Appendix 4.B: The RIM-toolbox, a MATLAB *problem-independent implementation*

This section aims to facilitate access to this new algorithm. Moreover, a problem independent implementation allows us to reduce entry costs to potential users. Firstly, we provide a brief overview of the different files involved and the role of each one within the full procedure. We then briefly explain how the model specific file should be changed and how to execute the main function.

Routines description

A short description of each *general* file (those which do not depend on the specific problem to be solved) is included below. This implementation aims to reduce the entry cost to new users.

- `RIM_EXE.m`

  This main function basically recovers initial information about the problem to be solved and other technical parameters as well. Additionally, it coordinates the steps to be carried out along the process.

- `RIMn.m`

  This function groups the steps relating to the RIM implementation if the problem has n decision variable(s).

- `RIMn_getting_fields.m`

  This function initialises the structure array which summarize each execution if the problem has n decision variable(s).

- `RIMn_interval.m`

  At a given interval, this function carries out the intra-stage algorithm if the problem has n decision variable(s).

- `RIMntransition.m`

  Once selected intervals have been established, this function determines the initial intervals to be used in the next stage, if the problem has n decision variable(s).

- `RIM_ANALYSER.m`

This function allows to recover the results from previous executions. It offers different options which may be divided into those producing a graph and others producing a list of compatible values.

- `RIM_graph.m`

  This function organizes the previous information in order to draw compatible values detected.

- `RIM_obs_res.m`

  This function lists the different compatible values obtained for each interval. It also contains different choices which differ in the different points included.

- `RIM_spy.m`

  This auxiliary function has been adapted from standard function `spy` in order to be used within this set of routines. It mainly shows the non-zero values (those points satisfying the required precision), given a grid relative to a specific stage and interval.

- `RIMcomb_index.m`

  It enables to consider different values for each parameter of the model. The function uses these vectors to organise the different scenarios to solve the problem.

Some examples are also included to be used as a template if a new problem is implemented. Based on the different uses of RIM explained above (see table 4.1 for further information), the files are classified to provide proper starting points.

- *RIM*1: Constrained problem with 1 decision variable.

  `LS_QL.m`, `LS_CD.m`, `LS_CES0p5.m`, `D_QL.m`, `D_CD.m` and `D_CES0p5.m`.

- *RIM*1*u*: Unconstrained problem with 1 decision variable.

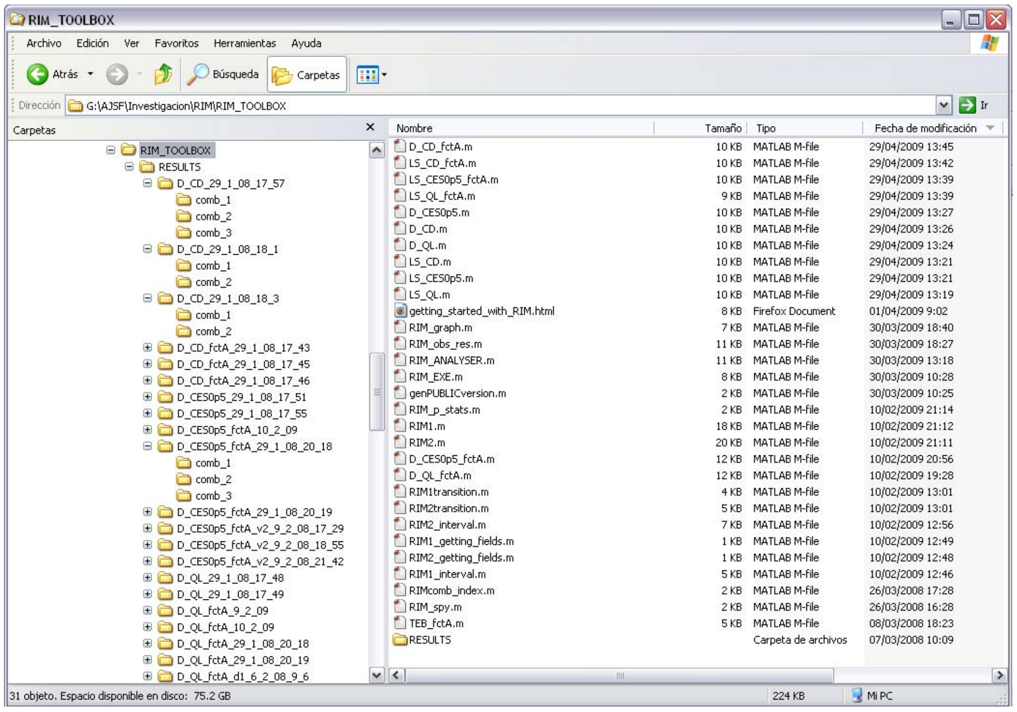
  `LS_QL_fctA.m`, `LS_CD_fctA.m`, `LS_CES0p5_fctA.m` and `D_CD_fctA.m`.

- *RIM*2: Constrained problem with 2 decision variables.

  `D_QL_fctA.m` and `LS_CES0p5_fctA.m`.

Before moving on to the next section, which contains which general information on the first steps with this algorithm, we will briefly discuss the folder structure that is created using these routines. Figure 4.5 shows the folder structure of this toolbox. From the baseline folder (`RIM_TOOLBOX`), *log* files (which allow to follow the resolution process in detail) and *mat* files (which save the information to be used later within MATLAB) are the result of each execution. All the files relative to this execution (log and mat files) are saved into a specific folder labelled according to the name of the specific file of the problem plus the execution date. Furthermore, a subfolder `comb_K` is created for each combination of parameters of the problem, where K is the order of the combination.

Fig. 4.5: Folder structure of RIM-toolbox



G:\AJSF\Investigacion\RIM\RIM_TOOLBOX

```
RIM_TOOLBOX
  RESULTS
    D_CD_29_1_08_17_57
      comb_1
      comb_2
      comb_3
    D_CD_29_1_08_18_1
      comb_1
      comb_2
    D_CD_29_1_08_18_3
      comb_1
      comb_2
    D_CD_fctA_29_1_08_17_43
    D_CD_fctA_29_1_08_17_45
    D_CD_fctA_29_1_08_17_46
    D_CES0p5_29_1_08_17_51
    D_CES0p5_29_1_08_17_55
    D_CES0p5_fctA_10_2_09
    D_CES0p5_fctA_29_1_08_20_18
      comb_1
      comb_2
      comb_3
    D_CES0p5_fctA_29_1_08_20_19
    D_CES0p5_fctA_v2_9_2_08_17_29
    D_CES0p5_fctA_v2_9_2_08_18_55
    D_CES0p5_fctA_v2_9_2_08_21_42
    D_QL_29_1_08_17_48
    D_QL_29_1_08_17_49
    D_QL_fctA_9_2_09
    D_QL_fctA_10_2_09
    D_QL_fctA_29_1_08_20_18
    D_QL_fctA_29_1_08_20_19
    D_QL_fctA_d1_6_2_08_9_6
```

Nombre	Tamaño	Tipo	Fecha de modificación
D_CD_fctA.m	10 KB	MATLAB M-file	29/04/2009 13:45
LS_CD_fctA.m	10 KB	MATLAB M-file	29/04/2009 13:42
LS_CES0p5_fctA.m	10 KB	MATLAB M-file	29/04/2009 13:39
LS_QL_fctA.m	9 KB	MATLAB M-file	29/04/2009 13:39
D_CES0p5.m	10 KB	MATLAB M-file	29/04/2009 13:27
D_CD.m	10 KB	MATLAB M-file	29/04/2009 13:26
D_QL.m	10 KB	MATLAB M-file	29/04/2009 13:24
LS_CD.m	10 KB	MATLAB M-file	29/04/2009 13:21
LS_CES0p5.m	10 KB	MATLAB M-file	29/04/2009 13:21
LS_QL.m	10 KB	MATLAB M-file	29/04/2009 13:19
getting_started_with_RIM.html	8 KB	Firefox Document	01/04/2009 9:02
RIM_graph.m	7 KB	MATLAB M-file	30/03/2009 18:40
RIM_obs_res.m	11 KB	MATLAB M-file	30/03/2009 18:27
RIM_ANALYSER.m	11 KB	MATLAB M-file	30/03/2009 13:18
RIM_EXE.m	8 KB	MATLAB M-file	30/03/2009 10:28
genPUBLICversion.m	2 KB	MATLAB M-file	30/03/2009 10:25
RIM_p_stats.m	2 KB	MATLAB M-file	10/02/2009 21:14
RIM1.m	18 KB	MATLAB M-file	10/02/2009 21:12
RIM2.m	20 KB	MATLAB M-file	10/02/2009 21:11
D_CES0p5_fctA.m	12 KB	MATLAB M-file	10/02/2009 20:56
D_QL_fctA.m	12 KB	MATLAB M-file	10/02/2009 19:28
RIM1transition.m	4 KB	MATLAB M-file	10/02/2009 13:01
RIM2transition.m	5 KB	MATLAB M-file	10/02/2009 13:01
RIM2_interval.m	7 KB	MATLAB M-file	10/02/2009 12:56
RIM1_getting_fields.m	1 KB	MATLAB M-file	10/02/2009 12:49
RIM2_getting_fields.m	1 KB	MATLAB M-file	10/02/2009 12:48
RIM1_interval.m	5 KB	MATLAB M-file	10/02/2009 12:46
RIMcomb_index.m	2 KB	MATLAB M-file	26/03/2008 17:28
RIM_spy.m	2 KB	MATLAB M-file	26/03/2008 16:28
TEB_fctA.m	5 KB	MATLAB M-file	08/03/2008 18:23
RESULTS		Carpeta de archivos	07/03/2008 10:09

31 objeto. Espacio disponible en disco: 75.2 GB 224 KB Mi PC

4.B.2 Getting started with RIM

This subsection seeks to explain how the Rational Iterative Multisection (RIM) algorithm may be used. In order to avoid discrepancies between this document and the routines, only basic instructions are explained here and more details are included in the help sections of the routines. Further information is available at `help <routine name>`

- **How should a new problem be implemented?**

  This toolbox only requires to create a new file (`new_model.m`, hereafter) where all the specific information relative to problem must be included. This file is organized using blocks in which different information about it (parameters, other relevant variables, etc.) should be introduced. Some examples have been distributed within this version in order to facilitate this process. It is strongly recommended to follow the instructions described as they explain how each block should be completed. Next step is to solve the problem.

- **How should a previously implemented problem be solved?**

  Only the function `RIM_EXE` is necessary, although other auxiliary functions are executed through this main function. After starting a MATLAB session, set the folder where you have all RIM routines as "Current Directory"[1]. Next, execute the command `RIM_EXE('new_model')` and problem will be solved. More optional parameters may also be specified within the previous command. For further information, see `help rim_exe` in the MATLAB prompt.

- **How should obtained results be analysed?**

  This routine set includes some tools implemented jointly within the function `RIM_ANALYSER` although some of them could be used separately. Some analyse the solution graphically whereas others do so from a numerical point of view.

  The parameter tool allows to choose which tool is executed. A basic example is included below:

  ```
  RIM_ANALYSER('D_QL','29_1_08_17_48',...
  'observing_results',2,[1 2 3],[1 1 1],[1 5 10])
  ```

  where `'D_QL'` (`mod_file`) and `'29_1_08_17_48'` (fecha) and 2 (combs) help to recover the results and `'observing_results'` indicates the tool to be used. The other parameters control the stages and intervals to be considered. For further details, see `help rim_analyser` in the MATLAB prompt.

Notes

[1]This step is necessary only if you did not add the directory to the permanent Matlab Path