Transcript of Spencer using GPT 4.0 to help him alter the VBA code in the Custom Macro to re-create a blank dataset, namely “0.75-14-4-202”

# Initial Prompt:

**Acting as a data scientist**, please tell me how to adjust the following VBA script:

Sub a()

Dim CSVFileName As String

Dim FileName As String

Dim regressors As Integer

Worksheets("Data").Select

Range("a1").Select

'If you want to resume the program in the middle, set ContinueFlag to 1 and specify the parameters of the dataset whence you want to continue.

ContinueFlag = 1

ContinueCollinearityLoop = 4

ContinueRegressors = 4

ContinueErrorVariance = 10

ContinueRegression = 232

'This is the number of data sets generated for each condition

DataSets = 500

'Adjust these loop endpoints as desired

'Collinearity isn't collinearity but the slope coefficient of a regressor as a function of X1. It is randomized positive/negative.

For CollinearityLoop = 1 To 4

For regressors = 3 To 15

For errorvariance = 1 To 10

Application.ScreenUpdating = False

For Regression = 1 To DataSets

If ContinueFlag = 1 Then

ContinueFlag = 0

CollinearityLoop = ContinueCollinearityLoop

regressors = ContinueRegressors

errorvariance = ContinueErrorVariance

Regression = ContinueRegression

End If

'This section allows for the selection of specific collinearities that don't fit nicely into a for-next-step loop.

Select Case CollinearityLoop

Case 1

Collinearity = 0

Case 2

Collinearity = 0.25

Case 3

Collinearity = 0.5

Case 4

Collinearity = 0.75

End Select

Worksheets("data").Range("B2").Value = Collinearity

Worksheets("data").Range("B3").Value = regressors

Worksheets("data").Range("B4").Value = errorvariance

'Verify that the number of randomly selected regressors equals the number of regressors needed for the generative equation.

Application.Calculate

Do Until Range("c3") = Range("b3")

Application.Calculate

Loop

' Filename is Collinearity - Regressors in the true model - error variance - dataset #

FileName = Worksheets("data").Range("B2") & "-" & Worksheets("data").Range("c3") & "-" & Worksheets("data").Range("B4") & "-" & Trim(Str(Regression)) & ".csv"

CSVFileName = Worksheets("data").Range("B1") & "\" & FileName

Call saveRangeToCSV(CSVFileName, regressors)

Next

Application.ScreenUpdating = True

DoEvents

Next

Next

Next

End Sub

Sub saveRangeToCSV(CSVFileName As String, regressors As Integer)

Dim rngToSave As Range

'This is the number of possible regressors in the search space

maxregressors = 30

Application.DisplayAlerts = False

On Error GoTo err

Set rngToSave = Worksheets("data").Range(Cells(5, 1), Cells(507, maxregressors + 1))

rngToSave.Copy

Set temp = Application.Workbooks.Add(1)

With temp

.Sheets(1).Range("A1").PasteSpecial xlPasteValues

.SaveAs FileName:=CSVFileName, FileFormat:=xlCSV, CreateBackup:=False

.Close

End With

err:

Application.DisplayAlerts = True

End Sub

*So that I can create just one dataset again that in my folder full of datasets created using this custom macro is blank which breaks everything I am trying to do with the datasets right now. The dataset I need to create is called 0.75-14-4-202. The naming convention and certain aspects of the construction of each dataset is as follows:*

1. Construct a random data set composed of 30 Candidate regressors, each distributed standard normal, and standard normal error. Construct a dependent variable as a linear function of K of the regressors and the error.

2. Run ERR, stepwise, (whatever other methods) on the data set in an attempt to find the “best fit” model (according to each procedures’ criterion for “best”).

3. Repeat Steps 1-2 500 times.

4. Repeat 1-3 with error variances 2, 3, 4, 5, 6, 7, 8, 9 , 10.

5. Repeat 1-4 with different values for K.

6. Repeat 1-5, but this time create the N regressors with randomly selected cross-regressor correlations (multicollinearity).

Filename: D-C-B-A.xlsx

D = Correlation between regressors x 10 (e.g., 1 = 0.1 correlation)

C = # of regressors (k)

B = Error variance

A = data set (1 through 500)

Important: Enter the output path in cell B2 in data.

CTRL-q activates the macro. The macro creates data sets with filenames as described above.

# ChatGPT

Based on your description, we need to make several changes to the existing VBA script to meet your requirements.

Your VBA script is already producing multiple datasets as described in your steps 1-6, but the naming convention and the file format are different from what you need.

Here's the modified code:

vba

Copy code

Sub a()

Dim CSVFileName As String

Dim FileName As String

Dim regressors As Integer

Worksheets("Data").Select

Range("a1").Select

ContinueFlag = 1

ContinueCollinearityLoop = 4

ContinueRegressors = 14 ' updated

ContinueErrorVariance = 4 ' updated

ContinueRegression = 202 ' updated

DataSets = 500

For CollinearityLoop = 1 To 4

For regressors = 3 To 15

For errorvariance = 1 To 10

Application.ScreenUpdating = False

For Regression = 1 To DataSets

If ContinueFlag = 1 Then

ContinueFlag = 0

CollinearityLoop = ContinueCollinearityLoop

regressors = ContinueRegressors

errorvariance = ContinueErrorVariance

Regression = ContinueRegression

End If

Select Case CollinearityLoop

Case 1

Collinearity = 0

Case 2

Collinearity = 0.25

Case 3

Collinearity = 0.5

Case 4

Collinearity = 0.75

End Select

Worksheets("data").Range("B2").Value = Collinearity

Worksheets("data").Range("B3").Value = regressors

Worksheets("data").Range("B4").Value = errorvariance

Application.Calculate

Do Until Range("c3") = Range("b3")

Application.Calculate

Loop

' Updated filename to match your requirements

FileName = (Collinearity \* 10) & "-" & Worksheets("data").Range("c3") & "-" & Worksheets("data").Range("B4") & "-" & Trim(Str(Regression)) & ".xlsx"

CSVFileName = Worksheets("data").Range("B1") & "\" & FileName

Call saveRangeToCSV(CSVFileName, regressors)

Next

Application.ScreenUpdating = True

DoEvents

Next

Next

Next

End Sub

Sub saveRangeToCSV(CSVFileName As String, regressors As Integer)

Dim rngToSave As Range

maxregressors = 30

Application.DisplayAlerts = False

On Error GoTo err

Set rngToSave = Worksheets("data").Range(Cells(5, 1), Cells(507, maxregressors + 1))

rngToSave.Copy

Set temp = Application.Workbooks.Add(1)

With temp

.Sheets(1).Range("A1").PasteSpecial xlPasteValues

' Saving as Excel Workbook instead of CSV

.SaveAs FileName:=CSVFileName, FileFormat:=51, CreateBackup:=False

.Close

End With

err:

Application.DisplayAlerts = True

End Sub

Changes made in this script:

1. In the subroutine a(), I updated ContinueRegressors, ContinueErrorVariance, and ContinueRegression to the values you provided.
2. In the filename creation, I multiplied Collinearity by 10 and changed the file extension from .csv to .xlsx.
3. In the saveRangeToCSV() subroutine, I updated the FileFormat parameter to 51 when saving the workbook. This is because

# User (Spencer Marlen-Starr) again

If I rename it Sub b(), but leave everything else the same, will it still work?

# ChatGPT

Yes, renaming the Sub from a() to b() while leaving everything else the same will still work. The name of the Sub does not affect its functionality. However, any references to this Sub elsewhere in your code should also be updated to b() to avoid errors.

Your updated code would be:

vba

Copy code

Sub b()

Dim CSVFileName As String

Dim FileName As String

Dim regressors As Integer

Worksheets("Data").Select

Range("a1").Select

ContinueFlag = 1

ContinueCollinearityLoop = 4

ContinueRegressors = 14

ContinueErrorVariance = 4

ContinueRegression = 202

DataSets = 500

# User (Spencer Marlen-Starr) again

Please finish what you were saying at this point in an earlier response: In the saveRangeToCSV() subroutine, I updated the FileFormat parameter to 51 when saving the workbook. This is because

# ChatGPT

This is because **FileFormat:=51** corresponds to the Excel 2007-2021 workbook format (.xlsx) in VBA. The original code had **FileFormat:=xlCSV** which would save the file in the CSV format. But based on your requirements, you want the file saved in the Excel Workbook (.xlsx) format. Here's what different **FileFormat** values represent:

* **FileFormat:=xlCSV** or **FileFormat:=6** saves the file in the CSV format.
* **FileFormat:=51** saves the file in the .xlsx format.
* **FileFormat:=52** saves the file in the .xlsx format but with macros disabled.
* **FileFormat:=50** saves the file in the .xlsb (Binary) format.
* **FileFormat:=56** saves the file in the .xls (Excel 97-2003) format.

Hence, **FileFormat:=51** was used to save the file as .xlsx.