Statistical Analysis of Method Comparison Studies

Practical computer exercises

1.0

15/02/2008
MEGA Centre, SPH, University of Melbourne
www.biostat.ku.dk/~bxc/MethComp/Melb.2008

Bendix Carstensen — Steno Diabetes Center, Gentofte, Denmark
& Department of Biostatistics, University of Copenhagen
bxc@steno.dk
http://staff.pubhealth.ku.dk/~bxc/

Lyle Gurrin — School of Population Health, University of Melbourne
Carlton, Victoria, Australia
lgurrin@unimelb.edu.au
http://www.epi.unimelb.edu.au/about/staff/gurrin-lyle

Contents

Course program

Friday 15th February 2008

09:15 – 09:45	**Lecture 1**: Simple comparisons of measurement methods. Introduction to computing.
09:45 – 10:45	**Practical 1**: Limits of agreement, Bland-Altman-plots: • Milk • Plasma volume.
10:45 – 11:15	**Morning Tea**
11:15 – 12:00	**Lecture 2**: Designs with replicate measurements. Allocation of sources of variation. The general model.
12:00 – 13:00	**Lunch**
13:00 – 13:30	**Lecture 3**: Non-constant bias. Conversion between methods
14:30 – 16:00	**Practical 2**: Data with replicate measurements by each method. • Oximetry. • Systolic blood pressure. • HbA_{1c}

Chapter 1

Introduction to computing

Pwd The course is both theoretical and practical, i.e. the aim is to convey a basic understanding of the problems in method comparison studies, but also to convey practical skills in handling the statistical analysis.

The practicals assume that you bring your own laptop. In the following is a brief overview of the software and other files you must download.

1.1 Software

The most convenient software for desk-calculator type of calculations and simulation as well as simple statistical computing is the free software package R. Moreover the more advanced models covered in this course are only implemented in R.

In order to be able to write scripts (programs) in R and keep them for future use (and modification for other purposes) a good editor with interface to R is convenient. `Tinn-R` is the answer. (Tinn = Tinn Is Not Notepad).

1.1.1 R and `BRugs` / `R2WinBUGS`

`BUGS` (Bayesian inference Using Gibbs Sampling) is a programming language for specification of models that allow description in hierarchical terms, specifically as directed acyclic graphs (DAGs). It was first released in the 1990es for a Unix platform, but is now available in many guises for various platforms. `BUGS` is the generic name for any of these.

R is a free software package for statistics and graphics. R can be extended with *packages* that contains extra functions. Two packages that allows the user to access `BUGS` from within R are, `R2WinBUGS` and `BRugs`.

`R2WinBUGS` is an interface to `WinBUGS`, and requires that `WinBUGS` is installed on your computer separately from R. Calling `WinBUGS` requires that you specify a file with the BUGS code in it, as well as the exact location on the computer where `WinBUGS` is installed.

`BRugs` on the other hand is an R-package with includes a full version of `OpenBUGS`. Thus having installed `BRugs` you have in principle access to all relevant functions for this course.

However, `BRugs` was not quite stable when it first appeared around 2005, so the methods we shall use are only (so far) implemented using the `R2WinBUGS` interface. Hence you need to obtain `WinBugs`.

1.1.2 Installation

R can be obtained from www.r-project.org. Click on `CRAN`, choose a mirror (that is, from where

you want to download it), click on the link to Windows and after that choose `base`. Download `R-2.6.0-win32.exe` to your computer, and run this installation file.

Then fire up R, and at the command prompt type:

```
install.packages( c("R2WinBUGS","coda","Epi") )
```

This will install the three mentioned packages provided you are connected to the net.

`Epi` is a package designed for epidemiological use. It contains some functions for display of estimates that may be useful.

1.1.3 MethComp

Finally you will have to install the non-official package for R, `MethComp`[1], which contains all the . It is available from http://staff.pubhealth.ku.dk/~bxc/MethComp/Archive/?C=M;O=D — this link should bring up the latest version of the package at the top of the display. Download the file `MethComp_0.1.xx.zip` and unpack it in the folder `c:\Program Files\R\R-2.6.0\library`.

1.1.3.1 Using MethComp

When using the function `MethComp` there will be a call to `WinBUGS`, and therefore the place on your computer where `WinBUGS` is installed must be supplied. That can either be done in the call to the function:

```
MethComp( ..., bugs.directory="c:/Program Files/WinBUGS14" )
```

— or wherever you installed `WinBUGS`. But the default for `MethComp` is to look for the R-option `bugs.directory`. Therefore, if you start your R-session by saying:

```
options(bugs.directory="c:/Program Files/WinBUGS14")
```

you don't have to bother about this any more in you session.

Even more sophisticated, you can add the line defining the option to the file `.Rprofile` which you find in the folder `c:\Program Files\R\R-2.6.0\etc`. Then R will automatically set this option every time you fire it up.

1.1.4 WinBUGS

`WinBUGS` is needed in order to be able to run `R2WinBUGS`.

`WinBUGS` can be obtained from the `WinBUGS` homepage http://www.mrc-bsu.cam.ac.uk/bugs. `WinBUGS` will only work if you have a certificate which is free. To obtain one, register at the `WinBUGS` homepage and you will get an e-mail with the certificate and which tells you how to install the certificate.

1.1.5 Tinn-R

This text editor is free and can be downloaded from http://www.sciviews.org/Tinn-R. The advantage is that it has a coloured syntax-highlighting for R and a direct interface to R so that highlighted parts of the text can be sent directly to R.

[1] It will soon be an official package for R but it has only been under development during the last year.

Chapter 2

Introduction to the MethComp package

The purpose of the MethComp package is to provide computational tools to manipulate, display and analyze data from method comparison studies. The package lives off a particular structure of data.

2.1 Data structures

In general we are concerned with measurements by different methods, on different items (persons, samples), possibly replicated.

Often such data are represented by a row of measurements for each item, with possible replicates listed either below or beside each other. This implicitly assumes that some replicate measurements belong together, which is not necessarily the case in all situations.

Therefore all functions in MethComp assume data to be represented in the "long" form, with one measurement on each row, and columns to indicate method, item and replicate. Specifically, we assume the following columns are available in a data frame:

- `meth` The measurement method. Numeric or factor.
- `item` Identification of item (person, sample). Numeric or factor.
- `repl` Replicate number. Numeric.
- `y` The measurement by method `meth` on item `item`, replicate number `repl`.

There is no class attached to this kind of data frame, the functions in MethComp check whether the relevant columns are present.

2.2 Function overview

The following is a brief overview of the functions in the MethComp package. The full documentation is in the help pages for the functions, and an illustration of the way they work can be obtained by referring to the printed manual at the end of this document or on the fly by typing e.g.:

```
example( plot.meth )
```

There is no excuse for not reading the help file before using a function:

"If all else fails, RTFM".

2.2.1 Graphical functions

`BA.plot` Makes a Bland-Altman plot of two methods from a data frame with method comparison data, and computes limits of agreement. The plotting etc is really done by a call to

`BlandAltman` draws a Bland-Altman plot and computes limits of agreement.

`plot.meth` Plots all methods against all other, both as a scatter plot and as a Bland-Altman plot.

`bothlines` Adds regression lines of y on x and vice versa to a scatter plot.

2.2.2 Data manipulating functions

`make.repl` Generates a `repl` column in a data frame with columns `meth`, `item` and `y`.

`perm.repl` Randomly permutes replicates within (method,item) and assigns new replicate numbers.

`to.wide` Transforms a data frame in the long form to the wide form where separate columns for each method are generated, with one row per (item,replicate).

`to.long` Reverses the result of `to.wide`.

`tab.repl` Tabulates replicates by methods and items.

`sim.meth` Simulates a dataset from a method comparison experiment for given parameters for bias, exchangeability and variance component sizes.

2.2.3 Analysis functions

`Deming` Performs Deming regression, i.e. regression with errors in both variables.

`BA.est` Estimates in the variance components models underlying the concept of limits of agreement, and returns the bias and the variance components. Assumes constant bias between methods.

`MethComp` Estimates via `BUGS` in the general model with non-constant bias (and in the future) possibly non-constant standard deviations of the variance components. Produces a `MethComp` object.

2.2.4 Reporting functions

These functions all take a `MethComp` object as input.

`print.MethComp` Prints a table of conversion equation between methods analyzed, with prediction standard deviations. Also gives summaries of the posteriors for the parameters that constitute the conversion algorithms.

`plot.MethComp` Plots the conversion lines between methods with prediction limits.

`plot.VarComp` Plots smoothed posterior densities for the variance component estimates.

Chapter 3

Practical exercises

3.1 Milk: Single measurements by two methods

The `milk` data from the `MethComp` package contains measurements of fat content of human milk (g/100 ml) determined by the measurement of glycerol released by enzymic hydrolysis of triglycerides (Trig) and measurements by the standard Gerber method (Gerber).

Load the dataset and take a look at its structure:

```
> data(milk)
> str(milk)
```

The data is arranged in the long form, i.e. with one measurement per line and two variables, item and method. If you want to have the two methods beside each other, you can use the `to.wide` function:

```
> mw <- to.wide(milk)
> str(mw)
```

The purpose of this exercise is to assess to what degree the two methods can be used interchangeably, or rather to quantify how much they differ so that an informed clinical decision can be made.

Also it will introduce some ways that you can display data with the facilities in the `MethComp` package.

1. Plot the two sets of measurements against each other, e.g. by using the two variables from the dataset in the wide form.

2. To get an overview of the relationship you can exploit the fact that the dataset has variables `item`, `meth` and `y`, and get a general overview by `plot.meth`, try:

   ```
   > plot.meth(milk)
   ```

3. You can also be more explicit about the Bland-Altman comparing the two methods, try:

   ```
   > BA.plot(milk)
   > BA.plot(milk,ymax=0.5)
   ```

   You will want to have a look at the help page for `BA.plot` and also for `BlandAltman` which is the function that really does the plotting. Note that options from `BA.plot` is passed on to the function `BlandAltman`.

4. What are the limits of agreemnt between the wo methods?

5. Formulate in plain words what this means.

6. Inspect the plot and try to assess whether the assumtions underlying the reporting of limits of agreement are fulfilled. (*Hint:* Try to regress the differences on the average, and translate the resulting regression equation to a linear relationship between the two methods.

7. Fit the two regression lines (i.e. regress `Gerber` on `Trig` and vice versa) and show them in a plot of the two methods:

   ```
   > summary( lm( Trig ~ Gerber, data=mw ) )$coef
   > summary( lm( Gerber ~ Trig, data=mw ) )$coef
   ```

   How do they relate to the equation derived from regression of the difference on the average?

8. Finally, try to make a regression allowing for errors in both variables, the so-called Deming regression:

   ```
   > with( mw, Deming( Trig, Gerber ) )
   ```

   Compare this with the relationship derived from the regression of the difference on the average.

9. Use the results to provide an improved predition equation for Gerber based on a measured value by Trig.

10. How could you have derived the relevant standard deviation from the regression of the differences on the averages?

3.2 Plasma volume: Single measurements by two methods

The `plvol` data from the `MethComp` package contains measurements of plasma volume is expressed as a percentage of the expected value for normal individuals.

1. Plot the the measurements from the two methods against each other.
2. Make a Bland-Altman plot and compute the limits of agreement. Try: @

   ```
   > BA.plot(plvol)
   ```

   Are these limits a reasonable summary of the data?
3. Make a log-transform of the data and re-do the analysis.
   Hint: You may use the `mult=TRUE` option to `BA.plot` to acheive this:@

   ```
   > BA.plot(plvol,mult=TRUE)
   ```

   Note that the explanation of the parameter `mult` is not on the help page for `BA.plot` but in that for `BlandAltman`.

   Does the log-transform give a better description of data?
4. Formulate a conclusion for the data in plain words, based on the log-transformed analysis.

3.3 Systolic blood pressure I: Replicate measurements by two methods

1. Load the systolic blood pressure data from the `MethComp` package, and take a look at the data using `?sbp`, `str()` and `plot.meth()`:

   ```
   > data(sbp)
   > str(sbp)
   > plot.meth(sbp)
   ```

   What is the immediate impression of the relationship of the methods to each other?

   How are the replicate measuremnts handeled?

2. We want to restrict our attention to the comparison of the two manual methods (J and R), but still using the replicate measurements.

   Are the replicates exchangeable within method and item?

   Make a Bland-Altman plot of the data for the two manual methods, and derive the limits of agreement, e.g.:

   ```
   > sbp <- subset( sbp, meth %in% c("J","R") )
   > BA.plot( sbp )
   ```

   Try to use the argument `ymax=` (the meaning of this is found on the help page for `BlandAltman()`).

   How does the use of the replicates for this Bland-Aætman plot and limits of agreemnt correspond to the exchangeability structure of data?

3. Fit the proper model for the data, reflecting the exchangeability of replicates:

   ```
   > sbp$item <- factor( sbp$item )
   > m1 <- lme( y ~ meth + item,
   +          random=list( item = pdIdent( ~ meth-1 ) ),
   +          weights = varIdent( form = ~1 | meth ),
   +          data = sbp )
   > m1
   ```

   Find the bias between methods, as well as the variance components.

4. A more direct way of getting at the variance components is to use the wrapper `BA.est()`, try e.g.:

   ```
   > BA.est( sbp, linked=FALSE )
   ```

5. Use these estimates to contruct limits of agreement for the difference J−R, and compare with the limits obtained by using the paired replicates as items.

6. What can be the possible explanantion for the dramatic differences?

   Try to plot the measurements for each person against replicate number, separately for each method. It may be easier to read the plots if you only do them for two methods at a time.

7. An alternative way of seeing the lack of exchangeability is to make the overview plot using a random permuation of the replicates. If replicates were truely exchangeable within methods the plot would look similar when permuting the replicates. Try to use the function `perm.repl()` to make a random permutation of replicates — for the sake of completeness reload the dataset so you have all three methods available:

```
> data(sbp)
> str(sbp)
> plot.meth(perm.repl(sbp))
```

8. What would be the correct analysis of these data? Use the function `BA.est()` to get the bias and the correct variance components. In particular, discuss the relative size of the variance components.

9. Compute limits of agreement based on the variance components from the model for the entire dataset.

10. Formulated this as a 95% prediction interval for R given a measurement by J, y_J.

11. Fit the model on the dataset with only measuremnt by the two physicians and compute the limits of agreement based on estimates from this. Compare with the previously computed limits of agreement.

3.4 Oximetry: Linked replicates with non-constant bias

The `ox` data from the `MethComp` package contains data from 61 children who had their blood oxygen content measured using two methods at the Royal Children's Hospital in Melbourne. The standard chemical method analysing gases in the blood based on co-oximetry (named "CO") is to be compared to a new method using a pulse oximeter to measure light reflectance transcutaneously (named "pulse"). Most children have three replicates on each method, which are linked, so replicate 1 for each of the two methods is done at the same time. Replicate measurements were taken in quick succession, so we assume that the linked pairs of measurements are exchangeable within person.

The purpose of this exercise is to demonstrate the facility in the `MethComp` package to estimate the variance between linked replicates (the item by replicate effect) while allowing for a random method by item effect and differing residual variances between methods. We also consider the possibility of non-constant bias.

1. Start by loading the dataset and take a look at its structure:

```
> library(MethComp)

> data(ox)
> str(ox)

'data.frame':        354 obs. of  4 variables:
 $ meth: Factor w/ 2 levels "CO","pulse": 1 1 1 1 1 1 1 1 1 1 ...
 $ item: num  1 1 1 2 2 2 3 3 3 4 ...
 $ repl: num  1 2 3 1 2 3 1 2 3 1 ...
 $ y   : num  78 76.4 77.2 68.7 67.6 68.3 82.9 80.1 80.7 62.3 ...

> head(ox)

  meth item repl    y
1   CO    1    1 78.0
2   CO    1    2 76.4
3   CO    1    3 77.2
4   CO    2    1 68.7
5   CO    2    2 67.6
6   CO    2    3 68.3
```

   The dataframe is already in the correct form for use with the `MethComp` package, with variables named `item`, `meth`, `repl` and `y`. Plot the two sets of measurements against each other using the `plot.meth` function:

```
> plot.meth(ox)

Note:
 Replicate measurements are taken as separate items!
```

2. Use the `BA.plot` function to generate a Bland-Altman plot of these data. What is the estimated average difference between measurements from the two methods? What are the limits of agreement between the two methods?

```
> par(mfrow = c(1,1))
> BA.plot(ox)
```

```
Limits of agreement with 95 % confidence intervals:

CO-pulse           Estimate      2.5 %    97.5 %
  mean difference  2.477401   1.561313  3.393489
  upper limit     14.581351  12.994640 16.168061
  lower limit     -9.626548 -11.213259 -8.039838
```

Are these limits large compared to the average oximetry measure and the range of the data?

3. The Bland-Altman procedure for generating the limits of agreement is based on a model with constant bias. Moreover, it does not divide the variation between different sources. With replicate measurements we can allocate the variation to the different sources using a variance component model:

   - method by item ("matrix" effect).
   - item by replicate (variation between linked sets).
   - residual variation for each method.

   The model can be fit by using the function `BA.est()`:

```
> BA.est(ox)

$bias
       CO     pulse
 0.000000 -2.470446

$sd.s
     MxI.CO   MxI.pulse         IxR    resid.CO resid.pulse
   2.928042    2.928042    3.415692    2.224868    3.994451
```

   Make sure that you understand what each of the variance components mean.

4. Note that the `MI` variance components are the same for `CO` and `pulse` since separate parameters cannot be estimated when there are only two methods. Compare the magnitude of the `IR` variance component for the item by replicate effect to both the `MI` variance component for the method by item effect and the residuals variances. Is this what you would expect given that the replicates are linked?

5. One of the drawbacks of using the `BA.est` routine is that we do not get standard errors or confidence intervals for the estimated variance parameters. The `MethComp` function produces summaries of the posterior distribution of estimated parameters in a Bayesin setup.

   You must use the argument `beta = FALSE` in the call to `MethComp` to fit a model with constant bias:

```
> ox.mi.ir <- MethComp( ox, random=c("mi","ir"), n.iter=5000, beta = FALSE )

Comparison of 2 methods, using 354 measurements
on 61 items, with up to 3 replicate measurements.
( 2 * 61 * 3 = 366 ):

No. items with measurements on each method:
        #Replicates
Method  1 2  3 #Items #Measurements: 354
  CO    1 4 56     61                177
  pulse 1 4 56     61                177
```

```
Simulation run of a model with
- fixed bias (beta==1)
- method by item and item by replicate interaction:
- using 3 chains run for 5000 iterations
  (of which 2500 are burn-in),
- monitoring every 3 values of the chain:
- giving a posterior sample of 2500 observations.
```

Summarise the results by using the `print` function on the resulting `MethComp` object `ox.mi.ir`:

```
> print(ox.mi.ir)

Conversion formula:
 y_to = alpha + beta * y_from +/- 2*sd.pred:

      From:     CO                 pulse
             alpha   beta sd.pred  alpha   beta sd.pred
To:
CO           0.000  1.000   2.205  2.482  1.000   6.235
pulse       -2.482  1.000   6.235  0.000  1.000   4.091

Variance components (standard deviations):
                     50%   2.5%  97.5%     0%   100%
sigma.mir[CO]    2.2055 0.7159 3.0194 0.2165 3.5280
sigma.mir[pulse] 4.0910 3.4305 4.8675 2.9490 6.0040
sigma.mi[CO]     2.9270 2.2020 3.7920 1.8020 4.3260
sigma.mi[pulse]  2.9270 2.2020 3.7920 1.8020 4.3260
sigma.ir[CO]     3.4830 2.7935 4.3135 2.2370 4.9110
sigma.ir[pulse]  3.4830 2.7935 4.3135 2.2370 4.9110
sigma.tot[CO]    5.0900 4.5012 5.7433 4.1058 6.2836
sigma.tot[pulse] 6.1425 5.4747 7.0446 5.1625 7.9153
```

Use the `plot` function for `MethComp` objects to produce a scatterplot displaying the linear equations relating one method to the other (recall that the slope has been constrained to be 1):

```
> plot(ox.mi.ir, pl.obs = TRUE)
```

Use the `plot.VarComp` function to display smoothed posterior densities for the variance components separately for each method (although only the residual variances differ between methods):

```
> par(mfrow = c(2,1))
> plot.VarComp(ox.mi.ir)
```

Are the residual variances equal?

6. Expand the mode to allow for non-constant bias. This is the default option for `MethComp`, so you can just omit the `beta` argument:

```
> ox.mi.ir.beta <- MethComp( ox, random=c("mi","ir"), n.iter=5000)

Comparison of 2 methods, using 354 measurements
on 61 items, with up to 3 replicate measurements.
( 2 * 61 * 3 = 366 ):

No. items with measurements on each method:
        #Replicates
Method  1 2  3 #Items #Measurements: 354
```

```
  CO    1 4 56     61                177
  pulse 1 4 56     61                177

Simulation run of a model with
- method by item and item by replicate interaction:
- using 3 chains run for 5000 iterations
  (of which 2500 are burn-in),
- monitoring every 3 values of the chain:
- giving a posterior sample of 2500 observations.
```

Summarise the results of the `MethComp` fit and use the `plot.MethComp` function to display the equations relating the mean measurements on each method as above.

```
> print(ox.mi.ir.beta)

Conversion formula:
 y_to = alpha + beta * y_from +/- 2*sd.pred:

      From:     CO                  pulse
             alpha   beta sd.pred  alpha   beta sd.pred
To:
CO           0.000  1.000   1.482 -6.235  1.120  15.191
pulse        5.566  0.893  13.453  0.000  1.000   4.287

Variance components (standard deviations):
                     50%   2.5%   97.5%     0%    100%
sigma.mir[CO]     1.4815 0.2810  2.7809 0.1581  3.4320
sigma.mir[pulse]  4.2865 3.5560  5.0119 2.5730  5.8590
sigma.mi[CO]     10.1434 2.8958 13.2072 2.1610 16.7080
sigma.mi[pulse]   8.9478 2.6664 11.5598 2.0408 13.9946
sigma.ir[CO]      3.9445 3.0488  4.6780 2.4844  5.1922
sigma.ir[pulse]   3.4640 2.7641  4.2907 2.4227  5.3182
sigma.tot[CO]    10.9972 5.0623 13.8480 4.5781 17.2204
sigma.tot[pulse] 10.5170 5.8946 12.8387 5.3765 14.9000

> plot(ox.mi.ir.beta, pl.obs = TRUE)
```

Is $\beta_{2\cdot1}$ different from 1.00?

7. What are the implications for comparing oximetry measurements made on the same infant?

3.5 Measurement of HbA_{1c} I: Machine and specimen as method

The `hba1c` data from the `MethComp` package contains measurements of HbA_{1c}, i.e. the fraction of the hemoglobin in the blood that is glycosylated, and is usually reported a s a percentage. Glycosylation of the hemoglobin depends on the glucose (sugar) concentration in the blood. The red blood cells that contain the hemoglobin have an average lifetime of 3 months, so HbA_{1c} is therefore a marker of long term (i.e. 3 month) blood glucose regulation. It is used for monitoring of diabetes patients — normal person have a level of HbA_{1c} about 4–5% whereas diabetes usually have higher values, the normal treatment target for HbA_{1c} is a value below 6.6%.

At Steno Diabetes Center, HbA_{1c} is monitored routinely for all patients, and the laboratory therefore has a machine to analyze blood samples for HbA_{1c}. At a certain point the machine (Biorad, version Classic `BR.VC`) were to be replaced, so two candidate machines were brought in and blood samples from a number of patients were measured on all three machines. Blood was sampled both as capillary blood and venous blood. Finally blood was stored an analyze on different days.

The primary aim of the study was to investigate which of the machines were the more accurate, secondary aims to see if there were substantial differences between measurements based on capillary and venous blood and finally to provide a conversion algorithm between "old" measurements and "new" measurements to avoid breaks in the clinical series for patients.

1. Load the `hba1c` data and take a look at the structure, e.g.:

   ```
   > data( hba1c )
   > with( hba1c, table( d.samp, d.ana ) )
   > with( hba1c, table( dev, type, d.ana ) )
   ```

2. Note that the dataset does not have the standard structure, it lacks a definition of method and replicate. Provide these by using the interaction between `dev` and `type` and the day of analysis as replicate number.

   You may want to use the function `transform` and to create the interaction, the function (surprise, surpise) `interaction`, ie. create an updated dataframe, `hb`, say:

   ```
   > hb <- transform( hba1c, meth = interaction( dev, type ),
   +                         repl = d.ana )
   ```

3. Make an overview plot of the data in order to get an impression of the likely variations worth considering:

   ```
   > plot.meth( hb )
   ```

   What is the major first impression of the precision and relative bias of the different instruments?

4. Can we consider the replicates exchangeable within methods?

5. Specify a "standard model" for analyzing these data and fit it using `MethComp`:

   ```
   > m0 <- MethComp( hb )
   ```

   Remember to to put it into an object; the result is quite large, and therefore it is more handily represented by its default print method, so just type the name of the

   ```
   > m0
   ```

6. There is a zillion arguments to `MethComp`, but for a start we just use the default settings — in "real" applications one would use a larger number of iterations in order to be on the safe side. Since there are 6 methods we can plot the variance components associated with each of them in a 2 by 3 layout, try:

```
> par(mfrow=c(2,3))
> plot.VarComp(m0)
```

7. The posterior distributions of the variance components may not be very well determined, so try to re-fit the model using substantially more iterations. Also, try to enclose the call to `MethComp` in a `system.time()` in order to see how much time it takes, e.g.:

```
> system.time(
+ m1 <- MethComp( hb, n.iter=1000, n.chains=5 )
+            )
```

8. After a longer simulation try to do a more detailed plot by fiddling the graphics parameters a bit:

```
> par(mfrow=c(2,3),mar=c(3,1,2,1),mgp=c(3,1,0)/1.6)
> plot.VarComp(m1,grid=seq(0,1.5,0.1))
```

9. Try to form conclusions about the machines and speciemns based on the posterior distributions of the variance components.

Chapter 4

Solutions to exercises

4.1 Milk: Single measurements by two methods

First we load the dataset and take a look at its structure:

```
> data(milk)
> str(milk)

'data.frame':        90 obs. of  3 variables:
 $ meth: Factor w/ 2 levels "Gerber","Trig": 2 2 2 2 2 2 2 2 2 2 ...
 $ item: int  1 2 3 4 5 6 7 8 9 10 ...
 $ y   : num  0.96 1.16 0.97 1.01 1.25 1.22 1.46 1.66 1.75 1.72 ...

> head(milk)

  meth item    y
1 Trig    1 0.96
2 Trig    2 1.16
3 Trig    3 0.97
4 Trig    4 1.01
5 Trig    5 1.25
6 Trig    6 1.22
```

The data is arranged in the long form, i.e. with one measurement per line and two variables, item and method. Using the `to.wide` function puts the dat in a more familiar format:

```
> mw <- to.wide(milk)
> str(mw)

'data.frame':        45 obs. of  4 variables:
 $ item  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ id    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Trig  : num  0.96 1.16 0.97 1.01 1.25 1.22 1.46 1.66 1.75 1.72 ...
 $ Gerber: num  0.85 1 1 1 1.2 1.2 1.38 1.65 1.68 1.7 ...
 - attr(*, "reshapeWide")=List of 5
  ..$ v.names: chr "y"
  ..$ timevar: chr "meth"
  ..$ idvar  : chr "id"
  ..$ times  : Factor w/ 2 levels "Gerber","Trig": 2 1
  ..$ varying: chr [1, 1:2] "Trig" "Gerber"

> head(mw)

  item id Trig Gerber
1    1  1 0.96   0.85
```

```
2    2  2 1.16   1.00
3    3  3 0.97   1.00
4    4  4 1.01   1.00
5    5  5 1.25   1.20
6    6  6 1.22   1.20
```

1. We plot the two sets of measurements against each other, using the two variables from the dataset in the wide form:

   ```
   > with( mw, plot( Trig ~ Gerber, pch=16,
   +                 xlim=(lims<-range(c(Trig,Gerber))), ylim=lims ) )
   > abline(0,1)
   ```

   The last statement just adds the identity line.

2. Exploiting that the `milk` dataset has variables `item`, `meth` and `y`, we can get a general overview using:

   ```
   > plot.meth(milk,var.names=TRUE)
   ```

   Note the use of the `var.names=` argument to annotate the plot with the variable names to avoid confusion of what is on the axes.

3. We can get a proper Bland-Altman plot with a explicit calculation of the limits of agreement (including the uncertainty in these as devised in [**?**]):

   ```
   > BA.plot(milk)

   Limits of agreement with 95 % confidence intervals:
   ```

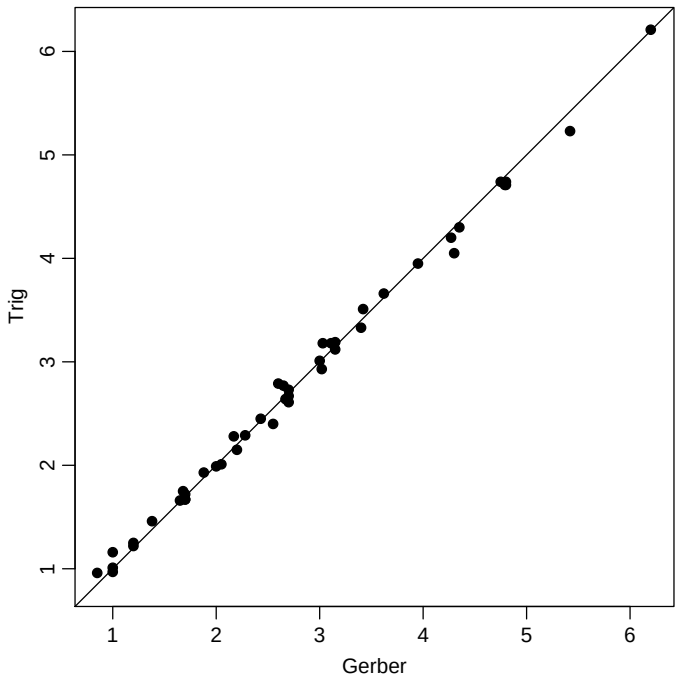

Figure 4.1: *Scatter plot of the* `milk` *data.*

```
Gerber-Trig            Estimate       2.5 %      97.5 %
  mean difference  0.0002222222 -0.02600408  0.02644852
  upper limit      0.1713171325  0.12589185  0.21674241
  lower limit     -0.1708726880 -0.21629797 -0.12544741
```

```
> BA.plot(milk,ymax=0.5)

Limits of agreement with 95 % confidence intervals:

Gerber-Trig            Estimate       2.5 %      97.5 %
  mean difference  0.0002222222 -0.02600408  0.02644852
  upper limit      0.1713171325  0.12589185  0.21674241
  lower limit     -0.1708726880 -0.21629797 -0.12544741
```

4. From the figure and from printout, we see that the limits of agreement are $(-0.17, 0.17$g/100 ml.

5. This means that the difference between future measurements by Gerber and Trig with 95% probability will be between -0.17 and 0.17 g/100ml.

6. The Bland-Altman plot looks very nice with an average that is very flat. Regressing the differences on the averages gives:

```
> summary( lm( I(Gerber-Trig) ~ I((Gerber+Trig)/2), data=mw ) )$coef
```

```
                        Estimate Std. Error   t value    Pr(>|t|)
(Intercept)          -0.07904017 0.02906123 -2.719781 0.009386433
I((Gerber + Trig)/2)  0.02827097 0.00944454  2.993367 0.004559424
```

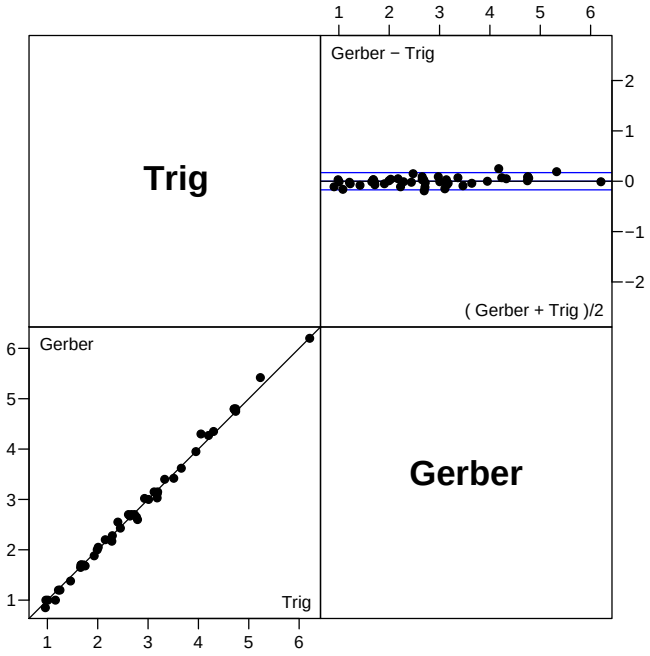

Figure 4.2: *Overview plot of the* `milk` *data.*

Strangely enough, the slope is significant, although the resulting relationship is not impressive. In general we have:

$$y - x = \alpha + \beta\left(\frac{x+y}{2}\right) \qquad \Longleftrightarrow \qquad y = \frac{\alpha}{1-\beta/2}\frac{1+\beta/2}{1-\beta/2}x$$

so the regression coefficients of the difference on the mean ($\alpha = -0.078, \beta = 0.028$) implies a relationship:

$$\text{Gerber} = -0.072/(1-0.014) + (1+0.014)/(1-0.014)\text{Trig} = -0.073 + 1.028\text{Trig}$$

This type of regression is tantamount to minimizing the squared deviations orthogonal to the identity line.

The range of the measurements is broadly speaking from 1 to 5, i.e. the contribution of the slope is about 0.15, largely in the same ballpart as the limits of agreement.

7. The two regression lines shows the same as the result of regression differences on means:

```
> summary( lm( Trig ~ Gerber, data=mw ) )$coef

               Estimate  Std. Error    t value     Pr(>|t|)
(Intercept) 0.08308899 0.028301786   2.935821 5.323062e-03
Gerber      0.97028609 0.009174537 105.758594 1.323266e-53

> summary( lm( Gerber ~ Trig, data=mw ) )$coef

               Estimate Std. Error    t value     Pr(>|t|)
(Intercept) -0.07456776 0.02980128  -2.502167 1.622649e-02
Trig         1.02667683 0.00970774 105.758594 1.323266e-53
```

We can plot the two lines using the function `bothlines`:

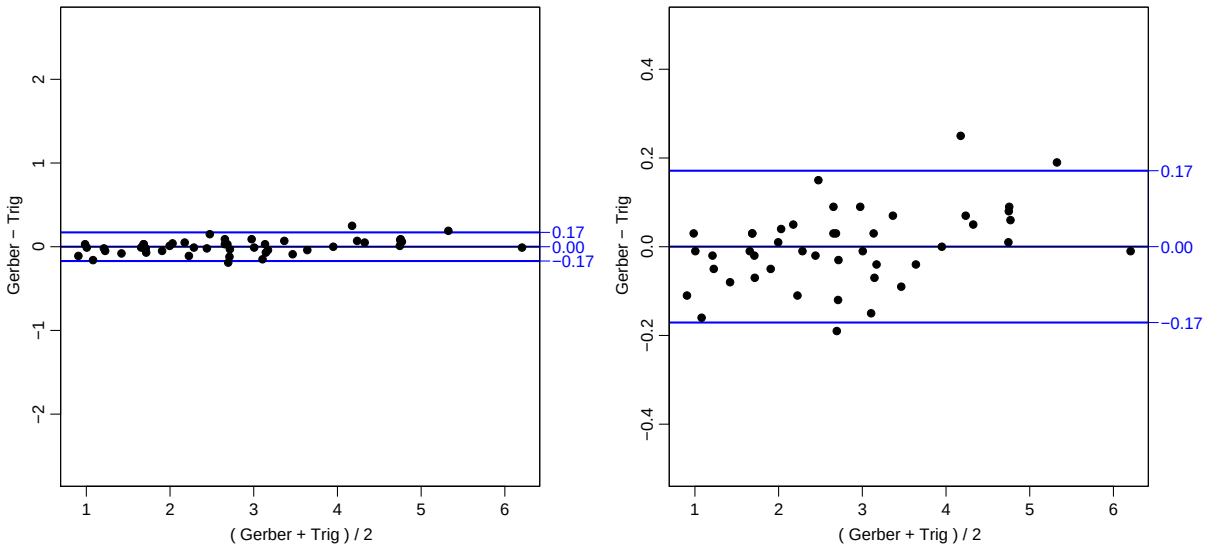

Figure 4.3: *Bland-Altman plots of the* `milk` *data, left panel with the sam extent of the data on both axes, the right one with explicitly defined y-axis.*

```
> with( mw, plot( Trig, Gerber, pch=16, xlim=c(0,6), ylim=c(0,6) ) )
> abline(0,1)
> with( mw, bothlines( Trig, Gerber ) )
```

8. A regression allowing for errors in both variables, the so-called Deming regression:

```
> with( mw, Deming( Trig, Gerber ) )

   Intercept        Slope   sigma.Trig sigma.Gerber
-0.08025171   1.02870424   0.05679647   0.05679647
```

   gives a result which is very close to that from the ordinary regression.

9. The advantage of this is that it provides an estimate of the residual standard deviation, which can be used for construction of prediction limits. Using the estimated residual standard deviation from this regresson, 0.057 as the basis for a prediction interval from one method to another would give a relationship:

$$\text{Gerber} = -0.080 + 1.029\text{Trig} \pm 1.96 \times 0.057 == -0.080 + 1.029\text{Trig} \pm 0.111$$

   i.e. limits that are only 2/3 as wide as those based on a constant difference between methods.

10. The residual standard deviation could also have been derived from the regression of the difference on the averages This gave a residual standard deviation of 0.08033, so the estimated standard deviation for each of the components (assuming that it is the same) would be $0.08033/sqrt2 = 0.0568$.

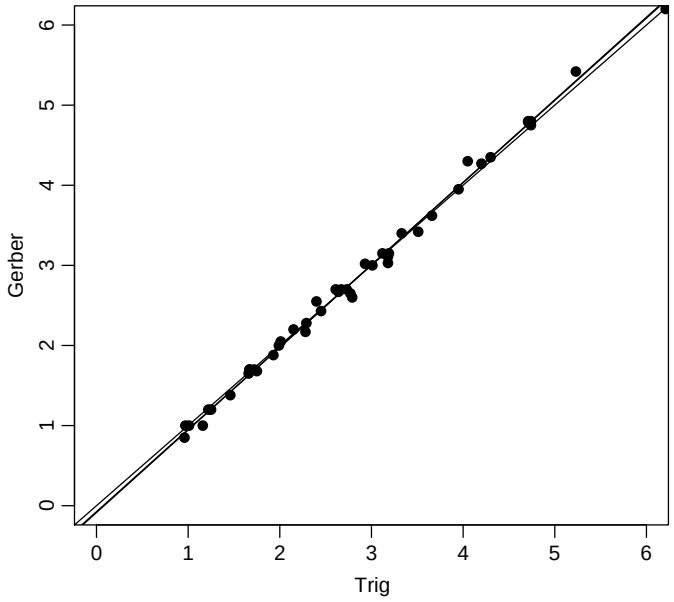

Figure 4.4: *Scatter plot of data with the two different regression lines. They are parctically indestinguishable.*

4.2 Plasma volume: Single measurements by two methods

The `plvol` data from the `MethComp` package contains measurements of plasma volume is expressed as a percentage of the expected value for normal individuals, measured by two different methods,

1. The two methods plotted against each other:@

```
> pw <- to.wide(plvol)
> with(pw, plot( Hurley ~ Nadler, pch=16, xlim=range(plvol$y), ylim=range(plvol$y) ) )
> abline( 0,1 )

null device
          1
```

2. `BA.plot` produces a Bland-Altman plot and computes the limits of agreement (we use the `ymax` argument to get a sensible range on the y-axis — otherwise the extent is as the x-axis):@

```
> BA.plot(plvol,ymax=15)

Limits of agreement with 95 % confidence intervals:

Nadler-Hurley       Estimate      2.5 %     97.5 %
  mean difference  -9.262626  -9.741879  -8.783373
  upper limit      -4.553001  -5.383091  -3.722911
  lower limit     -13.972252 -14.802342 -13.142162
```

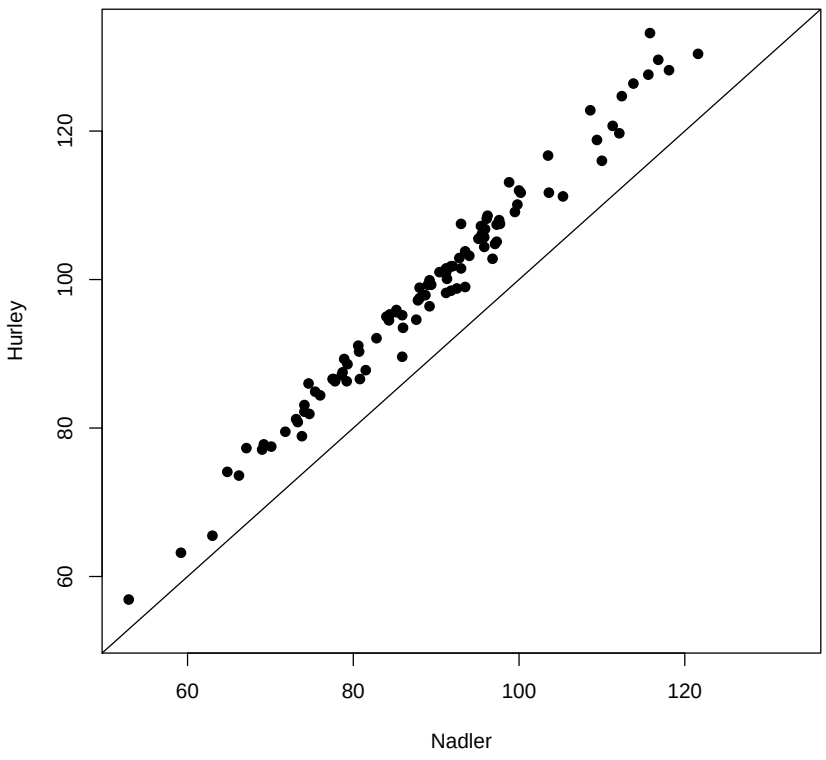

Figure 4.5: *Plot of two methods of measuring plasma volume.*

```
null device
          1
```

Clearly, there is both a decreasing differnce with increasing value of the plasma volume as well as an increase in variance with measurement level, hence these limits do not provide a reasonable summary of the data — they are much wider than necessary for a given level of the plasma volume.

3. If we log-transform the data and re-do the analysis we may get something more sensible. We can use the `mult=TRUE` option to `BA.plot` to acheive this in one go:@

```
> BA.plot(plvol,mult=TRUE,ymax=log(1.25))

Limits of agreement with 95 % confidence intervals:

Nadler/Hurley  Estimate     2.5 %    97.5 %
  mean ratio  0.9058334 0.9019213 0.9097625
  upper limit 0.9451922 0.9381330 0.9523045
  lower limit 0.8681136 0.8616301 0.8746460

null device
          1
```

It is immediately apparent from the plot The log-transform gives a much better description of data.

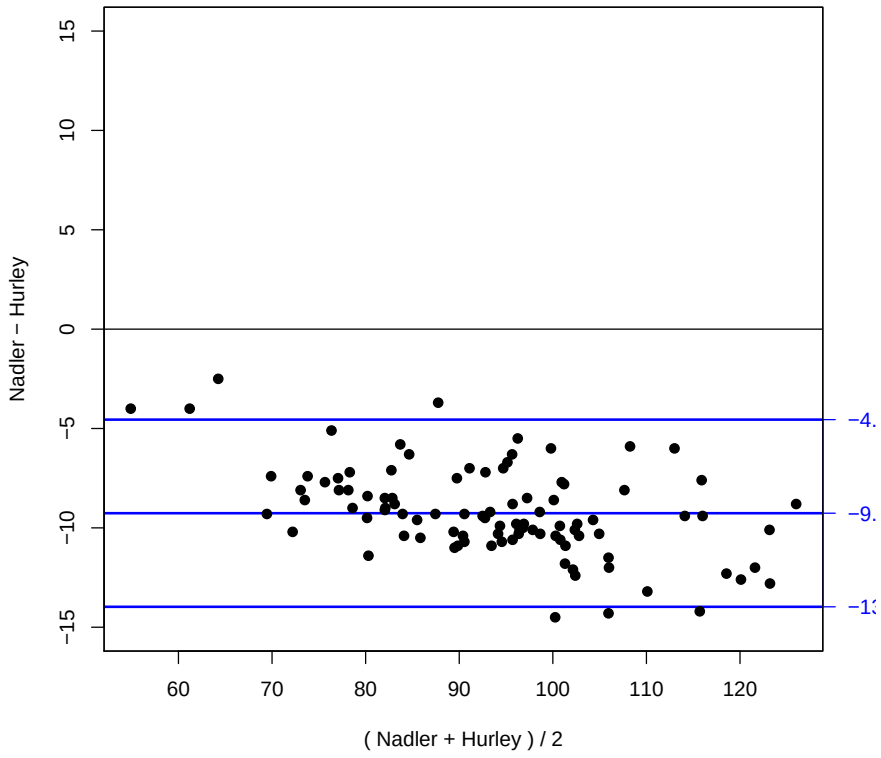

Figure 4.6: *Bland-Altman plot of two methods of measuring plasma volume.*

4. The estimated *ratio* between the Nadler and Hurley methods is 0.90 and the ratio of future measurements by the two methods is with 95% probability between 0.87 and 0.95. Alternatively, we may sat that for a given measurment by the Hurley method, the Nadler method will with 95% yield a measurement which is between 87 and 95% of this.

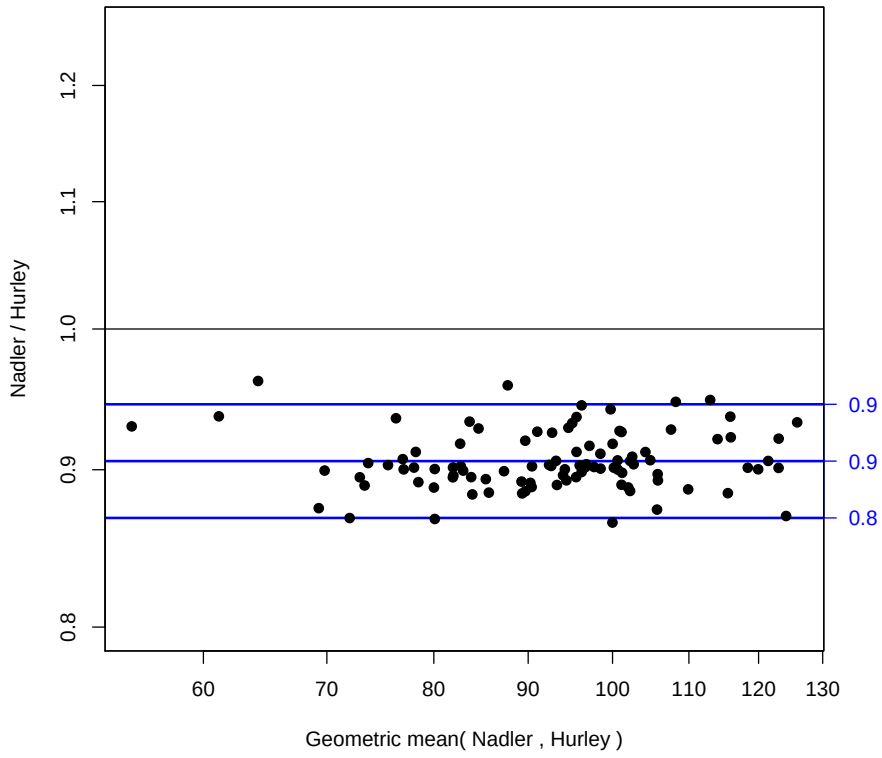

Figure 4.7: *Bland-Altman plot of two methods of measuring plasma volume, using log-transformed data, i.e. a relative scale.*

4.3 Systolic blood pressure I: Replicate measurements by two methods

1. We first load the systolic blood pressure data from the `MethComp` package.

```
> data(sbp)
> str(sbp)

'data.frame':        765 obs. of  4 variables:
 $ meth: Factor w/ 3 levels "J","R","S": 1 1 1 1 1 1 1 1 1 1 ...
 $ item: num  1 2 3 4 5 6 7 8 9 10 ...
 $ repl: num  1 1 1 1 1 1 1 1 1 1 ...
 $ y   : num  100 108 76 108 124 122 116 114 100 108 ...

> plot.meth(sbp)

Note:
 Replicate measurements are taken as separate items!
```

   Clearly the two manual measurements are in much closer agreement than any of them are with the automatic.

2. We want to restrict our attention to the comparison of the two manual methods, but using the replicate measurements.

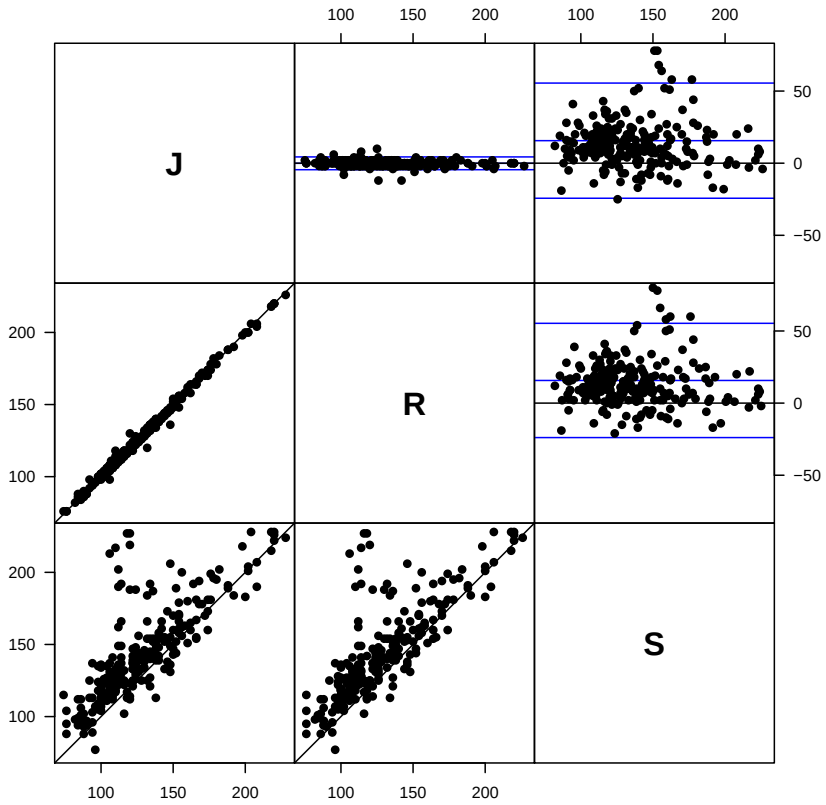

Figure 4.8: *Graphical overview of the* `sbp` *data. The methods J and R are two human observers, whereas method S is an automatic device.*

In this context it is important that we recognize whether the replicates are linked across the two methods or not. In this case they are not, *i.e.* replicates are exchangeable within methods and items.

```
> sbp <- subset( sbp, meth %in% c("J","R") )
> str( sbp )

'data.frame':        510 obs. of  4 variables:
 $ meth: Factor w/ 3 levels "J","R","S": 1 1 1 1 1 1 1 1 1 1 ...
 $ item: num  1 2 3 4 5 6 7 8 9 10 ...
 $ repl: num  1 1 1 1 1 1 1 1 1 1 ...
 $ y   : num  100 108 76 108 124 122 116 114 100 108 ...

> BA.plot( sbp )

Limits of agreement with 95 % confidence intervals:

J-R                  Estimate      2.5 %     97.5 %
  mean difference  0.08627451 -0.1925236  0.3650726
  upper limit      4.51711010  4.0342176  5.0000026
  lower limit     -4.34456108 -4.8274536 -3.8616686
```

A slightly more informative plot can be obtained by explicitly regulatin the y-dimension of the plot by the argument `ymax=`:

```
> BA.plot( sbp, ymax=10 )

Limits of agreement with 95 % confidence intervals:

J-R                  Estimate      2.5 %     97.5 %
  mean difference  0.08627451 -0.1925236  0.3650726
  upper limit      4.51711010  4.0342176  5.0000026
  lower limit     -4.34456108 -4.8274536 -3.8616686
```

3. In order to proper partition the variance and produce limits of agreement or a translation between the two observers, we should fit the relevant variance component model, assuming exchangeable replicates:

$$y_{mir} = \alpha_m + \mu_i + c_{mi} + e_{mir}, \qquad c_{mi} \sim \mathcal{N}(0, \tau_m^2), \quad e_{mir} \sim \mathcal{N}(0, \sigma_m^2)$$

However, since we only have two methods, we cannot identify separate variance components τ_1 and τ_2, so we are forced to assume that $\tau_1 = \tau_2$. The model above is fitted to the dataset by (note that we must first assure that item is a factor):

```
> sbp$item <- factor( sbp$item )
> m1 <- lme( y ~ meth + item,
+          random=list( item = pdIdent( ~ meth-1 ) ),
+          weights = varIdent( form = ~1 | meth ),
+          data = sbp )
> m1

Linear mixed-effects model fit by REML
  Data: sbp
  Log-restricted-likelihood: -1406.537
  Fixed: y ~ meth + item
 (Intercept)        methR        item2        item3        item4        item5
103.38260819  -0.08627451   5.98930951 -22.01069049   1.99358570  13.65383807
       item6        item7        item8        item9       item10       item11
```

```
 25.98930951   5.98930951   7.99786190   2.99572380  -2.34616193   0.98717141
      item12       item13       item14       item15       item16       item17
 10.99144761   8.32478094   1.99358570   1.33119523  15.33119523  -2.00641430
      item18       item19       item20       item21       item22       item23
 14.66452857  33.33119523  43.33119523  53.33547143  40.01069049  66.00213810
      item24       item25       item26       item27       item28       item29
 60.65597617  39.32691904  27.32691904  37.66239047  45.32691904 115.99358570
      item30       item31       item32       item33       item34       item35
 95.66666667 -15.00855239  14.99144761  18.66452857  22.00213810  15.99358570
      item36       item37       item38       item39       item40       item41
-12.66880477   4.32478094 105.33119523  25.00000000  30.66025237  -9.00427620
      item42       item43       item44       item45       item46       item47
 -8.00641430  17.65811427  58.66025237  -2.01069049  24.32905714  11.32691904
      item48       item49       item50       item51       item52       item53
 31.31836664  49.32691904 -11.67521906  52.66452857  -1.33974763   1.32264284
      item54       item55       item56       item57       item58       item59
 -4.34188573 -24.01069049   1.66239047   5.65383807  75.65383807  52.99572380
      item60       item61       item62       item63       item64       item65
 35.99786190  93.65811427 -11.67094286  24.32905714  36.99572380  33.66239047
      item66       item67       item68       item69       item70       item71
 53.98930951  29.66239047   9.65811427  13.32691904  17.65811427 112.66452857
      item72       item73       item74       item75       item76       item77
 30.66025237  53.99358570 -19.33974763  70.65597617  75.66239047  13.32691904
      item78       item79       item80       item81       item82       item83
 15.33119523   4.66025237   6.32905714  36.98717141   4.98717141   6.99572380
      item84       item85
 -2.00641430  12.65597617

Random effects:
 Formula: ~meth - 1 | item
 Structure: Multiple of an Identity
              methJ        methR Residual
StdDev: 0.0002614756 0.0002614756 5.528081
```

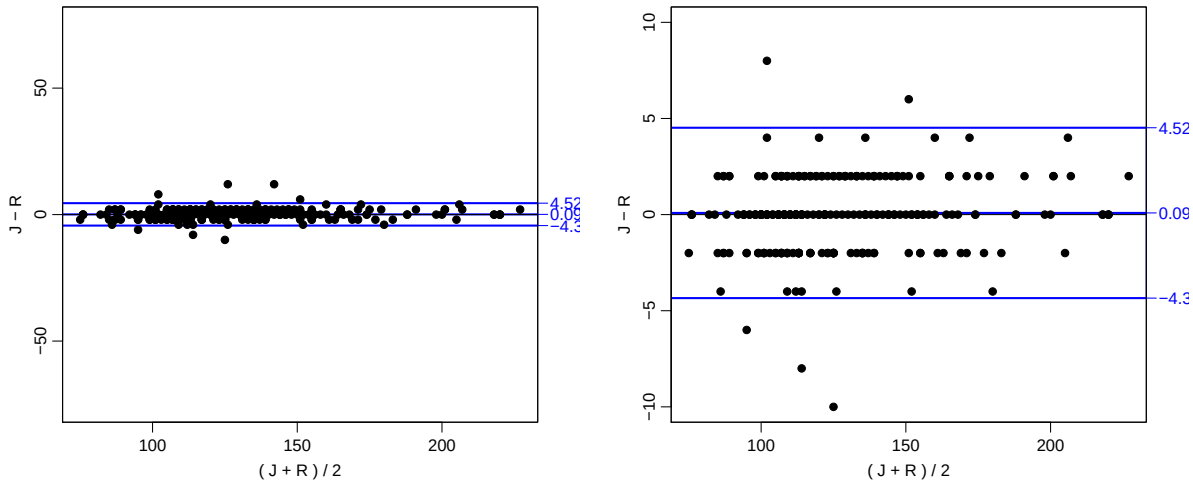

Figure 4.9: *Bland-Altman plot of the* sbp *data. Note that although replicates are exchangeable within method, the ordering of the replicates in the data has been used to pair the single measuremnst when doing the Bland-Altman plot. The only difference between the plots is the scaling of the y-axis.*

```
Variance function:
 Structure: Different standard deviations per stratum
 Formula: ~1 | meth
 Parameter estimates:
       J        R
1.000000 1.006435
Number of Observations: 510
Number of Groups: 85
```

Now, the output from `lme` is pretty difficult to read, but the residual standard deviations are $\sigma_J = 5.528081$ and $\sigma_R = 5.528081 \times 1.006435 = 5.563654$, whereas $\tau = 0.0002614756$ — absolutely negligible. Also from the output we get the difference between methods J and R to -0.08627451.

4. An easier way to get the relevant estimates is to use the wrapper `BA.est`, where the only necessary specification is the dataset (assuming that columns `meth`, `item`, `repl` and `y` are present) and whether replicates are linked across methods:

```
> BA.est( sbp, linked=FALSE )

$bias
          J           R
 0.00000000 -0.08627451

$sd.s
       MxI.J        MxI.R      resid.J      resid.R
0.0002586885 0.0002586887 5.5280812722 5.5636544764
```

Which is identical to the quantities we fished out of the `lme` output. Actually `BA.est` fits exactly the model we fitted, and then extracts the quantities that we are interested in.

5. The limits of agreement between the two manual observers is then for R−J $-0.0862 \pm 1.96 \times \sqrt{2 \times 0.000261^2 + 5.528^2 + 5.528^2} = (-15.46, 15.29)$, i.e. on average they agree, but in order to be sure to enclose 95% of all differences we need an interval approximately as 0 ± 15mmHg.

6. These limits of agreement ar dramatically at variance with those computed from using the dataset-ordering of the replicates. Clearly, one may suspect that replicates may be linked, and indeed when looking at the tracks of the mesurements by methods J and R across replicates as in figure 4.10, the measurements are far more similar that you would expect if replicates were exchangeable. You can see the tracking by doing e.g. (after reloading `sbp` so as to get all three methods back):

```
> data(sbp)
> sbp$item <- factor(sbp$item)
> with( subset(sbp), plot( repl, y, type="n", xaxt="n" ) )
> with( subset(sbp,meth=="J"), for( i in 1:nlevels(item) )
+                              lines(repl[as.numeric(item)==i],
+                                    y[as.numeric(item)==i],
+                                  col=rainbow(nlevels(item))[i] ) )
> with( subset(sbp,meth=="R"), for( i in 1:nlevels(item) )
+                              lines(repl[as.numeric(item)==i],
+                                    y[as.numeric(item)==i],
+                                  col=rainbow(nlevels(item))[i], lty=2 ) )

> str(sbp)
```

```
'data.frame':        765 obs. of  4 variables:
 $ meth: Factor w/ 3 levels "J","R","S": 1 1 1 1 1 1 1 1 1 1 ...
 $ item: Factor w/ 85 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ repl: num  1 1 1 1 1 1 1 1 1 1 ...
 $ y   : num  100 108 76 108 124 122 116 114 100 108 ...
```

```
> with( subset(sbp), plot( repl, y, type="n", xaxt="n" ) )
> with( subset(sbp,meth=="J"), for( i in 1:nlevels(item) )
+                                lines(repl[as.numeric(item)==i],
+                                      y[as.numeric(item)==i],
+                                    col=rainbow(nlevels(item))[i] ) )
> with( subset(sbp,meth=="S"), for( i in 1:nlevels(item) )
+                                lines(repl[as.numeric(item)==i],
+                                      y[as.numeric(item)==i],
+                                    col=rainbow(nlevels(item))[i], lty=2 ) )
```

The resuts is shown in figure 4.10.

7. An alternative way of seeing the lack of exchangeability is to make the overview plot using a random permuation of the replicates. If replicates were truely exchangeable within methods the plot would look similar when permuting the replicates — and it does not!

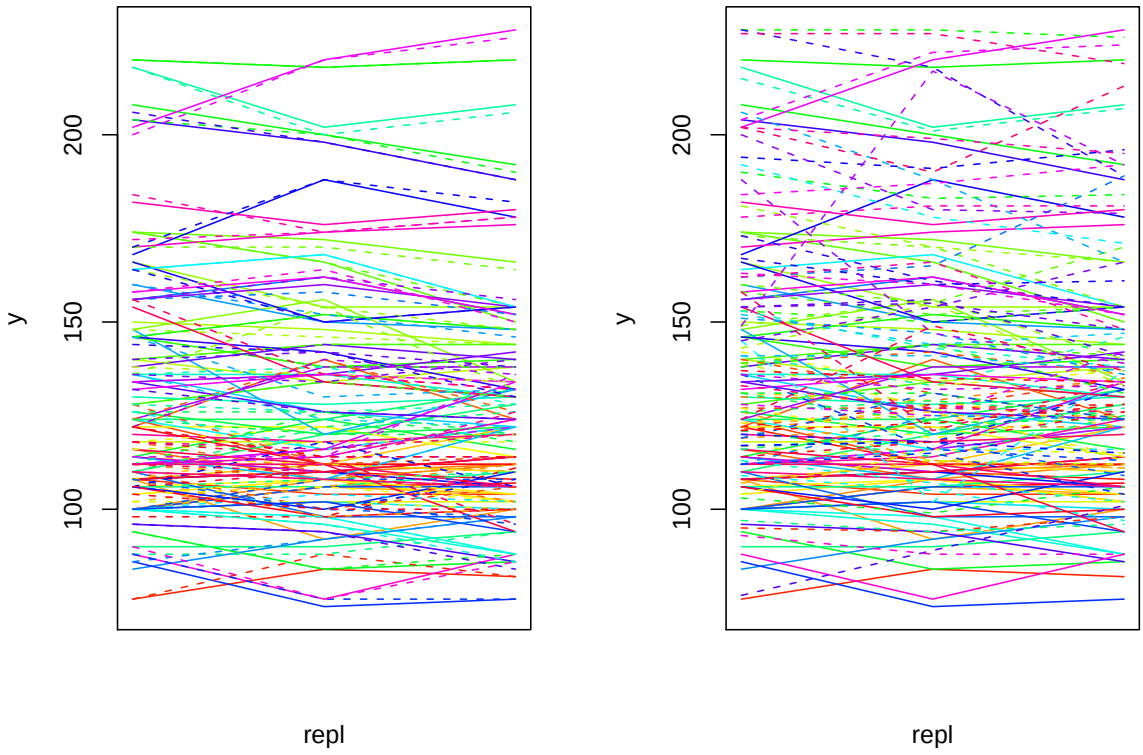

Figure 4.10: *Plot of measurements of systolic blood pressure versus replicate number; measurments on the same item (individual) are by the same colour, measurements by observer J are connected with full lines, those by observer R (left) and S (right) with broken lines.*

For completeness we reload the data to get observations by all three methods included, and then make overview plots with different random permutations within (method,item):

```
> data(sbp)
> str(sbp)

'data.frame':        765 obs. of  4 variables:
 $ meth: Factor w/ 3 levels "J","R","S": 1 1 1 1 1 1 1 1 1 ...
 $ item: num  1 2 3 4 5 6 7 8 9 10 ...
 $ repl: num  1 1 1 1 1 1 1 1 1 ...
 $ y   : num  100 108 76 108 124 122 116 114 100 108 ...

> plot.meth(perm.repl(sbp))

Note:
 Replicate measurements are taken as separate items!
```

8. Therefore the correct analysis will be one based on a model where a random item by replicate effect is included to accomodate the linking of replicates:

```
> BA.est(sbp,exch=FALSE)

$bias
          J           R           S
 0.00000000 -0.08627451 15.61960784

$sd.s
       MxI.J        MxI.R        MxI.S          IxR      resid.J      resid.R
 0.338472324  0.001055065 18.077129472  5.887152180  1.630052808  1.546668617
     resid.S
 9.142806279
```

   The substantial item by replicate interaction (`IR`) indicates that repliactes are `not` exchangeable within methods.

9. The resulting estimates from this model gives limits of agreement for J−R:

$$0.0863 \pm 1.96 \times \sqrt{0.3385^2 + 0.0011^2 + 1.6301^2 + 1.5467^2} = 0.0863 \pm 4.4540 = (-4.37, 4.54)$$

   which is much more agreement with the limits computed based on the simplistic way of taking replicates as items — a procedure wich is actually close to correct if replicates are linked as it turns out they are.

10. Alternatively this could be formulated as a 95% prediction interval for R given a measurement by J, y_{J}, which would be

$$y_{\mathrm{R}}|y_{\mathrm{J}} = y_{\mathrm{J}} - 0.0863 \pm 4.4540 = y_{\mathrm{J}} + (-4.54; 4.37)$$

11. The above analysis is based on the correct analysis of the entire dataset, including the information from the machine measurement S. If we fit the model on the restricted dataset, we of course get a common method by item interaction term because we then only have two methods:

```
> BA.est(subset(sbp,meth!="S"),exch=FALSE)
```

```
$bias
          J           R
 0.00000000 -0.08627451

$sd.s
    MxI.J     MxI.R       IxR   resid.J   resid.R
0.2483701 0.2483701 5.9329618 1.4858696 1.6674599
```

Based on these estimates we get the limits of agreement to be:

$$0.0863 \pm 1.96 \times \sqrt{2 \times 0.2484^2 + 1.4859^2 + 1.6674^2} = 0.0863 \pm 4.4313 = (-4.35, 4.52)$$

i.e. the same as before, based on all three methods.

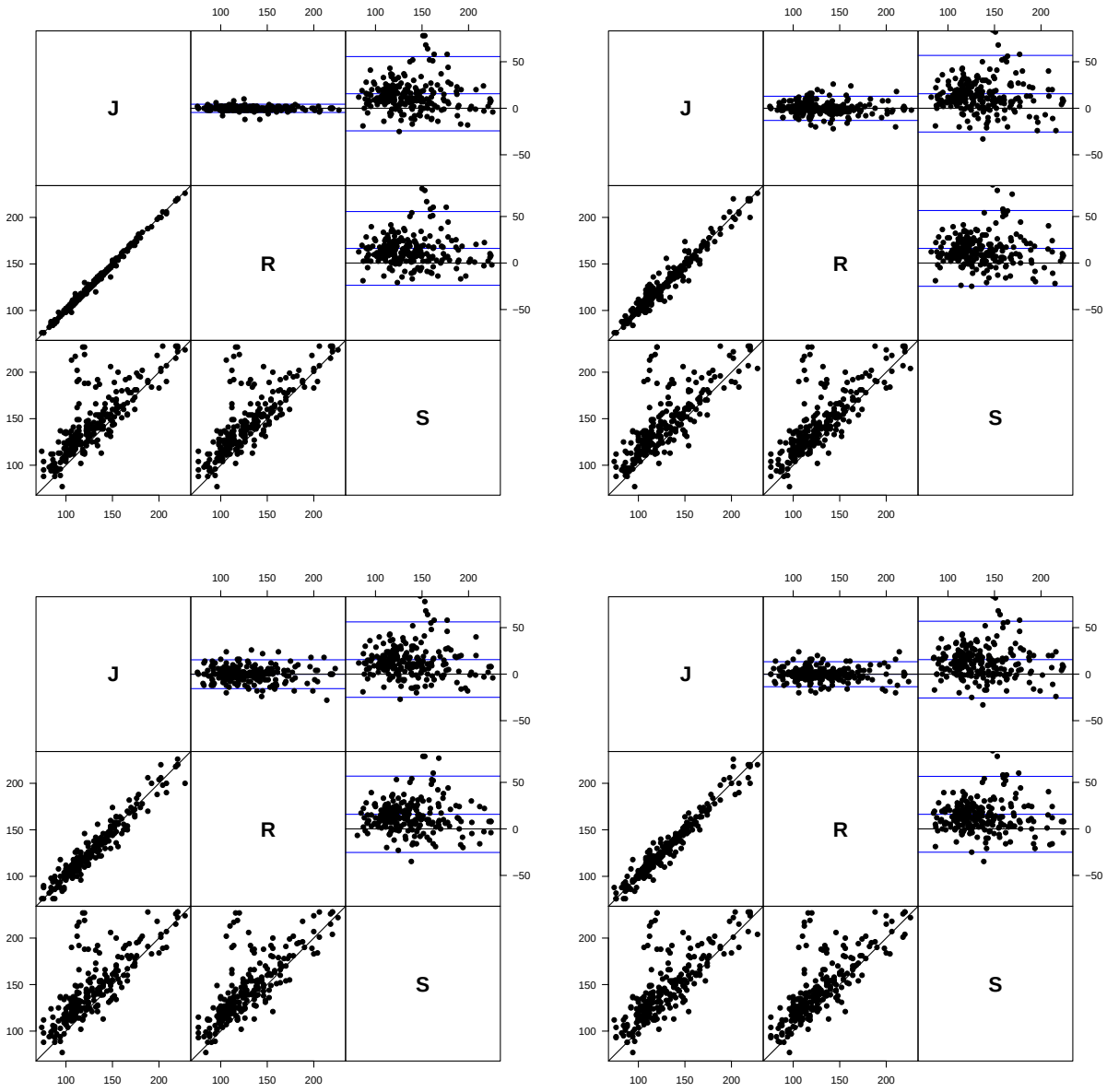

Figure 4.11: *Graphical overview of the* `sbp` *data; top left panel with the original replicate numbers used for matching; the other with replicates permuted randomly within methods.*

4.4 Oximetry: Linked replicates with non-constant bias

1. Having loaded the data we can plot the two sets of measurements against each other using the `plot.meth` function, which produces the plot in figure 4.12:

```
> data(ox)
> plot.meth(ox)

Note:
 Replicate measurements are taken as separate items!
```

2. We use the `BA.plot` function to generate a more detailed version of the Bland-Altman plot than the one available from the `plot.meth` function, which is displayed in 4.13:

```
> par(mfrow = c(1,1))
> BA.plot(ox)

Limits of agreement with 95 % confidence intervals:

CO-pulse           Estimate      2.5 %    97.5 %
  mean difference  2.477401   1.561313  3.393489
  upper limit     14.581351  12.994640 16.168061
  lower limit     -9.626548 -11.213259 -8.039838
```

From the printed output of the `BA.plot` function we find that the estimated average difference between measurements is 2.48 with 95% confidence interval (1.56,3.39). The lower

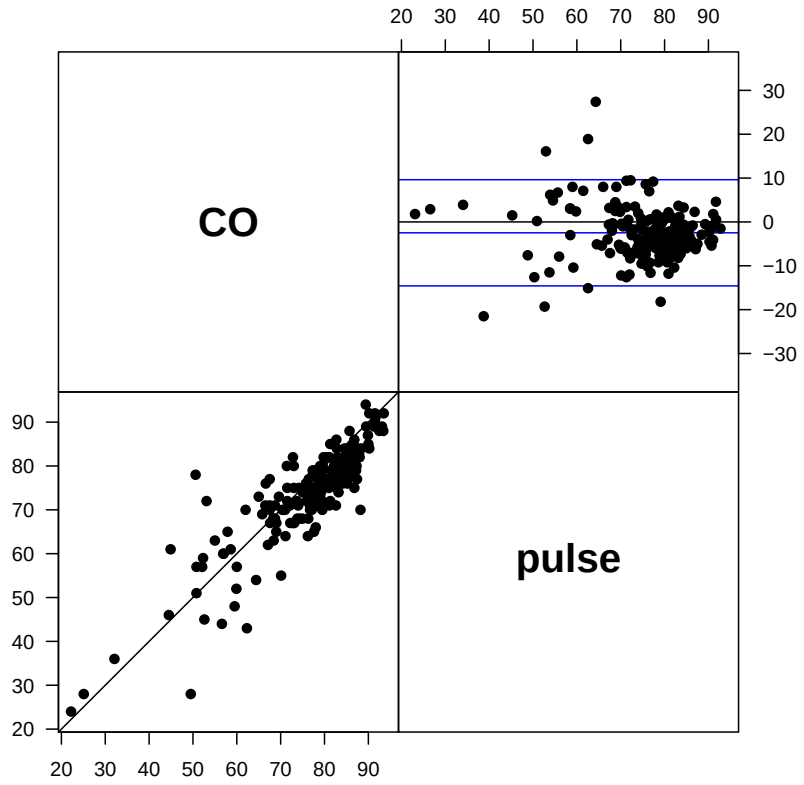

Figure 4.12: *A scatterplot (lower left) and Bland-Altman plot (upper right) of the oximetry data.*

and upper limits of agreement between the two methods are -9.62 (95% confidence interval (-11.21,-8.04)) and 14.58 (95% confidence interval (12.99,16.17) respectively. The average difference of about 2.5 is fairly small compared to the mean oximetry measurement of 74 but the limits of agreement are quite wide and encompass both positive and negative values (so there is uncertainty about the direction of any bias).

3. We run the `BA.est` function to fit a linear mixed effect model that estimates the variance components:

```
> BA.est(ox)

$bias
       CO     pulse
 0.000000 -2.470446

$sd.s
     MxI.CO   MxI.pulse         IxR    resid.CO resid.pulse
   2.928042    2.928042    3.415692    2.224868    3.994451
```

The assumption that the residual variances for `CO` and `pulse` is not borne out by the estimates, since the estimated residual variance for co-oximetry (`resid.CO` in the output) is 2.22, almost half as large as the corresponding value for pulse oximetry (`resid.CO` in the output) of 3.99. The estimated value of the `IR` variance component is 3.42, which is larger

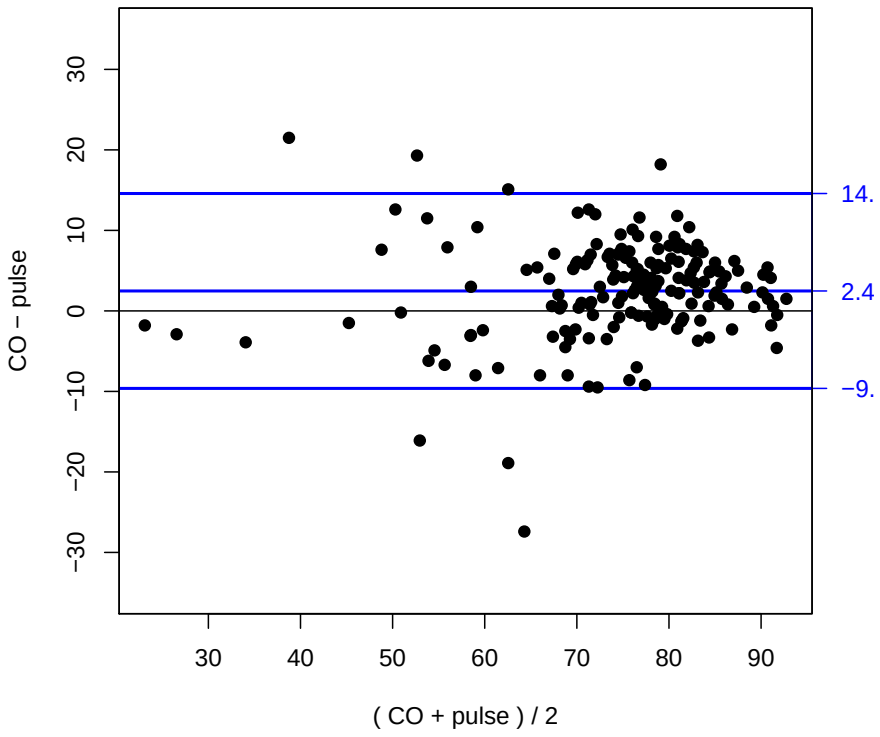

Figure 4.13: *A Bland-Altman plot of the oximetry data with 95% confidence bands for the mean difference and the limits of agreement.*

than the estimate of 2.93 for the `MI` variance component (note that `MI.CO` and `MI.pulse` are the same since we have only two methods of measurement). These variance components lie in between the estimated residual variance for the two methods. We might expect the `IR` variance component to comparable in size to the other variance components since the replicates are linked.

4. Running the `MethComp` routine and using the corresponding `print` function produces the following output:

```
> ox.mi.ir <- MethComp( ox, random=c("mi","ir"), n.iter=5000, beta = FALSE )

Comparison of 2 methods, using 354 measurements
on 61 items, with up to 3 replicate measurements.
( 2 * 61 * 3 = 366 ):

No. items with measurements on each method:
        #Replicates
Method  1 2  3 #Items #Measurements: 354
  CO    1 4 56     61                177
  pulse 1 4 56     61                177

Simulation run of a model with
- fixed bias (beta==1)
- method by item and item by replicate interaction:
- using 3 chains run for 5000 iterations
  (of which 2500 are burn-in),
- monitoring every 3 values of the chain:
- giving a posterior sample of 2500 observations.

> print(ox.mi.ir)

Conversion formula:
 y_to = alpha + beta * y_from +/- 2*sd.pred:

      From:     CO                  pulse
             alpha   beta sd.pred  alpha   beta sd.pred
To:
CO           0.000  1.000   2.205  2.482  1.000   6.235
pulse       -2.482  1.000   6.235  0.000  1.000   4.091

Variance components (standard deviations):
                    50%   2.5%  97.5%     0%   100%
sigma.mir[CO]    2.2055 0.7159 3.0194 0.2165 3.5280
sigma.mir[pulse] 4.0910 3.4305 4.8675 2.9490 6.0040
sigma.mi[CO]     2.9270 2.2020 3.7920 1.8020 4.3260
sigma.mi[pulse]  2.9270 2.2020 3.7920 1.8020 4.3260
sigma.ir[CO]     3.4830 2.7935 4.3135 2.2370 4.9110
sigma.ir[pulse]  3.4830 2.7935 4.3135 2.2370 4.9110
sigma.tot[CO]    5.0900 4.5012 5.7433 4.1058 6.2836
sigma.tot[pulse] 6.1425 5.4747 7.0446 5.1625 7.9153
```

The `plot` function produces a scatterplot displaying the linear equations relating one method to the other (recall that the slope has been constrained to be 1.00):

```
> plot(ox.mi.ir, pl.obs = TRUE)
```

The `plot.VarComp` function produces smoothed posterior densities for the variance components separately for each method (note that the residual variance is different between methods since the `MI` and `IR` variance components are constrained to be the same):

```
> par(mfrow = c(2,1))
> plot.VarComp(ox.mi.ir)
```

The graph strongly supports the contention that the two residual variances are not equal since the support for the posterior density of each hardly overlap at all.

5. We now estimate both intercept and slope parameters using `MethComp` and summarise the results using the `print` routine. We also need to display a summary of the `WinBUGS` output in order to get standard errors for the estimated `beta` parameters:

```
> ox.mi.ir.beta <- MethComp( ox, random=c("mi","ir"), n.iter=50000, n.chains=5 )

Comparison of 2 methods, using 354 measurements
on 61 items, with up to 3 replicate measurements.
( 2 * 61 * 3 = 366 ):

No. items with measurements on each method:
        #Replicates
Method  1 2  3 #Items #Measurements: 354
  CO    1 4 56     61                177
  pulse 1 4 56     61                177

Simulation run of a model with
- method by item and item by replicate interaction:
- using 5 chains run for 50000 iterations
```

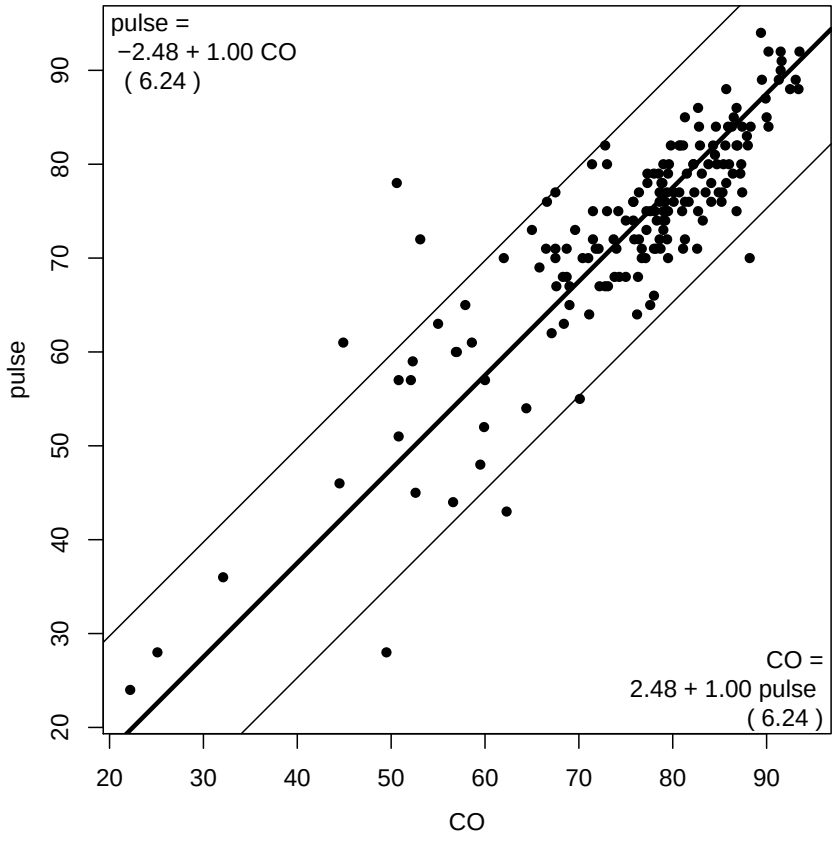

Figure 4.14: *A scatterplot of the oximetry data with the linear equations displayed. The slope of the linear relationship between methods has been constrained to 1.00.*

```
  (of which 25000 are burn-in),
- monitoring every 25 values of the chain:
- giving a posterior sample of 5000 observations.

> print(ox.mi.ir.beta)

Conversion formula:
 y_to = alpha + beta * y_from +/- 2*sd.pred:

      From:     CO                   pulse
             alpha   beta sd.pred  alpha   beta sd.pred
To:
CO           0.000  1.000   1.693 -6.143  1.118   6.861
pulse        5.494  0.894   6.141  0.000  1.000   4.252

Variance components (standard deviations):
                     50%   2.5%  97.5%     0%   100%
sigma.mir[CO]    1.6930 0.1624 2.7541 0.0080 3.4790
sigma.mir[pulse] 4.2520 3.6000 4.9950 2.7950 5.6850
sigma.mi[CO]     3.2361 2.3540 4.3725 1.7292 5.9055
sigma.mi[pulse]  2.9016 2.0674 3.9871 1.5312 5.1428
sigma.ir[CO]     3.8176 3.0755 4.6000 2.4494 5.4701
sigma.ir[pulse]  3.4072 2.7637 4.1548 2.3568 4.7490
sigma.tot[CO]    5.3238 4.6498 6.2071 4.0230 7.1866
sigma.tot[pulse] 6.2130 5.5447 7.0627 4.9905 7.8520

> ox.mi.ir.beta$summary
```

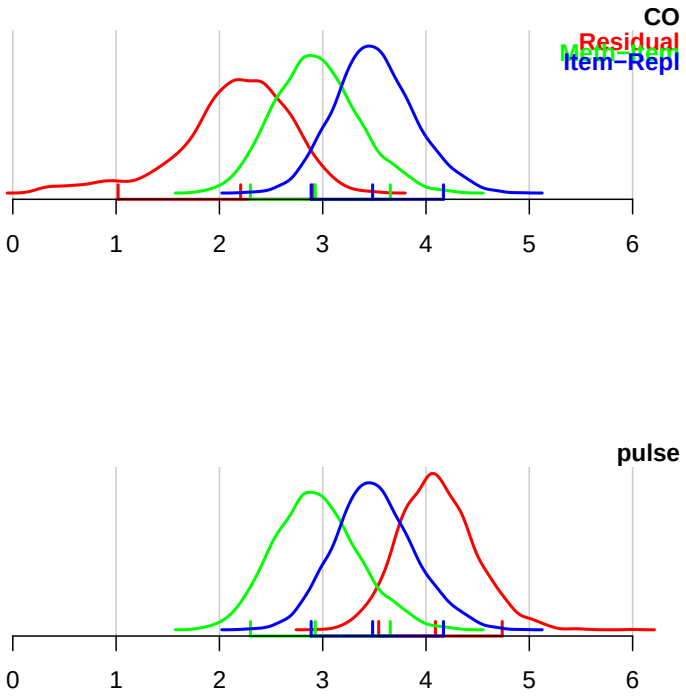

Figure 4.15: *Smoothed density plots of the variance components estimated using* `MethComp`.

	50%	2.5%	97.5%	0%	100%
sigma.mir[CO]	1.6930000	0.1624475	2.754075	7.969000e-03	3.479000
sigma.mir[pulse]	4.2520000	3.5999500	4.995000	2.795000e+00	5.685000
deviance	1701.0000000	900.3425000	1854.000000	-1.671000e+02	1965.000000
sigma.mi[CO]	3.2360810	2.3539726	4.372515	1.729207e+00	5.905460
sigma.mi[pulse]	2.9015867	2.0674123	3.987075	1.531233e+00	5.142803
sigma.ir[CO]	3.8176315	3.0755477	4.599954	2.449434e+00	5.470080
sigma.ir[pulse]	3.4072121	2.7636846	4.154757	2.356828e+00	4.749040
sigma.tot[CO]	5.3237855	4.6497669	6.207116	4.022972e+00	7.186631
sigma.tot[pulse]	6.2130279	5.5446962	7.062660	4.990504e+00	7.851984
sd.repl.1	1.6930000	0.1624475	2.754075	7.969000e-03	3.479000
alpha.2.1	5.4941673	-3.5961255	13.086928	-1.267093e+01	19.077858
beta.2.1	0.8943498	0.7959178	1.013467	7.171552e-01	1.121133
id.int.2.1	53.2977307	-69.3122580	158.500665	-Inf	9509.630000
sd.pred.2.1	6.1408208	5.3938022	7.291373	4.832050e+00	8.615599
sd.repl.2	4.2520000	3.5999500	4.995000	2.795000e+00	5.685000
alpha.1.2	-6.1436089	-16.4395839	3.554608	-2.641166e+01	11.301898
beta.1.2	1.1181308	0.9867116	1.256411	8.919550e-01	1.394398
id.int.1.2	53.3241721	-67.3218174	163.394472	-1.241980e+04	Inf
sd.pred.1.2	6.8613372	5.9302165	8.187480	5.265580e+00	9.916277

The summary output provides reasonable evidence that the slope of the linear relationship is different from 1.00, in fact close to 0.90 for the prediction of pulse oximetry from co-oximetry. This implies that the average difference in measurements between the two methods will increase with the magnitude of the underlying measurement. The `plot` method for `MethComp` can be used to display the observed data, fitted line with prediction limits and equations:

```
> plot(ox.mi.ir.beta, pl.obs = TRUE)
```

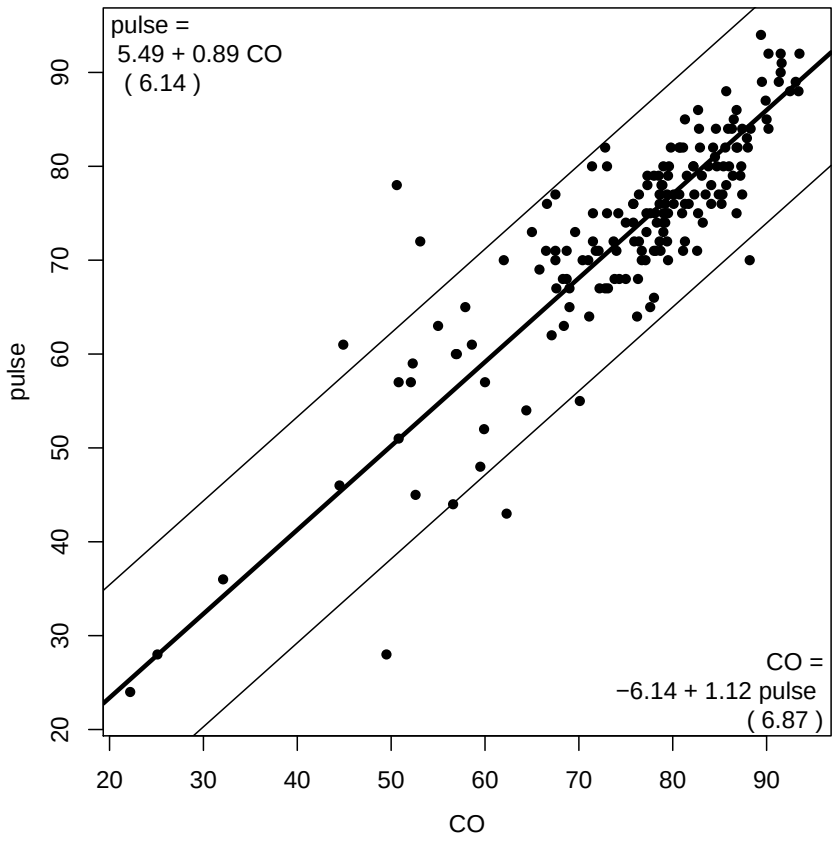

Figure 4.16: *Smoothed density plots of the variance components estimated using* `MethComp`.

4.5 Measurement of HbA_{1c} I: Machine and specimen as method

1. First we load the `hba1c` data and take a look at the structure of data:

```
> data( hba1c )
> with( hba1c, table( d.samp, d.ana ) )

      d.ana
d.samp   1   2   3   4   5
     1  38 114 113 114  76
     2   0  38 114 114 114

> with( hba1c, ftable( dev, type, d.ana ) )

           d.ana  1  2  3  4  5
dev   type
BR.V2 Cap        19 38 38 38 19
      Ven         0 19 38 38 38
BR.VC Cap         0 19 38 38 38
      Ven         0 19 38 38 38
Tosoh Cap        19 38 38 38 19
      Ven         0 19 37 38 38
```

2. The dataset does not have the standard structure, it lacks a definition of method and replicate. We can provide these by using the interaction between `dev` and `type` and the day of analysis as replicate number:

```
> hb <- transform( hba1c, meth = interaction( dev, type ),
+                         repl = d.ana )
> str( hb )

'data.frame':        835 obs. of  8 variables:
 $ dev   : Factor w/ 3 levels "BR.V2","BR.VC",..: 2 2 2 2 2 2 2 2 1 1 ...
 $ type  : Factor w/ 2 levels "Cap","Ven": 2 2 2 2 1 1 1 1 2 2 ...
 $ item  : num  12 12 12 12 12 12 12 12 12 12 ...
 $ d.samp: num  1 1 1 1 1 1 1 1 1 1 ...
 $ d.ana : num  2 3 4 5 2 3 4 5 2 3 ...
 $ y     : num  8.7 8.7 8.7 8.7 9.2 9 8.8 8.7 9.4 9.3 ...
 $ meth  : Factor w/ 6 levels "BR.V2.Cap","BR.VC.Cap",..: 5 5 5 5 2 2 2 2 4 4 ...
 $ repl  : num  2 3 4 5 2 3 4 5 2 3 ...

> tab.repl( hb )

            #Replicates
Method       3  4 #Items #Measurements: 835
  BR.V2.Cap  0 38     38                152
  BR.VC.Cap 19 19     38                133
  Tosoh.Cap  0 38     38                152
  BR.V2.Ven 19 19     38                133
  BR.VC.Ven 19 19     38                133
  Tosoh.Ven 20 18     38                132
```

   Note that the replication structure i slightly different between machines and specimens (venous/capillary). This is because of technical limitations; only some machines and specimens allow analysis on the same day as the sampling.

3. In figure 4.17 is an overview plot of the data. This plot is made under the assumption that replicates are linked by replicate number, in this case day of analysis. This is presumably a sensible assumption, but we will see later.

```
> plot.meth( hb )

Note:
 Replicate measurements are taken as separate items!
```

There is a tendency that comparisons with the machine `BR-VC` have a higher variance than other comparisons.

4. It is difficult to say if we can consider the replicates exchangeable within methods. But since samples are analyzed on different days we would suspect that there were some linking, so an individual by replicate interaction ma be in its place.

5. The "standard model" for analyzing data of this kind is:

$$y_{mir} = \alpha_m + \beta_m \mu_i + a_{ir} + c_{mi} + e_{mir}$$

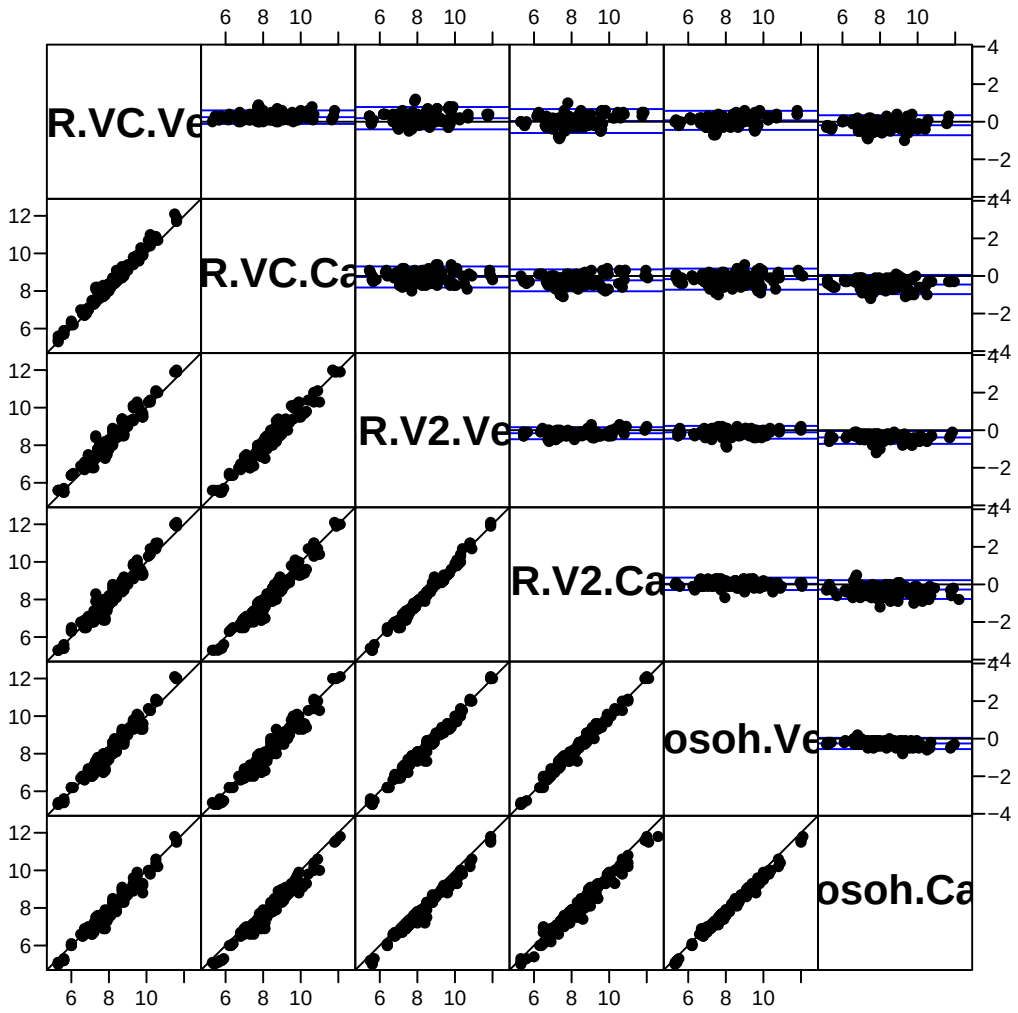

Figure 4.17: *Overview of the HbA_{1c}-data. Replicate measurements on the same day of analysis are lined as items.*

This is actually the most elaborate model fitted by `MethComp` but also the default:

```
> m0 <- MethComp( hb )

Comparison of 6 methods, using 835 measurements
on 38 items, with up to 5 replicate measurements.
( 6 * 38 * 5 = 1140 ):

No. items with measurements on each method:
           #Replicates
Method       3  4 #Items #Measurements: 835
  BR.V2.Cap  0 38     38                152
  BR.VC.Cap 19 19     38                133
  Tosoh.Cap  0 38     38                152
  BR.V2.Ven 19 19     38                133
  BR.VC.Ven 19 19     38                133
  Tosoh.Ven 20 18     38                132

Simulation run of a model with
- method by item and item by replicate interaction:
- using 3 chains run for 2000 iterations
  (of which 1000 are burn-in),
- monitoring all values of the chain:
- giving a posterior sample of 3000 observations.
```

The resulting `MethComp` object `m0` is quite big, so it is more handily represented by its default print method:

```
> m0

Conversion formula:
 y_to = alpha + beta * y_from +/- 2*sd.pred:

                  From: BR.V2.Cap BR.VC.Cap Tosoh.Cap BR.V2.Ven BR.VC.Ven Tosoh.Ven
To:
BR.V2.Cap alpha              0.000    -0.669    -0.333    -0.483    -1.675    -0.054
          beta               1.000     1.056     1.076     1.041     1.212     1.013
          sd.pred            0.096     0.325     0.247     0.217     0.388     0.199
BR.VC.Cap alpha              0.634     0.000     0.282     0.168    -0.924     0.668
          beta               0.947     1.000     1.023     0.987     1.145     0.960
          sd.pred            0.307     0.135     0.329     0.304     0.432     0.299
Tosoh.Cap alpha              0.309    -0.275     0.000    -0.150    -0.892     0.153
          beta               0.929     0.977     1.000     0.969     1.083     0.949
          sd.pred            0.232     0.326     0.150     0.222     0.369     0.194
BR.V2.Ven alpha              0.464    -0.170     0.154     0.000    -1.041     0.397
          beta               0.961     1.013     1.032     1.000     1.152     0.975
          sd.pred            0.209     0.309     0.227     0.081     0.365     0.174
BR.VC.Ven alpha              1.382     0.806     0.823     0.903     0.000     1.244
          beta               0.825     0.873     0.923     0.868     1.000     0.849
          sd.pred            0.322     0.382     0.329     0.316     0.072     0.305
Tosoh.Ven alpha              0.053    -0.696    -0.161    -0.408    -1.465     0.000
          beta               0.987     1.042     1.054     1.026     1.178     1.000
          sd.pred            0.196     0.313     0.203     0.178     0.358     0.087

Variance components (standard deviations):
                        50%   2.5%  97.5%     0%   100%
sigma.mir[BR.V2.Cap] 0.0963 0.0665 0.1393 0.0530 0.1680
sigma.mir[BR.VC.Cap] 0.1349 0.1147 0.1613 0.0996 0.1930
sigma.mir[Tosoh.Cap] 0.1501 0.1178 0.1837 0.1049 0.2079
sigma.mir[BR.V2.Ven] 0.0814 0.0624 0.1033 0.0481 0.1280
sigma.mir[BR.VC.Ven] 0.0721 0.0570 0.0886 0.0488 0.1067
```

```
sigma.mir[Tosoh.Ven] 0.0867 0.0671 0.1133 0.0536 0.1406
sigma.mi[BR.V2.Cap]  0.1226 0.0537 0.2229 0.0294 0.3074
sigma.mi[BR.VC.Cap]  0.2279 0.1417 0.3070 0.0669 0.4159
sigma.mi[Tosoh.Cap]  0.0551 0.0061 0.7604 0.0024 0.9819
sigma.mi[BR.V2.Ven]  0.1110 0.0546 0.4561 0.0266 0.6384
sigma.mi[BR.VC.Ven]  0.2823 0.1984 0.5359 0.1653 0.7667
sigma.mi[Tosoh.Ven]  0.0339 0.0025 3.2895 0.0011 4.1769
sigma.ir[BR.V2.Cap]  1.2978 1.1309 1.4929 1.0396 1.6305
sigma.ir[BR.VC.Cap]  1.2272 1.0540 1.4236 0.9367 1.5527
sigma.ir[Tosoh.Cap]  1.1916 0.6241 1.3904 0.5545 1.5027
sigma.ir[BR.V2.Ven]  1.2368 0.9722 1.4250 0.8298 1.5686
sigma.ir[BR.VC.Ven]  1.0654 0.8275 1.2767 0.7396 1.3834
sigma.ir[Tosoh.Ven]  1.2875 1.0529 1.5779 0.9603 1.8564
sigma.tot[BR.V2.Cap] 1.3085 1.1420 1.5045 1.0450 1.6486
sigma.tot[BR.VC.Cap] 1.2553 1.0892 1.4509 0.9843 1.5832
sigma.tot[Tosoh.Cap] 1.2029 0.8763 1.4001 0.7980 1.5139
sigma.tot[BR.V2.Ven] 1.2448 1.0538 1.4318 0.9616 1.5775
sigma.tot[BR.VC.Ven] 1.1100 0.9215 1.3028 0.8249 1.4035
sigma.tot[Tosoh.Ven] 1.3557 1.1714 3.5378 1.0749 4.3734
```

6. We can also get a graphical overview of the relationships between the methods by using the function `plot.MethComp`. Since the resulting object is of class `MethComp`, it suffices to say:

```
> plot( m0, grid=TRUE )
```

7. There is a zillion arguments to `MethComp` (did you remember to type "`?MethComp`"?), but for a start we just use the default settings — in "real" applications one would use a larger number of iterations in order to be on the safe side. Since there are 6 methods we can plot the variance components associated with each of them in a 2 by 3 layout, try:

```
> par(mfrow=c(2,3))
> plot.VarComp(m0)
```

8. Clearly, the posterior distributions of the variance components in figure 4.19 are not very well determined, so we re-fit the model using substantially more iterations. Try to enclose the call to `MethComp` in a `system.time()` in order to see how much time it takes.

```
> system.time(
+ m1 <- MethComp( hb, n.iter=10000, n.chains=5 )
+            )

Comparison of 6 methods, using 835 measurements
on 38 items, with up to 5 replicate measurements.
( 6 * 38 * 5 = 1140 ):

No. items with measurements on each method:
            #Replicates
Method       3  4 #Items #Measurements: 835
  BR.V2.Cap  0 38     38                152
  BR.VC.Cap 19 19     38                133
  Tosoh.Cap  0 38     38                152
  BR.V2.Ven 19 19     38                133
  BR.VC.Ven 19 19     38                133
  Tosoh.Ven 20 18     38                132

Simulation run of a model with
- method by item and item by replicate interaction:
- using 5 chains run for 10000 iterations
  (of which 5000 are burn-in),
```

```
- monitoring every 5 values of the chain:
- giving a posterior sample of 5000 observations.

   user  system elapsed
   6.14    0.06  789.95
```

9. Having done this more elaborate simulation we can get a more detailed plot by fiddling the graphics parameters a bit:

```
> par(mfrow=c(2,3),mar=c(3,1,2,1),mgp=c(3,1,0)/1.6)
> plot.VarComp(m1,grid=seq(0,1.8,0.1))
```

10. Based on the posterior distributions shown in figure 4.20, the following conclusions may be drawn:

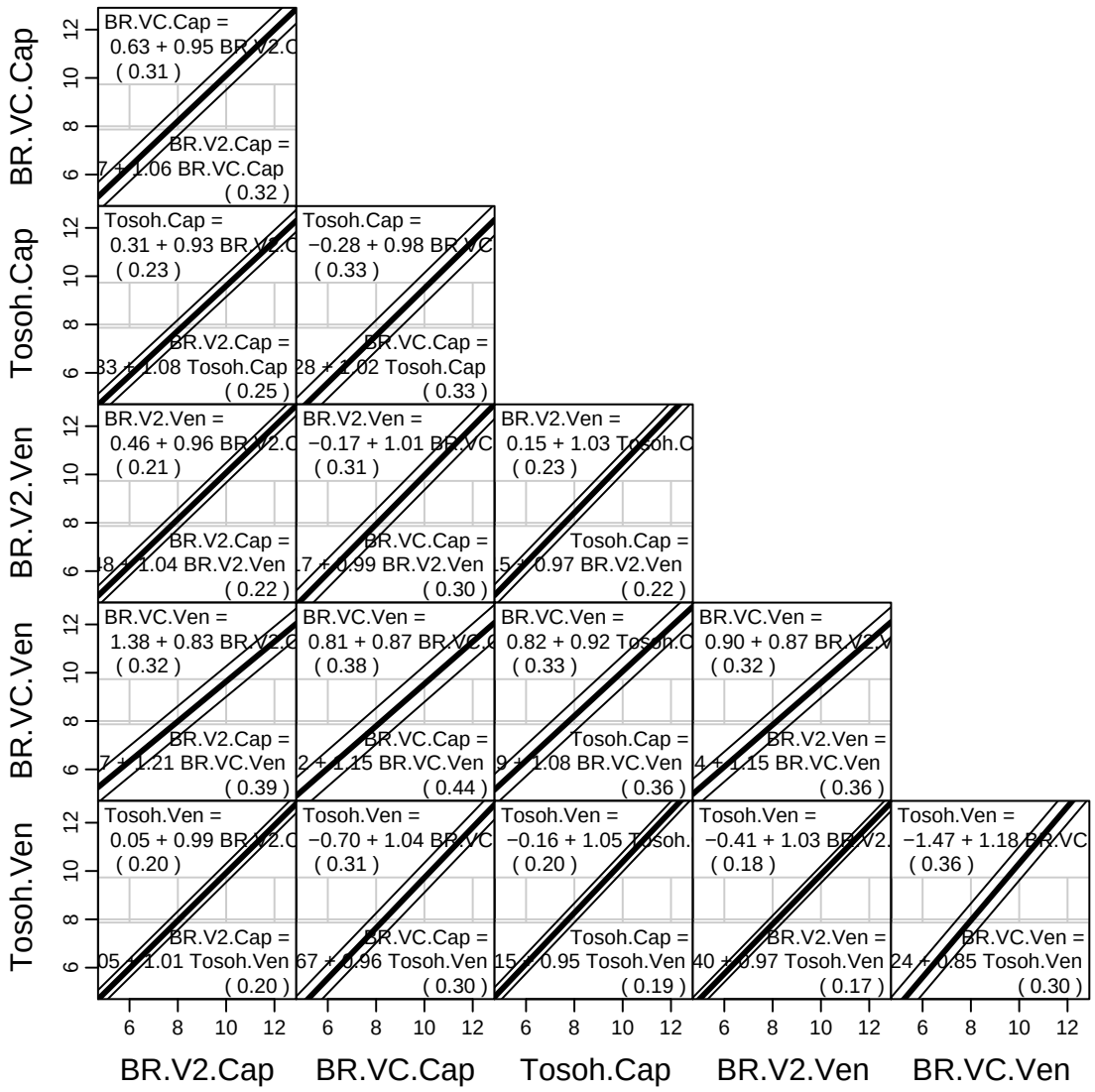

Figure 4.18: *Estimated translation formulae between methods, based on the posterior distribution of the identifiable translation parameters.*

- The method×item effect is largest for `BR.VC` (the existing machine) and smallest for `Tosoh`. This indicates that `Tosoh` has be best stability of measurements across patients.
- The residual variance is pretty much the same across machines, but substantially smaller for venous than for capillary samples.
- The item×replicate variance component may be large, but is very badly estimated, i.e. there is presumably not much information about it in the dataset.

  One explanation may be that there is a systematic effect of replicate — recall that replicates are not exchangeable because they refer to different days of analysis. Hence a possibility would be to explore whether there was a systematic effect of analysis day alone or analysis day by machine. This systematic feature is however not accommodated by the `MethComp` function. This can be implemented by using the `code.only` argument of `MethComp`, which produces the `BUGS` code in a separate file, which can then be edited to accommodate the systematic effects mentioned.

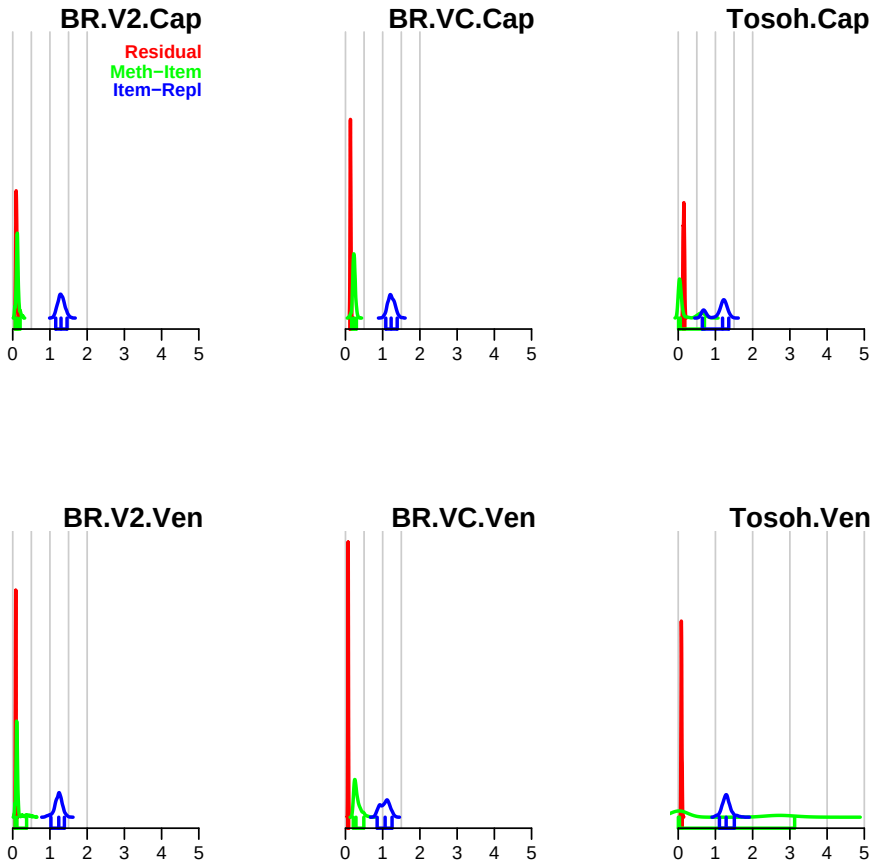

Figure 4.19: *Posterior distributions of the variance components for the 6 methods, based on 1000 burn-in and 1000 samples from 3 chains.*

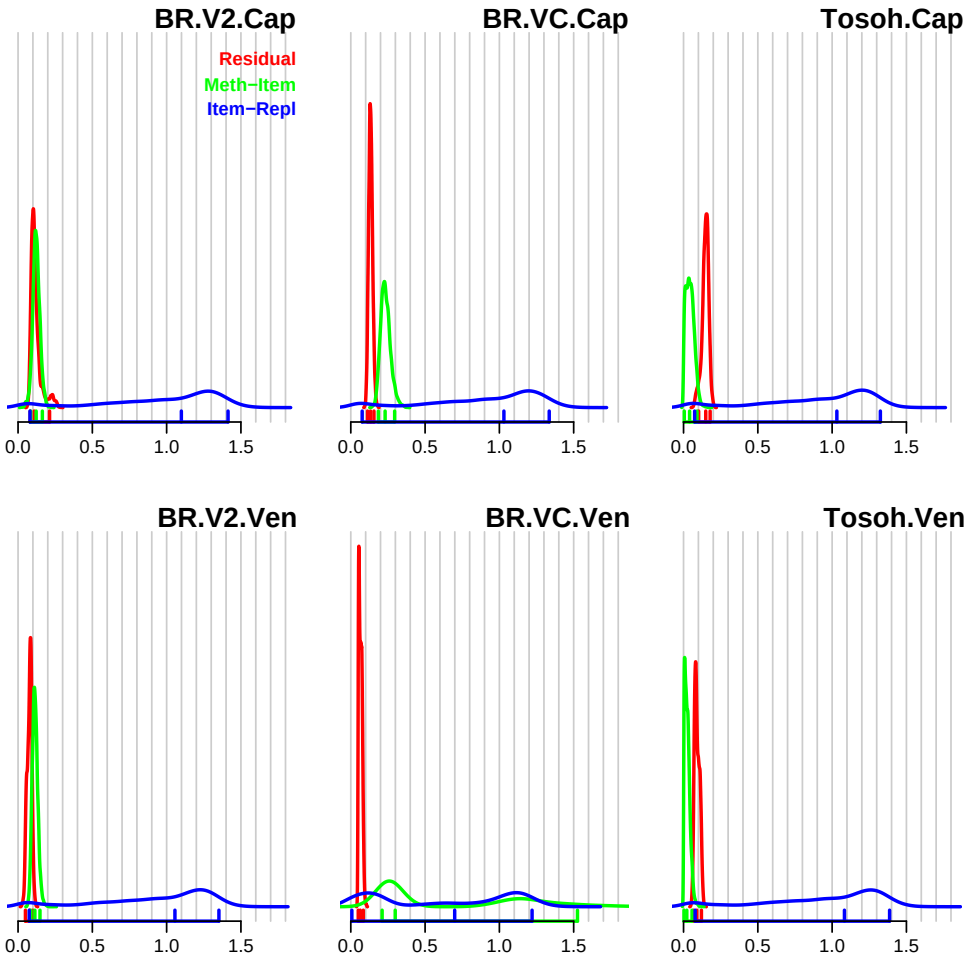

Figure 4.20: *Posterior distributions of the variance components for the 6 methods, based on 5000 burn-in and 5000 samples from 5 chains.*

Chapter 5

MethComp manual

`abconv` *Derive linear conversion coefficients from a set of indeterminate coefficients*

Description

If a method comparison model is defined as $y_{mi} = \alpha_m + \beta_m \mu_i$, $m = 1, 2$ the coefficients of the linear conversion form method 1 to 2 are computed as well as the point where the linear conversion function intersects the identity line. The function is designed to work on numerical vectors of posterior samples from BUGS output.

Usage

```
abconv( a1, b1 = 1:4, a2 = NULL, b2 = NULL,
        col.names = c("alpha.2.1", "beta.2.1", "id.2.1") )
```

Arguments

`a1`	Numerical vector of intercepts for first method. Alternatively a dataframe where the vectors are selected from.
`b1`	Numerical vector of slopes for first method. If `a1` is a dataframe, this is assumed to be a numerical vector of length 4 pointing to the columns of `a1` with the intercepts and slopes.
`a2`	Numerical vector of intercepts for second method.
`b2`	Numerical vector of slopes for second method.
`col.names`	Names for the resulting three vectors.

Value

A dataframe with three columns: intercept and slope for the conversion from method 1 to method 2, and the value where the conversion is the identity.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

References

B Carstensen: Comparing and predicting between several methods of measurement, Biostatistics, 5, pp 399-413, 2004

See Also

`BA.plot`, `MethComp`

Examples

```
abconv( 0.3, 0.9, 0.8, 0.8 )
```

`ass.measures` *Association measures for method comparison studies. Please don't!*

Description

Computes correlation, mean squared difference, concordance correlation coefficient and the association coefficient. `middle` and `ends` are useful utilities for illustrating the shortcomings of the association measures, see the example.

Usage

```
ass.measures(x, y)
middle(w, rm = 1/3)
ends(w, rm = 1/3)
```

Arguments

`x`	vector of measurements by one method.
`y`	vector of meásurements by another method.
`w`	numerical vector.
`rm`	fraction of data to remove.

Details

These measures are all flawed since they are based on the correlation in various guises. They fail to address the relevant problem of AGREEMENT. It is recommended NOT to use them. The example gives an example, illustrating what happens when increasingly large chunks of data in the middle are removed.

Value

`ass.measures` return a vector with 4 elements. `middle` and `ends` return a logical vector pointing to the middle or the ends of the `w` after removing a fraction of `rm` from data.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

References

Shortly...

See Also

MethComp.

Examples

```
cbind( zz <- 1:15, middle(zz), ends(zz) )
data( sbp )
bp <- subset( sbp, repl==1 & meth!="J" )
bp$meth <- factor( bp$meth )
tab.repl( bp )
plot.meth( bp )
bw <- to.wide( bp )
```

```
with( bw, ass.measures( R, S ) )
# See how it gets better with less and less data:
rbind(
with( subset( bw, middle( R+S ) )   , ass.measures( R, S ) ),
with(         bw                    , ass.measures( R, S ) ),
with( subset( bw, ends( R+S      ) ), ass.measures( R, S ) ),
with( subset( bw, ends( R+S, 0.4 ) ), ass.measures( R, S ) ),
with( subset( bw, ends( R+S, 0.6 ) ), ass.measures( R, S ) ),
with( subset( bw, ends( R+S, 0.8 ) ), ass.measures( R, S ) ) )
```

`BA.est` *Bias and variance components for a Bland-Altman plot.*

Description

A variance component model is fitted to method comparison data with replicate measurements in each method by item stratum. The purpose is to simplify the construction of a correct Bland-Altman-plot.

Usage

```
BA.est( data, linked=TRUE, exch=!linked )
```

Arguments

`data`	A data frame representing method comparison data with replicate measurements, i.e. with variables `meth`, `item`, `repl` and `y`.
`linked`	Are the replicated linked within item across methods?
`exch`	Are the replicated exchangeable within (item,methods)? This argument is just for convenience.

Details

The model fitted is:

$$y = \alpha_m + \mu_i + c_{mi} + a_{ir} + e_{mir}, \quad \text{var}(c_{mi}) = \tau_m^2, \quad \text{var}(a_{ir}) = \nu^2, \quad \text{var}(e_{mir}) = \sigma_m^2,$$

We can only fit separate variances for the τs if more than two methods are compared (i.e. `nM` > 2).

NOTE: Currently, the author is too ignorant about the ins and outs of lme to find out how to get different τs when `nM`>2. Therefore the estimates in this case are equal i.e. from the model postulated.

Value

A list with two elements:

`bias`	Vector of estimates of α_m, the first element is always 0.
`sd.s`	Vector of sd of the variance components, `nM` τs, one ν if `linked`, and `nM` σs.

Author(s)

Bendix Carstensen

References

Carstensen, Simpson & Gurrin: Statistical models for assessing agreement in method comparison studies with replicate measurements, Preprint, Departmet of Biostatistics, University of Copenhagen, 2007, http://www.pubhealth.ku.dk/bs/publikationer/Research_report_07-14.pdf/.

See Also

BA.plot,tab.repl,perm.repl

Examples

```
data( ox )
BA.est( ox )
BA.est( ox, exch=TRUE )
data( sbp )
BA.est( sbp )
BA.est( sbp, exch=TRUE )
```

BA.plot *Bland-Altman plot for dataframes*

Description

Computes limits of agreement and produces a Bland-Altman plot of differences versus averages for two methods of measurement. The function is just a wrapper allowing a dataframe with columns item, meth and y (and possibly repl) to be used as input to a Bland-Altman plot, using BlandAltman.

Usage

```
BA.plot( y1, y2, meth.names = NULL,
         mean.repl = FALSE, comp.levels=1:2, ...)
```

Arguments

y1	Measurements by method 1. Alternatively a dataframe with columns meth, item, y, and possibly repl.
y2	Corresponding measurements by method 2. Ignored if y1 is a dataframe.
meth.names	Names for the two methods. Used for annotation of the plot. If not supplied and y1 is a dataframe this is derived from the factor level names of meth.
mean.repl	Logical. If there are replicate measurements by each method should the means by item and meth be formed before further ado. WARNING: This will give too narrow limits of agreement.
comp.levels	Levels of the meth factor to compare. May be used to switch the order of the methods compared by specifying comp.meth=2:1.
...	Further arguments passed on the the BlandAltman function. The arguments passed to BlandAltman are used for fine-tuning the appearance of the plot.

Value

A list with 2 elements:

lim.agree	A matrix of limits of agreement as rows and estimate and c.i. as columns.
p.value	P-value for the hypothesis that the mean difference is 0. Usually a lame thing to use.

Side effect: A Bland-Altman plot is produced using the function BlandAltman.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

References

JM Bland and DG Altman: Statistical methods for assessing agreement between two methods of clinical measurement, Lancet, i, 1986, pp. 307-10

See Also

BlandAltman, MethComp

Examples

```
data( ox )
str( ox )
# A wrong and a correct plot of the data.
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot( ox, mean.repl=TRUE , ymax=30 )  # Too narrow limits
BA.plot( ox, mean.repl=FALSE, ymax=30 ) # (Almost) correct limits
# The same illustrating the use of method names
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot( ox, mean.repl=TRUE , meth.names=c("one","two"), ymax=30 )
BA.plot( ox, mean.repl=FALSE, meth.names=c("one","two"), ymax=30 )
```

BlandAltman *Bland-Altman plot of differences versus averages.*

Description

For two vectors of equal length representing measurements of the same quantity by two different methods, the differences are plotted versus the average. The limits of agreement (prediction limits for the differences) are plotted, optionally with c.i.s.

Usage

```
BlandAltman(x, y,
          x.name = NULL, y.name = NULL, maintit = "",
          cex = 1, pch = 16, col.points = "black", col.lines = "blue",
          limx=NULL, limy=NULL, ymax=NULL, eqax=FALSE, xlab=NULL, ylab=NULL,
          print = TRUE, conf.int = FALSE, reg.line=FALSE,
          digits = 2, alpha = 0.05, mult=FALSE,
          ... )
```

Arguments

x	Numerical vector of measurements by 1st method.
y	Numerical vector of measurements by 2nd method. Must of same length as x.
x.name	Label for the 1st method (x).
y.name	Label for the 2nd method (y).
maintit	Main title for the plot
cex	Character expansion for the points.
pch	Plot symbol for points.
col.points	Color for the points.
col.lines	Color for the lines indicating limits of agreement.
limx	x-axis limits.
limy	y-axis limits.
ymax	Scalar. The y-axis will extend from -ymax to +ymax.
eqax	Logical. Should the range on x- and y- axes be the same?
xlab	x-axis label.
ylab	y-axis label.
print	Logical: Should the limits of agreement and the c.i.s of these be printed?

`conf.int`	Logical: Should confidence inetrvals for the mean difference and the limits of agreement be plotted too?
`reg.line`	Logical: Should the regression of difference on averages be drawn?
`digits`	How many decimal places should be used when printing limits of agreement? Used both for the printing of results and for annotation of the plot.
`alpha`	1 minus confidence level used when computing confidence intervals and limits of agreement.
`mult`	Logical. Should data be log-transformed and reporting be on a multiplicative scale?
`...`	Further arguments passed on the the plot() function making the plot.

Value

A list with 2 elements:

`lim.agree`	A matrix of limits of agreement as rows and estimate and c.i. as columns.
`p.value`	P-value for the hypothesis that the mean difference is 0. Usually a lame thing to use.

Author(s)

Jaro Lajovic,⟨jaro.lajovic@mf.uni-lj.si⟩, 2004; modified 2007 by Bendix Carstensen ⟨bxc@steno.dk⟩, http://www.biostat.ku.dk/~bxc.

References

JM Bland and DG Altman: Statistical methods for assessing agreement between two methods of clinical measurement, Lancet, i, 1986, pp. 307-310.

JM Bland and DG Altman. Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999.

See Also

`BA.plot`, `MethComp`.

Examples

```
data( ox )
par( mfrow=c(1,2) )
# Wrong to use mean over replicates
mtab <- with( ox, tapply( y, list(item, meth), mean ) )
CO <- mtab[,"CO"]
pulse <- mtab[,"pulse"]
BlandAltman( CO, pulse )
# (almost) Right to use replicates singly
oxw <- to.wide( ox )
CO <- oxw[,"CO"]
pulse <- oxw[,"pulse"]
BlandAltman( CO, pulse, mult=TRUE )
BlandAltman( CO, pulse, eqax=TRUE )
```

`bothlines` *Add regression lines to a plot*

Description

Add the regression lines of y on x AND x on y to the plot. Optionally add the line obtained by allowing errors in both variables (Deming regression).

Usage

```
bothlines(x, y, Dem = FALSE, sdr = 1, col = "black", ...)
```

Arguments

`x`	Numeric vector
`y`	Numeric vector
`Dem`	Logical. Should the Deming regression line be added too?
`sdr`	Numeric. The assumed ratio of standard deviations used in the Deming regression.
`col`	Colour of the lines. Can be a vector of up to 3 elements, one for each line.
`...`	Additional arguments passed on to `abline`, which does the actual plotting.

Value

None.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

`abline`.

Examples

```
data( ox )
oxw <- to.wide(ox)
attach( oxw )
plot( CO, pulse )
abline(0,1)
bothlines( CO, pulse, Dem=TRUE, col=rainbow(3), lwd=2 )
plot( CO, pulse,pch=16 )
abline(0,1, col=gray(0.7), lwd=2)
bothlines( CO, pulse, Dem=TRUE, col=c(rep("transparent",2),"black"), lwd=2 )
```

---

cardiac *Measurement of cardiac output by two different methods.*

Description

For each subject cardiac output is measured repeatedly (three to six times) by impedance cardiography (IC) and radionuclide ventriculography (RV).

Usage

```
data(cardiac)
```

Format

A data frame with 120 observations on the following 4 variables.

`meth` a factor with levels `IC` `RV`

`item` a numeric vector giving the item number.

`repl` a numeric vector with replicate number.

`y` the measuremnts of cardiac output.

Details

It is not entirely clear from the source whether the replicates are exchangeable within (method,item) or whether they represent pairs of measurements. From the description it looks as if replicates are linked between methods, but in the paper they are treated as if they were not.

Source

The dataset is adapted from table 4 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland & Altman by Dr LS Bowling, see: Bowling LS, Sageman WS, O'Connor SM, Cole R, Amundson DE. Lack of agreement between measurement of ejection fraction by impedance cardiography versus radionuclide ventriculography. Critical Care Medicine 1993; 21: 1523-27.

Examples

```
library( MethComp )
library( R2WinBUGS )
data(cardiac)
options( bugs.directory="c:/Stat/Bugs/WinBUGS14/" )
card.mi.ir <- MethComp( cardiac, random=c("mi","ir"), n.iter=100 )
card.mi    <- MethComp( cardiac, random=c("mi"), n.iter=100 )
card.mi.ir
card.mi
```

`Deming` *Regression with errors in both variables (Deming regression)*

Description

The function makes a regression of y on x, assuming that both x and y are measured with error. This problem only has an analytical solution if the ratio of the variances is known, hence this is required as an input parameter.

Usage

```
Deming(x, y, vr = sdr^2, sdr = sqrt(vr),
       boot = FALSE, keep.boot = FALSE, alpha = 0.05)
```

Arguments

`x`	numerical variable.
`y`	numerical variable.
`vr`	The assumed known ratio of the (residual) variance of the `y`s relative to that of the `x`s. Defaults to 1.
`sdr`	do. for standard deviations. Defaults to 1. `vr` takes precedence if both are given.
`boot`	Should bootstrap estimates of standard errors of parameters be done? If `boot==TRUE`, 1000 bootstrap samples are done, if `boot` is numeric, `boot` samples are made.
`keep.boot`	Should the 4-column matrix of bootstrap samples be returned? If `TRUE`, the summary is printed, but the matrix is returned invisibly. Ignored if `boot=FALSE`
`alpha`	What significance level should be used when displaying confidence intervals?

Details

The formal model underlying the procedure is based on a so called functional relationship:

$$x_i = \xi_i + e_{1i}, \qquad y_i = \alpha + \beta\xi_i + e_{2i}$$

with $\text{var}(e_{1i}) = \sigma$, $\text{var}(e_{2i}) = \lambda\sigma$, where λ is the known variance ratio.

The estimates of the residual variance is based on a weighting of the sum of squared deviations in both directions, divided by $n - 2$. The ML estimate would use $2n$ instead, but in the model we actually estimate $n + 2$ parameters — α, β and the n ξs.

This is not in Peter Sprent's book (see references).

Value

If `boot==FALSE` a named vector with components `Intercept`, `Slope`, `sigma.x`, `sigma.y`, where `x` and `y` are substituted by the variable names.

If `boot==TRUE` a matrix with rows `Intercept`, `Slope`, `sigma.x`, `sigma.y`, and colums giving the estimates, the bootstrap standard error and the bootstrap estimate and c.i. as the 0.5, $\alpha/2$ and $1 - \alpha/2$ quantiles of the sample.

If `keep.boot==TRUE` this summary is printed, but a matrix with columns `Intercept`, `Slope`, `sigma.x`, `sigma.y` and `boot` rows is returned.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc.

References

Peter Sprent: Models in Regression, Methuen & Co., London 1969, ch.3.4.

WE Deming: Statistical adjustment of data, New York: Wiley, 1943. [This is a reference taken from a reference list — I never saw the book myself].

See Also

MethComp

Examples

```
# Some data
x <- runif(100,0,5) + rnorm(100)
y <- 2 + 3 * x + rnorm(100,sd=2)
# Deming regression with equal variances, variance ratio 2.
Deming(x,y)
Deming(x,y,vr=2)
Deming(x,y,boot=TRUE)
bb <- Deming(x,y,boot=TRUE,keep.boot=TRUE)
str(bb)
# Plot data with the two classical regression lines
plot(x,y)
abline(lm(y~x))
ir <- coef(lm(x~y))
abline(-ir[1]/ir[2],1/ir[2])
abline(Deming(x,y,sdr=2)[1:2],col="red")
abline(Deming(x,y,sdr=10)[1:2],col="blue")
# Comparing classical regression and "Deming extreme"
summary(lm(y~x))
Deming(x,y,vr=1000000)
```

`fat` *Measurements of subcutaneous and visceral fat*

Description

43 persons had Subcutaneous and Visceral fat thickness measured at Steno Diabetes Center in 2006 by two observers; all measurements were done three times. The interest is to compare the measurements by the two observers. Persons are items, observers are methods, the three replicates are exchangeable within (person,observer)=(item,method)

Usage

```
data(fat)
```

Format

A data frame with 258 observations on the following 6 variables.

`Id` Person id.

`Obs` Observers, a factor with levels `KL` and `SL`.

`Rep` Replicate — exchangeable within person and observer.

`Sub` Subcutaneous fat measured in mm.

`Vic` Visceral fat measured in mm.

Examples

```
data(fat)
str(fat)
```

`glucose` *Glucose measurements by different methods*

Description

74 persons in 5 centres in Finland had blood glucose measured by 11 different methods, based on 4 different types of blood. Each person had blood sampled at 0, 30, 60 and 120 min after a 75 g glucose load.

Usage

```
data(glucose)
```

Format

A data frame with 1302 observations on the following 6 variables.

`meth` Method of measurement. A factor with 11 levels: `n.plas1` `n.plas2` `h.cap` `h.blood` `h.plas` `h.serum` `m.plas` `m.serum` `o.cap` `s.serum` `k.plas`.

`type` Type of blood sample. A factor with 4 levels: `blood` `plasma` `serum` `capil`

`item` Person id.

`time` Time of blood sampling. Minutes since glucose load.

`cent` Center of sampling. Except for the two first methods, `n.plas1` and `n.plas2`, samples were analyzed at the centres too

`y` Glucose measurement in mmol/l.

Source

The study was conducted at the National Public Health Institute in Helsinki by Jaana Lindström.

References

B Carstensen, J Lindström, J Sundvall, K Borch-Johnsen1, J Tuomilehto & the DPS Study Group: Measurement of Blood Glucose: Comparison between different Types of Specimens. Annals of Clinical Biochemistry, to appear.

Examples

```
data( glucose )
str( glucose )
# Use only plasma and serum as methods and make a Bland-Altman plot
gluc <- subset( glucose, type %in% c("plasma","serum") )
gluc$meth <- gluc$type
gluc$repl <- gluc$time
BA.plot( gluc )
```

`hba1c` *Measurements of HbA1c from Steno Diabetes Center*

Description

Three analysers (machines) for determination of HbA1c (glycosylated haemoglobin) were tested on samples from 38 individuals. Each had drawn a venous and capillary blood sample. These were analysed on five different days.

Usage

```
data(hba1c)
```

Format

A data frame with 835 observations on the following 6 variables.

`dev` Type of machine used. A factor with levels `BR.V2`, `BR.VC` and `Tosoh`.

`type` Type of blood analysed (capillary or venous). A factor with levels `Cap` `Ven`

`item` Person-id. A numeric vector

`d.samp` Day of sampling.

`d.ana` Day of laboratory analysis.

`y` The measured value of HbA1c.

Details

In the terminology of method comparison studies, methods is the cross-classification of `dev` and `type`, and replicate is `d.ana`. It may be of interest to look at the effect of time between `d.ana` and `d.samp`, i.e. the time between sampling and analysis.

Source

Bendix Carstensen, Steno Diabetes Center.

References

These data were analysed as example in: Carstensen: Comparing and predicting between several methods of measurement, Biostatistics 5, pp. 399–413, 2004.

Examples

```
data(hba1c)
str(hba1c)
```

`make.repl` *Generate a numbering of replicates within (item,method)*

Description

Replicate numbers are generated within (item,method) in a dataframe representing a method comparison study. The function assumes that observations are in the correct order within each (item,method), i.e. if replicate observations are non-exchangeable within method, linked observations are assumed to be in the same order within each (item,method).

Usage

```
make.repl( data )
```

Arguments

`data`	A data frame with columns `meth`, `item` and `y`.

Value

A dataframe with a column, `repl` added or replaced.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

`perm.repl`

Examples

```
data(ox)
xx <- subset( ox, item<4 )[,-3]
cbind( xx, make.repl(xx) )
cbind( make.repl(xx), perm.repl(xx) )
```

`MethComp` *Fit a model for method comparison studies using WinBUGS*

Description

A model linking each of a number of methods of measurement linearly to the "true" value is set up in BUGS and run via the function `bugs` from the `R2WinBUGS` package.

Usage

```
MethComp( data,
        random = c("mi", "ir"),
          beta = TRUE,
      n.chains = 3,
        n.iter = 2000,
      n.burnin = n.iter/2,
        n.thin = ceiling((n.iter - n.burnin)/1000),
bugs.directory = options("bugs.directory")[[1]],
         debug = FALSE,
       clearWD = TRUE,
bugs.code.file = "qwzx.bug",
     code.only = FALSE,
           ... )
## S3 method for class 'MethComp':
summary(object, ...)
## S3 method for class 'MethComp':
print(x, across, digits=3, ... )
```

Arguments

`data`	Data frame with variables `meth`, `item`, `repl` and `y`. `y` represents a measurement on an `item` (typically patient or sample) by method `meth`, in replicate `repl`.
`random`	Which random effects should be included in the model?. Enter `NULL` if none is desired.
`beta`	Logical. Should a slope other than 1 be allowed? If `FALSE` the bias between methods will be assumed constant.
`n.chains`	How many chains should be run by WinBUGS — passed on to `bugs`.
`n.iter`	How many total iterations — passed on to `bugs`.
`n.burnin`	How many of these should be burn-in — passed on to `bugs`.
`n.thin`	How many should samples — passed on to `bugs`.
`bugs.directory`	Where is WinBUGS (>=1.4) installed — passed on to `bugs`. The default is to use a parameter from options(). If you use this routinely, this is most conveniently set in your `.Rprofile`.
`debug`	Should WinBUGS remain open after running — passed on to `bugs`.
`clearWD`	Should the working directory be cleared for junk files after the running of WinBUGS — passed on to `bugs`.
`bugs.code.file`	Where should the bugs code go?
`code.only`	Should `MethComp` just create a bugs code file and a set of inits?
`...`	Additional arguments passed on to bugs.
`object`	A `MethComp` object
`x`	A `MethComp` object
`across`	Should the summary of conversion formulae be printed with α, β and prediction sd. across or down?
`digits`	Number of digits after the decimal point when printing.

Details

The model set up for an observation y_{mir} is:

$$y_{mir} = \alpha_m + \beta_m(\mu_i + b_{ir} + c_{mi}) + e_{mir}$$

where b_{ir} is a random `item` by `repl` interaction (included if `"ir" %in% random`) and c_{mi} is a random `meth` by `item` interaction (included if `"mi" %in% random`). The μ_i's are parameters in the model but are not monitored — only the αs, βs and the variances of b_{ir}, c_{mi} and e_{mir} are monitored and returned. The estimated

parameters are only determined up to a linear transformation of the μs, but the linear functions linking methods are invariant. The identifiable conversion parameters are:

$$\alpha_{m\cdot k} = \alpha_m - \alpha_k\beta_m/\beta_k, \quad \beta_{m\cdot k} = \beta_m/\beta_k$$

The posteriors of these are derived and included in the `posterior`, which also will contain the posterior of the variance components (the sd's, that is). Furthermore, the posterior of the point where the conversion lines intersects the identity as well as the prediction sd's between any pairs of methods are included.
The function `summary.MethComp` method gives estimates of the conversion parameters that are consistent. Clearly,

$$\text{median}(\beta_{1\cdot 2}) = 1/\text{median}(\beta_{2\cdot 1})$$

because the inverse is a monotone transformation, but there is no guarantee that

$$\text{median}(\alpha_{1\cdot 2}) = \text{median}(-\alpha_{2\cdot 1}/\beta_{2\cdot 1})$$

and hence no guarantee that the parameters derived as posterior medians produce conversion lines that are the same in both directions. Therefore, `summary.MethComp` computes the estimate for $\alpha_{2\cdot 1}$ alpha.2.1 as

$$(\text{median}(\alpha_{1\cdot 2}) - \text{median}(\alpha_{2\cdot 1})/\text{median}(\beta_{2\cdot 1}))/2$$

and the estimate of $\alpha_{1\cdot 2}$ correspondingly. The resulting parameter estimates defines the same lines.

Value

If `code.only==FALSE`, an object of class `MethComp` which is a list with three components:

`summary`	Matrix with a summary of the posterior of the variance components and the parameters linking the methods.
`posterior`	Dataframe with the posterior samples of the interesting parameters.
`org.summary`	Summary of the original parameters as monitored by `WinBUGS`.
`random`	A character sting indicationg which random effects are in the model.
`methods`	A character string of the names of the methods.
`data`	The original data frame used in the computations. This is intended for us in `plot.MethComp`.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc, Lyle Gurrin, University of Melbourne, http://www.epi.unimelb.edu.au/about/staff/gurrin-lyle.

References

B Carstensen: Comparing and predicting between several methods of measurement, Biostatistics, 5, pp 399-413, 2004

See Also

`BA.plot`, `\code{BA.plot}`

Examples

```
data( ox )
str( ox )
MethComp( ox, code.only=TRUE, bugs.code.file="ox-ex.bug", random=c("mi") )
shell( "type ox-ex.bug" ) # only works on windows

### These next lines only work if you properly name the path to WinBUGS
### What is written here is not necessarily correct on your machine.
library(R2WinBUGS)
# options( bugs.directory="c:/Program Files/WinBUGS14/" )
options( bugs.directory="c:/Stat/Bugs/WinBUGS14/")
ox.res <- MethComp( ox, random=c("mi"), n.iter=100 )
str( ox.res )
str( ox.res[[2]] )
print( ox.res )
```

`milk` *Measurement of fat content of human milk by two different methods.*

Description

Fat content of human milk determined by measurement of glycerol released by enzymic hydrolysis of triglycerides (Trig) and measurement by the Standard Gerber method (Gerber). Units are (g/100 ml).

Usage

```
data(milk)
```

Format

A data frame with 90 observations on the following 3 variables.

`meth` a factor with levels `Gerber Trig`

`item` sample id

`y` a numeric vector

Source

The dataset is adapted from table 3 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. See: Lucas A, Hudson GJ, Simpson P, Cole TJ, Baker BA. An automated enzymic micromethod for the measurement of fat in human milk. Journal of Dairy Research 1987; 54: 487-92.

Examples

```
data(milk)
str(milk)
plot(milk)
plot( y[meth=="Trig"]~y[meth=="Gerber"],data=milk,
      xlab="Fat (g/100 ml; Gerber)",
      ylab="Fat (g/100 ml; Trig.)")
abline(0,1)
```

`ox` *Measurement of oxygen saturation in blood*

Description

61 children had their blood oxygen content measured at the Children's Hospital in Melbourne, either with a chemical method analysing gases in the blood (`CO`) or by a pulse oximeter measuring transcutaneously (`pulse`). Replicates are linked between methods; i.e. replicate 1 for each of the two methods are done at the same time. However, replicate measurements were taken in quick succession so the pairs of measurements are exchangeable within person.

Usage

```
data(ox)
```

Format

A data frame with 354 observations on the following 4 variables.

`meth` Measurement methods, factor with levels `CO`, `pulse`

`item` Id for the child

`repl` Replicate of measurements. There were 3 measurements for most children, 4 had only 2 replicates with each method, one only 1

`y` Oxygen saturation in percent.

Examples

```
data(ox)
str(ox)
with( ox, table(table(item)) )
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot( ox, ymax=20 )
BA.plot( ox, ymax=20, mean.repl=TRUE )
```

`perm.repl` *Randomly permute replicates within methods*

Description

Replicates are randomly permuted within (item,method) in a dataframe representing a method comparison study.

Usage

```
perm.repl( data )
```

Arguments

`data`	A data frame with columns `meth`, `item` and `y`. If `repl` is in the dataframe too, it will be overwritten.

Details

The function is designed to explore the effect of exchanging the replicates within (item,method). If replicates are truly exchangeable within methods, the inference should independent of this permutation.

Value

A dataframe where the rows (i.e. replicates) are randomly permuted within (`meth`,`item`), and subsequently ordered by (`meth`,`item`,`repl`).

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

`make.repl`

Examples

```
data(ox)
xx <- subset( ox, item<4 )
cbind( xx, perm.repl(xx) )
# Replicates are linked in the oximetry dataset, so randomly permuting
# them clearly inflates the limits of agreement:
par( mfrow=c(1,2), mar=c(4,4,1,4) )
BA.plot(           ox , ymax=30, digits=1 )
BA.plot( perm.repl(ox), ymax=30, digits=1 )
```

`plot.meth` *Make pairwise scatter plots and Bland-Altman plot of several methods.*

Description

Makes scatter plots (below diagonal) and Bland-Altman plots (above diagonal) for method comparison data. All panels have the same scale extent.

Usage

```
plot.meth(data,
          which = NULL,
         col.LA = "blue",
       cex.name = 2,
      var.range,
     diff.range,
      var.names = FALSE )
```

Arguments

`data`	Data frame with data.
`which`	Which variables represent the different methods. If the dataframe has a column `meth`, data are automatically converted to wide format and all methods compared.
`col.LA`	What color should be used for the limits of agreement.
`cex.name`	Character expansion factor for plotting method names
`var.range`	The range of the axes in the scatter plot and the x-axis in the Bland-Altman plot be?
`diff.range`	The range of yaxis in the Bland-Altman plot. Defaults to a range as the x-axis, but centered around 0.
`var.names`	If logical: should the individual panels be lebelled with the variable names?. If character, then the values of the character will be used to label the methods.

Value

None.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

`plot.MethComp`

Examples

```
data( sbp )
plot.meth( sbp )
plot.meth( sbp, var.names=TRUE )
```

`plot.MethComp` *Plot estimated conversion lines and formulae.*

Description

Plots the pairwise conversion formulae between methods from a `MethComp` object.

Usage

```
plot.MethComp( x,
             axlim = range( attr(x,"data")$y ),
             which,
          lwd.line = c(3,1), col.line = rep("black",2), lty.line=rep(1,2),
               eqn = TRUE, digits = 2,
              grid = FALSE, col.grid=gray(0.8),
            pl.obs = FALSE,
           col.pts = "black", pch.pts = 16, cex.pts = 0.8,
               ... )
```

Arguments

`x`	A `MethComp` object
`axlim`	The limits for the axes in the panels
`which`	Numeric vector or vector of method names. Which of the methods should be included in the plot?
`lwd.line`	Numerical vector of length 2. The width of the conversion line and the prediction limits. If the second values is 0, no prediction limits are drawn.
`col.line`	Numerical vector of length 2. The color of the conversion line and the prediction limits.
`lty.line`	Numerical vector of length 2. The line types of the conversion line and the prediction limits.
`eqn`	Should the conversion equations be printed on the plot?. Defaults to `TRUE`.
`digits`	How many digits after the decimal point shoudl be used when printing the conversion equations.
`grid`	Should a grid be drawn? If a numerical vector is given, the grid is drawn at those values.
`col.grid`	What color should the grid have?
`pl.obs`	Logical or character. Should the points be plotted. If `TRUE` or `"repl"` paired values of single replicates are plotted. If `"perm"`, replicates are randomly permuted within (item, method) befor plotting. If `"mean"`, means across replicates within item, method are formed and plotted.
`col.pts`	What color should the observation have.
`pch.pts`	What plotting symbol should be used.
`cex.pts`	What scaling should be used for the plot symbols.
`...`	Parameters to pass on. Currently not used.

Value

Nothing. The lower part of a (M-1) by (M-1) matrix of plots is drawn, showing the pairwise conversion lines. In the corners of each is given the two conversion equations together with the prediction standar error.

See Also

MethComp, print.MethComp

Examples

```
data( hba1c )
str( hba1c )
hba1c <- transform( hba1c, meth = interaction(dev,type),
                           repl = d.ana )
# options( bugs.directory="c:/Program Files/WinBUGS14/" )
options( bugs.directory="c:/Stat/Bugs/WinBUGS14/")
hb.res <- MethComp( hba1c, n.iter=100 )
str( hb.res )
par( ask=TRUE )
plot( hb.res )
plot( hb.res, pl.obs=TRUE, which=1:4 )
```

`plot.VarComp` *Plot the a posteriori densities for variance components*

Description

When a method comparison model i fitted and stored in a MethComp object, then the posterior distributions of the variance components are plotted, in separate displays for method.

Usage

```
plot.VarComp( x,
          which,
       lwd.line = rep(2, 4),
       col.line = c("red", "green", "blue", "black"),
       lty.line = rep(1, 4),
           grid = TRUE,
       col.grid = gray(0.8),
            rug = TRUE,
          probs = c(5, 50, 95),
        tot.var = FALSE,
        same.ax = TRUE,
      meth.names = TRUE,
        VC.names = "first",
             ... )
```

Arguments

`x`	A `MethComp` object.
`which`	For which of the compared methods should the plot be made?
`lwd.line`	Line width for drawing the density.
`col.line`	Color for drawing the densities.
`lty.line`	Line type for drawing the densities.
`grid`	Logical. Should a vertical grid be set up? If numeric it is set up at the values specified. If `same.ax`, the range of the grid is taken to be the extent of the x-axis for all plots.
`col.grid`	The color of the grid.
`rug`	Should a small rug at the bottom show posterior quantiles?
`probs`	Numeric vector with numbers in the range from 0 to 100, indicating the posterior percentiles to be shown in the rug.

`tot.var`	Should the posterior of the total variance also be shown?
`same.ax`	Should the same axes be used for all methods?
`meth.names`	Should the names of the methods be put on the plots?
`VC.names`	Should the names of the variance components be put on the first plot (`"first"`), the last (`"last"`), all (`"all"`) or none (`"none"`). Only the first letter is needed.
`...`	Parameters passed on the `density` furnction that does the smoothing of the posterior samples.

Details

The function generates a series of plots, one for each method compared in the `MethComp` object supplied (or those chosen by `which=`). Therefore the user must take care to set `mfrow` or `mfcol` to capture all the plots.

Value

A list with one element for each method. Each element of this is a list of densities, i.e. of objects of class `density`, one for each variance component.

Author(s)

Bendix Carstensen, www.biostat.ku.dk/~bxc

See Also

`plot.MethComp`, `MethComp`

Examples

```
data( ox )
MC.ox <- MethComp( ox )
par( mfrow=c(2,1) )
plot.VarComp( MC.ox, grid=c(0,15) )
```

`plvol` *Measurements of plasma volume measured by two different methods.*

Description

For each subject (`item`) the plasma volume is expressed as a percentage of the expected value for normal individuals. Two alternative sets of normal values are used, named Nadler and Hurley respectively.

Usage

```
data(plvol)
```

Format

A data frame with 198 observations on the following 3 variables.

`meth` a factor with levels `Hurley` `Nadler`

`item` a numeric vector

`y` a numeric vector

Source

The datset is adapted from table 2 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland & Altman by C Doré, see: Cotes PM, Doré CJ, Liu Yin JA, Lewis SM, Messinezy M, Pearson TC, Reid C. Determination of serum immunoreactive erythropoietin in the investigation of erythrocytosis. New England Journal of Medicine 1986; 315: 283-87.

Examples

```
data(plvol)
str(plvol)
plot( y[meth=="Nadler"]~y[meth=="Hurley"],data=plvol,
      xlab="Plasma volume (Hurley) (pct)",
      ylab="Plasma volume (Nadler) (pct)" )
abline(0,1)
par( mar=c(4,4,1,4) )
BA.plot(plvol)
```

`sbp` *Systolic blood pressure measured by three different methods.*

Description

For each subject (`item`) there are three replicate measurements by three methods (two observers, J and R and the automatic machine, S). The replicates are exchangeable within method, item.

Usage

```
data(sbp)
```

Format

A data frame with 765 observations on the following 4 variables:

`meth` Methods, a factor with levels `J`(observer 1), `R`(observer 2) and `S`(machine)

`item` Person id, numeric.

`repl` Replicate number, a numeric vector

`y` Systolic blood pressure masurement, a numeric vector

Source

The dataset is adapted from table 1 in: JM Bland and DG Altman: Measuring agreement in method comparison studies. Statistical Methods in Medical Research, 8:136-160, 1999. Originally supplied to Bland & Altman by E. O'Brien, see: Altman DG, Bland JM. The analysis of blood pressure data. In O'Brien E, O'Malley K eds. Blood pressure measurement. Amsterdam: Elsevier, 1991: 287-314.

Examples

```
data(sbp)
par( mfrow=c(2,2), mar=c(4,4,1,4) )
BA.plot( sbp, comp=1:2 )
BA.plot( sbp, comp=2:3 )
BA.plot( sbp, comp=c(1,3) )
library( R2WinBUGS )
options( bugs.directory="c:/Stat/Bugs/WinBUGS14/" )
# Grossly inadequate number of iterations
sbp.1 <- MethComp( sbp, random=c("mi"), n.iter=100 )
sbp.1
```

`scint` *Relative renal function by Scintigraphy*

Description

Measurements of the relative kidney function (=renal function) for 111 patients. The percentage of the total renal function present in the left kidney is determined by one reference method, `DMSA` (static) and by one of two dynamic methods, `DTPA` or `EC`.

Usage

```
data(scint)
```

Format

A data frame with 222 observations on the following 5 variables:

`meth` Measurement method, a factor with levels `DMSA`, `DTPA`, `EC`.

`item` Patient identification.

`y` Percentage of total kidney function in the left kidney.

`age` Age of the patient.

`sex` Sex of the patient, a factor with levels `F`, `M`.

Source

F. C. Domingues, G. Y. Fujikawa, H. Decker, G. Alonso, J. C. Pereira, P. S. Duarte: Comparison of Relative Renal Function Measured with Either 99mTc-DTPA or 99mTc-EC Dynamic Scintigraphies with that Measured with 99mTc-DMSA Static Scintigraphy. International Braz J Urol Vol. 32 (4): 405-409, 2006

Examples

```
data(scint)
str(scint)
# Make a Bland-Altman plot for each of the possible comparisons:
par(mfrow=c(1,2),mgp=c(3,1,0)/1.6,mar=c(3,3,1,3))
BA.plot(scint,comp.levels=c(1,2),ymax=15,digits=1,cex=2)
BA.plot(scint,comp.levels=c(1,3),ymax=15,digits=1,cex=2)
```

`sim.meth` *Simulate a dataframe containing replicate measurements on the same items using different methods.*

Description

A dataframe is simulated that represents data from a method comparison study based on parameters specified by the user.

Usage

```
sim.meth( Ni = 100,
          Nm = 2,
          Nr = 3,
          nr = Nr,
       alpha = rep(0,Nm),
        beta = rep(1,Nm),
    mu.range = c(0, 100),
```

```
 sigma.mi = rep(5,Nm),
 sigma.ir = 2.5,
sigma.mir = rep(5,Nm),
   m.thin = 1,
   i.thin = 1 )
```

Arguments

`Ni`	The number of items (patient, animal, sample, unit etc.)
`Nm`	The number of methods of measurement.
`Nr`	The (maximal) number of replicate measurements for each (item,method) pair.
`nr`	The minimal number of replicate measurements for each (item,method) pair. If `nr<Nr`, the number of replicates for each (meth,item) pair is uniformly distributed on the points `nr:Nr`, otherwise `nr` is ignored. Different number of replicates is only meaningful if replicates are not linked, hence `nr` is also ignored when `sigma.ir>0`.
`alpha`	A vector of method-specific intercepts for the linear equation relating the "true" underlying item mean measurement to the mean measurement on each method.
`beta`	A vector of method-specific slopes for the linear equation relating the "true" underlying item mean measurement to the mean measurement on each method.
`mu.range`	The range across items of the "true" mean measurement. Item means are uniformly spaced across the range.
`sigma.mi`	A vector of method-specific standard deviations for a method by item random effect. Some or all components can be zero.
`sigma.ir`	Method-specific standard deviations for the item by replicate random effect.
`sigma.mir`	A vector of method-specific residual standard deviations for a method by item by replicate random effect (residual variation). All components must be greater than zero.
`m.thin`	Fraction of the observations from each method to keep.
`i.thin`	Fraction of the observations from each item to keep. If both `m.thin` and `i.thin` are given the thinning is by their componentwise product.

Details

Data are simulated according to the following model for an observation y_{mir}:

$$y_{mir} = \alpha_m + \beta_m(\mu_i + b_{ir} + c_{mi}) + e_{mir}$$

where b_{ir} is a random `item` by `repl` interaction (with standard deviation for method m the corresponding component of the vector $\sigma_i r$), c_{mi} is a random `meth` by `item` interaction (with standard deviation for method m the corresponding component of the vector $\sigma_m i$) and e_{mir} is a residual error term (with standard deviation for method m the corresponding component of the vector $\sigma_m ir$). The μ_i's are uniformly spaced in a range specified by `mu.range`.

Value

A dataframe with columns `meth`, `item`, `repl` and `y`, representing results from a method comparison study.

Author(s)

Lyle Gurrin, University of Melbourne, http://www.epi.unimelb.edu.au/about/staff/gurrin-lyle, Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

tab.repl, plot.meth, MethComp

Examples

```
sim.meth( Ni=4, Nr=3 )
xx <- sim.meth( Nm=3, Nr=5, nr=2, alpha=1:3, beta=c(0.7,0.9,1.2), m.thin=0.7 )
tab.repl( xx )
plot.meth( xx )
```

`tab.repl` *Table of replicates by each method*

Description

Creates a table classified by method and no. of replicate measurments which in each entry has the number of items with that number of replicates on that method.

Usage

```
tab.repl(data)
```

Arguments

`data`	Data frame with variables `meth`, `item`, `repl` and `y`. `y` represents a measurement on an `item` (typically patient or sample) by method `meth`, in replicate `repl`.

Value

A table classified by method and no. of replicate measurements. Two colums with the total number of items and measuremnts on each method is at added to the table.

Author(s)

Bendix Carstensen, ⟨bxc@steno.dk⟩

See Also

MethComp

Examples

```
data(ox)
tab.repl(ox)
```

`to.wide` *Functions to convert between long and wide representations of data.*

Description

These functions are merely wrappers for `reshape`. Given the complicated syntax of `reshape` and the particularly simple structure of this problem, the functions facilitate the conversion enormously.

Usage

```
to.wide( data )
to.long( data, vars )
```

Arguments

`data`	A dataframe
`vars`	The variables representing measurements by different methods. Either a character vector of names, or a numerical vector with the number of the variables in the dataframe.

Details

If `data` represents method comparisons with exchangeable replicates within method, the transformation to wide format does not necessarily make sense.

Value

A dataframe.

Author(s)

Bendix Carstensen, Steno Diabetes Center, http://www.biostat.ku.dk/~bxc

See Also

`tab.repl`

Examples

```
data( milk )
str( milk )
mw <- to.wide( milk )
str( mw )
( mw <- subset( mw, item < 3 ) )
to.long( mw, 3:4 )
```