Introduction to the R package *sugaR*

Plots for presenting a diabetes diary

Steffen Möller
steffen_moeller@gmx.de

May 29, 2009

1 Motivation

Diabetes is a complex disease - in its molecular etiology and for its impact on the patient's life. While the disease and the patient develop, continuous updates of the treatment are required. Whenever things become complicated, which is when the first diagnosis is made or when the remission ("honeymoon") ends, one needs to think - and talk between doctors, the patient and/or the parents. One needs a diagram where the key information is collected, if possible on a single page.

The Internet offers several tools to display sugar levels and insulin dosage over time. Such are helpful, but in their presentation of sugar levels, the temporal development of those values are commonly taken out of their respective context. This may be a result of the commonly observed emphasis of tools on the collection of data. And with much data being available, the tracking of measurements over the day becomes difficult if not impossible. When the response to insulin however is changing rapidly, one is likely to solely concentrate on the past week or a shorter time for the analysis - older values are obsolete. Also, one will take as much information into account as the self-prepared notes allow to derive from them. This is more than the typical (time,concentration,carbs,units) quadruplets, i.e. the activity can be estimated, the onset of sleep, stress, ... arbitrary regular and irregular events.

The focus of this package is on the presentation, not on the data gathering. Consequently, in order to fully exploit the functionality of this package one may need to add data to the system that is not (yet) available electronically in routine. The package today offers to use symbols with the typical line plots to help investigating the interplay of glucose and insulin. The connecting lines allow the tracking of the development over the day. The thickness of these lines represent the physical activity. This supports finding explanations for the variability of measurements and a more hollistic modelling.

Note: these plots shall help communicating the contents of your diabetes diary. No more than that. Consult a doctor just the way you did before, just possibly with sugaR-crafted plots in your pocket.

2 The data - and a basic introduction to R data types

Admittedly, the input of data for the moment is still too tedious. Later versions of this packages shall be able to interface with other software, of which there is plenty, to collect the data manually with a graphical user interface or to read out glucose meters.

For now, the data is expected in a form that can be directly given to the statistics suite R. This is intuitive for those who are already good at R programming, but demands some training for novices. This section explains how to prepare your own input.

2.1 Installation

This package needs the statistics suite R to run. The R shell can be started like every other program and is available on all current computing platforms from `http://www.r-project.org`. The software can be installed like other packages in the CRAN archive by the R command `install.packages("sugaR")`.

2.2 Lists and Arrays

Lists are one-dimensional arrays, where every entry is accessible under a special name - or it index. Arrays can be of any dimension. This package uses two-dimensional arrays, i.e. tables to pair the time of the day with the sugar concentrations and/or other traits of the disease. The tables don't need to be entered directly. Instead, routines

$u(string)$	conversion of string in "$HH : MM$" format into a floating point number of the value $HH + (MM/60)$
$h(string)$	converts a string of comma-sparated rows of a table into a matrix, for which every row first defines a time in HH:MM format and an arbitrary number of values.

Some basics for dealing with R:

`a <- 5`	– assignment of the value 5 to a variable named a
`a <- c(1,2)`	– assignment of the tupel (1,2)
`list("a"=1, "b"=c(1,2))`	– list with two heterogeneous entries
`u("18:10")`	– call of function u with string argument, function u is provided by this package to convert the string into a numeric value

The sugar.over.time function expects all parameters to be passed as lists. For most types of data, this area lists of days, and for every day a matrix collects values over time. To ease data entry, the function h transforms a single string into a matrix. The string shall separate measurements by commas, and multiple values measured shall be separated by blanks. The first value is always the time, converted by the above mentioned function u. This way, the string "8 180,9

190,10 200,11 190,12 180" is converted by h to what would otherwise be produced by `matrix(c(8,180,9,190,10,200,11,190,12,180),2,byrow=TRUE)`:

time	values
8	180
9	190
10	200
11	190
12	180

2.3 Invocation and documentation

The example data presented here can be loaded with the R commands `library(sugaR); data(diabetesDiary)`. An invocation of the sugar.over.time function can be observed with `library(sugaR); example(sugar.over.time)`. `library` needs to be called only once for every R shell started.

Documentation on R in general is available on the Project's web site. Online documentation can be spawned from within the R shell:

?command	– lookup of man page of that command
??string	– search for string in available man pages
help.search("string")	– like ??string
vignette()	– lists vignettes (like this document) and can open them

This package has man pages with examples for the functions u, h and *sugar.over.time*. The latter is the function that produces the figures. The following sections explain the preparation of the data set. The function is then invoked just the way, the *help.search* function is invoked, following the schema sugar.over.time(data.$datatype$=$someValue$,...) .

2.4 Sugar measurements

Sugar measurements are presented as lists of arrays of measurements. Every list entry shall represent a day. The name of the list entry is the date in the YYYYMMDD format. Example:

```
myGlucose<-list(
"20090426"=h("8:47 139,11:12 75,16:18 112,19:46 71,21:48 75",ncol=2),
"20090501"=h("8:00 212,12:12 291,13:49 236,15:42 291,17:42 112,19:15
                    74,20:35 202,22:39 139",ncol=2)
)
```

The h function needs an extra argument to learn that there should always be pairs of time and a single value to be expected. This sums up to two columns, hence the argument $ncol = 2$. To invoke the sugar.over.time function with this data, one would call `sugar.over.time(data.glucose=myGlucose)`.

2.5 Intake

The amount of insulin injected is commonly noted together with the amount of carbohydrates taken with the food. Extra insulin injected to correct for hyperglucaemia is noted separately again. This led to the following representation:

```
myIntake<-list(
"20090426"=h("8:47 1.5 1.3,10:00 0.45,11:00 0.45, 11:12 2 0.8,12:17
   2.7 2.1,12:24 1.9 1.5,12:25 0.4 0.3,12:27 1.5 1.2,13:09 0.3
   0.2,13:30 0.45,16:18 2.5 2.0,17:09 2.5 2.0,17:29 2 1.5,18:22
   0.66666,18:28 0.3333333,19:46 0.5,19:56 0.7 0.4,20:00 1.2 0.7,20:10
   2 1.2,20:11 0.5 0.3,21:48 0.7",ncol=4),
"20090427"=h(paste(
        "6:48 1.5 1.3 0.7,10:15 1 0.6 0.4,10:19 0.27 0.2,",
        "12:51 1.65 1.3,13:22 0.3 0.2,13:37 2 1.6,13:49 1 0.8,",
"15:04 1 0 0,18:28 2 1.2,18:50 1 0.6,19:38 1.5 0.9,",
"22:07 0 0 0.6",sep=""),ncol=4)
# more data omitted
)
```

The arguments to h will be long. The day 20090427 has its argument broken into multiple lines to help with the presentation in this document. The "paste" of the partial strings to a single line is not required. As it can be seen from the specification of the previous day, the newlines and blanks in the string will be ignored. But when entering your own data, it may be found preferable to use no more than a single line per day. The data produced by h for the first day is of the type

time	carbs/12	bolus	correction
8.78	1.50	1.3	0
10.00	0.45	0.0	0
11.00	0.45	0.0	0
11.20	2.00	0.8	0
12.28	2.70	2.1	0
12.40	1.90	1.5	0
12.42	0.40	0.3	0
12.45	1.50	1.2	0
13.15	0.30	0.2	0
13.50	0.45	0.0	0
16.30	2.50	2.0	0
...			

The table now expected has four columns, hence the argument $ncol = 4$. To invoke the sugar.over.time function with this data, one would call `sugar.over.time(data.glucose=myGlucose, data.intake=myIntake)`.

2.6 Basal rate

The basal rate is either a list of pairs (start time, value) or of (duration, value) with initial start time at midnight. This application chose the second way to represent the basal rate. The notation below offers only specify the hourly rates.

```
myBasal<-list(
#               0  1   2  3   4   5  6  7  8  9  10  11 12 13  14  15 16  17  18  19 20 21 2
  "A'"=cbind(1,c(.3,.25,.25,.3,.4,.45,.5,.5,.5,.4,.15,.1,.1,.15,.15,.3,.45,.45,.45,.5,.55,.6

)
```

There is yet no direct link between the sugar values and the selection of profiles for basal rates. An idea how to achieve this properly still needs to arise. To invoke the sugar.over.time function with this data, one would call `sugar.over.time(data.glucose=myGlucose, data.intake=myIntake, data.basal=myBasal)`.

2.7 Activity

The description of physical activity over the day is important for the interpretation of the data. It represented by the thickness of connecting lines. The default is 1 (weak). The max is 5.

```
myActivities<-list(
  "20090501"=list(
      list(f=u("15:00"),t=u("16:15"),a=3,c="medium-crazy"),
      list(f=u("18:00"),t=u("19:15"),a=2,c="walking the dog")
  )
)
```

This time, two times are needed, i.e. one for the start time and one for the end. The a entry denotes the degree of activity, c is the comment. In the current implementation, the comments are not shown in the graph. To invoke the sugar.over.time function with this data, one would call `sugar.over.time(data.glucose=myGlucose, data.intake=myIntake, data.basal=myBasal, data.activities=myActivities)`.

2.8 Factors for determining insulin dosage

Lastly, for the presentation of insulin levels, the reference is important. That reference is dynamic, it depends on the amount of carbs consumed. Commonly, this relation is proportional, time-dependent linear factors are determined. These are again pairs of start times and a single value, the factor.

```
myFactors<-list(
  "20090101"=matrix(
    c(u("0:00"),0.6,
```

```
      u("6:00"),0.8,
      u("11:00"),0.8,
      u("16:00"),0.7,
      u("18:00"),0.6),byrow=2)
)
```

The h function could have been used rather than explicitly writing the $u(..)$ function on every row. The ordering of the days in the list and also of the times within the day is important for the subsequent interpretation of the data. The current implementation performs no checks on the semantics of the data presented - which probably it should.

To invoke the sugar.over.time function with this data, one would call `sugar.over.time(data.glucose=myGlucose, data.intake=myIntake, data.basal=myBasal, data.activities=myActivities, data.factors=myFactors)`.

3 The plot

This package offers some helper functions for the data input, the core functionality however is gathered in the *sugar.over.time* function. The plot it produces shows

- the sugar concentration,
- carbohydrate consumption,
- insulin dosage and
- physical activity

over time in one single graph. The basal rates may be shown in a second plot underneath.

Once accustomed to the way the data is represented, the figure is comparatively straight-forward to interpret. The following sections introduce every graphical feature one at a time.

Days and colours

In the current implementation, continous measurements are presented with the same colour. Every day has its very own colour, though[1]. On the plot, all colours are shown in the same intensity. In the same order as the data appears in the input lists, days are first shown in colours of long wavelenght (red) and then with continously shorter wavelengths towards blue. This way, besides the time of the day, also the date is coded into the plot - albeit less obviously.

[1] Future versions will allow not only the nights, but also other events, i.e. the change of the catheter to change the colour.

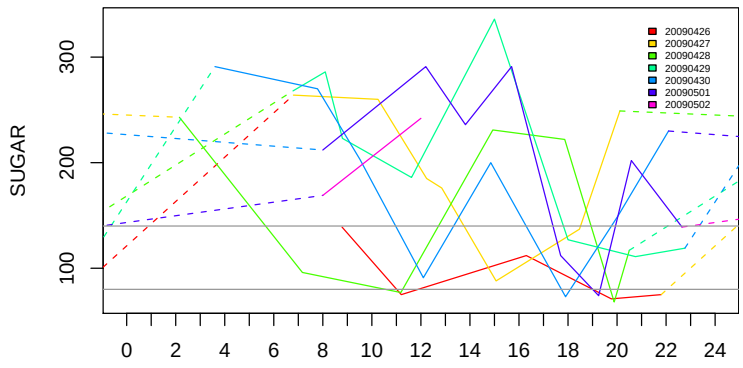

Figure 1: Blood glucose concentration - lines connect measurements, dotted between days

Glucose level

The sugar values are given as (time,concentration) points and subsequent measurements are connected by lines. Multiple days are such shown in a single graph as represented by figure 1. It is produced by the sugar.over.time function with only the data.glucose parameter set.

The lines between days are dotted. This is since over the night there are commonly fewer measurements than over the day, thus the confidence in the interpolation is much lower - if any. Every day in represented by a different colour. The dotted lines are presented in the colour of the respective upcoming or previous day.

Activity

A strong activity may help explaining later observed drops in blood glucose levels. This package may be unique in displaying this information. The activity is represented by the thickness of the line.

The example data of this first release will be edited further, figure 2 shows only one activity such activity (blue line).

Carbohydrate intake

The intake of food is indicated by filled circles. The area or the radius can be set to represent the amount taken in. The size is relative to the maximal value found in the graph and such as such be invariant to the actual unit as long as it linearly scales to gramm (e.g. Broteinheiten are just fine (gramm/12)).

When looking at figure 2, one will find it still difficult to make much sense out of it. The question to investigate is if the consumption of many carbohydrates are somehow correlated to subsequent observations of hyper- or hypglycaemia.

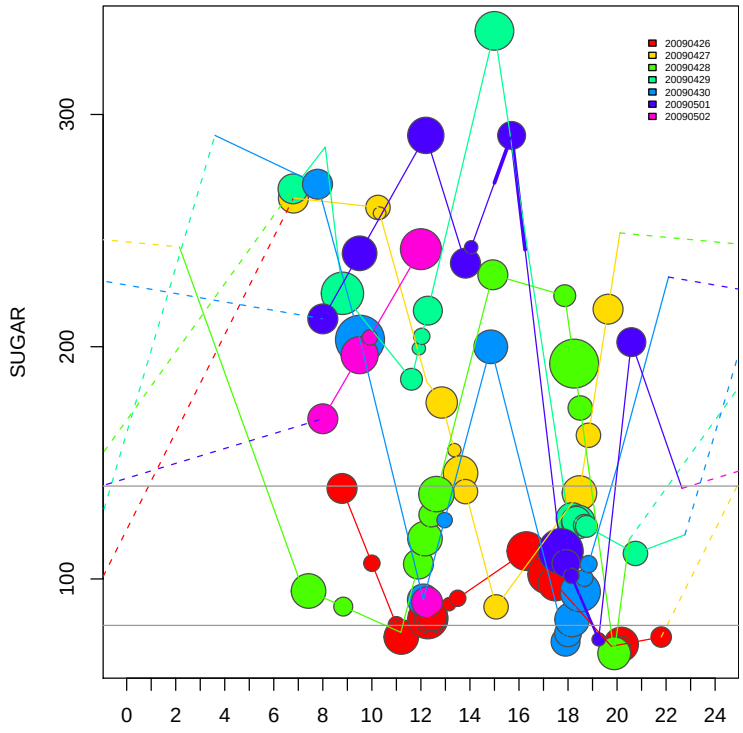

Figure 2: Sugar levels shown together with carbohydrate intake (filled circles) and activity (line width)

This would indicate a requirement to change the factor. The figure seems to indicate that larger carbohydrate-rich meals in the evening are followed by a drop in the blood sugar value, while lower food uptake led to an increase. The evening factor therefore needs to be lowered, and the basal rate increased. What is missing are graphs to investigate that correlation directly without omitting essential information. Please expect first such prototypes over the summer of 2009.

Looking at the graph one will find events that are not immediately understood, e.g. a big meal at high glucose levels. The plot needs further annotation here, which can be expected for the next release. Then, one may for instance distinguish a high blood sugar that was resulting from food intake (depending on the time of the meal and its glucose index there is continously more sugar being added to the blood) from another one that may be derived from a lack of insulin (maybe a shower that has taken longer than anticipated).

The data should be analysed with sufficiently large printouts. This allows e.g. for hand-written comments. To help the readability of the circles, one may decide to let the radios of the circles, not their area, reflect the sugar level (see Figure 3.

An alternative to the representation as circles is that in the form of *thermometers*. With only single values to be displayed, those will appear as solid boxes (Figure 4). A later plot will use thermometers to display insulin levels together with the glucose intake.

Basal rate

The basal rate is presented in a separate plot. If the data presented is shown on an hourly or half-hourly basis, and equal for all profiles of basal rate, then the data will be shown in a histogram-like manner. Otherwise, the data will be shown in the form of overlapping boxes, which are more difficult to interpret but a better idea was still lacking. Internally, graphs for sugar levels and basal rates are completely independent. Plotting the basal rate without sugar levels is not supported, though. Figure 5 shows the basal rate underneath the plot already shown alone in figure 2. The insulin curve was emperically determined. By comparing more such curves, it seems likely to find a mathematical description with fewer parameters than the hourly rates. An important parameter that is missing for the interpretation of the insulin rates and their effect is the onset of sleep.

The plot of figure 5 explictly requested the `labels.language="english"` and the `labels.type="adults"`. This concerns the description of the Y axis, invisible with the current settings of the margins is the description of the X axis. Figure 6 instead set `labels.language="german"` and `labels.type="kids"`. One can easily observe the shorter wording and the strict use of capitals only.

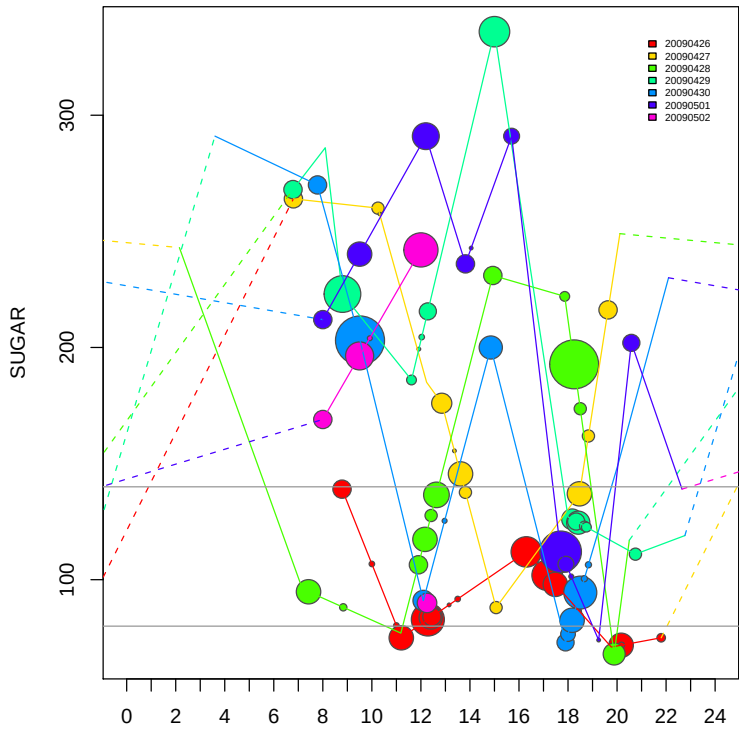

Figure 3: Blood glucose concentration - levels represented by area of circles, not by their diameter

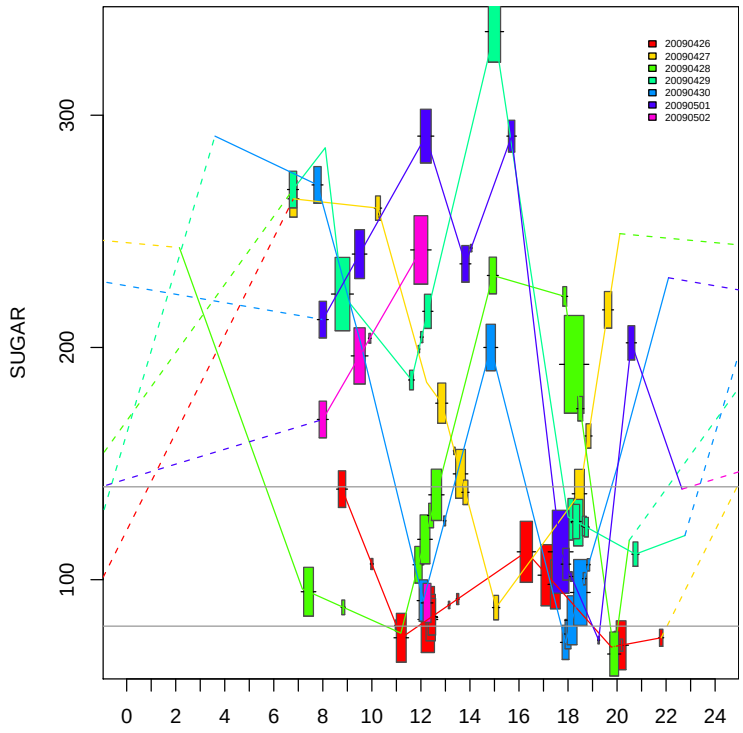

Figure 4: Blood glucose concentration shown with thermometers instead of circles which represent carbohydrate uptake by their size.

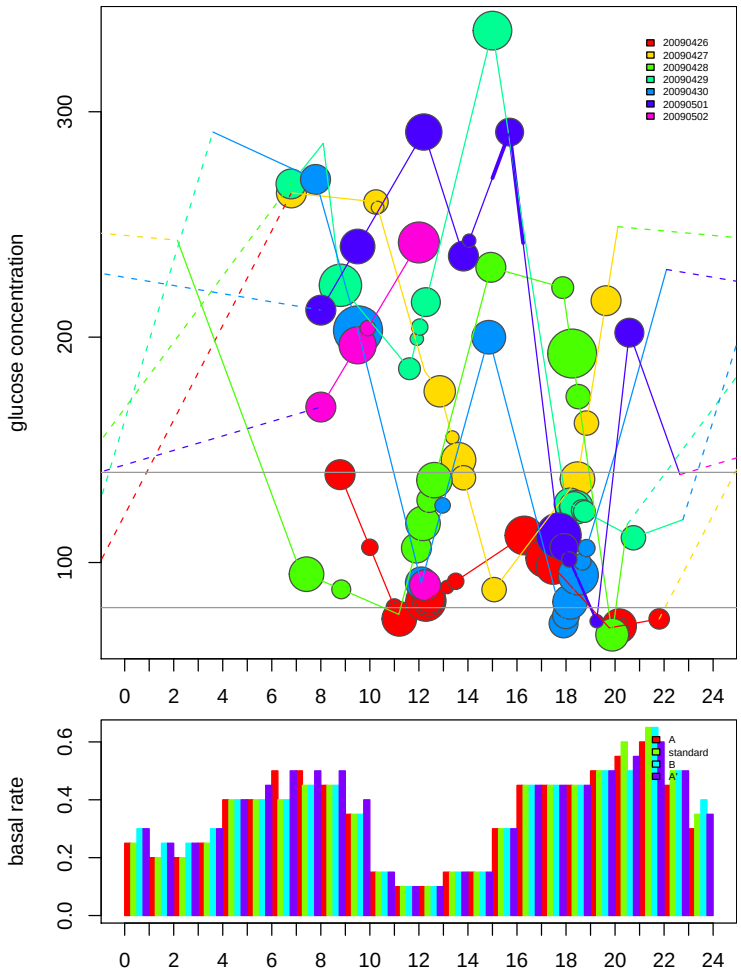

Figure 5: Sugar and food intake on top, basal rates at the bottom

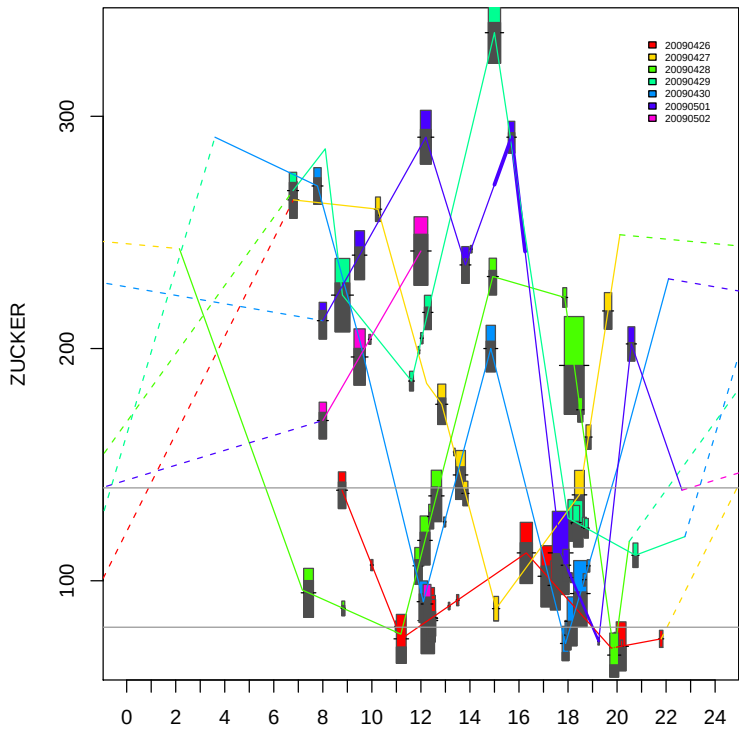

Figure 6: Carbohydrate uptake is represented by the height of the thermometers, the insulin dosage is represented by the "temperature".

Insulin dosage

The amount of insulin given in reaction (or in preparation) of carbohydrate intake is *not* shown in the previous figures. It was omitted with the hindsight that there are established rules to which one obeys and hence it is sufficient to know just how much was eaten. Also, the amount of compensation for high sugar concentrations is already determined as soon as one observes an event of hyperglucemia. Consequently, one does not ultimatively need to explicitly represent this graphically.

On the other hand, there needs to be somebody to first describe these rules. And it would be beneficial to have a display for their evalution. Another point of view is the request for an optical impression on the degree of correctness with which these instructions were followed.

Figure 6 shows the plot of Figure 5 with thermometers, which different from Figure 4 now displays in black the "height" of the insulin level. At 50% of the thermometer hight, the insulin dosage directly obeys the rule. The factors are passed via the `data.factors` argument.

4 Final words

A new program also for the juvenile

For younger patients it may be disturbing to see many adults stirring at some data that they don't undertand at all. And while still in the hospital they might not necessarily escape such conversations easily. Consequently, some little effort was made to support not only multiple languages but also easier wordings for the little ones. There should think of some extra image placed somewhere, eventually. Kids still won't understand the graph, most likely, but they might then read a word or two and be proud of understanding something. At the same time, the author has some confidence that the graph presented may contribute to the education in how to treat the disease at any age.

Legal issues

The source code of the diagram is made publicly available under the terms of the Affero GPL-3. This shall allow everyone to improve on the code. But nobody is allowed to hide those improvements when the fruits of the extra developments appear publicly.

The Affero GPL-3 license is already very explicit about it: in no way shall the authors of this program be reliable for anything. The user of this tool is solely responsible for every decision that he or she makes.

Upcoming development

To achieve the following is on my todo list as time permits:

- further annotation of the data
  - presentation of extranous events, e.g. the change of a catheter
  - allowing longer days than 24 hours, trigger new "days" by these events
- ties between basal rate and the sugar levels
  - indication of temporal changes
  - indication of profile changes
- easier input of data
- data reduction
- suggestions for amending basal rates

Contributing with patches

The code for the plots is just above 500 lines long and not overly complicated. If you have some background in programming paired with ideas, you might want to offer your improvements to appear in this package. This would also support the transfer of knowledge between the users of this package and is much appreciated.

The source code of this project is maintained publicly on the pkg-escience subversion repository[2] It can be checked out easily and anonymously with `svn co svn://svn.debian.org/svn/pkg-escience/r-cran-sugaR` and patches are produced with `svn diff` .

Contributing with money

This program is made available freely, under the assumtion that many users of this software are happily adopting some ideas to help their own or someone else's treatment. This is a good thing and every such feedback is strongly motivating. However, the author was reminded from several sides, the he could very much make use of some money, too. Hence, if you think that this work has helped to adjust the treatment, then please forward some money to the paypal account of steffen_moeller@gmx.de[3].

If you are a professional in health care, research or industry and think that this work has helped with the treatment of your patients, then please contact me for an official invoice for an ammount that you are suggesting, if this is needed or wanted prior to sending money.

[2] `http://svn.debian.org/wsvn/pkg-escience/r-cran-sugaR/#_r-cran-sugaR_`

[3] `https://www.paypal.com/cgi-bin/webscr?cmd=_donations&business=steffen_moeller%40gmx%2ede&item_name=SugaR&currency_code=EUR&no_shipping=1`