Preparing your data for analysis

Prepared by:
Dr Cameron Hurst
QIMR Berghofer Statistics Unit

Preamble

This document provides guidelines about how you should prepare your data before consulting a biostatistician. Ideally, your first meeting with the biostatistician once you have collected your data should involve analysis of your data (i.e. addressing your research questions) rather than the backwards-and-forwards of preparing data that we often see. Following the steps outlined in this document could save you lots of time. In other words, spending 15 minutes reading this document now could save you, and the biostatistician, a lot of time and effort later; time best spent producing strong and compelling results.

Contents

1 Software

Before we get into the minutiae of preparing our data, we should first think about software. **For data preparation, MS Excel is probably one of the most useful tools.** It has a large number of functions that can take the pain out of data preparation, especially when we would otherwise have to perform a repetitive task a large number of times. The exception will be if we are dealing with a very

big dataset which may have hundreds of variables and hundreds of thousands of patient records. Typically, interacting with these large datasets should be performed via a database management system.

For analysis, the biostatisticians will probably employ one of the main (and well put together) statistical analysis packages like R, Stata or SAS. While these three packages have some different approaches in how they deal with data, through syntax, it is quite easy for the biostatisticians to skirt around these slight differences.

2 Coding your data

2.1 Coding values

Before we go into more details, let's consider a simple little dataset:

Figure 1: A toy dataset in MS Excel

	A	B	C	D	E	F	G	H
1	UR_number	dob	gender	metformin	hba1c	bmi	hypertension	biomarkerXYZ
2	1005854	10/14/1996	Female	No	8.2	17.92822	No	2865
3	3600741	7/12/1998	Female	Yes	11.6	21.09375	No	**<10**
4	877537	2/1/1996	Male	No	7.6	30.18675	No	3488
5	856863	4/3/1996	Female	No	8.5	21.45357	Yes	4568
6	3010061	9/30/1998	Male	No	9.3	29.67523	No	5388
7	3298329	1/21/1993	Female	No	7.4	24.53266	No	3571
8	1043109	1/30/1993	Male	No	8.9	27.26627	No	**>10000**
9	852052	2/27/1992	Female	No	9.6	20.17325	No	3096
10	256467	2/27/1992	Female	No	8.5	20.1017	No	3616
11	889914	11/25/1998	Female	No	9.2	20.36913	Yes	146
12	3331506	6/6/1995	Male	No	8.4	21.77788	No	6316
13	1191567	4/7/1994	Female	No	8.7	**-999**	No	6256
14	883539	5/21/1998	Female	No	10.7	25.97375	No	3247
15	897958	6/29/1998	Female	No	8.2	27.72641	Yes	1586
16	258899	9/6/1993	Male	No	7.8	18.63336	No	1012
17	239447	7/25/1991	Female	No	9.4	27.43755	No	7304

The first thing we should note about the dataset in Figure 1 is that **variables are represented by columns and observations by rows**. This is the appropriate format for a large majority of statistical packages. Most clinical datasets will be represented in this way by default, but software used by many laboratory-based researchers will present data the other way round (where variables are listed in the rows). If this is the case, the data will need to be transposed before it can be read into most statistical packages.

The dataset provided in Figure 1 also has a couple of problems that would need to be resolved before we could conduct an analysis. First, we will deal with **coding the data**.

Generally speaking, statistical software doesn't like words. These packages can deal with it, but it causes major hassles that can be easily avoided by numerically coding our data.

For example, in statistical software:

$$No \neq no \neq NO$$

It is much better to numerically code our categorical variables. Generally with binary variables you should code them with 0s and 1s (usually 0 implies absence

and 1 suggests presence), and multiclass variables from *1, 2, ... , k*, where *k* is the number of classes. Consider Figure 2 below.

Figure 2: Toy dataset with dummy coded variables added

	A	B	C	D	E	F	G	H	I	J	K
1	UR_number	dob	gender	metformin	hba1c	bmi	hypertension	biomarkerXYZ	htn	metf	gen
2	1005854	10/14/1996	Female	No	8.2	17.92822	No	2865	0	0	1
3	3600741	7/12/1998	Female	Yes	11.6	21.09375	No	**<10**	0	1	1
4	877537	2/1/1996	Male	No	7.6	30.18675	No	3488	0	0	0
5	856863	4/3/1996	Female	No	8.5	21.45357	Yes	4568	1	0	1
6	3010061	9/30/1998	Male	No	9.3	29.67523	No	5388	0	0	0
7	3298329	1/21/1993	Female	No	7.4	24.53266	No	3571	0	0	1
8	1043109	1/30/1993	Male	No	8.9	27.26627	No	**>10000**	0	0	0
9	852052	2/27/1992	Female	No	9.6	20.17325	No	3096	0	0	1
10	256467	2/27/1992	Female	No	8.5	20.1017	No	3616	0	0	1
11	889914	11/25/1998	Female	No	9.2	20.36913	Yes	146	1	0	1
12	3331506	6/6/1995	Male	No	8.4	21.77788	No	6316	0	0	0
13	1191567	4/7/1994	Female	No	8.7	**-999**	No	6256	0	0	1
14	883539	5/21/1998	Female	No	10.7	25.97375	No	3247	0	0	1
15	897958	6/29/1998	Female	No	8.2	27.72641	Yes	1586	1	0	1
16	258899	9/6/1993	Male	No	7.8	18.63336	No	1012	0	0	0
17	239447	7/25/1991	Female	No	9.4	27.43755	No	7304	0	0	1

We have recoded the three binary variables, *gender*, *metformin* and *hypertension* into binary variables containing only zeros and ones (also called dummy variables). You will note that for both *metf* and *htn* that 0 implies absence (or No), and 1 suggests presence (or Yes). For the gender variable, it is somewhat arbitrary which group is zero and which is 1, but as a general rule we want the referent variable to be zero.

Finally, you may have noticed that I did not delete the original variables *gender, metformin* and *hypertension* from the dataset. This is because storage is cheap and you never know when you may need to go back to the original data.

Hints: Numerically coding categorical variables

- Where possible, numerically code your categorical variables
- Avoid deleting old variables. You never know when you may have to go back to them

2.2 Naming variables

When you do a lot of programming, you very quickly learn to hate long winded, inconsistent variable naming. Here, I will just briefly mention a couple of conventions that makes life easier for everyone.

Hints: Naming your variables

1. Avoid long and complicated variables names like DoesPatientHaveHypertension and just use a simple short name instead, like htn
2. Avoid different cases in the variable name. All lowercase is preferable. For example, bmi is preferable to BMI or Bmi
3. Do not use special or reserved characters in variable names (e.g. %, $, &, etc.). For example, **hba1c%** is an illegal variable name.
4. Statistical packages generally do not let you start variable names with a number (although they can be elsewhere in the name).

2.3 Data dictionary (*aka* codebooks)

Although the suggestions above make life a lot easier when it comes to reading in and analysing your data, there is also a cost: Details about which group is which, and even what the variables names actually mean may not be obvious. To get around this problem we should always create a **Data dictionary** (*aka* codebook). This document gives us (and others using your data) a key letting us know what the variables are, and what their values represent. Let's consider an abridged version of the data in Figure 2.

Figure 3: Toy dataset with dummy coded variables added

	A	B	C	D	E	F	G	H
1	UR_number	dob	hba1c	bmi	biomarkerXYZ	htn	metf	gen
2	1005854	10/14/1996	8.2	17.92822	2865	0	0	1
3	3600741	7/12/1998	11.6	21.09375	**<10**	0	1	1
4	877537	2/1/1996	7.6	30.18675	3488	0	0	0
5	856863	4/3/1996	8.5	21.45357	4568	1	0	1
6	3010061	9/30/1998	9.3	29.67523	5388	0	0	0
7	3298329	1/21/1993	7.4	24.53266	3571	0	0	1
8	1043109	1/30/1993	8.9	27.26627	**>10000**	0	0	0
9	852052	2/27/1992	9.6	20.17325	3096	0	0	1
10	256467	2/27/1992	8.5	20.1017	3616	0	0	1
11	889914	11/25/1998	9.2	20.36913	146	1	0	1
12	3331506	6/6/1995	8.4	21.77788	6316	0	0	0
13	1191567	4/7/1994	8.7	**-999**	6256	0	0	1
14	883539	5/21/1998	10.7	25.97375	3247	0	0	1
15	897958	6/29/1998	8.2	27.72641	1586	1	0	1
16	258899	9/6/1993	7.8	18.63336	1012	0	0	0
17	239447	7/25/1991	9.4	27.43755	7304	0	0	1

If you were presented with the data in Figure 3, some variables are clear, but you may struggle understanding what the other variables are, and what their values mean. Figure 4 represents a data dictionary that you might present along with your data.

Figure 4: A simple data dictionary

	A	B	C	D
1	**Variable name**	**Description**	**Values**	**Comment**
2	UR_number	Patient Unit Record Number		
3	dob	Date of Birth		
4	hba1c	Glycated hemogloblin		Percentage
5	bmi	Body mass index		
6	biomarkerXYZ	Some fictious biomarker		Has lower LOD 10, and upper LOD of 10000
7	htn	Hypertension	0=No, 1=Yes	
8	metf	Pateint prescribed Metformin	0=No, 1=Yes	
9	gen	Gender	0=Male, 1=Female	

When creating your data dictionary, it is best to imagine you are writing it for somebody else. That is, if you got hit by a bus tomorrow, would somebody be able to salvage your data

Hints: Avoid losing your data dictionary

If using Excel to get your data ready, it is a good idea to save your data dictionary in a completely separate file (not just a separate worksheet). When you covert your data into a text file, **Excel typically only saves the active sheet** (so if in a separate worksheet, your data dictionary will be lost).

3 De-identification and Re-identification

Part of our QIMR/Mater/Metro North agreements is that the biostatisticians are not allowed to see any information that might allow patients to be identified. This includes names, addresses and telephone numbers, **but it also includes patient UR numbers**. However, we can't just delete all of this identifying information as it is very important that patients remain identifiable WITHIN a dataset. What this means is that the biostatistician needs a unique patient identifier (a study-specific patient ID), but this identifier alone, cannot be used to work out the actual identity of the patient. You can do this by assigning a unique identifier within the dataset (a study-based patient ID) which maps to patients UR number, and then remove the patient UR number from the data to be provided to the biostatistician. However, it is very important (for reasons that will become clear later) that you (as the investigator) retain the key that maps the patient UR number to the study-based patient ID. In Figure 5 below we have generated a study patient ID (green column) that maps to the patient UR number (red column).

Figure 5: Adding a patient study ID variable

UR_number	PatientID	dob	hba1c	bmi	biomarkerXYZ	htn	metf	gen
1005854	p_1	10/14/1996	8.2	17.92822	2865	0	0	1
3600741	p_2	7/12/1998	11.6	21.09375	**<10**	0	1	1
877537	p_3	2/1/1996	7.6	30.18675	3488	0	0	0
856863	p_4	4/3/1996	8.5	21.45357	4568	1	0	1
3010061	p_5	9/30/1998	9.3	29.67523	5388	0	0	0
3298329	p_6	1/21/1993	7.4	24.53266	3571	0	0	1
1043109	p_7	1/30/1993	8.9	27.26627	**>10000**	0	0	0
852052	p_8	2/27/1992	9.6	20.17325	3096	0	0	1
256467	p_9	2/27/1992	8.5	20.1017	3616	0	0	1
889914	p_10	11/25/1998	9.2	20.36913	146	1	0	1
3331506	p_11	6/6/1995	8.4	21.77788	6316	0	0	0
1191567	p_12	4/7/1994	8.7	**-999**	6256	0	0	1
883539	p_13	5/21/1998	10.7	25.97375	3247	0	0	1
897958	p_14	6/29/1998	8.2	27.72641	1586	1	0	1
258899	p_15	9/6/1993	7.8	18.63336	1012	0	0	0
239447	p_16	7/25/1991	9.4	27.43755	7304	0	0	1

From the data above we would then create 2 files:

1. A dataset with the UR number removed **for the biostatistician** (Figure 6)
2. A file containing just the UR number and patient study ID **for you** (as the investigator) to retain (Figure 7)

Figure 6: De-identified (but re-identifiable) dataset for the biostatistician

PatientID	dob	hba1c	bmi	biomarkerXYZ	htn	metf	gen
p_1	10/14/1996	8.2	17.92822	2865	0	0	1
p_2	7/12/1998	11.6	21.09375	**<10**	0	1	1
p_3	2/1/1996	7.6	30.18675	3488	0	0	0
p_4	4/3/1996	8.5	21.45357	4568	1	0	1
p_5	9/30/1998	9.3	29.67523	5388	0	0	0
p_6	1/21/1993	7.4	24.53266	3571	0	0	1
p_7	1/30/1993	8.9	27.26627	**>10000**	0	0	0
p_8	2/27/1992	9.6	20.17325	3096	0	0	1
p_9	2/27/1992	8.5	20.1017	3616	0	0	1
p_10	11/25/1998	9.2	20.36913	146	1	0	1
p_11	6/6/1995	8.4	21.77788	6316	0	0	0
p_12	4/7/1994	8.7	**-999**	6256	0	0	1
p_13	5/21/1998	10.7	25.97375	3247	0	0	1
p_14	6/29/1998	8.2	27.72641	1586	1	0	1
p_15	9/6/1993	7.8	18.63336	1012	0	0	0
p_16	7/25/1991	9.4	27.43755	7304	0	0	1

Figure 7: File mapping study patient ID to UR number to be retained by investigator

UR_number	PatientID
1005854	p_1
3600741	p_2
877537	p_3
856863	p_4
3010061	p_5
3298329	p_6
1043109	p_7
852052	p_8
256467	p_9
889914	p_10
3331506	p_11
1191567	p_12
883539	p_13
897958	p_14
258899	p_15
239447	p_16

Take care: De-identifying and re-identifiability

- We have to be VERY careful when we de-identify data that we can re-identify it. The main reason is that we may make mistakes or omit information from the first dataset we provide the biostatistician. If we are unable to re-identify the data, we have to start again from scratch.
- We should also keep a copy of the original dataset in an unaltered form. Again we may need to go back to this. It is a good

idea to keep the Patient UR - study-based patient ID file in the same place as the original data

- There are some other issues that come into the generation of study patient IDs. We will come back to this topic when we consider longitudinal and other repeated measure data (Section 5).
- **Don't just use Excel to hide patient identifying information**, the identifying information needs to be physically removed from the dataset. **In fact, avoid using Excel's data (column or row) hiding features altogether.**

4 Missing values

4.1 Standard missing values

The various coding people use for missing values can often cause problems with software. As a general principle, we should avoiding mixing data types in a single column. For example, a column containing: 2,5,3.7,8.1,*missing*,7.6 will cause most statistics packages to declare the variable as a string (that is, sets of characters), making it difficult to perform mathematical operations on.

Another convention that people often use is using an impossible number for missing values (e.g. $BMI = -99$). This approach was popularized by SPSS where you can tell this package that a specific value represents a missing value. This is a particularly dangerous approach, because if you forget (and people often do) that these 'impossible values' may be used in the calculation of any sample statistics.

Generally speaking, we should code missing values in an unambiguous way. Perhaps the best way of coding a missing value is by using a missing value (i.e leave it blank).

Hints: Missing values

1. The best way to represent missing values is with a missing value: **Just leave it blank**
2. Avoid the impossible number approach (often employed by SPSS users) like, 99,-99,-999 etc.

4.2 Censored data

Often, we are presented with a situation where our measurement instrument is unable to measure values below or above a certain threshold (detection limits). This represents a type of missing data that is **censored**. Let's go back to our toy dataset.

Figure 8: A toy dataset with a doubly censored variable

	A	B	C	D	E	F	G	H
1	UR_number	dob	gender	metformin	hba1c	bmi	hypertension	biomarkerXYZ
2	1005854	10/14/1996	Female	No	8.2	17.92822	No	2865
3	3600741	7/12/1998	Female	Yes	11.6	21.09375	No	**<10**
4	877537	2/1/1996	Male	No	7.6	30.18675	No	3488
5	856863	4/3/1996	Female	No	8.5	21.45357	Yes	4568
6	3010061	9/30/1998	Male	No	9.3	29.67523	No	5388
7	3298329	1/21/1993	Female	No	7.4	24.53266	No	3571
8	1043109	1/30/1993	Male	No	8.9	27.26627	No	**>10000**
9	852052	2/27/1992	Female	No	9.6	20.17325	No	3096
10	256467	2/27/1992	Female	No	8.5	20.1017	No	3616
11	889914	11/25/1998	Female	No	9.2	20.36913	Yes	146
12	3331506	6/6/1995	Male	No	8.4	21.77788	No	6316
13	1191567	4/7/1994	Female	No	8.7	**-999**	No	6256
14	883539	5/21/1998	Female	No	10.7	25.97375	No	3247
15	897958	6/29/1998	Female	No	8.2	27.72641	Yes	1586
16	258899	9/6/1993	Male	No	7.8	18.63336	No	1012
17	239447	7/25/1991	Female	No	9.4	27.43755	No	7304

In Figure 8 we can see that *biomarkerXYZ* is both left censored (i.e. has a lower limit of detection of 10 units) and right censored (i.e. has an upper detection limit of 10,000 units). One of the problems is that if we tried to import this variable as is, we have a variable with mixed data types; most values are numerical, but the < 10 and >10000 values are actually character strings. This will cause problems for our statistics software. *How do we solve this problem?* Again we use two dummy variables, one for the lower detection limit (lowerDT), and one for the upper detection limit (upperDT). Then we can treat the values of <10 and >10000 as special cases of missing values.

Figure 9: Using dummy variables to code doubly censored variable

ur	dob	gender	metformin	hba1c	bmi	hypertension	biomarkerXYZ	lowerDL	upperDL
1005854	10/14/1996	Female	No	8.2	17.92822	No	2865	0	0
3600741	7/12/1998	Female	Yes	11.6	21.09375	No		1	0
877537	2/1/1996	Male	No	7.6	30.18675	No	3488	0	0
856863	4/3/1996	Female	No	8.5	21.45357	No	4568	0	0
3010061	9/30/1998	Male	No	9.3	29.67523	No	5388	0	0
3298329	1/21/1993	Female	No	7.4	24.53266	No	3571	0	0
1043109	1/30/1993	Male	No	8.9	27.26627	No		0	1
852052	2/27/1992	Female	No	9.6	20.17325	No	3096	0	0
256467	2/27/1992	Female	No	8.5	20.1017	No	3616	0	0
889914	11/25/1998	Female	No	9.2	20.36913	No	145	0	0
3331506	6/6/1995	Male	No	8.4	21.77788	No	6316	0	0
1191567	4/7/1994	Female	No	8.7		No	6256	0	0
883539	5/21/1998	Female	No	10.7	25.97375	No	3247	0	0
897958	6/29/1998	Female	No	8.2	27.72641	No	1586	0	0
258899	9/6/1993	Male	No	7.8	18.63336	No	1012	0	0
239447	7/25/1991	Female	No	9.4	27.43755	No	7304	0	0

Using the dummy variables to indicate both the presence and type of missing data (detection limits) give us two advantages. First, we have now converted our variable, *biomarkerXYZ*, into a purely numerical variable (won't cause problems for our stats software). Second, we can use the detection limit dummy variables for data imputation (methods for guessing what value *should* be there). It should be noted that most areas have their favourite approach to replacing limit of detection (LOD) censored values. Some areas use the LOD value itself, while others use some (at least for lower LODs) value between zero and the lower LOD.

The main point we want to make here though is that it is very simple to impose these particular approaches if you have your data formatted as in Figure 9 (regardless of what approach tends to be used in your area).

5 Longitudinal, repeated measures and clustered data

In many clinical studies we are presented with data where observations are correlated or clustered together. Examples of such data include:

- Measuring patients repeatedly over time (**longitudinal data**)
- Measuring patients repeatedly over 'space' (e.g. Ophthalmological study considering two eyes from each patient)
- Studies that consider multiple clinics (a multi-centre study). Patients attending the same clinic are likely to have more similar characteristics than patients attending different clinics (often called **clustered data**)

From a data preparation point of view all three of these problems are similar, so I will focus only on longitudinal data (the most common study design involving data clustering). Longitudinal data can cause distinct problems for people getting their data ready. In particular, two issues seem to arise again and again: **(1) the confusion between wide and long formats; and (2) (after de-identification) failing to link repeated values for a given individual via a patient-specific study ID**. It is not that formatting longitudinal data is that difficult, it is more that issues that may cause problems analytically simply don't occur to people. Consequently, we have to send them away to re-format their data.

Let's start by looking at a simple longitudinal dataset represented in **wide** format. Figure 10 provides a dataset containing baseline covariates (Gender and Age), and a longitudinal outcome (hba1c).

Figure 10: Simple longitudinal dataset represented in wide format

	A	B	C	D	E	F
1	UR_number	dob	gender	age	hba1c_Visit1	hba1c_Visit2
2	1005854	10/14/1996	Female	53	8.2	9.7
3	3600741	7/12/1998	Female	79	11.6	10.1
4	877537	2/1/1996	Male	64	7.6	
5	856863	4/3/1996	Female	67	8.5	10.7
6	3010061	9/30/1998	Male	48	9.3	9.3

In **wide format data**, each patient has a single row, and repeated measures are represented by adding additional columns. Now let's look at the same data in **long format** (Figure 11).

Figure 11: Simple longitudinal dataset represented in long format

	A	B	C	D	E	F
1	UR_number	dob	gender	age	visit	hba1c
2	1005854	10/14/1996	Female	53	1	8.2
3	3600741	7/12/1998	Female	79	1	11.6
4	877537	2/1/1996	Male	64	1	7.6
5	856863	4/3/1996	Female	67	1	8.5
6	3010061	9/30/1998	Male	48	1	9.3
7	1005854	10/14/1996	Female	53	2	9.7
8	3600741	7/12/1998	Female	79	2	10.1
9	877537	2/1/1996	Male	64	2	
10	856863	4/3/1996	Female	67	2	10.7
11	3010061	9/30/1998	Male	48	2	9.3

A couple things to note from Figure 11:

1. Each patient can now have multiple rows (one associated with each visit)
2. We have added a variable *visit* which tells which visit
3. There is some redundancy in the data. Baseline covariates are just copied and pasted for each visit
4. Patient 877537 has a missing value for visit 2 (highlighted in green). We could delete this observation if we wanted to (However, we don't need to, our statistical software will deal with this)

Now, de-identifying the data...

Figure 12: Simple longitudinal dataset represented in long format (final version)

	A	B	C	D	E	F	G
1	Patient_ID	dob	gender	age	visit	dateVisit	hba1c
2	p_1	10/14/1996	Female	53	1	2/1/2018	8.2
3	p_2	7/12/1998	Female	79	1	8/1/2018	11.6
4	p_3	2/1/1996	Male	64	1	12/1/2018	7.6
5	p_4	4/3/1996	Female	67	1	21/1/2018	8.5
6	p_5	9/30/1998	Male	48	1	2/2/2018	9.3
7	p_1	10/14/1996	Female	53	2	18/3/2019	9.7
8	p_2	7/12/1998	Female	79	2	12/2/2019	10.1
9	p_4	4/3/1996	Female	67	2	4/1/2019	10.7
10	p_5	9/30/1998	Male	48	2	3/3/2019	9.3

A couple of things to note in Figure 12: (1) I have included a *date of visit* variable; and (2) the Patient ID (Like the UR number in Figure 11), links different observations from the same patient together. This is very important when we want to run our models to analyse the longitudinal outcomes.

Almost always, the **best way to model longitudinal data is in long format**. Long format deals much better, for example, with inconsistency in numbers and/or timing of visits. Furthermore, long format deals better with having many variables measured over time (for example, time-varying covariates), especially when the frequency of measurement differs for different variables (e.g. blood pressure is measured daily, but blood sugar weekly). Finally, it is much easier

(programmatically) to convert **BACK TO** wide format **FROM** long format than the other way around.

There are a couple of tricky issues that arise when converting your data from wide to long. To avoid these problems we suggest that you consider the following issues.

Take care:

- De-identification and re-identifiability can easily be messed up in the conversion of longitudinal data from wide to long format. Always:
  1. Add your study patient ID in the first step (i.e. while data still in wide format)
  2. Don't de-identify your data (e.g. remove UR number) until the final step. Make sure study patient ID and UR number still maintain a one-to-one relationship (i.e. they match).
  3. Where possible try to keep date of visit (as well as adding a visit number). Sometimes models are more accurate when we consider time continuously.

6 Exporting you data ready for analysis

There are many ways that you can export your data. You can save it as an Excel file, save it in the native format of the stats software that you have been using for data editing (e.g. an SPSS ".sav" file), but we strongly suggest that you save a copy of your data in a **CSV** (*Comma Separated Values*) file. A CSV file is simply a text file which uses commas as delimiters (to tell the software when to move on to the next value)

Hints: Saving your data in a Comma Separated Value (CSV) text file format

1. Open and edit your data in Excel
2. Save your data in the native Excel format (Just in case)
3. In Excel, go to 'Save as' and in the 'Save as type' filed choose CSV format (MS-DOS or Mac format doesn't really matter)
4. Take care not to include commas in the data-fields themselves. Comment-like variables (e.g. doctor's notes on a patient) are a particularly dangerous place for them.

REMEMBER: Make sure your data dictionary has been saved elsewhere. Many people make the data dictionary an extra sheet in their Excel file and then lose it when they convert their Excel file into a csv (or other text format) file.

7 Basic principles

There are a couple of simple steps that you can take to avoid the pain and suffering of getting your data ready (including unnecessary trips backwards-and forwards to the biostatistician):

Hints: Good practice in data preparation and management

1. **Always keep the original** version of your data.
2. **Save a copy of your data for each successive step** in the data preparation process (so one mistake doesn't mean you have to do it ALL over again). It is a good idea to use a date or version number in the file name.
3. When transforming or creating new variables based on old ones, **always keep the original variables in the data file** (e.g. if you create BMI, retain the weight and height variables). You may at a later time change your mind about how variables are represented, or even used in the model.
4. If you have the skills, **do all of your data preparation in a package like R, SAS or Stata using syntax** (then all of your steps can be run again in a few seconds with the press of a button). Also, using syntax is like keeping a log of all of your actions and choices.
5. **ALWAYS create a data dictionary**
6. **Create a logbook of decisions** you have made about the data (patients you have retrospectively chosen to exclude, how you chose cut-offs for clinical variables etc.). This logbook can often be incorporated into the data dictionary
7. **Make sure your data is de-identified, BUT re-identifiable** (i.e. replace UR number with a study-specific patient ID code) AND save the UR-study ID pairings in a secure place (preferably along with the original data). Also, this is probably best done in the VERY LAST STEP.
8. When in doubt, **ask the statistician first**

Checklist

1. **Variable names:**
   Renamed variables with short but informative names

   Removed spaces and not included special characters (e.g. , % etc.)

   Don't start variable names with a number

2. **Coding values:** Generated numerical codes for your categorical variables (e.g. 0=Male, 1=Female) Dealt appropriately with missing values (best practice is to leave blank)
3. Removed any type of mixed datatypes (e.g. in a column of heights we may have: 160, 188, 170cm, 165). The use of the 'cm' will cause problems.

   During steps 1 and 2 is the best time to create your data dictionary.

4. **Longitudinal data: (if applicable)**
   Converted from wide to long format, including:
   Creating a single column for each time varying variable. Replicating (repeating) your baseline covariates for every observation.

   Included a time count variable (e.g. Visit = 1, 2, ...). If possible, (and in addition to the time count variable) include the actual visit date.

   Included a patient specific identifier (e.g. PatientID) so repeated observations from the same patient can be linked together.

5. **De-identification and Re-identifiability** Generated a unique study patient identifier (e.g. PatientID) variable that will allow individual patients to be distinguished WITHIN the dataset

   Linked the generated patient identifier to the patient UR number, and then saved this in a separate location (the statistician should not see this)

   If the data are longitudinal, made sure that repeated values from the same patient have the same PatientID

6. **Provided a data file appropriate for analysis:** Generated a CSV file (or at least an Excel file) that will allow data to be easily imported into statistical software
   Kept copies of the dataset for all of the major intermediate steps of data preparation.

   Don't forget to save your data dictionary in a separate file (don't just keep in as an extra worksheet in your original Excel data file, it might get accidently lost when you convert the Excel file into a text (e.g. csv) format)

I will end this by saying something that might be obvious, but people clearly forget: **Unlike us, computers are generally not good at deduction**. They won't say things like: "Aha, I see the word 'Male' AND the word 'male' in the same

column, it is clear that the researcher meant these to be the same". In fact what would happen is that when we got this data into the stats package, we would have three gender categories (Males, males and Females). Furthermore, computers (via the stats package) won't deduce that patients in red are in the control group, and patients in green are in the treatment group. In the interests of efficiency (and time), it is always a good idea to have a look at your data first and think about what a computer may or may not understand.

A simple example

Finally, let's finish by looking at poorly presented data, and how we can rework this into a dataset following good data management principles. In Figure 13 we have a poorly presented dataset.

Figure 13: Poorly presented data

Control						
Treated						
	Patient: David Smith					
	Age	BMI	Gender	SBPvist1	SBPvisit2	SBPvisit3
	25	27.3	Male	128	131	126
	Patient: Sara Jones					
	Age	Gender	BMI	SBPvisit1	SBPvisit3	
	22	Female	24.5	115	127	
	Patient: John Lee					
	Age	BMI	Gender	SBPvisit1	SBPvisit2	SBPVISIT3
	38	29.8	M	132	125	120
	Patient: Ezra Neusbaum					
	Age	BMI	Gender	SBPvisit1	SBPvisit2	SBPVISIT3
	29	27.2	Male	125	124	125

Now what is wrong with this data?

1. Colour is used to represent group membership (and statistics software doesn't acknowledge colour)
2. Patients' names are used allowing patient identification
3. Patient identifier (name in this case) is put above (rather than beside) each observation so data is not in a neat rectangle (would cause problems for stats software)

Figure 14: Data presented in Figure 13 re-formatted for analysis

patient_id	group	visit	age	bmi	gender	sbp
1	0	1	25	27.3	0	128
1	0	2	25	27.3	0	131
1	0	3	25	27.3	0	126
2	0	1	22	24.5	1	115
2	0	3	22	24.5	1	127
3	1	1	29	29.8	0	132
3	1	2	29	29.8	0	125
3	1	3	29	29.8	0	120
4	1	1	27	27.2	0	125
4	1	2	27	27.2	0	124
4	1	3	27	27.2	0	125

Figure 15: Data dictionary for the dataset presented in Figures 13 and 14.

VARIABLE NAME	DESCRIPTION	VALUES	COMMENT
patient_id	Study specific patient id		See file XYZ for original UR numbers
group	Treatment group	0: Control 1: Treated	Based on intention-to-treat
visit	Clinic visit	1: Baseline 2: 1 month 3: 6 months	
age	Age at baseline		Baseline covariate
bmi	Body mass index at baseline		Baseline covariate
gender		0: male 1: female	Baseline covariate
sbp	Systolic blood pressure		Time-varying outcome

4. Ordering of variables is not consistent between patients: 1. Gender and BMI have been transposed for patient 2 and; 2. Visit 3 value has been put into visit 2's column for patient 2 (who has a missing value for visit 2)
5. Inconsistent coding has been used for the gender variable (Male vs M)
6. Mixed upper and lower case for the same variable (SBPVISIT3 vs SBPvisit3). Stats software will see these as completely separate variables.

Now let's look at the same dataset reconfigured following practice we advocate in this document. Figure 14 provides the new dataset, and Figure 15 the accompanying data dictionary.