Course Materials from Advanced GAMS Class

Excel Spreadsheet in Charge of GAMS

Bruce A. McCarl

Specialist in Applied Optimization
Professor of Agricultural Economics, Texas A&M
Principal, McCarl and Associates

mccarl@tamu.edu
mccarl@bihs.net
agecon.tamu.edu/faculty/mccarl

979-693-5694
979-845-1706

Excel Spreadsheet in Charge of GAMS

Introduction

Most of the time when we are dealing with GAMS we think of it being in charge. However, we can deal with an application where GAMS is entirely in the background and the user does not really know it is being used. In such a setting another program needs to pass data to GAMS and receive back the solution. Here I present an example in the context of Microsoft Excel. In this case GAMS will be called by EXCEL and will operate in the background. The files used herein are available in the zip file http://www.gams.com/mccarl/excelgams.zip

A little history is in order before beginning. A number of years ago GAMS Corporation had a section on their web site displaying linkage programs and an early version of this EXCEL program was resident on it. However while I had seen the entry I was unaware that it was a GAMS / EXCEL interface. Later Rob Davis with the Bureau of Reclamation came to one of my classes and told me he had been using an interface based on the GAMS page. I then looked into it and in the process reprogrammed the EXCEL macros slightly to improve the process and separate out the code functions to facilitate use by otheres. I also received assistance from Erwin Kalvelagen and Paul van der Eijk at GAMS Corporation in terms of finding the path for the GAMS system by reading the GAMSIDE.ini file. So here is an EXCEL / GAMS interface -- a product of many.

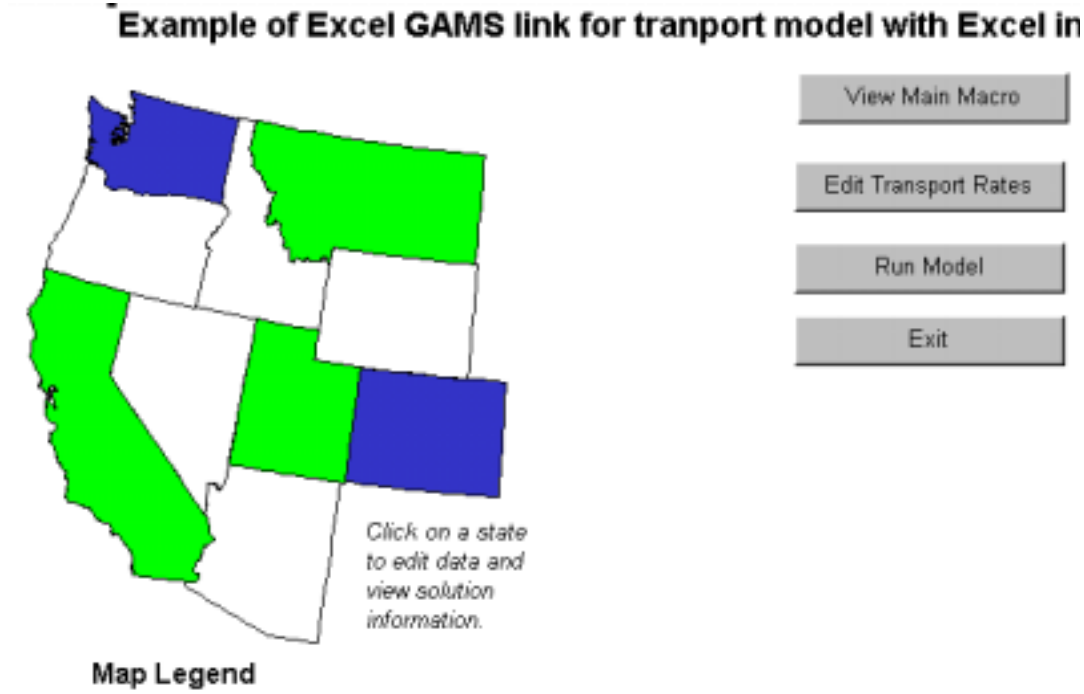
Excel Spreadsheet in Charge of GAMS

Basic Concept

The base application involves a spreadsheet built around a transportation model.

The basic driving mechanism is a map developed in a package like ARCVIEW, then imported into POWERPOINT and in turn EXCEL. Then that map was ungrouped into states and each state identified as an object in EXCEL which when clicked on causes certain actions to occur. Buttons were also defined to carry out certain tasks.

The first page of the spreadsheet appears as follows



Excel Spreadsheet in Charge of GAMS

Basic Sheet Design

(trnsxcll.xls,trnsxcll,gms)

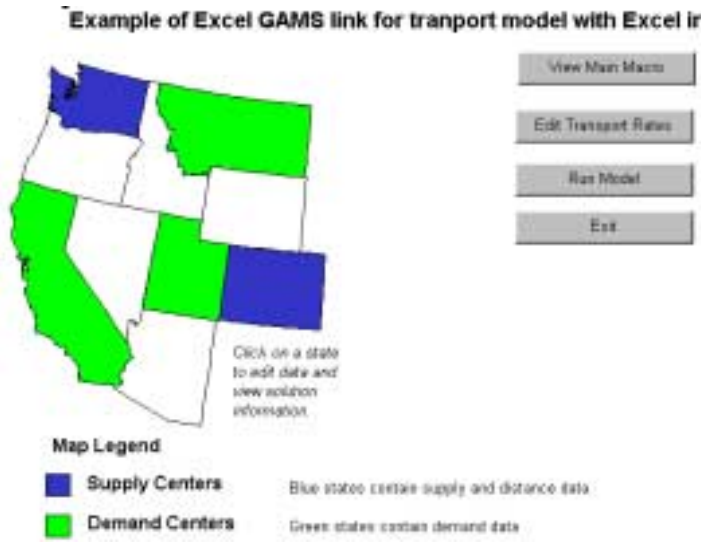
This page is set up so that pressing on one of the colored state maps transfers one to a page associated with that state. Also pressing one of the buttons causes a macros to run which runs GAMS, exits, etc.

The spreadsheet (trnsxcll.xls) involves

1. a nain sheet that contols the model (mapsheet)
2. 5 state sheets (Washington, California, Montana, Utah, Colorado)
3. one sheet for transport rate information (trancost)
4. one sheet that collects all of the data to be sent to GAMS (input)
5. one sheet that receives the output from GAMS (results)
6. 4 macros which manage the application

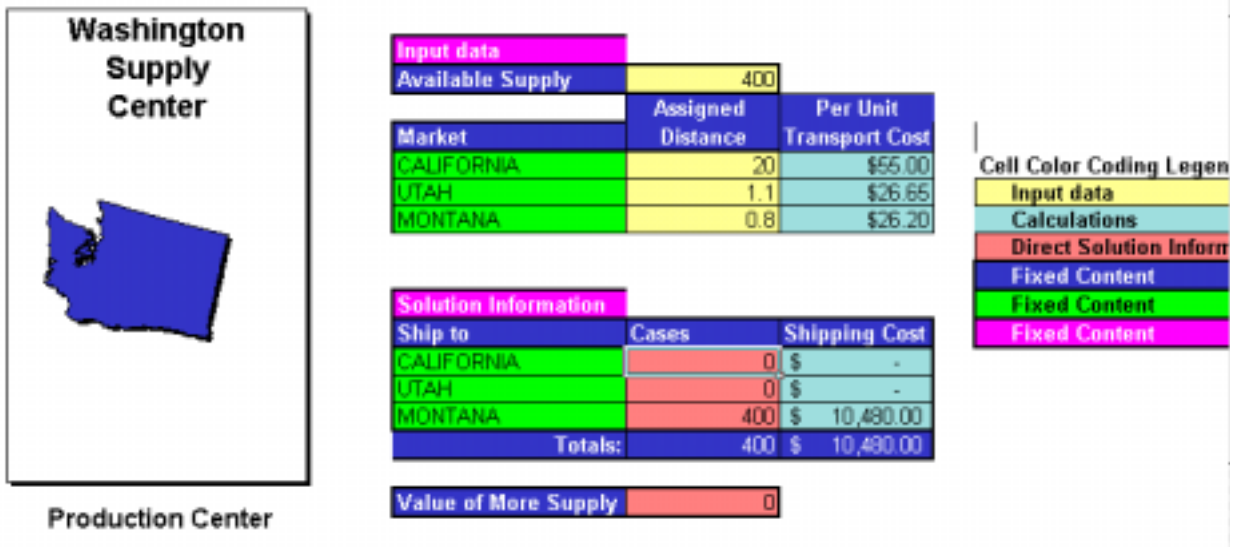
An associated GAMS file (trnsxcll.gms) receives the data and sends

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls,trnsxcll,gms)



Now lets examine the application then we will delve inside it.

Pressing the state of Washington brings up the sheet



Excel Spreadsheet in Charge of GAMS

This sheet contains a mixture of solution information and raw data

The solution information is in the solution information table and comes from the results sheet via formulae like that in cell F14

=Results!C2

The input data is picked up in the input sheet such as in cell C21 which contains the formula

=wash_supply

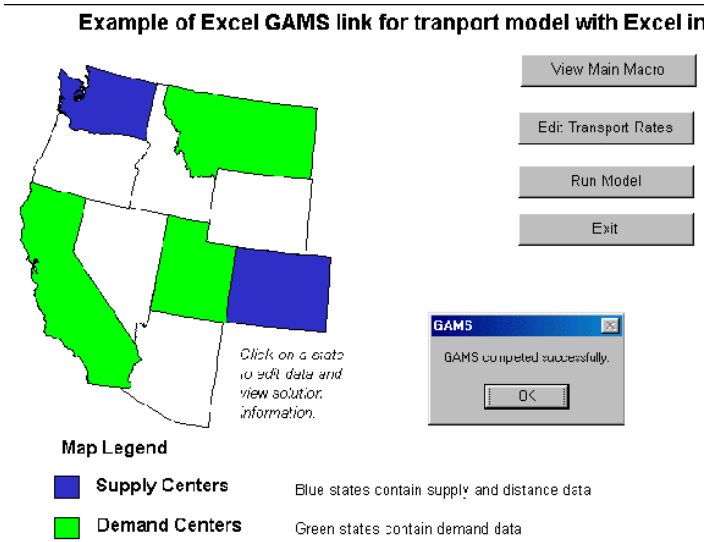
where wash_supply is the range name for the cell F4

Excel Spreadsheet in Charge of GAMS

(trnsxcll.xls,trnsxcll,gms)

Now suppose we want to run GAMS

We do this by going to the main sheet and pressing the button run model



If you then look at the lower bar on you screen you will see some icons flicker in and out as GAMS runs in the background culminating with the screen pictured above that tells you GAMS has finished.

You just ran GAMS but other than the little dialogue box you would never have known

Excel Spreadsheet in Charge of GAMS (trnsxcll.gms)

How is this done? Lets start from the beginning .We have a base GAMS model underlying this application

```
SETS         Supply     Locations of supply points /
$INCLUDE supply.set
/;
             Demand     Location of Demand markets /
$INCLUDE demand.set
/;
PARAMETERS          Available(supply)  Supply available in cases /
$ONDELIM
$INCLUDE supply.tbl
$OFFDELIM
/
          Needed(demand)  demand requirement in cases /
$ONDELIM
$INCLUDE demand.tbl
$OFFDELIM
/ ;
TABLE Distance(supply,demand)  distance in thousands of miles
$ONDELIM
$INCLUDE distance.tbl
$OFFDELIM
        ;
set tranparm parameters of transport rate function     /fixed,     permile/
parameter tranrate(tranparm) transport rate data /
$ONDELIM
$INCLUDE tranrate.tbl
$OFFDELIM
  /      ;
PARAMETER Cost(supply,demand)  transport cost in thousands of dollars per case ;
        Cost(supply,demand) = tranrate("Fixed")
                            + tranrate("permile") * Distance(supply,demand) / 1000;
positive VARIABLES        ship(supply,demand)  shipment quantities in cases
variable         Z       total transportation costs in thousands of dollars ;
EQUATIONS        COSTacct            define objective function
                 SUPPLYbal(supply)   observe supply limits at sources
                DEMANDbal(demand)   satisfy demand requirements at markets ;
COSTacct ..  Z  =E=SUM((supply,demand),Cost(supply,demand)*ship(supply,demand)) ;
SUPPLYbal(supply) ..   SUM(demand, ship(supply,demand))  =L=  Available(supply) ;
DEMANDbal(demand) ..   SUM(supply, ship(supply,demand))  =G=  needed(demand) ;
MODEL TRANSPORT /ALL/ ;
SOLVE TRANSPORT USING LP MINIMIZING Z ;
*       Output data to ascii csv file
FILE csv Report File /output.csv        /;
csv.pc = 5;
PUT csv;
PUT 'PLANT','MARKET','CASES' /;
LOOP((supply,demand),      PUT supply.tl, demand.tl, ship.l(supply,demand):10:0 /;);
```

Excel Spreadsheet in Charge of GAMS
(trnsxcll.gms)

In that code we have an input section that includes 6 files

```
SETS          Supply     Locations of supply points /
$INCLUDE supply.set
/
              Demand     Location of Demand markets /
$INCLUDE demand.set
/;
PARAMETERS         Available(supply)  Supply available in cases /
$ONDELIM
$INCLUDE supply.tbl
$OFFDELIM
/
              Needed(demand)  demand requirement in cases /
$ONDELIM
$INCLUDE demand.tbl
$OFFDELIM
/ ;
TABLE Distance(supply,demand)  distance in thousands of miles
$ONDELIM
$INCLUDE distance.tbl
$OFFDELIM
       ;
set tranparm parameters of transport rate function     /fixed,     permile/
parameter tranrate(tranparm) transport rate data /
$ONDELIM
$INCLUDE tranrate.tbl
$OFFDELIM
  /     ;
```

These give set names and data for the model and will be sent from Excel. The on/offdelim's are used so csv files can be passed

Excel Spreadsheet in Charge of GAMS
(trnsxcll.gms)

In that code we have an output section that writes a csv delimited file

```
FILE csv Report File /output.csv        /;
csv.pc = 5;
PUT csv;
PUT 'PLANT','MARKET','CASES' /;
LOOP((supply,demand), PUT supply.tl, demand.tl, ship.l(supply,demand):10:0 /;);
LOOP(supply,     PUT supply.tl, supplybal.m(supply):10:4 /;);
LOOP(demand,     PUT demand.tl, demandbal.m(demand):10:4 /;);
put "cost",z.l /;
put "modelstat" transport.modelstat;
PUTCLOSE csv;
```

That code saves the shipment levels, supply shadow prices, demand shadow prices, model objective function and model solution status to a file called output.csv. The command csv.pc=5 makes this come out in comma delimited form.

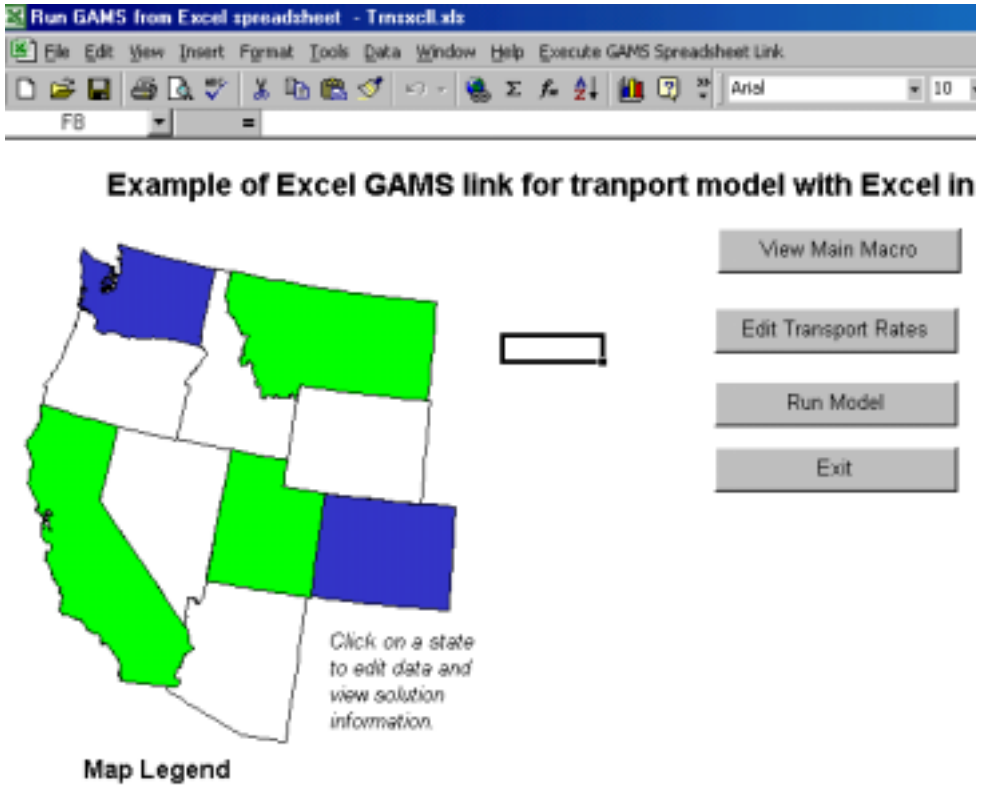
After execution output.csv looks like

```
"PLANT","MARKET","CASES"
"WASHINGTON","CALIFORNIA",0
"WASHINGTON","UTAH",0
"WASHINGTON","MONTANA",400
"COLORADO","CALIFORNIA",300
"COLORADO","UTAH",100
"COLORADO","MONTANA",100
"WASHINGTON",0.0000
"COLORADO",-0.1500
"CALIFORNIA",7.2500
"UTAH",6.0500
"MONTANA",6.2000
```

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

Now lets look at the reflection of that output in EXCEL. Back in the spreadsheet select edit from the menu

You then get



Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

Does the content of the sheet Results look familiar?

	A	B	C
1	PLANT	MARKET	CASES
2	WASHINGTON	CALIFORNIA	0
3	WASHINGTON	UTAH	0
4	WASHINGTON	MONTANA	500
5	COLORADO	CALIFORNIA	300
6	COLORADO	UTAH	100
7	COLORADO	MONTANA	100
8	WASHINGTON	0	
9	COLORADO	-0.15	
10	CALIFORNIA	27.25	
11	UTAH	26.05	
12	MONTANA	26.2	
13	cost	26425	
14	modelstat	1	
15			
16			

This is an import of the output.csv file.put by GAMS

```
"PLANT","MARKET","CASES"
"WASHINGTON","CALIFORNIA",0
"WASHINGTON","UTAH",0
"WASHINGTON","MONTANA",400
"COLORADO","CALIFORNIA",300
"COLORADO","UTAH",100
"COLORADO","MONTANA",100
"WASHINGTON",0.0000
"COLORADO",-0.1500
"CALIFORNIA",7.2500
"UTAH",6.0500
```

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

Open the sheet Input. At the top it contains

	A	B	C	D	E
1	PLANT	MARKET	CASES		
2	WASHING	CALIFORNIA	0		
3	WASHING	UTAH	0		
4	WASHING	MONTANA	400		
5	COLORAD	CALIFORNIA	300		
6	COLORAD	UTAH	100		
7	COLORAD	MONTANA	100		
8	WASHING	0			
9	COLORAD	-0.15			
10	CALIFORN	7.25			
11	UTAH	6.05			
12	MONTANA	6.2			
13	cost	5805			
14	modelstat	1			
15		CALIFORNIA	300		
16		UTAH	100		
17		MONTANA	600		
18					
19					
20		Production Center	Capacity		
21		WASHINGTON	600		
22		COLORADO	500		

The data are basicall formatted for GAMS. For example exporting B25:E27 in csv format we get (with addition of X1)

```
X1,CALIFORNIA,UTAH,MONTANA
WASHINGTON,20,1.1,0.8
COLORADO,1.4,0.6,0.7
```

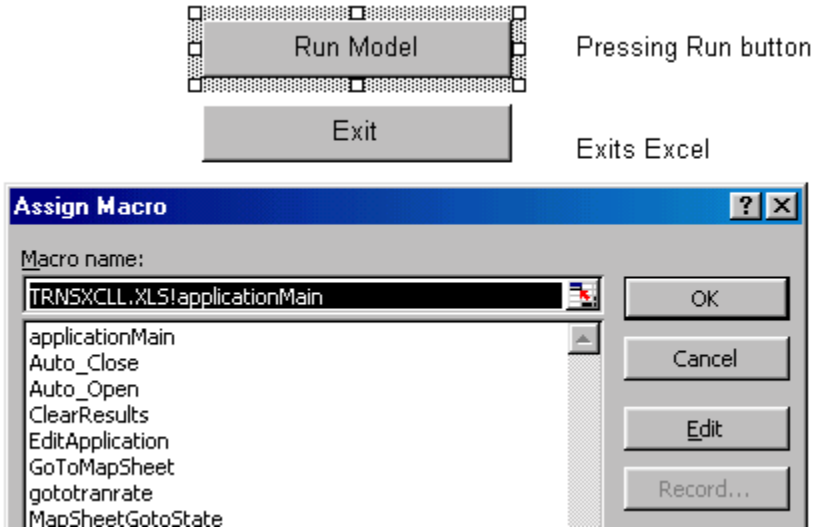
which is the file demand.tbl included in trnsxcll.gms through

```
TABLE Distance(supply,demand)  distance in thousands of miles
$ONDELIM
$INCLUDE distance.tbl
```

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

So how do the programs talk? Through Excel macros

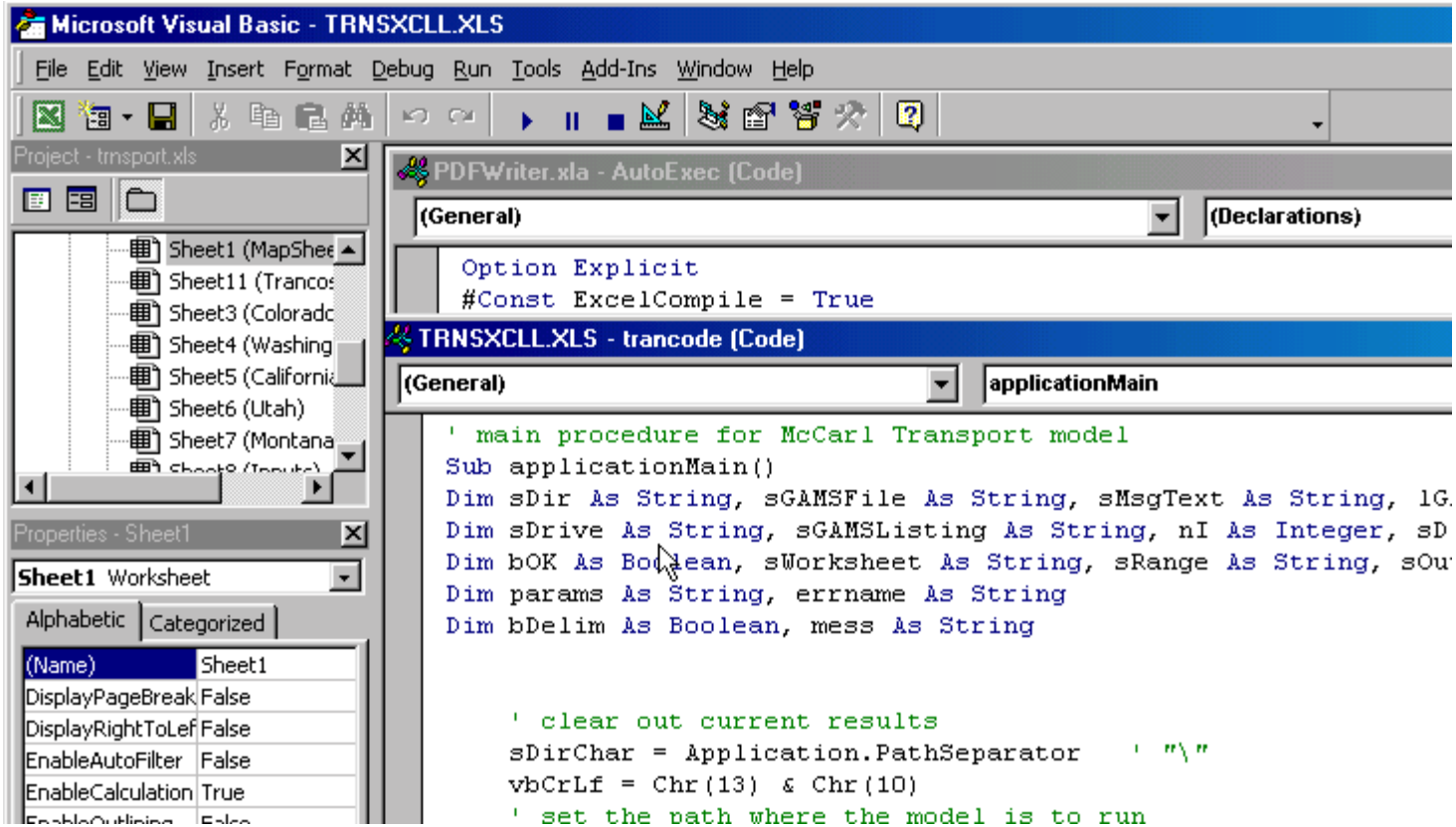
In the main sheet (mapsheet) of the spreadsheet right click on the run model button selecting assign macro



Note the macro assigned to this button is called applicationMain. Go ahead and click on it and choose to edit.

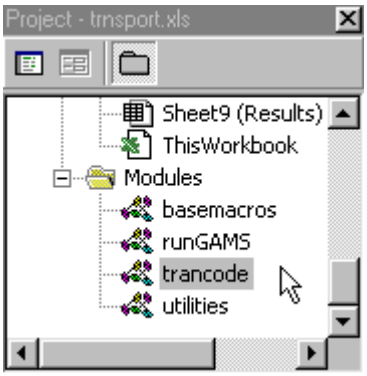
Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

You have now opened the visual basic editor and should have a screen like



Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

Here you are in module trancode with macro applicationmain highlighted. In top left under menu bar you find the Visual Basic Macros which run this application.



There are 4 macro code segments only one of which you ordinarily need be concerned with.

Visual Basic Segment	Code Purpose
trancode	Code which runs the transport application. This will change according to application
basemacro	Base macros for starting up and managing the menu. This is likely Generic but could change if one wished to do more with menu alterations.
runGAMS	Code for running the GAMS job. This should not be changed by the user.
utilities	Code for managing file transfers between GAMS and Excel. This should not be changed by the user.

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

Now lets look at 4 essential pieces of applicationmain

1. Passing data to GAMS
2. Causing GAMS to run
3. Getting back answer from GAMS and telling user about GAMS run status

1. Passing data to GAMS

Six code segments exist in applicationmain which are like

```
sWorksheet = "Inputs"
sRange = "distancedat"
sOutFile = sDir & sDirChar & Trim(Range("filefordistancetbl"))   ' "distance.tbl"
bDelim = True
bOK = XL2Txt(sWorksheet, sRange, sOutFile, bDelim)
```

This

a. saves name of sheet (Inputs in this case) where the data item resides that is to be written in sWorksheet
b. saves range name (distancedat in this case) in sheet identified in step a where data item resides in sRange (could be a1:c3 or any other valid excel range).
c. Composes file location and name using name in named range (in this case the file name is in filefordistancedat and exists in the Inputs sheet)
d. Uses function from utilities XL2Txt visual basic code segment to dump data to file

2. Causing GAMS to run

Two functions are required in the macro for GAMS to run.

The first involves specifying the name of the file to run and the lst file. This is done in the following two statements. Note that the macro will supply the directory to find the files as long as the determination of sDir and sDirChar are not altered. These two names are placed into variables as follows

```
sGAMSFile = sDir & sDirChar & "trnsxcll.gms"
   sGAMSListing = sDir & sDirChar & "trnsxcll.lst"
```

The second involves constructing the GAMS call and consists of statements where the path for the GAMS system directory, and any needed additional parameters for the GAMS call are specified. This is done by

```
params = ""
 lGAMSRet = GAMSrun(params, sGAMSFile)
```

Note in GAMSrun the name of the GAMS executable is found using

```
return_code = GetPrivateProfileString("EXECUTE",
  "EXECUTABLE", "*NOT FOUND*", strBuffer, lenstr,
  "gamside.ini")
```

and the statements just below that.

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

3. Getting back answer from GAMS and insuring all is right

This is done by the code

```
If lGAMSRet = 0 Then
  ' GAMS was successful, import the GAMS output files as generated by the model
  If Len(Dir(sOutFile)) = 0 Then
     MsgBox "GAMS Output file not found: " & sOutFile
  Else
     Application.ScreenUpdating = False
     bOK = Txt2XLDump(sWorksheet, sRange, sOutFile)
     Application.ScreenUpdating = True
  End If
  sMsgText = "GAMS completed successfully."
 ' check optimality / feasibility status here
   mess = optstatus()
   If Not mess = "Optimal" Then
      sMsgText = "Bad Model Result when running GAMS." & vbCrLf & vbCrLf & mess & "."
      MsgBox sMsgText, vbOKOnly + vbCritical, "GAMS"
     If MsgBox("Do you want to look at the listings file?", vbYesNoCancel + vbDefaultButton2 + vbQuestion) = vbYes
Then
        Shell "Notepad.exe " & sGAMSListing, 1
      End If
  Else
     MsgBox sMsgText, vbOKOnly, "GAMS"
  End If
Else
  'GAMS terminated improperly
   sGAMSErrorText = GamsErrorString(lGAMSRet)
  ActiveWorkbook.Worksheets("MapSheet").TextBoxes("text box 31").Text = "Errors running GAMS."
  sMsgText = "Errors encountered" & vbCrLf & vbCrLf & "Error Code " & lGAMSRet & ": " & sGAMSErrorText & "."
       MsgBox sMsgText, vbOKOnly + vbCritical, "GAMS"
       If MsgBox("Do you want to look at listings file?", vbYesNoCancel + vbDefaultButton2 + vbQuestion) = vbYes Then
           Shell "Notepad.exe " & sGAMSListing, 1
       End If
  End If
```

That code

1. Checks that GAMS completed normally
2. If so retrieves output into Excel using txt2 xc
3. Checks optimality status
4. If error has occured messages user and allows user to access LST file

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)

The other aspect of the implementation involves the sheets in the Excel workbook.

Here data are entered in sheets identified by state name and Inputs sheet has formulas which copy that data. This can be seen by looking at formulas in distance matrix which refer to data in the Washington and Colorado sheets.

Formulae are also added in the state sheets to copy the solution data from the Results sheet. See the shipment levels and shadow price cell entries for the state sheets.

One can do this in another way

```
Set oProdCenters = Worksheets("Inputs").Range("supplyset")
nProdCnt = oProdCenters.Rows.Count
Set oResults = Worksheets("Results").Range("A2:C7").CurrentRegion
For nI = 1 To nProdCnt
  sProd = oProdCenters.Cells(nI, 1)
  For Each oX In Worksheets(sProd).Range("E14:F16")
     sMarket = oX.Value
     For nJ = 2 To oResults.Rows.Count
          If oResults.Cells(nJ, 1) = sProd And oResults.Cells(nJ, 2) = sMarket Then
             oX.Cells(1, 3) = oResults.Cells(nJ, 3)
            Exit For
          End If
      Next
   Next
Next
```

This code matches up names and copies in information

Excel Spreadsheet in Charge of GAMS
(trnsxcll.xls)
Summary Steps

Now how can you do this

You need to follow 4 steps

1. Set up your GAMS model that will be fed by the spreadsheet
   a. Specify the format of the data items and their files which will be passed from spreadsheet
   b. Develop a put file that contains the data to be imported into the spreadsheet. Set this up in csv format.
2. Develop the spreadsheet and include separate sheets named inputs and results.
   a. Manually build the input sheet so it has the components from the other application sheets components you wish to pass to GAMS. Then save this as csv files for each items. Make sure you can include them in the GAMS program without error. Adjust the format in the inputs sheet and resave the csv until error free inclusion is attained.
   b. Develop put statements in GAMS with the results to be passed to EXCEL. Import the resultant file into EXCEL then link output parts of the other sheets so the answers are copied back
3. Develop other sheets to support application
4. Adapt macro so it will run GAMS for your application saving proper files for inclusion and importing output file