Data Coding and Integrity

by Roger Mundry

Leibniz ScienceCampus Primate Cognition, Göttingen

last modified: February 28, 2025

baseline

a typical empirical study usually consists of about the following steps:

- idea or questions
- hypotheses and predictions
- planning of experiment/study
- data collection
- data analysis
- writing the paper

but in between these is an additional step (actually two) whose importance is quite often overlooked:

data coding and correctness checks

data coding

data coding refers to everything related to how to enter data into one or several files

it involves questions like...

- should the date be in one or several files and/or data sheets?
- how to structure data sheets?
- how to code categorical data?
- how to code dates and times?
- how to name variables?
- ...

all these (and many others) are important questions which have relevance for how easy it'll be to analyze the data

in the best case, one can analyze the data straight away

in the worst case it'll need hours, days, weeks, or even way more time to shape the data such that one can begin with the analysis

unfortunately, data are often structured such that it needs considerable effort before one can begin with the analysis

data integrity

data integrity refers to everything related to the correctness and consistency of the data

it refers to questions like...

- do things with the same label *always* label the same thing?
- do things with different labels *always* label different things?
- and are these also true across files?
- are dates entered in a consistent format?
- does each individual have *only* one sex and/or does it *always* belong the same species
- ...

unfortunately, the answers to such questions are often 'no'

and in such a case, one might spend a lot of time with data cleaning before one can begin with any analysis

this lesson

with this lesson I want to give a brief introduction about how to structure data and how to check for their integrity

both are really important parts of each project

and depending on how well structured and clean the data are, one either can begin enjoying the analysis straight away, or one first has to overcome a potentially huge hurdle and do lots of boring work before the interesting stuff can begin

I'll begin with data coding and then deal with data integrity

data coding: structure

first, *data structure* refers to the question about into how many files and/or data sheets data should be entered?

for instance, when one has data for different individuals, one might consider having one file or sheet for each individual

or one might consider putting the data of all individuals in the same file and sheet

which is better

when looking for an answer, it helps a lot to think about it from the endpoint:

what will be needed for the eventual analysis?

and the answer to this question is simple:

all data must be in a *single* rectangular data table

that means, all data should be entered into a single sheet!

data structure

but there is more to be said about this file/data structure:

- its columns are for *different* variables
- all values of the same kind should be in the same column
- all relevant information should be coded in a column
- its rows are for different observations or data points
- all information relevant in a certain row, *must* be present in it

in addition, column names should be in and only in the *first* row

all this might be a little abstract

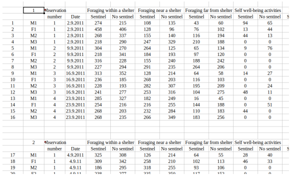
so here is an example which 'beautifully' illustrates all kinds of deviations from this rule

data structure

	1	bservation		Foraging within a shelter		Foraging near a shelter		Foraging far from shelter		Self well-being activities		
		number	Date	Sentinel	No sentinel	Sentinel	No sentinel	Sentinel	No sentinel	Sentinel	No sentinel	S
1	M1	1	2.9.2011	274	215	108	135	43	60	94	65	
2	F1	1	2.9.2011	458	406	128	96	76	102	13	44	
3	M2	1	2.9.2011	268	337	155	140	116	194	44	13	
4	M3	1	2.9.2011	218	290	247	329	210	188	0	0	
5	M1	2	9.9.2011	304	270	264	125	65	134	9	76	
6	F1	2	9.9.2011	218	341	184	193	97	120	0	0	
7	M2	2	9.9.2011	316	228	155	240	188	242	0	0	
8	M3	2	9.9.2011	227	294	291	235	264	206	0	0	
9	M1	3	16.9.2011	313	352	128	214	64	58	14	27	
10	F1	3	16.9.2011	236	185	268	203	116	103	0	0	
11	M2	3	16.9.2011	228	193	282	307	195	209	0	24	
12	M3	3	16.9.2011	241	277	253	316	104	275	48	11	
13	M1	4	23.9.2011	285	327	182	249	16	45	0	0	
14	F1	4	23.9.2011	254	216	216	255	144	188	0	51	
15	M2	4	23.9.2011	268	203	232	284	110	183	44	0	
16	M3	4	23.9.2011	268	235	266	349	183	256	0	0	
	2	bservation		Foraging within a shelter		Foraging near a shelter		Foraging far from shelter		Self well-being activities		
		number	Date	Sentinel	No sentinel	Sentinel	No sentinel	Sentinel	No sentinel	Sentinel	No sentinel	S
17	M1	1	4.9.2011	325	308	126	214	64	55	28	40	
18	F1	1	4.9.11	309	342	258	210	102	113	46	33	
19	M2	1	4.9.11	186	295	318	255	93	106	0	0	

who can name the problems?

data structure



the name of the group and the individual are indicated in the same column

different individuals have the same name (e.g., M1)

group is not indicated in each row

sex and rank of the individuals don't have their respective own columns but are coded in the name of the individual

the observed durations are coded in many different columns

whether foraging took place within, near to, or far from a shelter is coded in different columns but should be coded as a variable

the same applies to whether foraging took place in the presence or absence of a sentinel

there are two rows with column names, and neither of them is in row 1

data structure

and here is how the data need to look like for an analysis (first ten rows):

	group	subject	sex	rank	date	sentinel	shelter	duration
1	g.1	g.1_M1	M	1	2.9.2011	present	within	274
2	g.1	g.1_F1	F	1	2.9.2011	present	within	458
3	g.1	g.1_M2	M	2	2.9.2011	present	within	268
4	g.1	g.1_M3	M	3	2.9.2011	present	within	218
5	g.1	g.1_M1	M	1	9.9.2011	present	within	304
6	g.1	g.1_F1	F	1	9.9.2011	present	within	218
7	g.1	g.1_M2	M	2	9.9.2011	present	within	316
8	g.1	g.1_M3	M	3	9.9.2011	present	within	227
9	g.1	g.1_M1	M	1	16.9.2011	present	within	313
10	g.1	g.1_F1	F	1	16.9.2011	present	within	236

notice the differences:

group ID, sentinel presence, and distance to shelter are all coded in each row

columns for the sex and rank were added

all durations are now in a single column

subject names are now the combination of group and subject, such that each subject has a unique ID

coding

apart from the structure of the data, also how things are coded in data sheets is of relevance

for instance, how dates and time or categorical variables are coded requires attention

in addition to coding considerations, I'll also to treat the naming of columns

coding: categorical variables

a *categorical* variable is one which isn't numeric

examples are...

- the names of individuals and the social groups or enclosures in which they live
- the species and sex of individuals
- highest parental education
- labels of experimental treatments (e.g., 'treatment' and 'control')
- and many others

basically, two options do exist, coding with their proper labels and coding them with numbers

which is better?

the answer is simple: code them with their labels

but why might some people have told you the opposite, and why is coding with proper labels better?

coding: categorical variables

so why might some people have told you to code categorical variables with numbers?

in the past, people often used statistical softwares like SPSS which can't handle categories coded with labels

and I'm quite convinced that this the reason why some people developed the habit of coding categorical variables with numbers

but today most people use R for their analyses, and R can perfectly handle categories coded with proper labels

coding: categorical variables

and why is coding categorical variables with labels advantageous?

there are many reasons for that:

- when one uses numbers, one doesn't need to store elsewhere the information about which number means what
- numbers cannot mistakenly taken as numeric in an analysis or when it comes to plotting
- numbers are harder to process and remember
- its totally easy to turn proper labels into numbers when needed but potentially way more complicated the other way round
- ...

coding: categorical variables

but maybe just having a look at the same data, once coded with proper labels and once with numbers convinces you that proper labels make a data set much more digestible:

	species	subject	nr.seen	tot.dur
1	giraffe	Patricia	3	3.10
2	wilde beast	Joshua	1	4.50
3	wilde beast	James	2	2.27
4	impala	Kimberly	2	5.02
5	wilde beast	Daniel	2	6.35
6	giraffe	Kevin	2	2.09

	species	subject	nr.seen	tot.dur
1	2	38	3	3.10
2	4	23	1	4.50
3	4	18	2	2.27
4	3	27	2	5.02
5	4	11	2	6.35
6	2	26	2	2.09

coding of dates

a particular issue is the coding of dates (e.g., date of testing or an observation, birthdays, etc.)

one of the issues is that there too many conventions in use wrt...

- the order of date, month, and year (e.g., DMY, YMD, YDM, etc.), with how many digits years are coded (e.g., 98, 1998)
- how months are coded (e.g., February, Feb, FEB, 2)
- whether and how they are separated (e.g., 28.02.2025, 28/02/2025, 28-02-2025, 28022025)
- etc. etc.

importantly, not only different people might use different conventions

rather, even a single person might be inconsistent in the coding of dates[1]

[1] I have seen that many times

coding of dates

things can get particularly messy when multiple people provide data for the same data set

in such cases it can happen that one needs to throw away about a third of the data because for dates like 2015/05/11 it can be impossible to know in hindsight whether May 11th or November 5th was meant

another issue is that programs like Excel or LibreOffice Calc try to 'help' us which frequently creates just a mess

or had none of you ever experienced that you entered something like 1.5 in an Excel table and then it somehow turned into 1/May/1903?

coding of dates

my conclusion from all that mess was that the only safe way to code dates is to split them up into three columns, one for each of day, month, and year

that resolves a lot of the issues[2]

and it definitely prevents softwares trying to 'help' us[3]

[2] but not all, e.g., coding months by names or numbers or coding years with two or four digits

[3] something I definitely hate

coding of times

coding of times is usually done using the format hh:mm:ss

and *usually* that works quite well

however, when working with spreadsheet programs (e.g., Excel) it can still happen that for some of them a date is added (e.g., 1903-01-01)

I had also experienced cases in which a program automatically added a time zone and 'normal' and daylight saving time (e.g., CET and CEST)

as such things can again cause trouble later on, the safest way to enter times is to again spread them over three column, one for each of hours, minutes and seconds

coding of decimals

unfortunately, even how decimals are coded (e.g., 3,14 or 3.14) requires attention

this is because both formats (comma and point as decimal separator) are in use and science is a pretty international and collaborative endeavor...

it probably happened already to some of you that you entered something like 3.14 into a cell in an Excel table and Excel turned into something like May 14 (maybe adding 1903)[4]

if you enter the data yourself you will notice that and swear and then try to fix it somehow

but if you get a larger spreadsheet file from a collaborator, then such things might happen in rows somewhere in the data where you do not see it

in such a case the data might get in essence destroyed...

[4] this is because the standard settings of a German Windows OS accept the period as a date separator

coding of decimals

another, related issue is the 'digit grouping symbol' for which the standard settings of a German Windows OS again use the period

as a consequence of that, 3.14 might turn into 3.140 and 3.14159265358979 might turn into 314.159.265.358.979...

some 30+ years ago, I myself had destroyed a data set[5] by not being aware of the issue

since then I have the strict habit of always ensuring to work in the international format (period as the decimal separator) and to make sure that the period isn't set to be accepted as anything else (e.g., date separator or digit grouping symbol)

later on, I added a second rule:

never ever do anything in a spreadsheet file and then save it[6]

[5] which had taken me lot of time to enter the data into it

[6] actually, I probably use the save button in a spreadsheet software no more than once a year, and if I do, its actually the save-as button...

column names

to ensure compatibility with the data analysis/handling softwares I'm aware of apply these three rules[7]

- begin column names with a character
- let them consist of only characters, numbers, periods('.'), and the underscore ('_')
- *never* include any special characters like brackets, hyphens ('-'), spaces, Greek characters like α or β, and the like

plus, its usually a good idea to have column names which are intelligible/ meaningful but not too long at the same time

and don't forget, each column has its own and complete name and that column names need to be present in and only in the *first* row of the table

[7] I'm aware that what follows are only two rules, but many people seem to need the third explicitly

column names

in a spreadsheet program like Excel, one can use as column names whatever one wants

but the column names might then not be compatible with what other softwares accept for column names

the usual consequence of that is that the other software will modify the column names somehow

in such a case it might be hard to handle and keep track of things

aggregation?

quite frequently people approach me for help with the analysis of some data which comprise something like proportions of rates

for instance, they might want to analyze how much the proportion of correct answers differs between two experimental conditions

of they want to analyze how many of a certain kind of behaviour occurred per minute of observation time (or experiment duration)

however, although such proportions or rates are usually easy to analyse (or model), this is usually impossible when they come in the form of proportions or rates

instead, 'proportions' should rather be coded with two columns indicating the number of correct responses and the number of trials

and 'rates' should rather be coded with two columns indicating the number of events and the observation duration

I can't go into details here, just keep in mind that entering aggregated data is usually a bad idea as it might make the intended analysis impossible[8]

[8] and also keep in mind that aggregation is usually easy whereas 'disaggregation' is usually impossible

using formats (e.g., color) and/or comments?

sometimes I see data which come in the form of an excel file in which essential information is coded using colors or formats[9]

doing so is *always* an extremely bad idea

this is because the colors will only be present in Excel, but when you load the data into, for instance, R, they are lost

but even when working *with* Excel, they usually do not help much

so rather than using colors, add columns in which you code the respective information

the same applies to comments:

simply *never* use them, since the respective information is hard to extract within Excel and lost once one uses any other software

again, feel free to add columns comprising what otherwise would be in the comments

[9] e.g., 'before the anaysis we need to exclude the rows highlighted in red'

coding: wrap-up

this section certainly just scratched the surface

in fact, there are many other issues which might gain relevance

for instance, sometimes the data come in the form of many files (e.g., eye-tracking or pupil dilation data which might comprise one file per individual and trial)

in such a case, the naming of the file will be of crucial importance...

I can't go into more detail here

but hopefully this provided a reasonable starter

importantly, when coding your own data, always think from the end point

that is, ask yourself, what do I eventually need for the intended analysis, and which ways of coding will allow for the analysis?

data integrity

unfortunately, once the data are in the format needed for the analyses, one usually cannot begin with it

this is because at this stage there might still be bugs in the data

in fact, probably up to several times a week I'm asking someone a question like...

- are the data clean and ready for the analysis?
- is it such that all the partners an individual interacted with are also present in the individuals it *could* have interacted with?
- is it true that the sum of the times an individual performed different behaviors is equal to the total observation time?
- and many other such questions

and guess what peoples' answers usually are

- "(I think) it should"
- "I'm almost sure"
- and such things

and that essentially means "no"...

data integrity

so in all such cases we need to check the data

however, even in the few cases when people say 'yes' in response questions like the above, it might still be a good idea to check

for instance, just a few days ago, someone provided me a new version of a data set (we had already worked with) which appeared to comprise some bugs

the new version of the data was supposed to be fixed wrt these issues

but it turned out some were still in there...

so its actually a *very* good idea to very carefully scrutinize the data for correctness, integrity, and the like

this is even more required if the intended analysis is complex and potentially computation intensive and time consuming

in fact, it can be quite a disaster if one, after a two months analysis (and a lot of consumed electricity), realizes that the data used comprise errors, or?

data integrity

so what we are going to do now is to inspect a data table for internal consistency

importantly, what we are going to do, does by far not cover all the checks that might be required for a solid check

obviously, the data at hand vary considerable from project to project

so one has to carefully think what will be required to ensure clean data for any particular data set

I'm coming back to that

getting started with the data

let's read the data into a data frame in R:

```
setwd("<...>")
xdata=read.csv(file="reconcilation_data_v2.csv",
  header=T, stringsAsFactors=T, sep=",")
```

in the above code, <...> needs to be replaced by the path to the folder in which the data provided are

note also the argument `stringsAsFactors` set to `T`

this has the effect that all columns with a single entry which isn't a number (or blank or an `NA`) is set to be of mode factor[10]

[10] if you call `read.csv` with its default (`stringsAsFactors=F`) then such columns will be of mode 'character'

the idea of the data

the data are about 'reconciliations'

reconciliations can happen after a conflict between two individuals

it is affiliative behaviour of one of the two individuals involved in a conflict towards the other, shortly after the conflict took place

the data were collected during focal sampling[11] of females

[11] the behavior and the social interactions of a single individual is recorded for a period of time

the data

the relevant columns in the file are

Subject: the ID of the observed focal female

Partner: the ID of the individual with which the focal had a conflict

Reconci: whether a reconciliation happened (1) or not (0)

Affilia: the strength of the reconciliation (weak, average, or strong)

Support: whether one of the individuals involved in the conflict got support from a third party individual

ConfInt: intensity of the conflict (physical or non-physical)

ConfDur: duration of the conflict (in seconds)

Context: context of the conflict (food, sex, or other)

RankDif: rank difference between the two individuals involved in the conflict (none (n), small (s), medium (m), or large (s))

Latency: latency between conflict and reconciliation (if any)

the data

PartSex: sex of the individual the focal had a conflict with

RconDur: duration of the reconciliation (if any)

IniCnfl: ID if the initiator of the conflict (subject or partner)

IniReco: ID if the initiator of the reconciliation (subject or partner)

but let's begin to explore the data

what we do with this exercise is to mimic a situation in which I am very regularly: working with data others provided

I could imagine, that the same could happen to you, too

in fact, today quite a fraction of students and scholars works with data provided by others

getting an overview

the first thing I always do after I had read data into data frame is to get an overview about what I have

and I'm ± always using the function `str` ('structure') for that purpose

let's have a look at what it reveals:

```
str(xdata)
```

```
'data.frame': 914 obs. of  17 variables:
 $ CaseNr : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Subject: Factor w/ 8 levels "Ash","Bet","Deb",..: 1 1 1 2 8 7 7 1 6 6 ...
 $ Partner: Factor w/ 47 levels "Ama","And","Ant",..: 9 39 9 11 7 7 13 7 3 46 ...
 $ Reconci: int  0 0 0 0 0 1 0 0 0 0 ...
 $ Affilia: Factor w/ 9 levels "average","Average",..: 2 2 2 2 2 5 2 8 2 8 ...
 $ Support: Factor w/ 4 levels "n","no","y","yes": 2 2 2 2 2 2 2 2 2 4 ...
 $ ConfInt: Factor w/ 2 levels "NonPhys","Physical": 1 1 1 1 1 1 1 1 1 1 ...
 $ ConfDur: Factor w/ 102 levels "1","10","1003",..: 1 1 1 1 23 66 92 28 39 75 ...
 $ Context: Factor w/ 3 levels "Food","Other",..: 2 2 2 2 1 2 1 2 3 1 ...
 $ RankDif: Factor w/ 4 levels "l","m","n","s": 4 1 4 1 4 4 1 2 1 1 ...
 $ Latency: int  NA NA NA NA NA 3 NA NA NA NA ...
 $ PartSex: Factor w/ 4 levels "F","Female","M",..: 2 3 2 4 2 2 4 2 4 2 ...
 $ RconDur: Factor w/ 122 levels "1,0","10,0","102,0",..: NA NA NA NA NA 99 NA NA NA NA ...
 $ RenewAg: int  1 0 0 0 0 0 0 0 1 0 ...
 $ IniCnfl: Factor w/ 3 levels "beide","Partner",..: 3 3 3 3 3 2 2 2 3 3 ...
 $ IniReco: Factor w/ 4 levels "beide","None",..: 2 2 2 2 2 4 2 2 2 2 ...
 $ TimToRA: int  264 NA NA NA NA NA NA NA 526 NA ...
```


getting an overview

so what can we learn about the data from what `str(xdata)` reveals?

what should one maybe pay attention to?

the first thing I always pay attention to here is the number of rows and check whether it is correct

I do so because under certain circumstances it can happen that not all the rows are read into the resulting object

obviously, such a check requires knowledge about the data

and such knowlewdge is actually required throughout what we're going to do now (as you gonna see)

`str(xdata)` also reveals the mode of each column (here `factor` when it is not numeric and `integer` when it is)

and when it is factor it also tells the number of levels and names a few of them

so what does that reveal?

getting an overview

for Subject and Partner we see that they have 8 and 47 levels, respectively

and in any real setting we can compare these numbers with what we expect (are they maybe too large?)

the next potentially interesting column is *Affilia*

this is a factor with 9 levels

but why 9 levels, shouldn't it be three (weak, average, or strong)?

getting an overview

so let's have a look:

```
levels(xdata$Affilia)
```

```
[1] "average"  "Average"  "Average " "strong"   "Strong"
[6] "Strong "  "weak"     "Weak"     "Weak "
```

what we see here is something which frequently occurs in manually entered data:

spelling is inconsistent wrt spelling details (first character upper or lower case)

and some of the entries are followed space

btw, Excel reveals the latter but not the former (at least in Pivot Tables when I still used it)

with the column *Support* we obviously have a very similar problem

```
levels(xdata$Support)
```

```
[1] "n"   "no"  "y"   "yes"
```


getting an overview

and with the column *PartSex* we also have a similar problem

```
levels(xdata$PartSex)
```

```
[1] "F"      "Female" "M"      "Male"
```

and this time we're going to fix it

what we could do to fix it, is to reduce the spelled out sexes to their first character

however, we can't do this as long as the respective variable is a factor

so we first need to use the function `as.character` and then can use `substr` to extract the first character:

```
xdata$PartSex=as.character(xdata$PartSex)
xdata$PartSex=substr(x=xdata$PartSex, start=1, stop=1)
unique(xdata$PartSex)
```

```
[1] "F" "M" NA
```


getting an overview

another problem seems to be in the column *ConfDur*[12] which is a factor with many levels but should be numeric

so lets explore the issue

```
levels(xdata$ConfDur)
```

```
  [1] "1"    "10"   "1003" "1004" "102"  "11"   "111"  "115"  "12"   "122"
 [11] "125"  "129"  "13"   "14"   "146"  "15"   "156"  "16"   "17"   "178"
 [21] "18"   "181"  "19"   "196"  "2"    "20"   "205"  "21"   "210"  "22"
 [31] "23"   "231"  "232"  "24"   "25"   "255"  "26"   "27"   "28"   "29"
 [41] "290"  "297"  "3"    "30"   "300"  "31"   "32"   "325"  "33"   "33?"
 [51] "34"   "35"   "36"   "362"  "379"  "39"   "4"    "40"   "41"   "410"
 [61] "43"   "45"   "48"   "49"   "490"  "5"    "50"   "51"   "53"   "55"
 [71] "56"   "57"   "570"  "58"   "6"    "60"   "604"  "61"   "62"   "63"
 [81] "64"   "65"   "67"   "68"   "69"   "7"    "70"   "71"   "72"   "74"
 [91] "78"   "8"    "83"   "86"   "87"   "89"   "9"    "91"   "94"   "95"
[101] "96"   "99"
```

well, there is a 'duration' being '33?' which is obviously not a number

the probably safest way to fix it might be to turn it into an `NA`

but I wouldn't do that before having checked with the person who provided the data

[12] conflict duration

getting an overview

nevertheless, let's replace the '33?' by an `NA` and then turn the variable into numeric as we later need the variable again:

```
xdata$ConfDur[xdata$ConfDur=="33?"]=NA
xdata$ConfDur=as.character(xdata$ConfDur)
xdata$ConfDur=as.numeric(xdata$ConfDur)
```

getting an overview

obviously, there is a somewhat similar problem with *RconDur*[13]
so lets have a look at that one too:

```
levels(xdata$RconDur)
```

```
  [1] "1,0"    "10,0"   "102,0"  "1054,0" "108,0"  "11,0"   "12,0"   "13,0"
  [9] "130,0"  "132,0"  "135,0"  "136,0"  "14,0"   "141,0"  "143,0"  "145,0"
 [17] "146,0"  "15,0"   "152,0"  "153,0"  "156,0"  "158,0"  "16,0"   "163,0"
 [25] "1646,0" "166,0"  "17,0"   "171,0"  "172,0"  "173,0"  "18,0"   "182,0"
 [33] "184,0"  "187,0"  "19,0"   "192,0"  "194,0"  "196,0"  "197,0"  "2,0"
 [41] "20,0"   "2028,0" "209,0"  "21,0"   "22,0"   "221,0"  "223,0"  "224,0"
 [49] "23,0"   "234,0"  "24,0"   "244,0"  "246,0"  "249,0"  "25,0"   "26,0"
 [57] "265,0"  "266,0"  "27,0"   "278,0"  "28,0"   "286,0"  "298,0"  "3,0"
 [65] "30,0"   "309,0"  "31,0"   "319,0"  "32,0"   "320,0"  "321,0"  "33,0"
 [73] "335,0"  "34,0"   "343,0"  "345,0"  "35,0"   "36,0"   "37,0"   "39,0"
 [81] "393,0"  "397,0"  "4,0"    "405,0"  "408,0"  "41,0"   "410,0"  "426,0"
 [89] "432,0"  "44,0"   "441,0"  "45,0"   "47,0"   "48,0"   "5,0"    "50,0"
 [97] "501,0"  "51,0"   "53,0"   "531,0"  "54,0"   "541,0"  "559,0"  "56,0"
[105] "58,0"   "59,0"   "6,0"    "61,0"   "62,0"   "63,0"   "65,0"   "66,0"
[113] "663,0"  "69,0"   "7,0"    "8,0"    "869,0"  "88,0"   "9,0"    "90,0"
[121] "97,0"   "98,0"
```

it seems, these numbers had been entered on a computer with the comma specified as the decimal separator

[13] reconciation duration

getting an overview

let's fix this (again, we need to first turn the column into character):

```
xdata$RconDur=as.character(xdata$RconDur)
xdata$RconDur=gsub(x=xdata$RconDur, pattern=",",
  replacement=".", fixed=T)
xdata$RconDur=as.numeric(xdata$RconDur)
```

finally, let's check what the levels of the factors *IniCnfl* and *IniReco* are:

```
levels(xdata$IniCnfl)
```

```
[1] "beide"   "Partner" "Subject"
```

```
levels(xdata$IniReco)
```

```
[1] "beide"   "None"    "Partner" "Subject"
```

both look okay

getting an overview

let's look at `str(xdata)` again to see what it looks like now:

```
str(xdata)
```

```
'data.frame': 914 obs. of  17 variables:
 $ CaseNr : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Subject: Factor w/ 8 levels "Ash","Bet","Deb",..: 1 1 1 2 8 7 7 1 6 6 ...
 $ Partner: Factor w/ 47 levels "Ama","And","Ant",..: 9 39 9 11 7 7 13 7 3 46 ...
 $ Reconci: int  0 0 0 0 0 1 0 0 0 0 ...
 $ Affilia: Factor w/ 9 levels "average","Average",..: 2 2 2 2 2 5 2 8 2 8 ...
 $ Support: Factor w/ 4 levels "n","no","y","yes": 2 2 2 2 2 2 2 2 2 4 ...
 $ ConfInt: Factor w/ 2 levels "NonPhys","Physical": 1 1 1 1 1 1 1 1 1 1 ...
 $ ConfDur: num  1 1 1 1 19 5 8 21 28 6 ...
 $ Context: Factor w/ 3 levels "Food","Other",..: 2 2 2 2 1 2 1 2 3 1 ...
 $ RankDif: Factor w/ 4 levels "l","m","n","s": 4 1 4 1 4 4 1 2 1 1 ...
 $ Latency: int  NA NA NA NA NA 3 NA NA NA NA ...
 $ PartSex: chr  "F" "M" "F" "M" ...
 $ RconDur: num  NA NA NA NA NA 53 NA NA NA NA ...
 $ RenewAg: int  1 0 0 0 0 0 0 0 1 0 ...
 $ IniCnfl: Factor w/ 3 levels "beide","Partner",..: 3 3 3 3 3 2 2 2 3 3 ...
 $ IniReco: Factor w/ 4 levels "beide","None",..: 2 2 2 2 2 4 2 2 2 2 ...
 $ TimToRA: int  264 NA NA NA NA NA NA NA 526 NA ...
```

if we'd want to work with these data, we'd need to fix the columns *Affilia* and *PartSex*

but as here I want to put focus on data checks, we leave them as they are

getting an overview

by now we had already spotted quite a few bugs in the data

assuming we had fixed the columns Affilia and PartSex, could we now begin with the analyses?

what do you think?

well, we had only detected (and partly removed) the immediately and most obvious issues

but there could be many others

what would be things which one should pay attention to?

integrity checks

I'd begin with what could be referred to as *integrity checks*

in case of the data here, such integrity checks involve checks of the entries in two or more columns

for instance,

- individuals cannot cannot aggress themselves
- each individual has to always belong to the same sex
- when there is a reconciliation duration, there should also be a reconciliation
- for any pair of subject and partner, the rank difference probably should always be the same
- ...

perhaos you think that it would be ridiculous to conduct such checks

but my experience definitely is that no bug is too stupid or ridiculous to not possibly happen

integrity checks

for instance, I'd already seen individuals tested before they were born, elephant signs spotted in the middle of the Atlantic ocean (by people who where in Africa), and so on

there is really nothing one should take for granted

it is really such that many people coming to me with 'clean' data and asking for help with their analysis, and leave my office a few minutes later because I spotted mistakes

btw, although observational data and data collected in the wild are more prone to errors

but there can be also many in experimental data

so let's conduct a few such integrity checks

integrity checks

for instance, are there any aggressive interactions an individual had with itself?[14]

```
any(as.character(xdata$Subject)==as.character(xdata$Partner))
```

```
[1] TRUE
```

so this happens...

let's heck how many times this happens in the data

```
sum(as.character(xdata$Subject)==as.character(xdata$Partner))
```

```
[1] 10
```

let's exclude them for now[15]

```
xddata=subset(xdata,
  as.character(xdata$Subject)!=as.character(xdata$Partner))
```

[14] note that both columns need to be converted to character before the can be compared

[15] in any real setting I'd ask the person who provided the data to solve the issue

integrity checks

next we could check whether there are any partners with more than one sex

I usually do that by first cross-tabulating partner sex and partner ID:

```
xx=table(xdata$Partner, xdata$PartSex)
head(xx)
```

```
     F  M
Ama  0 57
And 39  1
Ant  0 23
Ash  0  1
Bar  0  4
Bet  1  0
```

per row of the resulting table, there should be only one column with a value >0

so I'd check each entry in the table for whether it is >0 (xx>0) and then count the number of the respective cells per row of the table

```
table(apply(X=xx>0, MARGIN=1, FUN=sum))
```

```
 1  2
46  1
```

=> there is indeed one partner that is present with two sexes in the data

integrity checks

to figure out which partner that is, you can do the following:

```
xx.tab=apply(X=xx>0, MARGIN=1, FUN=sum)
xx[xx.tab==2, ]
```

```
       F  M
  And 39  1
```

the individual named 'And' is usually labelled to be a female but once as a male

also regarding this issue one would have to ask the person who provided the data about how to handle it

integrity checks

let's also check whether each reconciliation has a duration and whether each reconciliation duration is associated with a reconciliation in the column headed *Reconci*

both can be achieved in one go:

```
table(xdata$Reconci, !is.na(xdata$RconDur))
```

```
    FALSE TRUE
  0   620    1
  1     0  293
```

there is one case in the data in which a reconciliation is indicated, but the corresponding entry in the column headed *Reconci* is 0 (i.e., it indicates that no reconciliation happened)...

integrity checks

one last thing we do explore here today is whether the rank difference of any given pair of subject and partner is constant in the data

assuming that the rank difference for two individuals is the same, irrespective of which of them is the subject and which the partner, this is going to be a little more complicated

this is because we first need a new vector with each entry comprising the names of the subject and the partner in alphabetical order

there are many ways of achieving that, here is one

first, I extract the two relevant columns into an object:

```
xx=xdata[, c("Subject", "Partner")]
```


integrity checks

then we can call an `apply`, this time with a self defined function

this self defined function first sorts the individuals by their names (function `sort`) and then glues them together in a single word (function `paste`):

```
dyad=apply(xx, 1, function(x){
  paste(sort(x), collapse="_")
})
head(xx)
```

```
[1] "Ash_Cha"  "Ash_Ron"  "Ash_Cha"  "Bet_Dan"  "Bri_Mary" "Bri_Marg"
```

now we can cross-tabulate the rank difference against the dyad and then do the same as we did earlier for partner and partner sex:

```
xx=table(dyad, xdata$RankDif)
table(apply(X=xx>0, MARGIN=1, FUN=sum))
```

```
  1   2
237   4
```

=> there are 4 dyads in which the rank difference varies

plausibility checks

the last kind of checks I want to briefly consider are *plausibility checks*

they usually don't reveal clear cases of errors

but they might reveal suspicious cases

for instance, if the experimental trials usually lasted 10 to 15 seconds, one that lasted 130 seconds seems pretty suspicious

or if a group of individuals was observed and the individuals usually occur about 20 to 30 times in the data, individuals which occur way more or way less frequently seem also pretty suspicious

let's look at `str(xdata)` again and think about what kind of plausibility checks seem reasonable

plausibility checks

the probably first thing that would jump into my eye is the relatively large number of unique partners

here I'd definitely inspect the sorted names for subtle and likely erroneous variations (like we had in the column headed *Affilia*):

```
sort(levels(xdata$Partner))
```

```
 [1] "Ama"  "And"  "Ant"  "Ash"  "Bar"  "Bet"  "Bri"  "Car"  "Cha"  "Chr"
[11] "Dan"  "Dav"  "DonF" "DonM" "Emi"  "Jen"  "Jes"  "Joh"  "Jos"  "Josh"
[21] "Kar"  "Ken"  "Kev"  "Kim"  "Lin"  "Lis"  "Marg" "Mark" "Mary" "Mat"
[31] "Mel"  "Mic"  "Mit"  "Nan"  "Pat"  "Pau"  "Ric"  "Rob"  "Ron"  "San"
[41] "Sar"  "Step" "Stev" "Sus"  "Tho"  "Tim"  "Wil"
```

here everything seems to be fine:

not variations with regard to lower/uppercase (e.g., "And" or "and"), no trailing spaces (e.g.. "And "), no obvious typos[16]

[16] altogether, in the past 10 to 15 years I had probably spend many days with nothing other than this: looking at long, sorted lists of words, searching them for unwanted subtle variations

plausibility checks

we could, for instance, also check how common each of the subjects and partners is present in the data:

```
table(xdata$Subject)
```

```
 Ash  Bet  Deb  Eli  Emi  Lis Marg Mary
 108   73   99   62  111   87  186  188
```

to me, that looks quite plausible

and the same for the partner (let's sort the output as there are quite a few unique partners):

```
sort(table(xdata$Partner))
```

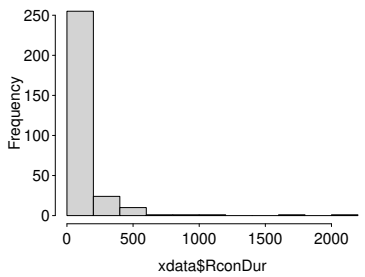
```
 Ash  Bet  Lis Marg Mary  Emi  Bar  Jos  Mit  Tho  Jes DonM  Ric  Ken  Rob Stev  Jen  Kim
   1    1    2    2    2    3    4    5    5    5    8    9    9   10   10   10   11   13
 Sar  Pat  Sus  Ron  San Josh Mark Step  Chr  Kar  Cha  Dan  Ant  Nan  Bri DonF  Kev  Pau
  13   15   15   17   17   18   18   18   20   21   22   23   24   24   25   25   26   26
 Mat  Mel  Lin  Dav  Joh  Mic  And  Tim  Car  Wil  Ama
  27   28   35   37   37   38   40   45   46   47   57
```

to me, that looks quite reasonable, too (but I'd definitely ask whether the rarer individuals are okay)

plausibility checks

or we could have a look at the durations of the reconciliations:

```
hist(xdata$RconDur)
```



this might or might not be okay

but without good knowledge of the data/what happened during the observations, we cannot know

plausibility checks

as we don't have unlimited time, I end here with the plausibility and integrity checks

however, there are definitely other also important ones

for instance, I'd also check the distribution of the latencies to reconciliation

and I'd also check whether the entries in the column Latency are consistent with those in the column Reconci

wrap-up

with this lesson I wanted to raise awareness about the importance of careful checks of the data before one begins with any analysis

such checks can be very time consuming and the process of checking and cleaning the data can be truly painful

however, they are always needed as at one or another stage of data collection, entry, processing, etc., humans will be involved, and humans inevitably make mistakes

but please don't think, that data gathered by machines and programs are by any means better

wrap-up

for instance, I still vividly remember a study for which several researchers followed primates with a GPS to record their daily tracks

all of them must have had discovered at some point during the months long observation period that they can set the display of their GPS to their personal language preferences

as a consequence, the final data set comprised a dozen of different date formats with regard to the order of day, month, and year, the separator (",", " ", "_", "/") between day, month, and year, and whether the months was coded using numbers or abbreviated names (and the latter came in different languages)...

okay, these were still humans who messed it up

but what about an update of the software of an eye-tracking or pupil dilation machine or the use of the same software in different locales?

all such things I had seen over the last years, and I remember the trouble I had because of them...

wrap-up

as I sad already before, today many people analyse data which were recorded not by themselves alone but data sets to which multiple people provided their part

such data sets often amplify the problem as even in the absence of typos and the like[17], different people might enter data in different languages (e.g., 'barbary macaque', Berberaffe', or 'Macaque de Barbarie')

but even in case of a small data set, coded by a single and extremely trustworthy person like yourself, don't trust yourself

at least I can say for me that I don't trust myself when it comes to data coding, processing, and programming, etc., and I benefit from that a lot in my daily work... in fact, I check and check and check, and often I find mistakes that I made

[17] which is rare

a word of caution

what I'd showed you today was not at all meant as something like 'this is what you have to do to ensure good quality data'

rather, I wanted to only raise awareness about the disparate need of such checks

which particular checks will be required will certainly vary from project to project, and it might vary a lot

so for each particular project you are involved in, you need to carefully come up with the checks needed

this might not always be easy and straightforward and require some thinking out of the box

but I think it could also be fun

so be creative and play detective with your data and scrutinize them from every possible angle

in the end you might benefit from that a lot

a word of caution

I think, as researchers and scientists, we have the responsibility to work with data that are correct

and I think, we should take this responsibility very serious

from my own experience, I have the impression that quite a few scientists don't take this responsibility very serious

but I think we all should

or what's the benefit of analyzing erroneous data?

thank you very much!

```
 [1] "typo"    "Typo"    "typo "   "typoe"   "ytpo"
 [6] "tai-po"  "tYpo"    "typoo"   "ytop"    "phython"
[11] "pyto"    "..."
```