



An Introduction to GAMS with GAMS Studio

(General Algebraic Modelling System)

Table of Contents

1. Introduction.....	3
GAMS	3
GAMS Studio.....	4
2. GAMS with GAMS Studio for MS-Windows.....	5
Installing GAMS.....	5
Basic Settings.....	7
3. GAMS Studio.....	10
Using GAMS with GAMS Studio	10
A First Programme in GAMS Studio.....	11
4. Testing a GAMS Installation	16
5. Project Explorer	19
Controlling the Choice of Active Project.....	19
Moving Files between Projects and Changing the Main File	20
Open a File from Project Explorer.....	21
Open Terminal	21
Open log tab.....	22
Open Location.....	22
6. Help with Studio	24
7. Search and Replace	26
Selection of Files or Contents (Selection) to be Searched.....	27
Hints.....	28
8. Opening and Using GDX files.....	29
GDXViewer	29
Filtering in GDX Viewer	31
Output Data from GDX Viewer.....	34
9. Reference files	36
10. Configuring Studio.....	39



Practical CGE Modelling: Introduction to GAMS with GAMS Studio

Settings.....	39
Command Line Options.....	40
Configuring Studio to Run Commands by Default.....	42
11. Model Libraries.....	43
Using a GAMS User Model Library.....	44
12. Comparing Files.....	51
13. Studio with Modular Projects	52
File and Directory Structure.....	52
Search & Replace in Modular Models.....	57
14. PIN View	59
15. Notes on Debugging a GAMS Programme in Studio.....	62
Compilation Errors.....	62
Execution Errors	63

1. Introduction

GAMS

The GAMS (General Algebraic Modelling System) suite consists of the base GAMS module and a collection of solvers. GAMS is a high level programming language amongst whose objectives is to allow programmers to prepare programmes that are transparent. When a programme is run GAMS converts it into code compatible with a specified solver, executes the programme by calling the solver and writes a report file back. GAMS is attractive because

- it can use a range of specialised solvers without requiring the user to know their specific syntax;
- the separation of data and the logic of a problem allows the size of the problem to be increased without increasing the complexity of the representation;
- the programme is its own documentation; and
- it looks after a number of common programming problems, like other high level languages e.g., dimensionality.

“GAMS was developed to [overcome a series of mathematical programming problems] by

- providing a high-level language for the compact representation of large and complex models;
- allowing changes to be made in model specifications simply and safely;
- allowing unambiguous statements of algebraic relationships; and
- permitting model descriptions that are independent of solution algorithms” (Brooke *et al.*, 1998, p1).

A major, if not the major, use of GAMS is for optimisation models. Examples of such models are linear programming (LP) and Mixed Integer (MIP) models. CGE models can also be formulated as optimisation problems.

Among the solvers available are

Solvers

- BDMLP LP
- MINOS NLP

- CONOPT3/4 NLP optimizer
- CPLEX LP and Mixed Integer Programme solver
- LINDO nonlinear problems with continuous and/or discrete variables
- SNOPT Sparse Nonlinear Optimizer
- DICOPT Mixed integer Nonlinear Programming
- PATH Mixed complementarity problem solver
- PATHNLP NLP optimiser)

To use GAMS you need a programme file. All GAMS programme files are prepared as text files and saved as `[filename].gms`. (Note: it is not necessary to limit the filename to 8 characters, or to avoid spaces, but we find it useful to avoid excessively long file names). The easiest way to prepare a GAMS programme is by using a modern text file editor or Integrated Development Environment (IDE), that can also read GAMS output files. Standard GAMS output files are written (automatically) to disk as `[filename].lst/.log` files where the filename is the same filename for the `[filename].gms` file..

GAMS is available for is available for MS-Windows, LINUX and Mac OS X (see <https://www.gams.com/download/>).

GAMS has a video that provides a generic overview of GAMS ([Informs 2020 Workshop - Part 1: Introduction to GAMS - YouTube](#)), that you may find informative.

GAMS Studio

Many programme file editors available (see <http://www.mpsge.org/inclib/tools.htm> for some suggested EMACS); and no doubt the suggestion that “*Editors are like mother ducks - once you've identified yours...*” is likely to hold.

We recommend GAMS’s new text editor: **GAMS Studio**.¹

¹ We have verified the Mac OS X version of Studio, but we do not use Apple Macs and therefore cannot be certain that all the functionality of the MS-windows version is available in the Mac OS X version, and vice versa.



2. GAMS with GAMS Studio for MS-Windows

Installing GAMS

The latest version of GAMS is available as a single exe file. Installation instructions from GAMS and some additional instructions are available from the web site (<http://www.gams.com>). To read the installation instructions a copy of Adobe Acrobat Reader is required (available free from <http://www.adobe.com>). It is also valuable to have access to file compression software, e.g., WinZip (<http://www.winzip.com>), RAR (<http://www.win-rar.com>), both of which can be used for a period without charge; we will assume you are using WinZip. We are assuming that you are using Windows 10/11 in 64 bit mode.

The size of models, i.e., variables and equations, and solvers that can be used depend on the license you hold. For CGE analysis we use the PATH and CONOPT solvers, with a preference for the PATH solver for our main models. GAMS also offers a ‘demo’ license so that new users can test GAMS before purchasing (https://www.gams.com/try_gams/).²

Before installing GAMS, you will need a license file. If your institution has a valid GAMS license you should obtain a copy of the license file. If your institution does NOT have a valid license you will need to purchase a license file (https://www.gams.com/buy_gams/) or apply for a demo license (https://www.gams.com/try_gams/). If purchasing GAMS, you will need the PATH solver for the CGE courses and the PATH and CONOPT solvers for the SAM estimation course.

If your course requires a special license file, cgemod will supply you with a copy.

The installation process is standard for different versions of Windows. In brief the installation process for Windows 10/11 is as follows

² “Our demo licenses are valid for 12 months, and are ideal for those who want to get a first impression of GAMS.

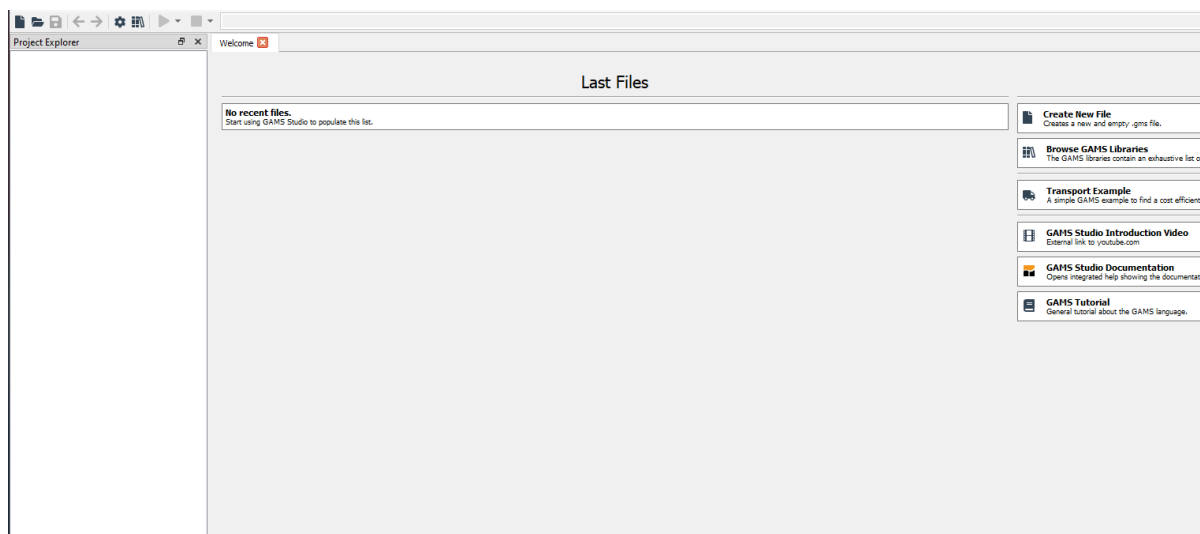
All language features are enabled, and the restrictions listed below are quite generous. You will be able to run your first experiments before hitting the limits too soon.

- You can generate and solve linear models (LP, RMIP, and MIP) that do not exceed 2000 variables and 2000 constraints
- For all other model types the model cannot be larger than 1000 variables and 1000 constraints.
- Etc.” (https://www.gams.com/try_gams/)

- i) The latest version of GAMS is available from GAMS at <http://www.gams.com/download/>. It is necessary to select the version of GAMS consistent with your operating system: in this document it is assumed that you are using a version of Windows.
- ii) Read the installation notes provided on the GAMS website (<https://www.gams.com/download/>).
- iii) Copy the file 'windows_x86_64.exe' to a directory/folder³, e.g., TEMP, on your C drive.
- iv) Run the programme 'windows_x86_64.exe'. It is WISE for you to install GAMS in a directory on the top level of the C Drive (say C:\GAMS); this is **not** the default chosen by the installation software, but it does have some advantages. (It is assumed for all the exercises etc., detailed in this and related documents that GAMS has been installed into the directory C:\GAMS. If not, you will need to adjust the path appropriately.)
- v) Follow the installation instructions as they appear on the screen.
- vi) When installing GAMS you will be asked to choose the editor. There are two options GAMS Studio, the default, or GAMSIDE; **choose** GAMS Studio.
 - a. You will be asked for a license file: point the installation programme to the license file (`gamslice.txt`) (full or demo version).
- vii) The installation of a Windows version of GAMS will conclude by offering the option to launch GAMS Studio; select this option. Figure 2.1 illustrates how GAMS Studio should appear on your screen.
- viii) The final step will be to run a few models to check the installation was successful, this is explained below.

This brief description is NOT a substitute for reading the installation guide or using the Help facility in GAMS Studio.

³ A 'directory' and a 'folder' are the same; directory is the 'old' DOS name while 'folder' was the name used by Apple in the 1980s. Directory is the name used in this document.

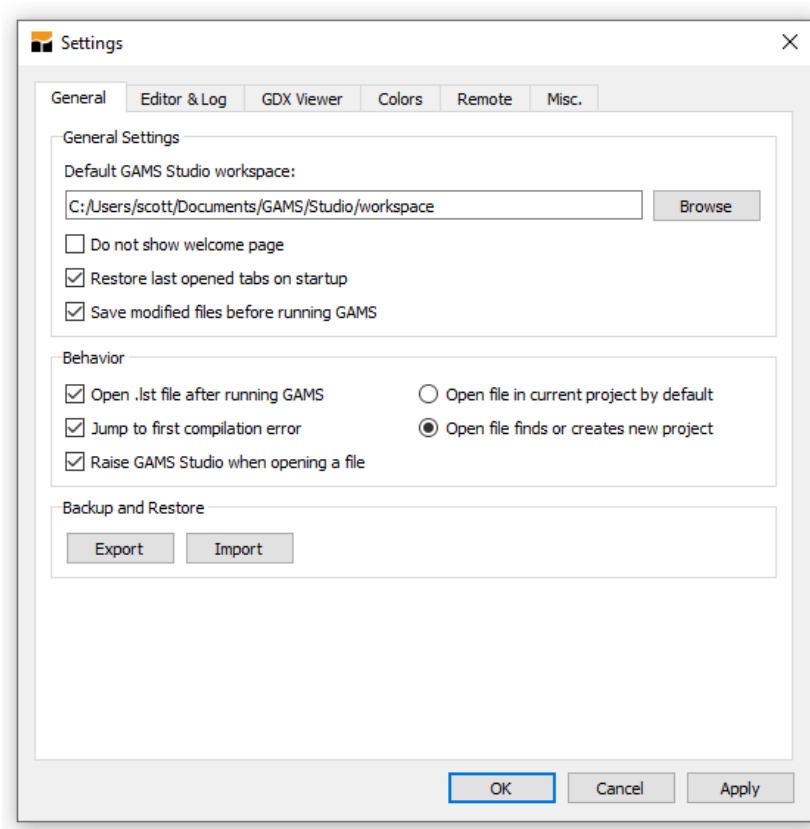
Figure 2.1 GAMS Studio when First Opened

Basic Settings

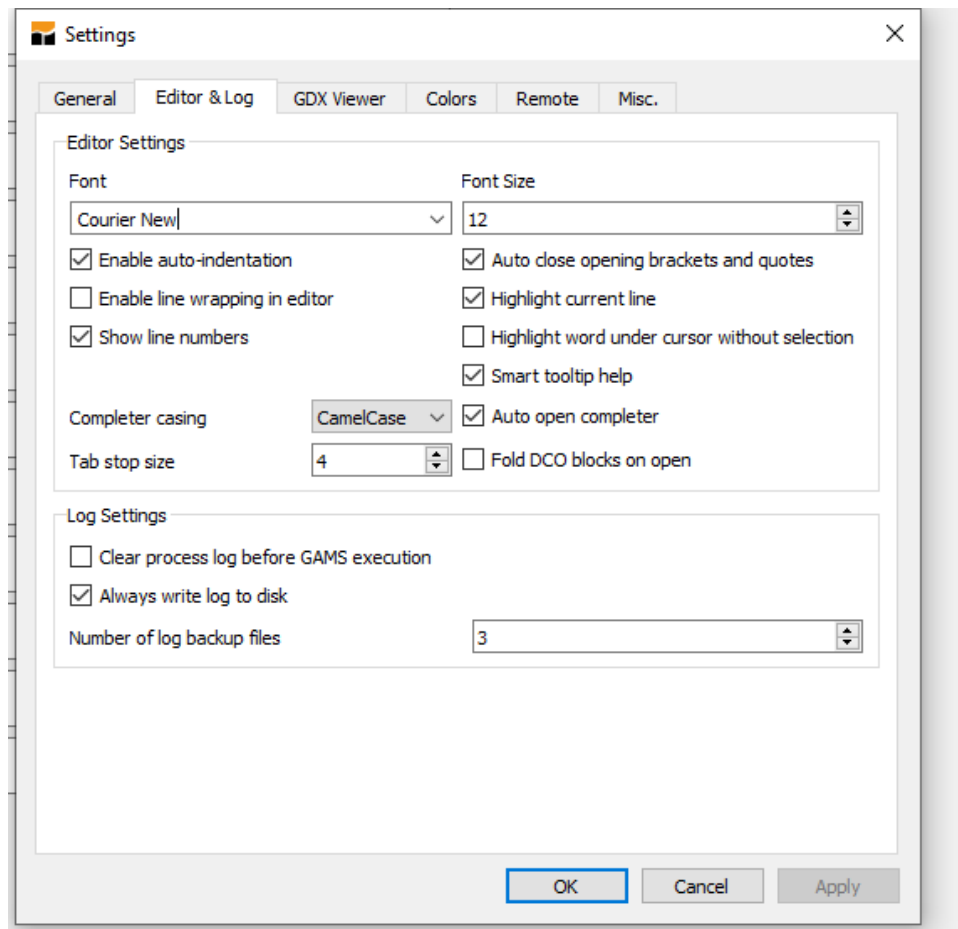
GAMS Studio provides multiple options for configuring the editor to suit the preferences of individual users. As users gain experience the selected preferences will develop, but when starting it is helpful to have a set of default settings. We assume in this guide that the default settings are those identified in this section.

Open the settings dialogue: use `File>Settings`, or `F7` or the settings wheel (⚙️). This should open with the ‘General’ settings tab in view. Make sure that the default GAMS Studio workspace is `C:/Users/****/Documents/GAMS/Studio/workspace`, where `****` identifies the user (or the equivalent on your PC). Make sure that the option ‘Open file finds or creates a new project’ is selected. Make sure the other settings in this tab are those shown in Figure 2.2.

We will subsequently, section 10, advise you to change the option ‘Open file finds or creates a new project’ to the option ‘Open file in current project by default’.

Figure 2.2 **GAMS Studio General Settings**

Selected the 'Editor & Log' tab. Set the font to Courier New and the font size to 12; GAMS needs to use a fixed pitch font, e.g., Courier New, and 12 points for the font size is a good starting point (the view size can be easily adjusted in Studio without needing to modify the settings). Make sure the other settings in this tab are those shown in Figure 2.3.

Figure 2.3 GAMS Studio Editor and Log Settings

For the other tabs in the settings dialogue the defaults can be left unchanged. At this stage do not worry about understanding the setting selections; they will be more than adequate for some time and for all the courses offered by cgemod.



3. GAMS Studio

(General Algebraic Modelling System STUDIO)

Using GAMS with GAMS Studio

To use GAMS, you need a programme file. All GAMS programme files are prepared as text files and saved as `[filename].gms`.⁴ To prepare a GAMS programme it is necessary to use a text file editor. In the past it was necessary to use a text file editor to write the programme file and read the resulting (list) file(`[filename].lst`) and to run GAMS from DOS. GAMS Studio allows you to do all this from a single Windows based (integrated) interface. This makes things very much easier, but there are one or two features that are initially a little tricky to grasp.

Why advocate using Studio for our courses?⁵ The reasons are simple.

1. Studio comes packaged with the GAMS software and therefore only requires the installation of GAMS to give users an editor without needing to install extra programmes.
2. Using a single editor means that all participants are working in the same software environment, which makes it easier to prepare material and provide support.
3. Studio provides all the utilities and routines that are required for cgemod courses.
4. Studio is the future editor for GAMS.
5. Studio can be used on MS-Windows, LINUX and Mac OS X platforms.⁶

More importantly however, we choose Studio because we like many of the features offered by Studio and therefore, we use Studio for our day-to-day work.

The guidance in this document is intended to supplement the guides available from the GAMS web site (see [An Introduction to GAMS Studio - YouTube](#)) or from the link in the ‘Getting Started’ menu to ‘**GAMS Studio Introduction Video**’. The GAMS video for Studio

⁴ Notes: (i) It is no longer necessary to limit filenames to 8 characters and to avoid spaces – but it may be good practice to avoid over long filenames and directory names so as to avoid long paths. (ii) It is also good practice to use multiple directories to avoid difficulties identifying related files.

⁵ We did not transition to Studio until the software has matured. Early versions of Studio, from 2018 until 2021, did not contain functionality that we consider essential. This has changed and, moreover, the GAMS corporation has been very responsive to feedback, from users, in the evolution of Studio.

⁶ We have only been able partially to test Studio in Mac OS X and LINUX environments: Mac OS X and LINUX are not our working environments and without using them for long periods we cannot fully test Studio in those environments.

is dated; Studio has evolved appreciably since the video was made. We encourage you to use GAMS's official guides (they may not include all the latest features of GAMS Studio because Studio is an evolving environment).

When you first open Studio, the screen will appear similar to that in Figure 2.1. At the top of the screen is a standard toolbar with dropdown menus, on the left-hand side is a window with the title 'Project Explorer' and in the centre is the 'Welcome' window. The 'Welcome' window has three menus 'Last Files', 'Getting Started' and 'Further Help'. We will explore these components in more depth later in this guide.

The 'Getting Started' menu provides a link to the 'GAMS Studio Introduction Video'; this is worth viewing. The exercises associated with this introduction to GAMS Studio and the pre-course exercises are based on the 'GAMS Tutorial', so while worth viewing it may be better to do so in conjunction with the exercises.

We will start by opening a GAMS programme. For this we will use the Transport Example provided by GAMS and which is used in the exercises that accompany this set of training materials.

A First Programme in GAMS Studio

Click on the third item, 'Transport Example', in the 'Getting Started' menu. The view will change to that illustrated in Figure 3.1 (Figure 3.1 has a few layout adjustments to make it easier to see the content in the figure). There are three important things to note:

1. The 'Project Explorer' has been populated with a directory/folder symbol 'trnsport', which identifies that 'trnsport' is a PROJECT, with a file 'trnsport.gms' identified as part of the project. The Project Explorer details the files currently in 'use'.
2. A tab 'trnsport.gms' has been open in the main part of the screen; the orange symbol indicate that is the file open for editing. This is the window in which files are edited.
3. A 'Process Log' has appeared as a window on the right of the screen. This is where Studio will report the processes that are and have been implemented.

Running the transport example will allow you to envisage the basic operation of Studio. (See the video about Studio – there is a link from the

Figure 3.1 A First Programme in Studio

```

GAMS Studio
File Edit GAMS MIRO Tools View Help
Project Explorer Welcome transport.gms Process Log
transport
  transport.gms
1 $title A Transportation Problem (TRNSPORT,SEQ=1)
2
3 $onText
4 This problem finds a least cost shipping schedule that meets
5 requirements at markets and supplies at factories.
6
7
8 Dantzig, G B, Chapter 3.3. In Linear Programming and Extensions.
9 Princeton University Press, Princeton, New Jersey, 1963.
10
11 This formulation is described in detail in:
12 Rosenthal, R E, Chapter 2: A GAMS Tutorial. In GAMS: A User's Guide.
13 The Scientific Press, Redwood City, California, 1988.
14
15 The line numbers will not match those in the book because of these
16 comments.
17
18 Keywords: linear programming, transportation problem, scheduling
19 $offText
20
21 $set
  
```

Process Log

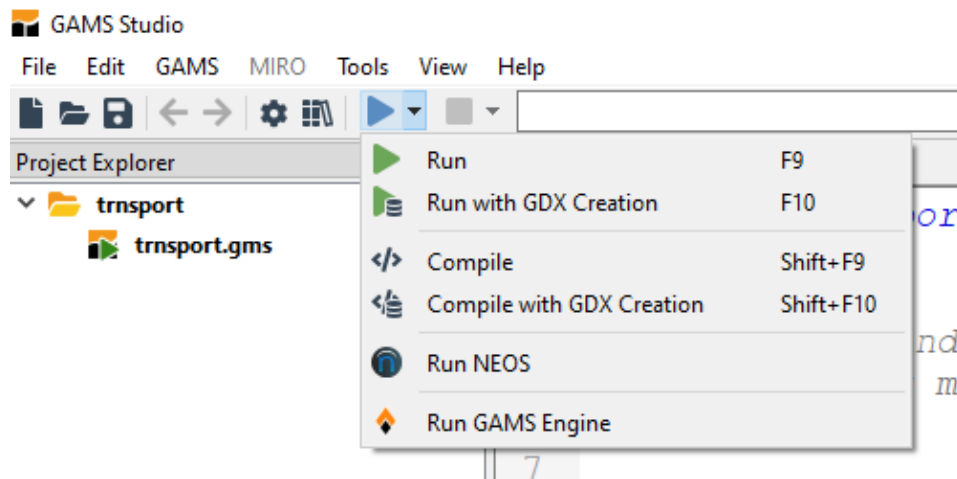
```

Info [11:4
Info [11:4
Info [13:1
Info [13:1
  
```

For typical users of Studio there are 4 options for ‘running’ a programme; these can be accessed in three different ways that produce similar results (see Figure 3.2). The selected option requires clicking on the small black arrowhead next to the green triangle on the toolbar to open the dropdown menu; a slighted extended menu can be obtained by clicking on the GAMS menu. The 4 standard options are Run (F9), Run with GDX Creation (F10), Compile (Shift+F9) and Compile with GDX Creation (Shift+F10); our experience using Studio is that we almost always use Run with GDX Creation (F10) or, less often, Compile with GDX Creation (Shift+F10). We encourage using the keyboard commands identified in the menus.

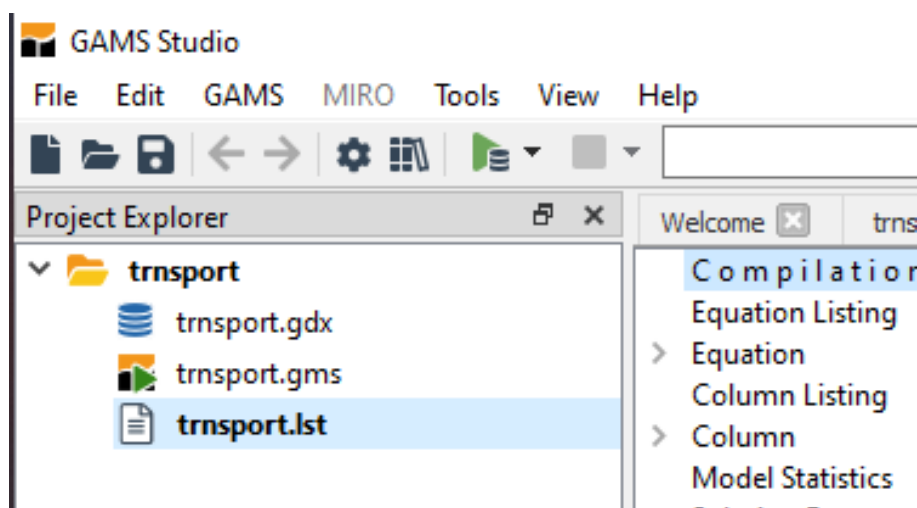
Press F10 (Run with GDX Creation). The contents of all three windows will change. We rarely use F9 or Shift+F9 since the creation of the GDX file is so valuable and takes no noticeable time, but we do use Shift+F10, especially when trying to debug a programme.

Figure 3.2 Running GAMS from Studio



The Project Explorer now identifies three files in the directory transport: the gms file – `transport.gms` – that contains the programme, the lst file – `transport.lst` – that contains a listing of the output when the programme ran; and the.gdx file – `transport.gdx` – that contains information about the data, variables, equations, results etc., generated by GAMS. Note how the file identified in bold – `transport.lst` – is file/tab open in the editor window.

Figure 3.3 The Transport Problem in Studio – Project Explorer



It is important to note that **GREEN** triangle on top of the Studio symbol is associated with the gms file – `transport.gms`. The **GREEN** triangle identifies the ‘main file’ in each currently open PROJECT. The active PROJECT is identified by the directory that is in **bold**.

If the user chooses to run a programme, e.g., by using F10, then the programme that will run will be the **main file**, i.e., the file with the **GREEN** triangle, in the active PROJECT, irrespective of the file currently being viewed in the EDITOR window.

Figure 3.4 **The Transport Problem in Studio – Process Log**

```

--- Starting compilation
--- transport.gms\(66\) 3 Mb
--- Starting execution: elapsed 0:00:00.043
--- transport.gms\(43\) 4 Mb
--- Generating LP model transport
--- transport.gms\(64\) 4 Mb
--- 6 rows 7 columns 19 non-zeroes
--- Executing CPLEX \(SolveLink=2\): elapsed 0:00:00.187

IBM ILOG CPLEX 35.1.0 r82a9585 Released Apr 29, 2021 WEI x

--- *** This solver runs with a community license.
--- GMO memory 0.50 Mb (peak 0.50 Mb)
--- Dictionary memory 0.00 Mb
--- Cplex 20.1.0.1 link memory 0.00 Mb (peak 0.00 Mb)
--- Starting Cplex

Version identifier: 20.1.0.1 | 2021-04-07 | 3a818710c
CPXPARAM_Advance 0
CPXPARAM_Simplex_Display 2
CPXPARAM_Threads 1
CPXPARAM_MIP_Display 4
CPXPARAM_MIP_Tolerances_AbsMIPGap 0
Tried aggregator 1 time.
LP Presolve eliminated 0 rows and 1 columns.
Reduced LP has 5 rows, 6 columns, and 12 nonzeros.
Presolve time = 0.00 sec. (0.00 ticks)

Iteration      Dual Objective      In Variable
1              73.125000      x(seattle,new-york) demand(
2              119.025000      x(seattle,chicago) demand
3              153.675000      x(san-diego,topeka) deman
4              153.675000      x(san-diego,new-york) supply

--- LP status (1): optimal.
--- Cplex Time: 0.00sec (det. 0.01 ticks)

Optimal solution found
Objective:      153.675000

--- Reading solution for model transport
--- Executing after solve: elapsed 0:00:00.622
--- transport.gms\(66\) 4 Mb
--- GDX File C:\Users\scott\Documents\GAMS\Studio\workspace\
*** Status: Normal completion
--- Job transport.gms Stop 07/26/21 13:38:03 elapsed 0:00:00.

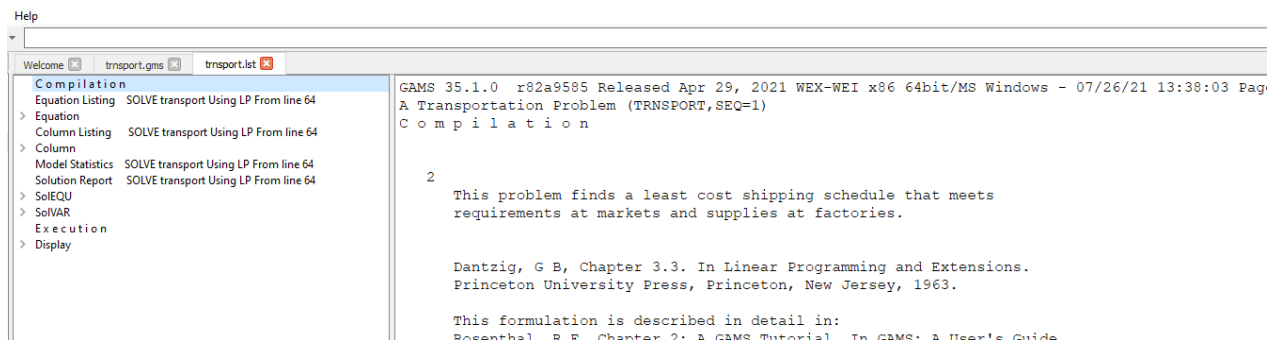
```

Multiple projects can be open in the Project Explorer. The current active project will be the project in bold and moving between projects can be achieved by double clicking on the

any file in the project that is to become the active project. The contents of the ‘Project Explorer’ window will be examined in more depth below.

The ‘Process Log’ window records what has happened when a programme is run. This programme ran to a ‘Normal completion’ (see the second last line). Clicking on BLUE items in the log opens the l(i)st file – `transport.lst` – and moves the cursor to specific places in the file. Items in GREEN open the identified file in the editor window, e.g., ‘`transport.gms`’ and ‘`transport.gdx`’, and where appropriate at the point in the linked file references in the lst file. Click on the third last line of the Process Log – ‘GDX File ...’; this will open the GDX file. Later there will be items in RED – syntax errors – that when clicked on will open files in the editor window at the point in the file where the error exists.

Figure 3.5 The Transport Problem in Studio – Editor



Finally, there is the editor window. By default, the editor window will display the l(i)st file – `transport.lst` – in two parts. The left-hand part is an index file that can be used to navigate the list file, e.g., `SOLVAR` links to the report of the solutions for the variables (try double clicking on `SOLVAR` and then clicking on the arrowhead to open a sub menu and click on one item in the sub menu. As you get to know programmes this will be one of several ways to explore what has happened when running a programme.

You can adjust the layout of the windows in Studio to suit your preferences; this guide will continue with the default configuration.


4. Testing a GAMS Installation

To test your GAMS installation, you can run a number of sample programmes that are provided by GAMS. These programmes are stored in one of the Model Libraries included when GAMS was installed.

In Studio select `GAMS > Model Library Explorer` or press `F6`. The window displayed in Figure 4.1 will appear. Each tab links to a specific library, each of which contains many sample programmes. To test the installation, we will select and run 4 programmes. These are

1. `trnsport` (LP :objective value: 153.675).
2. `chenery` (NLP: objective value: 1058.9).
3. `bid` (MIP: optimal solution: 15210109.512).
4. `procsol` (MINLP: optimal solution 1.31164890297).
5. `scarfmcp` (MCP: no objective function).

Figure 4.1 Model Library Explorer

 Model Library Explorer

Search:

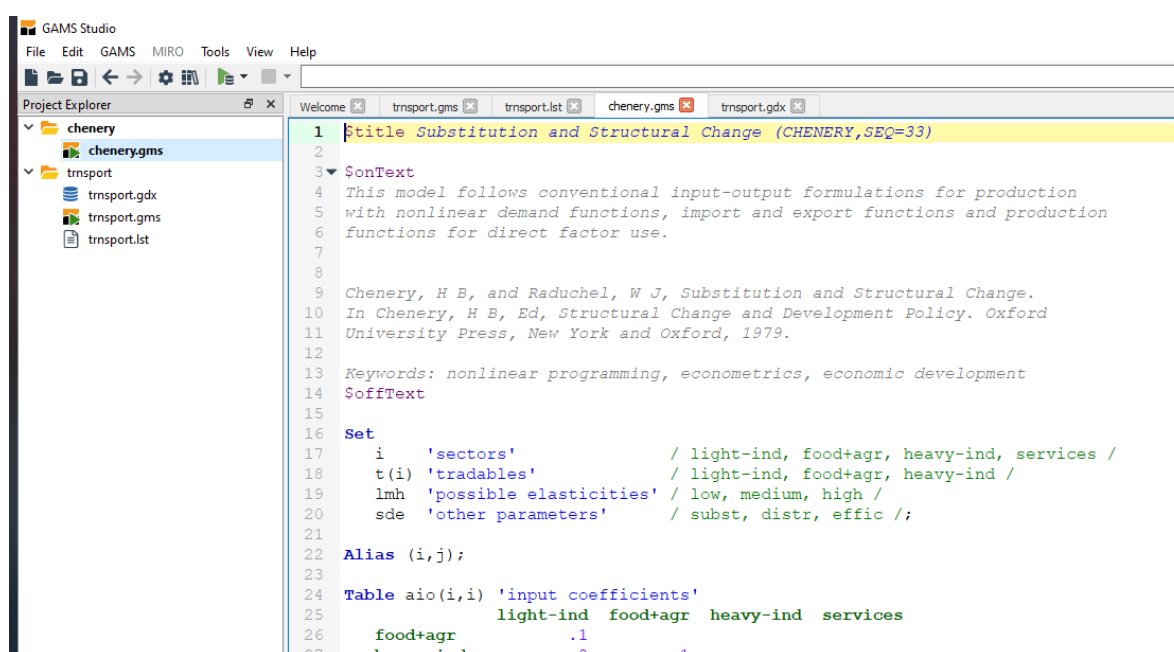
Model Library (428)		Test Library (849)		API Library (61)		Data Utilities Library (140)		EMP Library (104)		FIN Library (42)		NOA Library	
SeqNr	Lic	Name	Application Area	Type	Contributor	Description							
064	D	ABEL	Macro Economics	NLP	Kendrick, D	Linear Quadratic Control Problem							
208	D	ABSMIP	Mathematics	MIP	GAMS Develop	Discontinuous functions abs() min() max() sign() as MIPs							
088	D	AGRESTE	Agricultural Economics	LP	Kutcher, G P	Agricultural Farm Level Model of NE Brazil							
008	D	AIRCRAFT	Management Science and OR	LP	Dantzig, G B	Aircraft Allocation Under Uncertain Demand							
189	C	AIRSP	Stochastic Programming	LP	Dantzig, G B	Aircraft Allocation							
196	C	AIRSP2	Stochastic Programming	DECIS	Dantzig, G B	Aircraft Allocation - stochastic optimization with DECIS							
060	D	AJAX	Management Science and OR	LP	CDC	Ajax Paper Company Production Schedule							
124	D	ALAN	Finance	MIN..	Manne, A S	A Quadratic Programming Model for Portfolio Analysis							
165	D	ALKYL	Chemical Engineering	NLP	Berna, T J	Simplified Alkylation Process							
396	D	ALLBASES	Micro Economics	MIP	Dantzig, G B	Enumerate all Feasible Basic Solutions of the Transportation							
170	D	ALPHA...	Recreational Models	MIP	de Wetering,	Alphametics - a Mathematical Puzzle							
031	C	ALUM	International Trade	MIP	Brown, M	World Aluminum Model							
074	D	AMPL	Management Science and OR	LP	Fourer, R	AMPL Sample Problem							

The first programme – `trnsport` – has already been run so all that you need to do is verify that a ‘Normal Completion’ has been recorded and that an ‘Optimal solution found’ with an objective value of 153.67500. These can all be found in the ‘Process Log’ or the ‘Solution Report’ in the file `trnsport.lst`.

Open the setting dialogue box, F7, and in the General tab change the selected option is ‘Open file finds or creates a new project’ rather than the alternative ‘Open file in current project by default’.

In the Model Library Explorer, select the tab ‘Model Library’ and search for chenery; clicking on the column labels sorts the columns while using the Search box finds a file if you know its name. Click anywhere on the line for the model chenery; the results are shown in Figure 4.2.

Figure 4.2 Model Chenery in Studio



Note the following:

1. The programme – chenery .gms – has opened in the editor
2. The project/group chenery has been added to the Project Explorer
3. The project chenery has been defined as the active project – the project folder is in bold

Run the programme ‘chenery .gms’ using F10 (or the equivalent) and note the changes in Studio. Verify that a ‘Normal Completion’ has been recorded and that an ‘Optimal solution found’ with an objective value of 1058.9199. HINT: look in lst file.

Practical CGE Modelling: Introduction to GAMS with GAMS Studio

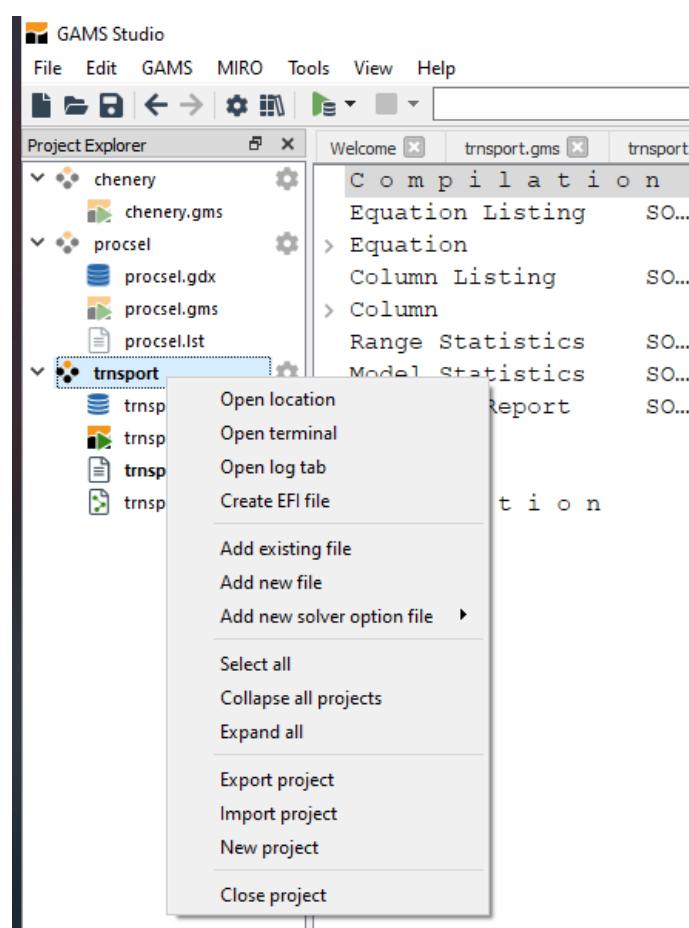
Repeat the process for the models 'bid', 'procel' and 'scarfmcp' (NB: procel and scarfmcp are in the many Model Library but that the search option searches across all libraries so it is easily found); in each case verify that a 'Normal Completion' has been recorded and that an 'Optimal solution found' with an objective values given above.

Note how for each programme a new project is created. This means that in Studio multiple projects can be open at the same time.

5. Project Explorer

There are many options available to the user within the Project Explorer in Studio. These are typically accessed by right clicking on a project name or a file within the project visible in the Project Explorer window; the options available will depend on the item that is right clicked. Some of these options are self-explanatory, e.g., Select All, Close All/File, Expand All, Close File/Group; you should choose them and observe the consequences. Some of the options are discussed further below.

Figure 5.1 Right Click on Project Name



Controlling the Choice of Active Project

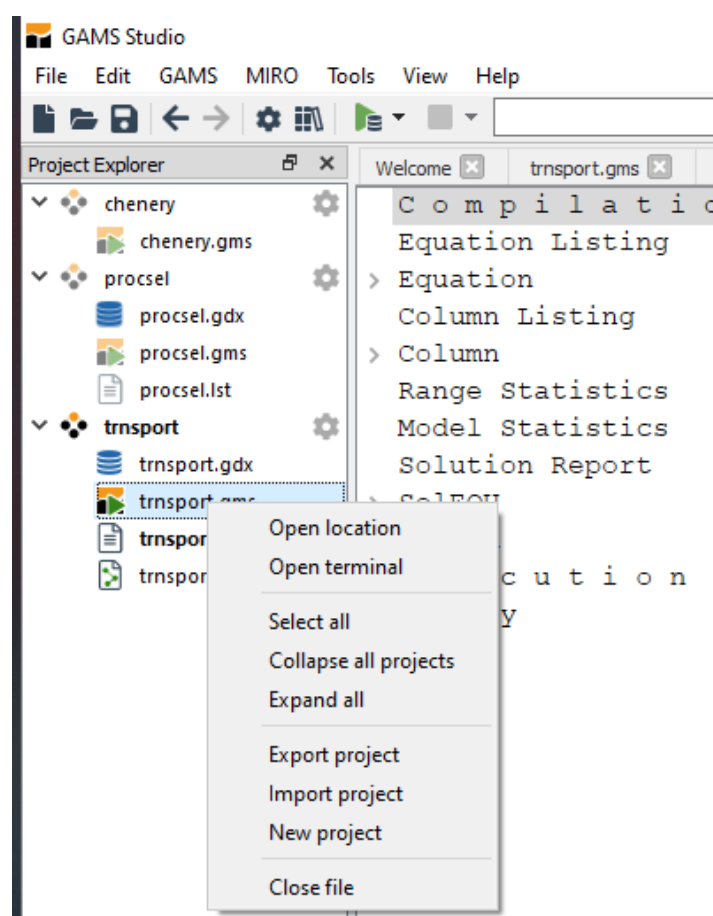
The active project is always the project directory whose label is in **bold**. You can change the active project by double clicking on a file in the project you want to be active. If you change

the active project, you will also change the file that will be run when pressing F9/F10, i.e., the **main** file with the **GREEN** triangle in the active project ‘directory’.

Moving Files between Projects and Changing the Main File

You can move files between projects in the Project Explorer. For instance, assume you want to move the file `procel.gms` into the project `chenery`. Click on the file ‘`procel.gms`’ and drag it into the `chenery` directory. You now have two `gms` files in the `chenery` directory, one is the ‘main’ file – `chenery.gms`. If you want to make ‘`procel.gms`’ the ‘main’ file, right click on the file `procel.gms` and choose the option – Set as Main File. Note how the **GREEN** triangle moves to the file `procel.gms`.

Figure 5.2 Right Click on File Name





The main file of the active project will always be the programme that is run when choosing a run option, e.g., F9 or F10.

SUGGESTION: you can have multiple gms files in a single project and switch between them using the `Set as Main File` option. While this is viable, and GAMS will allow the user to do so, we discourage the practice especially for new users. We suggest that each project should have one, or at most a few, gms files. In the Practical CGE, single country CGE and global CGE courses we have only one gms file in each project; in the single country and global CGE courses we have two projects linked to the same directory with a single gms file in each project. (This is consistent with our practice when using our advanced models for conducting research and/or consultancy: we only have one gms files in each project but two gms files in one directory with two projects. This has some benefits when using command line options, e.g., `save` and `restart`.)

Open a File from Project Explorer

Files can be opened from Project Explorer by right clicking on the project and selecting the file to be opened, e.g., `Add existing file` and `Add new file`. ‘`Add existing file`’ takes you to the location where the project files are stored and allows you to open an existing file; note that you will need to select the file type (bottom right of the dialogue box) or type in the file name. **HINT:** in the File name box the wildcard option (*) is useful, e.g., `*.inc`, `*.gms`, etc., makes only those files with specified file types visible. ‘`Add new file`’ takes you to the location where the project files are stored and allows you to create a new file of the selected type; note that you will need to select the file type in the bottom dialogue box.

Our preference is to open files using the keyboard shortcuts `Ctrl+O` or `Ctrl+Shift+O`, or `Ctrl+N`, i.e., `File > Open` or `File > Open in a New Project` or `File > New` menus. When a programme is run Studio opens certain files by default whereas others need to be selected. (Space below to the special and useful files.)

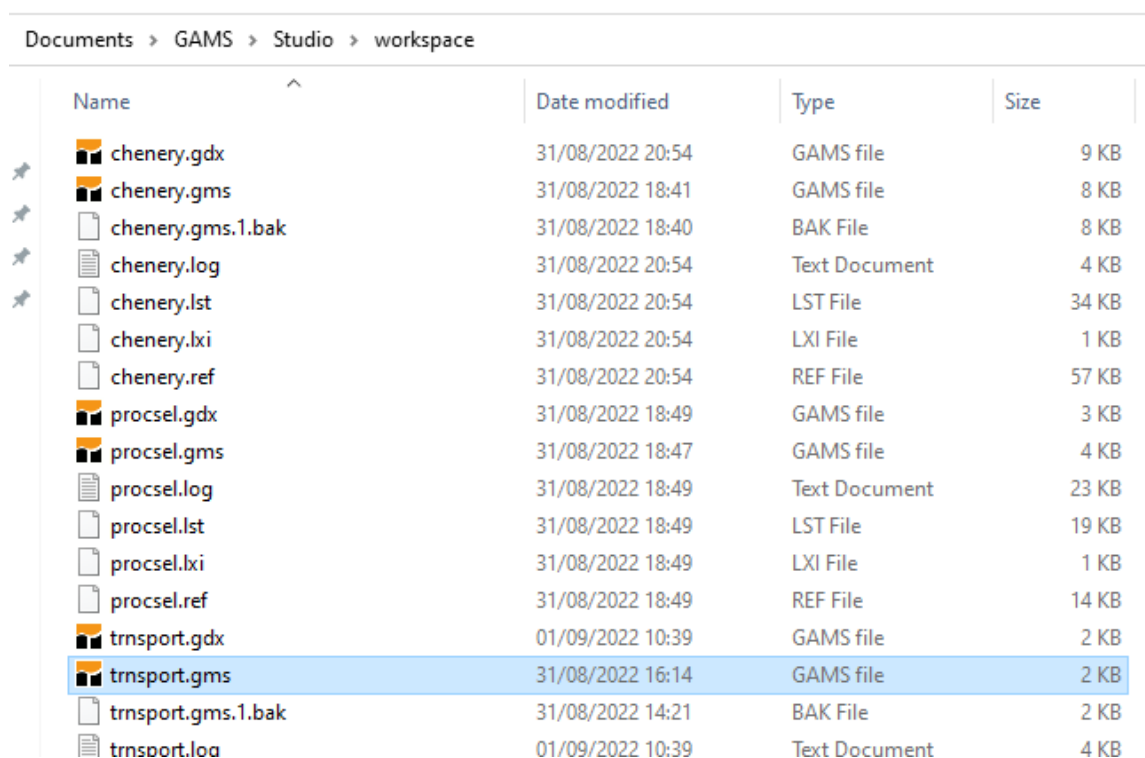
Open Terminal

Open terminal opens a ‘command prompt’ for use when running programmes in command line mode. It is useful but an advanced feature not required for any cgemod course.

Open log tab

This brings the log tab for the selected project to the foreground of the Process Log, NB it does have to be for an active project.

Figure 5.3 Open Location from Studio



Name	Date modified	Type	Size
chenery.gdx	31/08/2022 20:54	GAMS file	9 KB
chenery.gms	31/08/2022 18:41	GAMS file	8 KB
chenery.gms.1.bak	31/08/2022 18:40	BAK File	8 KB
chenery.log	31/08/2022 20:54	Text Document	4 KB
chenery.lst	31/08/2022 20:54	LST File	34 KB
chenery.lxi	31/08/2022 20:54	LXI File	1 KB
chenery.ref	31/08/2022 20:54	REF File	57 KB
procsel.gdx	31/08/2022 18:49	GAMS file	3 KB
procsel.gms	31/08/2022 18:47	GAMS file	4 KB
procsel.log	31/08/2022 18:49	Text Document	23 KB
procsel.lst	31/08/2022 18:49	LST File	19 KB
procsel.lxi	31/08/2022 18:49	LXI File	1 KB
procsel.ref	31/08/2022 18:49	REF File	14 KB
transport.gdx	01/09/2022 10:39	GAMS file	2 KB
transport.gms	31/08/2022 16:14	GAMS file	2 KB
transport.gms.1.bak	31/08/2022 14:21	BAK File	2 KB
transport.loa	01/09/2022 10:39	Text Document	4 KB

Open Location

The option to open location is very useful. If you right click on any label in the Project Explorer and select ‘Open location’ the window (in windows explorer) in Figure 5.3 will open.

This illustrates something that is useful for this introduction to Studio that we consider an unwise practice when using Studio for real. Note how all the files that were opened when installing and testing GAMS and Studio were saved to a single directory (C:\Users\???\Documents\GAMS\Studio\workspace). The workspace directory was created when GAMS was installed and is required as a default.

WARNING: It is easy, when first using Studio to end up with files relating to a single project in different groups/project directories in the Project Explorer. It is important to keep all the files relating to a project together (see above). Hence, we advocate keeping all files relating to a project in the same project directory in Project Explorer.

This is useful for this introduction to GAMS Studio, but we believe that using the workspace directory for your own work is NOT good practice.

Each project should have its own dedicated directory; to ensure all files relating to that project in one location. This may be a sub directory.

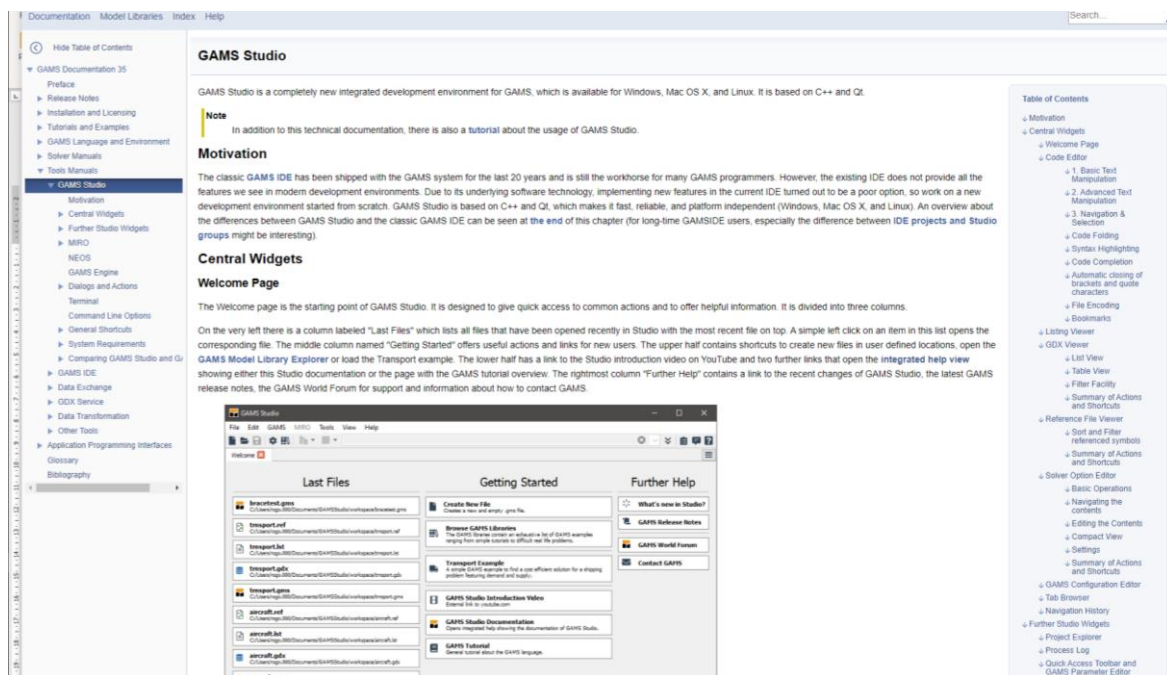
This reflects the fact that each project can have many files, which means that finding files can become difficult and labourious if they are all stored in a single directory. Indeed, in our work the files for each project are typically stored in sub-directories within a master directory to aid with file management. This is the practice adopted in all cgemod courses.

6. Help with Studio

The Help offered by GAMS is comprehensive. You should note that GAMS is a dynamic environment for which updates (releases) are ‘frequent’ (typically 4 to 6 releases each year), so the documentation may not be perfect.

To access the help facilities for Studio use `Help > Studio Documentation`. Alternatively, you can use `Help > GAMS Documentation`, or `F1`, or `View > Help`, and then Studio documentation can be accessed as part of the full GAMS documentation. The help window will stay open until it is closed. To close the help window use `View > Help`.

Figure 6.1 Studio Documentation/Help



The help systems have indices on the left and right-hand sides. The left-hand side covers the full GAMS documentation, while the right-hand side relates to sub sections.

It is recommended that you familiarize yourself early with several parts of the Studio help. First, the section that details the keyboard short cuts for advanced text manipulation and navigation and selection, Figure 6.2 (basic text manipulation uses keyboard commands that are ‘standard’). And second, the section dedicated to ‘settings’, Figure 6.3; you will in due course wish to customise the settings to mirror your preferences.

NOTE: Studio is a computer programme; hence it will take you time to master its features, so over time you should expect to use the help regularly to enhance your capacity and to keep up to date with changes.⁷

Figure 6.2 Keyboard Shortcuts

2. Advanced Text Manipulation			
Action	Shortcut	macOS	Description
Remove Line	Shift - Del		Removes the current line
Duplicate Line	Ctrl - D	Command - D	Duplicates the current line
Lowercase	Alt - Shift - L	Option - Shift - L	Toggle selection to lower case
Uppercase	Alt - Shift - U	Option - Shift - U	Toggle selection to upper case
Tab	Tab	Tab	Add spaces till next tab size
Untab	Shift - Tab	Shift - Tab	Remove spaces till previous tab size
Indent	Ctrl - I	Command - I	Indent complete line
Outdent	Ctrl - Shift - I	Command - Shift - I	Outdent complete line
Find	Ctrl - F	Command - F	Open Search & Replace window, filling search field with selected text if there is any.
Find Next	F3	F3	Jump to next search result if
Find Previous	Shift - F3	Shift - F3	Jump to previous search result

3. Navigation & Selection			
Action	Shortcut	macOS	Description
Zoom In	Ctrl - + or Ctrl - = or Ctrl - Wheel Up	Command - + or Command - = or Command - Wheel Up	Zoom in
Zoom Out	Ctrl - - or Ctrl - Wheel Down	Command - - or Command - Wheel Down	Zoom out
Reset Zoom	Ctrl - 0	Command - 0	Reset zoom
Match Parentheses	Ctrl - B	Command - B	Jump to matching parenthesis

Figure 6.3 Settings

... Help

... a specific location for user defined model libraries. This location can be accessed by opening the Settings dialog (File > Settings) and Library Explorer tries to find user defined model libraries. Adding a new library is done by copying the required files into a subdirectory in that require a restart of the Model Library Explorer to become visible.

Settings

The GAMS Studio settings dialog can be accessed via the File > Settings menu entry or with the hotkey F7. Users can change certain asp

General page

On the "General" tab users control general settings of Studio. Some additional hints:

- "Default GAMS Studio workspace". This path is used as a default location for models imported from the Model Library Explorer.
- "Automatically save modified files before running GAMS" is an option that saves all modified files when the user starts a GAMS run. to go through all files and make sure that changes were saved manually.
- "Open .lst file after running GAMS" automatically puts the generated lst file into the foreground after each GAMS run.
- "Jump to first compilation error" is an automatism that aims to make working with Studio a little easier. When compilation errors occur

⁷ For each new release GAMS provides 'release notes' that detail changes. It is wise to read these notes so that you are aware of changes.

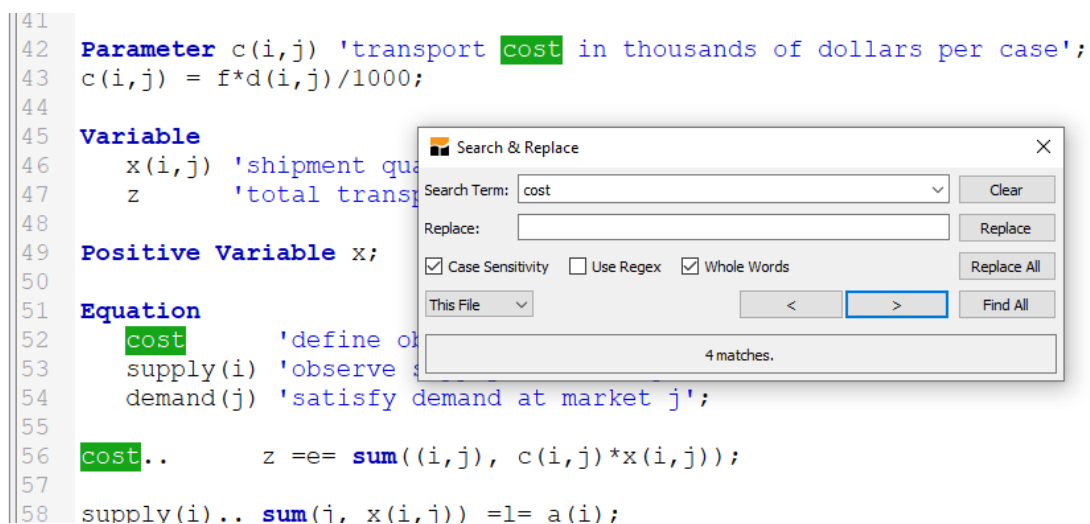
7. Search and Replace

The ‘Search (find) and Replace’ facility in Studio offers numerous useful features. The search and replace facility can be accessed using `Ctrl+F` or `Edit > Search and Replace`, both of which open the dialogue box illustrated in Figure 7.1. The guidance below is based on a run of the `transport.gms` programme with GDX and reference file creation.

Figure 7.1 shows the results of a `Find All` search of the `transport.gms` file for the word `cost` with the options `Case Sensitivity` and `Whole Word` selected and the search limited to `This File`. 4 matches were found, each of which is highlighted in **GREEN** and the references to `cost` can be stepped through using the forward (`>`) and back (`<`) arrows.

The range of the search can be controlled by selecting an option in the dropdown box below the `Case Sensitivity` option. In Figure 7.1 it is limited to `This File`, i.e., the file currently viewed in the editor.

Figure 7.1 Search and Replace



```

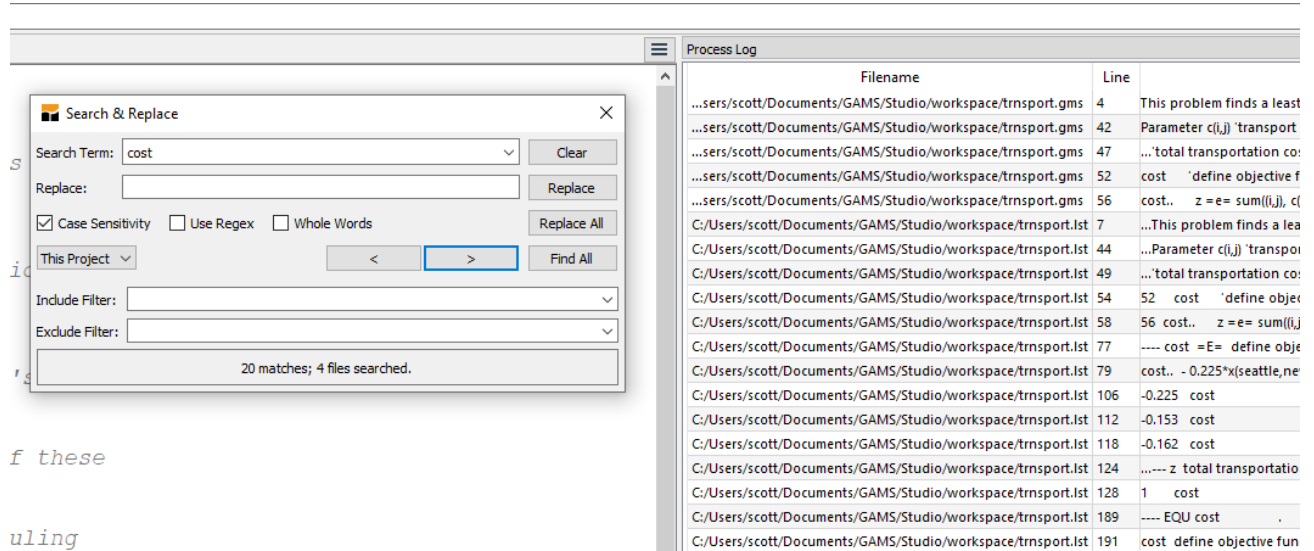
41
42 Parameter c(i,j) 'transport cost in thousands of dollars per case';
43 c(i,j) = f*d(i,j)/1000;
44
45 Variable
46   x(i,j) 'shipment quantity';
47   z      'total transport cost';
48
49 Positive Variable x;
50
51 Equation
52   cost      'define cost';
53   supply(i) 'observe supply at market i';
54   demand(j) 'satisfy demand at market j';
55
56   cost..    z =e= sum((i,j), c(i,j)*x(i,j));
57
58   supply(i).. sum(i, x(i,i)) =l= a(i);

```

The Search option allow for multiple ranges. Figure 8.2 illustrates the results of a `Find All` search of the `transport.gms` file for the word `cost` with the option `Case Sensitivity` selected and the search scope being `This Project`. 30 matches were found in 4 files searched, each of which is highlighted in **GREEN** and the references to `cost` can be stepped through using the forward (`>`) and back (`<`) arrows. It is also possible to search

using the information reported in the Process Log: clicking on any line in the process log will bring the identified file to the front in the editor panel and highlight the reference by converting the highlighting from GREEN to BLUE.

Figure 8.2 Search Results



If something is entered into the Search Term and the Replace boxes the Replace All and Replace buttons become active. According to the scope of the search, replacements can be implemented through one or multiple files.

Selection of Files or Contents (Selection) to be Searched

The ability to select the files that will be searched is useful. The interpretation of the phrases used in the dropdown menu is:

1. "This File" limits the search to the currently active file and is the default. File patterns are ignored for this option.
2. "Selection" limits the search to a selected block of text.
3. "This Project" searches all files that belong to the same group as the currently opened file. A group of files are all children of the same node in the Project Explorer.
4. "Open Tabs" searches all files currently opened in Studio (except GDX files).
5. "All Files" searches all files that appear in the Project Explorer and are searchable.
6. "Folder" searches all files in a designated Folder.

Hints

1. Define the terms to be searched for closely.
 - a. Use parentheses if appropriate, e.g., (j) will produce different results to j with whole word selected.
 - b. GAMS is case insensitive, but you may choose a convention that uses UPPER and lower case to aid readability.
 - c. Regex – is an advanced search term interpretation mode and that stands for Regular Expressions. When activated, instead of a single search term users can specify a pattern that matches an array of different words. Further information about regular expressions is available in the Help system.
2. Note that only files that are open in a Project will be searched, i.e., files that are referenced are not searched if they are not open in a Project.
3. Be careful if using Replace All, especially if doing so over multiple files.

8. Opening and Using GDX files

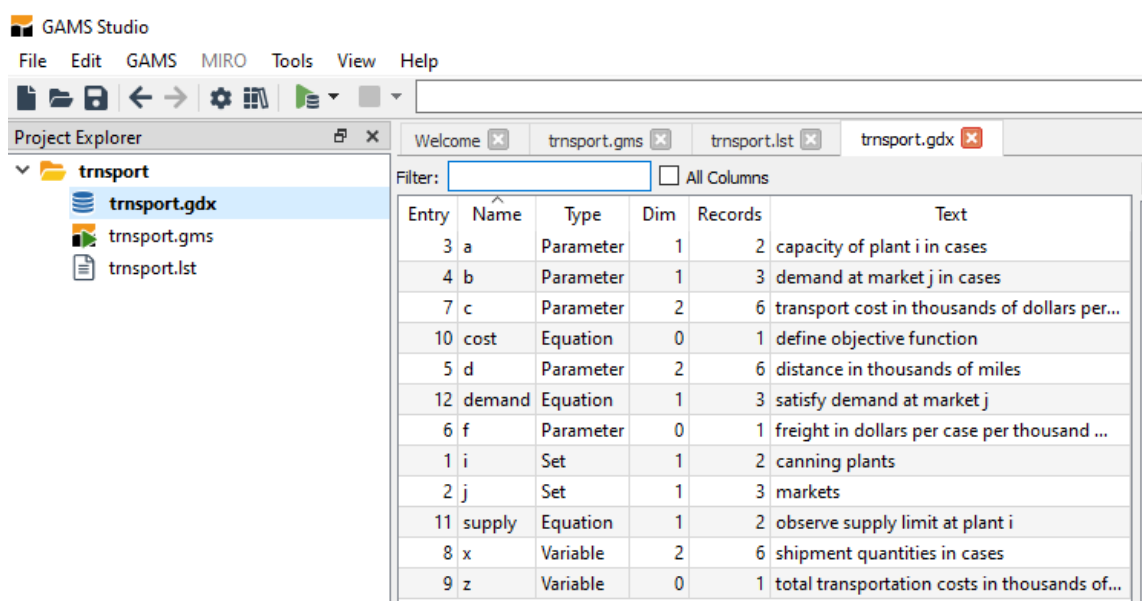
GAMS Data Exchange (GDX) is a series of utilities that have massively impacted on the productivity of the GAMS software; it is difficult to overstate the value of GDX. The GDX utilities are overwhelmingly accessed and used in GAMS programme files and are covered in, and critical for, the cgemod courses; the focus here is on the use of GDX within Studio.

It is important to understand that the standard way of importing, exporting and reporting data relating to a GAMS programme is through GDX, e.g., when data are read in from Excel the data are first transformed into GDX format and then accessed by GAMS. Reading from and writing to GDX files is extremely fast.

GDXViewer

If a programme [filename].gms is run with GDX creation, e.g., using F10, then a GDX file will be created called [filename].gdx, and be opened automatically in the editor window. The output for the `transport.gms` programme – `transport.gdx` – is shown in Figure 8.1.

Figure 8.1 GDX in Studio Editor



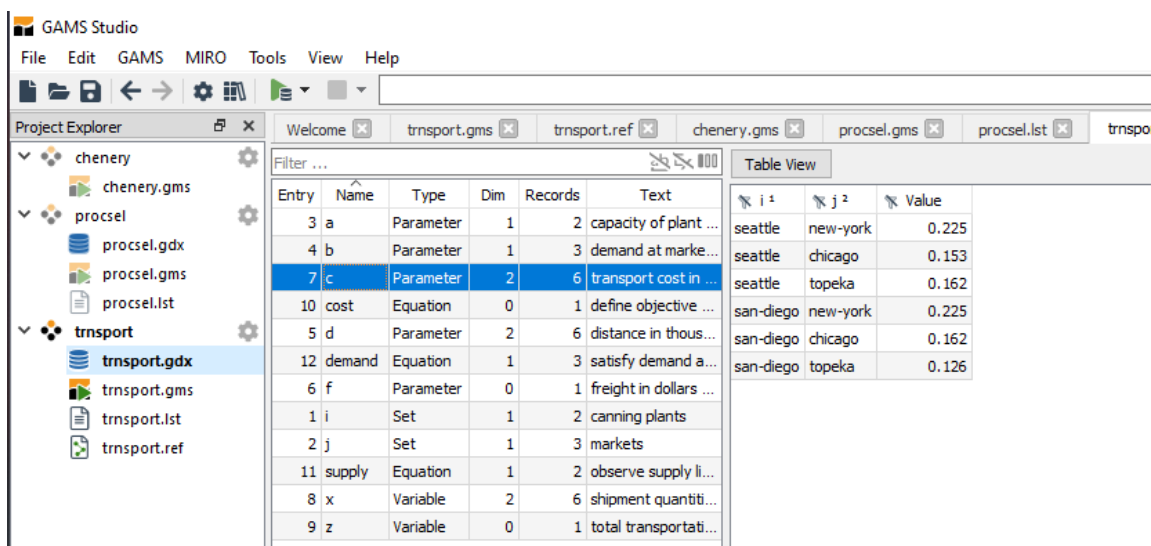
Entry	Name	Type	Dim	Records	Text
3	a	Parameter	1	2	capacity of plant i in cases
4	b	Parameter	1	3	demand at market j in cases
7	c	Parameter	2	6	transport cost in thousands of dollars per...
10	cost	Equation	0	1	define objective function
5	d	Parameter	2	6	distance in thousands of miles
12	demand	Equation	1	3	satisfy demand at market j
6	f	Parameter	0	1	freight in dollars per case per thousand ...
1	i	Set	1	2	canning plants
2	j	Set	1	3	markets
11	supply	Equation	1	2	observe supply limit at plant i
8	x	Variable	2	6	shipment quantities in cases
9	z	Variable	0	1	total transportation costs in thousands of...

The GDX file contains all the data – sets and parameters – used by the model, the results – variables – and information about the equations. At this stage the information will

have little meaning for you (although by the time you complete the exercises linked to this guide you should be able to interpret the information); for now, it is enough to see how the information can be manipulated.

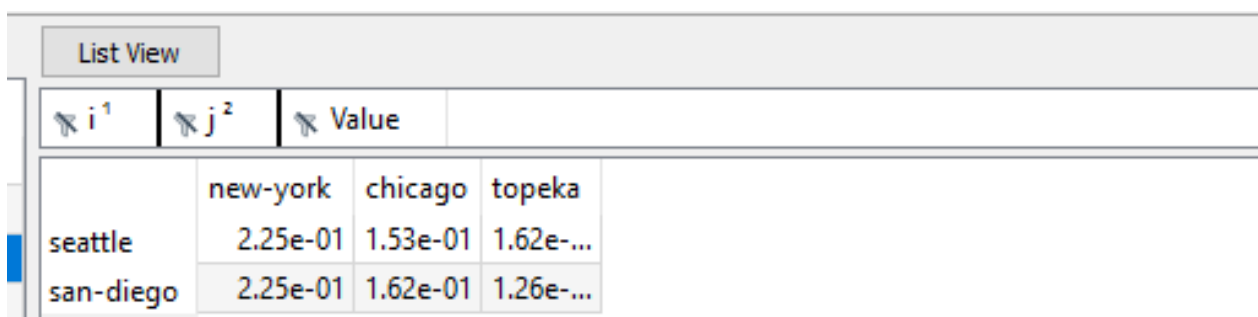
Select the parameter `c`, entry 7, which will open a view (Figure 8.2). We want to modify the view in 2 ways; first we want a `Table View`, so click on the `Table View` button, and second, we want to change the numerical format to `e-format`, so click on the `Preferences` button and select `e-format` in the format box. You should achieve the layout shown in Figure 8.3. (Note that lingering the pointer on the format box triggers a information box).

Figure 8.2 Parameter `c` in GDX List View



You should experiment with the manipulation of the views in the GDX window.

Figure 8.3 Parameter `c` in GDX Table View and Formatted



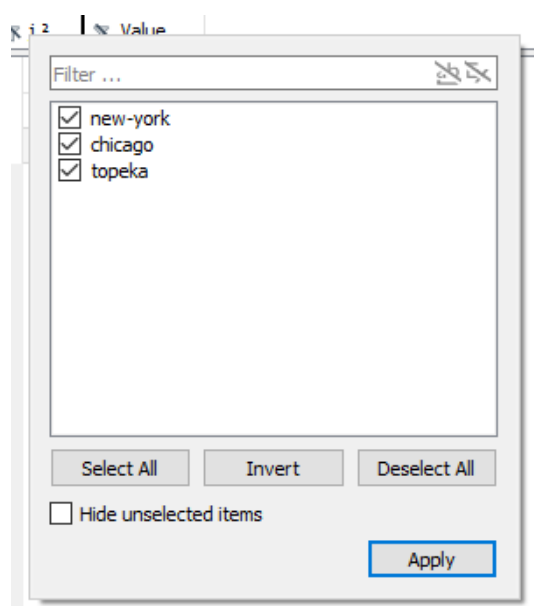
The screenshot shows the 'List View' tab of the GDX window. The table displays the parameter 'c' in e-format, with columns for `i1`, `j2`, and Value. The data is as follows:

<code>i¹</code>	<code>j²</code>	Value	
	new-york	chicago	topeka
seattle	2.25e-01	1.53e-01	1.62e-...
san-diego	2.25e-01	1.62e-01	1.26e-...

Filtering in GDX Viewer

It is often the case that GDX Tables or Lists need to be filtered; most commonly when there are substantial numbers of elements in each index. GDX includes a system for filtering the data that is illustrated in Figure 8.4. To filter on the index/set j right-click on the column index for j (see Figure 8.3) and this will produce the view in Figure 8.4. The filtering can be done in multiple ways: the check boxes for each element can be changed individually (they toggle), all elements can be selected (`Select All`) or deselected (`Deselect All`) or the selection can be inverted (`Invert`), and elements can be found by entering the name in the `Filter ...` Box.

Figure 8.4 Filtering a Parameter the GDX Table or List View



The Values column in Table View or List View also allows for some filtering. If you right-click on the Values column the dialogue box in Figure 8.5 appears. This reports the `min(imum)` and `max(imum)` values and allows the user to control the treatment of special values.

The transport model is small so the various components of the GDX file have limited dimensions. This means that the full extent of the advantages of the filtering system are unlikely to be appreciated with this GDX file. But some time experimenting with the filtering options using this small GDX file will be a good investment.

Figure 8.5 Values Filtering

Min: 0.126 Max: 0.225

Exclude

Show Special Values:

EPS +INF -INF

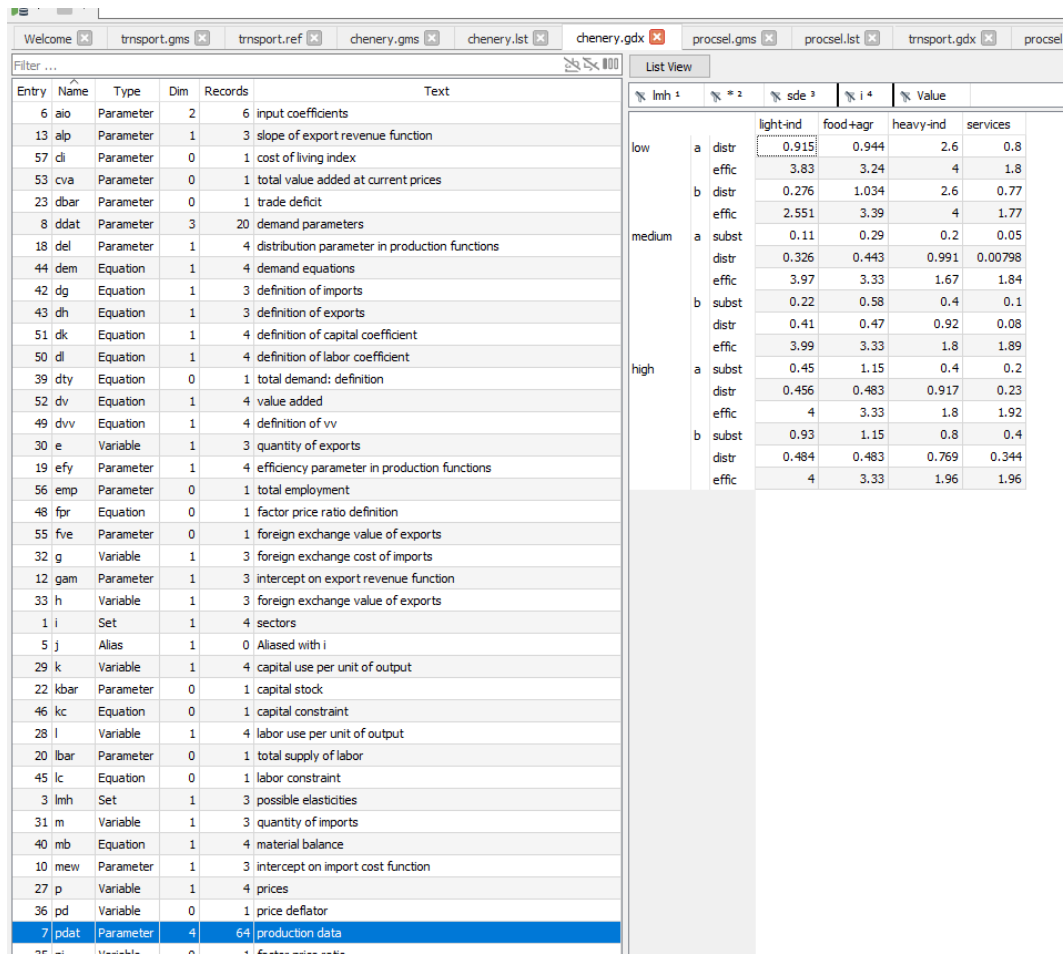
NA UNDF Acronyms

Reset Apply

Filtering is especially important when examining data for larger models with components that have more than 2 dimensions. The Chenery model has parameters and variables with 4 dimensions and up to 64 records; Figure 8.6 illustrates a filtering of the `pdat` parameter from the file `Chenery.gdx` in the default Table View. Note how by default there are 3 dimensions in each row and 1 dimension in each column. The dimensions in the rows and columns can be changed by holding the left mouse button on an element and then dragging the elements to the desired row or column. This takes a bit of practice, but the indication that you are succeeding is that a line appears between the indices.

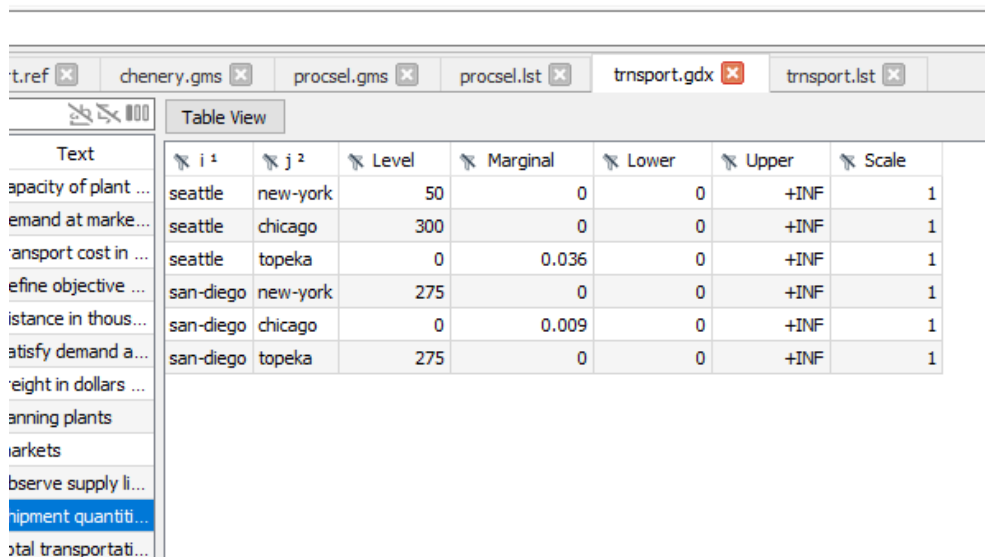
You should experiment with various filtering using the file `chenery.gdx` (you generated this when testing the installation).

Figure 8.6 Filtering a GDX Table or List with More than 2 Dimensions



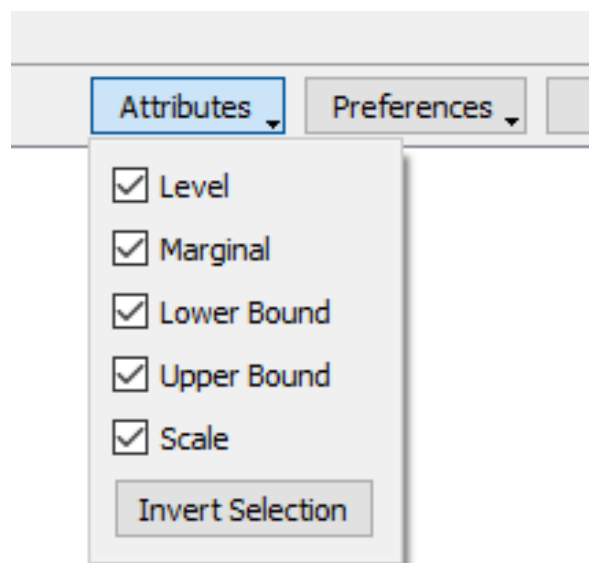
All variables in GAMS have 5 attributes – Level (*.L), Marginal (*.M), Lower (*.LO), Upper (*.UP), Scale – and so when variables are recorded in GDX the values for all the attributes are recorded. Figure 8.7 illustrates the List View for the variable x in the `transport.gdx` file. The Attributes dropdown menu, Figure 8.8, allows the user to filter by attribute, e.g., the Levels values are often of interest so the other attributes can be filtered out. Note that there is an `Invert Selection` button that can reduce the need to use the mouse.

Figure 8.7 Filtering a Variable the GDX Table or List View



Text	i	j	Level	Marginal	Lower	Upper	Scale
capacity of plant ...	seattle	new-york	50	0	0	+INF	1
emand at marke...	seattle	chicago	300	0	0	+INF	1
ransport cost in ...	seattle	topeka	0	0.036	0	+INF	1
efine objective ...	san-diego	new-york	275	0	0	+INF	1
istance in thous...	san-diego	chicago	0	0.009	0	+INF	1
atisfy demand a...	san-diego	topeka	275	0	0	+INF	1
eight in dollars ...							
anning plants							
arkets							
bserve supply li...							
ipment quanti...							
otal transportati...							

Figure 8.8 Filtering a Variable Attributes in the GDX Table or List View

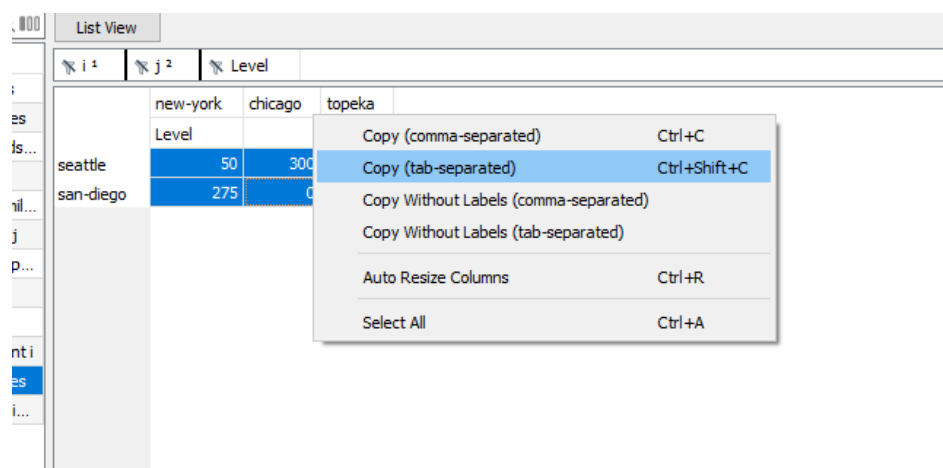


Output Data from GDX Viewer

You can also export information from the GDX viewer. In the `trnsport.gdx` file choose the parameter `c`, select Table View and `g-format`, with Squeeze Trailing Zeros. You can now select all using `Ctrl+A`, or clicking in the top left cell, or selecting the cells (note the labels are not highlighted but when exporting the labels are carried over).

Right clicking on the cells, see Figure 8.9, also gives you options for exporting, which allow you to copy the filtered and formatted table to other programmes, e.g., Excel, Word, etc. As is common the options box provides details for the keyboard shortcuts. NB: if exporting to Excel the better option is Copy (tab-separated) (Ctrl+Shift+C).

Figure 8.9 Copy Options in GDX View



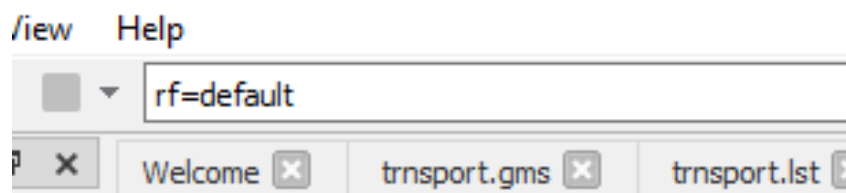
There are other ways in GAMS to output GDX data to other programmes; typically these involve using the GDX utilities in more sophisticated ways; some of those ways are explored in various courses offered by cgemod.

9. Reference files

Reference files are, effectively, indices to GAMS programmes. In a simple programme, such as the `transport.gms` programme used in cgemod's open access training materials, an index may not be considered particularly useful, but in larger and more complex programmes that run over multiple files and thousands of lines of code, indices are invaluable.

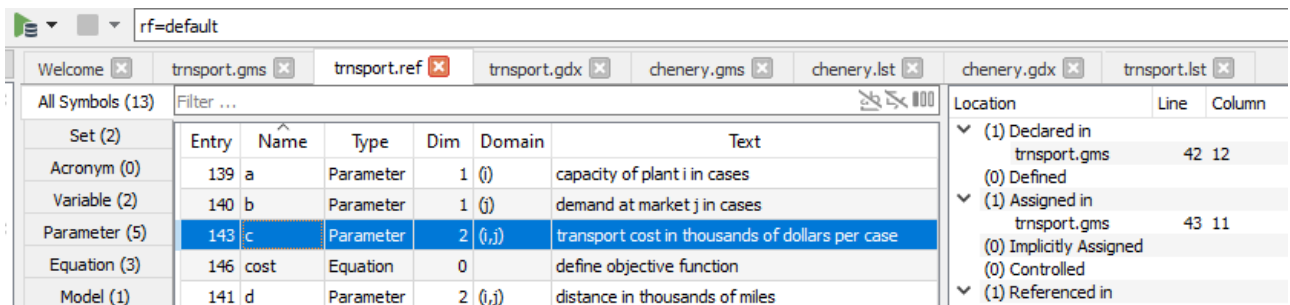
A reference file can be generated by using the command line (see more on command line options in section 10) to pass an instruction to GAMS. Figure 9.1 shows how a reference file with the name of the programme (in this case `transport`) can be created by entering the instruction `rf=default` in the command line. Enter this instruction in the command line and run the programme.

Figure 9.1 Command line Instruction for Reference File



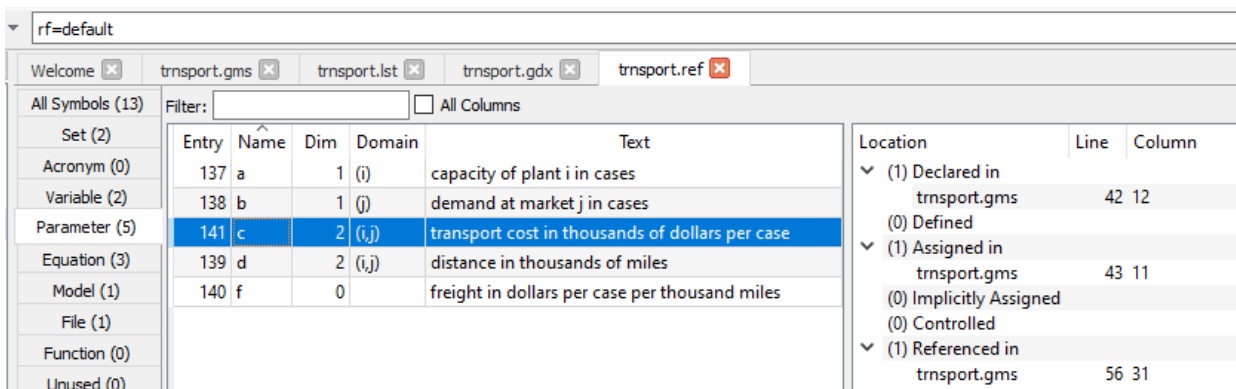
Open the reference file – `transport.ref` – in the Editor window. The results should look similar Figure 9.2. The information in the reference file is reported in three frames within the Editor window. The left-hand frame provides a means for filtering between different types of symbols: it opens with `All Symbols` listed. The middle frame lists the symbols with information about their names, types, dim(ensions), domains and descriptions (text). Finally, the right-hand frame contains information about the location of various attributes about a selected symbol. Thus, selections are made in the left and middle frames to acquire the information in the right-hand frame.

Figure 9.2 Viewing the Reference File



Now select `Parameter` in the left-hand frame, and (parameter) `c` in the middle frame, which will produce the result in Figure 9.3. Now the right-hand frame can be used to locate where attributes of the parameter `c` occur in the programme. To find where parameter `c` is declared click on the line below ‘Declared in’: it will open the file `transport.gms` at the point where `c` is declared. Similarly, you can identify where `c` is assigned and referenced, i.e., each of the attributes with a non-zero number in parentheses.

Figure 9.3 Location Information for Parameter c



Use the reference file to explore how you can find various components of the programme `transport.gms`.

The benefits of the reference file are better seen for a more complex model. The illustration in Figure 9.4 is for a variant of a global model (GLB_3) run without experiments. Note, *inter alia*, how there are now 868 symbols, and 10 Files were used in a programme that ran to over 8,000 lines of code. If the set `c` is selected, it is declared in one file but defined another. Furthermore, it is Controlled in and Reference in several other files. If, for instance,



the file in which c is Defined is clicked on in the location window, Studio will both open that file and move to the line (89) and column (10) where c is defined.

This illustrates how valuable a reference file can be; in fact, we generate a reference file even when we are working with models that we know well. When working with models that are not known, or have not been used for a few weeks, the reference file is a critical tool.

HINT: Using a reference file with the Search and Replace tool is also useful.

Figure 9.4 Reference File for Large Model

Set (172)	Entry	Name	Dim	Domain	Text	Location	Line	Co
Acronym (0)	147	a	1	(sac)	activity accounts			
Variable (142)	182	a_h	1	(sac)	Activities and households			
Parameter (418)	156	a_k_fx	1	(a)	Activities with fixd capital			
Equation (115)	155	aagg	1	(aagg)	Aggregate activity groups			
Model (1)	148	aagr	1	(a)	Agricultural activities			
File (10)	152	acns	1	(a)	Construction activities			
Function (1)	225	acx	2	(a,r)	activities purchased domestically			
Unused (59)	226	acxn	2	(a,r)	activities NOT purchased domestically			
File Used (10)	150	afd	1	(a)	Food activities			
	151	aind	1	(a)	Industrial activities			
	193	aleon	1	(a)	activities with Leontief top level prodn function			
	149	anat	1	(a)	Natural resource activities			
	241	ap	1		Aliased with a			
	191	aqx	2	(a,r)	Activities and regions with CES fn at Level 1			
	192	aqxn	2	(a,r)	Activities and regions with Leontief fn at Level 1			
	154	aser	1	(a)	Service activities			
	153	auti	1	(a)	Utility activities			
	136	c	1	(sac)	commodity accounts	glb_CC_data_load.inc	89	10
	144	cagg	1	(cagg)	Set of commodity groups for reports			

10. Configuring Studio

While Studio looks and behaves like a standard Windows programme, and therefore it is not hard to make adjustments, there are some (option) setting that it is helpful to know about from the beginning since they make it much easier to use GAMS.

Settings

All the setting option can be accessed using `File > Settings` or `F7` or the settings wheel (⚙) on the toolbar. The default settings are a good starting point, but some settings may need changing early on to meet your needs.

1. The default settings in the General tab are a good starting point, but we recommend selecting the option ‘Open file in current project by default’.

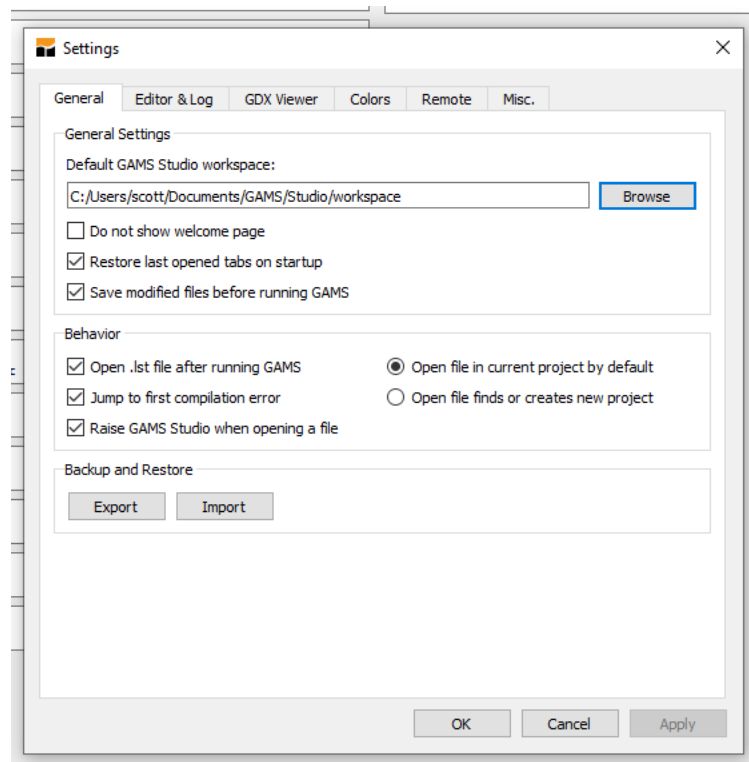
We recommend this, especially for new users, because it is less error prone. For most users it is more common to be opening a file that is part of the current project, i.e., using `Ctrl+O`, as opposed to opening a new project, i.e., `Ctrl+Shift+O`, requires the user to make a positive decision to open a new project. (see Figure 10.1).

2. In the Editor & Log tab
 - a. use a fixed pitch font, e.g., Courier New, (GAMS programmes can depend on the alignment of entries in columns; this can only be easily assured by using a fixed pitch font)
 - b. choose a fontsize that suits you (the choice is inevitably a compromise between having enough information visible on screen and a fontsize that is easy for you to read). In Studio the font size in the editor panel can be increased (`Ctrl+ +`) or decreased (`Ctrl+ -`) at any time.
3. In the Colors tab the default settings are a good starting point (some planned changes in Studio may change this).
4. The MIRO tab relates to a specialist and advanced feature that is in its infancy, so leave until it is needed for your projects.
5. The Misc tab settings are a good starting point.

- a. **One bit of information is required: for the User model library click on the ‘Open Location’ button and make a note of the path identified (this information will be used in the next section).**

Leave the other settings for now, although you can adjust them later as you become more familiar with Studio.⁸

Figure 10.1 **GAMS Settings**



Command Line Options

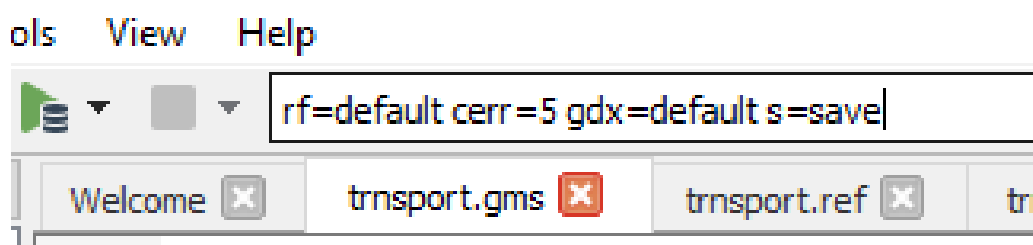
The command line instructions can be very useful and GAMS Studio has enhanced the usability of the command line. Figure 10.2 illustrates a selection of command line options

1. `rf=default`: instructs GAMS to generate a reference file using the same filename as the programme, i.e., `transport.ref`.
2. `cerr=5`: instructs GAMS to stop after encountering 5 cumulative errors (the user can define the number they prefer).

⁸ We will use these settings during the online course; a common appearance can make it easier for an instructor to provide support across multiple users.

3. `gdx=output`: instructs GAMS to generate a gdx file with the name `output.gdx` – this file would contain the same information as produced for the file `transport.gdx` after running with F10.
4. `s=save`: instructs GAMS to generate a work file or scratch file with the name `save` that can be used to sub divide components of a programme – these files are usually deleted at the end of a run (we use this a lot in our day-to-day work and in the `cgemod` courses because it makes the processes more efficient).

Figure 10.2 Command Line Instructions



Studio provides tools that make command line instructions more efficient.

First, using F10, i.e., run with gdx creation, avoids the need for the command `gdx=output`. And second, other commands can be set up as defaults every time any programme is run – below we will illustrate how to do this for `rf` and `cerr`.

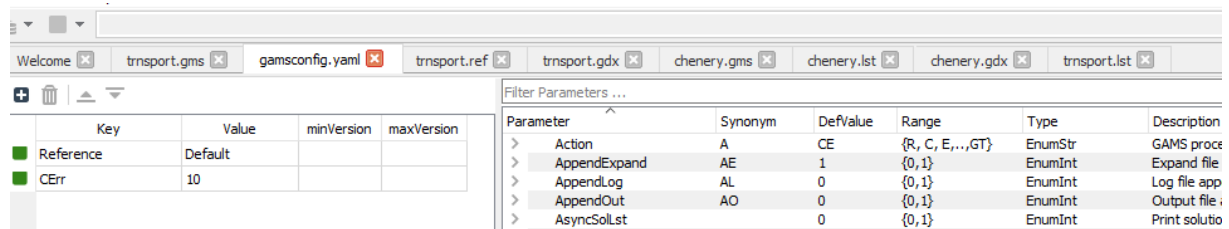
The command line instructions in Studio are project specific unless they have been set up as defaults. We find with the `SAVE` and `RESTART` option that this works best across two projects calling files from a single directory. A common way to work is to run a model that replicates the current situation and then run a separate programme, experiment, that shocks the model to evaluate how changes in model parameters impact upon the results for the model variables. The work file for the model is saved using `s=save`⁹ in the command line and the programme with the experiment uses `r(estart)=save` in the command line. Because these command lines are in separate projects they are preserved when files are saved/programmes are run. The use of the save and restart facility is explained in detail in `cgemod` courses.

⁹ If a user wants to send the work file to a designated directory the instruction can be written to include a path, e.g., `s=sandr\save`, and the restart instruction would need to include the part.

Configuring Studio to Run Commands by Default.

Studio provides a method for implementing many instructions by default. Open GAMS > Default GAMS Configuration; this will open the configuration editor in the Editor window (see Figure 10.3). The configuration editor has two frames: in the right-hand frame are the available commands, in the left-hand frame are the commands that are implemented by default.

Figure 10.3 Default GAMS Configuration



To instruct GAMS to generate a reference file by default find the line labelled ‘reference’ (HINT: use the search) and drag the reference parameter into the left-hand frame, See Figure 10.3. The selected parameter – reference – is identified by a **GREEN** box. If the Value box in the left-hand frame is left empty the reference file created every time a GAMS programme is run will have the name `1.ref`. If the value box contains the word default then the reference file will be named after the programme name, i.e., for `[filename].gms`, as `[filename].ref`. This is a preferred option.

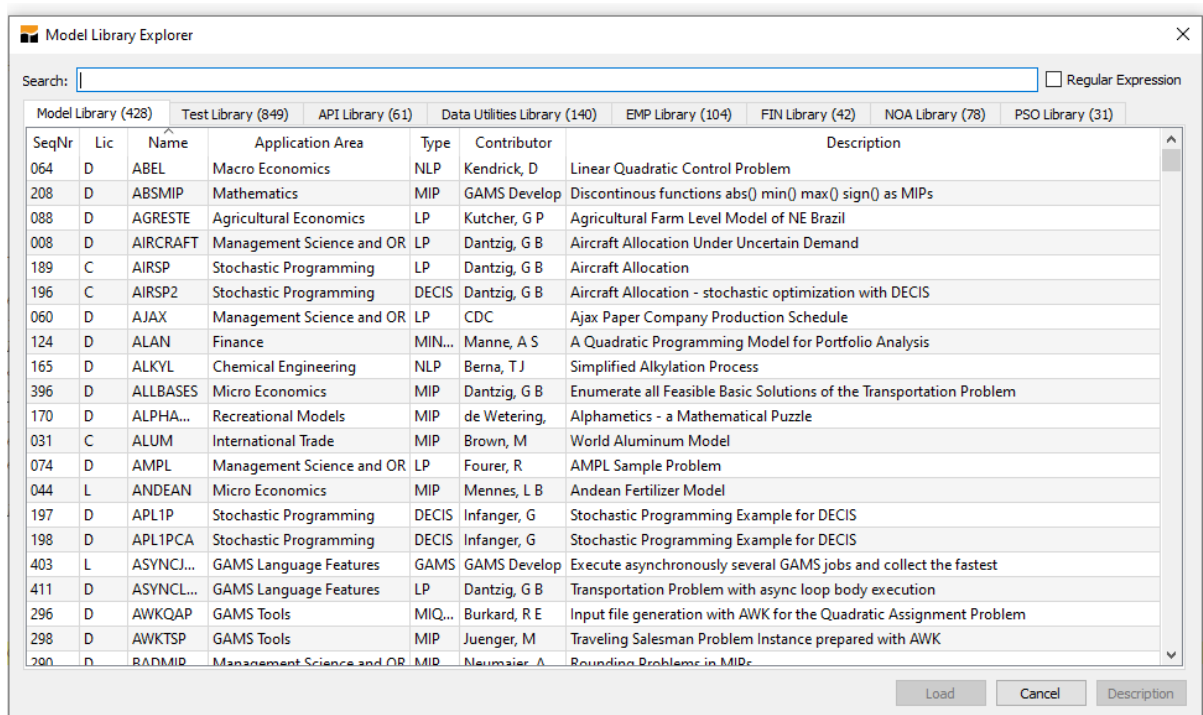
NB: a file `gamsconfig.yaml` will be created, and need to be saved (note the asterisk in the tab after its name). Use the location option in the Project Explorer to find where GAMS saves this file.

To instruct GAMS to stop after a specified number of errors file by default find the line labelled ‘CEer’ and drag the CEer parameter into the left-hand frame, See Figure 10.3. The selected parameter – CEer – is identified by a **GREEN** box. Select the number of cumulative errors after which the programme with stop and enter this number in the Value box in the left-hand frame. Our experience suggests somewhere between 5 and 10 is a practical value.

11. Model Libraries

GAMS comes with several large model libraries. To access the libraries, select GAMS > Model Library Explorer or press F6. This opens the Model Library Explorer, see Figure 11.1. Note how there are multiple tabs each of which contains a different library. These libraries are very valuable source for code to help you develop your use of GAMS.

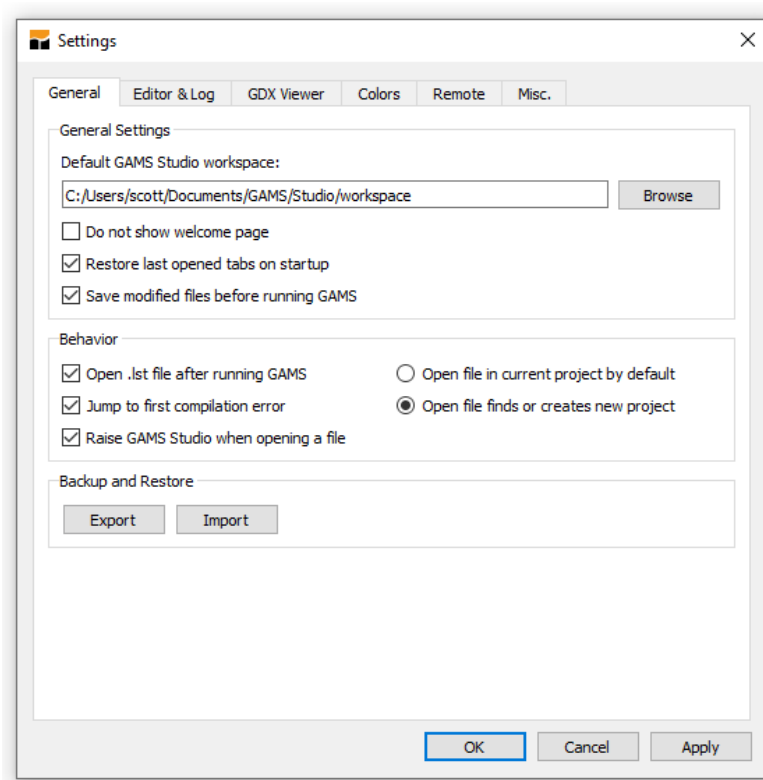
Figure 11.1 Model Library Explorer



SeqNr	Lic	Name	Application Area	Type	Contributor	Description
064	D	ABEL	Macro Economics	NLP	Kendrick, D	Linear Quadratic Control Problem
208	D	ABSMIP	Mathematics	MIP	GAMS Develop	Discontinuous functions abs() min() max() sign() as MIPs
088	D	AGRESTE	Agricultural Economics	LP	Kutcher, G P	Agricultural Farm Level Model of NE Brazil
008	D	AIRCRAFT	Management Science and OR	LP	Dantzig, G B	Aircraft Allocation Under Uncertain Demand
189	C	AIRSP	Stochastic Programming	LP	Dantzig, G B	Aircraft Allocation
196	C	AIRSP2	Stochastic Programming	DECIS	Dantzig, G B	Aircraft Allocation - stochastic optimization with DECIS
060	D	AJAX	Management Science and OR	LP	CDC	Ajax Paper Company Production Schedule
124	D	ALAN	Finance	MIN...	Manne, A S	A Quadratic Programming Model for Portfolio Analysis
165	D	ALKYL	Chemical Engineering	NLP	Berna, T J	Simplified Alkylation Process
396	D	ALLBASES	Micro Economics	MIP	Dantzig, G B	Enumerate all Feasible Basic Solutions of the Transportation Problem
170	D	ALPHA...	Recreational Models	MIP	de Wetering,	Alphametics - a Mathematical Puzzle
031	C	ALUM	International Trade	MIP	Brown, M	World Aluminum Model
074	D	AMPL	Management Science and OR	LP	Fourer, R	AMPL Sample Problem
044	L	ANDEAN	Micro Economics	MIP	Mennes, L B	Andean Fertilizer Model
197	D	APLIP	Stochastic Programming	DECIS	Infanger, G	Stochastic Programming Example for DECIS
198	D	APLIPCA	Stochastic Programming	DECIS	Infanger, G	Stochastic Programming Example for DECIS
403	L	ASYNCL...	GAMS Language Features	GAMS	GAMS Develop	Execute asynchronously several GAMS jobs and collect the fastest
411	D	ASYNCL...	GAMS Language Features	LP	Dantzig, G B	Transportation Problem with async loop body execution
296	D	AWKQAP	GAMS Tools	MIQ...	Burkard, R E	Input file generation with AWK for the Quadratic Assignment Problem
298	D	AWKTSP	GAMS Tools	MIP	Juenger, M	Traveling Salesman Problem Instance prepared with AWK
200	D	RADMIP	Management Science and OR	MIP	Neumaier, A	Bounding Problems in MIPs

If you click on the column title the library will be sorted in the alphanumeric order of the entries in that column. Click on the column 'SeqNr', the first entry in that column will now be '001' with the name TRNSPORT.

You have accessed files from the GAMS libraries, but by default the files were downloaded to the workspace directory (C:/Users/****/Documents/GAMS/Studio/workspace). This has been convenient for this introduction to Studio, but far less convenient for day-to-day use. The default destination directory is controlled in the Settings dialogue box (see Figure 11.2). This can be changed to any other directory using the Browse button on the General Settings tab. However, this method is limited.

Figure 11.2 GAMS Studio General Settings

Using a GAMS User Model Library

The 'User Model Library' facility in GAMS provides an indexing facility whereby collections of models can be archived and then easily accessed from Studio. All the files contained within a User Model Library are stored in a directory and accessed via the Model Library Explorer. An important task when a user wants to add a 'User Model Library' is to ensure it is saved in a location that the Model Library Explorer will explore. We will illustrate the process using the transport model files setup for this introduction to Studio and the directory structure we advocate for cgemod courses.

We advocate creating a master directory for all the training materials associated with cgemod's courses. So

1. In Windows Explorer create a directory, say, `C:\cgemod_training`. We suggest doing this on the top level of your working drive (this would be the C drive for us.)
2. Add the sub directory `cgemod_lib` to the master directory, i.e.,
`C:\cgemod_training\cgemod_lib`. This sub directory will be used to store



all the User Model Library files for cgemod courses; with each component for each course more files will be added to this directory and the index will be updated. (It is not strictly necessary to store the files in this way, but mistakes happen, and this method means that correcting certain common mistakes is quicker.)

3. Add the sub directory downloads to the master directory, i.e.,
C:\cgemod_training\downloads. This sub directory to store zipped archives of model files that are downloaded for cgemod courses.
4. Download the `trans.zip` file from the cgemod site (http://cgemod.org.uk/int_gams.html) and save it in the download sub directory.
5. Unzip the contents of the file into the sub directory called `cgemod_lib`.
6. Copy all of the files in the `cgemod_lib` sub directory –just the contents of the directory NOT the directory itself.
7. Open Studio settings - File > Settings or F7 or the Settings Wheel on the toolbar and go to the Misc tab.
8. Change the path for the User Model library, see Figure 11.3, from
C:/Users/.../Documents/GAMS/modellibs (the exact path will depend on the settings on your PC) to C:/cgemod_training/cgemod_lib.
9. Close down the Studio Settings window remembering to click OK at the bottom of the Misc tab.

In Studio open the Model Library Explorer, F6., and expand the window to the right and then click on the tab Introduction to GAMS Studio; you may need to scroll across the tabs to find the tab. The results should look like Figure 11.4. Note the ‘person’ figure associated with a User Model Library.

We, cgemod, use the 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 only in that our libraries supply multiple files with each download while GAMS libraries, typically, 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.

Figure 11.3 GAMS Studio Misc Settings

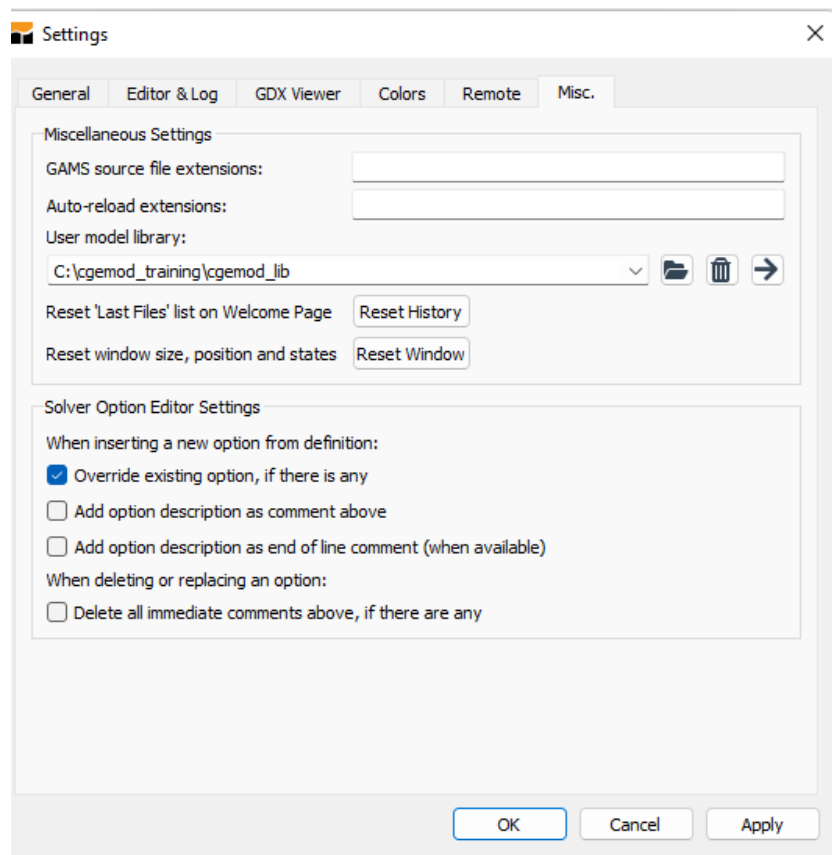
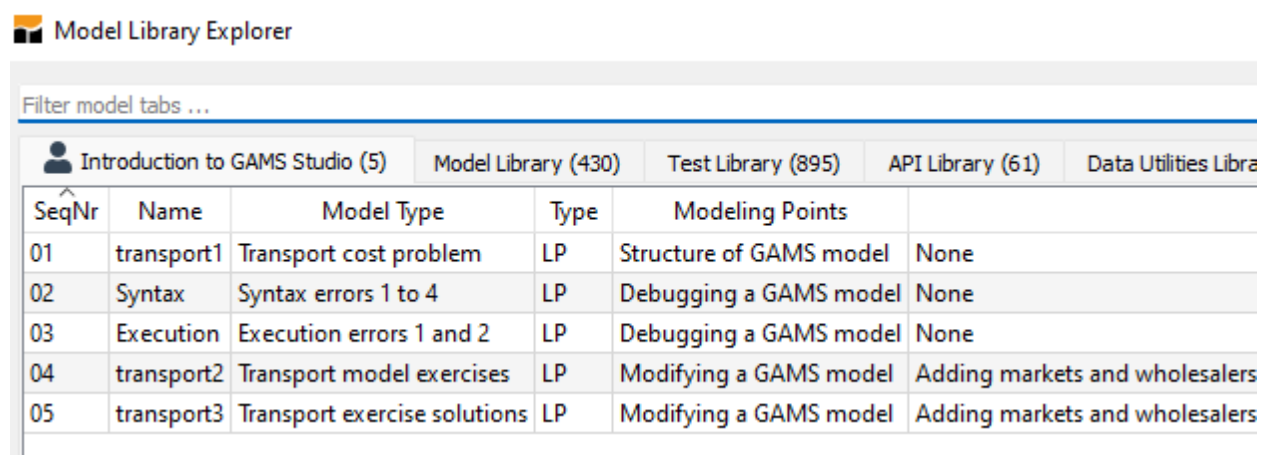


Figure 11.4 Model Library Explorer cw User Model Library

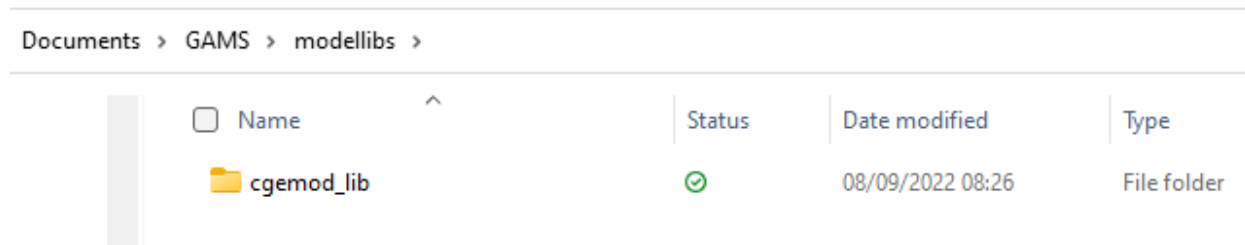


An alternative arrangement involves adding the User model library to the default path for model libraries. After step 5 above, i.e., after unzipping the archive from the cgemod web site (http://cgemod.org.uk/int_gams.html)

1. Copy all the files in the `cgemod_lib` sub directory –just the contents of the directory NOT the directory itself.
2. Open Studio settings - `File > Settings` or `F7` or the `Settings Wheel` on the toolbar and go to the `Misc` tab.
3. Click on the ‘`Open Location`’ button (`→`), which will open a `Windows Explorer` window. You need to make sure the directory `...\GAMS\modellibs` is open.
4. Create a sub directory call `...\GAMS\modellibs\cgemod_lib`
5. Then Paste the contents of the `C:\cgemod_training\cgemod_lib` directory into the `...\GAMS\modellibs\cgemod_lib` directory.
6. Close down the `Studio Settings` window remembering to click `OK` at the bottom of the `Misc` tab.

The result should be like that in `Figure 11.5`. Note how the `modellibs` directory is in `GAMS` directory in the user’s documents directory.

Figure 11.5 **GAMS\modellibs Directory**



In `Studio` it is necessary that we establish a new directory for each new project so that a destination directory is specified for the download. To do this carry out the following

1. In `Studio` select `File>New Project`. This opens a window – `Import Project` – that is essentially a `Windows Explorer Window`, see `Figure 11.6`.
2. We wish to add a new (sub) directory in the directory `C:\cgemod_training`; so double click on `C:\cgemod_training` and then right-click in the left-hand panel and choose `New>Folder` from the menu. (see `Figure 11.6`)

3. Give the New Folder a name. In this example we call the directory `transport`, and SELECT this directory (Figure 11.7 illustrates the sub-directory that should be created).
4. The Studio panels should appear automatically once the new directory has been created. (Figure 11.8 illustrates how the Project Explorer and Editor panels should appear).
5. In Studio press F6 and in the Model Library Explorer select the `Introduction to GAMS Studio` and then select the library file `transport1` and choose Load (or double click of the name), which is SeqNr: 1. (Figure 11.9.)
6. The `trans.gms` model will now be displayed in the editor window and be listed in the Project Explorer as being in the project `transport`. (see Figure 11.10).
7. 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_training\Transport`. (See Figure 11.11).
8. Studio can now be used to work with the `transport` model as done previously in this introduction and in the associated exercises.

This is how each new project can be created from files in a User Model Library. It also demonstrates how multiple files can be provided from a single library entry. This method is used through all the courses offered by cgemod.

It is also a method for establishing a new project in Studio when starting from scratch.

Figure 11.6 Create a New Directory from New Project Menu

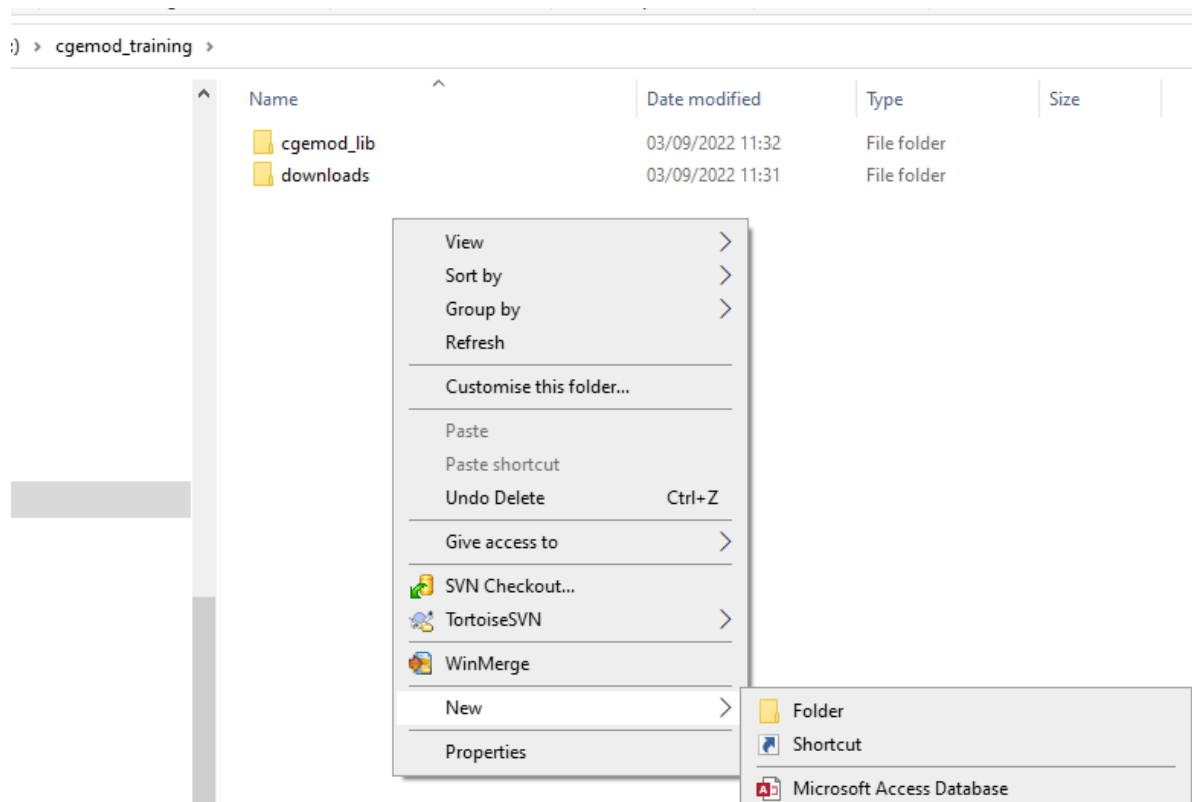


Figure 11.7 cgemod_training/transport Directory

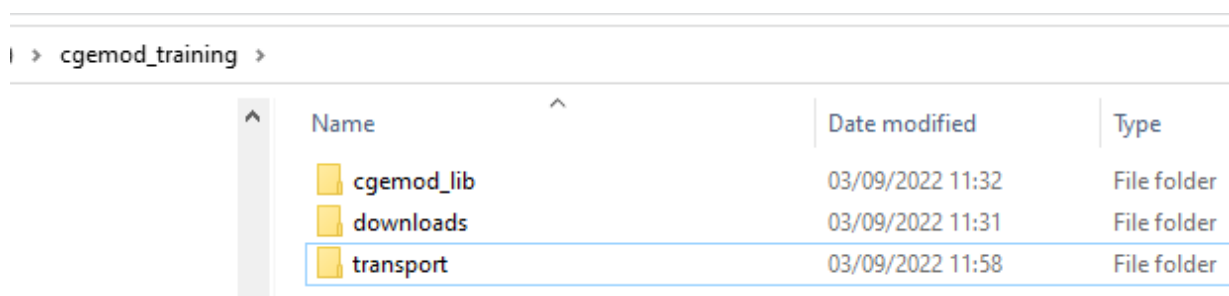


Figure 11.8 New Project View in Studio

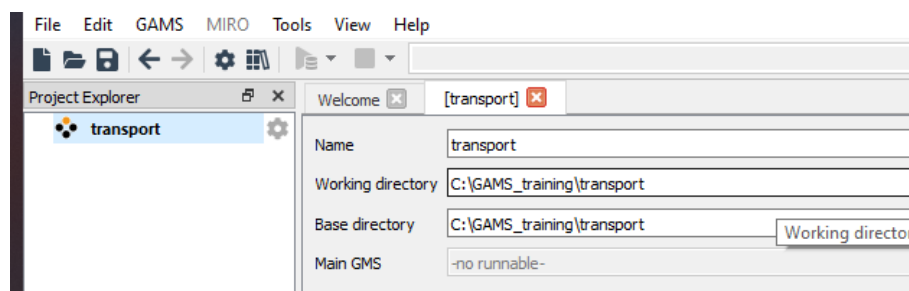


Figure 11.9 User Model Library in Studio

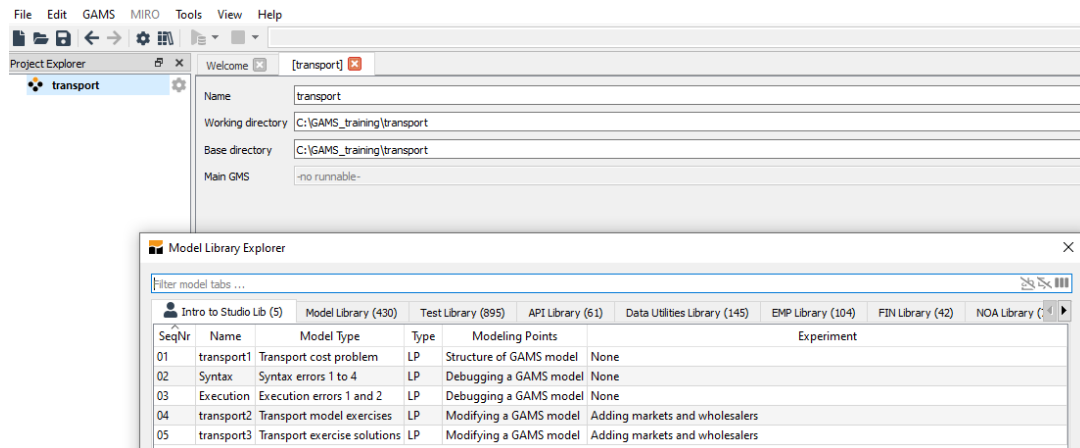


Figure 11.10 trans.gms in project Transport

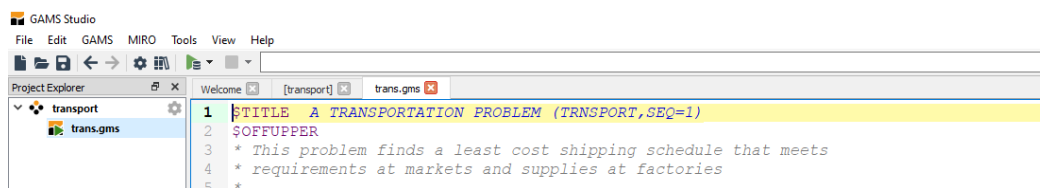
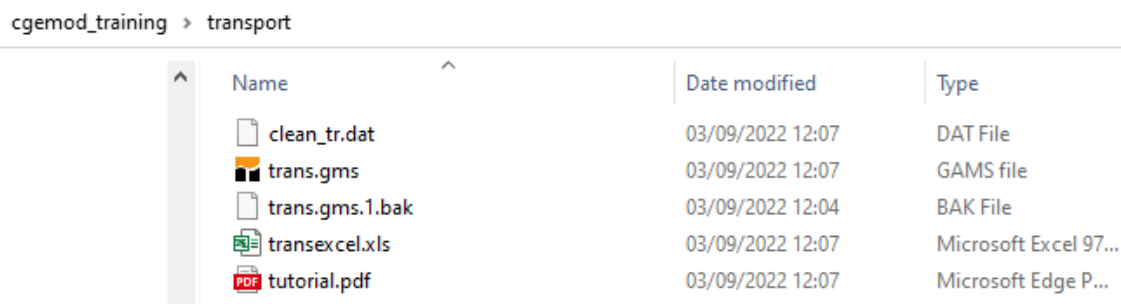


Figure 11.11 Directory on C:\cgemod_training\Transport



12. Comparing Files

It is sometimes helpful to compare the content of 2, or more, files. GAMSIDE included a utility, `Diff TextFiles`, that allowed for comparing 2 files, but no equivalent utility is included with Studio. (The developers of GAMS Studio have suggested that a similar utility may at some point be added for Studio.)

Access to comparison software is valuable. They allow easy comparisons between different model versions, which can be helpful when searching for ‘changes’, ‘errors’ and ‘bugs’. They can also be helpful when developing new versions of existing models.

There are several options for Windows operating systems are available, including

1. MS Word: using `Review>Compare` produces a comparison of two selected documents
2. WinMerge: using `File>Open` then selecting 2 or 3 documents to compare
3. NotePad++: open two documents and then using `Plugins>Compare`

MS Word and NotePad++ are the easiest to use, while WinMerge has the most features with MS Word the least. MS Word is free if you have MS Office/Office 365, while NotePad++ (<https://notepad-plus-plus.org/>) and WinMerge (<https://winmerge.org/>) are open source. As always with open-source programmes be careful when installing them, especially if they come from third-party sites.

The courses offered by cgemod assume the participant has access to comparison software. Our examples assume WinMerge.

13. Studio with Modular Projects

The examples used so far in this introduction are based on simple models where all the model codes are contained in a single `gms` file, and all the files are contained in a single directory. However, many modern models are modular with different components of the model in different files that are selected when the model is compiled; these components may be in multiple sub directories that may or not be included in a ‘master’ directory. One of the major developments in Studio, compared to GAMSIDE, is the enhanced facilities for organising, searching and using models that are modular. This section provides a brief introduction to using Studio with a modular model that may, or may not, be more complex. The example used for this section is the `STAGE_t` model as it is used in the (single country) Recursive Dynamic CGE course.

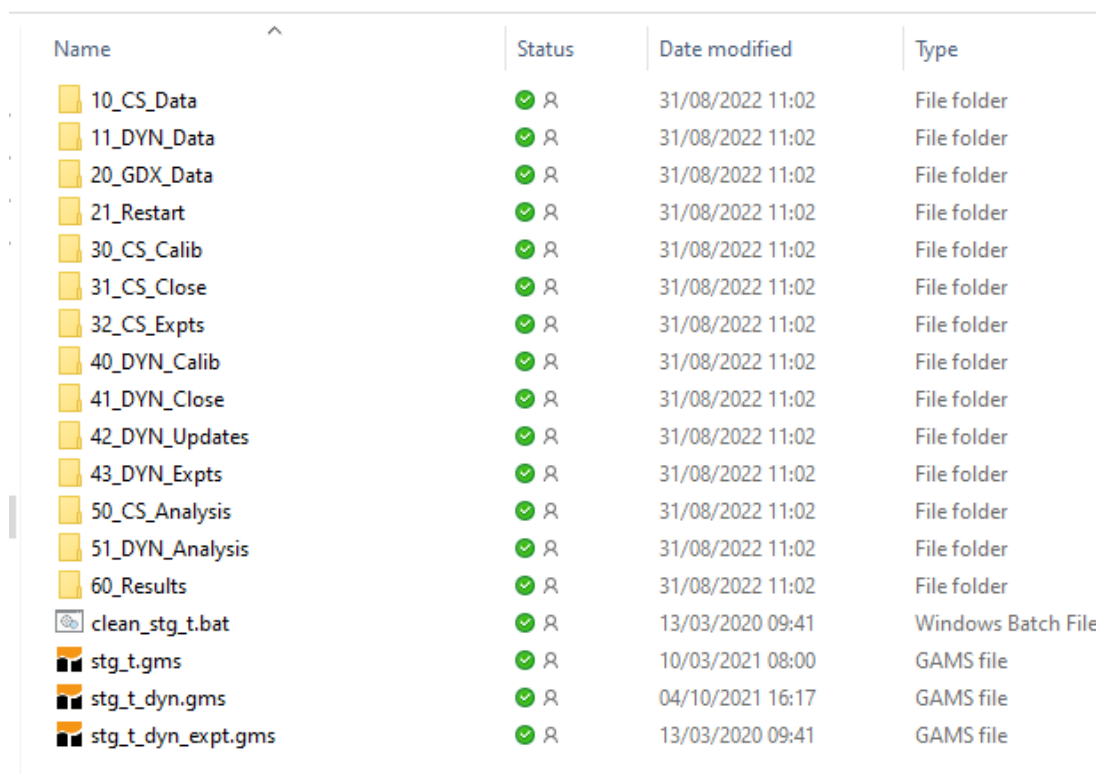
File and Directory Structure

The directory structure and organisation of `STAGE_t` reflects our view about an appropriate method; this has in part been influenced by the functionality provided by Studio (see Figure 13.1). We suggest that all sub directories and files relating to a specific version of a model should be in a single master directory. Each master directory can support multiple ‘projects’ – model variants. Each model variant has only a single model file, e.g., `stg_t.gms`, implemented as unique project, and each model variant has one or more experiment file, e.g., `stg_t_expt.gms`, implemented as unique projects. This requires the implicit presumption that the output from the model file is saved as a work file and each experiment file restarts from the work file. The only `gms` files are saved to the top-level of the master directory. Each `gms` file generates a reference file and generic GDX file; these are saved to the top-level of the master directory.

The model and experiment `gms` files each call in multiple `INCLUDE` files at compile time; all these files are labelled `[filename].inc` and saved in sub directories. It is not a requirement of GAMS to label nested files as `*.inc` files, e.g., all the nested `INCLUDE` files could be labelled `*.gms` or even `*.txt`, but is our preference to always label `INCLUDE` files as `*.inc` for added clarity. Data are supplied to the models and experiments as Excel or GDX files; all Excel files are converted to GDX files by the model and experiments.

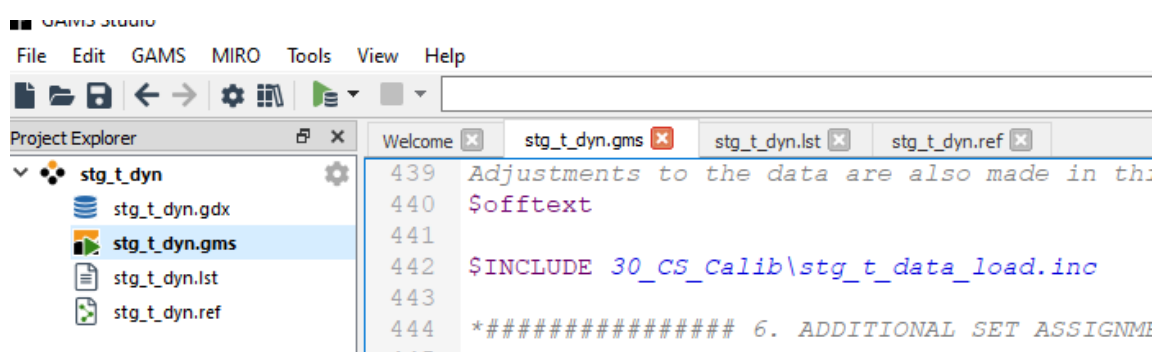
Assume that the intention is to work with the `stg_t_dyn.gms` model. Figure 13.2 illustrates how Studio might appear after the `stg_t_dyn.gms` model has been run (F10) and the reference file has been opened and the `gms` file scrolled to line 442. A problem at this point could be defined as a lack of information about the contents of the various, and apparently unidentified, `INCLUDE` files.

Figure 13.1 STAGE_t in Windows Explorer



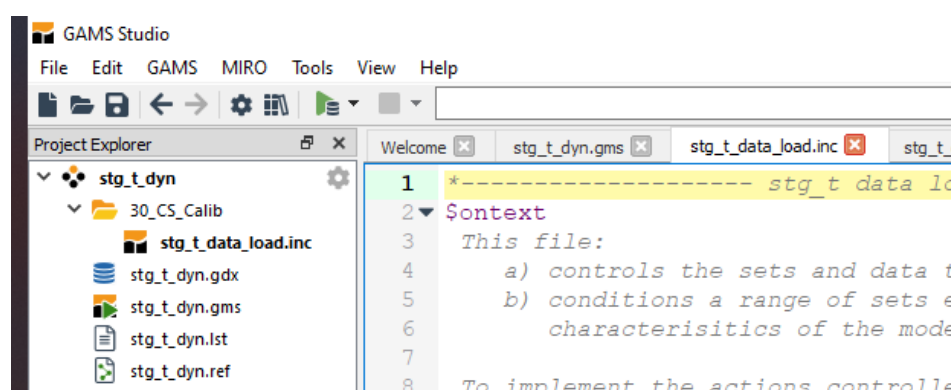
Name	Status	Date modified	Type
10_CS_Data	✓ R	31/08/2022 11:02	File folder
11_DYN_Data	✓ R	31/08/2022 11:02	File folder
20_GDX_Data	✓ R	31/08/2022 11:02	File folder
21_Restart	✓ R	31/08/2022 11:02	File folder
30_CS_Calib	✓ R	31/08/2022 11:02	File folder
31_CS_Close	✓ R	31/08/2022 11:02	File folder
32_CS_Expts	✓ R	31/08/2022 11:02	File folder
40_DYN_Calib	✓ R	31/08/2022 11:02	File folder
41_DYN_Close	✓ R	31/08/2022 11:02	File folder
42_DYN_Updates	✓ R	31/08/2022 11:02	File folder
43_DYN_Expts	✓ R	31/08/2022 11:02	File folder
50_CS_Analysis	✓ R	31/08/2022 11:02	File folder
51_DYN_Analysis	✓ R	31/08/2022 11:02	File folder
60_Results	✓ R	31/08/2022 11:02	File folder
clean_stg_t.bat	✓ R	13/03/2020 09:41	Windows Batch File
stg_t.gms	✓ R	10/03/2021 08:00	GAMS file
stg_t_dyn.gms	✓ R	04/10/2021 16:17	GAMS file
stg_t_dyn_expt.gms	✓ R	13/03/2020 09:41	GAMS file

Figure 13.2 STAGE_t in Studio



To overcome this problem, it can be useful to be able to have all, or a selection, of the INCLUDE files open in Studio. Studio provides multiple ways to achieve this; each of which has advantages in different situations. The most obvious option is `Ctrl+O` (if the setting ‘Open file in current project by default’ is chosen) or `Ctrl+Shift+O` (if the setting ‘Open the file finds or creates a new project’ is chosen). Figure 13.3a illustrates the outcome if the file `stg_t_data_load.inc` from the sub directory `30_CS_Calib` has been opened. Note how the Project Explorer shows that file `stg_t_data_load.inc` is from the sub directory `30_CS_Calib`. The same outcome could be achieved by right-clicking on the project name and selecting `Add Existing File` from the popup menu.

Figure 13.3a STAGE_t in Studio – Opening & Finding Files



Another way to open a file is to `Ctrl+click` on an `$INCLUDE` command in any file. This is illustrated in Figure 13.2; if the user chooses to `Ctrl+click` on file details, i.e., the text in BLUE, in line 442, the file directory `30_CS_Calib\stg_t_data_load.inc` is opened producing the same outcome illustrated in Figure 13.3a.

When first starting to explore a modular model, it can be helpful to be able to get an overview of the model structure and the various nested files. One way to achieve that is compile the model (`Shift+F10`)¹⁰ with the option to generate a reference file. If the reference file is opened and the symbols are filtered for `File Used`, this informs the user that 9 files, including the master `*.gms` file were used, see Figure 13.3b. The locations of the

¹⁰ Choosing to compile rather than run the model may be necessary if there is syntax or execution problems with the model.

files used are listed in the right-hand pane in the order in which they were accessed. Moreover, if the user clicks on any of the file locations the respective file is opened in Studio, e.g., clicking on the second file in the file locations list opens the file 30_CS_Calib\stg_t_data_load.inc as illustrated in Figure 13.3a.

One simple option would be to open all the files used in turn. This is in addition to using the reference file to select individual symbols and open the files in which they were used, see Figure 9.4 above.

Figure 13.3b STAGE_t in Studio – Opening & Finding Files

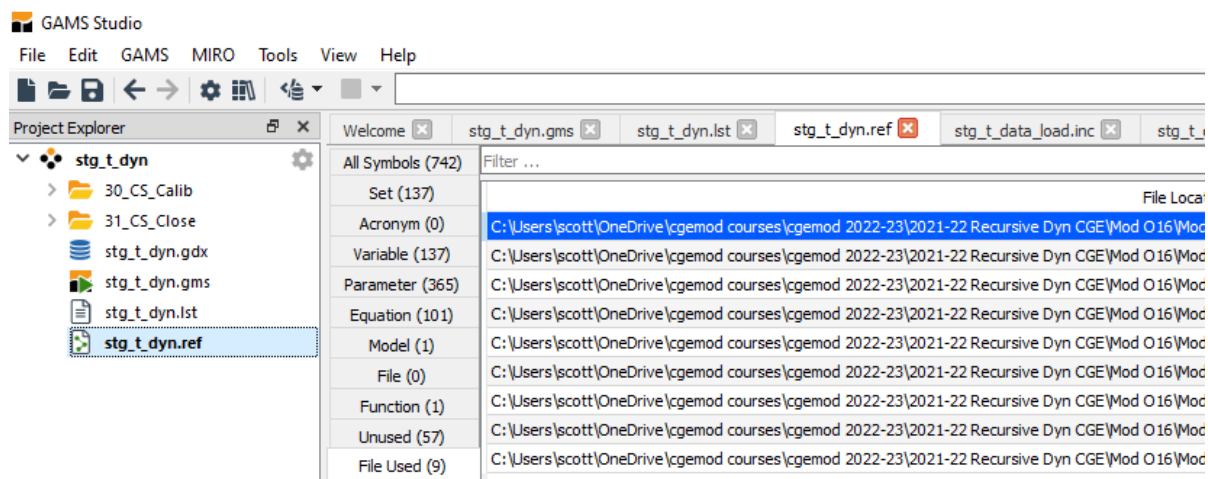


Figure 13.3c STAGE_t in Studio – Opening & Finding Files

```

Total time = 2781 Ms
--- . stg_t_data_load.inc(149) 2 Mb
--- GDxin=C:\Users\scott\OneDrive\cgemod courses\cgemod 20
--- GDX File ($gdxIn) C:\Users\scott\OneDrive\cgemod cours
--- . stg_t_data_load.inc(182) 3 Mb
--- GDxin=C:\Users\scott\OneDrive\cgemod courses\cgemod 20
--- GDX File ($gdxIn) C:\Users\scott\OneDrive\cgemod cours
--- . stg_t_data_load.inc(545) 3 Mb
--- stg_t_dyn.gms(521) 3 Mb
--- . stg_t_diagnost.inc(557) 3 Mb
--- stg_t_dyn.gms(527) 3 Mb
--- . stg_t_parmcalib.inc(485) 3 Mb
--- GDxin=C:\Users\scott\OneDrive\cgemod courses\cgemod 20
--- GDX File ($gdxIn) C:\Users\scott\OneDrive\cgemod cours
--- . stg_t_parmcalib.inc(1477) 3 Mb
--- .. stg_t_calibchk.inc(165) 3 Mb
--- . stg_t_parmcalib.inc(1490) 3 Mb
--- stg_t_dyn.gms(797) 3 Mb
--- . stg_t_varinit.inc(403) 3 Mb
--- stg_t_dyn.gms(830) 3 Mb
--- . stg_t_samchk.inc(292) 3 Mb
--- stg_t_dyn.gms(841) 3 Mb
--- . stg_t_struct.inc(42) 3 Mb
--- GDxin=C:\Users\scott\OneDrive\cgemod courses\cgemod 20
--- GDX File ($gdxIn) C:\Users\scott\OneDrive\cgemod cours
--- . stg_t_struct.inc(517) 3 Mb
--- stg_t_dyn.gms(1539) 3 Mb

```

Alternatively, when the programme is compiled the process log identifies all the included files in GREEN. If the user clicks on an INCLUDE file that file will be opened if currently closed or brought to the front in the editor pane. Note how clicking on the reference to the *.gms file, in GREEN, immediately before each INCLUDE file is referenced will open the *.gms file on the line where the INCLUDE file is cited, e.g., *stg_t_dyn.gms* (521) where 521 identifies the line number.

Figure 13.3d STAGE_t in Studio – Open Files & Modular Structure

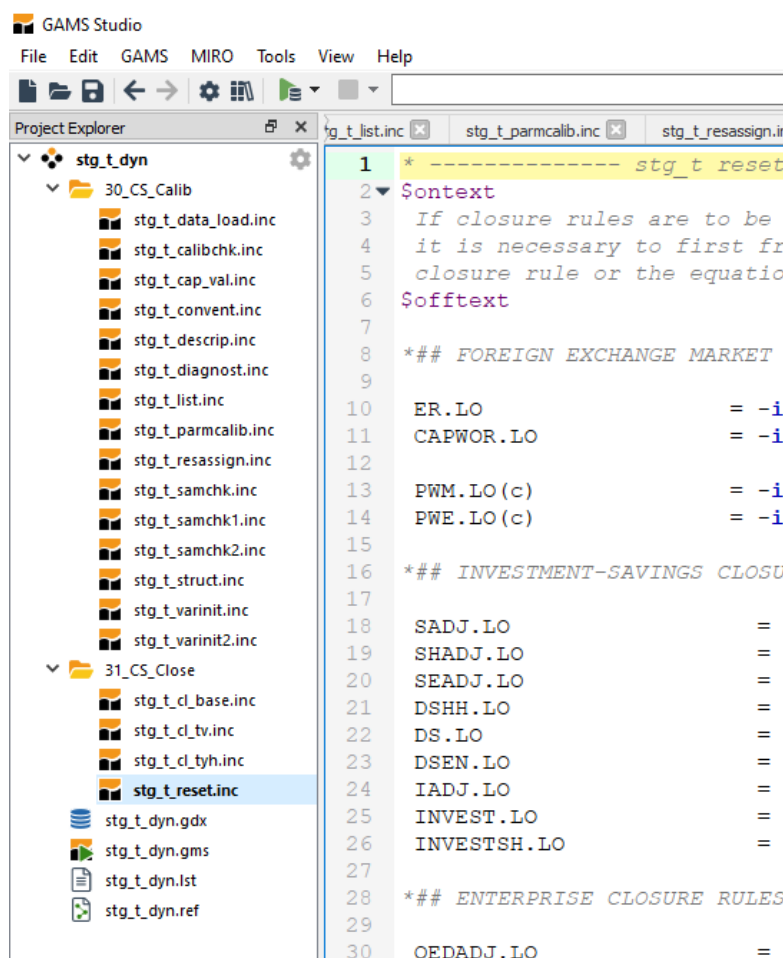


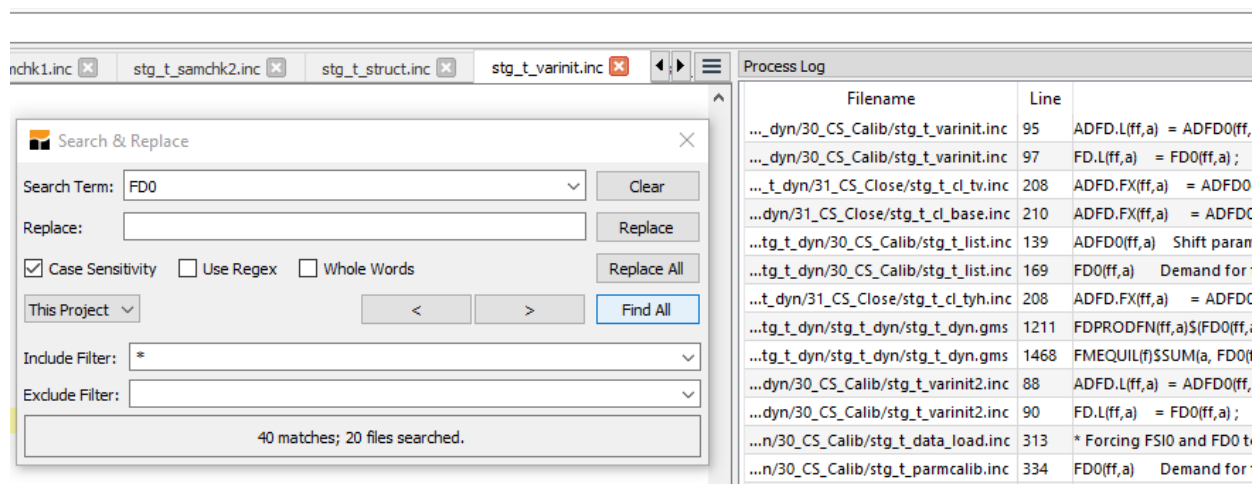
Figure 13.3d illustrates a situation where all the INCLUDE files in 2 sub directories (30_CS_Calib and 30_CS_Close) have been opened in Studio. Clearly since only 9 files have been used there must be unused files that have been opened; just because a file exists does not mean it is used. Also note that if a block of files are opened at the same time that they are in alphabetic order and not the order in which they are/were accessed, but if the files are opened individually they listed in the order in which they were opened. Having all the

contents of sub directories listed may involve clutter; the sub directories can be collapsed by clicking on the down arrow by each sub directory. If all sub directories are collapsed Studio would look like the case illustrated in Figure 13.3b. Right clicking in the Project Explorer offers the options to Collapse all projects or Expand All.

Search & Replace in Modular Models

The previous discussion of the Search & Replace options, in section 8, primarily focused on searching the contents individual files. In modular models it can be useful to search across multiple files. Studio offers 2 extremely useful Search & Replace search options that are especially useful when working with modular models.

Figure 13.4a Search and Replace in STAGE_t – ‘This Project’ filter

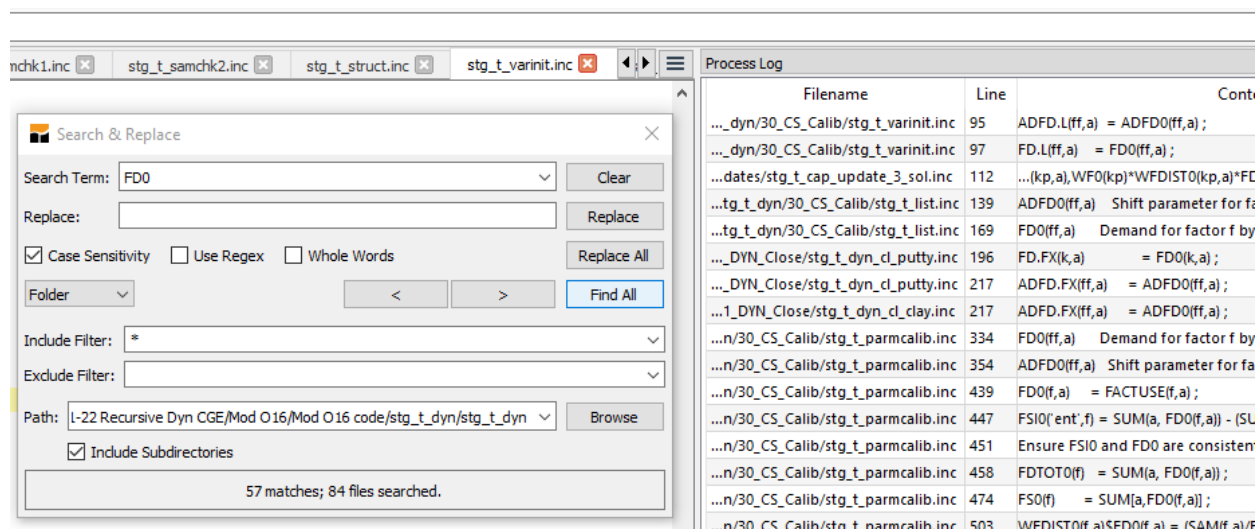


The first option is to define the Search over This Project (see the dropdown menu). The illustration in Figure 13.4a shows the results from a search for FD0 across the open files illustrated in Figure 13.3d, except for stg_t_dyn.lst that was closed prior to the search. This reports 10 matches; 20 files searched, and the Process Log provides links to each occurrence of the terms FD0, or the arrows < or > can be used to step through the matches. This illustrates several things to be considered: first, the definition of This Project is those files currently open in the Project Explorer, second, the search is over the files that are open NOT the files used in the model, and third references to FD0 in files used by the model are not identified. These second and third considerations

could be partially overcome by including the *.lst file in the search since the *.lst file will include contents from all the files used.

Another option is to extend the search over all files included in the folder, option Folder, that hosts the project. The results from this search are illustrated in Figure 13.4b, again the *.lst file was excluded. There are now 57 matches; 84 files searched, with each reference listed in the Process Log. Again the arrows < or > can be used to step through the matches.

Figure 13.7b Search and Replace in STAGE_t – ‘Folder’ filter



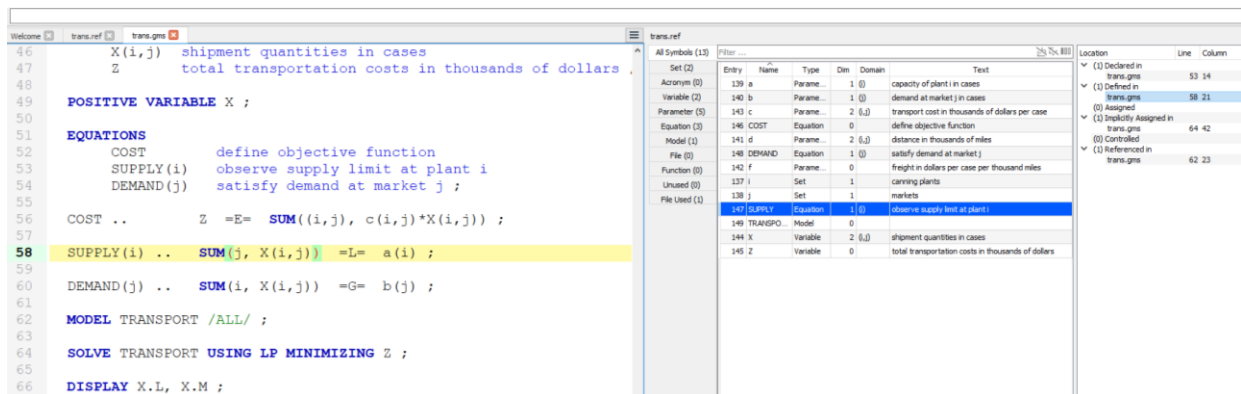
The Folder search option is flexible, e.g., the specific folder to be searched can be identified by the Path that can be found using Browse.

14. PIN View

“Studio provides a Pin View to carry a clone of a selected tab. This allows user to have two file contents visible at the same time” (GAMS Studio documentation, Sept 2022). The key advantage of the PIN view options is the facility that PIN offers to work interactively across (related) documents.

Suppose the user wants to use a reference file interactively with the main model file. The reference file can be pinned using – VIEW>PIN Right or PIN Below; this will pin a clone of the reference file to the right or below the editor pane and the editor pane can be used as normal. For the transport model PIN Right is illustrated in Figure 14.1a and PIN Below is illustrated in Figure 14.1b.

Figure 14.1a PIN Right View: trans.gms and trans.ref



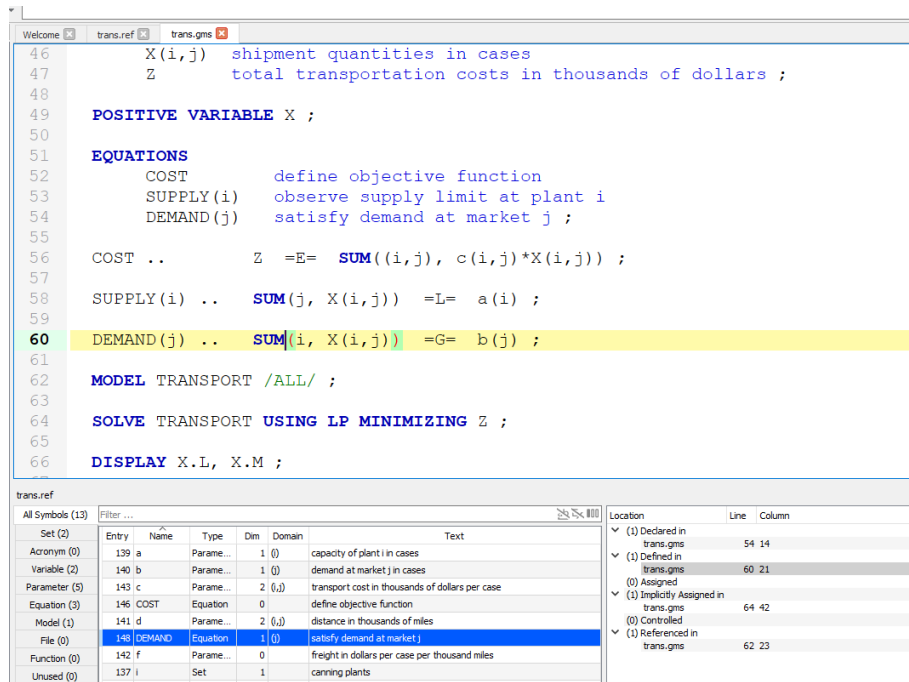
Entry	Name	Type	Dim	Domain	Text
Set (2)					
Acronym (0)	139 a	Parame...	1 (0)		capacity of plant i in cases
Variable (2)	140 b	Parame...	1 (0)		demand at market i in cases
Parameter (5)	143 c	Parame...	2 (0,1)		transport cost in thousands of dollars per case
Equation (3)	146 COST	Equation	0		define objective function
Model (1)	141 d	Parame...	2 (0,1)		distance in thousands of miles
File (0)	148 DEMAND	Equation	1 (0)		satisfy demand at market i
Function (0)	142 f	Parame...	0		freight in dollars per case per thousand miles
Unused (0)	137 i	Set	1		canning plants
File Used (1)	147 SUPPLY	Equation	1 (0)		observe supply limit at plant i
	149 TRANSPD	Model	0		
	144 X	Variable	2 (0,1)		shipment quantities in cases
	145 Z	Variable	0		total transportation costs in thousands of dollars

Once the reference file is pinned the user can work interactively to use the reference file to tracker symbols through the GAMS model.

Similarly, a GDX file can be pinned, and the user can then use the model file to identify how the various symbols (equations, variable, parameters, sets) are specified.

The PIN view is particularly useful when working with a modular model like that described above (section 13). If the reference file is pinned then when selecting the attribute (declared, assigned, referenced) of a symbol will open the pertinent file in the editor window See Figure 14.2. In this illustration a parameter, *ac*, was selected and the attribute chosen was ‘Assigned in’: this opened the file *stg_t_parmcalib.inc* at line 180.

Figure 14.1b PIN Below View: trans.gms and trans.ref

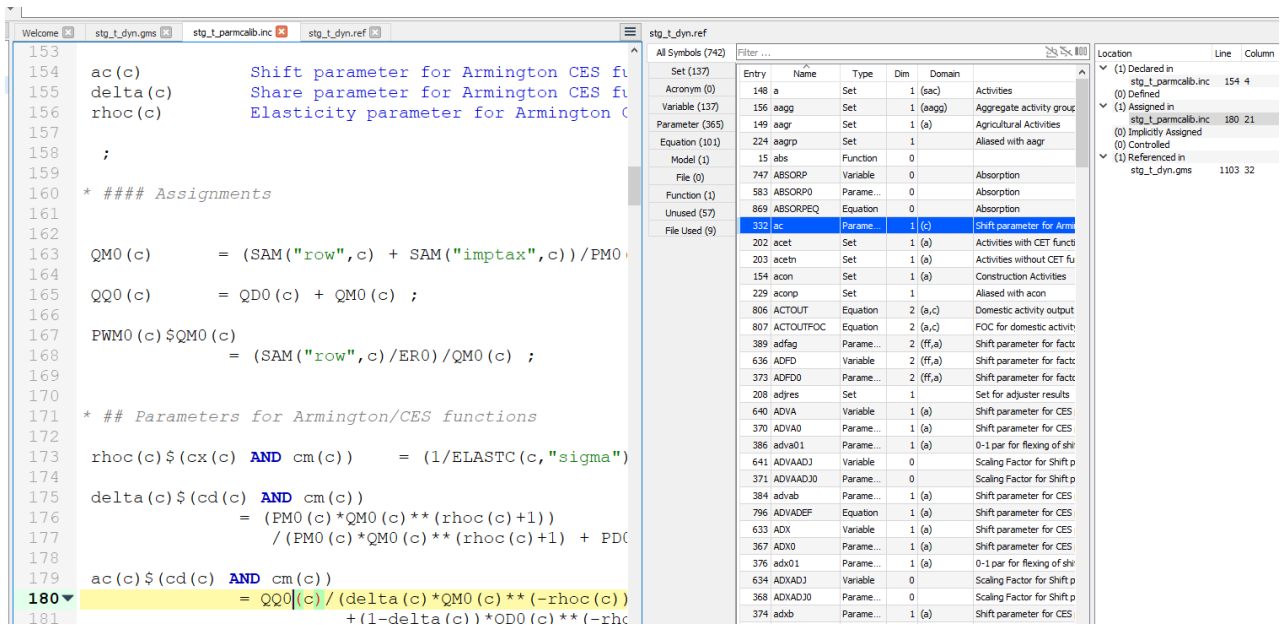


```

46      X(i,j)  shipment quantities in cases
47      Z       total transportation costs in thousands of dollars ;
48
49      POSITIVE VARIABLE X ;
50
51      EQUATIONS
52      COST      define objective function
53      SUPPLY(i) observe supply limit at plant i
54      DEMAND(j) satisfy demand at market j ;
55
56      COST ..   Z =E= SUM((i,j), c(i,j)*X(i,j)) ;
57
58      SUPPLY(i) .. SUM(j, X(i,j)) =L= a(i) ;
59
60      DEMAND(j) .. SUM(i, X(i,j)) =G= b(j) ;
61
62      MODEL TRANSPORT /ALL/ ;
63
64      SOLVE TRANSPORT USING LP MINIMIZING Z ;
65
66      DISPLAY X.L, X.M ;
  
```

Set (2)	Entry	Name	Type	Dim	Domain	Text
Acronym (0)	139	a	Param...	1 (0)		capacity of plant i in cases
Variable (2)	140	b	Param...	1 (0)		demand at market j in cases
Parameter (5)	143	c	Param...	2 (0,0)		transport cost in thousands of dollars per case
Equation (3)	146	COST	Equation	0		define objective function
Model (1)	141	d	Param...	2 (0,0)		distance in thousands of miles
File (0)	148	DEMAND	Equation	1 (0)		satisfy demand at market j
Function (0)	142	f	Param...	0		freight in dollars per case per thousand miles
Unused (0)	137	i	Set	1		canning plants

Figure 14.2 Reference File for a Modular Model: PIN Right View



```

153
154 ac(c)      Shift parameter for Armington CES fu
155 delta(c)  Share parameter for Armington CES fu
156 rhoc(c)   Elasticity parameter for Armington C
157
158 ;
159
160 * ##### Assignments
161
162
163 QM0(c)     = (SAM("row",c) + SAM("imptax",c))/PM0
164
165 QQ0(c)    = QD0(c) + QM0(c) ;
166
167 PWM0(c) $QM0(c)
168           = (SAM("row",c)/ER0)/QM0(c) ;
169
170
171 * ## Parameters for Armington/CES functions
172
173 rhoc(c) $(cx(c) AND cm(c)) = (1/ELASTC(c,"sigma"))
174
175 delta(c) $(cd(c) AND cm(c))
176           = (PM0(c)*QM0(c)**(rhoc(c)+1))
177             / (PM0(c)*QM0(c)**(rhoc(c)+1) + PD0
178
179 ac(c) $(cd(c) AND cm(c))
180           = QQ0(c) / (delta(c)*QM0(c)**(-rhoc(c))
181                 + (1-delta(c))*OD0(c)**(-rhoc
  
```

Set (137)	Entry	Name	Type	Dim	Domain	Text
Acronym (0)	148	a	Set	1 (a,c)		Activities
Variable (137)	156	aagg	Set	1 (aagg)		Aggregate activity group
Parameter (365)	149	aagr	Set	1 (a)		Agricultural Activities
Equation (101)	224	aagrp	Set	1		Allied with aagr
Model (1)	15	abs	Function	0		Absorption
File (0)	747	ABSORP	Variable	0		Absorption
Function (1)	583	ABSORPO	Param...	0		Absorption
Unused (57)	869	ABSORPEQ	Equation	0		Absorption
File Used (9)	332	ac	Param...	1 (c)		Shift parameter for Arm
	202	acet	Set	1 (a)		Activities with CET Funct
	203	acetrn	Set	1 (a)		Activities without CET fu
	154	acon	Set	1 (a)		Construction Activities
	229	acomp	Set	1		Allied with acon
	806	ACTOUT	Equation	2 (a,c)		Domestic activity output
	807	ACTOUTFOC	Equation	2 (a,c)		FOC for domestic activit
	389	adfaq	Param...	2 (ff,a)		Shift parameter for fact
	636	ADFD	Variable	2 (ff,a)		Shift parameter for fact
	373	ADFD0	Param...	2 (ff,a)		Shift parameter for fact
	208	adjres	Set	1		Set for adjuster results
	640	ADVA	Variable	1 (a)		Shift parameter for CES
	370	ADVA0	Param...	1 (a)		Shift parameter for CES
	386	advad01	Param...	1 (a)		0-1.par for flexing of shi
	641	ADVAADJ	Variable	0		Scaling Factor for Shift p
	371	ADVAADJ0	Param...	0		Scaling Factor for Shift p
	384	advab	Param...	1 (a)		Shift parameter for CES
	796	ADVADEF	Equation	1 (a)		Shift parameter for CES
	633	ADX	Variable	1 (a)		Shift parameter for CES
	367	ADX0	Param...	1 (a)		Shift parameter for CES
	376	advd1	Param...	1 (a)		0-1.par for flexing of shi
	634	ADXDJ	Variable	0		Scaling Factor for Shift p
	368	ADXDJ0	Param...	0		Scaling Factor for Shift p
	374	advdb	Param...	1 (a)		Shift parameter for CES

There are circumstances in which it can be helpful to view two files that are related.

Figure 14.3 illustrates the use of the PIN view to simultaneously view the equations in a model file (stg_t_dyn.gms) and the file where parameters in the equations are calibration stg_t_parmcalib.inc. Both files can be edited and saved as usual.



Figure 14.3 Viewing a Model and Parameter Calibration file: PIN Right View

```
stg_t_dyn.gms | stg_t_parmcalib.inc* | stg_t_dyn.ref | stg_t_parmcalib.inc*
;
##### 15. EQUATIONS ASSIGNMENTS #####
* ----- TRADE BLOCK -----
* ### Exports Block
* For some c there are no exports hence only implement for ce(c)
PEDEF(c)$ce(c).. PE(c) =E= PWE(c) * ER * (1 - TE(c))
                    - SUM(m,ioqttqe(m,c) * PTT(m)) ;
PE.FX(c)$ (NOT ce(c)) = 0.0 ;
* For some c there are no exports hence only implement for ce(c)
CET(c)$ (cd(c) AND ce(c))..
    QXC(c) =E= at(c) * (gamma(c) * QE(c) ** rhot(c) +
                    (1 - gamma(c)) * QD(c) ** rhot(c) ** (1/rhot(c)) ) ;
QXC.FX(c)$ (NOT cx(c)) = 0.0 ;
ESUPPLY(c)$ (cd(c) AND ce(c))..
    QE(c) =E= QD(c) * ((PE(c)/PD(c)) * ((1 - gamma(c))
                    / gamma(c)) ** (1/(rhot(c) - 1))) ;
QE.FX(c)$ (NOT ce(c)) = 0.0 ;
EDEMAND(c)$ced(c)..
    QE(c) =E= econ(c) * ((PWE(c)/pwse(c)) ** (-eta(c))) ;
* For c with no exports OR for c with no domestic production
* domestic supply is by CETALT
100 eta(c) export demand elasticity
101
102 ;
103
104 * ### Assignments
105
106 * ## Initial values for variables
107
108 QEO(c) = (SAM(c,"row") - SAM("exptax",c)
109          - SUM(m,TTTRANS(m,
110
111 * Using {QXC0(c) = (SUM(a,SAM(a,c))/PXC0(c))} .
112 * QD0(c) = QXC0(c) - QEO(c) ;
113
114 QD0(c) = (SUM(a,SAM(a,c))/PXC0(c)) - QEO
115
116 PWE0(c)$QEO(c)
117          = (SAM(c,"row")/ER0)/QEO(c) ;
118
119 * ## Parameters for CET functions
120
121 rhot(c)$ (cd(c) AND ce(c)) = (1/ELASTC(c,"oi
122
123 gamma(c)$ (cd(c) AND ce(c))
124          = 1/(1+PD0(c)/PE0(c) * (QE0(c)/QD0
125
126 * Using {QXC0(c) = (SUM(a,SAM(a,c))/PXC0(c))} .
127
128 at(c)$ (cd(c) AND ce(c))
129          = (SUM(a,SAM(a,c))/PXC0(c))
130          / (gamma(c) * QE0(c) ** rhot(c) +
131            QD0(c) ** rhot(c) ** (1/rho
132
```

15. Notes on Debugging a GAMS Programme in Studio

Computer programmes nearly always have numerous errors. GAMS Studio provides easy ways to find where the errors occur, especially compilation errors. The examples below are from the Syntax files in the examples provided for an introduction to Studio.

1. Compilation error messages appear in **RED** in the Process Log pane. Click on these messages and the editor window for the [Filename].gms file is chosen and the cursor moves to where the error was noticed. Clicking on the exclamation mark in the red circle by each error in the gms file shuts the view to where error is reported in the *.lst file.
 - In the [Filename].lst file you will find information about the type of error. The error is marked by **** and a \$#, where # is a number, on the line below where the error was noticed. More information on the meanings of the \$# codes is given where the error is in the list file (you can also search for the string 'Error Messages'). The information given is (usually) helpful.
 - A brief description of the nature of each error is reported in the *.lst file.

Compilation Errors

Compilation errors are (essentially) syntax errors. In the listing file GAMS will provide useful suggestions about the likely cause of the error message. The listing file also contains markers that make it easy to find the compilation errors: search for the text string "*****".

Some general principles may be helpful

- start from the top of the programme and work down – this one reason for selecting the CErr option and setting it to a low(ish) number;
- solve each error as it appears - do not skip onto the next error without a good **programming** reason;
- do not make too many changes at a time - as you become more familiar with GAMS and compilation errors the number of errors corrected at each stage will increase, but when starting out it is easy to compound errors;
- if substantial changes are made to the code save the input file with a revised name before running the file.

Common syntax errors include the following

- failing to end an operation with a “;” - GAMS often identifies this type of error as occurring on the next line of code or at the next keyword;
- assigning or using a parameter or variable before it has been declared;
- using a parameter or variable before it has been declared or assigned;
- spelling mistakes;
- the “*” used for comments and/or to comment out lines of code is NOT the first item in the line of code (NB: a space is an item in a line of code);
- set operations trying to use sets that are already under control - this is where the aliases become very useful.

The solutions to most syntax problems are relatively simple. Controlling sets is one type of syntax error that can prove somewhat less tractable.

Execution Errors

Execution errors are trickier. They can often arise because the model has been incorrectly specified. There are some ways to help find execution errors.

- DISPLAY statements for parameters and initial values for variables can be used to check the values returned by the programme.
- Outputting all model information to GDX will later prove useful, i.e., using F10 in Studio.
- The listing of values for variables from the model should be consistent with the values from the database.
- Generating a reference file is useful (see above to achieve this as a default).
- If the model is consistent the left-hand and right-hand sides of the equations should equate, or at least contain no "significant" discrepancies. This can be checked in the equation listing (controlled by the “limrow” and “limcol” options linked to the solve statement). Searching for the text string “***” is the quick way to find errors. A discrepancy is indicated by the statement “LHS = <value>”, where the error is only likely to be important if the value is greater than about 1.0E-5. This indicates that the problem is probably associated with the definitions of the parameters and variables in that equation.



Practical CGE Modelling: Introduction to GAMS with GAMS Studio

As with compilation errors it is unwise to try and do too much at once. If substantial changes are made to the code, it is wise to do so in a new version of the input file.