How does "RD" work?

Good question. Reading in data is done by what is essentially a cascade of scripts and functions (there are big differences between the two -- type 'help function' and 'help script' at the MATLAB command line to learn all about them), coordinated by "RD.M," a script that prompts the user to select a data file. RD looks at the filename extension (the letters following the '.') and runs the script appropriate to the type of data it thinks it is being told to load. There are scripts to load the following types of data:

- plain ASCII (.txt) -- must contain NOTHING but numerical data, arrayed in a rectangular matrix where each channel's data is in its own column.
- ober2 (.obr) -- NOT Orbit data. Use their utility to convert to Orbit to ASCII data.
- ASD (.asd) -- ASYST's (don't ask...) native data format.
- RTRV (.dat) -- private OMLAB format from the distant past.
- Lab (.lab) -- private OMLAB format generated by our LabVIEW acq software, based on the original RTRV format.
- Bin (.bin) -- 'float32' format data, arranged either contiguously or interleaved)

(WARNING: Just because you may have a file with one of these file extensions in the name DOES NOT MEAN that the information in the file is of the proper format for RD to read. Of the listed types, it is likely that you will only have ASCII, ober2, or bin files that can be read. '.dat' is a pretty generic file extension, and such a file could have be generated by any number of programs. Remember: garbage in --> garbage out.)

The called script will then look for bias adjust file (the name will be "adjbias_xxx.txt" where the "xxx" will be the series name of the data files -- usually the subject's initials) in the same directory as the data file being loaded. This bias adjust file contains additional information needed to load, offset and scale the data. (A default -- i.e containing no scaling/offset information -- version of this file is generated by "biasgen.m", an interactive function that will prompt the user to enter the necessary information, and should already exist before running "RD" -- right after finishing a data taking session is a good time to do this. Useful scaling/offset information will be generated by running "cal" later.)

Assuming all goes well, the bias file is read in, and its scaling values are applied to the data being read in. RD will display the progess, showing what channels have been found, adjusted and placed into memory. The acceptable channel names are: "lh" (LE hor), "rh" (RE hor), "lv" (LE vert) , "rv" (RE vert), "lt" (LE tors), "rt" (RE tors), "st" (Hor stim) and "sv" (Vert stim).

****

A geeky note to programmers: the data arrays (lh, rh, etc.) are declared to be global by RD (actually by a script it calls), so if you wish to write any analysis functions you should declare the data arrays global in your function, which is -- I believe -- a faster, less memory-intense method than passing these potentially huge arrays as arguments. If you disagree, fine. This is practically a religious question in some quarters...

****

Next, RD will search for post-processing directions (i.e. additional transformations to perform on the data) such as filtering commands (stored in '.f' files), re-zeroing commands (stored in '.z' files), unfolding commands (stored in '.u' files), and information about saccadic timing (stored in '.s' files). Samples of these files, with an explanation, are in the folder "sample post proc" in "tutorials".

These files are named by appending the appropriate extension to the base name (i.e. the name up to the '.') of the data file. If any of these files do not exist (and there is no requirement that they do), RD will mention it in passing, and continue.

The first time a data file is loaded, there will be no information about scaling and offset factors, so they will be "1"s and "0"s, respectively. (i.e. shift the data by zero, and multiply it by one.) You will probably want to calibrate the data at this time. This involves running "cal" and telling it how to shift and scale the data so that it corresponds to where the subject was known (or strongly suspected) to be looking during the ***MONOCULAR*** calibration records you took. (You DID take monocular calibration data, didn't you? For an explanation of why this is a good and necessary thing, and detailed instructions how to properly calibrate your subject, see "Cal instructions", which should be located in the same directory as this document.)

In a typical calibration record, the subject looks at known targets (e.g. 0, +15 deg, -15 deg). When you run "cal", your goal is to adjust the eye movement data to match the target data. "Cal" is so well written (!) and intuitive (!!) that if you pay attention to the prompts it gives you, this is a painless process. When "cal" is finished, it will display the offset and scaling factors in a form that you can copy and paste into the "adjbias.txt" file.

You can then type "clear all" at the MATLAB prompt, and then run "RD" again. It should then go through the sequence again, this time loading in and applying your calibration factors. Plot the data (you can use "plth" to display horizontal data, "plthv" to display horizontal and vertical data, or "plthvt" to display h/v/t data) to verify that the data is now properly calibrated.