Lecture 2: Data Frame, Matrix, List

Abhijit Dasgupta

September 19, 2018

Preamble

Practice makes perfect

- Start using RSeek
- Other resources on website http://www.araastat.com/BIOF339_PracticalR
- Beg, Borrow, Steal code that you need
  - R is open-source, so is meant to be shared

R coding conventions

```r
# This is a comment, which doesn't get evaluated

1:3 # This is also a comment
```

```
## [1] 1 2 3
```

```r
# Multi-line code

x <- c(1, 2,
       3, 4, 5, 6,
       7)
x
```

```
## [1] 1 2 3 4 5 6 7
```

Google has a style guide for how to write R code

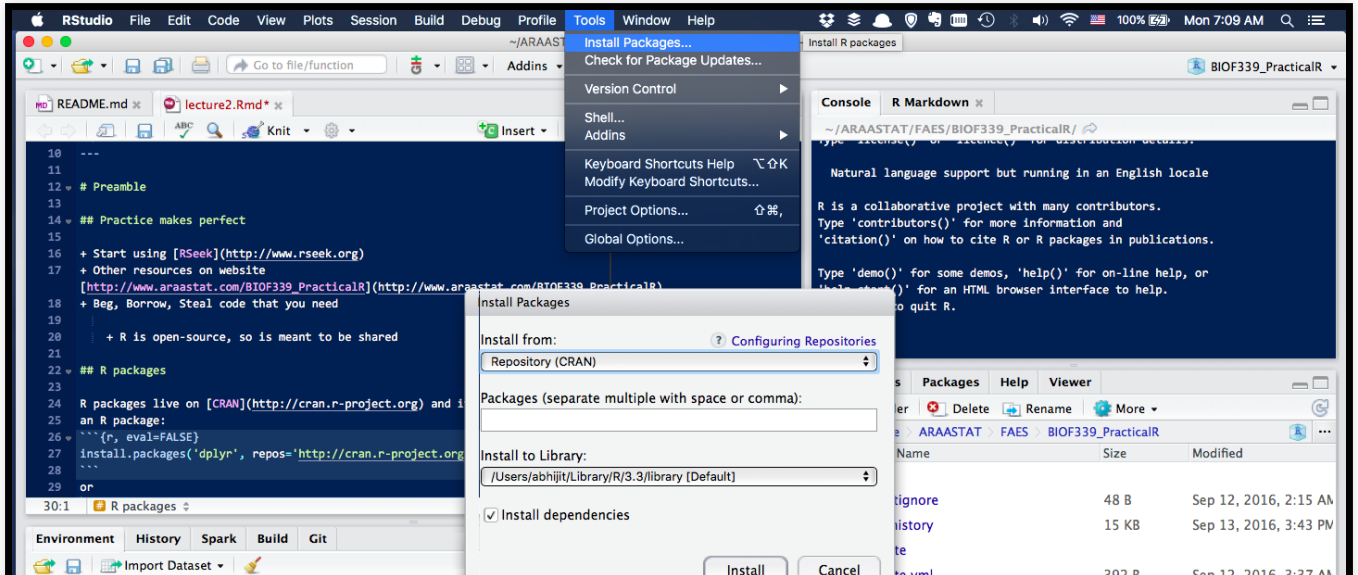
R packages

R packages live on CRAN and its mirrors. To install an R package:

```
install.packages('dplyr', repos='http://cran.r-project.org')
```

or

```
knitr::include_graphics('lecture2_img/install_package.png')
```



R Packages

To use a package, or rather, use the functions from the package, you have to load it into R

```
library(dplyr)
```

We'll talk about packages later in the semester.

We will concentrate now on what is known as **Base R**, that is, the functions that are available when R is installed

Loading data

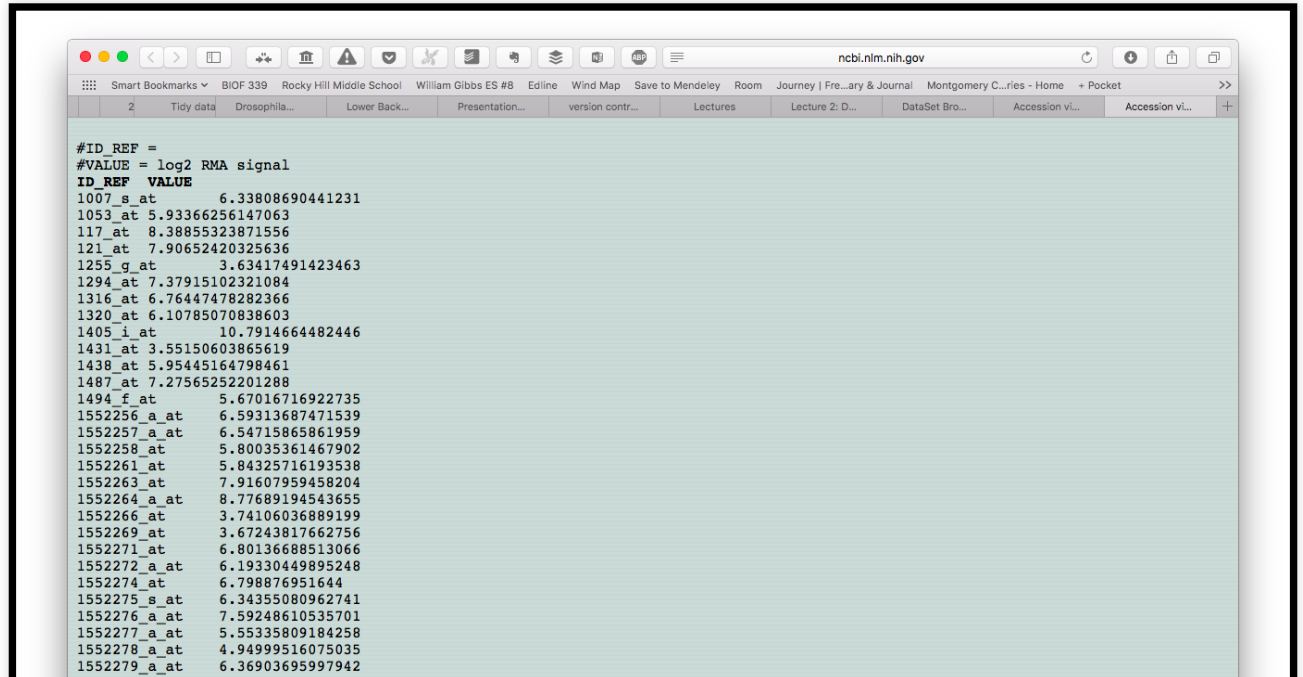
We will usually load CSV files, since they are the easiest for R. The typical suggestion if you have Excel data is to save the sheet as a CSV and then import it into R.

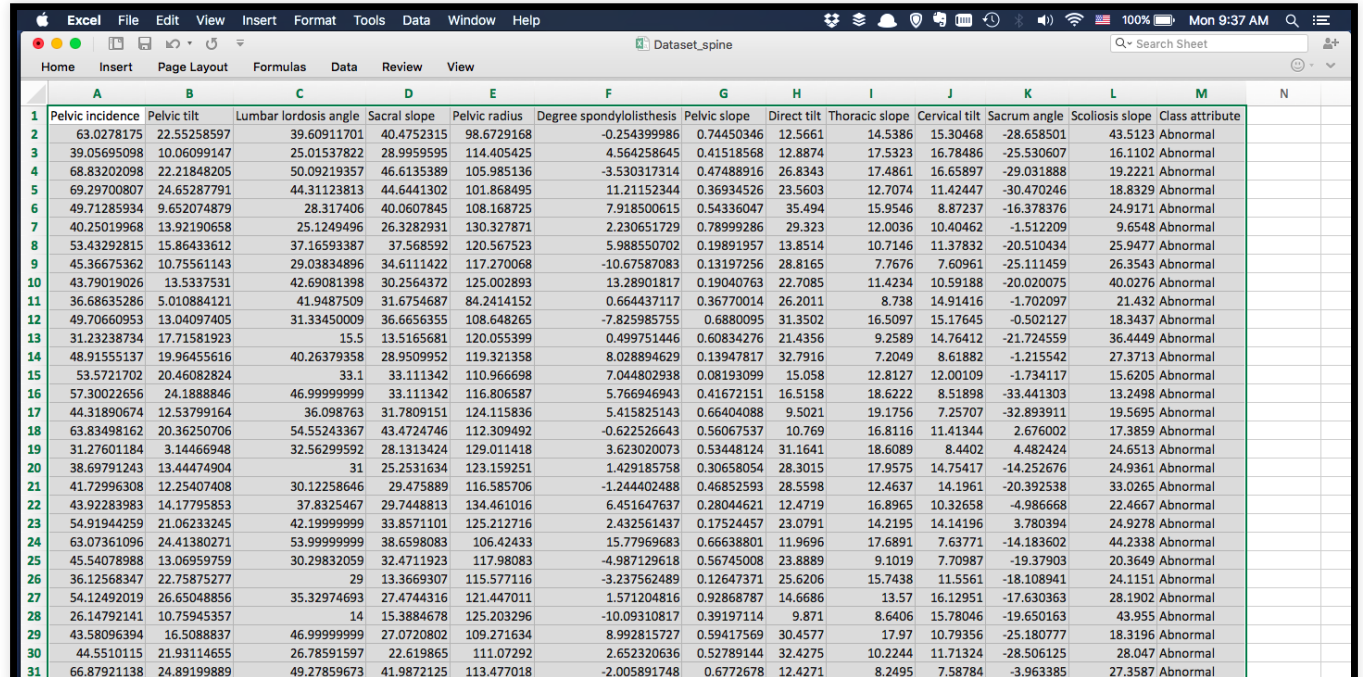
You can also load Excel files directly using either the `readxl` or `rio` packages

The structure of data sets

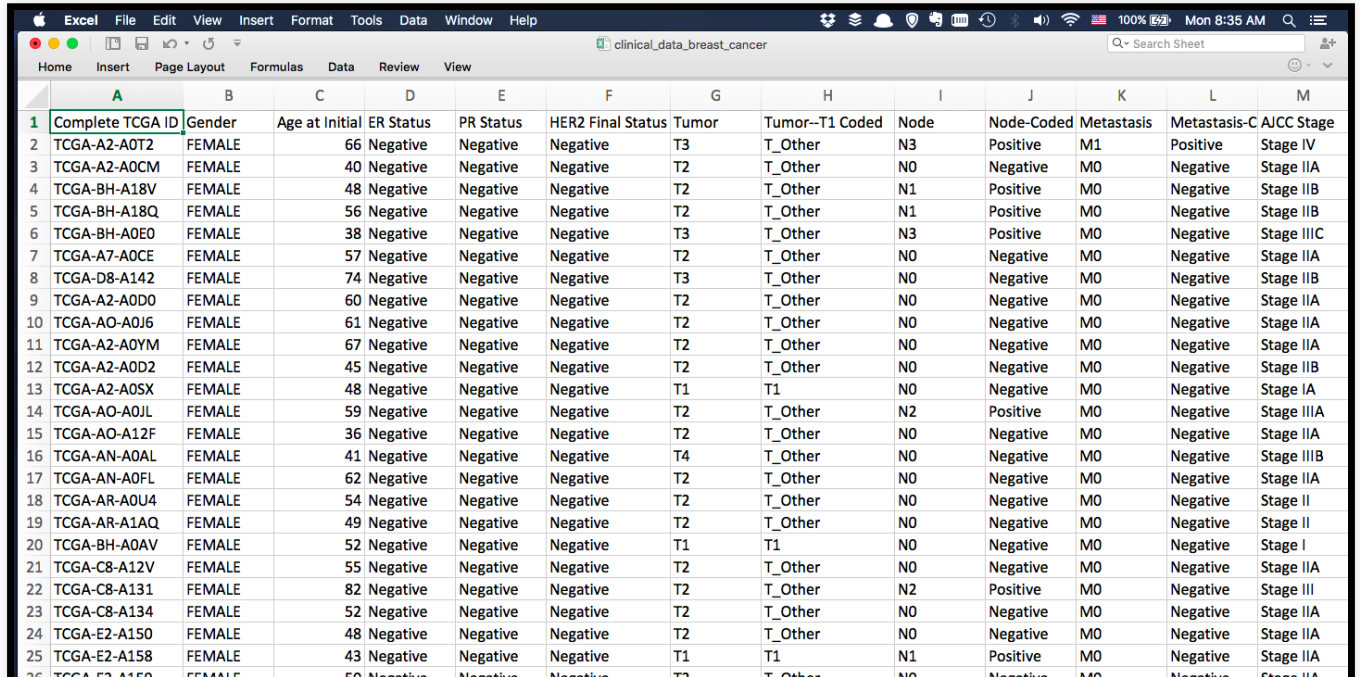
Tables

- Data is typically in a rectangular format
  - spreadsheet, database table
  - CSV (comma-separated values) or TSV (tab-separated values) files
- Characteristic
  - Rows are observations
  - Columns are variables


ncbi.nlm.nih.gov
Smart Bookmarks
BIOF 339
Rocky Hill Middle School
William Gibbs ES #8
Edline
Wind Map
Save to Mendeley
Room
Journey | Fre...ary & Journal
Montgomery C...ries - Home
+ Pocket
2
Tidy data
Drosophila...
Lower Back...
Presentation...
version contr...
Lectures
Lecture 2: D...
DataSet Bro...
Accession vi...
Accession vi...
#ID_REF =
#VALUE = log2 RMA signal
ID_REF VALUE
1007_s_at 6.33808690441231
1053_at 5.93366256147063
117_at 8.38855323871556
121_at 7.90652420325636
1255_g_at 3.63417491423463
1294_at 7.37915102321084
1316_at 6.76447478282366
1320_at 6.10785070838603
1405_i_at 10.7914664482446
1431_at 3.55150603865619
1438_at 5.95445164798461
1487_at 7.27565252201288
1494_f_at 5.67016716922735
1552256_a_at 6.59313687471539
1552257_a_at 6.54715865861959
1552258_at 5.80035361467902
1552261_at 5.84325716193538
1552263_at 7.91607959458204
1552264_a_at 8.77689194543655
1552266_at 3.74106036889199
1552269_at 3.67243817662756
1552271_at 6.80136688513066
1552272_a_at 6.19330449895248
1552274_at 6.798876951644
1552275_s_at 6.34355080962741
1552276_a_at 7.59248610535701
1552277_a_at 5.55335809184258
1552278_a_at 4.94999516075035
1552279_a_at 6.36903695997942

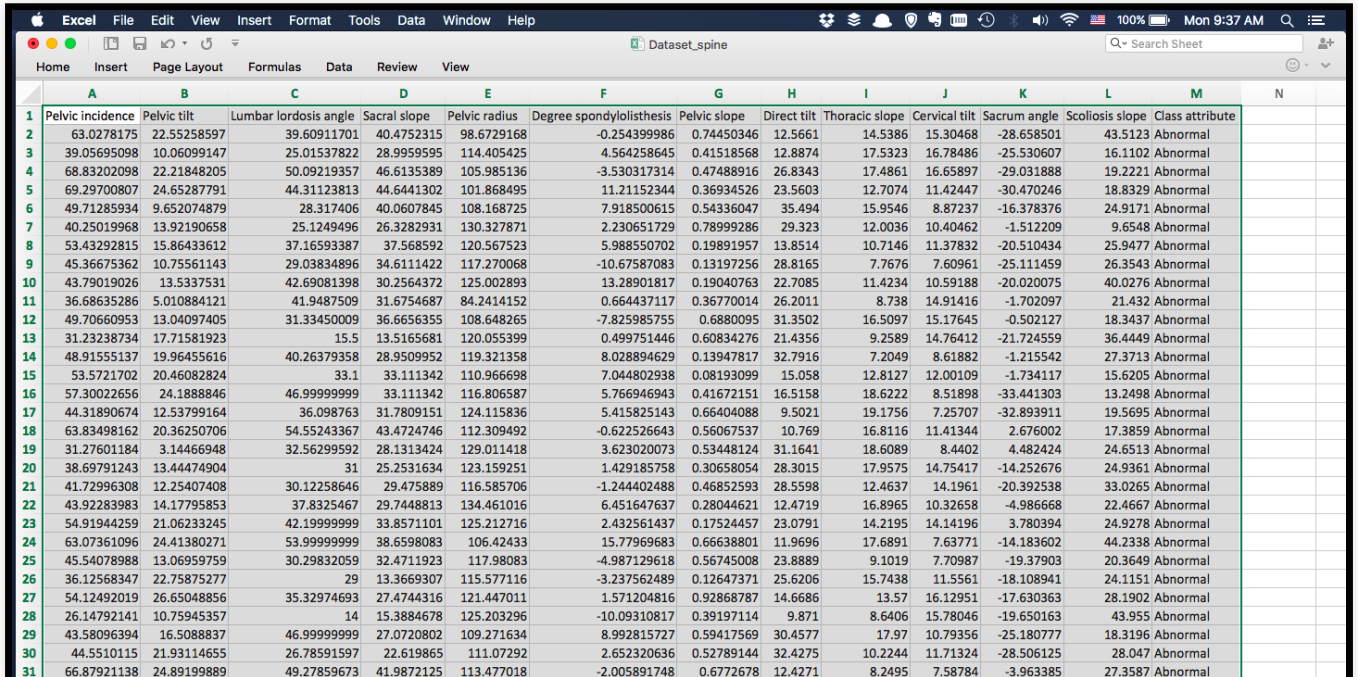


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Pelvic incidence	Pelvic tilt	Lumbar lordosis angle	Sacral slope	Pelvic radius	Degree spondylolisthesis	Pelvic slope	Direct tilt	Thoracic slope	Cervical tilt	Sacrum angle	Scoliosis slope	Class attribute	
2	63.0278175	22.55258597	39.60911701	40.4752315	98.6729168	-0.254399986	0.74450346	12.5661	14.5386	15.30468	-28.658501	43.5123	Abnormal	
3	39.05695098	10.06099147	25.01537822	28.9959595	114.405425	4.564258645	0.41518568	12.8874	17.5323	16.78486	-25.530607	16.1102	Abnormal	
4	68.83202098	22.21848205	50.09219357	46.6135389	105.985136	-3.530317314	0.47488916	26.8343	17.4861	16.65897	-29.031888	19.2221	Abnormal	
5	69.29700807	24.65287791	44.31123813	44.6441302	101.868495	11.21152344	0.36934526	23.5603	12.7074	11.42447	-30.470246	18.8329	Abnormal	
6	49.71285934	9.652074879	28.317406	40.0607845	108.168725	7.918500615	0.54336047	35.494	15.9546	8.87237	-16.378376	24.9171	Abnormal	
7	40.25019968	13.92190658	25.1249496	26.3282931	130.327871	2.230651729	0.78999286	29.323	12.0036	10.40462	-1.512209	9.6548	Abnormal	
8	53.43292815	15.86433612	37.16593387	37.568592	120.567523	5.988550702	0.19891957	13.8514	10.7146	11.37832	-20.510434	25.9477	Abnormal	
9	45.36675362	10.75561143	29.03834896	34.6111422	117.270068	-10.67587083	0.13197256	28.8165	7.7676	7.60961	-25.111459	26.3543	Abnormal	
10	43.79019026	13.5337531	42.69081398	30.2564372	125.002893	13.28901817	0.19040763	22.7085	11.4234	10.59188	-20.020075	40.0276	Abnormal	
11	36.68635286	5.010884121	41.9487509	31.6754687	84.2414152	0.664437117	0.36770014	26.2011	8.738	14.91416	-1.702097	21.432	Abnormal	
12	49.70660953	13.04097405	31.33450009	36.6656355	108.648265	-7.825985755	0.6880095	31.3502	16.5097	15.17645	-0.502127	18.3437	Abnormal	
13	31.23238734	17.71581923	15.5	13.5165681	120.055399	0.499751446	0.60834276	21.4356	9.2589	14.76412	-21.724559	36.4449	Abnormal	
14	48.91555137	19.96455616	40.26379358	28.9509952	119.321358	8.028894629	0.13947817	32.7916	7.2049	8.61882	-1.215542	27.3713	Abnormal	
15	53.5721702	20.46082824	33.1	33.111342	110.966698	7.044802938	0.08193099	15.058	12.8127	12.00109	-1.734117	15.6205	Abnormal	
16	57.30022656	24.1888846	46.99999999	33.111342	116.806587	5.766946943	0.41672151	16.5158	18.6222	8.51898	-33.441303	13.2498	Abnormal	
17	44.31890674	12.53799164	36.098763	31.7809151	124.115836	5.415825143	0.66404088	9.5021	19.1756	7.25707	-32.893911	19.5695	Abnormal	
18	63.83498162	20.36250706	54.55243367	43.4724746	112.309492	-0.622526643	0.56067537	10.769	16.8116	11.41344	2.676002	17.3859	Abnormal	
19	31.27601184	3.14466948	32.56299592	28.1313424	129.011418	3.623020073	0.53448124	31.1641	18.6089	8.4402	4.482424	24.6513	Abnormal	
20	38.69791243	13.44474904	31	25.2531634	123.159251	1.429185758	0.30658054	28.3015	17.9575	14.75417	-14.252676	24.9361	Abnormal	
21	41.72996308	12.25407408	30.12258646	29.475889	116.585706	-1.244402488	0.46852593	28.5598	12.4637	14.1961	-20.392538	33.0265	Abnormal	
22	43.92283983	14.17795853	37.8325467	29.7448813	134.461016	6.451647637	0.28044621	12.4719	16.8965	10.32658	-4.986668	22.4667	Abnormal	
23	54.91944259	21.06233245	42.19999999	33.8571101	125.212716	2.432561437	0.17524457	23.0791	14.2195	14.14196	3.780394	24.9278	Abnormal	
24	63.07361096	24.41380271	53.99999999	38.6598083	106.42433	15.77969683	0.66638801	11.9696	17.6891	7.63771	-14.183602	44.2338	Abnormal	
25	45.54078988	13.06959759	30.29832059	32.4711923	117.98083	-4.987129618	0.56745008	23.8889	9.1019	7.70987	-19.37903	20.3649	Abnormal	
26	36.12568347	22.75875277	29	13.3669307	115.577116	-3.237562489	0.12647371	25.6206	15.7438	11.5561	-18.108941	24.1151	Abnormal	
27	54.12492019	26.65048856	35.32974693	27.4744316	121.447011	1.571204816	0.92868787	14.6686	13.57	16.12951	-17.630363	28.1902	Abnormal	
28	26.14792141	10.75945357	14	15.3884678	125.203296	-10.09310817	0.39197114	9.871	8.6406	15.78046	-19.650163	43.955	Abnormal	
29	43.58096394	16.5088837	46.99999999	27.0720802	109.271634	8.992815727	0.59417569	30.4577	17.97	10.79356	-25.180777	18.3196	Abnormal	
30	44.5510115	21.93114655	26.78591597	22.619865	111.07292	2.652320636	0.52789144	32.4275	10.2244	11.71324	-28.506125	28.047	Abnormal	
31	66.87921138	24.89199889	49.27859673	41.9872125	113.477018	-2.005891748	0.6772678	12.4271	8.2495	7.58784	-3.963385	27.3587	Abnormal	



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Complete TCGA ID	Gender	Age at Initial	ER Status	PR Status	HER2 Final Status	Tumor	Tumor--T1 Coded	Node	Node-Coded	Metastasis	Metastasis-C	AJCC Stage
2	TCGA-A2-A0T2	FEMALE	66	Negative	Negative	Negative	T3	T_Other	N3	Positive	M1	Positive	Stage IV
3	TCGA-A2-A0CM	FEMALE	40	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
4	TCGA-BH-A18V	FEMALE	48	Negative	Negative	Negative	T2	T_Other	N1	Positive	M0	Negative	Stage IIB
5	TCGA-BH-A18Q	FEMALE	56	Negative	Negative	Negative	T2	T_Other	N1	Positive	M0	Negative	Stage IIB
6	TCGA-BH-A0E0	FEMALE	38	Negative	Negative	Negative	T3	T_Other	N3	Positive	M0	Negative	Stage IIIC
7	TCGA-A7-A0CE	FEMALE	57	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
8	TCGA-D8-A142	FEMALE	74	Negative	Negative	Negative	T3	T_Other	N0	Negative	M0	Negative	Stage IIB
9	TCGA-A2-A0D0	FEMALE	60	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
10	TCGA-AO-A0J6	FEMALE	61	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
11	TCGA-A2-A0YM	FEMALE	67	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
12	TCGA-A2-A0D2	FEMALE	45	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIB
13	TCGA-A2-A0SX	FEMALE	48	Negative	Negative	Negative	T1	T1	N0	Negative	M0	Negative	Stage IA
14	TCGA-AO-A0JL	FEMALE	59	Negative	Negative	Negative	T2	T_Other	N2	Positive	M0	Negative	Stage IIIA
15	TCGA-AO-A12F	FEMALE	36	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
16	TCGA-AN-A0AL	FEMALE	41	Negative	Negative	Negative	T4	T_Other	N0	Negative	M0	Negative	Stage IIIB
17	TCGA-AN-A0FL	FEMALE	62	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
18	TCGA-AR-A0U4	FEMALE	54	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage II
19	TCGA-AR-A1AQ	FEMALE	49	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage II
20	TCGA-BH-A0AV	FEMALE	52	Negative	Negative	Negative	T1	T1	N0	Negative	M0	Negative	Stage I
21	TCGA-C8-A12V	FEMALE	55	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
22	TCGA-C8-A131	FEMALE	82	Negative	Negative	Negative	T2	T_Other	N2	Positive	M0	Negative	Stage III
23	TCGA-C8-A134	FEMALE	52	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
24	TCGA-E2-A150	FEMALE	48	Negative	Negative	Negative	T2	T_Other	N0	Negative	M0	Negative	Stage IIA
25	TCGA-E2-A158	FEMALE	43	Negative	Negative	Negative	T1	T1	N1	Positive	M0	Negative	Stage IIA

Let's look at a dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Pelvic incidence	Pelvic tilt	Lumbar lordosis angle	Sacral slope	Pelvic radius	Degree spondylolisthesis	Pelvic slope	Direct tilt	Thoracic slope	Cervical tilt	Sacrum angle	Scoliosis slope	Class attribute	
2	63.0278175	22.55258597	39.60911701	40.4752315	98.6729168	-0.254399986	0.74450346	12.5661	14.5386	15.30468	-28.658501	43.5123	Abnormal	
3	39.05695098	10.06099147	25.01537822	28.9959595	114.405425	4.564258645	0.41518568	12.8874	17.5323	16.78486	-25.530607	16.1102	Abnormal	
4	68.83202098	22.21848205	50.09219357	46.6135389	105.985136	-3.530317314	0.47488916	26.8343	17.4861	16.65897	-29.031888	19.2221	Abnormal	
5	69.29700807	24.65287791	44.31123813	44.6441302	101.868495	11.21152344	0.36934526	23.5603	12.7074	11.42447	-30.470246	18.8329	Abnormal	
6	49.71285934	9.652074879	28.317406	40.0607845	108.168725	7.918500615	0.54336047	35.494	15.9546	8.87237	-16.378376	24.9171	Abnormal	
7	40.25019968	13.92190658	25.1249496	26.3282931	130.327871	2.230651729	0.78999286	29.323	12.0036	10.40462	-1.512209	9.6548	Abnormal	
8	53.43292815	15.86433612	37.16593387	37.568592	120.567523	5.988550702	0.19891957	13.8514	10.7146	11.37832	-20.510434	25.9477	Abnormal	
9	45.36675362	10.75561143	29.03834896	34.6111422	117.270068	-10.67587083	0.13197256	28.8165	7.7676	7.60961	-25.111459	26.3543	Abnormal	
10	43.79019026	13.5337531	42.69081398	30.2564372	125.002893	13.28901817	0.19040763	22.7085	11.4234	10.59188	-20.020075	40.0276	Abnormal	
11	36.68635286	5.010884121	41.9487509	31.6754687	84.2414152	0.664437117	0.36770014	26.2011	8.738	14.91416	-1.702097	21.432	Abnormal	
12	49.70660953	13.04097405	31.33450009	36.6656355	108.648265	-7.825985755	0.6880095	31.3502	16.5097	15.17645	-0.502127	18.3437	Abnormal	
13	31.23238734	17.71581923	15.5	13.5165681	120.055399	0.499751446	0.60834276	21.4356	9.2589	14.76412	-21.724559	36.4449	Abnormal	
14	48.91555137	19.96455616	40.26379358	28.9509952	119.321358	8.028894629	0.13947817	32.7916	7.2049	8.61882	-1.215542	27.3713	Abnormal	
15	53.5721702	20.46082824	33.1	33.111342	110.966698	7.044802938	0.08193099	15.058	12.8127	12.00109	-1.734117	15.6205	Abnormal	
16	57.30022656	24.1888846	46.99999999	33.111342	116.806587	5.766946943	0.41672151	16.5158	18.6222	8.51898	-33.441303	13.2498	Abnormal	
17	44.31890674	12.53799164	36.098763	31.7809151	124.115836	5.415825143	0.66404088	9.5021	19.1756	7.25707	-32.893911	19.5695	Abnormal	
18	63.83498162	20.36250706	54.55243367	43.4724746	112.309492	-0.622526643	0.56067537	10.769	16.8116	11.41344	2.676002	17.3859	Abnormal	
19	31.27601184	3.14466948	32.56299592	28.1313424	129.011418	3.623020073	0.53448124	31.1641	18.6089	8.4402	4.482424	24.6513	Abnormal	
20	38.69791243	13.44474904	31	25.2531634	123.159251	1.429185758	0.30658054	28.3015	17.9575	14.75417	-14.252676	24.9361	Abnormal	
21	41.72996308	12.25407408	30.12258646	29.475889	116.585706	-1.244402488	0.46852593	28.5598	12.4637	14.1961	-20.392538	33.0265	Abnormal	
22	43.92283983	14.17795853	37.8325467	29.7448813	134.461016	6.451647637	0.28044621	12.4719	16.8965	10.32658	-4.986668	22.4667	Abnormal	
23	54.91944259	21.06233245	42.19999999	33.8571101	125.212716	2.432561437	0.17524457	23.0791	14.2195	14.14196	3.780394	24.9278	Abnormal	
24	63.07361096	24.41380271	53.99999999	38.6598083	106.42433	15.77969683	0.66638801	11.9696	17.6891	7.63771	-14.183602	44.2338	Abnormal	
25	45.54078988	13.06959759	30.29832059	32.4711923	117.98083	-4.987129618	0.56745008	23.8889	9.1019	7.70987	-19.37903	20.3649	Abnormal	
26	36.12568347	22.75875277	29	13.3669307	115.577116	-3.237562489	0.12647371	25.6206	15.7438	11.5561	-18.108941	24.1151	Abnormal	
27	54.12492019	26.65048856	35.32974693	27.4744316	121.447011	1.571204816	0.92868787	14.6686	13.57	16.12951	-17.630363	28.1902	Abnormal	
28	26.14792141	10.75945357	14	15.3884678	125.203296	-10.09310817	0.39197114	9.871	8.6406	15.78046	-19.650163	43.955	Abnormal	
29	43.58096394	16.5088837	46.99999999	27.0720802	109.271634	8.992815727	0.59417569	30.4577	17.97	10.79356	-25.180777	18.3196	Abnormal	
30	44.5510115	21.93114655	26.78591597	22.619865	111.07292	2.652320636	0.52789144	32.4275	10.2244	11.71324	-28.506125	28.047	Abnormal	
31	66.87921138	24.89199889	49.27859673	41.9872125	113.477018	-2.005891748	0.6772678	12.4271	8.2495	7.58784	-3.963385	27.3587	Abnormal	

Let's look at a dataset

```
data_spine <- read.csv('lecture2_data/Dataset_spine.csv')
```

```
head(data_spine)
```

```
##   Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 1         63.02782   22.552586              39.60912     40.47523
## 2         39.05695   10.060991              25.01538     28.99596
## 3         68.83202   22.218482              50.09219     46.61354
## 4         69.29701   24.652878              44.31124     44.64413
## 5         49.71286    9.652075              28.31741     40.06078
## 6         40.25020   13.921907              25.12495     26.32829
##   Pelvic.radius Degree.spondylolisthesis Pelvic.slope Direct.tilt
## 1      98.67292               -0.254400    0.7445035     12.5661
## 2     114.40543                4.564259    0.4151857     12.8874
## 3     105.98514               -3.530317    0.4748892     26.8343
## 4     101.86850               11.211523    0.3693453     23.5603
## 5     108.16872                7.918501    0.5433605     35.4940
## 6     130.32787                2.230652    0.7899929     29.3230
##   Thoracic.slope Cervical.tilt Sacrum.angle Scoliosis.slope
## 1        14.5386      15.30468   -28.658501         43.5123
## 2        17.5323      16.78486   -25.530607         16.1102
## 3        17.4861      16.65897   -29.031888         19.2221
```

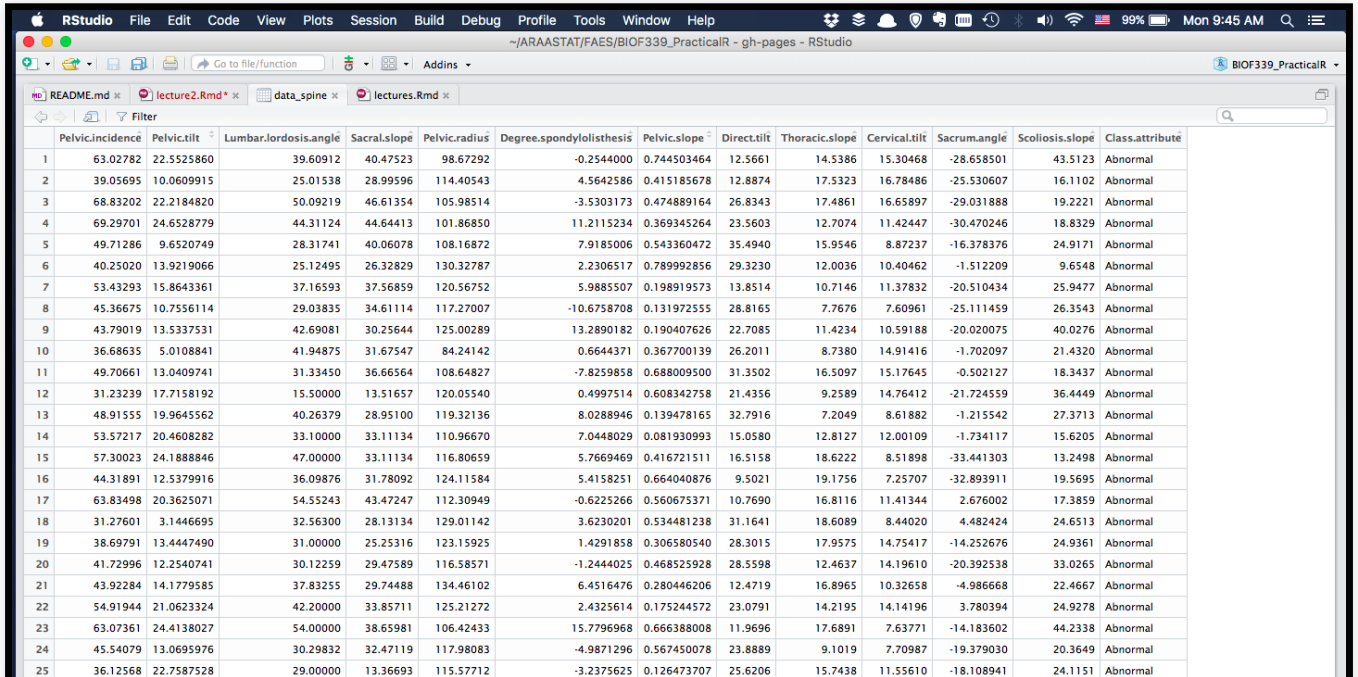
Ignore the first `##`; it denotes that this is R output

Let’s look at a dataset

- Assumes that the first row has variable names
- Replaces spaces with .
- Keeps numeric and character variables together

Let's look at a dataset

```
View(data_spine)  ## It looks like a matrix
```



	Pelvic.incidence	Pelvic.tilt	Lumbar.lordosis.angle	Sacral.slope	Pelvic.radius	Degree.spondylolisthesis	Pelvic.slope	Direct.tilt	Thoracic.slope	Cervical.tilt	Sacrum.angle	Scoliosis.slope	Class.attribute
1	63.02782	22.5525860	39.60912	40.47523	98.67292	-0.2544000	0.744503464	12.5661	14.5386	15.30468	-28.658501	43.5123	Abnormal
2	39.05695	10.0609915	25.01538	28.99596	114.40543	4.5642586	0.415185678	12.8874	17.5323	16.78486	-25.530607	16.1102	Abnormal
3	68.83202	22.2184820	50.09219	46.61354	105.98514	-3.5303173	0.474889164	26.8343	17.4861	16.65897	-29.031888	19.2221	Abnormal
4	69.29701	24.6528779	44.31124	44.64413	101.86850	11.2115234	0.369345264	23.5603	12.7074	11.42447	-30.470246	18.8329	Abnormal
5	49.71286	9.6520749	28.31741	40.06078	108.16872	7.9185006	0.543360472	35.4940	15.9546	8.87237	-16.378376	24.9171	Abnormal
6	40.25020	13.9219066	25.12495	26.32829	130.32787	2.2306517	0.789992856	29.3230	12.0036	10.40462	-1.512209	9.6548	Abnormal
7	53.43293	15.8643361	37.16593	37.56859	120.56752	5.9885507	0.198919573	13.8514	10.7146	11.37832	-20.510434	25.9477	Abnormal
8	45.36675	10.7556114	29.03835	34.61114	117.27007	-10.6758708	0.131972555	28.8165	7.7676	7.60961	-25.111459	26.3543	Abnormal
9	43.79019	13.5337531	42.69081	30.25644	125.00289	13.2890182	0.190407626	22.7085	11.4234	10.59188	-20.020075	40.0276	Abnormal
10	36.68635	5.0108841	41.94875	31.67547	84.24142	0.6644371	0.367700139	26.2011	8.7380	14.91416	-1.702097	21.4320	Abnormal
11	49.70661	13.0409741	31.33450	36.66564	108.64827	-7.8259858	0.688009500	31.3502	16.5097	15.17645	-0.502127	18.3437	Abnormal
12	31.23239	17.7158192	15.50000	13.51657	120.05540	0.4997514	0.608342758	21.4356	9.2589	14.76412	-21.724559	36.4449	Abnormal
13	48.91555	19.9645562	40.26379	28.95100	119.32136	8.0288946	0.139478165	32.7916	7.2049	8.61882	-1.215542	27.3713	Abnormal
14	53.57217	20.4608282	33.10000	33.11134	110.96670	7.0448029	0.081930993	15.0580	12.8127	12.00109	-1.734117	15.6205	Abnormal
15	57.30023	24.1888846	47.00000	33.11134	116.80659	5.7669469	0.416721511	16.5158	18.6222	8.51898	-33.441303	13.2498	Abnormal
16	44.31891	12.5379916	36.09876	31.78092	124.11584	5.4158251	0.664040876	9.5021	19.1756	7.25707	-32.893911	19.5695	Abnormal
17	63.83498	20.3625071	54.55243	43.47247	112.30949	-0.6225266	0.560675371	10.7690	16.8116	11.41344	2.676002	17.3859	Abnormal
18	31.27601	3.1446695	32.56300	28.13134	129.01142	3.6230201	0.534481238	31.1641	18.6089	8.44020	4.482424	24.6513	Abnormal
19	38.69791	13.4447490	31.00000	25.25316	123.15925	1.4291858	0.306580540	28.3015	17.9575	14.75417	-14.252676	24.9361	Abnormal
20	41.72996	12.2540741	30.12259	29.47589	116.58571	-1.2444025	0.468525928	28.5598	12.4637	14.19610	-20.392538	33.0265	Abnormal
21	43.92284	14.1779585	37.83255	29.74488	134.46102	6.4516476	0.280446206	12.4719	16.8965	10.32658	-4.986668	22.4667	Abnormal
22	54.91944	21.0623324	42.20000	33.85711	125.21272	2.4325614	0.175244572	23.0791	14.2195	14.14196	3.780394	24.9278	Abnormal
23	63.07361	24.4138027	54.00000	38.65981	106.42433	15.7796968	0.666388008	11.9696	17.6891	7.63771	-14.183602	44.2338	Abnormal
24	45.54079	13.0695976	30.29832	32.47119	117.98083	-4.9871296	0.567450078	23.8889	9.1019	7.70987	-19.379030	20.3649	Abnormal
25	36.12568	22.7587528	29.00000	13.36693	115.57712	-3.2375625	0.126473707	25.6206	15.7438	11.55610	-18.108941	24.1151	Abnormal

Let's look at a dataset

```
str(data_spine) ## Structure of a dataset
```

```
## 'data.frame':    310 obs. of  13 variables:
##  $ Pelvic.incidence        : num  63 39.1 68.8 69.3 49.7 ...
##  $ Pelvic.tilt             : num  22.55 10.06 22.22 24.65 9.65 ...
##  $ Lumbar.lordosis.angle   : num  39.6 25 50.1 44.3 28.3 ...
##  $ Sacral.slope            : num  40.5 29 46.6 44.6 40.1 ...
##  $ Pelvic.radius           : num  98.7 114.4 106 101.9 108.2 ...
##  $ Degree.spondylolisthesis: num  -0.254 4.564 -3.53 11.212 7.919 ...
##  $ Pelvic.slope            : num  0.745 0.415 0.475 0.369 0.543 ...
##  $ Direct.tilt             : num  12.6 12.9 26.8 23.6 35.5 ...
##  $ Thoracic.slope          : num  14.5 17.5 17.5 12.7 16 ...
##  $ Cervical.tilt           : num  15.3 16.78 16.66 11.42 8.87 ...
##  $ Sacrum.angle            : num  -28.7 -25.5 -29 -30.5 -16.4 ...
##  $ Scoliosis.slope         : num  43.5 16.1 19.2 18.8 24.9 ...
##  $ Class.attribute         : Factor w/ 2 levels "Abnormal","Normal": 1
```

So this is a `data.frame` object with 310 observations and 13 variables, of which one is a **factor** and the rest are **numeric**

It looks like a **list of things**

Dataframes

Dataframes are the primary mode of storing datasets in R

They were revolutionary in that they kept heterogeneous data together

They share properties of both a **matrix** and a **list**

```
class(data_spine)
```

```
## [1] "data.frame"
```

Technically, a data.frame is a list of vectors (or objects, generally) of the same length

Matrices

A **matrix** is a rectangular array of data *of the same type*

```
matrix(0, nrow=2, ncol=4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    0    0    0
## [2,]    0    0    0    0
```

```
matrix(letters, nrow=2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [
## [1,] "a"  "c"  "e"  "g"  "i"  "k"  "m"  "o"  "q"  "s"   "u"   "w"   "y
## [2,] "b"  "d"  "f"  "h"  "j"  "l"  "n"  "p"  "r"  "t"   "v"   "x"   "z
```

```
matrix(letters, nrow=2, byrow=T)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [
## [1,] "a"  "b"  "c"  "d"  "e"  "f"  "g"  "h"  "i"  "j"   "k"   "l"   "m
## [2,] "n"  "o"  "p"  "q"  "r"  "s"  "t"  "u"  "v"  "w"   "x"   "y"   "z
```

Matrices

You can create a matrix from a set of *vectors* of the same length

```
x <- c(1,2,3,4)
y <- c(10,20,30,40)
```

Put columns together

```
cbind(c(1,2,3,4), c(10,20,30,40)) ## Column bind
```

```
##      [,1] [,2]
## [1,]    1   10
## [2,]    2   20
## [3,]    3   30
## [4,]    4   40
```

Matrices

You can create a matrix from a set of *vectors* of the same length

```
x <- c(1,2,3,4)
y <- c(10,20,30,40)
```

Put rows together

```
example_matrix <- rbind(c(1,2,3,4), c(10,20,30,40)) ## Row bind
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

Extracting elements

```
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

```
example_matrix[1,] ## Extracts 1st row
```

```
## [1] 1 2 3 4
```

```
example_matrix[,2:3] ## extracts 2nd & 3rd columns
```

```
##      [,1] [,2]
## [1,]    2    3
## [2,]   20   30
```

```
example_matrix[1,4]
```

```
## [1] 4
```

Matrix properties

```
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

```
nrow(example_matrix) ## Number of rows
```

```
## [1] 2
```

```
ncol(example_matrix) ## Number of columns
```

```
## [1] 4
```

```
dim(example_matrix) ## shortcut for above
```

```
## [1] 2 4
```

Matrix arithmetic

```
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

```
example_matrix + 5 ## Add 5 to each element
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    6    7    8    9
## [2,]   15   25   35   45
```

```
example_matrix * 2 ## Multiply each element by 2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    4    6    8
## [2,]   20   40   60   80
```

Two matrices

```
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

```
example_matrix2 <- rbind(3:6, 9:12)
example_matrix2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    4    5    6
## [2,]    9   10   11   12
```

```
example_matrix + example_matrix2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    4    6    8   10
## [2,]   19   30   41   52
```

Two matrices

```
example_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
```

```
example_matrix2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    4    5    6
## [2,]    9   10   11   12
```

```
example_matrix * example_matrix2 ## Not matrix multiplication, but elemer
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    8   15   24
## [2,]   90  200  330  480
```

Two matrices

```
rbind(example_matrix, example_matrix2)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]   10   20   30   40
## [3,]    3    4    5    6
## [4,]    9   10   11   12
```

```
cbind(example_matrix, example_matrix2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    1    2    3    4    3    4    5    6
## [2,]   10   20   30   40    9   10   11   12
```

Two matrices

```
dim(example_matrix2)
```

```
## [1] 2 4
```

```
t(example_matrix2) ## Transpose of a matrix
```

```
##      [,1] [,2]
## [1,]    3    9
## [2,]    4   10

## [3,]    5   11
## [4,]    6   12
```

```
example_matrix %*% t(example_matrix2) ## Matrix multiplication
```

```
##      [,1] [,2]
## [1,]   50  110
## [2,]  500 1100
```

Lists

Lists are collections of arbitrary objects in R

```
example_list <- list(c('Andy','Brian','Harry'),
                     c(12, 16, 16),
                     c(TRUE, TRUE, FALSE),
                     matrix(1, nrow=2, ncol=3))
example_list
```

```
## [[1]]
## [1] "Andy"  "Brian" "Harry"
##
## [[2]]
## [1] 12 16 16
##
## [[3]]
## [1]  TRUE  TRUE FALSE
##
## [[4]]
##      [,1] [,2] [,3]
## [1,]    1    1    1
## [2,]    1    1    1
```

Extracting elements from lists

```
example_list[[3]]
```

```
## [1]  TRUE  TRUE FALSE
```

```
example_list[1:2]
```

```
## [[1]]
## [1] "Andy"  "Brian" "Harry"
##

## [[2]]
## [1] 12 16 16
```

Extracting elements from lists

```
example_list[[4]]
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    1
## [2,]    1    1    1
```

```
class(example_list[[4]])
```

```
## [1] "matrix"
```

```
example_list[[4]][1,]
```

```
## [1] 1 1 1
```

Named lists

```
example_named_list <- list('Names' = c('Andy','Brian','Harry'),
                      "YearsOfEducation" = c(12, 16, 16),
                      "Married" = c(TRUE, TRUE, FALSE),
                      'something' = matrix(1, nrow=2, ncol=3))


example_named_list[['Names']]
```

```
## [1] "Andy"  "Brian" "Harry"
```

```
example_named_list$Names
```

```
## [1] "Andy"  "Brian" "Harry"
```

```
example_named_list$Names[3]
```

```
## [1] "Harry"
```

Back to a Data Frame

A data.frame object is a **named list** where each element is of the same length

You can use both *matrix* and *list* functions to operate on data.frame objects!!

Data Frames

```
head(data_spine)
```

```
##   Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 1         63.02782   22.552586                39.60912     40.47523
## 2         39.05695   10.060991                25.01538     28.99596
## 3         68.83202   22.218482                50.09219     46.61354
## 4         69.29701   24.652878                44.31124     44.64413
## 5         49.71286    9.652075                28.31741     40.06078
## 6         40.25020   13.921907                25.12495     26.32829
##   Pelvic.radius Degree.spondylolisthesis Pelvic.slope Direct.tilt
## 1      98.67292                -0.254400    0.7445035     12.5661
## 2     114.40543                 4.564259    0.4151857     12.8874
## 3     105.98514                -3.530317    0.4748892     26.8343
## 4     101.86850                11.211523    0.3693453     23.5603
## 5     108.16872                 7.918501    0.5433605     35.4940
## 6     130.32787                 2.230652    0.7899929     29.3230
##   Thoracic.slope Cervical.tilt Sacrum.angle Scoliosis.slope
## 1        14.5386      15.30468   -28.658501         43.5123
## 2        17.5323      16.78486   -25.530607         16.1102
```

Data Frames

```
dim(data_spine)
```

```
## [1] 310  13
```

```
nrow(data_spine)
```

```
## [1] 310
```

```
data_spine_small <- data_spine[1:4,] ## Matrix operation
```

Data Frames

```
data_spine_small[,2] ## Matrix extraction by position
```

```
## [1] 22.55259 10.06099 22.21848 24.65288
```

```
data_spine_small[[2]] ## List extraction by position
```

```
## [1] 22.55259 10.06099 22.21848 24.65288
```

Data Frames

```
data_spine_small[['Pelvic.tilt']] ## Named list extraction
```

```
## [1] 22.55259 10.06099 22.21848 24.65288
```

```
data_spine_small[,'Pelvic.tilt'] ## Data frame named column extraction
```

```
## [1] 22.55259 10.06099 22.21848 24.65288
```

```
data_spine_small$Pelvic.tilt ## Dollar sign extraction
```

```
## [1] 22.55259 10.06099 22.21848 24.65288
```

Data Frames

My preference is for

1. *data frame named column extraction*

   ```
   data_spine_small[,'Pelvic.tilt'],
   ```

2. *named list extraction*

   ```
   data_spine_small['Pelvic.tilt']]
   ```

3. *Dollar-based extraction*

   ```
   data_spine_small$Pelvic.tilt
   ```

Data Frames

```
names(data_spine_small)
```

```
##  [1] "Pelvic.incidence"          "Pelvic.tilt"
##  [3] "Lumbar.lordosis.angle"     "Sacral.slope"
##  [5] "Pelvic.radius"             "Degree.spondylolisthesis"
##  [7] "Pelvic.slope"              "Direct.tilt"
##  [9] "Thoracic.slope"            "Cervical.tilt"
## [11] "Sacrum.angle"              "Scoliosis.slope"
## [13] "Class.attribute"
```

```
data_spine_small[,c('Pelvic.tilt', 'Pelvic.slope','Class.attribute')]
```

```
##    Pelvic.tilt Pelvic.slope Class.attribute
## 1     22.55259    0.7445035        Abnormal
## 2     10.06099    0.4151857        Abnormal
## 3     22.21848    0.4748892        Abnormal
## 4     24.65288    0.3693453        Abnormal
```

Filtering data frames

Boolean operators

Operator	Meaning
|	Or
&	And
!	Not

Filtering data frames

```
data_spine[data_spine$Pelvic.tilt > 20, ]
```

```
##     Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 1           63.02782    22.55259                39.60912     40.47523
## 3           68.83202    22.21848                50.09219     46.61354
## 4           69.29701    24.65288                44.31124     44.64413
## 14          53.57217    20.46083                33.10000     33.11134
## 15          57.30023    24.18888                47.00000     33.11134
## 17          63.83498    20.36251                54.55243     43.47247
## 22          54.91944    21.06233                42.20000     33.85711
## 23          63.07361    24.41380                54.00000     38.65981
## 25          36.12568    22.75875                29.00000     13.36693
## 26          54.12492    26.65049                35.32975     27.47443
## 29          44.55101    21.93115                26.78592     22.61986
## 30          66.87921    24.89200                49.27860     41.98721
## 35          59.59554    31.99824                46.56025     27.59730
## 39          55.84329    28.84745                47.69054     26.99584
## 44          66.28539    26.32784                47.50000     39.95755
## 46          50.91244    23.01517                47.00000     27.89727
```

```
subset(data_spine, Pelvic.tilt > 20) ## is equivalent
```

Filtering data frames

```
data_spine[data_spine$Pelvic.tilt > 20 & data_spine$Pelvic.slope > 0.85,
```

```
##     Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 26          54.12492    26.65049               35.32975     27.47443
## 76          70.22145    39.82272               68.11840     30.39873
## 84          81.10410    24.79417               77.88702     56.30993
## 99          77.65512    22.43295               93.89278     55.22217
## 106         65.00796    27.60261               50.94752     37.40536
## 112         84.99896    29.61010               83.35219     55.38886
## 129         90.51396    28.27250               69.81394     62.24146
## 179         80.65432    26.34438               60.89812     54.30994
## 231         65.61180    23.13792               62.58218     42.47388
## 303         54.60032    21.48897               29.36022     33.11134
##     Pelvic.radius Degree.spondylolisthesis Pelvic.slope Direct.tilt
## 26       121.4470                1.571205    0.9286879     14.6686
## 76       148.5256              145.378143    0.9466106     10.3840
## 84       151.8399               65.214616    0.9720056     10.5715
## 99       123.0557               61.211187    0.9249029     14.9502
## 106      116.5811                7.015978    0.8673241     12.1292
```

```
subset(data_spine, Pelvic.tilt > 20 & Pelvic.slope > 0.85)
```

Filtering data frames and selecting variables

```
data_spine[data_spine$Pelvic.tilt > 20 & data_spine$Pelvic.slope > 0.85,
           c('Direct.tilt', 'Class.attribute')]
```

```
##     Direct.tilt Class.attribute
## 26      14.6686        Abnormal
## 76      10.3840        Abnormal

## 84      10.5715        Abnormal
## 99      14.9502        Abnormal
## 106     12.1292        Abnormal
## 112      7.0551        Abnormal
## 129     13.5739        Abnormal
## 179     20.0845        Abnormal
## 231     30.0422          Normal
## 303     30.8554          Normal
```

Adding a variable

```
data_spine_small[,'bad.angle'] <- c('No','Yes','No','No')
data_spine_small
```

```
##   Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 1         63.02782    22.55259               39.60912     40.47523
## 2         39.05695    10.06099               25.01538     28.99596
## 3         68.83202    22.21848               50.09219     46.61354
## 4         69.29701    24.65288               44.31124     44.64413
##   Pelvic.radius Degree.spondylolisthesis Pelvic.slope Direct.tilt
## 1      98.67292                -0.254400    0.7445035     12.5661
## 2     114.40543                 4.564259    0.4151857     12.8874
## 3     105.98514                -3.530317    0.4748892     26.8343
## 4     101.86850                11.211523    0.3693453     23.5603
##   Thoracic.slope Cervical.tilt Sacrum.angle Scoliosis.slope
## 1        14.5386      15.30468    -28.65850         43.5123
## 2        17.5323      16.78486    -25.53061         16.1102
## 3        17.4861      16.65897    -29.03189         19.2221
## 4        12.7074      11.42447    -30.47025         18.8329
##   Class.attribute bad.angle
## 1        Abnormal        No
```

```
data_spine_small$bad.angle <- ...
data_spine_small[['bad.angle']] <- ...
```

Removing a variable

```
data_spine_small[, -c(13,14)]

data_spine_small[,-c('Class.attribute', 'bad.angle')]

## The next two commands change the original data set

data_spine_small[c('Class.attribute','bad.angle')] <- NULL



data_spine_small[['bad.angle']] <- NULL
```

Creating derived variables

```
data_spine_small$bad.angle <- ifelse(data_spine_small$Sacrum.angle > 80,
                                     'Yes','No')
```

Creating derived variables

```
data_spine_small <- transform(data_spine_small,
                              bad.angle = ifelse(Sacrum.angle > 80,
                                                 'Yes','No'))
```

For deriving multiple variables into a data frame

```
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1

## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

For deriving multiple variables into a data frame

```
mtcars <- transform(mtcars,
                    kmpg = mpg * 1.6, ## Numerical vector
                    low.mpg = ifelse(mpg < 16, 'Yes','No')  ## Factor
                    )
```

For deriving multiple variables into a data frame

```
str(mtcars)
```

```
## 'data.frame':    32 obs. of  13 variables:
##  $ mpg    : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl    : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp   : num  160 160 108 258 360 ...

##  $ hp     : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat   : num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt     : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec   : num  16.5 17 18.6 19.4 17 ...
##  $ vs     : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am     : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear   : num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb   : num  4 4 1 1 2 1 4 2 2 4 ...
##  $ kmpg   : num  33.6 33.6 36.5 34.2 29.9 ...
##  $ low.mpg: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 1 ...
```

Adding new data to a data frame

You can concatenate two data frames using `rbind` as long as the variable names and orders are the same

```
new_data = rbind(data_spine[1:4,], data_spine[c(8,22),])
new_data
```

```
##    Pelvic.incidence Pelvic.tilt Lumbar.lordosis.angle Sacral.slope
## 1          63.02782    22.55259               39.60912     40.47523
## 2          39.05695    10.06099               25.01538     28.99596
## 3          68.83202    22.21848               50.09219     46.61354
## 4          69.29701    24.65288               44.31124     44.64413
## 8          45.36675    10.75561               29.03835     34.61114
## 22         54.91944    21.06233               42.20000     33.85711
##    Pelvic.radius Degree.spondylolisthesis Pelvic.slope Direct.tilt
## 1       98.67292                -0.254400    0.7445035     12.5661
## 2      114.40543                 4.564259    0.4151857     12.8874
## 3      105.98514                -3.530317    0.4748892     26.8343
## 4      101.86850                11.211523    0.3693453     23.5603
## 8      117.27007               -10.675871    0.1319726     28.8165
## 22     125.21272                 2.432561    0.1752446     23.0791
##    Thoracic.slope Cervical.tilt Sacrum.angle Scoliosis.slope
## 1         14.5386      15.30468   -28.658501         43.5123
## 2         17.5323      16.78486   -25.530607         16.1102
## 3         17.4861      16.65897   -20.031888         19.2221
```

Adding new data to a data frame

You can add columns of a new data frame to an existing data frame using `cbind` as long as the columns have no common names

```
new_data2 <- cbind(data_spine[1:4, c('Pelvic.slope','Class.attribute')],
                   data.frame(Sex = c('M','F','M','M'),    ## Creating a
                              Race = c('W','B','As','B'))

                   )
new_data2
```

```
##   Pelvic.slope Class.attribute Sex Race
## 1    0.7445035        Abnormal   M    W
## 2    0.4151857        Abnormal   F    B
## 3    0.4748892        Abnormal   M   As
## 4    0.3693453        Abnormal   M    B
```