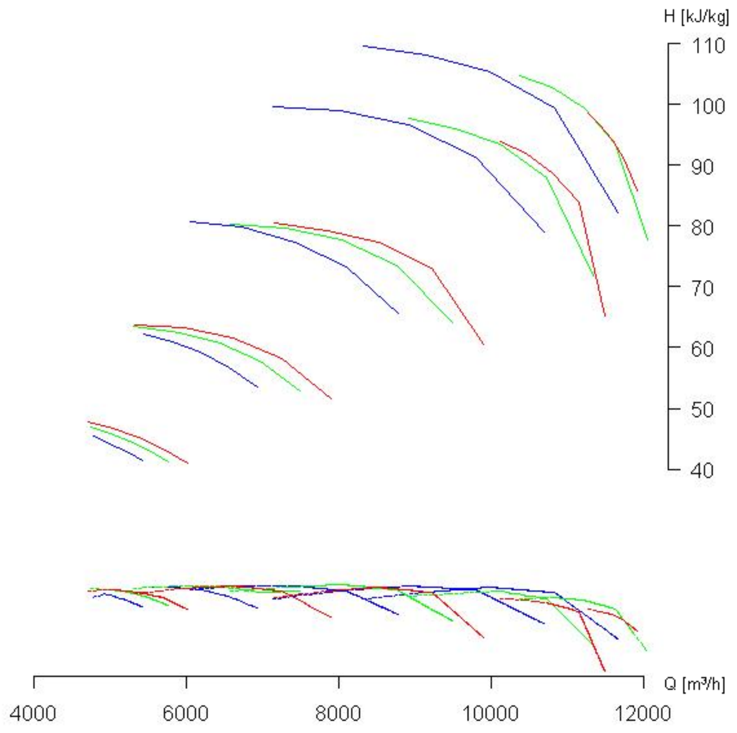


Figure 2: Dresser-Rand diagrams, above curves show H vs. Q, below are the efficiency versus flow curves.

To be able to use the data we need a sensible notation. Let

- $i \in \{1, 2, \dots, 5\}$ be the index for increasing rotational speed n_i, such that $n_1 = 6033$ rpm and $n_5 = 9050$ rpm,
- $j \in \{1, 2, 3\}$ be the index for the molweights, where $m_1 = 39.9$, $m_2 = 44.3$ and $m_3 = 47.5$,
- $I \in \{-1, 0, 1\}$ be an indicator function used to identify the the datapoint with the highest efficiency at each curve. See Figure 8 and explanation below,
- $l \in \{1, 2, \dots, 5\}$ denote the datapoint of each curve. $l = 1$ is the datapoint with the lowest flow, and $l = 5$ is the datapoint with the highest flow,
- $L \in \{1, 2, \dots, 5\}$ denote the datapoint with the highest efficiency on each curve.

To analyse the data, Q and H are plotted as a function of the order of the data, see figure 3 and 4 respectively. The datapoints are ordered with increasing rotational speeds and increasing molweight. The 25 first measurements are for molweight 39.9, the next 25 are for molweight 44.3 and the last 25 measurements are for molweight 47.5. Similarly, the first 5 measurements are for the lowest rotational speed, 6033. Measurements 6-10 are for the rotational speed 6895 and so on to the the measurements 21-25 for rotational speed 9050. Then for measurements 26-30 where the data for the molweight 44.3 begins again, the rotational speed 6033 begins.

A clear trend in the data is seen. The pattern for both H, figure 4 and Q, figure 3, looks like a zig-zag, with a period of five datapoints for every step in the zig-zag. When plot gets to the next molweight the zig-zag again start at the bottom and works its way up in the same manner with periods of 5 datapoints. The zig-zag pattern is a lot clearer for H than for Q. Looking at the rightmost datapoints for Q the zig-zag smoothens out. Looking at the peaks for the zig-zags, for H the peaks decrease with bigger molweight in a seemingly linear manner, while for Q the peaks increase the bigger molweight, but not equally evident.

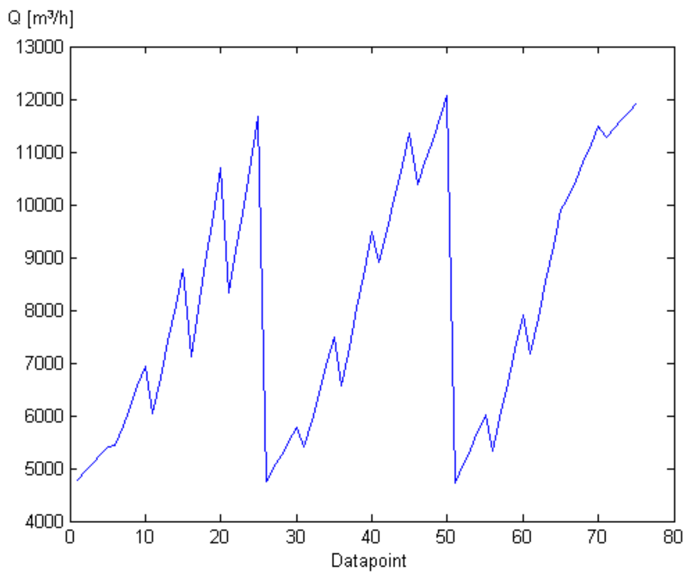


Figure 3: All datapoints plotted versus their Q-value, in the order they are arranged in the dataset.

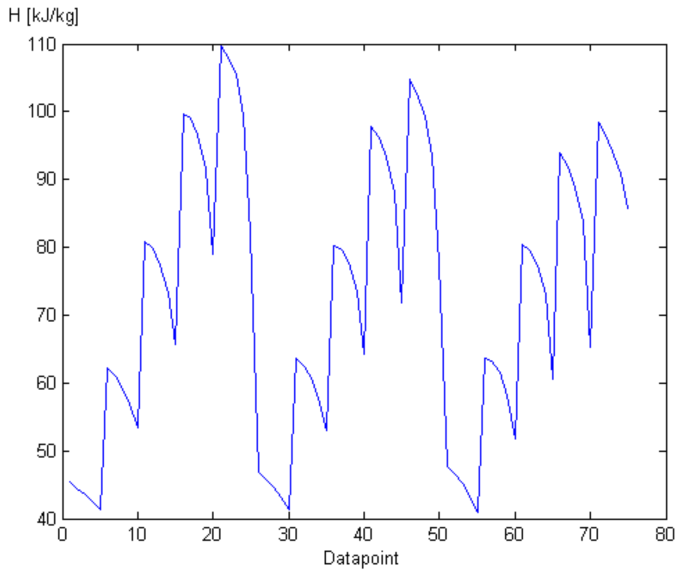


Figure 4: All datapoints plotted versus their H-value, in the order they are arranged in the dataset.

Next, both Q and H are plotted as a function of the different explanatory variables, molweight, rotational speed and efficiency.

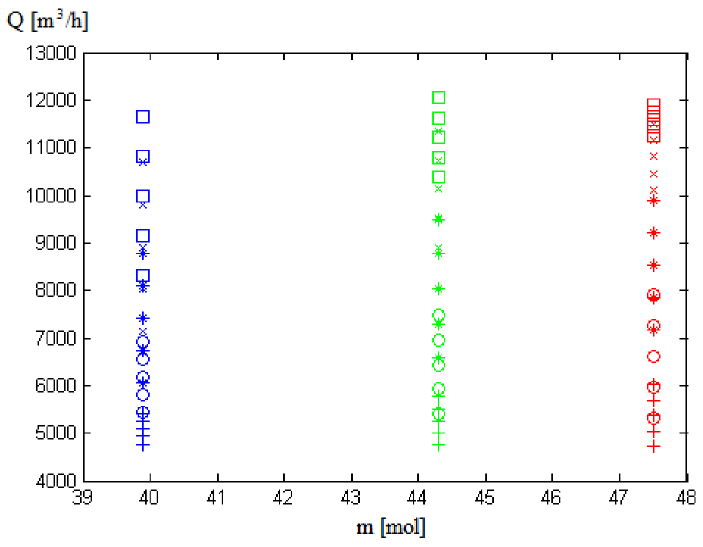
In Figure 5 Q and H is plotted versus molweight. A color and figure indexing is used. Blue color for the lowest molweight, 33.9, green color for 44.3 and red color for molweight 47.5. The different symbols are for the different rotational speeds. The circle is for the lowest rotational speed, 6033, the asterisk is for 6895, the cross for 7757, the square for 8619 and the plus-sign for 9050, see table 1 for a legend. Looking at the crossplot between molweight and H first, it seems that every curve is clustered.

It is also clear that there is a linear dependence within each rotational speed. For high rotational speeds the tendency is that datapoints decrease with higher molweight while for smaller rotational speeds the points seem to follow a horizontal line. It is the exact same tendency for small rotational speeds when Q is plotted versus molweight, while for higher rotational speeds the values increase the higher molweight. The main point is that there is a linear dependence for each rotational speed.

Looking at Figure 6 the different molweights are spread out more than the rotational speed were in the crossplot between molweight and H and Q. However, a general increasing trend is a lot more evident in these two plots than in Figure 5.

Table 1: Legend for the different datapoints.

Rotational speed	Molweight		
	33.9	44.3	47.5
6033	○	○	○
6895	∗	∗	∗
7757	×	×	×
8619	□	□	□
9050	+	+	+



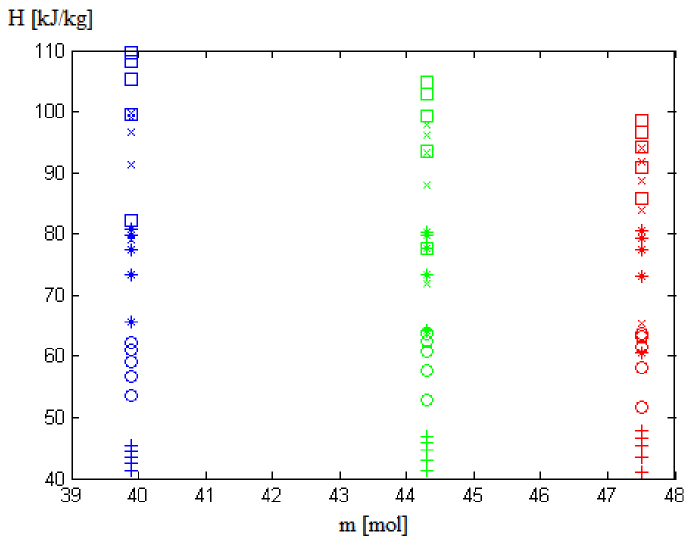
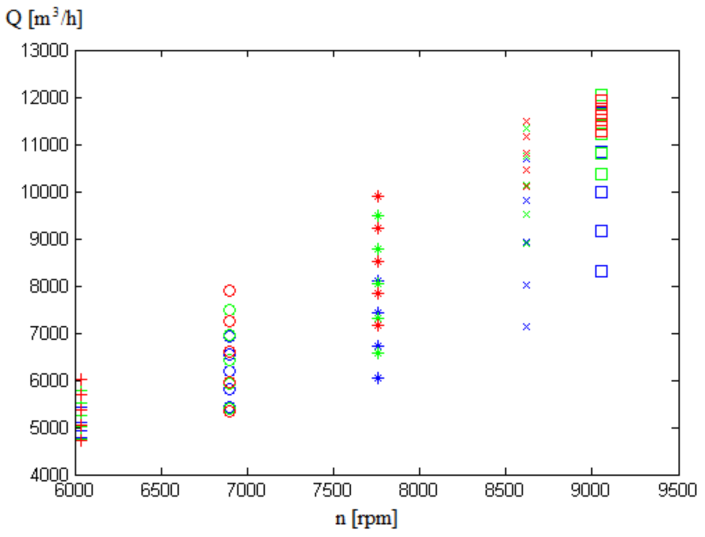


Figure 5: Molweight plotted versus Head and Flow.



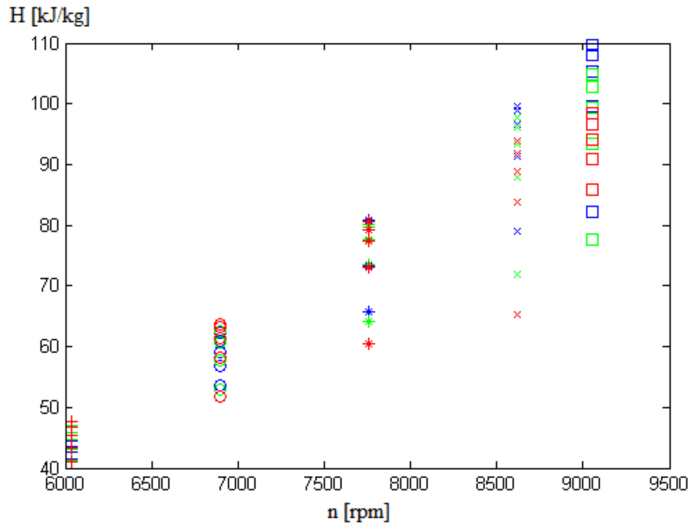


Figure 6: Rotational Speed plotted versus Head and Flow.

In figure 7 the effects are plotted versus H and Q. For both H and Q the datapoints form a cluster, or a horseshoe, with the high and low rotational speeds acting as the tips of the shoe. Based on these plots an indicator function is made,

$$I_{ijl} = \begin{cases} -1 & \text{if} \quad Q_{ijl} < Q_{ijL} \\ 0 & \text{if} \quad Q_{ijl} = Q_{ijL} \\ 1 & \text{if} \quad Q_{ijl} > Q_{ijL} \end{cases} \tag{3}$$

where the suffix L indicates the datapoint with the largest effect on the curve i, j. See figure 8 for a graphical representation of the indicator function. As seen in the figure the general trend is that medium rotational speeds have a more clustered effect whilst the smaller and larger rotational speeds are more spread out across the curves.

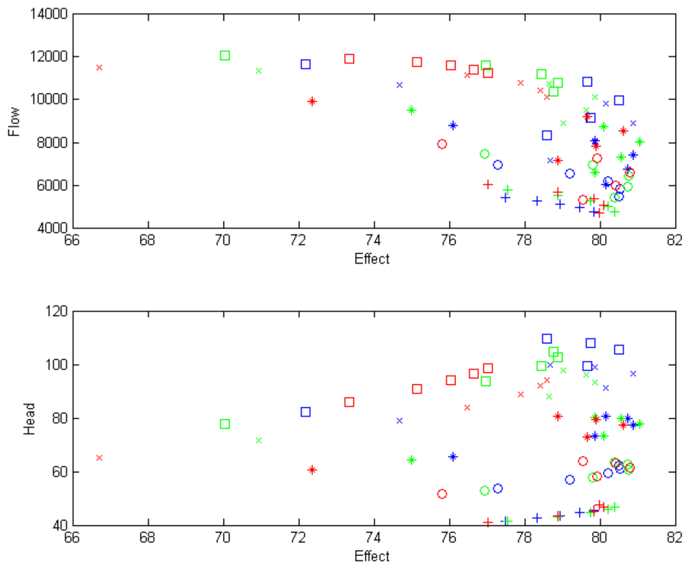


Figure 7: Effect plotted versus Head and Flow.

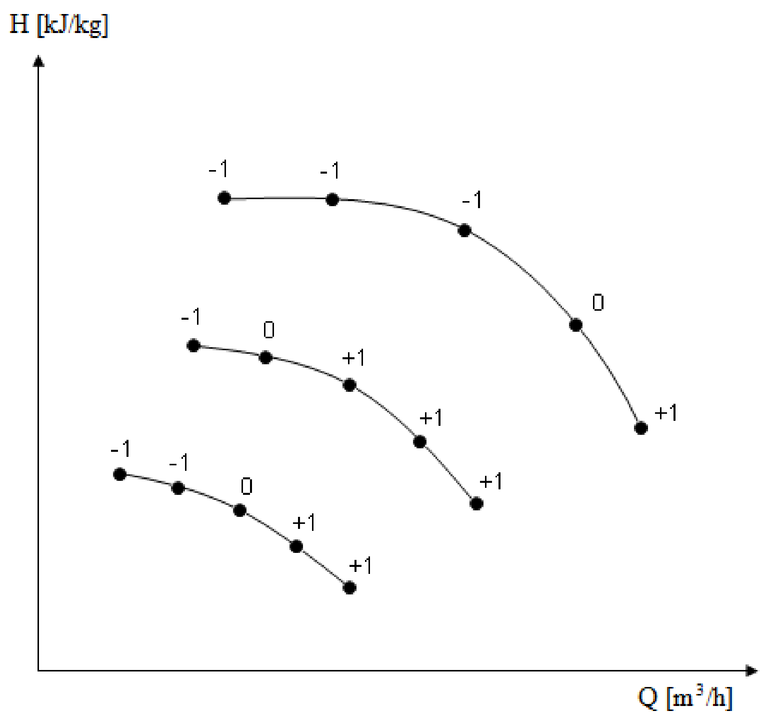


Figure 8: Figure explains how the indicator function applies on the curves.

In previous thesis and papers, such as Herdahl [7] and Ambjørnsen [1], a reference curve has been used when modelling the data. A reference curve gives the data for one of the most common conditions. The models are based in this reference curve. The problem with modelling the data in this manner is that with different datasets, the reference curve will change. In [7] a reference curve was used when making a model. The data were parameterized with angles and distances from the reference curve, see Figures 9 and 10. A problem with this model were that datapoints very close to the reference curve got the worst predictions due to the fact that the angles obviously varied a lot more closer to the reference curve than further away. Looking at Figure 9, especially the datapoints 25 and 70 for the angles are big outliers. These outliers are a result of the parametrization made as a basis from a reference curve. These bad results around the reference curve is a big drawback, as good predictions around the reference curve is desireable since this is the condition the compressors are mainly run at.

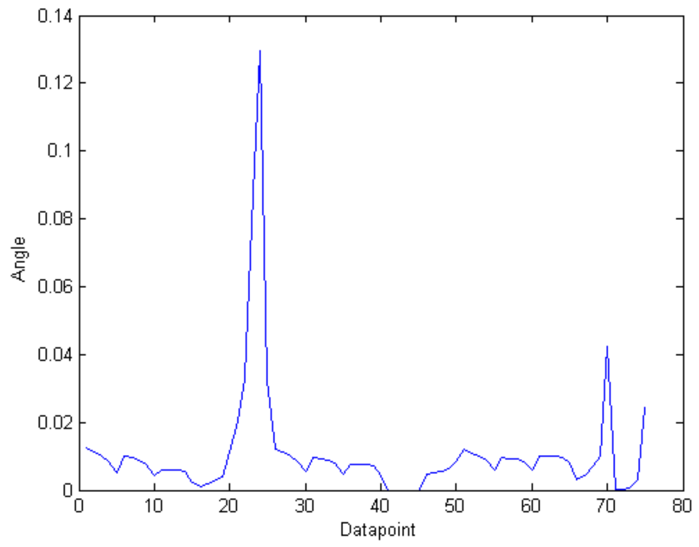


Figure 9: Angles from reference curve to every other curve.

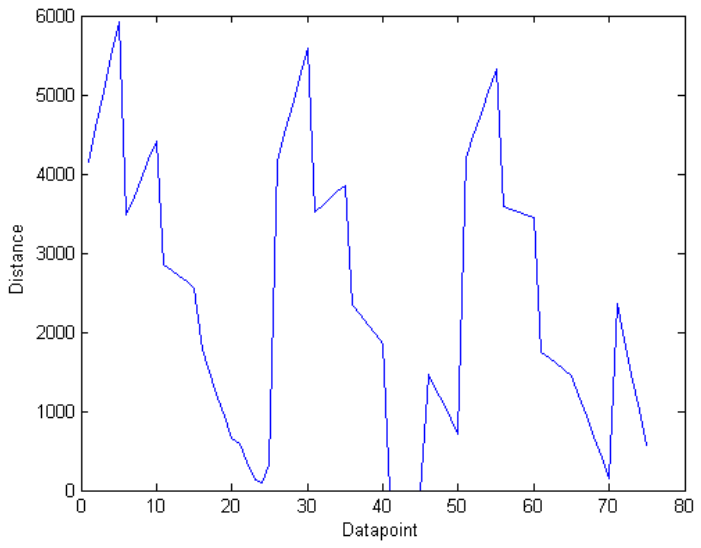


Figure 10: Distances from reference curve to every other curve.

3 Statistical Model and Theory

This chapter is divided into 6 parts. First, an introduction to the linear mixed model is given along with an example as a motivation for the use of linear mixed model for the Q/H dataset. The next section will show how the linear mixed model will be used on the Q/H dataset. Further, cross validation and its use on the Q/H dataset will be introduced and explained. Finally, there will be two chapters on model choice.

3.1 Linear Mixed Model

Linear Mixed Models, LMM, are widely used to describe clustered data. This model allows for correlations among repeated measurements made on individual clusters by incorporating appropriate random effects. The model is commonly expressed in hierarchial form, see Laird and Ware [8], Dobsen [11] and Gelfand [5], for $i = 1, \ldots, m$

$$\begin{aligned} &\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \mathbf{Z}_i\mathbf{b}_i + \boldsymbol{\varepsilon}_i, \\ &\mathbf{b}_i \sim \mathbf{N}_q(\mathbf{0}, \boldsymbol{\Psi}), \\ &\boldsymbol{\varepsilon}_i \sim \mathbf{N}_{n_i}(\mathbf{0}, \boldsymbol{\sigma}^2\boldsymbol{\Lambda}_i). \end{aligned} \tag{4}$$

where

- $\mathbf{y}_i$ is the $n_i \times 1$ response vector for observations in the ith cluster,
- $\mathbf{X}_i$ is the $n_i \times p$ model matrix for the fixed effects for observations in cluster i,
- $\boldsymbol{\beta}$ is the $p \times 1$ vector of the fixed-effect coefficients,
- $\mathbf{Z}_i$ is the $n_i \times q$ model matrix for the random effects for observations in cluster i,
- $\mathbf{b}_i$ is the $q \times 1$ vector of random-effect coefficients for cluster i,
- $\boldsymbol{\varepsilon}_i$ is the $n_i \times 1$ vector of errors for observations in cluster i,
- $\boldsymbol{\Psi}$ is the diagonal $q \times q$ covariance matrix for the random effects,
- $\boldsymbol{\sigma}^2\boldsymbol{\Lambda}_i$ is the diagonal $n_i \times n_i$ covariance matrix for the errors in cluster i.

Equally in form of equations for each measurement,

$$\begin{aligned} y_{ij} &= \beta_1 x_{1ij} + \cdots + \beta_p x_{pij}, \\ &\quad + b_{i1} z_{1ij} + \cdots + b_{iq} z_{qij} + \varepsilon_{ij}, \\ b_{ik} &\sim \mathbf{N}(0, \psi_k^2), \quad k = 1, \ldots, q, \\ \varepsilon_{ij} &\sim \mathbf{N}(0, \boldsymbol{\sigma}^2\lambda_{ij}), \quad 1, \ldots, n_i. \end{aligned} \tag{5}$$

Figure 11 shows a DAG for the linear mixed model. An equivalent formulation

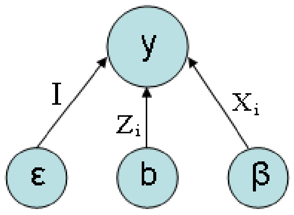


Figure 11: DAG for the linear mixed model, data y are modelled by fixed effects β, random effects b and noise term ε

that explicitly shows the regression as a model of conditional expectation can be given as,

$$[\mathbf{y_i}|\mathbf{b_i}] \sim \mathbf{N}(\mathbf{Z_i}\boldsymbol{\beta} + \mathbf{Z_i}\mathbf{b_i}, \boldsymbol{\sigma}^2\boldsymbol{\Lambda}_\mathbf{i}). \tag{6}$$

And marginally,

$$\mathbf{y_i} \sim \mathbf{N}(\mathbf{X_i}\boldsymbol{\beta}, \mathbf{Z_i}\boldsymbol{\Psi}\mathbf{Z_i^T} + \boldsymbol{\sigma}^2\boldsymbol{\Lambda}_\mathbf{i}). \tag{7}$$

$$\begin{aligned} E[\mathbf{y_i}] &= \mathbf{X_i}\boldsymbol{\beta} \\ Var[\mathbf{y_i}] &= \mathbf{Z_i}\boldsymbol{\Psi}\mathbf{Z_i^T} + \boldsymbol{\sigma}^2\boldsymbol{\Lambda}_\mathbf{i}. \end{aligned} \tag{8}$$

An example of an LMM given by Bryk and Raudenbuch [2], gives the results of a math-achievement test for different students. The information available on the different student is which school the student goes to, the socioeconomic status of the families, the student's school's average achievement on the math test and if the school is public or catholic.

By examining scatterplots and boxplots between math achievement and socioeconomic status for public schools and catholic schools, the trend is that there is a stronger positive relationship between socioeconomic status and math achievement in the public schools than in catholic schools. However, the average level of math achievement in catholic schools are greater than in public schools.

This data analysis give the background to propose a LMM. Using the centered socioeconomic status, *cses* for the schools, the individual-level equation for student j in school i is,

$$\text{mathach}_{ij} = \alpha_{0i} + \alpha_{1i}\text{cses}_{ij} + \epsilon_{ij}. \tag{9}$$

Further, at the school level, the intercepts and slopes will depend upon catholic and public schools and average level of socioeconomic status in the schools,

$$\alpha_{0i} = \gamma_{00} + \gamma_{01}\text{meanses}_i + \gamma_{02}\text{sector}_i + u_{0i} \tag{10}$$

$$\alpha_{1i} = \gamma_{10} + \gamma_{11}\text{meanses}_i + \gamma_{12}\text{sector}_i + u_{1i} \tag{11}$$

Substituting the school-level equation into (9) and rearranging terms gives an equation consisting of γ's, u's and ϵ. The γ's will then be the fixed effects, while the u's and ϵ will be the random effects. A more detailed explanation of this example can be obtained reading Bryk and Raudenbuch [2] and Laird and Ware [8].

The usual ways to estimate the random effects are by MLE , or Restricted Maximum Likelihood, REML. Usual MLE may be biased as it estimates MLE simultaniously. This is not a problem where the variance of the data y does not depend on the fixed effects β. In (7), we obtain parameter estimates by least squares,

$$\hat{\beta} = [\mathbf{X^T R^{-1} X}]^{-1} \mathbf{X^T R^{-1} y}. \tag{12}$$

$\mathbf{X}$ is built up of x_i's and $\mathbf{R} = \sigma^{\mathbf{2}}$. The problems by using least squares is that we will get too few degrees of freedom and bad predictions, if the dimension of the dataset is close to the dimension of β. The solutions does not take advantage of cluster effects, i.e. variations across and inwards clusters. This is where LMM expands the methods which take the cluster effects into consideration, and introduces the b_i's as structured random effects.

REML filters away the fixed effects β_i before the parameters in the covariance matrix are estimated,

$$\tilde{\mathbf{y}} = [\mathbf{I} - \mathbf{X}(\mathbf{X^T X})^{-1}\mathbf{X^T}]\mathbf{y} = \mathbf{By}. \tag{13}$$

$$E[\tilde{\mathbf{y}}] = \mathbf{0}. \tag{14}$$

$$Var[\tilde{\mathbf{y}}] = \mathbf{B}[\mathbf{Z\Psi Z^T} + \boldsymbol{\sigma}^{\mathbf{2}}\mathbf{\Lambda_i}]\mathbf{B^T}. \tag{15}$$

The expression is independent of the fixed effects β_i's. Now, the likelihood for $\tilde{y}$ can be optimated with respect to $\sigma^{\mathbf{2}}\mathbf{\Lambda_i}$ and $\mathbf{\Psi}$, see Wood [3].

3.2 Linear Mixed Model for Head and Flow

An assumption for the model is that the data Q and H can be modelled seprately. Set

$$y = \begin{cases} Q, & \text{if considering flow} \\ H, & \text{if considering head} \end{cases}$$

for the sake of simplicity. The dataset y will be modelled with an LMM. As explained in the previous chapter this means that y will be modelled with three additive parts,

- Fixed effects i in a multiple regression model $\boldsymbol{\beta}$, $\quad \boldsymbol{\beta} = (\beta_1 \ldots \beta_p)'$,
- Random effects i in a structured model $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_q$, $\quad \boldsymbol{b}_i = (b_{i1} \ldots b_{iq})'$,

- Residual effect $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_m$.

such that

$$\boldsymbol{y}_i = X_i\boldsymbol{\beta} + Z_i\boldsymbol{b}_i + \boldsymbol{\varepsilon}_i, \quad i = 1, \ldots, m, \quad \boldsymbol{y}_i = (y_{i1} \ldots y_{in})'. \tag{16}$$

In this thesis, the same model has been used for both Q and H data. Later in chapter 3.5 different models are discussed. For the basis model the variables used are for the **X**-matrix all m,n and I. The **X**-matrix will look like this,

$$\mathbf{X} = \begin{bmatrix} 1 & n_{111} & m_{111} & \mathrm{I}_{111} \\ 1 & n_{112} & m_{112} & \mathrm{I}_{112} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & n_{534} & m_{534} & \mathrm{I}_{534} \\ 1 & n_{535} & m_{535} & \mathrm{I}_{535} \end{bmatrix} \tag{17}$$

Here the p from chapter 3.1 is 4. Similarly the variable used to make the **Z**-matrix in the basis model is only I. Hence q will be of size 2,

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 & \mathrm{I}_{111} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & & & \vdots & \vdots & \vdots & & & \vdots \\ 1 & 0 & & & & \mathrm{I}_{115} & 0 & & & \\ 0 & 1 & & & & 0 & \mathrm{I}_{121} & & & \\ \vdots & \vdots & & & & \vdots & \vdots & & & \\ 0 & 1 & \ddots & & & 0 & \mathrm{I}_{125} & \ddots & & \\ \vdots & 0 & & & \vdots & \vdots & \vdots & & & \vdots \\ & \vdots & & & 0 & & \vdots & & & 0 \\ & & & & 1 & & & & & \mathrm{I}_{531} \\ \vdots & & & \vdots & \vdots & \vdots & & & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & 0 & \cdots & \cdots & 0 & \mathrm{I}_{535} \end{bmatrix} \tag{18}$$

The model will be presented as model 2 in the results chapter.

The fixed effects $\boldsymbol{\beta}$ are common for all curves, while the matrix of covariates X_i may vary between the curves. The fixed part of the model is a regular multiple regression model,

$$\boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_m \end{pmatrix} = \begin{bmatrix} \mathbf{X_1} \\ \vdots \\ \mathbf{X_m} \end{bmatrix} \cdot \boldsymbol{\beta} + \boldsymbol{v} \quad = \quad X \cdot \boldsymbol{\beta} + \boldsymbol{v}. \tag{19}$$

Usually the $\boldsymbol{v}$'s are independent and identically distributed. In LMM they have a structure because of Z, and can therefore have different variances and correlations. Assume $\boldsymbol{v}$ independent and $v_j = N(0, \sigma^2), \quad j = 1, \ldots, n \cdot m$. Then,

$$\hat{\boldsymbol{\beta}} = [\mathbf{X^T X}]^{-1} \cdot \mathbf{X^T} \cdot \mathbf{y}, \quad \hat{\mathbf{y}} = \mathbf{X} \cdot \hat{\boldsymbol{\beta}}$$
$$\hat{\sigma}^2 = \frac{1}{nm-1} \sum_{j=1}^{nm} (y_j - \hat{y}_j)^2.$$

The random effects are different from every curve, $i = 1, \ldots, m$,

$$\boldsymbol{b}_i \sim N\left(\mathbf{0}, \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_q^2 \end{bmatrix}\right). \tag{20}$$

A priori the residual terms are independently distributed,

$$e_j \sim N(0, \sigma_{q+1}^2). \tag{21}$$

In an LMM the variance parameters have to be estimated,

$$\hat{\sigma}_1^2, \ldots, \hat{\sigma}_{q+1} \tag{22}$$

as well as $\hat{\beta}$. As $\hat{\mathbf{u}}$ constitute in the structured part of the model,

$$\hat{\boldsymbol{u}}_i = E[\hat{\boldsymbol{u}}_i | y]. \tag{23}$$

This will give the predicted data

$$\hat{\boldsymbol{y}}_i = \mathbf{X_i}\hat{\boldsymbol{\beta}} + \mathbf{Z_i}\hat{\mathbf{u}}_\mathbf{i}. \tag{24}$$

The variance of the prediction can also be obtained by using

$$Var[\boldsymbol{u}_i | y]. \tag{25}$$

which can be used to obtain uncertainity or to draw realizations with the correct amount of variablity.

3.3 Cross Validation

The next two subchapters will describe cross validation in general and for the Q/H dataset. There will be a comparison between the different models tried in subchapter, and an explanation for the choice of model with references to the dataset in the second subchapter.

Consider a dataset that can be represented by a function. It is desired to find a function that is closest to the "truth" by doing a regression analysis on the dataset to predict new data. The accuracy of a regression analysis is restricted by the size of the dataset. Obviously, the less data we have available, the poorer regression model we get.

As the dataset gets smaller, each datapoint will affect the regression model at a larger weight than with bigger datasets. Hence, any datapoint that is skewed can cause future data to differ more from the model than if these skewed datapoints were not weighted in the regression analysis. This is where cross validation is useful. Cross validation is a method that is better than the common residual analysis, as cross validation will give an indication of how future datapoints will fit the model. The idea is to choose a datapoint or a set of datapoints, and use the remaining dataset as a training set. Then we use our regression on the training set and estimate the future performance with the test set. We then measure the mean absolute test set error, which we use to evaluate the model. This type of cross validation is called the hold out method or the test set method.

This method can be expanded into the k-fold method where one choose k test subsets and repeat the test set method k times, where the kth subset is used as a test set and the other $k-1$ are used as the training set. The advantage of this method is that each data point is guaranteed to be in a test set once and in a training set $k-1$ times, hence the variance of the estimate will be smaller than with a single test set method. The downside with this method is that its algorithm will cost more.

Lastly the Leave One Out Cross Validation is just what it says it is. It is a k-fold method where k equals n, where n is the number of datapoints. It means the regression is done on the whole data set except that single point, and predictions are made for that single point not in the training set.

Cross validation can also be used to detect outliers or anomalous parts of a dataset. If the leave-n-out prediction intervals for a block of size n misses most of the data, the model is probably not valid for that block.

3.4 Cross Validation on the Head and Flow Dataset

Typically, a smart way to use cross validation on the Q/H dataset is to remove single curves from the dataset,

$$\underbrace{\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_k \\ \vdots \\ \boldsymbol{y}_m \end{bmatrix}}_{old} \qquad \underbrace{\boldsymbol{y}_{-k} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_{k-1} \\ \boldsymbol{y}_{k+1} \\ \vdots \\ \boldsymbol{y}_m \end{bmatrix}}_{new}$$

where $\boldsymbol{y}_k$ is removed. The new dimension on the vector, $\boldsymbol{y}_{-k}$, where $-k$ denotes that the kth kurve is removed, will be $n \cdot m - n$. When $\boldsymbol{y}$ is altered, $\mathbf{X}$ and $\mathbf{Z}$-matrices must also be altered,

$$\underbrace{\mathbf{X} = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_m \end{bmatrix}}_{old} \qquad \underbrace{\mathbf{X}_{-k} = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{k-1} \\ \boldsymbol{x}_{k+1} \\ \vdots \\ \boldsymbol{x}_m \end{bmatrix}}_{new},$$

$$\underbrace{\mathbf{Z} = \begin{bmatrix} z_1 & 0 & \\ 0 & \ddots & 0 \\ & 0 & z_m \end{bmatrix}}_{old} \qquad \underbrace{\mathbf{Z}_{-k} = \begin{bmatrix} z_1 & 0 & & & & \\ 0 & \ddots & 0 & & & \\ & 0 & z_{k-1} & 0 & & \\ & & 0 & z_{k+1} & 0 & \\ & & & 0 & \ddots & 0 \\ & & & & 0 & z_m \end{bmatrix}}_{new}.$$

$\boldsymbol{u}$ also needs to be reduced as both rows and columns in Z are changed,

$$\underbrace{\boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_m \end{bmatrix}}_{old} \qquad \underbrace{\boldsymbol{u}_{-k} = \begin{bmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_{k-1} \\ \boldsymbol{u}_{k+1} \\ \vdots \\ \boldsymbol{u}_m \end{bmatrix}}_{new}.$$

This can be done in a sequential manner, starting by removing the first curve ($k = 1$), then the second curve ($k = 2$), and so on until ($k = m$). For each curve the leave-k-out prediction $\hat{y}_k$ with the obsverved left out data y_k.

3.5 Model Choice

There are two critera for model choice. First, it is desired that the model is as accurate as possible. A model that fits the given data is intuitively a good model. However, if the model fits the given data too well there might be a chance that the data have been overparametrized. As a result of this overparamterization new data will in many cases fit the model poorly.

A variable to measure the accuracy of the model is the likelihood of the model. In Harville [6] the likelihood is given

$$\begin{aligned} L(\boldsymbol{\sigma};\mathbf{y}) &= -1/2\ln|V_\sigma| - 1/2\ln|\mathbf{X}'\mathbf{V}_\sigma^{-1}\mathbf{X}| \\ &= -1/2(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}})'\mathbf{V}_\sigma^{-1}(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}) \\ \\ \hat{\boldsymbol{\beta}} &= [\mathbf{X}'\mathbf{V}_\sigma^{-1}\mathbf{X}]^{-1}\mathbf{X}'\mathbf{V}_\sigma^{-1}\mathbf{y}. \end{aligned} \tag{26}$$

Similar algebraical expressions exist. The inverse of a matrix may be written in other terms, such that $\hat{\boldsymbol{\beta}}$ also will be written in other terms. In Laird and Ware [8] the likelihood for the LMM is written,

$$L = -\frac{((nm - \mathrm{rank}(\mathbf{X}'\mathbf{X}))\cdot\log(2\pi s_{r+1}) - \log(\det(\mathbf{T})) + (nm - \mathrm{rank}(\mathbf{X}'\mathbf{X}))}{2}, \tag{27}$$

where $n \times m$ is the number of data used in the model, s_{r+1} is the last element in the vector of prior choice of variance components and $\mathbf{T}$ is,

$$\mathbf{T} = \left[\frac{\mathbf{I}_{a\times a} + \mathbf{O}\cdot\mathbf{D}}{s}\right]^{-1},$$

where I is the identity matrix of dimension $a \times a$. a is given by the number of columns in the $\mathbf{X}'\mathbf{Z}$-matrix which varies between different models. $\mathbf{O}$ is given by

$$\mathbf{O} = \mathbf{Z}'\mathbf{Z} - (\mathbf{X}'\mathbf{Z})' \cdot (\mathbf{X}'\mathbf{X})^{+} \cdot \mathbf{X}'\mathbf{Z},$$

and $\mathbf{D}$ is given by

$$\mathbf{D} = \mathrm{diag}(s_1 \cdot \mathbf{r}),$$

where $\mathbf{r}$ is a diagonal matrix made up of the prior choice of variance components [8].

3.6 Model Choice for Head/Flow dataset

The basis model is given in chapter 1.2. When adding variables to the model columns are added to the $\mathbf{X}$-matrix. here $\mathbf{X}_{+c}$ denotes the new matrix when c new explanatory variables or product of these are added to the model.

$$\underbrace{\mathbf{X} = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_m \end{bmatrix}}_{old} \qquad \underbrace{\mathbf{X}_{+c} = \left[\begin{array}{c|ccc} \boldsymbol{x}_1 & \boldsymbol{v}_{11} & \dots & \boldsymbol{v}_{c1} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{x}_m & \boldsymbol{v}_{1m} & \dots & \boldsymbol{v}_{cm} \end{array}\right]}_{new},$$

The $\boldsymbol{v}$s are vectors of dimension $(\frac{m}{k})$. The $\mathbf{Z}$-matrix also needs to be expanded when new explanatory variables are added to the model. Again $\mathbf{Z}_{+c}$ denotes the new matrix when c new variables or products are added to the model.

$$\underbrace{\mathbf{Z} = \left[\begin{array}{ccc|ccc} \mathbf{1} & 0 & & \mathbf{I}_1 & 0 & \\ 0 & \ddots & 0 & 0 & \ddots & 0 \\ & 0 & \mathbf{1} & & 0 & \mathbf{I}_m \end{array}\right]}_{old}$$

$$\underbrace{\mathbf{Z}_{+c} = \left[\begin{array}{ccc|ccc|ccc|c|ccc} \mathbf{1} & 0 & & \mathbf{I}_1 & 0 & & \boldsymbol{w}_{11} & 0 & & & \boldsymbol{w}_{c1} & 0 & \\ 0 & \ddots & 0 & 0 & \ddots & 0 & 0 & \ddots & 0 & \dots & 0 & \ddots & 0 \\ & 0 & \mathbf{1} & & 0 & \mathbf{I}_m & & 0 & \boldsymbol{w}_{11} & & & 0 & \boldsymbol{w}_{cm} \end{array}\right]}_{new}$$

Again the $\boldsymbol{w}$s, $\mathbf{I}$s and $\mathbf{1}$ are vectors of dimension $(\frac{m}{k})$.

4 Results

This chapter is divided into 4 parts. The first subchapter gives an overview of the Graphical User Interface that has been made to represent the data, to choose a model and to get the results from the chosen model. The next subchapter gives various results from the chosen model. The cross validation subchapter give an overview over how the model fits when subtracting various curves from the dataset before estimating the parameters. The last subchapter gives alternative models.

4.1 Graphical User Interface for the Dataset

As a part of the thesis a Graphical User Interface, GUI, has been made to plot the data, analyze the data and to evaluate the different Linear Mixed Models. The GUI has been programmed using MATLAB's GUIDE add-on. Figure 12 shows a screenshot of how the GUI looks like after initializion in MATLAB. The GUI is launched simply writing **lmmgui_v2** in the MATLAB command line after loading appropriate files.

The GUI consists of a graphing area, checkboxes, pop-up menus and push buttons. The upper right panel called Model Choice is where the different explanatory variables for the linear mixed model for both Head and Flow are chosen. The idea is to click whichever explanatory variables are desired and press Estimate H to estimate the parameters of the model for Head. Similary, again choose which explanatory variables are desired for Flow and click Estimate Q. Once the Estimate-buttons are clicked the text-area to the corresponding Estimate-button will give a likelihood value for the model, along with numbers indicating which model has been chosen. The first 6 numbers indicates which of the fixed parameters has been chosen by the user, starting from top of the checkbox column to the bottom. The next 6 numbers indicates which of the random effects has been chosen by the user. The number 1 means that the corresponding effect is chosen, while the number 0 means it is not chosen in the model.

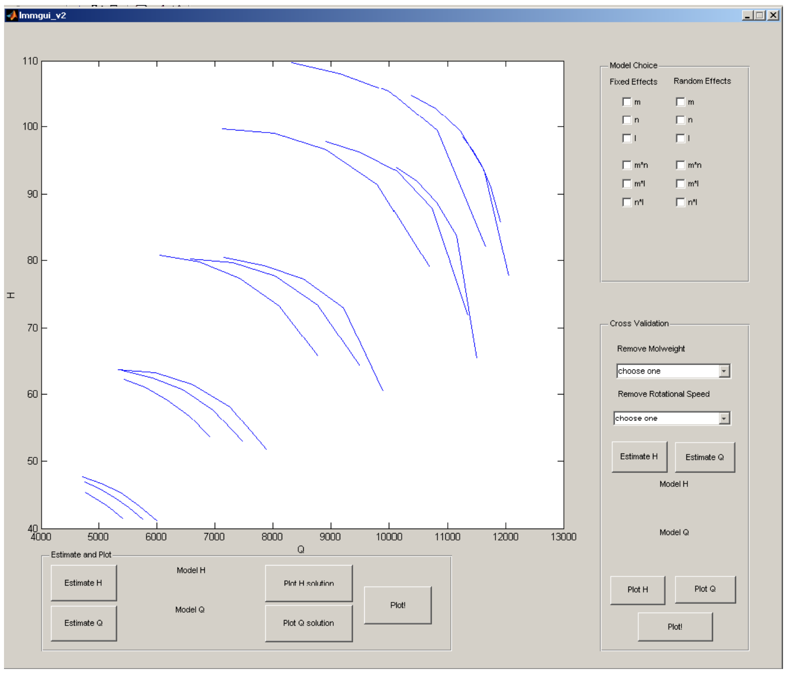
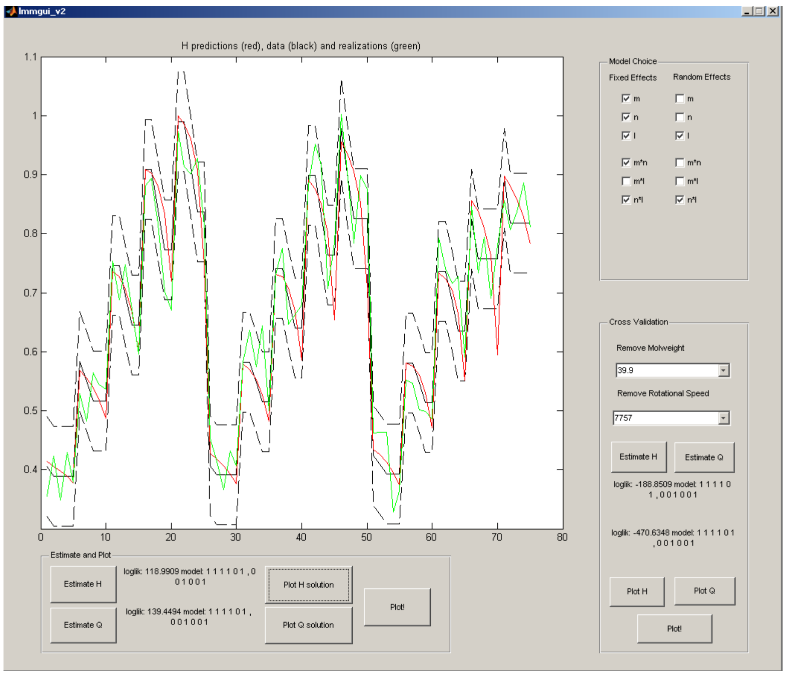


Figure 12: A screenshot of how the GUI looks like when it is first launched in MATLAB.

To look at how the model fit, click "Plot H" solution for the Head model, and "Plot Q" Solution for the Flow model. This will bring up a plot of the data together with predictions and realizations of the model, together with 95% confidence intervals. See figure 13 for a screenshot of this part of the GUI. Clicking several times on these two buttons will give new realizations.



Model Choice

Fixed Effects	Random Effects
✓ m	m
✓ n	n
✓ l	✓ l
✓ m*n	m*n
m*l	m*l
✓ n*l	✓ n*l

Remove Molweight: 39.9

Remove Rotational Speed: 7757

loglik: 118.9909 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

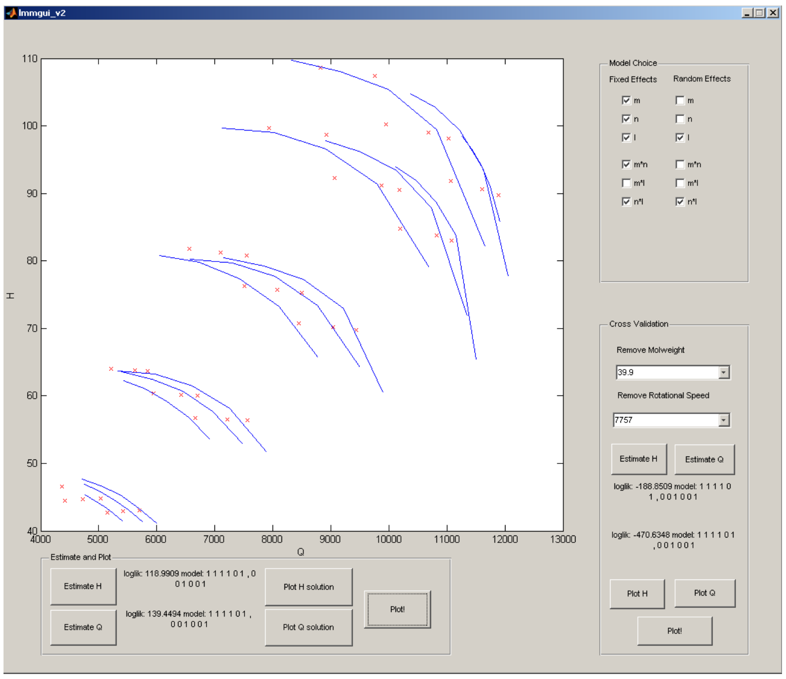
loglik: 139.4494 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

loglik: -188.8509 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

loglik: -470.6348 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

Figure 13: A screenshot of the GUI with the H predictions after estimating with some parameters and plotted.

To see how both the models for Head and Flow fit the original dataplot click “Plot!”. This will give a graph of the original data together with predictions from the models chosen. The panel Cross Validation is where the user can choose to remove a curve from the dataset when estimating the parameters for the model. Further, clicking “Plot H” and “Plot Q” will again give a plot of the predictions along with confidence intervals, realizations and the original data.



loglik: 118.9909 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

loglik: 139.4494 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

Model Choice

Fixed Effects	Random Effects
[✓ m]	m
[✓ n]	n
[✓ l]	[✓ l]
[✓ m*n]	m*n
m*l	m*l
[✓ n*l]	[✓ n*l]

Remove Molweight: 39.9

Remove Rotational Speed: 7757

loglik: -188.8509 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

loglik: -470.6348 model: 1 1 1 1 0 1 , 0 0 1 0 0 1

Figure 14: A screenshot of the GUI with the predicted data together with original data.

4.2 Results From the Chosen Model

In this chapter the results from the chosen model are presented. In figure 15 and 16 the data are plotted in red, the predictions are plotted in black and realizations are plotted in green for H and Q respectively.

Looking at figure 15 first, the model seems to fit quite well for most datapoints for the lowest molweight, datapoint 0 to 25. For the medium molweight, however, the model does not catch up with the outliers for some of the highest rotational speeds. The datapoints have generally a lower value than for the predictions. The same trend can be spotted for the highest molweight, where the data for the two highest rotational speeds have a much lower value than the predictions.

In figure 16 the same graph is displayed, but for Q this time. As discussed in chapter 2 the zig-zag pattern for Q is not as strong overall as it is for H. For the highest molweight, datapoint 50 to 75, the pattern is almost smoothened completely at some of the highest rotational speeds. This makes it harder to get good predictions as the pattern is not well defined. In general it seems that the right- and leftmost points of the curves have gotten the worst predictions. The model does not manage to match up with the very steep zig-zag for the two lower molweights, while it is too steep for the higher rotational speeds for the highest molweight.

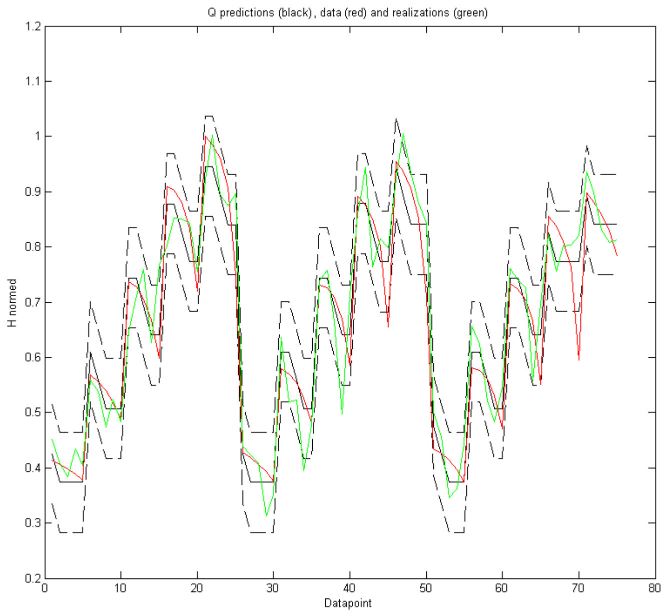


Figure 15: H predictions together with data, confidence intervals and realizations for model 2, basis model, with 3 fixed effects and 1 random effect for H.

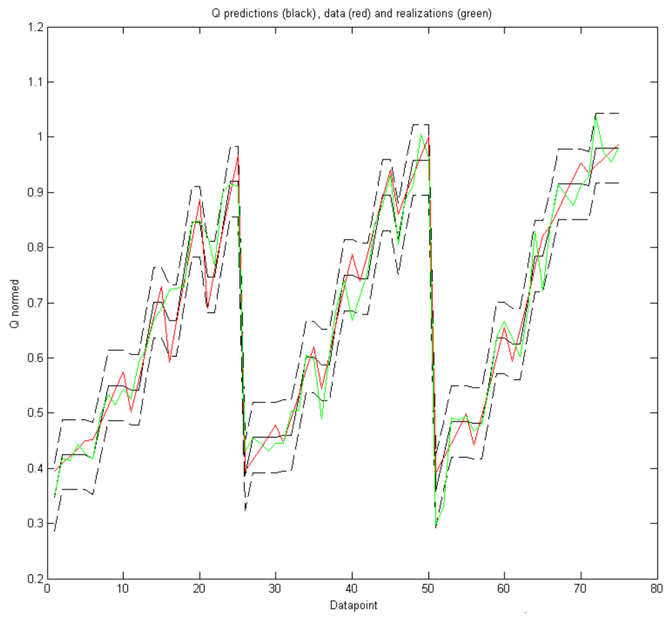


Figure 16: Q predictions together with data, confidence intervals and realization for model 2, basis model, with 3 fixed effects and 2 random effects for Q.

4.3 Cross Validation with Chosen Model

In table 2 the total error, (28), at each curve has been listed when that curve has been left out as the model's parameters has been calculated.

$$[H_{err_{ij}}, Q_{err_{ij}}] = \sum_{l=1}^{l=5} (\boldsymbol{y}_{-k_{ijl}} - \hat{\boldsymbol{y}}_{ijl})^2. \tag{28}$$

Here i, j indexes k as in chapter 3.4, i.e every curve k is indexed by rotational speed i and molweight j. Looking at table 2, it seems that when the error is big with the head predictions it is also big with flow. However, this is not always true. For Q predictions at molweight 33.9 and rotational speed 8619, curve $(4, 1)$ the error is a lot bigger than at the same molweight and rotational speed 9050. For H predictions the size of errors are opposite than for Q predictions at these two curves. The same thing can be seen at other curves, like curve $(1, 3)$ and $(2, 3)$, and curve $(1, 2)$ and $(2, 2)$.

Looking at curve $(4, 3)$ there is a very large difference between predicted curve and observed curve. Probably the data for this set of explanatory variables is not well represented by the model.

Table 2: Total error at each curve for model 2 using cross validation

Curve (i,j)	$H_{err_{ij}}$	$Q_{err_{ij}}$
(1,1)	2.41	7.28
(2,1)	11.32	6.95
(3,1)	8.90	8.32
(4,1)	19.97	28.12
(5,1)	57.24	14.58
(1,2)	6.54	5.42
(2,2)	8.76	4.50
(3,2)	10.10	10.26
(4,2)	38.61	12.34
(5,2)	61.22	11.98
(1,3)	9.99	7.27
(2,3)	6.26	8.10
(3,3)	20.46	7.82
(4,3)	112.15	10.78
(5,3)	14.66	3.19

4.4 Model Choice

In chapter 3.5 the criteria for model choice were discussed. The likelihood is calculated each model and compared. The calculation of the likelihood is integrated as a part of the GUI. It will appear in the text areas when the estimations of head and flow is done. In table 3 the likelihood is listed for different models. The general trend is that the more explanatory variables are put in the model, the greater the likelihood gets. Another thing to notice is that some of the variables affect the likelihood for head more than flow, and vice versa. The Indicator gives a greater impact on the likelihood for flow than for head, while m and n makes the likelihood greater for head than for flow. Another thing to notice is that the random effects affect the likelihood at a much greater rate when there are few fixed effects in the model opposed to the models with many fixed effects. Also, when using almost every explanatory variables and products the likelihood is a bit smaller than when removing some of the variables. It seems that to maximize the likelihood without using too many explanatory variables and products, the last model gives the highest likelihood for head, and the 2nd last model gives the highest likelihood for flow.

Below are predictions from three different models for both head and flow. Model 1 for both head and flow consist of only one fixed effect and random effect. In this model the molweight has been used as both the fixed and random effect. As seen in figure 17 and 19, the predictions are very poor. For both H and Q the model hardly match up with the zig-zag pattern. The model gives merely a stair-type prediction. It is obvious that this model is under-parametrized for both H and Q.

Looking at Figure 20 and 18, 6 different fixed effects and 6 different random effects have been used in model 2 for both H and Q. This model fits the data better than the simple model with only one random and one fixed effect. However, this model is probably overparametrized and future data might suffer.

Figure 16 and 15 gives the predictions for two models where fewer effects have been used. For both Q and H the fixed effects are m, n and I and the random effects are m and I. Looking at table 3 it is clear that these models for H and Q have near the same likelihoods as the models with 6 different fixed and random effects. The predictions showed in the two figures are also equally good. This gives an indication that model 3 for both H and Q is a better fit to the data when both the predictions and the amount of parametrization is taken into account.

Table 3: Likelihood for different models for Head and Flow

Fixed effects	Random effects	L_H	L_Q
m	m	70.01	64.45
m, n	m	96.69	86.32
m, I	m	88.01	110.54
m, n, I	m	118.09	141.18
n	m, n, I	107.33	117.13
I	m	88.00	110.26
n	m	97.31	81.78
n, I	I	119.13	137.76
m, n, I	m, I	118.10	141.69
m, n, I, m	m, I	116.30	140.52
$m, n, I, m\cdot, n \cdot I$	$I, n \cdot I$	119.00	139.44
$m, n, I, m \cdot n, n \cdot I, m \cdot I$	$I, n \cdot I$	116.78	138.13
m	$m, n, I, m \cdot n, n \cdot I, m \cdot I$	77.06	97.04
$m, I, m \cdot n$	$I, n \cdot I$	114.05	142.38
$m, n, I, n \cdot I$	$n, n \cdot I$	121.00	141.01

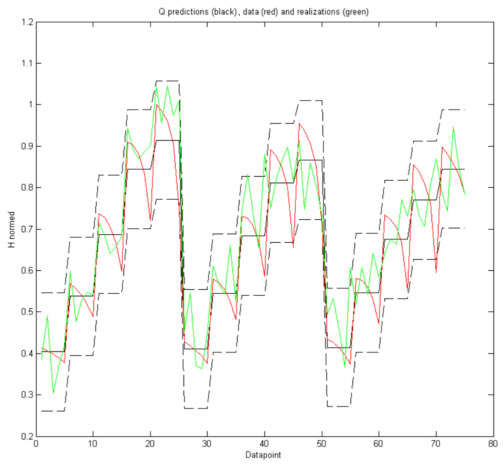


Figure 17: H predictions together with data, confidence intervals and realizations for model 1 with 1 fixed effect and 1 random effect for H.

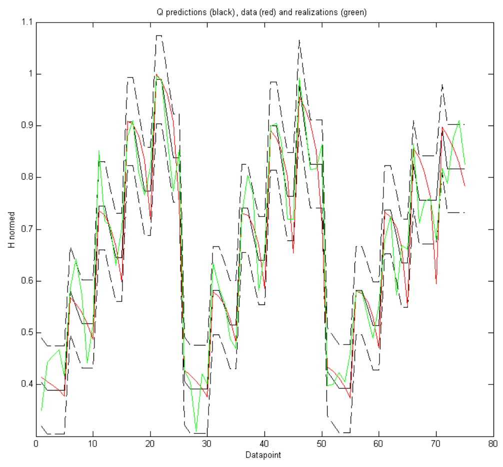


Figure 18: H predictions together with data, confidence intervals and realizations for model 3 with 6 fixed effects and 6 random effects for H.

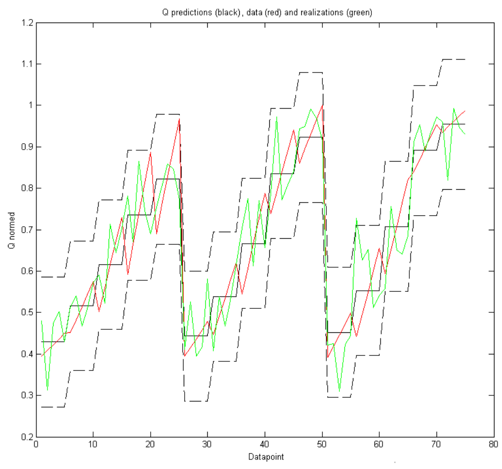


Figure 19: Q predictions together with data, confidence intervals and realizations for model 1 with 1 fixed effect and 1 random effect for Q.

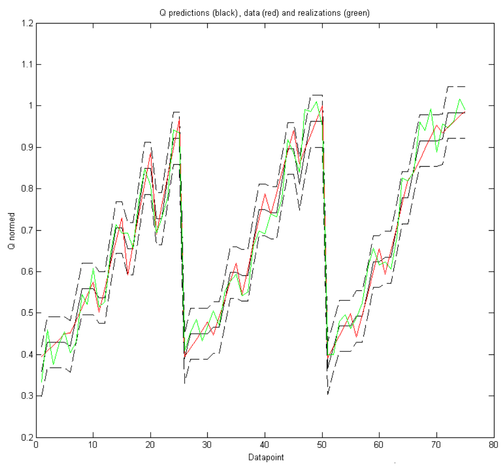


Figure 20: Q predictions together with data, confidence intervals and realizations for model 3 with 6 fixed effects and 6 random effects for Q.

Tables (4), (5) and (6) show the error (29) for both head and flow for each curve (i, j) for the three different models,

$$[H_{err_{ij}}, Q_{err_{ij}}] = \sum_{l=1}^{l=5} (\boldsymbol{y}_{ijl} - \hat{\boldsymbol{y}}_{ijl})^2 \tag{29}$$

Comparing the different models it is clear that the simplest model in table 2 gives very bad predictions at many curves. Especially Q predictions at molweight 33.9 and the two highest rotational speeds give very bad predictions compared to the two other models. But in general every curve are worse predicted in the simplest model than in the two other models. Comparing the basis model (model 2) with model 3 where every explanatory variable are used, and also products of these variables it seems they have fairly equally good predictions. The only curves there are an evident difference in predictions are the H predictions at the lowest rotational speed at every molweight.

Comparing the basis model with the cross validation in table 1, the results follow the same trend. They are high on the same curves, and low on the same curves. But in table 2 the numbers are extra high where they are already high in table 5. Similarly the numbers are a bit lower where they are already low in table 5.

Table 4: Total error at each curve for model 1

Curve (i,j)	$H_{err_{ij}}$	$Q_{err_{ij}}$
(1,1)	2.45	4.58
(2,1)	9.35	21.69
(3,1)	28.31	72.43
(4,1)	55.03	123.35
(5,1)	95.11	108.84
(1,2)	4.18	10.56
(2,2)	13.98	42.56
(3,2)	32.92	82.81
(4,2)	83.30	57.78
(5,2)	89.35	28.11
(1,3)	5.75	17.03
(2,3)	18.39	17.03
(3,3)	49.57	64.45
(4,3)	99.54	72.96
(5,3)	19.13	4.61

Table 5: Total error at each curve for model 2

Curve (i,j)	$H_{err_{ij}}$	$Q_{err_{ij}}$
(1,1)	2.60	7.48
(2,1)	9.44	7.14
(3,1)	8.96	7.98
(4,1)	20.79	20.28
(5,1)	45.34	13.78
(1,2)	8.13	5.84
(2,2)	7.32	4.72
(3,2)	10.14	9.16
(4,2)	35.63	10.71
(5,2)	54.14	10.53
(1,3)	9.92	6.89
(2,3)	6.40	7.64
(3,3)	18.39	7.72
(4,3)	84.95	9.45
(5,3)	13.03	4.39

Table 6: Total error at each curve for model 3

Curve (i, j)	$H_{err_{ij}}$	$Q_{err_{ij}}$
(1,1)	1.27	5.20
(2,1)	3.64	8.09
(3,1)	7.14	7.70
(4,1)	18.08	15.93
(5,1)	34.15	11.44
(1,2)	3.67	5.11
(2,2)	3.09	4.31
(3,2)	10.03	9.69
(4,2)	32.35	10.52
(5,2)	50.95	11.11
(1,3)	2.63	4.67
(2,3)	5.13	9.26
(3,3)	19.10	8.90
(4,3)	82.12	9.74
(5,3)	15.99	4.70

5 Conclusion

A linear mixed model has modelled head and flow data. The different explanatory used are molweight, rotational speed and an efficiency indicator.

The best predictions were achieved at the lowest molweights and rotational speeds. The model is not able to match up with the steeper zig-zag pattern for head values, and the more smoothened zig-zag pattern for flow values at larger molweights and rotational speeds.

A possible expansion of a linear mixed model for the head and flow dataset is to use a model that is more physical. Such a model might give better predictions in the "trouble" areas.

A GUI such as the one that has been made in this thesis could be very helpful at testing different models.

References

[1] T. K. Ambjørnsen. Compressor condition monitoring, 2006.

[2] A.S. Bryk and S.W. Raudenbuch. *Hierarchial Linear Models: Applications and Data Analysis Methods.* Newbury Park CA: Sage, 1992.

[3] A. Dobsen. *An Introduction to Generalized Linear Models.* Chapman and Hall, 2002.

[4] Dresser-Rand, 1200 West Sam Houston Pkwy. N. *Predicted Performance Curves.*

[5] A. E. Gelfand, S. K. Sahu, and B. P. Carlin. Efficient parametrisations for normal linear mixed models. *Biometrika*, 82(3):479, September 1995.

[6] D.A. Harville. Maximum likelihood approaches to variance component estimation and related problems. *Journal of the American Statistical Association*, 82:320, 1977.

[7] M. Herdahl. Linear mixed model for compressor head and flow data, 2007.

[8] N. M. Laird and J. H. Ware. Random-effects models for longitudinal data. 2002.

[9] W.N. Pearson, D.S. Henderson, and A.F. Armitage. A novel method for the performance control of a gas transmission compressor. *ASME Turbo Expo*, June 2002.

[10] J. M. Øverli. *Strømningsmaskiner*, volume 3. Tapir, 1992.

[11] S. N. Wood. *Generalized Additive Models.* Chapman and Hall/CRC, 2006.

APPENDIX

Figure 21 shows how the GUI is built up, i.e. what buttons and functions call to other functions. The code is listed in the rest of this appendix in verbatim.

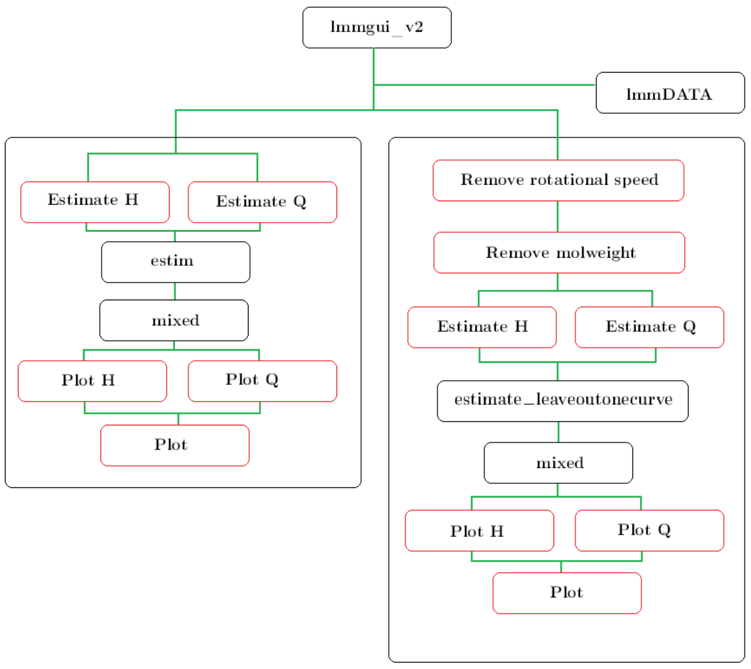


Figure 21: A DAG explaining how the GUI is built up. The black boxes are functions. The red boxes are not functions, but strings of different buttons in the GUI. The green lines gives an indication of which buttons or functions call at other functions. The right panel is for cross validation estimating.

```
function varargout = lmmgui_v2(varargin)
% LMMGUI_V2 M-file for lmmgui_v2.fig
%      LMMGUI_V2, by itself, creates a new LMMGUI_V2 or raises the existing
%      singleton*.
%
%      H = LMMGUI_V2 returns the handle to a new LMMGUI_V2 or the handle to
%      the existing singleton*.
%
%      LMMGUI_V2('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in LMMGUI_V2.M with the given input arguments.
%
%      LMMGUI_V2('Property','Value',...) creates a new LMMGUI_V2 or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before lmmgui_v2_OpeningFunction gets called.
An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to lmmgui_v2_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help lmmgui_v2

% Last Modified by GUIDE v2.5 21-May-2008 00:13:35

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @lmmgui_v2_OpeningFcn, ...
                   'gui_OutputFcn',  @lmmgui_v2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before lmmgui_v2 is made visible.
function lmmgui_v2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to lmmgui_v2 (see VARARGIN)

[M,k,mw,n,Q,H,eff]=lmmDATA;

for i=1:5
    q(i)=Q(i);
    h(i)=H(i);
end;
plot(q,h,'-');
```

```
hold;
for i=2:15
    for j=1:5
        q(j) = Q((i-1)*5+j);
        h(j) = H((i-1)*5+j);
    end;
    plot(q,h,'-');
end;
xlabel('Q')
ylabel('H')

handles.molweight=0;
handles.rotspeed=0;
handles.data=M;
mod=zeros(12,1);
handles.model=mod;
handles.Q=Q;
handles.H=H;



% Choose default command line output for lmmgui_v2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes lmmgui_v2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = lmmgui_v2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in molweight_popup.
function molweight_popup_Callback(hObject, eventdata, handles)
% hObject    handle to molweight_popup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns molweight_popup contents as cell
%        contents{get(hObject,'Value')} returns selected item from molweight_popup

switch get(handles.molweight_popup,'Value')
    case 1
        handles.molweight=39.9;
    case 2
        handles.molweight=44.3;
    case 3
        handles.molweight=47.5;
    otherwise
end
guidata(hObject, handles);
% --- Executes during object creation, after setting all properties.
function molweight_popup_CreateFcn(hObject, eventdata, handles)
% hObject    handle to molweight_popup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColo
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in rotspeed_popup.
function rotspeed_popup_Callback(hObject, eventdata, handles)
% hObject    handle to rotspeed_popup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns rotspeed_popup contents as cell array
%        contents{get(hObject,'Value')} returns selected item from rotspeed_popup


switch get(handles.rotspeed_popup,'Value')
    case 1
        handles.rotspeed=6033;
    case 2
        handles.rotspeed=6895;
    case 3
        handles.rotspeed=7757;
    case 4
        handles.rotspeed=8619;
    case 5
        handles.rotspeed=9050;
    otherwise
end
guidata(hObject, handles);
```

```
% --- Executes during object creation, after setting all properties.
function rotspeed_popup_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rotspeed_popup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgrou
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in estimate_leavecurve_q.
function estimate_leavecurve_q_Callback(hObject, eventdata, handles)
% hObject    handle to estimate_leavecurve_q (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


[X,Z,data,radnumre] = estimate_leaveoutonecurve(handles.molweight,handles.rotspeed


handles.radnumreQ=radnumre;
Qy=data;
handles.Qy2=Qy;

B = inv(X'*X)*X'*Qy;

S=var(Qy-X*B);
r=14;
ss20=handles.Qss20;

s20=S/3*ones(ss20,1);

% Last input is 2:REML estimation

[s2a,bQ,uQ,Is2,C,H,q,loglik,loops] = mixed(Qy,X,Z,r*ones((ss20-1),1),s20,2);

s20=3*S*ones(ss20,1);

% Last input is 2:REML estimation

[s2,bQ,uQ,Is2,C,H,q,loglik,loops] = mixed(Qy,X,Z,r*ones((ss20-1),1),s20,2);
```

```
disp('Variance estimates with starting point 1');

s2(1:2,1)

s2a(1:2,1)

disp('Variance estimates with starting point 2');

s2(3,1)

s2a(3,1)


logliktext=num2str(loglik);
mod1=num2str(handles.model(1));
mod2=num2str(handles.model(2));
mod3=num2str(handles.model(3));
mod4=num2str(handles.model(4));
mod5=num2str(handles.model(5));
mod6=num2str(handles.model(6));
mod7=num2str(handles.model(7));
mod8=num2str(handles.model(8));
mod9=num2str(handles.model(9));
mod10=num2str(handles.model(10));
mod11=num2str(handles.model(11));
mod12=num2str(handles.model(12));

inputtekst=['loglik: ',logliktext,' model: ',mod1,' ',mod2,' ',mod3,' ',mod4,' ',mod5,'

set(handles.modelQ_textleave,'String',inputtekst);


handles.QX2=X;
handles.QZ2=Z;
handles.s2Q2=s2;
handles.bQ2=bQ;
handles.uQ2=uQ;

guidata(hObject, handles);



% --- Executes on button press in plotbutton.
function plotbutton_Callback(hObject, eventdata, handles)
% hObject    handle to plotbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

handles.Q
handles.H
for i=1:5
    q(i)=handles.Q(i);
    h(i)=handles.H(i);
end;

hold off;

plot(handles.Qhat*12058,handles.Hhat*109.7,'xr');
hold;

plot(q,h,'-');
for i=2:15
    for j=1:5
        q(j) = handles.Q((i-1)*5+j);
        h(j) = handles.H((i-1)*5+j);
    end;
    plot(q,h,'-');
end;
xlabel('Q')
ylabel('H')




% --- Executes on button press in estimate_leavecurve_q_all_data.
function estimate_all_data_Q_Callback(hObject, eventdata, handles)
% hObject    handle to estimate_leavecurve_q_all_data (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

[data,X,Z,M,Morig,ss20,r] = estim(handles.model);


handles.QX=X;
handles.QZ=Z;
handles.Qss20=ss20;
handles.Qr=r;
handles.M=M
handles.HLP=[M(:,2),M(:,3),M(:,6),M(:,2).*M(:,3),M(:,2).*M(:,6),M(:,3).*M(:,6)];


%

% For each dimension of y (q,h), estimate the parameters

% by REML code,

%
```

```
% s20 = prior choice of variance components,

% We have r=2, for two random effects b_1, b_2

% We have

%

%

%----------------------------------

%

%

% volumstrøm Q

%------------------------------------

Qy=data(:,1);
handles.Qy=Qy;

handles.Qy=Qy;

B = inv(X'*X)*X'*Qy;

S=var(Qy-X*B);



s20=S/3*ones(ss20,1);

% Last input is 2:REML estimation

[s2a,bQ,uQ,Is2,C,H,q,loglik,loops] = mixed(Qy,X,Z,r*ones((ss20-1),1),s20,2);

s20=3*S*ones(ss20,1);

% Last input is 2:REML estimation

[s2,bQ,uQ,Is2,C,H,q,loglik,loops] = mixed(Qy,X,Z,r*ones((ss20-1),1),s20,2);
disp('s20')
s20

disp('Variance estimates with starting point 1');

s2(1:2,1)
```

```
s2a(1:2,1)

disp('Variance estimates with starting point 2');

s2(3,1)

s2a(3,1)

logliktext=num2str(loglik);
mod1=num2str(handles.model(1));
mod2=num2str(handles.model(2));
mod3=num2str(handles.model(3));
mod4=num2str(handles.model(4));
mod5=num2str(handles.model(5));
mod6=num2str(handles.model(6));
mod7=num2str(handles.model(7));
mod8=num2str(handles.model(8));
mod9=num2str(handles.model(9));
mod10=num2str(handles.model(10));
mod11=num2str(handles.model(11));
mod12=num2str(handles.model(12));

inputtekst=['loglik: ',logliktext,' model: ',mod1,' ',mod2,' ',mod3,' ',mod4,' ',m

set(handles.modelQ_text,'String',inputtekst);

handles.s2Q=s2;
handles.bQ=bQ;
handles.uQ=uQ;

guidata(hObject, handles);


% --- Executes on button press in fixed_m.
function fixed_m_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_m (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_m


checkboxStatus = get(handles.fixed_m,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(1)=1;
else
    %if box is unchecked,
    handles.model(1)=0;
end

guidata(hObject, handles);



% --- Executes on button press in fixed_n.
function fixed_n_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_n (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_n


checkboxStatus = get(handles.fixed_n,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(2)=1;
else
    %if box is unchecked,
    handles.model(2)=0;
end

guidata(hObject, handles);



% --- Executes on button press in fixed_i.
function fixed_i_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_i (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_i

checkboxStatus = get(handles.fixed_i,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(3)=1;
else
    %if box is unchecked,
    handles.model(3)=0;
end
guidata(hObject, handles);
```

```
% --- Executes on button press in fixed_mn.
function fixed_mn_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_mn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_mn


checkboxStatus = get(handles.fixed_mn,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(4)=1;
else
    %if box is unchecked,
    handles.model(4)=0;
end
guidata(hObject, handles);


% --- Executes on button press in fixed_mi.
function fixed_mi_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_mi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_mi


checkboxStatus = get(handles.fixed_mi,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(5)=1;
else
    %if box is unchecked,
    handles.model(5)=0;
end
guidata(hObject, handles);



% --- Executes on button press in fixed_ni.
function fixed_ni_Callback(hObject, eventdata, handles)
% hObject    handle to fixed_ni (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of fixed_ni
```

```
checkboxStatus = get(handles.fixed_ni,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(6)=1;
else
    %if box is unchecked,
    handles.model(6)=0;
end
guidata(hObject, handles);




% --- Executes on button press in random_m.
function random_m_Callback(hObject, eventdata, handles)
% hObject    handle to random_m (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_m


checkboxStatus = get(handles.random_m,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(7)=1;
else
    %if box is unchecked,
    handles.model(7)=0;
end
guidata(hObject, handles);



% --- Executes on button press in random_n.
function random_n_Callback(hObject, eventdata, handles)
% hObject    handle to random_n (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_n

checkboxStatus = get(handles.random_n,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(8)=1;
else
```

```
    %if box is unchecked,
    handles.model(8)=0;
end
guidata(hObject, handles);



% --- Executes on button press in random_i.
function random_i_Callback(hObject, eventdata, handles)
% hObject    handle to random_i (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_i

checkboxStatus = get(handles.random_i,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(9)=1;
else
    %if box is unchecked,
    handles.model(9)=0;
end
guidata(hObject, handles);



% --- Executes on button press in random_nm.
function random_nm_Callback(hObject, eventdata, handles)
% hObject    handle to random_nm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_nm

checkboxStatus = get(handles.random_nm,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(10)=1;
else
    %if box is unchecked,
    handles.model(10)=0;
end
guidata(hObject, handles);



% --- Executes on button press in random_ni.
function random_ni_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to random_ni (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_ni


checkboxStatus = get(handles.random_ni,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(12)=1;
else
    %if box is unchecked,
    handles.model(12)=0;
end
guidata(hObject, handles);




% --- Executes on button press in random_mi.
function random_mi_Callback(hObject, eventdata, handles)
% hObject    handle to random_mi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of random_mi


checkboxStatus = get(handles.random_mi,'Value');
if(checkboxStatus==1)
    %if box is checked,
    handles.model(11)=1;
else
    %if box is unchecked,
    handles.model(11)=0;
end
guidata(hObject, handles);


% --- Executes on button press in plot_q_solution.
function plot_q_solution_Callback(hObject, eventdata, handles)
% hObject    handle to plot_q_solution (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


X=handles.QX;
Z=handles.QZ;
ss20=handles.Qss20;
```

```
r=handles.Qr;
b=handles.bQ;
u=handles.uQ;
y=handles.Qy;
s2=handles.s2Q;

yhat=X*b+Z*u;

Qhat=yhat;
handles.Qhat=Qhat;

Dcap=zeros(r*(ss20-1),r*(ss20-1));

for i=1:(ss20-1),

    Dcap((i-1)*r+1:i*r,(i-1)*r+1:i*r)=s2(i,1)*eye(r);

end;

varu=Dcap-Dcap*Z'*inv(Z*Dcap*Z'+s2(ss20,1)*eye(size(y,1)))*Z*Dcap;

varyhat=Z*varu*Z'+s2(ss20,1)*eye(size(y,1));

Qreal=yhat+chol(varyhat)'*randn(size(X,1),1);

%figure(2)

%clf;

hold off;

plot(yhat,'k');

hold;

plot(yhat+1.96*sqrt(diag(varyhat)),'k--');

plot(yhat-1.96*sqrt(diag(varyhat)),'k--');

plot(y,'r')

plot(Qreal,'g');
xlabel('Datapoint');
ylabel('Q normed');
title('Q predictions (red), data (black) and realizations (green)');
print plot.ps

guidata(hObject, handles);
```

```
% --- Executes on button press in plot_h_solution.
function plot_h_solution_Callback(hObject, eventdata, handles)
% hObject    handle to plot_h_solution (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


X=handles.HX;
Z=handles.HZ;
ss20=handles.Hss20;
r=handles.Hr;
b=handles.bH;
u=handles.uH;
y=handles.Hy;
s2=handles.s2H;

yhat=X*b+Z*u;

Hhat=yhat;
handles.Hhat=Hhat;

Dcap=zeros(r*(ss20-1),r*(ss20-1));

for i=1:(ss20-1),

    Dcap((i-1)*r+1:i*r,(i-1)*r+1:i*r)=s2(i,1)*eye(r);

end;

varu=Dcap-Dcap*Z'*inv(Z*Dcap*Z'+s2(ss20,1)*eye(size(y,1)))*Z*Dcap;

varyhat=Z*varu*Z'+s2(ss20,1)*eye(size(y,1));

Hreal=yhat+chol(varyhat)'*randn(size(X,1),1);

%figure(2)

%clf;

hold off;

plot(yhat,'k');

hold;
```

```
plot(yhat+1.96*sqrt(diag(varyhat)),'k--');

plot(yhat-1.96*sqrt(diag(varyhat)),'k--');

plot(y,'r')

plot(Hreal,'g');
xlabel('Datapoint');
ylabel('H normed');
title('H predictions (red), data (black) and realizations (green)');

yhat=yhat*47.5;
y=y*47.5;
err=yhat-y;
err
guidata(hObject, handles);


% --- Executes on button press in estimate_all_data_h.
function estimate_all_data_h_Callback(hObject, eventdata, handles)
% hObject    handle to estimate_all_data_h (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



%-----------------------------------

%

% Løftehøyde

%

%-------------------------------------

[data,X,Z,M,Morig,ss20,r] = estim(handles.model);

handles.HX=X;
handles.HZ=Z;
handles.Hss20=ss20;
handles.Hr=r;





Hy=data(:,2);
handles.Hy=Hy;
```

```
B = inv(X'*X)*X'*Hy;

S=var(Hy−X*B);



s20=S/3*ones(ss20,1);

% Last input is 2:REML estimation

[s2a,bH,uH,Is2,C,H,q,loglik,loops] = mixed(Hy,X,Z,r*ones((ss20−1),1),s20,2);

s20=3*S*ones(ss20,1);

% Last input is 2:REML estimation

[s2,bH,uH,Is2,C,H,q,loglik,loops] = mixed(Hy,X,Z,r*ones((ss20−1),1),s20,2);

logliktext=num2str(loglik);
mod1=num2str(handles.model(1));
mod2=num2str(handles.model(2));
mod3=num2str(handles.model(3));
mod4=num2str(handles.model(4));
mod5=num2str(handles.model(5));
mod6=num2str(handles.model(6));
mod7=num2str(handles.model(7));
mod8=num2str(handles.model(8));
mod9=num2str(handles.model(9));
mod10=num2str(handles.model(10));
mod11=num2str(handles.model(11));
mod12=num2str(handles.model(12));

inputtekst=['loglik: ',logliktext,' model: ',mod1,' ',mod2,' ',mod3,' ',mod4,' ',mod5,'

set(handles.modelH_text,'String',inputtekst);

disp('Variance estimates with starting point 1');

s2(1:2,1)

s2a(1:2,1)

disp('Variance estimates with starting point 2');

s2(3,1)

s2a(3,1)
```

```
handles.s2H=s2;
handles.bH=bH;
handles.uH=uH;

guidata(hObject, handles);



% --- Executes on button press in estimate_leavecurve_h.
function estimate_leavecurve_h_Callback(hObject, eventdata, handles)
% hObject    handle to estimate_leavecurve_h (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%handles.HX
%handles.HZ
[X,Z,data,radnumre] = estimate_leaveoutonecurve(handles.molweight,handles.rotspeec

handles.radnumreH=radnumre;
Hy=data;
handles.Hy2=Hy;

B = inv(X'*X)*X'*Hy;

S=var(Hy-X*B);
r=14;
ss20=handles.Hss20;

s20=S/3*ones(ss20,1);

% Last input is 2:REML estimation

[s2a,bH,uH,Is2,C,H,q,loglik,loops] = mixed(Hy,X,Z,r*ones((ss20-1),1),s20,2);

s20=3*S*ones(ss20,1);

% Last input is 2:REML estimation

[s2,bH,uH,Is2,C,H,q,loglik,loops] = mixed(Hy,X,Z,r*ones((ss20-1),1),s20,2);



disp('Variance estimates with starting point 1');

s2(1:2,1)

s2a(1:2,1)

disp('Variance estimates with starting point 2');
```

```
s2(3,1)

s2a(3,1)

logliktext=num2str(loglik);
mod1=num2str(handles.model(1));
mod2=num2str(handles.model(2));
mod3=num2str(handles.model(3));
mod4=num2str(handles.model(4));
mod5=num2str(handles.model(5));
mod6=num2str(handles.model(6));
mod7=num2str(handles.model(7));
mod8=num2str(handles.model(8));
mod9=num2str(handles.model(9));
mod10=num2str(handles.model(10));
mod11=num2str(handles.model(11));
mod12=num2str(handles.model(12));

inputtekst=['loglik: ',logliktext,' model: ',mod1,' ',mod2,' ',mod3,' ',mod4,' ',mod5,'

set(handles.modelH_textleave,'String',inputtekst);



handles.HX2=X;
handles.HZ2=Z;
handles.s2H2=s2;
handles.bH2=bH;
handles.uH2=uH;

guidata(hObject, handles);




% --- Executes on button press in plot_q_leavecurve.
function plot_q_leavecurve_Callback(hObject, eventdata, handles)
% hObject    handle to plot_q_leavecurve (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

X=handles.QX2;
Z=handles.QZ2;
ss20=handles.Qss20;
r=handles.Qr-1;
b=handles.bQ2;
u=handles.uQ2;
y=handles.Qy2;
s2=handles.s2Q2;
```

```
yhat=X*b+Z*u;

Qhat=yhat;
handles.Qhat=Qhat;

Dcap=zeros(r*(ss20-1),r*(ss20-1));

for i=1:(ss20-1),

    Dcap((i-1)*r+1:i*r,(i-1)*r+1:i*r)=s2(i,1)*eye(r);

end;

varu=Dcap-Dcap*Z'*inv(Z*Dcap*Z'+s2(ss20,1)*eye(size(y,1)))*Z*Dcap;

varyhat=Z*varu*Z'+s2(ss20,1)*eye(size(y,1));

Qreal=yhat+chol(varyhat)'*randn(size(X,1),1);

%figure(2)

%clf;

hold off;

plot(yhat,'k');

hold;

plot(yhat+1.96*sqrt(diag(varyhat)),'k--');

plot(yhat-1.96*sqrt(diag(varyhat)),'k--');

plot(y,'r')

plot(Qreal,'g');

title('Q predictions (red), data (black) and realizations (green)');


guidata(hObject, handles);


% --- Executes on button press in plot_h_leavecurve.
function plot_h_leavecurve_Callback(hObject, eventdata, handles)
% hObject    handle to plot_h_leavecurve (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
radnumre=handles.radnumreH;
HLP=handles.HLP;
X=handles.HX2;
Z=handles.HZ2;
ss20=handles.Hss20;
r=handles.Hr-1;
b=handles.bH2;
u=handles.uH2;
y=handles.Hy2;
s2=handles.s2H2;

utelatt=HLP(radnumre(1):radnumre(5),:);

%utpred=utelatt*b;

yhat=X*b+Z*u;

Qhat=yhat;
handles.Qhat=Qhat;

Dcap=zeros(r*(ss20-1),r*(ss20-1));

for i=1:(ss20-1),

    Dcap((i-1)*r+1:i*r,(i-1)*r+1:i*r)=s2(i,1)*eye(r);

end;

varu=Dcap-Dcap*Z'*inv(Z*Dcap*Z'+s2(ss20,1)*eye(size(y,1)))*Z*Dcap;

varyhat=Z*varu*Z'+s2(ss20,1)*eye(size(y,1));

Qreal=yhat+chol(varyhat)'*randn(size(X,1),1);
%utpred
%plotme=[yhat(1:radnumre(1)-1)];
%plotme=[plotme utpred];
%plotme=[plotme yhat(radnumre(5)+1:length(yhat))];




hold off;

plot(yhat,'k');

hold;

plot(yhat+1.96*sqrt(diag(varyhat)),'k--');
plot(yhat-1.96*sqrt(diag(varyhat)),'k--');

plot(y,'r')

plot(Qreal,'g');

title('Q predictions (red), data (black) and realizations (green)');


guidata(hObject, handles);


% --- Executes on button press in plot_leavecurve.
function plot_leavecurve_Callback(hObject, eventdata, handles)
% hObject    handle to plot_leavecurve (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

The mixed function that is called does the estimation of the model parameters and the likelihood. It is written at the Slovak Academy of Sciences:

```
function [s2,b,u,Is2,C,H,q,loglik,loops] = mixed(y,X,Z,dim,s20,method);

%MIXED   Computes ML,REML,MINQE(I),MINQE(U,I),BLUE(b),BLUP(u)

%        by Henderson's Mixed Model Equations Algorithm.

%

%======================================================================

% Syntax:

%   [s2,b,u,Is2,C,H,q,loglik,loops] = mixed(y,X,Z,dim,s20,method);

%

%======================================================================

% Model: Y=X*b+Z*u+e,

%        b=(b_1',...,b_f')' and u=(u_1',...,u_r')',

%        E(u)=0, Var(u)=diag(sigma^2_i*I_{m_i}), i=1,...,r

%        E(e)=0, Var(e)=sigma^2_{r+1}*I_n,
%        Var(y)=Sig=sum_{i=1}^{r+1} sigma^2_i*Sig_i.
%        We assume normality and independence of u and e.
%
% Inputs:
%   y      - n-dimensional vector of observations.
%   X      - (n * k)-design matrix for
%            fixed effects b=[b_1;...;b_f],
%            typically X=[X_1,...,X_f] for some X_i.
%   Z      - (n * m)-design matrix for
%            random efects u=[u_1;...;u_r],
%            typically Z=[Z_1,...,Z_r] for some Z_i.
%   dim    - Vector of dimensions of u_i, i=1,...,r,
%            dim=[m_1;...;m_r], m=sum(dim).
%   s20    - A prior choice of the variance components,
%            s20=[s20_1;...;s20_r;s20_{r+1}].
%            SHOULD BE POSITIVE for method>0
%   method - Method of estimation of variance components;
%            0:NO estimation, 1:ML, 2:REML, 3:MINQE(I), 4:MINQE(U,I)
%
%=======================================================================
% Outputs:
%   s2     - Estimated vector of variance components
%            (sigma^2_1,..., sigma^2_{r+1})'.
%            A warning message appears if some of the estimated
%             variance components is negative or equal to zero.
%             In such cases the calculated Fisher information
%             matrices are inadequate.
%   b       - k-dimensional vector of estimated fixed effects beta,
%             b=[b_1;...;b_f]=(X'Sig^{-1}X)^{+}X'Sig^{-1}y.
%   u       - m-dimensional vector of EBLUP's of random effects U,
%             u=[u_1;...;u_r].
%   Is2     - Fisher information matrix for variance components;
%             if method=0 the output is Is2=[];
%             if metod=3 or method=4 the output is inversion of the
%             covariance matrix of MINQE calculated at estimated s2.
%   C       - g-inverse of Henderson's MME matrix, where
%             C=pinv([XX XZ; XZ' ZZ+inv(D)*s0]/s0), if inv(D) exists
%             or C=s0*[I 0; 0 D]*pinv([XX XZ*D; XZ' V]) otherwise
%   H       - Criterial matrix for MINQE calculated at priors s20;
%             if method=3
%             H_ij=trace(Sig_0^{-1}*Sig_i*Sig_0^{-1}*Sig_j),
%             if method=4
%             H_ij=trace((M*Sig_0*M)^{+}*Sig_i*(M*Sig_0*M)^{+}*Sig_j)
%   q       - (r+1)-dimensional vector of MINQE(U,I) quadratic forms
%             calculated at prior values s20;
%             if method=0,1,2 the output is q=[], otherwise
%             q_i=y'*(M*Sig_0*M)^{+}*Sig_i*(M*Sig_0*M)^{+}*y.
%   loglik - Log-likelihood evaluated at the estimated parameters;
%             if method=1 loglik=log-likelihood(ML),
%             if method=2 loglik=log-likelihood(REML),
%             if method=3 or method=4 loglik=[],
%         if method=0 loglik=informative value of
%             log of the joint pdf of (y,u).
%   loops  - Number of loops.
%
%========================================================================
% REFERENCES
%
% Searle, S.R., Cassela, G., McCulloch, C.E.: Variance Components.
% John Wiley & Sons, INC., New York, 1992. (pp. 275-286).
%
% Witkovsky, V.: MATLAB Algorithm mixed.m for solving
% Henderson's Mixed Model Equations.
% Technical Report, Institute of Measurement Science,
% Slovak Academy of Sciences, Bratislava, Dec. 2001.
% See http://www.mathpreprints.com.
%
% The algorithm mixed.m is available at
% http://www.mathworks.com/matlabcentral/fileexchange
% see the Statistics Category.
%
%========================================================================
% Ver.: 2.0
% Revised 21-Dec-2001 20:31:48
% Copyright (c) 1998-2001 Viktor Witkovsky

%=======================================================================
% CONTACT ADDRESS:
% Viktor Witkovsky
% Institute of Measurement Science
% Slovak Academy of Sciences
% Dubravska cesta 9
% 84219 BRATISLAVA, Slovak Republic
% Tel:(+421905) 223191
% Fax:(+4212) 54775943
% E-mail: umerwitk@savba.sk
% http://nic.savba.sk/sav/inst/umer/
%=======================================================================
%   BEGIN MIXED.M
%=======================================================================
%   This is the (only) required input.
%   The algorith mixed.m could be easily changed in such a way
%   that the required inputs will be y, a, XX, XZ, and ZZ,
%   and the call would be mixed(y,a,XX,XZ,ZZ,dim,s20,method);
%   instead of mixed(y,X,Z,dim,s20,method);
%=======================================================================
y=y(:);
yy=y'*y;
```

```
Xy=X'*y;

Zy=Z'*y;

XX=X'*X;

XZ=X'*Z;

ZZ=Z'*Z;

a=[Xy;Zy];

% end of required input parameters

n=length(y);

[k,m]=size(XZ);

rx=rank(XX);

s20=s20(:);

r=length(s20)−1;

Im=eye(m);

loops=0;

%======================================================================

%   METHOD=0:

%   No estimation of variance components

%   Output is BLUE(b), BLUP(u), and C,

%       calculated at chosen values s20

%======================================================================

if method==0,

   s0=s20(r+1);

   d=s20(1)*ones(dim(1),1);

   for i=2:r,
       d=[d;s20(i)*ones(dim(i),1)];
   end;
   D=diag(d);
   V=s0*Im+ZZ*D;
   A=[XX XZ*D;XZ' V];
   A=pinv(A);
   C=s0*[A(1:k,1:k) A(1:k,k+1:k+m);...
     D*A(k+1:k+m,1:k) D*A(k+1:k+m,k+1:k+m)];
   bb=A*a;
   b=bb(1:k);
   v=bb(k+1:k+m);
   u=D*v;
   Aux=yy-b'*Xy-u'*Zy;
   if all(s20),
   loglik=-((n+m)*log(2*pi)+n*log(s0)+log(prod(d))+Aux/s0)/2;
   else
   loglik=[];
   end;
   s2=s20;
   Is2=[];
   H=[];
   q=[];
   return;
end;
%======================================================================
%   METHOD=1,2,3,4: ESTIMATION OF VARIANCE COMPONENTS
%======================================================================
fk=find(s20≤0);
if any(fk),
   s20(fk)=100*eps*ones(size(fk));
   warning('Priors in s20 are negative or zeros !CHANGED!');
end;
sig0=s20;
s21=s20;
ZMZ=ZZ-XZ'*pinv(XX)*XZ;
q=zeros(r+1,1);
%======================================================================
%   START OF THE MAIN LOOP
%======================================================================
epss=0.00001; % Given precission for stopping rule
crit=1;
loopiter=1;
while ((loopiter<25)&(crit>epss)),
      loops=loops+1;
      sigaux=s20;
      s0=s20(r+1);
      d=s20(1)*ones(dim(1),1);
      for i=2:r,
          d=[d;s20(i)*ones(dim(i),1)];
      end;
      D=diag(d);
      V=s0*Im+ZZ*D;
      W=s0*inv(V);
      T=inv(Im+ZMZ*D/s0);
      A=[XX XZ*D;XZ' V];
      bb=pinv(A)*a;
      b=bb(1:k);
      v=bb(k+1:k+m);
      u=D*v;
%=======================================================================
%   ESTIMATION OF ML AND REML OF VARIANCE COMPONENTS
%=======================================================================
      iupp=0;
      for i=1:r,
          ilow=iupp+1;
          iupp=iupp+dim(i);
          Wii=W(ilow:iupp,ilow:iupp);
          Tii=T(ilow:iupp,ilow:iupp);
          w=u(ilow:iupp);
          ww=w'*w;
          q(i)=ww/(s20(i)*s20(i));
          s20(i)=ww/(dim(i)-trace(Wii));
          s21(i)=ww/(dim(i)-trace(Tii));
      end;
      Aux=yy-b'*Xy-u'*Zy;

      Aux1=Aux-(u'*v)*s20(r+1);

      q(r+1)=Aux1/(s20(r+1)*s20(r+1));

      s20(r+1)=Aux/n;

      s21(r+1)=Aux/(n-rx);

         if method==1,

               crit=norm(sigaux-s20);

               H=[];

               q=[];

         elseif method==2,

               s20=s21;

               crit=norm(sigaux-s20);

               H=[];

               q=[];

         else

               crit=0;

         end;

      loopiter=loopiter+1;

end;

disp(sprintf('EM algorithm for REML performed %d iterations',loopiter));

%==================================================================

%   END OF THE MAIN LOOP

%==================================================================

%   COMPUTING OF THE MINQE CRITERIAL MATRIX H
```

```
%=========================================================================

if (method==3 | method==4),

   H=eye(r+1);

   if method==4,

      W=T;

      H(r+1,r+1)=(n-rx-m+trace(W*W))/(sigaux(r+1)*sigaux(r+1)); %VW

   else

      H(r+1,r+1)=(n-m+trace(W*W))/(sigaux(r+1)*sigaux(r+1));

   end;

   iupp=0;

   for i=1:r;

       ilow=iupp+1;

       iupp=iupp+dim(i);

       trii=trace(W(ilow:iupp,ilow:iupp));

       trsum=0;

       jupp=0;

       for j=1:r,

           jlow=jupp+1;

           jupp=jupp+dim(j);

           tr=trace(W(ilow:iupp,jlow:jupp)*W(jlow:jupp,ilow:iupp));

           trsum=trsum+tr;

           H(i,j)=((i==j)*(dim(i)-2*trii)+tr)/(sigaux(i)*sigaux(j));

       end;

       H(r+1,i)=(trii-trsum)/(sigaux(r+1)*sigaux(i));

       H(i,r+1)=H(r+1,i);
   end;
end;
%=======================================================================
%   SET THE RESULTS: MINQE(I), MINQE(U,I), ML, AND REML
%=======================================================================
if (method==3 | method==4),
   s2=pinv(H)*q;
   loglik=[];
else
   s2=s20;
end;
fk=find(s2<0.000000000001);
if any(fk),
   warning('Estimated variance components are negative or zeros!');
end;
%=======================================================================
%   BLUE, BLUP, THE MME'S C MATRIX AND THE LOG-LIKELIHOOD
%=======================================================================
s0=s2(r+1);
d=s2(1)*ones(dim(1),1);
for i=2:r,
    d=[d;s2(i)*ones(dim(i),1)];
end;
D=diag(d);
```

```
V=s0*Im+ZZ*D;

W=s0*inv(V);;

T=inv(Im+ZMZ*D/s0);

A=[XX XZ*D;XZ' V];

A=pinv(A);

C=s0*[A(1:k,1:k) A(1:k,k+1:k+m);...

  D*A(k+1:k+m,1:k) D*A(k+1:k+m,k+1:k+m)];

bb=A*a;

b=bb(1:k);

v=bb(k+1:k+m);

u=D*v;

if (method==1),

   loglik=-(n*log(2*pi*s0)-log(det(W))+n)/2;

elseif (method==2),

   loglik=-((n-rx)*log(2*pi*s0)-log(det(T))+(n-rx))/2;

end;

disp(sprintf('Log likelihood for current model is %0.5g',loglik));


%=====================================================================

%   FISHER INFORMATION MATRIX FOR VARIANCE COMPONENTS

%=====================================================================

Is2=eye(r+1);

if (method==2 | method==4),

   W=T;

   Is2(r+1,r+1)=(n-rx-m+trace(W*W))/(s2(r+1)*s2(r+1)); %VW
```

```
else
   Is2(r+1,r+1)=(n−m+trace(W*W))/(s2(r+1)*s2(r+1));
end;
iupp=0;
for i=1:r;
   ilow=iupp+1;
   iupp=iupp+dim(i);
   trii=trace(W(ilow:iupp,ilow:iupp));
   trsum=0;
   jupp=0;
   for j=1:r,
       jlow=jupp+1;
       jupp=jupp+dim(j);
       tr=trace(W(ilow:iupp,jlow:jupp)*W(jlow:jupp,ilow:iupp));
       trsum=trsum+tr;
       Is2(i,j)=((i==j)*(dim(i)−2*trii)+tr)/(s2(i)*s2(j));
   end;
   Is2(r+1,i)=(trii−trsum)/(s2(r+1)*s2(i));
   Is2(i,r+1)=Is2(r+1,i);
end;
Is2=Is2/2;
%=======================================================================
%   EOF MIXED.M
%=======================================================================
```

Other functions called are lmmGUI which makes the dataset and estim which makes the Z and X matrices upon the users choice of model parameters;

```
  % modelparam er en vektor av lengde 12. Den angir hvilke parametere som
% skal være med i modellen. elementene i vektoren er enten 0 eller 1, der 1
% angir at parameteren skal være med i modellen. Når 0 er valgt skal
% parameteren ikke være med.

%                       FIXED            |      RANDOM
% estim(modelparam[m, n, I, m*n, m*I, n*I, m, n, I, m*n, m*I, n*I])


function [data,X,Z,M,Morig,ss20,r] = estim(modelparam);




% Datamatrisa M
% Kolonne 1 er k, et slags kjørenivå, men ikke gyldig som kovariat
% kolonne 2 er molvekt (m)
% kolonne 3 er turtall (n)
% kolonne 4 er volumstrøm (Q)
% kolonne 5 er løftehøyde (H)
% kolonne 6 er effekt (c)
%

M=[
1   39.9    6033    4769    45.4    79.85; ...
2   39.9    6033    4931    44.5    79.46; ...
3   39.9    6033    5093    43.6    78.95; ...
4   39.9    6033    5255    42.6    78.32; ...
5   39.9    6033    5417    41.4    77.48;...
1   39.9    6895    5436    62.3    80.50;...
2   39.9    6895    5808    61.0    80.54;...
3   39.9    6895    6181    59.2    80.21;...
4   39.9    6895    6553    56.8    79.19;...
5   39.9    6895    6926    53.5    77.29;...
1   39.9    7757    6051    80.8    80.15;...
2   39.9    7757    6734    79.8    80.75;...
3   39.9    7757    7417    77.4    80.87;...
4   39.9    7757    8100    73.3    79.86;...
5   39.9    7757    8784    65.7    76.10;...
1   39.9    8619    7130    99.7    78.68;...
2   39.9    8619    8021    99.0    79.87;...
3   39.9    8619    8912    96.6    80.88;...
4   39.9    8619    9803    91.4    80.17;...
5   39.9    8619    10694   79.0    74.68;...
```

```
1   39.9    9050    8320    109.7   78.60;...
2   39.9    9050    9155    108.1   79.74;...
3   39.9    9050    9991    105.4   80.52;...
4   39.9    9050    10826   99.5    79.68;...
5   39.9    9050    11662   82.1    72.20;...
1   44.3    6033    4751    46.9    80.38;...
2   44.3    6033    5005    45.9    80.22; ...
3   44.3    6033    5259    44.6    79.76; ...
4   44.3    6033    5512    43.1    78.88; ...
5   44.3    6033    5766    41.3    77.55; ...
1   44.3    6895    5396    63.6    80.39; ...
2   44.3    6895    5918    62.5    80.75; ...
3   44.3    6895    6441    60.7    80.76; ...
4   44.3    6895    6963    57.7    79.81; ...
5   44.3    6895    7486    52.9    76.94; ...
1   44.3    7757    6576    80.2    79.87; ...
2   44.3    7757    7307    79.7    80.57; ...
3   44.3    7757    8037    77.7    81.06; ...
4   44.3    7757    8768    73.4    80.09; ...
5   44.3    7757    9498    64.2    75.00; ...
1   44.3    8619    8910    97.8    79.04; ...
2   44.3    8619    9519    96.1    79.63; ...
3   44.3    8619    10128   93.4    79.88; ...
4   44.3    8619    10737   87.9    78.66; ...
5   44.3    8619    11345   71.8    70.93; ...
1   44.3    9050    10375   104.8   78.77; ...
2   44.3    9050    10795   102.8   78.89; ...
3   44.3    9050    11216   99.3    78.46; ...
4   44.3    9050    11637   93.5    76.97; ...
5   44.3    9050    12058   77.7    70.05; ...
1   47.5    6033    4713    47.7    79.98; ...
2   47.5    6033    5038    46.7    80.11; ...
3   47.5    6033    5363    45.3    79.83; ...
4   47.5    6033    5689    43.4    78.88; ...
5   47.5    6033    6014    41.0    77.04; ...
1   47.5    6895    5326    63.7    79.54; ...
2   47.5    6895    5969    63.2    80.42; ...
3   47.5    6895    6612    61.5    80.80; ...
4   47.5    6895    7256    58.2    79.92; ...
5   47.5    6895    7899    51.7    75.81; ...
1   47.5    7757    7158    80.5    78.89; ...
2   47.5    7757    7844    79.3    79.91; ...
3   47.5    7757    8529    77.3    80.62; ...
4   47.5    7757    9215    73.0    79.66; ...
5   47.5    7757    9901    60.5    72.35; ...
1   47.5    8619    10115   93.9    78.59; ...
2   47.5    8619    10461   91.9    78.43; ...
3   47.5    8619    10807   88.8    77.91; ...
4   47.5    8619    11153   83.8    76.49; ...
```

```
5   47.5    8619    11498   65.3    66.72; ...
1   47.5    9050    11266   98.5    77.04; ...
2   47.5    9050    11428   96.6    76.65; ...
3   47.5    9050    11591   94.2    76.05; ...
4   47.5    9050    11754   90.9    75.15; ...
5   47.5    9050    11916   85.8    73.35];

Morig=M;

M(:,2)=M(:,2)/max(M(:,2));
M(:,3)=M(:,3)/max(M(:,3));
M(:,4)=M(:,4)/max(M(:,4));
M(:,5)=M(:,5)/max(M(:,5));
M(:,6)=M(:,6)/max(M(:,6));


maxe=0;
maxpos=0;
for i=1:15
    for k=0:4
        if(M(1+((i-1)*5)+k,6)>maxe)
            maxpos=1+((i-1)*5)+k;
            maxe=M(1+((i-1)*5)+k,6);
        end;
    end;
    for l=0:4
        if(M(1+((i-1)*5)+l,4)<M(maxpos,4))
            M(1+((i-1)*5)+l,6)=-1;
        end;
        if(M(1+((i-1)*5)+l,4)>M(maxpos,4))
            M(1+((i-1)*5)+l,6)=1;
        end;
        if(M(1+((i-1)*5)+l,4)==M(maxpos,4))
            M(1+((i-1)*5)+l,6)=0;
        end;
    end;
    maxe=0;
    maxpos=0;
end;



% vektorer med de forskjellige variablene; "kjørenivå", molvekt, turtall, flow,
% head, og effekt

k=M(:,1);
m=M(:,2);
n=M(:,3);
q=M(:,4);
```

```
h=M(:,5);
e=M(:,6);





%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                  % X—matrisa nedenfor %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


teller=0;
for i=1:6
    if(modelparam(i)==1)
        teller=teller+1;
    end
end

HLP=[M(:,2),M(:,3),M(:,6),M(:,2).*M(:,3),M(:,2).*M(:,6),M(:,3).*M(:,6)];

X=zeros(75,teller+1);
X(:,1)=1;
ny=2;
for i=1:6
    if(modelparam(i)==1)
        X(:,ny)=HLP(:,i);
        ny=ny+1;
    end
end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                       %  Z—matrisa nedenfor  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%





teller=0;
for i=7:12
    if(modelparam(i)==1)
        teller=teller+1;
    end
end
```

```
ss20=teller+2;


r=5*3; % dette er antall grupper turtall innenfor hver molvekt multiplisert med an
k=5;
Z=zeros(75,(ss20-1)*r);


for i=1:r,
    Z((i-1)*k+1:i*k,i)=ones(k,1);
end;
ny=1;
for i=7:12
    if(modelparam(i)==1)
        for j=1:r
            Z((j-1)*k+1:j*k,(ny*15)+j)=HLP((j-1)*k+1:j*k,i-6);
        end
        ny=ny+1;
    end
end

data=[q h];
```