Transport Problem Exercises with GAMS Studio

These exercises provide an introduction to GAMS using the transport model to resolve syntax and execution errors and marginally extend the scope of the model.

Contents

Transport Problem Exercises with GAMS Studio 1
Contents 1
1. Introduction 2
Aims 2
Objectives 2
Process 2
2. Set up 3
Configuring GAMS 3
3. Exploring a GAMS Model 7
New Project 9
Populating the New Project from a User Model Library 10
Running a Model in GAMS Studio 12
3. Syntax and Execution Error Exercises 15
Syntax Error Exercises 15
Execution Error Exercises 16
4. Transport Model Exercises 18
Changing Unit Transport Costs 18
Changing Distances 18
New and Intermediate Markets 19
Intermediate Markets 19

An effort has been made to avoid errors, but some are bound to exist. If you find errors, please email, with the details, to scott@cgemod.org.uk

1. Introduction

GAMS, arguably as with all languages, is best learnt by doing. These exercises adopt that principle. These instructions are written assuming you are using these exercises prior to taking a course offered by cgemod.

Aims

The aims of this set of exercises are to develop

1. an understanding of the content and structure of GAMS models;
2. an understanding of the main components of a GAMS model;
3. an ability to identify and correct syntax errors;
4. an ability to identify and correct execution errors; and
5. an ability to modify a simple GAMS programme.

Objectives

At the end of the set of exercises the participant will

1. understand the structure of GAMS models;
2. understand the main components of a GAMS model;
3. be able to identify and correct common syntax errors;
4. appreciate how to correct common execution errors; and
5. be able to modify a simple GAMS programme.

Process

These exercises work by exploring a model; conducting exercises that generate syntax and execution errors; modifying a model; and running some simple experiments.

The basic code is provided as '`trans1.gms`', which is a version of the standard GAMS transport problem used by GAMS as their tutorial. All the exercise assume that the user is working with GAMS Studio as the editor.[1]

A guide to GAMS Studio is available at www.cgemod.org.uk/

[1] Instructions for using GAMS with GAMSIDE, and these exercises setup for GAMSIDE will remain available on www.cgemod.org.uk until, at least late 2023.

2. Set up

Before conducting these exercises, it is necessary to setup your PC to access the files required for the exercises. These exercises, and the associated instructions, assume that your PC has been setup inline with the instructions in this section. The reasons for this are simple: first, we can only provide instructions that accurately reflect a setup that is known to us, second, because the setup is known to us, we can anticipate likely difficulties, and third, the setup we advocate has been tested. We anticipate that users of GAMS will, with experience, develop their own ways of working; hence the setup we advocate is designed to provide a basis upon which users can develop their own way of working. We acknowledge that the setup we advocate reflects, to a greater or lesser, extent our way of working.

The instructions for these exercises were developed on Windows based PC. We know that GAMS and GAMS Studio are available for Apple and Linux based PC. Some testing has been conducted for **an** Apple PC, but we are not users of Apple or Linux PCs so we cannot guarantee that the instructions are accurate for Apple OR Linux PCs.

There are two steps that must first be taken

1. install GAMS and GAMS Studio on your PC, instructions for this are available in 'Intro to GAMS Studio.pdf' (www.cgemod.org.uk) (we assume that the path for GAMS is C:\GAMS); and
2. open Windows Explorer
   a. create the directory `C:\cgemod` – this is the master directory that will be assumed for all courses offered by cgemod (this should have been done when installing and testing GAMS with GAMS Studio);
   b. create the directory `C:\cgemod\downloads` – this is the directory we will expect you to use to store all the course materials provided for cgemod courses.

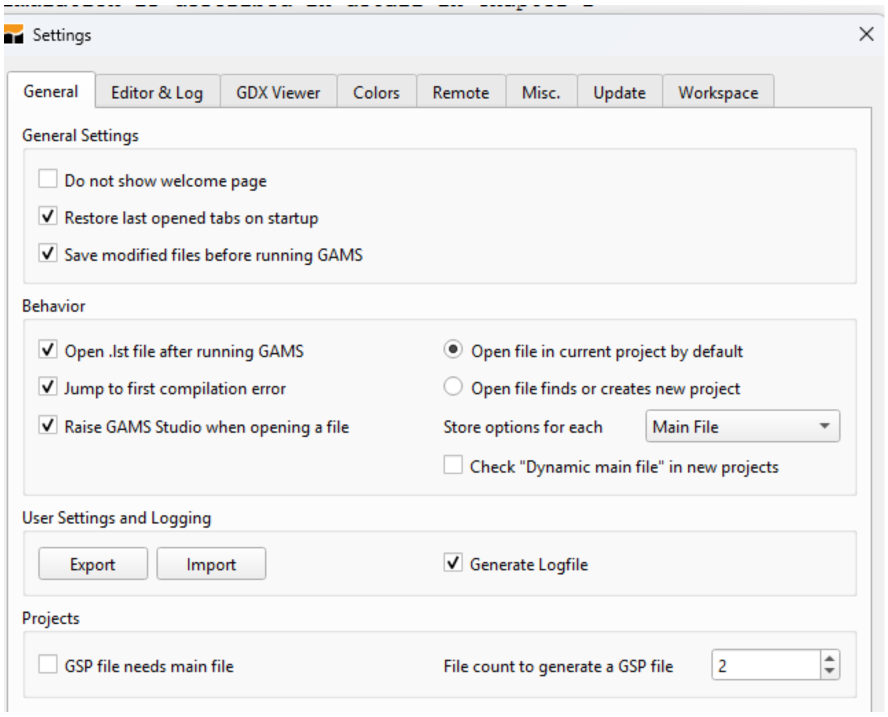
Configuring GAMS

There are different ways different users like to work, hence there are different 'appropriate' ways to configure GAMS Studio. For all cgemod courses we use a standard configuration of GAMS Studio. This section sets out the standard configuration used by cgemod.

With settings (F7) open select the General tab and choose the following options (Figure 2.1)

1. 'Open file in current project by default' (this means that you want to open a new file in a new project it is a conscious decision).
2. Set 'File count to generate GSP file' to 2 (this means GSP files appear in Windows Explorer as soon as programme is run)
3. Set 'GSP file needs main file' to OFF (this means a a GSP file can exist without a GMS file)
4. Set the 'Open *.lst file after running GAMS' to ON.

Figure 2.1 Settings: General tab



General Settings

- ☐ Do not show welcome page
- ✓ Restore last opened tabs on startup
- ✓ Save modified files before running GAMS

Behavior

- ✓ Open .lst file after running GAMS
- ✓ Jump to first compilation error
- ✓ Raise GAMS Studio when opening a file
- ● Open file in current project by default
- ○ Open file finds or creates new project
- Store options for each: Main File
- ☐ Check "Dynamic main file" in new projects

User Settings and Logging

- ✓ Generate Logfile

Projects

- ☐ GSP file needs main file
- File count to generate a GSP file: 2

With settings (F7) open select the Editor & Log tab and choose the following options (Figure 2.2)

1. Set the Font to Courier New (a standard fixed pitch font)
2. Set the Font Size to 12 (the choice here is personal, but we find 12 works for most users)

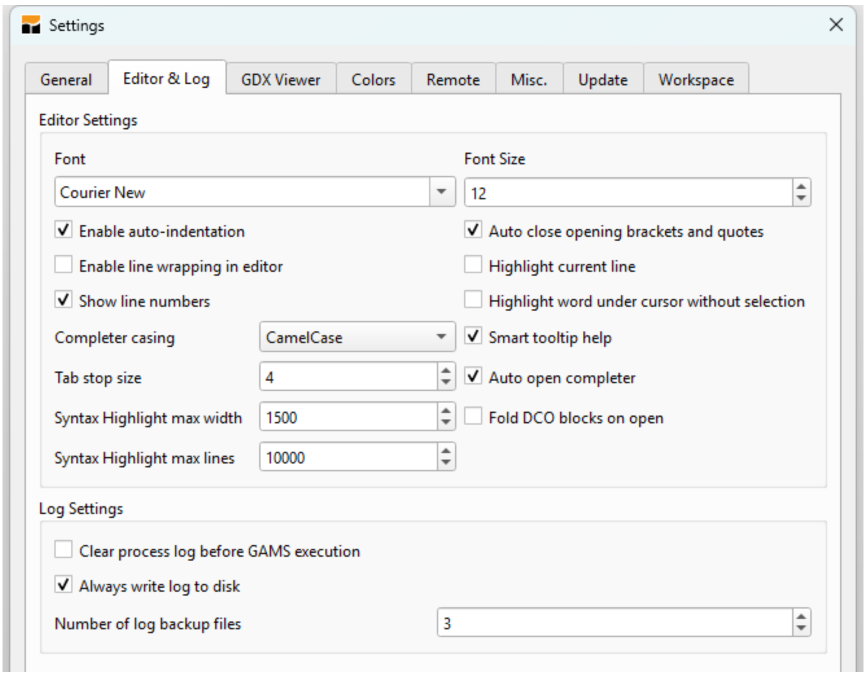
With settings (F7) open select the Workspace tab and choose the following options (Figure 2.3)

1. Set the Default Workspace to `C:\cgemod\error` (this means that if you make errors in loading files they are likely to end up in the directory `C:\cgemod\error`

2. Set 'Clean-up selected workspace directories on start-up' to OFF (this reduces the risk of deleting files that you prefer keeping)

With settings (`F7`) open select the Misc tab. This tab will need updating as you add more User Model libraries with different cgemod course. In Figure 2.4 it is set for `trans_lib`.

Figure 2.2 Settings: Editor & Log tab



Settings

General | Editor & Log | GDX Viewer | Colors | Remote | Misc. | Update | Workspace

Editor Settings

Font: Courier New
Font Size: 12

- [✓] Enable auto-indentation
- [✓] Auto close opening brackets and quotes
- [ ] Enable line wrapping in editor
- [ ] Highlight current line
- [✓] Show line numbers
- [ ] Highlight word under cursor without selection
- [✓] Smart tooltip help
- [✓] Auto open completer
- [ ] Fold DCO blocks on open

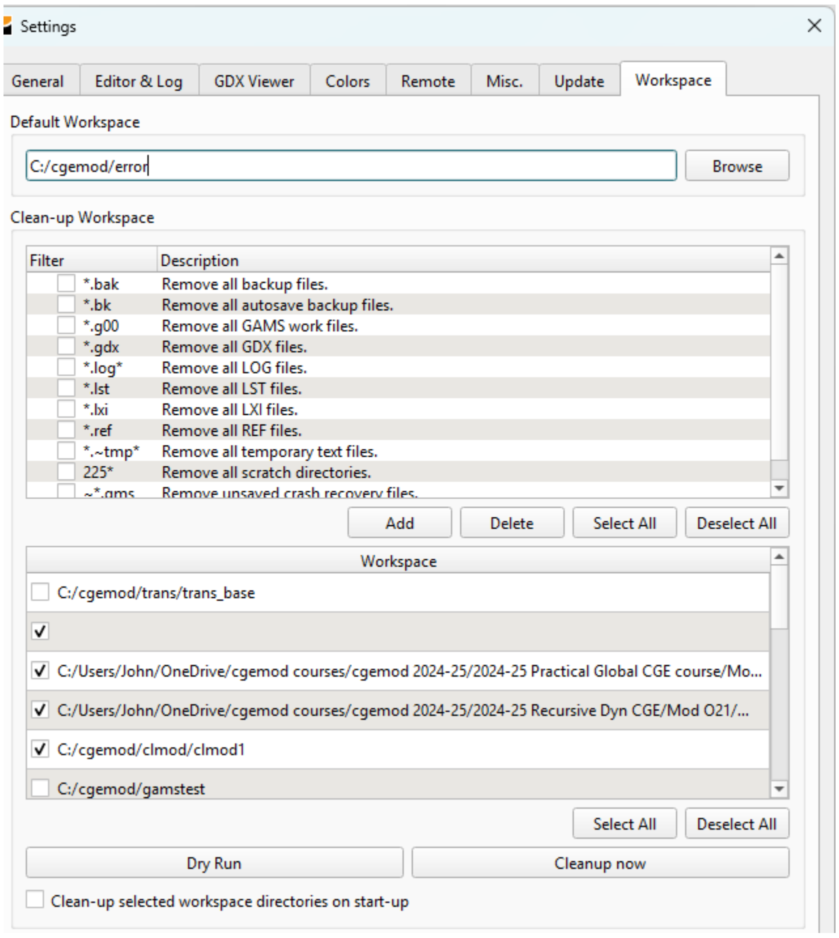
Completer casing: CamelCase
Tab stop size: 4
Syntax Highlight max width: 1500
Syntax Highlight max lines: 10000

Log Settings

- [ ] Clear process log before GAMS execution
- [✓] Always write log to disk

Number of log backup files: 3

Figure 2.3 Settings: Workspace tab



Default Workspace: C:/cgemod/error

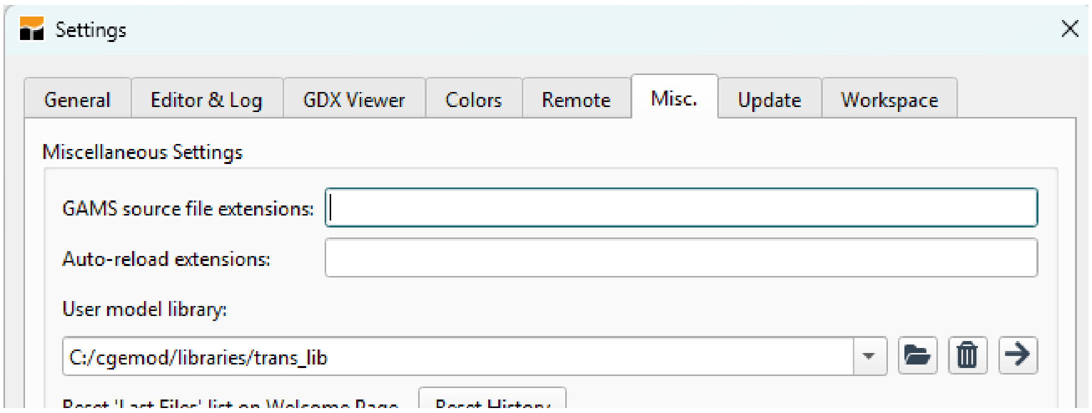
Clean-up Workspace

Filter	Description
☐ *.bak	Remove all backup files.
☐ *.bk	Remove all autosave backup files.
☐ *.g00	Remove all GAMS work files.
☐ *.gdx	Remove all GDX files.
☐ *.log*	Remove all LOG files.
☐ *.lst	Remove all LST files.
☐ *.lxi	Remove all LXI files.
☐ *.ref	Remove all REF files.
☐ *.~tmp*	Remove all temporary text files.
☐ 225*	Remove all scratch directories.
☐ ~*.gms	Remove unsaved crash recovery files.

Workspace
☐ C:/cgemod/trans/trans_base
✓
✓ C:/Users/John/OneDrive/cgemod courses/cgemod 2024-25/2024-25 Practical Global CGE course/Mo...
✓ C:/Users/John/OneDrive/cgemod courses/cgemod 2024-25/2024-25 Recursive Dyn CGE/Mod O21/...
✓ C:/cgemod/clmod/clmod1
☐ C:/cgemod/gamstest

☐ Clean-up selected workspace directories on start-up

Figure 2.4 Settings: Misc tab



Miscellaneous Settings

GAMS source file extensions:

Auto-reload extensions:

User model library: C:/cgemod/libraries/trans_lib

Reset 'Last Files' list on Welcome Page — Reset History

c.

3. Exploring a GAMS Model

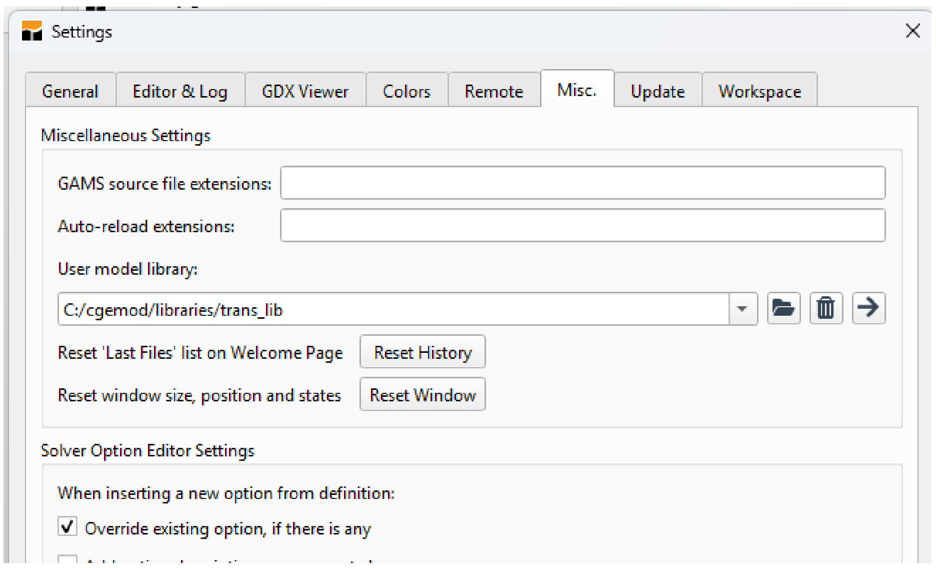
This exercise is designed to help you understand the structure of a GAMS model and the content of the files generated when that model has been run. This section repeats a lot of material in 'Intro to GAMS Studio.pdf'; this is deliberate.

The steps involved are these

1. Create a directory `C:\cgemod\libraries`, to contain the libraries of computer code you will build during this and subsequent courses, and then add a sub directory `trans_lib`, i.e., `C:\cgemod\libraries\trans_lib`.
2. Download the `trans.zip` file from the cgemod site (http://cgemod.org.uk/int_gams.html) and save it in your `downloads` directory.
3. Unzip the contents of the file `trans_lib.zip` into the library directory `C:\cgemod\libraries\trans_lib`. WinZip may by default unzip these files into a directory, usually called `trans`. If so, you will need to copy these files and paste them into your library directory `trans_lib`. The files in this directory must NOT be in sub directories.[1]
4. Create the directory `C:\cgemod\trans\trans_base` (this is the path that will be assumed in these instructions; do not create the directory on a drive that synchonises in real time with an external drive, e.g., OneDrive, because this can cause inconsistencies between versions of documents
5. Open GAMS Studio.
6. Open Studio settings - `File > Settings` or `F7` or the `Settings Wheel` on the toolbar and go to the `Misc` tab.
7. Set the User model library to `C:\cgemod\libraries\trans_lib` (you can browse to find the location of the `trans_lib` library.
8. Close the Studio Settings window, remember to click `Apply` and `OK` at the bottom of the `Misc` tab. The result, in Windows Explorer should be like that in Figure 3.2.
9. Open the `Model Library Explorer`, `F6`. The results should look like Figure 3.3, although you may have to scroll across to find the Introduction to GAMS Studio tab. Note the 'person' figure associated with a User Model Library.

[1] The default settings in WinZip now make you do this as a two-stage process. In WinZip 'classic view' you can avoid the two-step process, but it is fiddly and error prone. It is called progress!!!!

Figure 3.1 Settings – Misc tab



Settings

Miscellaneous Settings

GAMS source file extensions:

Auto-reload extensions:

User model library: C:/cgemod/libraries/trans_lib

Reset 'Last Files' list on Welcome Page [Reset History]

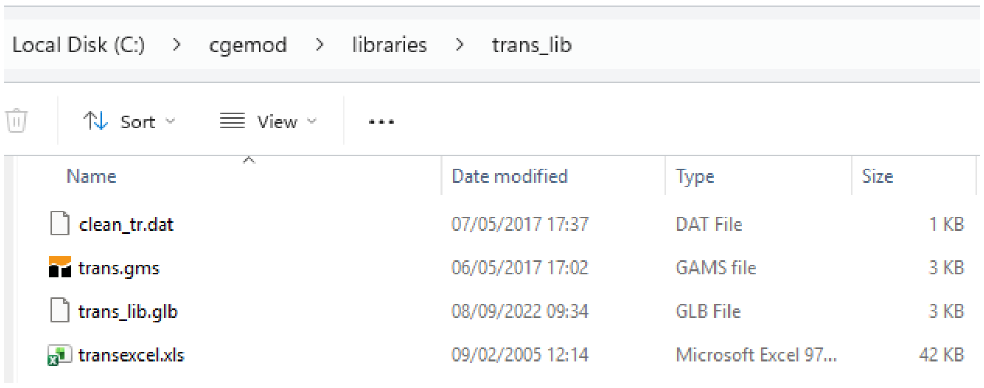
Reset window size, position and states [Reset Window]

Solver Option Editor Settings

When inserting a new option from definition:

☑ Override existing option, if there is any

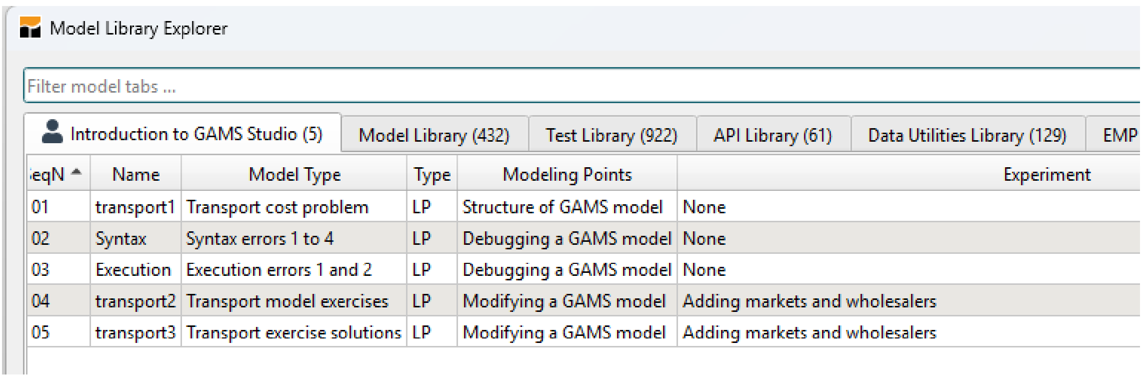
Figure 3.2 trans_lib Directory



Local Disk (C:) > cgemod > libraries > trans_lib

Name	Date modified	Type	Size
clean_tr.dat	07/05/2017 17:37	DAT File	1 KB
trans.gms	06/05/2017 17:02	GAMS file	3 KB
trans_lib.glb	08/09/2022 09:34	GLB File	3 KB
transexcel.xls	09/02/2005 12:14	Microsoft Excel 97...	42 KB

Figure 3.2 Model Library Explorer cw User Model Library (trans_lib)



Model Library Explorer

eqN	Name	Model Type	Type	Modeling Points	Experiment
01	transport1	Transport cost problem	LP	Structure of GAMS model	None
02	Syntax	Syntax errors 1 to 4	LP	Debugging a GAMS model	None
03	Execution	Execution errors 1 and 2	LP	Debugging a GAMS model	None
04	transport2	Transport model exercises	LP	Modifying a GAMS model	Adding markets and wholesalers
05	transport3	Transport exercise solutions	LP	Modifying a GAMS model	Adding markets and wholesalers

We, cgemod, use User Model Libraries to supply collections of files to participants on our courses. The contents of cgemod's User Libraries differ from standard GAMS libraries in that our libraries supply multiple files with each download while GAMS libraries only supply individual files. In addition, our libraries are set up with the intention of populating designated directories. Nevertheless, they operate in the same way as standard GAMS libraries.

New Project

Our courses are designed so that participants need regularly to create New Projects with each project having its own (sub) directory. The New Project feature in GAMS Studio is not efficient but can be used to create an empty project that can be populated with a new and blank programme file (**.gms) and then from a Model Library.[1] All courses offered by cgemod rely on User Model Libraries to provide access to structured course materials, so understanding how to create a New Project is important for the courses. The steps to follow are

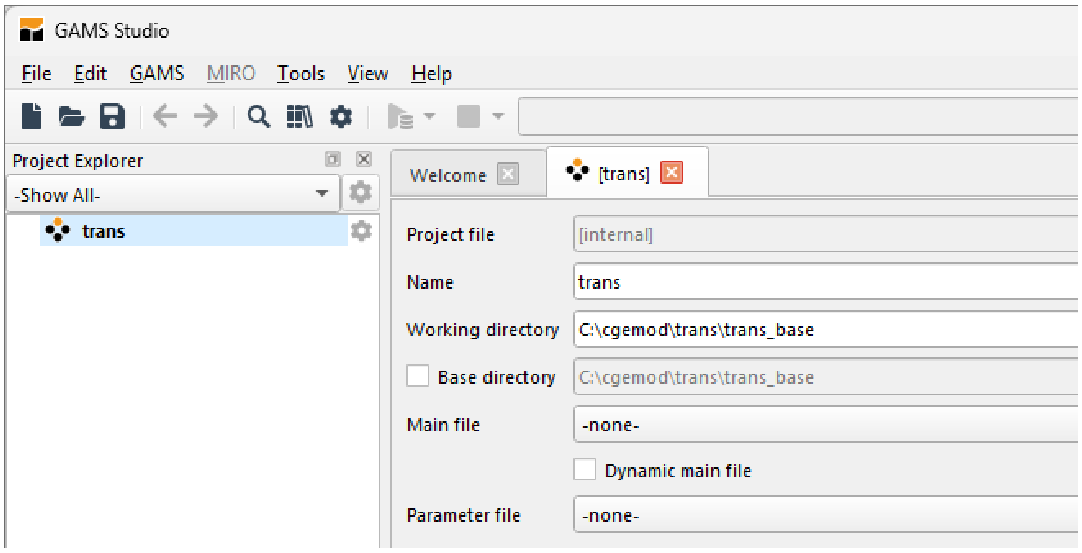
1. in Windows Explorer create the directory for the new project, e.g., `C:\cgemod\trans\trans_base`, (this should already exist)
2. open the Settings (`F7`)
3. on the General tab make sure that the option '`Open file in current project by default`' is selected,
4. choose `File>New Project` and this opens the project options in the edit viewer,
5. the default name for the project is `newProject`,
6. rename the project name as, for example, `trans` in the editor window (note the name in the Project Explorer will not change until the project is saved – see step 7 below),
7. set the Working directory to the new project directory, e.g., `C:\cgemod\trans\trans_base`, using the BROWSE button, (see Figure 3.3)
8. note the Base directory will default to the same name as the Working directory,
9. choose `File>Save` – **DO NOT USE SAVE AS**[2]
10. now choose `File>New` (see below) that will open a Windows Explorer window with defaults File name, `new***.gms`. (this step together with step 3 above is necessary to ensure that Studio is directed to use the new project's directory),

1 GAMS are reportedly exploring options to improve the efficiency of the New Project facility.
2 As of Studio version 1.22.2 using Save As produces an error.

11. click `SAVE` and `new**.gms` will appear in the trans project in the Project Explorer (see Figure 3.4).

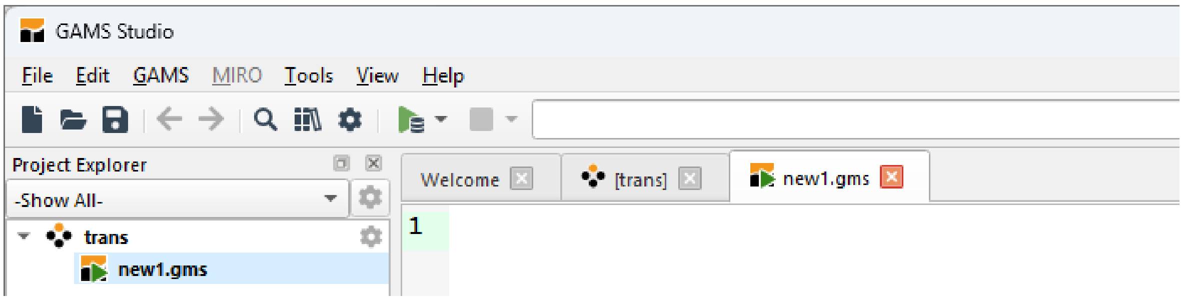
You now have a New Project that is effectively empty. This method will be used repeatedly although he instructions to create a New Project will be truncated.

Figure 3.3 New Project with Project Options in the editor



Project file: [internal]
Name: trans
Working directory: C:\cgemod\trans\trans_base
Base directory: C:\cgemod\trans\trans_base
Main file: -none-
Dynamic main file
Parameter file: -none-

Figure 3.4 New GMS file in New Project



<u>Populating the New Project from a User Model Library</u>

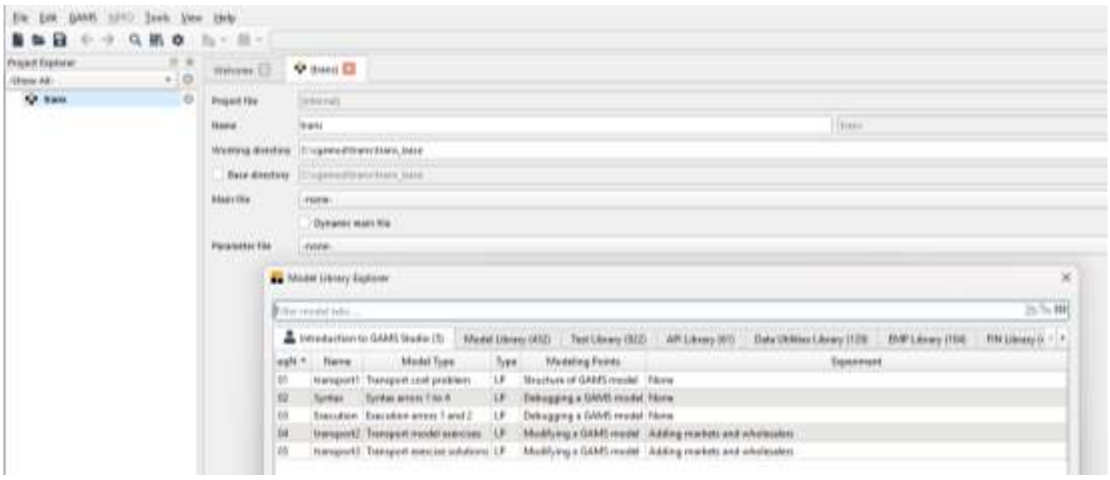
The next stage is to populate the directory with files from the User Model Library (`C:\cgemod\libraries\trans_lib`). To do this carry out the following

1. In Studio press `F6` and in the Model Library Explorer select the `Introduction to GAMS Studio Lib` and then select the library file `transport1`, which is SeqNr: 1, and choose Load (or double click of the name). (Figure 3.5)

2. The `trans.gms` model will now be displayed in the editor window and be listed in the Project Explorer as being in the project `trans`. (see Figure 3.6).
3. If you right-click on the project name and select `Open Location`, you will see that 4 files have been downloaded to the directory `C:\cgemod\trans\trans_base`. (See Figure 3.6).
    a. If the file `trans.gms` does NOT open in the editor pane the probable cause is that either you did not establish the library correctly or the file did not load from your working directory. Check first that the file `trans.gms` is in the library (`C:\cgemod\libraries\translib`); then check if has been downloaded to your working directory (`C:\cgemod\trans\trans_base`); and finally check to see if the file `trans.gms` is in the working directory. As should be evident the likely problem will be associated with the setting up of the directories. If this is the case delete what you have done and start again.
4. Studio can now be used to work with the transport model as done previously in this introduction and in the associated exercises.

This is a method for establishing a New Project in Studio and then populating the new project with files in a User Model Library. It also demonstrates how multiple files can be provided from a single library entry. This method is used in all courses offered by cgemod.

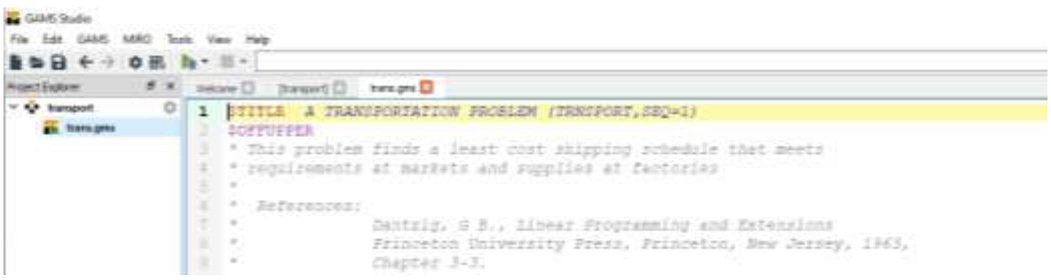
Figure 3.5 User Model Library in Studio



Project file: (internal)
Name: trans
Working directory: C:\cgemod\trans\trans_base
Base directory: C:\cgemod\trans\trans_base
Main file: -none-
Dynamic main file
Parameter file: -none-

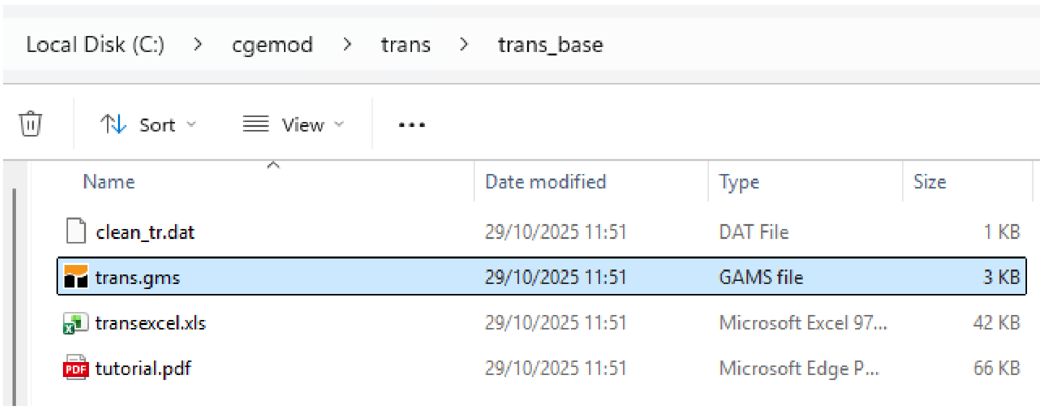
eqN	Name	Model Type	Type	Modeling Points	Equipment
01	transport1	Transport cost problem	LP	Structure of GAMS model	None
02	Syntax	Syntax errors 1 to 4	LP	Debugging a GAMS model	None
03	Execution	Execution errors 1 and 2	LP	Debugging a GAMS model	None
04	transport2	Transport model exercises	LP	Modifying a GAMS model	Adding markets and wholesalers
05	transport3	Transport exercise solutions	LP	Modifying a GAMS model	Adding markets and wholesalers

Figure 3.5 **trans.gms in project Trans**



```
$TITLE  A TRANSPORTATION PROBLEM (TRNSPORT,SEQ=1)
$OFFUPPER
* This problem finds a least cost shipping schedule that meets
* requirements at markets and supplies at factories
*
*  References:
*               Dantzig, G B., Linear Programming and Extensions
*               Princeton University Press, Princeton, New Jersey, 1963,
*               Chapter 3-3.
```

Figure 3.6 **Directory on C:\cgemod_training\Transport**



Name	Date modified	Type	Size
clean_tr.dat	29/10/2025 11:51	DAT File	1 KB
trans.gms	29/10/2025 11:51	GAMS file	3 KB
transexcel.xls	29/10/2025 11:51	Microsoft Excel 97...	42 KB
tutorial.pdf	29/10/2025 11:51	Microsoft Edge P...	66 KB

<u>Running a Model in GAMS Studio</u>

Use the file `trans.gms` in GAMS Studio together with the GAMS tutorial, which will have been downloaded to the working directory (you may want to print off a copy), and the (Introduction to GAMS Studio.pdf) to work through the code for the transport problem making sure that you understand what each line of the code means. Particular attention should be devoted to the following

1. Keywords (in blue in the standard GAMS Studio colouring for *.gms files).
2. The distinction between declaration and assignment (data entry) statements for parameters.
3. The declaration of variables and the different types and attributes of variables.
4. The declaration and assignment of equations.
5. The documentation facilities available within the code file.
6. The model and solve statements.
7. Display statements.

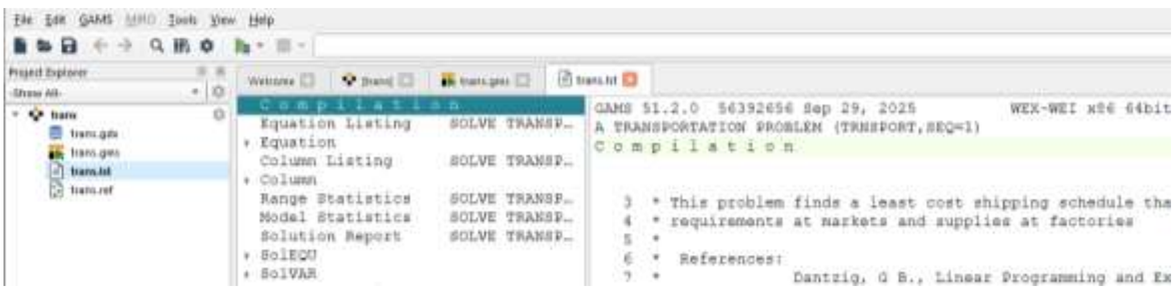
Once you have familiarized yourself with the code, the model should be run using `GAMS>Run with …` (`F10`) ensuring that a GDX and Ref files are created. The run command submits the programme file for compilation and, presuming it compiles without error, executes the programme. (If you wish solely to compilate the model choose `Shift+F10`.)

- The progress of a submitted programme is recorded in the `Process Log` window. Information recorded in this window is very useful and provides an way to debug a programme file (see below). The information from the `Process Log` is recorded as `[Filename].log` and saved in the project directory.
- The layout of the various windows is a matter of personal choice, but for the cgemod courses we assume you adopt the default settings as we do for the illustrations.
- The GAMS output file is returned automatically as a tabbed file in the editor window as `'trnsport.lst'`.

Explore the `Process Log`, or open the `trans.log` file, and explore its contents. This will report, *inter alia*, details of the model, the iteration steps taken to reach a solution, the fact that an optimal value was found and the value of the objective function.

Now look at the `trans.lst` file and explore its contents. This begins by repeating the model's code and then details the equations, the model statistics, a solve summary, the solution equations (`SolEQN`) and variables (`SolVAR`), and finally reports the levels and marginal values for the variable *X* (see Figure 3.7). Note how there are two panes in the editor window, the left pane is an index produced for each list file (`*.lxi`) and the right pane is the list file (`*.lst`). With the transport problem, the list file is short but soon the list files will start to grow, rapidly.

Figure 3.7 List File for the Transport Model



```
C o m p i l a t i o n
  Equation Listing       SOLVE TRANSP…
▸ Equation
  Column Listing         SOLVE TRANSP…
▸ Column
  Range Statistics       SOLVE TRANSP…
  Model Statistics       SOLVE TRANSP…
  Solution Report        SOLVE TRANSP…
▸ SolEQU
▸ SolVAR
```

```
GAMS 51.2.0  56392656 Sep 29, 2025           WEX-WEI x86 64bit
A TRANSPORTATION PROBLEM (TRNSPORT,SEQ=1)
C o m p i l a t i o n


   3  * This problem finds a least cost shipping schedule tha
   4  * requirements at markets and supplies at factories
   5  *
   6  *  References:
   7  *                Dantzig, G B., Linear Programming and Ex
```

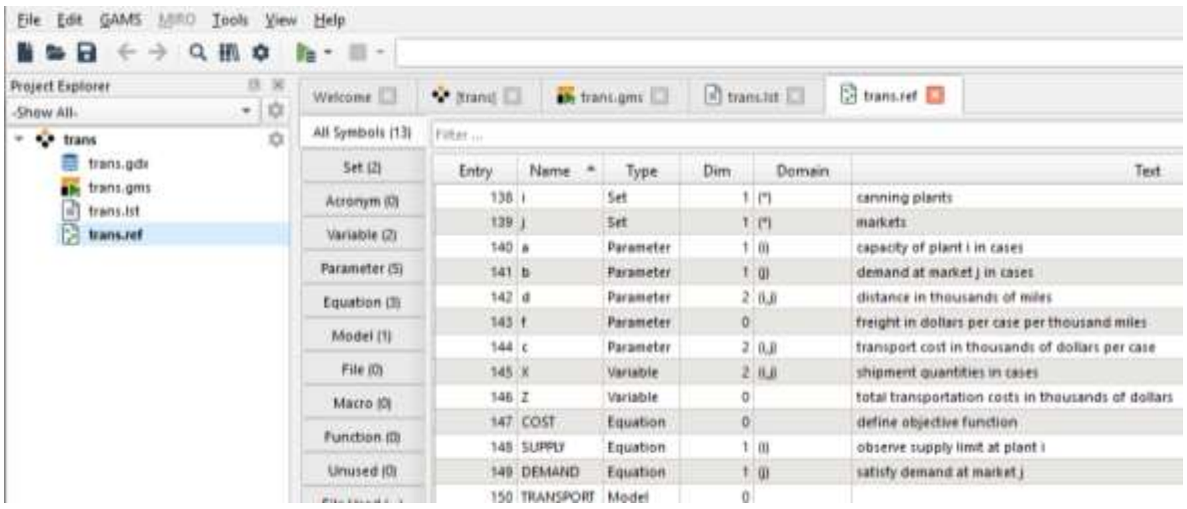
The model file for the transport problem is small, and it is easy to navigate it by scrolling through the file in GAMS Studio. This method will soon become inefficient, so it is good idea to be able to explore a model using some form of indexing; a reference file is the best way to do this for a GAMS programme. A reference file allows you to find components of the programme file by double clicking in the certain places so that the programme file comes to the fore with the cursory in the appropriate places.

For instance, assume you want to find where the parameter `a` was declared. Open the reference file (`trans.ref`) , click on the Parameters tab and then double click on the cell in the row for `a` and the entry for `Declared in` (see Figure 3.8).

Use the reference file to find the following

1. The declaration and definition statement for the COST equations, and where the COST equation was implemented.
2. Where the set *i* was defined and used as a control.
3. Where the model was defined and implemented.

Figure 3.8 Reference File for Transport Model



Entry	Name	Type	Dim	Domain	Text
138	i	Set	1	(*)	canning plants
139	j	Set	1	(*)	markets
140	a	Parameter	1	(i)	capacity of plant i in cases
141	b	Parameter	1	(j)	demand at market j in cases
142	d	Parameter	2	(i,j)	distance in thousands of miles
143	f	Parameter	0		freight in dollars per case per thousand miles
144	c	Parameter	2	(i,j)	transport cost in thousands of dollars per case
145	X	Variable	2	(i,j)	shipment quantities in cases
146	Z	Variable	0		total transportation costs in thousands of dollars
147	COST	Equation	0		define objective function
148	SUPPLY	Equation	1	(i)	observe supply limit at plant i
149	DEMAND	Equation	1	(j)	satisfy demand at market j
150	TRANSPORT	Model	0		

3. Syntax and Execution Error Exercises

These exercises are designed to help you to start solving your own programming errors. **Everyone** who programmes a computer makes errors **every time** they write a programme; it is essential to develop the skills that allow you to find errors easily and correct them. Starting to learn how to debug programmes early in the learning experience is very useful, it is also easier than leaving it until later since you will be working with simple programmes. Indeed, the clichés about 'learning from mistakes' could have been devised by reference to computer programming.

Syntax and execution error exercises can be conducted using a two directories each with several projects.

Syntax Error Exercises

For these exercises we will need a new directory.

1. in Windows Explorer create the directory for the new project, e.g., `C:\cgemod\trans\trans_syn`,
2. choose `File>New Project`,
3. name the newProject `trans_syn` in the editor window,
4. set the Working directory to the new project directory, e.g., `C:\cgemod\trans\trans_syn`
5. choose `File>Save`
6. now choose `File>New` and save the new file (new*.gms) in the `trans_syn` directory.
7. In Studio choose `F6` and the `Introduction to GAMS Studio Lib`, and load the library file `syntax`, which is SeqNr: 2,

There will be **four** *.gms files, labeled `trnserr#.gms` (where # is a number from 1 to 4), that are all perturbations of the basic transport linear programme. These four 'files' contains different types of syntax errors that are intended to be progressively more difficult to solve; in each case the solve statement is **not** implemented because of previous errors.

You should run each of the `trnserr#.gms` files in turn and solve the problems each presents before moving onto the next problem file. For each `trnserr#.gms` file there is a solution file, labeled `trnsol#.gms`. The solution file contains corrections for each of the errors and is annotated to provide explanations. Search for the string 'SMcD' in each solution file. You should modify the `trnserr#.gms` file and make sure you can correct the errors making as little use as possible of the solutions provided.

HINT 1: the output files `*.lst` and `*.log` with provide information about the syntax errors. It is important to learn to read and understand the information in these files. If you click on the red error messages in the `*.log` file they take you to where GAMS encountered the syntax error.

HINT 2: where GAMS encountered a syntax error may not be where the error occurred. The first error in `trnserr1.gms` is case in point.

HINT 3: comparing the file with errors with the solution file can be made easier by using the Pin View.

In Studio there are two obvious options for conducting these four syntax error exercises. We recommend option 2.

1. Open each gms file, `trnserr#.gms`, in the same project, making sure that you change the `Main File` for each exercise in turn; or
2. Open each gms file, `trnserr#.gms`, using `File>Open in a New Project`; this will keep all the files associated with each exercise together in the same project although they will all be saved to the directory `C:\cgemod\trans\trans_syn`. You then need to set the active project.

Execution Error Exercises

In Windows Explorer create the directory for the new project, e.g., `C:\cgemod\trans\trans_exec`. Then, in Studio, create a New Project with the project name `trans_exec` in the directory `C:\cgemod\trans\trans_exec` and then add new file to the `trans_exec` directory. Populate the directory with the file `Execution`, which is SeqNr: 3, from the library `Introduction to Studio Lib`. (If in doubt about the process consult the guidance above).

There are **two** *.gms files, labeled `trnserr#.gms` (where # is a numbers 5 and 6), that are perturbations of the basic transport linear programme. These two files contain different types of execution errors - in each case the solve statement **is** implemented but the model does not execute properly because of previous errors that did **not** include syntax errors. Execution errors are typically more difficult to solve than syntax errors.

You should run each of the trnserr#.gms files in turn, and solve the problems each presents, before moving onto the next problem file. When working with this set of problems in Studio, you can adopt either of obvious two options used for the syntax exercises (with suitable changes to file/directory names).

For each `trnserr#.gms` file there is a solution file, labeled `trnsol#.gms`. The solution file contains corrections for each of the errors and is annotated to provide explanations. Search for the string 'SMcD' in each solution file. You should modify the `trnserr#.gms` file and make sure you can correct the errors making as little use as possible of the solutions provided.

4. Transport Model Exercises

Having completed the syntax and execution error exercises it is relatively simple to begin to modify and extend a simple GAMS programme. There are four exercises

1. Changing Unit Transport Costs
2. Changing Distances
3. A New Market
4. Intermediate Markets

The first two exercises involve simply changing data and examining the changes in the results. The aims of these exercise are to

1. improve understanding of a GAMS programme,
2. develop programme organisation skills, and
3. develop an ability to analyse results.

The first three exercises are relatively straightforward and should be regarded as essential parts of the course. The fourth exercise, adding intermediate markets, requires changes to the equations and is therefore more complex. If you cannot complete the fourth exercise easily, it is recommended that you return to it later.

Changing Unit Transport Costs

Create the directory `C:\cgemod\trans\unit` with a New Project with the project name `unit` together with a new file. Populate the directory with files for the transport model, i.e., `Transport1` which is are SeqNr 1. The basic file is identical to the `trans.gms` file in the GAMS Model Library but other files are also downloaded to the directory.

Start from the file '`trans.gms`' and save the file with a new name, e.g., `unitcost.gms`. Then experiment with a series of systematic changes in the per unit transport cost. How do these changes affect deliveries to different markets and total transport costs?

Changing Distances

Create the directory `C:\cgemod\trans\dist` with a New Project with the project name `dist` together with a new file. Populate the directory with files for the transport model, i.e., `Transport1` which is are SeqNr 1. The basic file is identical to the `trans.gms` file in the GAMS Model Library but other files are also downloaded to the directory.

Start with the file '`trans.gms`' and save the file with a new name, e.g., `distance.gms`. Then experiment with a series of systematic changes in the distances between the plants and the markets. How do these changes affect deliveries to different markets and total transport costs?

HINT: if you have used `F10` to run the model the results will all be available in the file `distance.gdx`. The results of interest will be the levels results for the variables X and Z. You can easily copy and paste these into Excel to make the comparisons easy. (NB: the attributes options.)

New and Intermediate Markets

Create the directory `C:\cgemod\trans\market` with a New Project with the project name `market` together with a new file. Populate the directory with files for the transport model, i.e., `market` which is are SeqNr 4. The basic file is the `trans.gms` file in the GAMS Model Library but other files are also downloaded to the directory.

Start with the file '`trans.gms`' and save the file with a new name, e.g., `newmkt.gms`. Then adapt the GAMS code to include another market. Assume the new market is Charlestown and the demand at that market is for 225 units. It is left to you to decide how far Charlestown is from Seattle and San Diego, and by how much to increase production at either or both of Seattle and San Diego (the combined increase in total production capacity must be at least 175 units; it can be more). Experiment with changing various parameters. How do the changes you have made affect the results?

Be systematic and collect your results in one file.

A worked solution is available in the working directory as `newmkt_sol.gms`.)

Intermediate Markets

The necessary files are in the directory `C:\cgemod\trans\market`.

Many marketing systems are characterized by intermediate markets, e.g., wholesale markets. This exercise adapts the standard transport problem to introduce two intermediate markets, and hence the production by factories should first be sent to the intermediate markets and then transshipped to the final destinations. The transport costs should be different between the factories and the intermediate markets and between the intermediate markets and the final destinations. It is left to you to determine names for the intermediate markets, the locations of the intermediate markets and hence the distances between the various nodes in the model, the transport costs between nodes, and the quantities supplied by each factory and demanded at each destination.

Start from the file '`trans.gms`' and save the file with a new name, e.g., `wholesale.gms`. Then adapt the GAMS code to include intermediate markets. Note this exercise requires that you modify equations so you should first write out the new model BEFORE starting to code. Experiment with changing various parameters. How do the changes you have made affect the results?

A worked solution is available in the working directory as `wholesale_sol.gms`.