Reading Data

- `read.table`, `read.csv`, for reading tabular data
  - `file`, the name of the file, or a connection
  - `header`, logical indicating if the file ha a header
  - `sep`, a string indicating how the columns are separated.
  - `colClasses`, a character vector indicating the class of each column in the dataset
  - `nrows`, the number of rows in the dataset
  - `comment.char`, a character string indicating the comment character
  - `skip`, the number of lines to skip from the beginning
  - `stringsAsFactors`, should character variables be coded as factors?
- `readLines`, for reading lines of a text file
- `source`, for reading in R code files
- `dget`, for reading in R code files
- `load`, for reading in saved workspaces
- `unserialize`, for reading binary objects in R.

Writing Data

- `write.table`
- `writeLines`
- `dump`
- `dput`
- `save`
- `serialize`

Handling large datasets

Taking first 100 rows for better memory usage

```
initial<-read.table("datatable.txt", nrows=100)
classes<-sapply(initial,class)
tabAll<-read.table("datatable.txt",
                   `colClasses=classes)
```

dput-ing R Objects

```
y<- data.frame(a=1,b="a")
dput(y)
```

```
## structure(list(a = 1, b = structure(1L, .Label = "a", class = "factor")), class = "data.frame", row.r
## -1L))
```

```
dput(y, file="y.R")
new.y<-dget("y.R")
new.y
```

```
##   a b
## 1 1 a
```

Dumping R Objects

```
x<-"foo"
y<-data.frame(a=1,b="a")
dump(c("x","y"), file="data.R")
rm(x,y)     #this will remove x
source("data.R")  #this will again load x from source
x
```

```
## [1] "foo"
```