

Using R to Find Correlations

Michael A. Covington
Institute for Artificial Intelligence
The University of Georgia

2011 December 9

Introduction

In what follows I will demonstrate statistical analysis of an experiment that looks for a correlation between two measurements on each of a set of texts, using Excel to edit and prepare the data and R to analyze it.

My “Using R” tutorials partly repeat each other’s content so that each one will be complete by itself.

About R

R is freeware, downloadable from www.r-project.org. It is good for calculation, matrix arithmetic, statistics, and various kinds of machine learning.

Suggested reading: Peter Dalgaard (2002) *Introductory Statistics with R*. Berlin: Springer.

R is derived from an earlier language called S. To learn more about its advanced statistical functions see Venables and Ripley, *Modern Applied Statistics with S* (which includes R). To learn more about R as a programming language, see Braun and Murdoch, *A First Course in Statistical Programming with R*.

R will mystify you until you learn something about its data types and syntax. The data types are:

Numbers

Strings

Vectors (1-dimensional arrays of numbers or strings)

Factors (vectors of discrete named values, like an array of enums in C)

Lists (sequences whose components are named, not numbered)

Matrices (2-dimensional arrays)

Arrays (with any number of dimensions)

Data frames (2-dimensional arrays with named columns, for statistical data)

The syntax of R is unusual. Note that:

There are no semicolons between statements.

Vectors and arrays are indexed from 1, not 0.

All names are case-sensitive.

`<-` is assignment

`.` within a name is just an ordinary character

`$` picks out parts of lists or frames (such as `a$b`, which means the part of `a` named `b`)

`#` makes the rest of the line a comment

`%` marks special operators, such as `/%` (integer division), `%%` (matrix multiplication)

Vectors (sequences) have to be created, usually with the function `c()`.

Compare:	In Prolog:	<code>X = [1,2,3]</code>
	In R:	<code>X <- c(1,2,3)</code>

The help system is accessed with commands such as `help(t.test)` (for finding out about the function named `t.test`).

About data files

Excel's own file formats, `.xls` and `.xlsx`, are generally not understood by other software. Instead, we exchange spreadsheets with other software by using two other formats, **tab-delimited text** and **comma-separated values (CSV)**. These are text files that use, respectively, tab characters (Unicode 0009) and commas to separate the columns.

R reads tab-delimited text with the function `read.delim()` and CSV with `read.csv()`. In each case what you get is a data frame with the first line used as column labels.

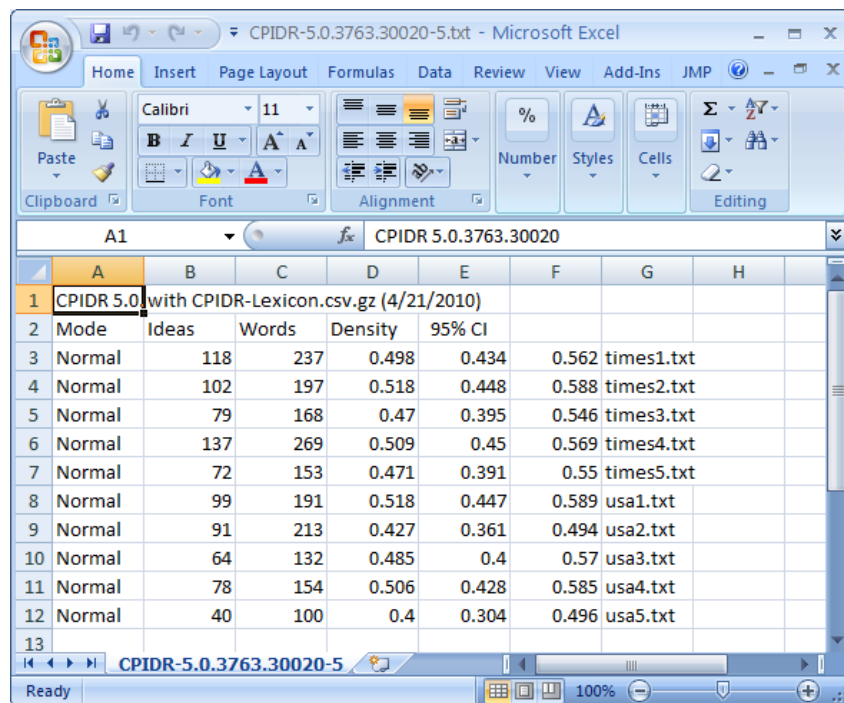
A sample experiment

I wanted to find out whether news stories with higher idea density also have higher vocabulary diversity. I took five stories each from *USA Today* and *The New York Times*. (This is not very

many!) I ran CPIDR to measure their idea density and MATTR to measure their vocabulary diversity over a 75-word window. (I couldn't use the default 500-word window because the shortest story was only 100 words long.)

Preparing the data file

CPIDR produced a tab-delimited text file. I opened it with Excel, and this is what I saw:



	A	B	C	D	E	F	G	H
1	CPIDR 5.0.3763.30020-5.txt - Microsoft Excel							
2	CPIDR 5.0.3763.30020							
3	Mode	Ideas	Words	Density	95% CI			
4	Normal	118	237	0.498	0.434	0.562	times1.txt	
5	Normal	102	197	0.518	0.448	0.588	times2.txt	
6	Normal	79	168	0.47	0.395	0.546	times3.txt	
7	Normal	137	269	0.509	0.45	0.569	times4.txt	
8	Normal	72	153	0.471	0.391	0.55	times5.txt	
9	Normal	99	191	0.518	0.447	0.589	usa1.txt	
10	Normal	91	213	0.427	0.361	0.494	usa2.txt	
11	Normal	64	132	0.485	0.4	0.57	usa3.txt	
12	Normal	78	154	0.506	0.428	0.585	usa4.txt	
13	Normal	40	100	0.4	0.304	0.496	usa5.txt	

The output from MATTR was also a tab-delimited text file, but it needed some cleaning up to remove warnings that the window was not much smaller than the text. (This casts some doubt on whether MATTR was making good measurements. But since this is just a statistics demonstration, I'll proceed.)

I opened the MATTR output file with Excel, deleted some error messages, and ended up with this:

	A	B	C	D	E	F	G	H
1	MATTR 2.0.3018.28419 2008/04/07							
2								
3	Words	Window	MATTR	Filename				
4	249	75	0.774	times1.txt				
5	201	75	0.811	times2.txt				
6	177	75	0.801	times3.txt				
7	271	75	0.791	times4.txt				
8	157	75	0.83	times5.txt				
9	198	75	0.816	usa1.txt				
10	216	75	0.76	usa2.txt				
11	136	75	0.788	usa3.txt				
12	155	75	0.785	usa4.txt				
13	105	75	0.757	usa5.txt				

Very important point: Both MATTR and CPIDR took the ten text files in the same order. If they had not done so, I would have had to do some sorting!

My next move was to copy the MATTR data and the CPIDR data into a single Excel spreadsheet. To prevent mixups I kept the filenames from both places. Also, I made sure **the top row is a set of column labels**. The result:

	A	B	C	D	E	F	G	H	I	J
1	Mode	Ideas	Words	Density	95% CI	95% CI2	Filename1	MATTR	Filename2	
2	Normal	118	237	0.498	0.434	0.562	times1.txt	0.774	times1.txt	
3	Normal	102	197	0.518	0.448	0.588	times2.txt	0.811	times2.txt	
4	Normal	79	168	0.47	0.395	0.546	times3.txt	0.801	times3.txt	
5	Normal	137	269	0.509	0.45	0.569	times4.txt	0.791	times4.txt	
6	Normal	72	153	0.471	0.391	0.55	times5.txt	0.83	times5.txt	
7	Normal	99	191	0.518	0.447	0.589	usa1.txt	0.816	usa1.txt	
8	Normal	91	213	0.427	0.361	0.494	usa2.txt	0.76	usa2.txt	
9	Normal	64	132	0.485	0.4	0.57	usa3.txt	0.788	usa3.txt	
10	Normal	78	154	0.506	0.428	0.585	usa4.txt	0.785	usa4.txt	
11	Normal	40	100	0.4	0.304	0.496	usa5.txt	0.757	usa5.txt	

I saved this on a **new file, as tab-delimited text**, taking care not to overwrite the original files.

The research question

We are interested in the columns called **Density** and **MATTR**. The question is, are these correlated? That is, does a higher value of one go with a higher value of the other?

Loading the data into R

This is how to read the whole data set into R and store it in a variable called **d**:

```
> d <- read.delim("c:\\Users\\mc\\Desktop\\correldata.txt")
```

Note the doubled backslashes. If you want R to put up an open file dialog box and let you pick a file, you can do it in this somewhat clumsy way:

```
> require(tcltk) # do this just once, to bring in the tcltk library
> d <- read.delim(tclvalue(tkgetOpenFile()))
```

To confirm that it was read in correctly, we can immediately display it:

```
> d
```

	Mode	Ideas	Words	Density	X95..CI	X95..CI2	Filename1	MATTR	Filename2
1	Normal	118	237	0.498	0.434	0.562	times1.txt	0.774	times1.txt
2	Normal	102	197	0.518	0.448	0.588	times2.txt	0.811	times2.txt
3	Normal	79	168	0.470	0.395	0.546	times3.txt	0.801	times3.txt
4	Normal	137	269	0.509	0.450	0.569	times4.txt	0.791	times4.txt
5	Normal	72	153	0.471	0.391	0.550	times5.txt	0.830	times5.txt
6	Normal	99	191	0.518	0.447	0.589	usa1.txt	0.816	usa1.txt
7	Normal	91	213	0.427	0.361	0.494	usa2.txt	0.760	usa2.txt
8	Normal	64	132	0.485	0.400	0.570	usa3.txt	0.788	usa3.txt
9	Normal	78	154	0.506	0.428	0.585	usa4.txt	0.785	usa4.txt
10	Normal	40	100	0.400	0.304	0.496	usa5.txt	0.757	usa5.txt

Note that two of the column labels have been changed a bit in order to make them valid names in R.

The columns we are interested in are **Density** and **MATTR**.

We can look at the individual columns as vectors (sequences of values):

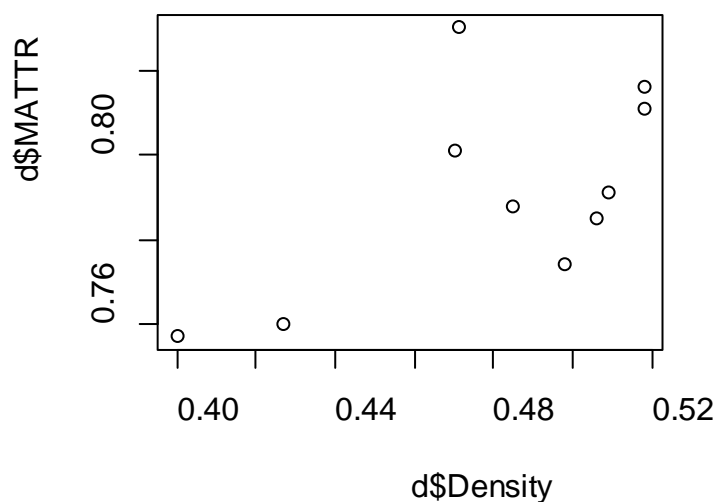
```
> d$Density
[1] 0.498 0.518 0.470 0.509 0.471 0.518 0.427 0.485 0.506 0.400
> d$MATTR
[1] 0.774 0.811 0.801 0.791 0.830 0.816 0.760 0.788 0.785 0.757
```

Scatterplot

The first thing we should do, when hunting for correlations, is to scatterplot one series of values against the other. In R, this is surprisingly easy to do:

```
> plot(d$Density,d$MATTR)
```

Here's what pops up in the graphics window:



It *does* look as though higher MATTR goes with higher Density. But how much confidence can we have that this isn't just an effect of random sampling? Go with your gut feeling here: **there are not enough dots to tell us quite what is going on.** That is why we will not get a significant p -value in the tests that follow.

Correlation testing

We can ask three things:

- *How closely* do increases in MATTR go with increases in Density? This is the *Pearson correlation coefficient* (r). (Note that some statistics packages give you r^2 .) If MATTR and Density are unrelated, it is 0. If they are on a line (so that one is a linear function of the other), it is 1.
- *How confident* should we be that the correlation is real, and not just a sampling effect? This is the p -value, the probability of seeing this much correlation or more just because of random sampling. The lower the p value, the better. In the biosciences we normally accept a correlation as real if $p < 0.05$.
- *How much* of an increase in MATTR goes with how much of an increase in Density? This is the slope of the regression line (m), assuming a linear relationship exists.

We can get the first two of these with one simple test:

```
> cor.test(d$Density,d$MATTR)

Pearson's product-moment correlation

data:  d$Density and d$MATTR
t = 2.1545, df = 8, p-value = 0.06333
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.03828379  0.89435827
sample estimates:
      cor
0.6059486
```

That is: $r = 0.6059486$, $p = 0.06333$.

Here p is not quite low enough to assure us that the correlation is real. The right thing to do is to run the same experiment with a larger sample, if possible.

Was that the right test?

Pearson's correlation coefficient assumes that the two populations have bell-shaped (normal) distributions and that the relationship between them, if any, is approximately a straight line. The Spearman "rho" (ρ) test drops both of these assumptions. **It will pick up curved relationships** (monotone increasing or decreasing), not just relationships that approximate a line. Here's how to do it:

```
> cor.test(d$Density,d$MATTR,method="spearman")
```

Spearman's rank correlation rho

```
data: d$Density and d$MATTR
S = 76.7321, p-value = 0.1111
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.5349569
```

Warning message:

```
In cor.test.default(d$Density, d$MATTR, method = "spearman") :
  Cannot compute exact p-values with ties
```

Not radically different – and the Spearman test objects to “ties” (examples with the same value of either variable) because it relies on being able to rank everything from highest to lowest.

A third alternative is Kendall's “tau” (τ) test, which, like Spearman's, does not assume a normal distribution or a linear relationship. Here it is:

```
> cor.test(d$Density,d$MATTR,method="kendall")
```

Kendall's rank correlation tau

```
data: d$Density and d$MATTR
z = 1.7961, p-value = 0.07249
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.4494666
```

Warning message:


```
In cor.test.default(d$Density, d$MATTR, method = "kendall") :
  Cannot compute exact p-value with ties
```

Like Spearman's test, Kendall's test objects to "ties."

Linear regression

Supposing there *is* a relationship between Density and MATTR, what is the slope of the line that best fits it?

We ask for a linear model of MATTR as a function of Density:

```
> lm(d$MATTR ~ d$Density)
```

Call:

```
lm(formula = d$MATTR ~ d$Density)
```

Coefficients:

(Intercept)	d\$Density
0.6166	0.3637

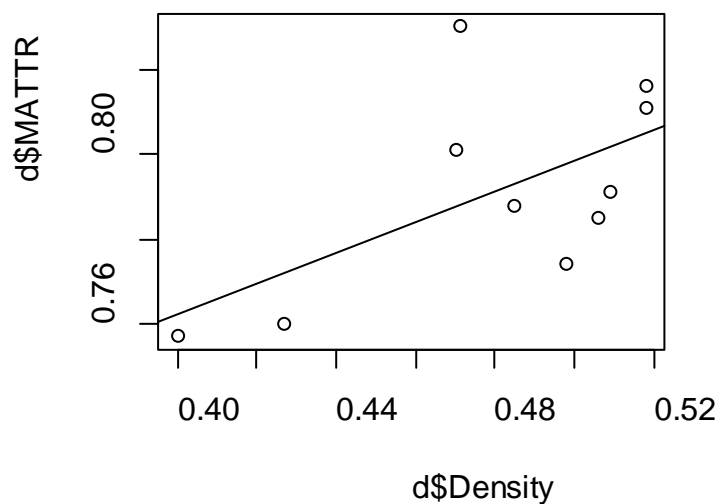
The slope is 0.3637. More fully, given a value of Density, our best guess of MATTR is

$$\text{MATTR} = 0.3637 \times \text{Density} + 0.6166$$

We can plot this line on the scatterplot by issuing two graphics commands:

```
> plot(d$Density, d$MATTR)
> abline(lm(d$MATTR ~ d$Density))
```

with this result:



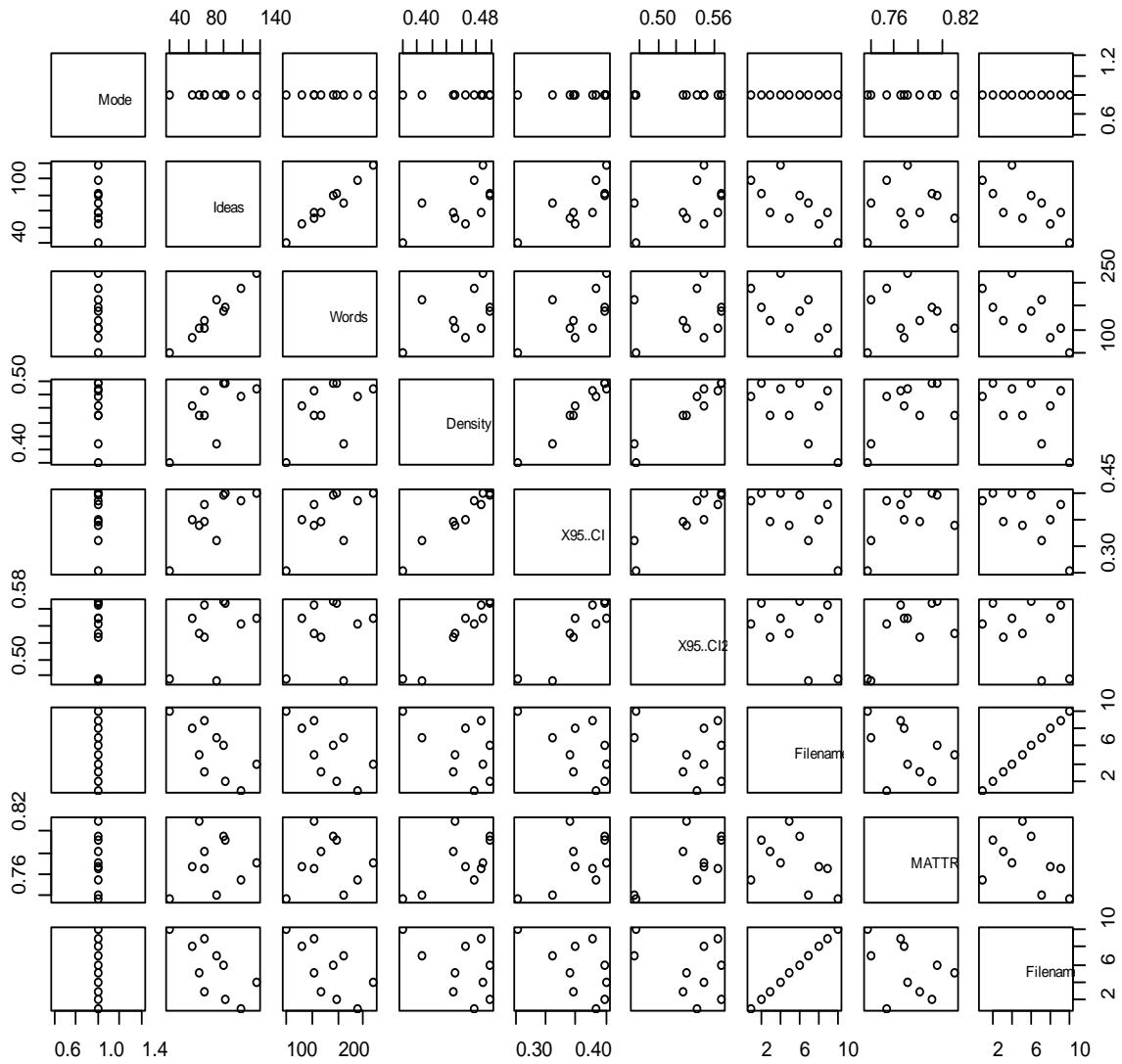
A word of sage advice: Do not become fascinated with linear regression as a way of predicting *everything*. It's better than random guessing or always predicting the average – but sometimes not much better.

Exploring a large data set with a scatterplot matrix

You can easily scatterplot *all* the columns in a data frame against each other. Do this:

```
> pairs(d)
```

You'll get a big graph:



Note that Mode has the same value everywhere, so it can only produce vertical or horizontal lines. Filename1 and Filename2 are matched (as well they should be!), so they produce perfect diagonal lines (showing no mixups). The rest of the scatterplots indicate correlations or lack thereof. Some of the correlations are uninteresting. For example, the total number of ideas in each text is closely correlated with the length of the text, and the confidence limits for density are closely correlated with density itself.