# Using A Script, Programs, and Pictures With the Anscombe Data

Thomas P Kline
University of Northern Iowa
Mathematics Department
Cedar Falls, Iowa
50614
thomas.kline@uni.edu

In1973, Professor Francis J. Anscombe published an article in the *American Statistician* in which he gave four sets of data that produce essentially the same results under linear regression and correlation. Professor Anscombe invented these data sets that allow statistics teachers to illustrate a need to visualize data prior to doing a linear regression.

Francis Anscombe died at age 83 in October of 2001 after a long teaching career that started at Cambridge in England and ended at Yale in the United States. He also taught at Princeton for seven years before moving to Yale in 1963.

Some students misuse the correlation coefficient (Pearson's r) in their analysis of bi-variate data. Professor Anscombe's four sets of bi-variate data can be used to help students understand that Pearson's r-value should be used in conjunction with other tools such as a scatter graph and residual analysis and should never be the sole determiner of linearity.

After years of using these data, I decided to "automate" the process. The use of programs, scripts and pictures can help any instructor automate the process of demonstrating the need for more than an r-value by leaving out the calculator steps needed to produce the linear regression equation, its graph, the value of the correlation coefficient as well as the coefficient of determination. The justification for this automation is that at this stage of instruction the purpose is to not concentrate on the details of doing a linear regression but rather to help wean students from or better yet to prevent students from developing an over reliance upon the correlation coefficient in their analysis of data.

This script uses 15 elementary programs and 10 pictures. Each picture uses 3097 bytes per picture so that all 10 pictures tie up 30970 bytes of RAM. If you are a heavy user of your Voyage200 or your 92Plus you may run out of RAM due to the amount of RAM demanded by pictures.

The Script that I am using at the present time:

```
C:cldeloff( )
  :List1={10,8,13,9,11,14,6,4,12,7,5}
  :List2={8.04,6.95,7.58,8.81,8.33,9.96,7.24,4.26,10.84,4.82,5.68}
C:LinReg  list1,list2:showstat
C:clup( )
  :List3={9.14,8.14,8.74,8.77,9.26,8.10,6.13,3.10,9.13,7.26,4.74}
C:LinReg  list1,list3:showStat
C:clup( )
  :List4={7.46,6.77,12.74,7.11,7.81,8.84,6.08,5.39,8.15,6.42,5.73}
C:LinReg  list1,list4:showstat
C:clup( )
  :List5={8,8,8,8,8,8,8,19,8,8,8}
  :List6={6.58,5.76,7.71,8.84,8.47,7.04,5.25,12.50,5.56,7.91,6.89}
C:LinReg  list5,list6:showstat
C:clup( )
C:all4( )
C:clup( )
C:slopa(11):anspics( )
C:clup( )
C:slopb(11):anpic2( )
C:clup( )
C:slopc(11):anpic3( )
C:clup( )
C:slopd(11):anspic4( )
C:clup( )
C:alted( )
C:clup( )
C:cldeloff( ):andatal( )
C:setGraph(“axes”,”off”)
C:steprs( )
C:clup( )
SetGraph(“axes”,”on”)
```

Programs used above:

**All4( )**

```
:Prgm
:PxlText “ r = “,10,4:PxlText string(rrr),10,70
:PxlText  “ r = “,25,4:PxlText string(raa),25,70
:PxlText “ r = “,40,4:PxlText  string(rmp),55,70
PxlText  “ r = “,55,4:PxlText  string(rpp),55,70
:Pause
:EndPrgm
```

```
:Alted( )

:Prgm
:RclPic altel
:Pause
:RclPic alte2
:Pause
:EndPrgm
```

```
:andata1( )
:Prgm
:clup( )
:DelVar x,y,d
:bginsc( )
:PlotsOff:FnOff
:LinReg list1,list2
:ShowStat
:Pause
:endsc( )
:EndPrgm
```

```
:anpic2( )
:Prgm
:ClrGraph
:RclPic  ansc2
:PxlText  “r= “,30,2:PxlText  string(raa),30,51

:Pause
:EndPrgm
```

```
:anpic3( )
:Prgm
:ClrGraph
:RclPic  anspic3
:PxlText  “r=  “,40,2:PxlText string(rmm),40,91
:Pause
:EndPrgm
```

```
:anspic4( )
:Prgm
:ClrGraph
:ClrDraw

:RclPic ansc4:Pxltext “r= “,30,2:PxlText string(rpp),30,41
:Pause
:EndPrgm
```

```
:anspics( )
:Prgm
:RclPic ansc1
:PxlText  “r=  “,30,2:PxlText string(rrr),30,91
:Pause
:EndPrgm
```

```
:cldeloff( )
:Prgm
:clup( )
:PlotsOff
:FnOff
:DelVar x,y,d
:setGraph(“axes”,”off”)
:EndPrg
```

```
:clup( )
:Prgm
:ClrDraw
:ClrGraph
:EndPrgm
```

**:slopa(d)**

```
:Prgm
:d=11
:sum(list1)→aaa
:sum(list2)→bbb
:list1*list2→ccc
:sum(ccc)→ddd
:list1^2→eee
:sum(eee)→fff
:list2^2→ggg
:sum(ggg)→ddd
(11*ddd-aaa*bbb)/(11*fff-aaa^2)→sss
:(bbb*fff – aaa*ddd)/(11*fff – aaa^2)→iii
:(11*ddd – aaa*bbb)/(√((11*fff – aaa^2)*(11*hhh – bbb^2)))→rrr
:EndPrgm
:slopb(d); slopc(d);slopd(11)
```

(similar to slopa but you use list3,etc. in place of list2 and use different variable names. In place of bbb I used baa but anything not already used will do.)

**:steprs( )**

```
:Prgm
:clup( )
:RclPic oner
:PxlText "Residual Plot for L1,L2",30,2
:Pause
:clup( )
:RclPic twor
:PxlText "Residual Plot for L1,L3",30,2
:Pause
:clup( )
:RclPic threr
:PxlText "Residual Plot for L1,L4",30,2
:Pause
:clup( )
:RclPic fourr
:PxlText "Residual Plot for L5,L6",30,42
:Pause
:clup( )
:EndPrgm
```

Note: "oner","twor", "threr", "fourr" are my names for stored pictures of the residual plots that I had worked out in advance and then stored as "pics."