

MINITAB Macro for Statistical Comparison of Linear Regressions

By

Xuejun Dong, Ph.D.

Distributed by

 *InfoClearinghouse.com*

©2005 Xuejun Dong
All Rights Reserved

Download at InfoClearinghouse.com

```

gmacro
lin_com
# This macro is to compare linear regressions
# (1) compare overall differences for k1 linear regression lines in
# terms of slopes and y-intercepts.
# (2) If some differences exist, conduct multiple comparisons
# (Tukey test) to detect which pairs are different.
# (3) To calculate 95% confidence intervals for slopes, Y-intercepts,
# Y-predictions corresponding to arbitrarily chosen X-values, or all of them
# at once.

# By Xuejun Dong, Ph.D
# North Dakota State University
# Central Grasslands Research Center
# Streeter, ND 58483
# Dec 2, 2005

#####
#                                     Purpose                                     #
#####
# Statistically comparing slopes and elevations of more than 2 regression lines #
# is not readily available from the standard output of statistical packages, but is #
# useful in research. Using MINITAB I compiled a recipe from Jerrold H. Zar (1984): #
# Bio-statistical Analysis (2nd Edition), Prentice Hall, Englewood Cliffs, New Jersey. #
#####

#####
#                                     Input requirements (read carefully)         #
#####
# (1) This macro assumes we have 2 to 17 regression lines to be compared, for situations #
# of having more than 17 lines, we will need to modify the macro a little. #
# (2) The x and y data for the lines are stored in consecutive columns from c1(x)-c2(y) #
# (first line), c3(x)-c4(y)(second line) and so on, until c33(x)-c34(y) (17th line). #
# (3) Assume we have already run the linear regressions from the interactive MINITAB, #
# and the results of Y-intercept, slope, r_sq, and p value for each of the regression #
# lines are statistically meaningful. Then enter outputs into c37-c40: c37(Y-intercept),#
# c38(slope), c39(r_sq), c40(p value). Use the following format to enter the data: the #
# first row of c37-c40 contains the Y-intercept, slope, r_sq and p value, respectively, #
# of the first regression line; the second row of columns c37-c40 contains results #
# for the second regression line,...., the 17th row of c37-c40 contains results for the #
# last (17th) regression line. Do not skip rows when entering data. #
# (4) In C36, enter the code (unique name) for each of the regression lines, beginning with #
# row 1 and ending with row 17 (if you have 17 regression lines). The codes entered in #
# this column (c36) will be used to identify the regressions lines in the outputs of the#
# macro. #
#####

#####
#                                     Optional Inputs                             #
#####
# If we want to calculate the 95% CI's for Y-predictions corresponding #
# to an arbitrarily chosen X value for each of the regression lines, we must enter the #
# selected X-values for each of the lines in the rows of c81 (in the order they appear #
# in c1-c34). #
#####

#####
#                                     Output results                             #
#####
# (1) Table of sums of squares will be stored in c42-c47 #
# (2) The n value (number of data points for each regression) will be stored in c41 #
# (3) The F statistic for comparing the overall differences in slopes will be stored in #
# the first cell of c48 (F_slope); and F statistic for comparing the elevations will be #
# stored in the first cell fo c49 (F_ele). You need to consult F table to see if these #
# values are statistically significant. #
# (4) The results of multiple comparisons for pairs of slopes will be stored in c52-c58. #

```

```

# The results for multiple comparisons for pairs of elevations will be stored in #
# c62-c68. Ignore the first row of these columns (they are used just to make the macro #
# run ok). If a pair of slope or elevation are statistically different, they are #
# labeled as "NEq" (Not Equal); if they are not different, a label of "Eq" (Equal) is #
# used. #
# (5) Results of 95% confidence intervals will be stored in c70-c81. #

#####

#####
# How to use it #
#####
# (1) MINITAB's spreadsheets are similar as those in EXCEL, and MINITAB's macro language is #
# similar to a high level programming language, such as TRUE BASIC. So, MINITAB has both #
# the advantage of spreadsheet softwares and high level programming languages. Sometimes, #
# using MINITAB can get the work done efficiently. #
# (2) TO use this macro, we first save this file in text format in our computer; then we #
# start MINITAB; in MINITAB environment, we enter our data (or read directly from #
# a lotus or excel file) into c1-c34 (assume we have a maximum of 17 regression lines to be #
# compared. See above "Input requirements" for how data is required to be stored in these #
# columns. Also, for how to enter other related information we get from standard #
# regression output. #
# (3) Click "session window" in MINITAB, which is the window to accept commands. In this #
# session window, if we don't see "MTB>" prmpt followed by a blinking cirSOR, click #
# "Editor/enable commands", so the prompt "MTB>" is displayed. #
# (4) Make sure the blinking cirSOR is right after the "MTB>" prompt. Assume this macro #
# file is stored at A Drive using the name "lin_com.txt". We enter this command: #
# #
# %'a:lin_com.txt' #
# #
# followed by Return. This will evoke the macro. During the macro run, there will be several#
# times of prompting us to enter some data depending on what comparisons we want to make. #
# (5) Last, any information (text or numeric) that begins after the "#" sign is considered #
# by MINITAB as annotations, and is ignored when compiling the main macro and #
# subroutines. #
#####

#####
# Description of output columns #
#####
# C41(n): data points corresponding to a regression line #
# C42(sum_xsq): sum of squares for the x variable #
# C43(sum_xy): sum of the x-y cross products #
# C44(sum_ysq): sum of squares for the y variable #
# C45(Resid_SS): Residual sum of squares #
# C46(Resid_DF): Residual Degree of freedom #
# c47(Type): Description of the SS type #
# c48(F_Slop): F value for the overall difference of the slopes #
# #
# c49(F_elev): F value for the overall difference of the elevations #
# c52(Line_1_slp): slope from a first line for comparison #
# c53(Line_2_slp): slope from a second line for comparison #
# c54(diff_slp): Difference between two slops #
# #
# c55(S.E._slp): Standard Error for the slopes #
# #
# c56(q_slp): q value for the slopes #
# c57(q_crit_slp): critical q value for the slopes #
# c58 (Eq_NEq_slp): Difference between two slopes ("Eq"="not different"; "NEq"="different") #
# #
# c62(Line_1_ele): elevation from a first line for comparison #
# #
# c63(Line_2_ele): elevation from a second line for comparison #
# #
# c64(diff_ele): Difference between two elevations #
# c65(S.E._ele): Standard Error for the elevations #
# c66(q_ele): q value for the elevations #

```

```

# c67(q_crit_ele): critical q value for the elevations
#
# c68(Eq_NEq_ele): Difference between two elevations ("Eq"="not different"; "NEq"="different")
#
# c70(code_for_CI): code of regression lines for the 95% confidence intervals
#
# c72(slp_low): lower bound 95% CI for a slope
#
# c73(slp_mean): mean value for a slope
#
# c74(slp_high): high bound 95% CI for a slope
#
# c75(ele_low): lower bound 95% CI for an elevation
#
# c76(ele_mean): mean value for an elevation
#
# c77(ele_high): high bound 95% CI for an elevation
#
# c78(obs_y_low): lower bound 95% CI for an arbitrary y value
#
# c79(ob_y_mean): mean value for an arbitrary y value
#
# c80(obs_y_high): high bound 95% CI for an arbitrary y value
#
#####

# First construct q distribution table (will be stored in C301-C317)
# and t distribution table (will be stored in C400-C401)
call q_table
call t_table

Note How many regression lines do you want to compare?
Note Enter the number and hit the Return key.
Note
Set C90;
  FILE "TERMINAL";
  NOBS 1.
COPY C90 k90
erase c90
copy k90 k1 # here we have k1 lines to be compared
              # k1 remains unchanged throughout the macro

Note
Note
Note Enter 1 to construct the sums of squares table.
Note Enter 0 if this table has already been constructed.
Set C90;
  FILE "TERMINAL";
  NOBS 1.
COPY C90 k90
erase c90

IF k90=1
  call sums_table
ELSE
  let k2=k1+1 # restore correct value of k2 that will be used throughout the macro
  let k3=k1+2 # restore correct value of k3 that will be used throughout the macro
  let k4=k1+3 # restore correct value of k4 that will be used throughout the macro
  goto 1
ENDIF
MLABEL 1

# Use the n values (c41) to look up t critical values
# from the t distribution table (C400-C401) and enter the results in c71
call t_lookup

Note
Note
Note
Note Enter 1 for comparing overall differences in slopes only.
Note Enter 2 for comparing overall differences in elevations only.
Note Enter 3 for comparing both overall slopes and elevations.
Note To skip this part, press 0.
Set C90;
  FILE "TERMINAL";

```

```
NOBS 1.
COPY C90 k90
erase c90
```

```
IF k90=1
  call over_slp
ELSEIF k90=2
  call over_ele
ELSEIF k90=3
  call over_both
ELSE goto 2
ENDIF
MLABEL 2
```

```
Note
Note
Note
```

```
Note Enter 1 for multiple comparisons of slopes.
Note Enter 2 for multiple comparisons of elevations.
Note Enter 3 for multiple comparisons for both slops and elevations
Note Enter 0 to skip this part
Set C90;
```

```
FILE "TERMINAL";
NOBS 1.
COPY C90 k90
erase c90
```

```
IF k90=1
  call mult_slp
ELSEIF k90=2
  call mult_ele
ELSEIF k90=3
  call mult_slp
  call mult_ele
ELSE
  goto 5
ENDIF
```

```
MLABEL 5
```

```
Note
Note
Note
```

```
Note Enter 1 for 95% confidence of intervals for slopes.
Note Enter 2 for 95% confidence of intervals for elevations.
Note Enter 3 for 95% confidence of intervals for arbitrary y's.
Note Enter 4 for 95% CI's for slopes, elevations and arbitrary y's.
Note Enter 0 if you want to exit
Set C90;
```

```
FILE "TERMINAL";
NOBS 1.
COPY C90 k90
erase c90
```

```
IF k90=1
  call CI_slp
ELSEIF k90=2
  call CI_ele
ELSEIF k90=3
  call CI_obs
ELSEIF k90=4
  call CI_slp
  call CI_ele
  call CI_obs
ELSE
  call ending
  exit
```

```
ENDIF
call ending
ENDMACRO
# end of main macro
```

```

# Subroutine 1: Construction of the sums of squares (SS) table
gmacro
sums_table
# k1 is not changed once determined at the beginning of the macro
# k2, k3 and k4 are scratch variables used repeatedly just in this subroutine; but
# in other parts of the macro, k2, k3 and k4 are not changed in value.
# k90 is scratch variable used throughout.

name c41 'n'
do k2=1:k1
  let k3=2*k2
  let c41(k2)=count(ck3)
enddo

name c42 'sum_xsq' # sum of x squared
name c43 'sum_xy' # sum of cross products
name c44 'sum_ysq' # sum of y squared
name c45 'Resid_SS' # Residual SS
name c46 'Resid_DF' # Residual Degrees of freedom
name c47 'Type'
do k2=1:k1
  let k3=2*k2-1
  let k4=2*k2
  let c42(k2)=sum(ck3*ck3)-(sum(ck3)*sum(ck3))/c41(k2)
  let c43(k2)=sum(ck3*ck4)-(sum(ck3)*sum(ck4))/c41(k2)
  let c44(k2)=sum(ck4*ck4)-(sum(ck4)*sum(ck4))/c41(k2)
  let c45(k2)=c44(k2)-c43(k2)*c43(k2)/c42(k2)
  let c46(k2)=c41(k2)-2
  let c199(1)="Line"
  let c200(1)=k2
  Text C200 c200 # MTB's unique way to change a text column to numeric
  Concatenate C199 C200 c198
  Numeric c200 c200
  let c47(k2)=c198(1)
enddo

erase c198-c200 # erase scratch variables

# next fill the last three rows of the sums_table

let k2=k1+1
let k3=k1+2
let k4=k1+3

let k90=sum(c45) # k90: scratch variable
let c45(k2)=k90

let k90=sum(c46)
let c46(k2)=k90
let c47(k2)="Pooled"

let k90=sum(c42)
let c42(k3)=k90

let k90=sum(c43)
let c43(k3)=k90

let k90=sum(c44)
let c44(k3)=k90

let c45(k3)=c44(k3)-c43(k3)*c43(k3)/c42(k3)
let c46(k3)=sum(c41)-k1-1
let c47(k3)="Common"
erase c90-c91 # make sure c90-c91 are ready for storing the stacked data
set c90
0
end
set c91

```

```

0
end
do k2=1:k1
  let k3=2*k2-1
  let k4=2*k2
  let k90=c41(k2)
  do k5=1:k90
    let k6=count(c90)+1
    let c90(k6)=ck3(k5)
    let c91(k6)=ck4(k5)
  enddo
enddo
erase k5-k6
let k2=k1+1 # onced changed here, will remain unchanged later on
let k3=k1+2 # onced changed here, will remain unchanged later on
let k4=k1+3 # onced changed here, will remain unchanged later on

let k90=sum(c41) # using this scratch variable
let c42(k4)=sum(c90*c90)-(sum(c90)*sum(c90))/k90
let c43(k4)=sum(c90*c91)-(sum(c90)*sum(c91))/k90
let c44(k4)=sum(c91*c91)-(sum(c91)*sum(c91))/k90
let c45(k4)=c44(k4)-c43(k4)*c43(k4)/c42(k4)
let c46(k4)=k90-2
let c47(k4)="Total"
  erase c90-c91
endmacro

# Subroutine 2: Overall difference in slops
gmacro
over_slp
let c48(1)=(c45(k3)-c45(k2))*c46(k2)/((k1-1)*c45(k2))
name c48 'F_slope'
endmacro

# Subroutine 3: Overall difference in elevations
gmacro
over_ele
let c49(1)=(c45(k4)-c45(k3))*c46(k3)/((k1-1)*c45(k3))
name c49 'F_elev'
endmacro

# Subroutine 4: Overall difference in both slopes and elevations
gmacro
over_both
let c48(1)=(c45(k3)-c45(k2))*c46(k2)/((k1-1)*c45(k2))
name c48 'F_slope'
let c49(1)=(c45(k4)-c45(k3))*c46(k3)/((k1-1)*c45(k3))
name c49 'F_elev'
endmacro

# Subroutine 5: Multiple comparisons for slopes
gmacro
mult_slp
# set missing value codes for columns c50-c56
# so that we can use "count" function to determine current number
# of rows the column contains during the macro run
do k5=52:58
  if k5=52 or k5=53
    let ck5(1)="a" # a way to assign a text value to a cell (also format the column)
  elseif k5=58
    let ck5(1)="a"
  else
    set ck5
  *
  end
endif
enddo
name c52 'line_1_slp'
name c53 'line_2_slp'
name c54 'differ_slp'
name c55 'S.E._slp'

```

```

name c56 'q_slp'
name c57 'q_crit_slp'
name c58 'Eq_NEq_slp'

# Look up critical q(0.05,DFp,k1) for slopes comparison
# using the values of k1 and DFp (pooled degree of freedom)
let k90=c46(k2) # recall k2=k1+1 throughout
name k90 'DFp' # DFp (pooled DF)
# k90, a scratch variable used often in the macro, contains the value of DFp only in here!

call q_lookup # look up the appropriate q value from the q distribution table

do k5=1:k1-1
  do k6=k5+1:k1+1
    if k6=k1+1
      break
    elseif k5=-1 or k6=0
      break
    else
      goto 3
    endif
  mlabel 3
  let k90=c45(k2)/c46(k2) # (s_Y.X^2)p
  let k91=sqrt(k90/2*(1/c42(k5)+1/c42(k6))) #SE
  let k92=(c38(k6)-c38(k5))/k91
  let k7=count(c52)+1
  let c52(k7)=c36(k5)
  let c53(k7)=c36(k6)
  let c54(k7)=c38(k6)-c38(k5)
  let c55(k7)=k91
  let c56(k7)=k92
  let c57(k7)=k93
  if c56(k7)>c57(k7)
    let c58(k7)="NEq" # use double quotation to assign text data to a cell
  else
    let c58(k7)="Eq"
  endif
enddo
enddo
erase k7 k90-k93
endmacro

# Subroutine 6: Multiple comparisons for elevations
gmacro
mult_ele
# for multiple comparisons (elevation or Y-intercepts)
# set missing value codes for columns c60-c66
# so that we can use "count" function to determine current number
# of rows the column contains during the macro run
do k5=62:68
  if k5=62 or k5=63
    let ck5(1)="a" # a way to assign a text value to a cell (also format the column)
  elseif k5=68
    let ck5(1)="a"
  else
    set ck5
    *
  end
endif
enddo
name c62 'line_1_ele'
name c63 'line_2_ele'
name c64 'differ_ele'
name c65 'S.E._ele'
name c66 'q_ele'
name c67 'q_crit_ele'
name c68 'Eq_NEq_ele'

# Look up critical q(0.05,DFp,k1) for Y-intercepts comparison
# using the values of k1 and DFp (pooled degree of freedom)

```

```

let k90=c46(k2)          # k90, a scratch variable, contains the value of DFp only in here!
name k90 'DFp'          # DFp (pooled DF)

call q_lookup # look up the appropriate q value from the q distribution table

let k90=c45(k3)/c46(k3)          #(s_Y.X^2)c
let k94=c43(k3)/c42(k3)          #bc (Equation 18.31 in Zar's book)
let k8=round(k1/2)
do k5=1:k1-1
  do k6=k5+1:k1+1
    if k6=k1+1
      break
    elseif k5=-1 or k6=0
      break
    else
      goto 4
    endif
  mlabel 4
  let k9=2*k5-1
  let k10=2*k5
  let k11=2*k6-1
  let k12=2*k6
  let k100=mean(ck9)          #X-bar,A
  let k101=mean(ck10)        #Y-bar,A
  let k102=mean(ck11)        #X-bar,B
  let k103=mean(ck12)        #Y-bar,B
  let k104=(k100-k102)*(k100-k102)/(c42(k5)+c42(k6)) # part of next equation.
  let k91=sqrt(k90/2*(1/c41(k5)+1/c41(k6)+k104)) #SE
  let k92=abs((k101-k103)-k94*(k100-k102))/k91 #q
  let k7=count(c62)+1
  let c62(k7)=c36(k5)
  let c63(k7)=c36(k6)
  let c64(k7)=c37(k6)-c37(k5)
  let c65(k7)=k91
  let c66(k7)=k92
  let c67(k7)=k93
  if c66(k7)>c67(k7)
    let c68(k7)="NEq"          # use double quotation to assign text to a cell
  else
    let c68(k7)="Eq"
  endif
endif
enddo
enddo
erase k7-k12
erase k90-k94
erase k100-k104
endmacro

# Subroutine 7: 95% CI's for slopes
gmacro
CI_slp
name c70 'code_for_CI'
name c72 'slp_low'
name c73 'slp_mean'
name c74 'slp_high'
do k5=1:k1
  let k90=c45(k5)/c46(k5)
  let k91=sqrt(k90/c42(k5))
  let c72(k5)=c38(k5)-c71(k5)*k91
  let c74(k5)=c38(k5)+c71(k5)*k91
  let c70(k5)=c36(k5)
  let c73(k5)=c38(k5)
endif
enddo
erase k5 k90 k91
endmacro

# Subroutine 8: 95% CI's for elevations
gmacro
CI_ele
name c70 'code_for_CI'
name c75 'ele_low'

```

```

name c76 'ele_mean'
name c77 'ele_high'
do k5=1:k1
  let k6=2*k5-1
  let k90=c45(k5)/c46(k5)
  let k91=sqrt(k90*(1/c41(k5)+mean(ck6)*mean(ck6)/c42(k5)))
  let c70(k5)=c36(k5)
  let c75(k5)=c37(k5)-c71(k5)*k91
  let c77(k5)=c37(k5)+c71(k5)*k91
  let c76(k5)=c37(k5)
enddo
erase k5-k6 k90-k91
endmacro

# Subroutine 9: 95% CI's for an possible observation
gmacro
CI_obs
name c70 'code_for_CI'
name c78 'obs_y_low'
name c79 'obs_y_mean'
name c80 'obs_y_high'
name c81 'Arbitrary_x_value'
do k5=1:k1
  let k6=2*k5-1
  let k90=c45(k5)/c46(k5)
  let k91=sqrt(k90*(1/c41(k5)+(c81(k5)-mean(ck6))*(c81(k5)-mean(ck6))/c42(k5)))
  let c70(k5)=c36(k5)
  let c78(k5)=c37(k5)+c38(k5)*c81(k5)-c71(k5)*k91
  let c80(k5)=c37(k5)+c38(k5)*c81(k5)+c71(k5)*k91
  let c79(k5)=c37(k5)+c38(k5)*c81(k5)
enddo
erase k5-k6 k90-k91
endmacro

# Subroutine 10: q-distribution table
# Do not alter the contents of this subroutine unless you fully understand
# this macro as a whole.
gmacro
q_table
# These are excerpts from Table B.5 of Zar (1984):Biostatistical Analysis (2nd Edition),
# Prentice Hall, Englewood Cliffs, New Jersey. page 528.
set c301 # v in Table B.5
1:20 24 30 40 60 120 1000000
end

# k(or p)=2 through 17 in Table B.5
set c302
17.97 6.085 4.501 3.927 3.635 3.461 3.344 3.261 3.199 3.151 3.113 3.082 3.055 3.033 3.014
2.998 2.984 2.971 2.960 2.950 2.919 2.888 2.858 2.829 2.800 2.772
end
set c303
26.98 8.331 5.910 5.040 4.602 4.339 4.165 4.041 3.949 3.877 3.820 3.773 3.735 3.702 3.674
3.649 3.628 3.609 3.593 3.578 3.532 3.486 3.442 3.399 3.356 3.314
end
set c304
32.82 9.798 6.825 5.757 5.218 4.896 4.681 4.529 4.415 4.327 4.256 4.199 4.151 4.111 4.076
4.046 4.020 3.997 3.977 3.958 3.901 3.845 3.791 3.737 3.685 3.633
end
set c305
37.08 10.88 7.502 6.287 5.673 5.305 5.060 4.886 4.756 4.654 4.574 4.508 4.453 4.407 4.367
4.333 4.303 4.277 4.253 4.232 4.166 4.102 4.039 3.977 3.917 3.858
end
set c306
40.41 11.74 8.037 6.707 6.033 5.628 5.359 5.167 5.024 4.912 4.823 4.751 4.690 4.639 4.595
4.557 4.524 4.495 4.469 4.445 4.373 4.302 4.232 4.163 4.096 4.030
end
set c307
43.12 12.44 8.478 7.053 6.330 5.895 5.606 5.399 5.244 5.124 5.028 4.950 4.885 4.829 4.782

```

```

4.741 4.705 4.673 4.645 4.620 4.541 4.464 4.389 4.314 4.241 4.170
end
set c308
45.40 13.03 8.853 7.347 6.582 6.122 5.815 5.597 5.432 5.305 5.202 5.119 5.049 4.990 4.940
4.897 4.858 4.824 4.794 4.768 4.684 4.602 4.521 4.441 4.363 4.286
end
set c309
47.36 13.54 9.177 7.602 6.802 6.319 5.998 5.767 5.595 5.461 5.353 5.265 5.192 5.131 5.077
5.031 4.991 4.956 4.924 4.896 4.807 4.720 4.635 4.550 4.468 4.387
end
set c310
49.07 13.99 9.462 7.826 6.995 6.493 6.158 5.918 5.739 5.599 5.487 5.395 5.318 5.254 5.198
5.150 5.108 5.071 5.038 5.008 4.915 4.824 4.735 4.646 4.560 4.474
end
set c311
50.59 14.39 9.717 8.027 7.168 6.649 6.302 6.054 5.867 5.722 5.605 5.511 5.431 5.364 5.306
5.256 5.212 5.174 5.140 5.108 5.012 4.917 4.824 4.732 4.641 4.552
end
set c312
51.96 14.75 9.946 8.208 7.324 6.789 6.431 6.175 5.983 5.833 5.713 5.615 5.533 5.463 5.404
5.352 5.307 5.267 5.231 5.199 5.099 5.001 4.904 4.808 4.714 4.622
end
set c313
53.20 15.08 10.05 8.373 7.466 6.917 6.550 6.287 6.089 5.935 5.811 5.710 5.625 5.554 5.493
5.439 5.392 5.352 5.315 5.282 5.179 5.077 4.977 4.878 4.781 4.685
end
set c314
54.33 15.38 10.35 8.525 7.596 7.034 6.658 6.389 6.186 6.028 5.901 5.798 5.711 5.637 5.574
5.520 5.471 5.429 5.391 5.357 5.251 5.147 5.044 4.942 4.842 4.743
end
set c315
55.36 15.65 10.53 8.664 7.717 7.143 6.759 6.483 6.276 6.114 5.984 5.878 5.789 5.714 5.649
5.593 5.544 5.501 5.462 5.427 5.319 5.211 5.106 5.001 4.898 4.796
end
set c316
56.32 15.91 10.69 8.794 7.828 7.244 6.852 6.571 6.359 6.194 6.062 5.953 5.862 5.786 5.720
5.662 5.612 5.568 5.528 5.493 5.381 5.271 5.163 5.056 4.950 4.845
end
set c317
57.22 16.14 10.84 8.914 7.932 7.338 6.939 6.653 6.437 6.269 6.134 6.023 5.931 5.852 5.785
5.727 5.675 5.630 5.589 5.553 5.439 5.327 5.216 5.107 4.998 4.891
end
endmacro

```

```

# Subroutine 11: t-distribution table.
# Do not alter the contents of this subroutine unless you fully understand
# this macro as a whole.
gmacro
t_table
# This are the excerpts of Table B.3 of Zar (1984):Bio-statistical Analysis (2nd Edition),
# Prentice Hall, Englewood Cliffs, New Jersey. pages 484-485.
set c400 # v in Table B.3
1:50
52:100/2
105:150/5
160:200/10
250:500/50
600:1000/100
1000000
end

set c401 # alpha92)=0.05 in Table B.3
12.706 4.303 3.182 2.776 2.571 2.447 2.365 2.306 2.262 2.228 2.201 2.179 2.160 2.145 2.131
2.120 2.110 2.101 2.093 2.086 2.080 2.074 2.069 2.064 2.060 2.056 2.052 2.048 2.045 2.042
2.040 2.037 2.035 2.032 2.030 2.028 2.026 2.024 2.023 2.021 2.020 2.018 2.017 2.015 2.014
2.013 2.012 2.011 2.010 2.009

2.007 2.005 2.003 2.002 2.000 1.999 1.998 1.997 1.995 1.994 1.993 1.993 1.992 1.991 1.990
1.989 1.989 1.988 1.987 1.987 1.986 1.986 1.985 1.984 1.984 1.983 1.982 1.981 1.980 1.979
1.978 1.978 1.977 1.976 1.976 1.975 1.974 1.973 1.973 1.972 1.969 1.968 1.967 1.966 1.965
1.965 1.964 1.963 1.963 1.963 1.962 1.960

```

```
end
endmacro
```

```
# Subroutine 12: Look up t values corresponding to each regression dataset.
# This will be used to calculate the 95% confidence intervals.
```

```
gmacro
t_lookup

let k102=count(c400)
do k100=1:k1
  let k101=c41(k100)-2
  do k103=1:k102
    let k104=c400(k103)-k101
    if k104=0
      let c71(k100)=c401(k103)
      goto 100
    elseif k104>0
      let k103=k103-1
      let c71(k100)=c401(k103)      # If df is not in the table, use a conservative df.
      goto 100
    else
      next
    endif
  enddo
mlabel 100
enddo
erase k100-k104
name c71 't_critic'
endmacro
```

```
# Subroutine 13: To look up q values from q-distribution table.
# This will be used for the multiple comparisons of slopes and elevations.
```

```
gmacro
q_lookup

let k300=k1+300
let k102=count(c301)
do k103=1:k102
  let k104=c301(k103)-k90
  if k104=0
    let k93=ck300(k103)
    goto 100
  elseif k104>0
    let k103=k103-1
    let k93=ck300(k103)      # If df is not in the table, use a conservative df.
    goto 100
  else
    next
  endif
enddo
mlabel 100
erase k102-k104 k300
name k93 'q_critic'
endmacro
```

```
# subroutine 14: ending of macro
```

```
gmacro
ending
```

```
erase c301-c317 c400-c401
erase k1-k4
```

```
Note
Note
```

```
Note Description of output results:
```

```
Note
```

```
Note C41(n): Data points corresponding to a regression line.
```

```
Note C42(sum_xsq): Sum of squares for the x variable.
```

```
Note C43(sum_xy): Sum of the x-y cross products.
```

```
Note C44(sum_ysq): Sum of squares for the y variable.
```

Note C45(Resid_SS): Residual sum of squares.
 Note C46(Resid_DF): Residual Degree of freedom.
 Note c47(Type): Description of the SS type.
 Note c48(F_Slop): F value for the overall difference of the slopes.
 Note c49(F_elev): F value for the overall difference of the elevations.
 Note c52(Line_1_slp): Slope of a first line for comparison.
 Note c53(Line_2_slp): Slope of a second line for comparison.
 Note c54(diff_slp): Difference between two slopes.
 Note c55(S.E._slp): Standard Error for the slopes.
 Note c56(q_slp): q value for the slopes.
 Note c57(q_crit_slp): critical q value for the slopes.
 Note c58 (Eq_NEq_slp): Difference between two slopes.
 Note c62(Line_1_ele): Elevation from a first line for comparison.
 Note c63(Line_2_ele): Elevation from a second line for comparison.
 Note c64(diff_ele): Difference between two elevations.
 Note c65(S.E._ele): Standard Error for the elevations.
 Note c66(q_ele): q value for the elevations.
 Note c67(q_crit_ele): Critical q value for the elevations.
 Note c68(Eq_NEq_ele): Difference between two elevations.
 Note c70(code_for_CI): Code of regression lines for the 95% CI's.
 Note c72(slp_low): Lower bound 95% CI for a slope.
 Note c73(slp_mean): Mean value for a slope.
 Note c74(slp_high): High bound 95% CI for a slope.
 Note c75(ele_low): Lower bound 95% CI for an elevation.
 Note c76(ele_mean): Mean value for an elevation.
 Note c77(ele_high): High bound 95% CI for an elevation.
 Note c78(obs_y_low): Lower bound 95% CI for an arbitrary y value.

 Note c79(ob_y_mean): Mean value for an arbitrary y value.
 Note c80(obs_y_high): High bound 95% CI for an arbitrary y value.
 Note
 Note
 Note The Macro has run successfully
 endmacro

#Appendix:

 # Notes for equations followed (excerpt from Zar's book)
 # Three formulae to remember for sums of squares of linear regression:
 # $Sgm(X^2) = (Xi - Xbar)^2 = Sgm(Xi^2) - (Sgm(Xi))^2/n$
 # $Sgm(Y^2) = (Yi - Ybar)^2 = Sgm(Yi^2) - (Sgm(Yi))^2/n$
 # $Sgm(XY) = Sgm((Xi - Xbar) * (Yi - Ybar)) = Sgm(Xi * Yi) - (Sgm(Xi)) * (Sgm(Yi)) / n$
 #
 #####
 # Table for computing sums of squares (SS) for k regression lines: #
 #####

#	Sgm(X^2)	Sgm(XY)	Sgm(Y^2)	Residual_SS	Residula_DF	#
# Reg 1	A1	B1	C1	SS1=C1-B1^2/A1	DF1=n1-2	#
# Reg 2	A2	B2	C2	SS2=C2-B2^2/A2	DF2=n2-2	#
#	#
#	#
# Reg k	Ak	Bk	Ck	SSk=Ck-Bk^2/Ak	DFk=nk-2	#
# "Pooled"				SSp=Sgm[1-k](SSi)	DFp=Sgm[1-k](ni-2)	#
# Reg					=Sgm[1-k]n-2k	#
# "Common"						#
# REg	Ac=Sgm[1-k]Ai	Bc=Sgm[1-k]Bi	Cc=Sgm[1-k]Ci	SSc=Cc-Bc^2/Ac	DFc=Sgm[1-k]ni-k-1	#
# "Total"						#
# Reg	At	Bt	Ct	SSt=Ct-Bt^2/At	DFt=Sgm[1-k]ni-2	#

 #
 # To test Ho: beta1=beta2=...=betak, one may calculate
 #

$$F = ((SSc - SSp) / (k - 1)) / (SSp / DFp) \quad \text{Eq.1}$$
 # a statistic with numerator and denominator degrees of freedom of k-1 and DFp,
 # respectively.
 #
 # IF Ho: beta1=beta2=...=betak is not rejected, then the common regression coefficient, bc,
 # may be used as an estimate of the beta underlying all k samples:
 #

```

#           bc=(Sgm[1-k](Sgm(XY)))/(Sgm[1-k](Sgm(X^2))) Eq.2
#
# Multiple comparisons among slopes
#
# Tukey test may be used to compare each pair of beta values, by H0=beta_A=beta_B
# and Ha=beta_A<>beta_B, where A and B can represent any two of the k regression lines.
# The test statistic is
#           q=(b_B-b_A)/SE                               Eq.3
#
# where b_A and b_B are two slopes, and SE is calculated as
#
#           SE=SQRT(((s_Y.X^2)p/2)*(1/(SgmX^2)A)+1/(SgmX^2)B) Eq.4
#
# The degree of freedom for determining critical value of q are pooled residual DF (DFp),and the
# number of lines to be compared (k) is used. In q table, loop up q(alpha, v,k).
#
# To compare overall difference of more than two elevations
# The following test statistic is used:
#
#           F=((SSt-SSc)/(k-1))/(SSc/DFc)                Eq.5
#
# Multiple comparisons among elevations:
#
# The Tukey test statistic is
#
#           q=ABS((Ybar_A-Ybar_B)-bc*(Xbar_A-Xbar_B))/SE Eq.6
#
#           SE=SQRT(((s_Y.X^2)c/2)*(1/nA+1/nB+(Xbar_A-Xbar_B)^2/(SgmX^2A+SgmX^2B))) Eq.7
#
# The critical q value is calculated as before.

```