International Institute for
Applied Systems Analysis
IIASA www.iiasa.ac.at

User's Guide for MESSAGE III

Messner, S. and Strubegger, M.

IIASA Working Paper

WP-95-069

July 1995

Messner, S. and Strubegger, M. (1995) User's Guide for MESSAGE III. IIASA Working Paper. WP-95-069 Copyright © 1995 by the author(s). http://pure.iiasa.ac.at/4527/

Working Papers on work of the International Institute for Applied Systems Analysis receive only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work. All rights reserved. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage. All copies must bear this notice and the full citation on the first page. For other purposes, to republish, to post on servers or to redistribute to lists, permission must be sought by contacting repository@iiasa.ac.at

Working Paper

User's Guide for MESSAGE III

Sabine Messner and Manfred Strubegger

WP-95-69
July 1995

IIASA International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 807 □ Fax: +43 2236 71313 □ E-Mail: info@iiasa.ac.at

User's Guide for MESSAGE III

Sabine Messner and Manfred Strubegger

WP-95-69
July 1995

Working Papers are interim reports on work of the International Institute for Applied Systems Analysis and have received only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work.

International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 807 □ Fax: +43 2236 71313 □ E-Mail: info@iiasa.ac.at

Contents

I Overview and Program Control of MESSAGE III 1

1 Introduction 3

1.1 Model Overview 3

1.2 Program Control 6

2 Running MESSAGE III 7

2.1 Matrix generation 8

2.2 Optimization 8

2.3 Sorting the solution 9

2.4 Producing tables 9

2.5 Analyzing the solution 9

2.6 Report of input data 10

2.7 A sample MESSAGE command file 10

3 Directory Structure and Files 11

3.1 Definition of Directories and Files 11

3.1.1 The Profile — `mms.pro` 11

3.1.2 The Directory File — `glob.dir` 13

3.1.3 The Region Directory File — `region.dir` 14

3.2 Files for the Matrix Generator — MXG 15

3.3 Files for the Optimizer — OPT 15

3.4 Files for the solution sorter — MKSF 16

3.5 Files for the report writer — REPO 16

3.6 Files for the calculation program — CAP 16

3.7 Files for the solution analyzer — ANA 17

II Input Data for MESSAGE III 19

4 General Remarks 21

5 Organization of the Data Bank 22

5.1 Organization of the TDB 23

5.2 Organization of ADB and LDB 23

5.3 Organization of the Update-Files 24
5.4 Counters and Labels of Years 24
5.5 Abbreviations used in the Formats 26
5.6 Format of Time Series 26
5.7 Identification of TDB: keyword `TDB:` 27
5.8 Identification of ADB: keyword `ADB:` 27

6 General Definitions **28**
6.1 Problem Identification: keyword `problem:` 28
6.2 Comment lines: keyword `description:` 28
6.3 Discount Rate: keyword `drate:` or `vdrate:` 28
6.4 Time Horizon: keyword `timesteps:` 28
6.5 Load Regions: keyword `loadregions:` 29

7 Energy Forms **31**
7.1 Energy Forms in the TDB: keyword `fuelidentification:` 31
7.2 Energy Forms in the ADB and LDB: keyword `energyforms:` 31

8 Demand Data **35**
8.1 Demands: keyword `demand:` 35
8.2 Load Curve: keyword `loadcurve:` 35

9 User-defined Relations **37**
9.1 User-defined Relations: keyword `relations[1/2]:` 37

10 Conversion Technologies **40**
10.1 Conversion Technologies: keyword `systems:` 40

11 Resource Data **54**
11.1 Resource Data: keyword `resources:` 54

12 Imports and Exports **59**
12.1 Import/Export Data: keywords `imports, exports:` 59

13 The General Definition File **63**

14 Control Input for the Matrix Generator **66**

15 Control Input for the Report Writer **67**

III The Post Processor of MESSAGE III 69

16 Introduction **71**
16.1 Introduction to the Application of CAP 71

17 Description of Input Files **72**
17.1 Control Variable File (`xx.tit`) 72
17.1.1 Content of blocks 73
17.2 Calculating Instructions (`xx.cin`) 76
17.2.1 Content of blocks 76
17.2.2 Format of mathematical operations 78
17.2.3 Control characters 79

18 Description of Data Files **81**
18.1 MESSAGE III Technology Name File (`region.tn`) 81
18.2 LP Solution File (`region_scen.dir`) 81
18.3 MESSAGE III Dump Files 81
18.4 Supplementary File (`region.fil`) 82
18.5 MEDEE-2 Dump File 83
18.6 Variable Location in the MEDEE-2 Dump File 84

19 Description of Output Files **85**
19.1 Tabular Output (`cin_scen.tab`) 85
19.2 Interface to Graphical Output 86

20 Description of Intermediate Files **87**
20.1 Interpolated Time Series Storage 87
20.2 Input Format Conversion 87

21 Description of Variables **88**
21.1 Variables Read from the MESSAGE III Dump Files 88
21.1.1 Technology variables 88
21.1.2 User defined relations 90
21.1.3 Demands 90
21.1.4 Other variables 90
21.2 Variables Read from the LP Solution File 90
21.3 Variables Read from the Supplementary Input File 91
21.4 Variables Read from the MEDEE-2 Dump File 91
21.5 Use of Internally Stored Values 92
21.6 Operators 92

22 Supporting Programs **95**
22.1 The Dimensioning Program CHDIM 95
22.2 Sorting the Solution — Program MKSF 96
22.3 Graphical Output 97
22.4 Analyzing the Solution — Program ANA 97

23 How to Get Started **99**
23.1 Complete Sets of Input Files to CAP 99
23.1.1 Calculations processing MEDEE-2 results 99
23.1.2 Calculations processing a MESSAGE III dump and a LP solution file 102
23.2 Samples for Mathematical Expressions used in CAP 104
23.3 Using CAP to Produce Tables 106
23.4 Using CAP in Interactive Mode 108

IV The Formulation of MESSAGE III **109**

24 Introduction **111**

25 Conversion Technologies **113**
25.1 Variables 113
25.1.1 Activities of Energy Conversion Technologies 113
25.1.2 Capacities of Energy Conversion Technologies 114
25.2 Constraints 114
25.2.1 Capacity Constraints 114
25.2.2 Upper Dynamic Constraints on Construction Variables 117
25.2.3 Lower Dynamic Constraints on Construction Variables 117
25.2.4 Upper Dynamic Constraints on Activity Variables 118
25.2.5 Lower Dynamic Constraints on Activity Variables 118

26 Domestic Resources **119**
26.1 Variables 119
26.1.1 Resource Extraction Variables 119
26.2 Constraints 119
26.2.1 Resource Availability per Grade 120
26.2.2 Maximum Annual Resource Extraction 120
26.2.3 Resource Depletion Constraints 120
26.2.4 Maximum Annual Resource Extraction per Grade 121
26.2.5 Upper Dynamic Resource Extraction Constraints 121
26.2.6 Lower Dynamic Resource Extraction Constraints 122
26.2.7 Dynamic Extraction Constraints per Grade 122

27 Imports and Exports **123**
27.1 Variables 123
27.1.1 Import Variables 123
27.1.2 Export Variables 123
27.2 Constraints 124
27.2.1 Imports per Country 124

27.2.2 Maximum Annual Imports . . . 124
27.2.3 Maximum Annual Imports per Country . . . 125
27.2.4 Upper Dynamic Import Constraints . . . 125
27.2.5 Lower Dynamic Import Constraints . . . 125
27.2.6 Dynamic Import Constraints per Country . . . 126
27.2.7 Constraints on Exports . . . 126

28 Energy Flows **127**
28.1 Constraints . . . 127
28.1.1 Demand Constraints . . . 127
28.1.2 Distribution Balance . . . 128
28.1.3 Transmission or Transportation Balance . . . 129
28.1.4 Central Conversion Balance . . . 129
28.1.5 Resource Extraction, Export and Import Balance . . . 130
28.1.6 Resource Consumption . . . 131

29 Stock-piles **132**
29.1 Variables . . . 132
29.1.1 Stock-pile Variables . . . 132
29.2 Constraints . . . 132
29.2.1 Stock-piling Constraints . . . 132

30 User-defined Relations **134**
30.1 Constraints . . . 134
30.1.1 Relation without Load Regions . . . 134
30.1.2 Relation with Load Regions . . . 135
30.1.3 Construction of Relations between Periods . . . 136
30.1.4 Special Handling of Demand Elasticities . . . 137

31 Objective and Cost Counters **138**
31.1 Constraints . . . 138
31.1.1 Cost Accounting Rows . . . 138
31.1.2 The Objective Function . . . 138

32 Special Features of the Matrix Generator **141**
32.1 The Time Horizon–Discounting the Costs . . . 141
32.2 Distributions of Investments . . . 141
32.3 The Load Curve . . . 143
32.4 Consideration of Load Variations in Conversion Technologies . . . 143
32.5 Lag Times Between Input and Output of a Technology . . . 144
32.6 Variable Inputs and Outputs . . . 144
32.7 The Contribution of Capacities Existing in the Base Year . . . 145

32.8 Capacities which Operate Longer than the Time Horizon 145
32.9 Own-Price Elasticities of Demand 146
32.10Supply Elasticities 147
32.11The Mixed Integer Option 149
32.12Nonlinearitites 149
32.13Multiobjective Optimization 150
Bibliography 152
Index 153

List of Tables

3.1 Example of `mms.pro` 12
3.2 Example of `mms.def` 12
3.3 Example of `glob.reg` 13
3.4 Example of `glob.dir` 13
3.5 Example of `region.dir` 14
3.6 Example of `region.use` 16

5.1 Keywords organizing the TDB 23
5.2 Keywords organizing the ADB and LDB 24
5.3 Labels and counters used in the format descriptions 25
5.4 Abbreviations used in the formats 26
5.5 Input switches for time series 26

6.1 Keywords for defining load regions 29

9.1 Keywords for user-defined relations 37

10.1 Keywords for conversion technologies 41
10.2 Format and occurence of keywords for conversion technologies 42

11.1 Keywords for resources 54
11.2 Keywords for grades of resources 55
11.3 Format and occurence of keywords for resources 55

12.1 Keywords for imports and exports 59
12.2 Format and occurence of keywords for import and export 60

13.1 Keywords in the GEN-file 63

14.1 Output control for the matrix generator. 66

15.1 List and description of control input for the report writer 68

Part I

Overview and Program Control of MESSAGE III

Chapter 1

Introduction

MESSAGE is a dynamic linear programming model designed for the optimization of energy supply and utilization. MESSAGE II was developed at IIASA on the basis of MESSAGE, a **M**odel for **E**nergy **S**upply **S**trategy **A**lternatives and their **G**eneral **E**nvironmental Impact [1]. The current version, MESSAGE III , has been developed on the basis of MESSAGE II [2]. The main emphasis in this development was put on a user-friendly interface for data entry and all program calls necessary. The data structures have been changed to a databank based on keywords, which simplifies automated data processing. In the mathematical formulation some options have been eliminated from the model, and the multiobjective option has been developed further. The reference point optimization method [3], adapted to dynamic modelling into a reference trajectory optimization method, is implemented.

MESSAGE III has the option to define investment variables as integer, which, in conjunction with an adequate solving algorithm (e.g., CPLEX [4], OSL [5] or MOPS [6] with the MIP facility) can be used to optimize mixed integer programming models. In connection with MINOS [7] it can also handle nonlinear objective functions.

The underlying principle of all MESSAGE models is the optimization of an objective function under a set of constraints defining the feasible region, containing all possible solutions of the problem. The objective helps to choose the solution considered best according to the criteria specified.

1.1 Model Overview

Energy Flows: Energy Carriers and Technologies

In first approximation MESSAGE could be labelled a physical flow model. Given a vector of demands for specified goods or services, it assures sufficient supply, utilizing the technologies and resources considered. In its usual application the model is used to evaluate energy systems, but any other problem dealing with commodity flows, where specified demands are to be met by a given set of supply options could be modelled.

The backbone of MESSAGE is the technical description of the modelled system. This includes the definition of the categories of energy forms considered, like, e.g., primary energy, final energy, useful energy, and the energy forms (commodities) actually used, e.g., coal or district heat, but also energy services, e.g., the tons of steel or useful space heat provided by the use of energy. The technologies are defined by their inputs and

outputs, the efficiency and the degree of variability, if more than one input or output exists, e.g., the possible production patterns of a refinery or a pass-out-turbine.

By all these definitions of energy carriers and technologies a so-called energy chain is structured, where the energy flows from supply to the demand side. The supplying energy carriers can belong to all categories except useful energy, they have to be chosen in light of the actual problem. Limits on the amounts available inside the region/area and import possibilities have to be specified. Together with the demands, that are exogenous to the model, the technical system provides the basic set of constraints: The demands have to be met by the energy flowing from domestic resources and imports through the modelled energy chain.

Capacities and Investments

The amount and quality of obtainable information can be increased considerably by accounting for existing installations and the need to construct new capacities of the technologies. By knowing which types of and how many installations are required to build up a desired system one can assess the effects on the economy.

The investment requirements can be distributed over the construction time of the plant and they can be subdivided into different categories to allow accounting for the requirements from some important industrial and commercial sectors. But also the needs for basic materials during construction of a technology as well as the utilization of non-energetic inputs during the operation of a plant can be accounted for, keeping track of the industrial branches they originate from in monetary terms or just accumulating the needs in physical units.

Demand Fluctuations: the Load Curve

For some energy carriers the timely availability causes considerable cost and management efforts. Electricity has to be provided by the utility at exactly the same time it is consumed. MESSAGE simulates this situation by subdividing each year into an optional number of so-called "load regions". The parts of the year aggregated into one load region can be chosen according to different criteria: just sorted according to the power requirements or aggregation of various typical parts of the year (summer/winter, day/night). The latter (semi-ordered) load representation opens the opportunity to model energy storage (e.g., transfer of energy from night to day, from summer to winter). Additionally, including a load curve improves the representation of power requirements and the utilization of different types of powerplants.

The Objective Function

Minimization of the total system costs is most commonly used as objective to choose a solution (this is also the default in MESSAGE). In this case for all costs occurring at later points in time the present value is calculated by discounting them to the first year of the calculation, the sum of the discounted costs represents the objective function value. Discounting makes the costs occurring in different points in time comparable, the discount rate chosen defines the weights different periods get in the optimization. In principle it should be equal to the long-term real interest rate, i.e. excluding inflation or any other opportunity costs. A high discount rate gives more weight or importance to

present expenditures than to future ones, while a low discount rate reduces these differences and thus favours investments decreasing the run-time expenditures for a technology.

The Time Horizon

The time horizon of a model application has to be chosen with regard to the problem; it could be long as well as short term. Even the use for a single point it time could give valuable results for complex problems. For the calculations this time horizon is split into periods of optional length, each of which is represented by a sample year in the model.

Absolute and Dynamic Limits

The development of the modelled system over time can be more or less predefined if relative or absolute limits for certain energy carriers or technologies are given. But additionally MESSAGE gives the possibility to introduce maximal or minimal growth (or decline) rates for the installation of new technologies and for the use of domestic and imported fuels. This allows to predefine a range of variability of the system over time, within that the model will dynamically choose an optimal strategy.

Exogenous Constraints

The most powerful module of MESSAGE III provides the opportunity to define constraints between all types of technology-related variables. The user could, among others, limit one technology in relation to some other technologies (e.g., a maximum share of wind energy that can be handled in an electricity network), give exogenous limits on sets of technologies (e.g., a common limit on all technologies emitting SO_2, that would be given in millions tons of SO_2), or define additional constraints between production and installed capacity (e.g., ensure take-or-pay clauses in international gas contracts forcing customers to consume a minimum share of the contracted level during summer months).

Multiobjective Optimization

MESSAGE supports two types of multiobjective optimization:

- Weights can be put on all technology variables, which yields a normal multiobjective approach, in that different types of activities (e.g., emissions of pollutants, energy imports) get different weights. The weights have to be comparable to each other and are to be chosen by the user.
- The Reference Trajectory Optimization approach [3] allows to define "reference levels" for each objective in the model, the optimization will–for all objectives–reach these levels as far as possible.[1] The advantage of this approach is, that no weights have to be defined and the reference levels are given in the natural units of the objective, e.g. in tons of SO_2 emitted.

[1]The optimization procedure of this method has to be included in the solving package, which has been accomplished for MINOS so far.

Other Features

Other features of MESSAGE III include

- demand and supply elasticities,
- stock-piling of fuels over the time horizon,
- inventories and last cores as they are required in nuclear reactors,
- unit sizes of new installations if a mixed integer programming solver is available, and
- nonlinear objective functions, if MINOS is used to solve the problem.

As far as these special features are not included in the mathematical formulation they are explained in chapter 32.

1.2 Program Control

Linear programming models generally use standard software packages for the optimization of a problem given in standardized format. Consequently, to run a model based on linear programming, at least two additional modules are required: a module generating the problem in the predefined format (usually called matrix) and a module reading the standard output and converting it to a readable format.
MESSAGE III includes some additional supporting modules. The following list gives all modules of MESSAGE III :

- MXG, the matrix generator,
- OPT, the optimizer (standard software package),
- REPO, the report writer,
- CAP, the post processor, and
- ANA, analyzing the solution.

Control over the software of MESSAGE III is performed by a single batch program, `run_msg`, that runs each of the modules of the software package. Chapter 2 describes the utilization and options of this program. The underlying directory structure and the accompanying definitions are described in chapter 3.

Chapter 2

Running MESSAGE III

The command to run MESSAGE III is

`run_msg`

When called without arguments `run_msg` gives help on it's usage:

```
usage: run_msg -f file
               file like $HOME/rundir/runfile
       run_msg -c module [args] -s scen region
               module: mxg, opt, mksf, cap, repo, ana
                       (opt includes mksf)
               args:
                  mxg:     -u upd_files  (full path)
                           -m             multi objective
                           -d             generate dumps only
                           -r             SCCS version (SID)
                  repo:    -x             -l long, -s short report
                  opt/rlp: -b             use old basis
                           -B basis       use basis
                           -p             use preprocessor (only MOPS)
                  cap:     -r x           region identifier
                           -R title2      second title line
                           -T title3      third title line
                              (titles must be missing in region.ti0 file!!)
                           [cinfile [-t titfile]] ...
                  ana/nbf: act, shp, lol, upl, at
```

`run_msg` can be used in two modes:

1. With flag `-f run_msg` will read a command file containing all commands to be executed. In this way a sequence of consecutive runs can be defined and executed in a queue.

2. In the other cases `run_msg` performes single commands that are defined in the command line.

In the second case `run_msg` requires the flags in a given order. The first flag, `-c`, defines the MESSAGE III module to be run. Then the flags specific for the module come, and finally the scenario has to be given (`-s`-flag).
The following modules can be selected after the flag `-c`:

- `mxg`: run the matrix generator,
- `opt` or `rlp`: run the optimization and sort the solution,
- `mksf`: just sort the solution,
- `repo`: run the report writer (tables of the input data),
- `cap`: run the post processor generating tables of the solution, and
- `ana` or `nbf`: prepare a file for an analysis of the solution.

2.1 Matrix generation

Matrix generation for MESSAGE III is started with the following command:

```
run_msg -c mxg [-u upd_file[, ...]...] -s scen region
```

The matrix generator reads the input data defining the problem. These input data are given in the technology data bank (TDB), the application data bank (ADB), the local or scenario data bank (LDB), and optional update files given after the flag `-u` separated by commas without blanks. The structure of these data bank files is described in the documentation of the input data for MESSAGE III , their location on the file system is described in chapter 3.2. The output of the matrix generator is the problem formulation in standardized format (MPS-format) and some files containing information used by other programs of MESSAGE III .

2.2 Optimization

Optimization for MESSAGE III is started with the following command:

```
run_msg -c opt [-b] [-B basis] -s scen region
```

Flag `-b` tells MESSAGE to use the old basis of the same problem, while flag `-B` can be set to use a basis of another problem or another scenario of the same problem. See section 3.3 for more details on the files read and created by the optimizer. If MESSAGE is started with flag `-c opt` the solution sorter will be started immediately after an optimal solution is reached. If the problem is infeasible, the sorter will not be started automatically and no tables can be created, then.
Optimization in MESSAGE III is performed using a standardized LP- or MIP-optimization package. Presently MESSAGE III supports the following optimization packages:

1. MINOS from Stanford University, that can also be used in conjunction with nonlinear formulations of the objective function or some constraints,

2. MOPS from Freie Universität Berlin, that can also handle binary variables (i.e. variables taking only 0 or 1 as value)
3. OSL from IBM that can be used for Mixed Integer Programming (SOS-Sets can only be used if the control program calling OSL is programmed accordingly),
4. CPLEX from Cplex Optimization, that can also be used for Mixed Integer Programming, and
5. HOPDM, an interior point optimizer supplied by IIASA.

The optimizer can be selected in the definition file `mms.def` (see table 3.2. Generally, the statistical output and iteration log is on file `region.itl`. For MOPS, the statistical information is on file `region.sta`.

2.3 Sorting the solution

Sorting the solution can be initiated separately from optimizing the problem. Generally this setp is included in optimization and there is no necessity to perform it separately. For special cases sorting can be called by

```
run_msg -c mksf -s scen region
```

MESSAGE will read the solution file as provided by the solver defined in file `region.min` in directory `intm` and produce the direct access file (see section 3.4).

2.4 Producing tables

The calculator program of MESSAGE III , CAP, is called by

```
run_msg -c cap [-r x] [-R title2] -T [title3] \
       [cinfile [-t titfile] ] ... -s scen region
```

If no `cinfile` is given, all `cinfiles` defined in `region.use` in the same directory as the `cinfiles`, see table 3.6, will be used to generate tables. In the absence of additional title information, the titlefile with extension tit will be used, if the -T-flag or -R and -T-flags are given, the titlefile with extension ti0 will be used under the assumption that the title lines given in the command line are missing in the file. In addition to the tables, definitions for the MESSAGE graphics program are generated. CAP is described in a separate part of the documentation of MESSAGE III (see chapter IV). Flag `-r` is used for postprocessing in case of large aggregated models as operated with the IIASA version of MESSAGE. Generation and operation of such models is not described in this document.

2.5 Analyzing the solution

The solution analyzer of MESSAGE III , ANA, is called by

```
run_msg -c ana type -s scen region
```

where `type` stands for one of the following:

- `act` print activity levels,
- `shp` print shadow prices (dual variables),
- `lol` print variables at lower limits,
- `upl` print variables at upper limits and
- `at` print status of variable/constraint (at lower/upper bond, in basis).

The solution analyzer produces a file called `region_scen.nbf` in subdirectory `res`.

2.6 Report of input data

To obtain a report of the input data of MESSAGE III , REPO has to be called:

```
run_msg -c repo -x -s scen region
```

where `-x` can be `-l` for a long report and `-s` for a short report. In the long report, a set of tables is created for each technology including time series as required, while the short report evaluates each technology in one line for the first model year.

2.7 A sample MESSAGE command file

MESSAGE can be used with a command file, that can start a sequence of full MESSAGE runs from matrix generation up to preparing tables. The runs would be started with:

```
usage: run_msg -f file
```

where `file` stands for a command file that has the following syntax:

```
region from pr1 to pr2 basis bassw scen scenario
```

where `from`, `to`, `basis` and `scen` are keywords and `region` stands for the region or case modelled, `pr1` is the first program to be run and `pr2` is the last program to be run. `bassw` can be one of the following: none, old, or a filename of a basis to be used. `scenario` is the scenario to be analysed. `pr1` and `pr2` can be either `mxg`, `opt` or `cap`. `pr2` has to be a program that runs after `pr1`.
The command file can contain a sequence of lines, the corresponding model runs will be performed in sequence. The following example shows a sample command file:

```
wor from mxg to cap basis old scen ref
wor from mxg to cap basis data/wor_ref.bsn scen co2
```

This command file runs region wor from matrix generation to table generation without basis for scneario ref and afterwards runs the same region for scneario co2 using the basis of scenario ref.

Chapter 3

Directory Structure and Files

3.1 Definition of Directories and Files

The directory structure and files to be used can be defined relatively freely for each installation of MESSAGE III . Environment variable `MMS_PRO`, which is usually set in the .login, .profile or autoexec.bat file of a user or PC, defines the file from that all further definitions are derived. It has to contain the absolute path to this file, which is usually called `mms.pro`. All files defined in `mms.pro` or in subsequent files are usually located in the same directory as `mms_pro`. All definition files in this directory contain definition lines with the items in a line separated by tabs or blanks. If the number of items can vary or an item can contain blanks, only tabs are allowed as separation character.

3.1.1 The Profile — `mms.pro`

`mms.pro` contains definitions of further definition files that are identified by keywords. It has four mandatory lines containing

— a header line

— the path to a file with general definitions (`mms.def`)

— the path to the file defining the model or regions (`glob.reg`)

— the path to the file defining the directories (`glob.dir`)

In addition one file has to be defined for each region or model: `region.dir` giving the data directories for the region. Table 3.1 gives an example for `mms.pro` containing the definition for 11 world regions and a world model.

General definitions are given in `mms.def`. Table 3.2 shows the structure of this file. The entries in the file are optional and define the editor used for interactive purposes, the optimizer called and a file containing region identifiers for global models aggregated from more than one reginoal model (The description of operating aggregated models is not included in this document).

Optimizers presently supported are:

- minos, see [7],

Table 3.1: Example of `mms.pro`

file	full filename
definitions	/p/ecs/rscen/mms.files/mms.def
glob.regfile	/p/ecs/rscen/mms.files/glob.reg
glob.dirfile	/p/ecs/rscen/mms.files/glob.dir
afr.dirfile	/p/ecs/rscen/mms.files/afr.dir
cpa.dirfile	/p/ecs/rscen/mms.files/cpa.dir
eeu.dirfile	/p/ecs/rscen/mms.files/eeu.dir
fsu.dirfile	/p/ecs/rscen/mms.files/fsu.dir
lam.dirfile	/p/ecs/rscen/mms.files/lam.dir
mea.dirfile	/p/ecs/rscen/mms.files/mea.dir
nam.dirfile	/p/ecs/rscen/mms.files/nam.dir
pao.dirfile	/p/ecs/rscen/mms.files/pao.dir
pas.dirfile	/p/ecs/rscen/mms.files/pas.dir
sas.dirfile	/p/ecs/rscen/mms.files/sas.dir
weu.dirfile	/p/ecs/rscen/mms.files/weu.dir
wor.dirfile	/p/ecs/rscen/mms.files/wor.dir

Table 3.2: Example of `mms.def`

editor	vi
optimizer	cplex
regid	/p/ecs/rscen/mms.files/regfile

- osl, see [5],
- mops, see [6],
- cplex, see [4] and
- hopdm, see [8].

The corresponding definitions of the regions or models are given in `glob.reg` (see table 3.3 for an example). The columns in `glob.reg` contain the following information:

— identification of region

— identifications of regions that are aggregated to generate output for this region

— description of region

— +: full region also optimized
-: region that is only aggregated for output

— name of technology data bank for this region (`agg` stands for aggregated regions here). If a region is not included in the `glob.reg` file the TDB has to have the same name as the ADB.

Table 3.3: Example of `glob.reg`

nam	nam	North America	+	ind
lam	nam	Latin America	+	dev
weu	weu	Western Europe	+	ind
eeu	eeu	Eastern Europe	+	ind
fsu	fsu	Former Soviet Union	+	ind
sas	sas	South East Asia	+	dev
pas	pas	Pacific Other Asia	+	dev
pao	pao	Pacific OECD	+	ind
afr	afr	Subsaharan Africa	+	dev
mea	mea	Middle East N.Africa	+	dev
cpa	cpa	Centr. pl. Asia & China	+	dev
wor	wor	WORLD	+	ind
ind	nam weu pao	Industrialized Countries	-	agg
res	fsu eeu	Restructuring Economies	-	agg
dev	lam pas sas afr mea cpa	Developing Countries	-	agg

3.1.2 The Directory File — `glob.dir`

`glob.dir` identifies the general directory structure, i.e. where the TDB's, sources, executables, shellfiles etc. are located on the file system. Entries that are not required in an application can be omitted. Table 3.4 shows the definitions of the IIASA installation.

Table 3.4: Example of `glob.dir`

#call	answer
rundir	/p/ecs/rscen/rundir
tdb	/p/ecs/rscen/supply/tdb
cin	/p/ecs/rscen/supply/cin
grf	/p/ecs/rscen/supply/grf
src	/p/ecs/rscen/src
obj	/p/ecs/rscen/obj
bin	/p/ecs/rscen/bin
optim	/p/ecs/rscen/obj
tools	/p/ecs/rscen/obj
help	/p/ecs/rscen/src/help/e
specs	/p/ecs/rscen/supply/specs

The first column in `glob.dir` identifies the following options:

— `rundir`: directory for instruction files for `run_opt`.

— `tdb`: directory with all TDB's

— `cin`: directory with definition of cin-files for each region (the name of the definition files is `region.use`, the file structure is described in table 3.6.

— `src`: path to source codes and definition files of MESSAGE III (if available, only interactive)

— `obj`: path to executable codes of MESSAGE III

— `bin`: path to shell files of MESSAGE III

— `optim`: path to optimizer

— `tools`: path to supporting programs

— `help`: path to help files

— `specs`: path to standardized specifications (for report writing, optimization control etc.)

3.1.3 The Region Directory File — `region.dir`

`region.dir` identifies the directory structure for region `region`, i.e. where the ADB, CIN-file, LDB etc. are located. Table 3.5 shows the definitions of the IIASA installation.

Table 3.5: Example of `region.dir`

#call	answer
supply	/p/ecs/rscen/supply/nam
demand	/p/ecs/rscen/demand/nam
cin	/p/ecs/rscen/supply/cin
tdb	/p/ecs/rscen/supply/tdb
adb	/p/ecs/rscen/supply/nam/data
ldb	/p/ecs/rscen/supply/nam/data
gen	/p/ecs/rscen/supply/nam/data
data	/p/ecs/rscen/supply/nam/data

The first column identifies the following options:

— `supply`: directory where MESSAGE III runs are performed

— `demand`: reserved for demand module

— `cin`: directory with CIN-files

— `tdb`: directory with TDB

— `adb`: directory with ADB

— `ldb`: directory with LDB

— `gen`: directory with GEN-file, Title-file and file with additional information (FIL)

— `data`: supplementary data directory

3.2 Files for the Matrix Generator — MXG

The matrix generator reads the following files:

— TDB (mandatory, defined in `glob.reg`),

— GEN (optional, path given in `region.dir`, name is `region.gen`),

— ADB (mandatory, path given in `region.dir`, name is `region.adb`),

— LDB (mandatory, path given in `region.dir`, name is `region_scen.ldb`), and

— Update-files given with path in the call of `run_msg`.

and writes the following files (all in subdirectory `intm` of item `supply` in `region.dir`)

— matrix, name is `region.mat` or `region.mps` (for CPLEX),

— general dump, name is `region.dpi`,

— resource dump, name is `region.dpr`,

— dump for additional relations, name is `region.dpa`,

— dump for variable entries, name is `region.dpv`,

— dump for bounds, name is `region.dpb`,

— definitions for sorting the solution (module `mksf`), file `region.min` (e.g. solver to be used, name of region, time steps),

— additional technology descriptions from comment lines for module `repo`, file `region.tn`, and

— additional resource descriptions from comment lines for module `repo`, file `region.rn`.

Some of the above files, which are only used to transfer information between the modules of MESSAGE, are no formatted ASCII-files.

3.3 Files for the Optimizer — OPT

The optimizer used by MESSAGE reads the matrix written by the matrix generator (`region.mat` or `region.mps`), the control input generated by `run_opt` and, if required, the basis of a previous model run. It writes the solution in standardized form (`region.sol` or `region.txt` in subdirectory `intm`) and the basis of the new solution (`region.bsn` in subdirectory `data`), as well as some statistical output, that differs among the various optimiziers.

3.4 Files for the solution sorter — MKSF

The program sorting the solution that was produced by the optimizer and generating a file containing all information as a binary hash-table, `mksf`, reads the solution (`region.sol` or `region.txt`) and the specific information on `region.min` that identifies the time steps, discount rate and optimizer. It writes a direct access file, `region_scen.dir`, with the sorted solution, `region_scen.int` and `region_scen.rec` containing the record length of the direct access file.

3.5 Files for the report writer — REPO

The report writer, `repo`, reads all dumps generated by the matrix generator and the files with the additional information `region.tn` and `region.rn` and a file controlling the amount of information created and the units in that the output should be generated. The standard control files for `repo` are in the directory with the cin-files as `srepin` for the short report and `lrepin` for the long report. `repo` creates a file containing the report of the input data (`region.srp` or `region.lrp` in the result directory) and the table of contents (`region.sct` or `region.lct` in directory `res`).

3.6 Files for the calculation program — CAP

The calculation program, `cap`, reads all dumps generated by the matrix generator, the sorted solution and additional files describing this solution (see section 3.4) and the file containing additional time series, `region.fil`. It operates according to the calculation instructions in `xx.cin`, where `xx` is either defined in the command line or taken from `region.use` (see table 3.6). Control of time steps to report on, title information, formats etc. are defined in `xx.tit` that is also defined in `region.use` or in the command line.

The file `region.use` as shown in table 3.6 contains an optional number of lines with two entries: the cin-file and the corresponding tit-file. The path to the cin-file, tit-file and `region.fil` are defined in `region.dir` (see table 3.5), entry `cin` for the cin-file and `gen` for the tit-file and for `region.fil`.

Table 3.6: Example of `region.use`

`cinfile1`	`titfile1`
`cinfile2`	`titfile2`

For each cin-file `cap` creates three files:

— `cin_scen.tab`, the tabular output,

— `cin_scen.grf`, the graphics information and

— `cin_scen.grc`, the titles for the graphics,

where `cin` stands for the name of the cin-file. All these output files are located in subdirectory `res` of the region.

3.7 Files for the solution analyzer — ANA

The solution analyzer, `ana`, reads some dumps generated by the matrix generator, the sorted solution and additional files describing this solution (see section 3.4) and the files with the additional information `region.tn`. It creates a file called `region_scen.nbf` in the subdirectory `res` of the region or application.

Part II

Input Data for MESSAGE III

Chapter 4

General Remarks

This part of the User's Guide of MESSAGE III describes the input files required to set up a model with MESSAGE III and generate an MPS-file (to be solved with a commercial package) or a report of the input data.
An application of MESSAGE III set up with the following files:

- An Application Data Bank (ADB) defining the problem
- A file containing general definitions (GEN, optional)
- A Technology Data Bank (TDB) containing general technology characteristics
- A Local Data Bank (LDB) for scenario parameters
- A file with title information and control for the Post-Processor CAP (TIT)
- A file with calculating instructions defining the tabular output for the Post-Processor CAP (CIN)
- A file with the definition of the directories used (DIR)
- Update files containng changes that overwrite all other definitions

The region has to be defined in file `glob.reg`. Generally the settings have to be provided in `mms.pro`, which is identified by environment variable `MMS_PRO` (see section 3.1).

Chapter 5

Organization of the Data Bank

The data bank of MESSAGE III contains the data for one specific application. The file structure of the data bank is organized hierarchically in three levels:

1. The technology data bank (TDB) consisting of the general technology descriptions like, e.g., costs, efficiencies, etc., and the definition of the energy carriers related to these technologies. The TDB is designed to be used for all applications based on the same technical assumptions.

2. The application data bank (ADB) comprises the general settings for a specific application or country (e.g., locally different costs for solar energy because of lower or higher solar influx). There is one ADB defined for each application.

3. The local data bank (LDB) comprising all changes done to the data locally and the scenario definitions. One LDB for each application and each scenario has to be available.

4. Changes to the definitions and data in the LDB can be included using update-files, e.g. in cases where the same changes apply to several regions.

5. General settings, like identifiers of levels of energy carriers or format used in the MPS-file, can be changed in an optional file labelled GEN-file.

The LDB sets the standards of what is included in a model: the lists of items are loaded from the LDB for the following components:

- energy forms and levels,
- additional relations,
- technologies and
- resources and grades of the resources.

When these lists are loaded, the data are included starting from the TDB, then the ADB and last the LDB; in addition the changes given in the update-file are included. Consequently in a full application the LDB will include a list of all energy forms, relations and technologies from the ADB. When a user eliminates one element, it will be taken out of the LDB and consequently not appear in the model any longer.

An additional option to change the data is by using the update file (UPD). This file is organized in a similar way as the LDB, but it is not required to contain all items. Only the items existing in the UPD-file will be changed, the others will remain unchanged.

All files of the data bank are organized by keywords, that indicate the type of information to come. Generally the files have a predefined structure with required and optional keywords. The input format is free, but the definition and data belonging to a keyword have to come in the same line, continuation is possible with a backslash.

5.1 Organization of the TDB

The technology data bank contains the keywords given in table 5.1 as section headings (column REQ indicates if the keyword is required (R) or optional (O)):

Table 5.1: Keywords organizing the TDB

Keyword	REQ	Description
`TDB:`	R	identification of the data bank: see also section 5.7 on page 27
`description:`	O	description of the data bank, is given in the following lines up to the next keyword
`fuelidentification:`	R	list of all fuel identifiers used in the technology data bank. These fuel identifiers are used in the definition of energy inputs and outputs (instead of the fuelid/levelid given in the ADB and LDB, see also section 10.1. The same identifications are given in the definitions of the fuels for the ADB and (if necessary) the LDB, see also section 7.2 on page 31.
`systems:`	R	definition of conversion technologies; see also section 10.1 on page 40
`endata`	R	end of input

Required data follow in the sequence given in the description of the respective keyword, optional data are identified by keywords, the input follows in the same input line (continuation is possible with a backslash at the very end of the line to be continued).

5.2 Organization of ADB and LDB

The application and local data bank have the same general structure. They contain the keywords given in table 5.2 as section headings:

Required data follow in the sequence given in the description of the respective keyword, optional data are identified by keywords, the input follows in the same input line (continuation is possible with a backslash at the very end of the line to be continued).

Table 5.2: Keywords organizing the ADB and LDB

Keyword	REQ	Description
`TDB:`	R	Identification of TDB; see section 5.7 on page 27
`ADB:`	O	Identification of ADB; see section 5.8 on page 27
`problem:`	R	Problem name; see section 6.1 on page 28
`description:`	O	description of the data bank; see section 6.2 on page 28
`drate:/vdrate:`	R	Discount rate; see section 6.3 on page 28
`timesteps:`	R	Definition of periods; see section 6.4 on page 28
`loadregions:`	O	Definition of load regions; see section 6.5 on page 29
`energyforms:`	R	Definition of energy chain; see section 7.2 on page 31
`demand:`	R	Demand data; see section 8.1 on page 35
`loadcurve:`	R	Load distribution of demands; see section 8.2 on page 35
`relations1:`	O	Additional relations type 1; see section 9.1 on page 37
`relations2:`	O	Additional relations type 2; see section 9.1 on page 37
`systems:`	R	Conversion technologies; see section 10.1 on page 40
`resources:`	O	Resource data; see section 11.1 on page 54
`imports:`	O	Imports of energy; see section 12.1 on page 59
`exports:`	O	exports of energy; see section 12.1 on page 59
`endata`	R	End of input

5.3 Organization of the Update-Files

An update file can contain the same entries as an LDB. However, it is not required to contain all mandatory keywords and all technologies/energy forms etc. to be included in the model (as the LDB is). An update-file has to contain the keywords relevant for the changes to be done, the item to be changed and the new data lines.

5.4 Counters and Labels of Years

Table 5.3 shows the counters and labels used in the description of the data formats (variables of values in square brackets give dimensions of arrays):

Table 5.3: Labels and counters used in the format descriptions

ID	Description
BYR	Base year (defined in section timesteps:)
LP[NT]	Length of period for all NT periods (defined in section timesteps:)
NLR	Number of load regions (defined in section loadregions:)
NT	Number of time steps (also referred to as number of periods, defined in section timesteps:)
YR[NT+1]	First year for each of the NT periods and end of last period (defined in section timesteps:)
TLR	Number of load regions for a technology (1 if the technology has no load regions, else it is equal to NLR).

5.5 Abbreviations used in the Formats

Table 5.4 shows the abbreviations used in the description of the formats in this documentation:

Table 5.4: Abbreviations used in the formats

Abbrev.	Description
f	real number
*f	optional number of real numbers
n	integer number
*n	optional number of integer numbers
ts	time series
a	character
a*	character string of optional length
xa	character string of maximal length x
[x]	x is optional
*(x)	x may be repeated

5.6 Format of Time Series

All data that can be given as time series for the `NT` time steps in the model are stored in the following format:

```
isw values
```

where `isw` is a switch defining the number of values to be read and their meaning. Table 5.5 gives the options availabale for `isw`. Data read as time series are denoted by a "ts" in the format descriptions. For time series defining limits, single values can be omitted by typing `free` or `1.e7` instead of a number.

Table 5.5: Input switches for time series

`isw`	no of values	description
-2	no input	no constraint generated
-1	no input	defaults are used, constraints are free (nonbinding)
0	one value	the single value is used as constant for all `NT` time steps
1	`NT` values	one value is given for each of the `NT` time steps
2	2 values	a first value is given for the first period, the second value is the annual growth for the whole time horizon
3	`NT` values	a first value is given for the first period, the following values are the annual growth for the periods

5.7 Identification of TDB: keyword TDB:

Under keyword TDB: the technology data bank to be used with the current ADB or LDB is identified in order to exclude errors in file handling.

Format: TDB: 8a

5.8 Identification of ADB: keyword ADB:

Under keyword ADB: the application data bank to be used with the current LDB is identified in order to exclude errors in file handling.

Format:
ADB: 8a
LDB: 8a

Chapter 6

General Definitions

The keywords described in this chapter give the general definitions of a problem: The name and comments, discount rate, time periods and load regions (if applicable).

6.1 Problem Identification: keyword `problem:`

Definition of the 8-character problem name used for identification in the optimization package that will be printed in the report of the problem starting at the first page.

Format: `problem:` 8a

6.2 Comment lines: keyword `description:`

Optional number of comment lines describing the problem.

Format: `#` 65a

6.3 Discount Rate: keyword `drate:` or `vdrate:`

The discount rate determines the relative importance of future events. It can be constant or variable over time.

Format: `drate:` n

Format: `vdrate:` ts

Checks: Has to be a nonnegative number

6.4 Time Horizon: keyword `timesteps:`

Definition of base year (`BYR`) and model years (first year per period plus end of last period; `YR[i]`) in one line.

Format: `timesteps:` (NT+2)*n

Checks: Have to be positive; `YR[i]> YR[i-1]>`

Derived Variables: The number of time steps `NT` is the number of entries minus 2. The period length (`LP[i]`) is the number of years between two model years. The length

of periods before the first model year (LPO): the number of years between BYR and YR[1].

Example:
The timestep definition

```
timesteps: 1990 1995 2000 2005 2010 2020 2030 2040 2050 2060
```

gives 1990 as base year for that existing capacities etc. have to be defined, 1995 is the first year in the model. The model has 8 periods, the first starts in 1995 and ends before 2000, the last one starts in 2050 and ends before 2060. Period lengths are three times 5 years and five times 10 years, LPO ist 5 years.

6.5 Load Regions: keyword loadregions:

The data for a load curve are organized in a hierarchical way starting from the full year. The year is divided into typical parts (e.g., seasons), each of which is again subdivided (e.g. into days, see also section 32.3). The data necessary for each of these groupings are identified by the keyword ltype. additionally the lenght of the load regions is given by the keyword length. This is the last keyword of the section for load regions. Table 6.1 gives an overview of the keywords available to define load regions. They are mandatory if loadregions are defined.

Table 6.1: Keywords for defining load regions

Keyword	REQ	Description
ltype	Y	Part of the year for storage (e.g., season, day)
length	Y	Length of the load regions

Keyword ltype

For each typical part of the year an identification, the number of occurences per hierarchically next part and the number of load regions it is subdivided into is given (in this order in the line of ltype).

Format: ltype *(8a n n)

Checks: The number of load regions and the number of occurences have to be positive numbers. The numbers giving the occurences have to grow for consecutive keywords ltype.

Derived Variables: The overall number of load regions (NLR) is the product of the number of load regions over all parts of the year, i.e. the product of the last entries of keyword ltype. It has to be less or equal 10 (in the present installation). If there is no keyword loadregions NLR is 1 and LR[1] is 1..

Keyword length

The basic data for defining a load curve are the lengths of the NLR load regions (LR[i]).

Format: `length` NLR*n

Checks: Have to be bigger than 0. and less or equal 1. The sum of the `LR[i]` has to be 1. Have to be defined if `NLR>1`

Example for a semi-ordered load curve

```
loadregions:
    ltype season 1 4 week 13 2
    length .179 .071 .179 .071 .179 .071 .179 .071
```

defines the year to consist of 4 seasons, each of which has 13 weeks. Each week is subdivided into 2 parts (workdays and saturday/sunday). The lengths of the load regions correspond to this setup. The demand load curves have to be given for these load regions, corresponding to winter workday, winter weekend, spring workday etc.

Example for an ordered load curve

```
loadregions:
    ltype ordered 1 5
    length .018 .009 .304 .334 .334
```

defines the load curve of the year to be sorted and grouped into 5 load regions, one for electricity peak, one for district heat peak, then rest of strong demand, medium demand and low demand.

Chapter 7

Energy Forms

Energy forms are defined in a different way in the TDB than in the ADB and LDB: In the TDB only the type of energy carrier is indicated by a character string, while in the other two data banks level and fuel identifier are defined exactly. The corresponding energy carrier(s) from the TDB are associated to these fuel identifiers.

7.1 Energy Forms in the TDB: keyword `fuelidentification:`

The energy forms in the TDB are defined by a character string, that is used to identify the potential inputs and outputs of technologies. They are linked to the identification in the actual model in the section with keyword `energyforms:` in the ADB and LDB. One entry is read per line, the input ends with an asterisk ("*").

Format: 20a

7.2 Energy Forms in the ADB and LDB: keyword `energyforms:`

Levels of Energy Forms

The energy forms are defined for various levels of energy (e.g., useful, final, secondary, primary energy). The definition of energy forms is done for the consecutive levels, where the level is identified by

```
levelname lid [fx]
```

where `levelname` is a unique identification of the level with up to twenty characters; `lid` is a unique one-character identification of the level used in the data bank to identify the levels. A "fx" after `lid` means that the conditions for the energy flow on this level are defined as equalities (see section 28).

The first level has to be the demand level. If the demand level has `lid`="U", special modeling of end-use is applied (see sec 25). "U" may only be used as demand (i.e. first) level identifier. The level identification "D" is preserved for specific uses and not allowed in this version. The level identification "Q" means that the fuels on this level are

cumulated over the years. It has to be the last level identifier. For these fuels the stock existing in the base year has to be given and the remainder of a period will be forwarded into the next year (see section 29).

Format: 20a a [2a]

Checks: Each character can only be used once as level identifier. The level identifier must not be "D". If "U" is used as level identifier, it has to be the first one. "Q" has to be the identifier of the last level, which has stock-piles.

Energy Forms

After the definition of a level an optional number of energy forms can be defined as follows:

```
fuelname fid [fatt] [stock] [ = TDB_name [TDB_name]... ]
```

where `fuelname` is an identification of the fuel, that is up to twenty characters long, `fid` is a one-character id of the fuel. Both have to be unique on this level. `fatt` is used to set some attributes to the fuel. It can be one of the following characters:

`l` - the fuel is modelled with load regions
`n` - the fuel is a nuclear fuel

Default is blank. `fatt` cannot be used for fuels on a level with `lid` "U" or "Q". A nuclear fuel is a fuel where an initial core can be used at installation time and a final quantity becomes available is possible (see section 28.1.5).

For fuels on level "Q" `stock` can be given (default is zero, it has to be nonnegative).

After these identifications the name(s) related to the fuel in the TDB (up to 20 characters) are given. These names are used to translate and check fuel names when retrieving technologies from the TDB.

Format: 20a a [a] [f] [= 20a [20a] ...]

Checks: On each level each character can only be used once as fuel identifier. On level "U" no attribute (load regions or nuclear fuel) are allowed. Stocks are only allowed on level "Q".

Keyword

After the identification of a fuel a free number of comment lines (each of them starting with #) can follow. Alternatively the next fuel of the same level or the end of fuel definition on this level comes.

Format: `#` 65a

Keyword *

The end of fuel definition on a level is indicated by an asterisk (*), next is a new `levelname` or the next section heading.

Example for an energy form definition in the TDB

```
fuelidentification:
    space_hat
    sp_elec
    electricity
    d_heat
    fuel_oil
    light_oil
    nat_gas
    crude_oil
    uranium
    plutonium
*
```

defines 7 energy carriers to be used in the TDB.

Example for an energy form definition in the ADB or LDB

```
energyforms:
useful u
    space_hat h
# Useful space heat on demand level
    sp_elec e
# Final demand for non-substitutable(specific) electricity
*
final f
    electricity e l
    d_heat d l
    liq_fuel l = fuel_oil light_oil
# liquid fuels include all refinery products
    nat_gas g
*
secondary x
    electricity e l
    d_heat d l
    liq_fuel l = fuel_oil light_oil
# liquid fuels include all refinery products
    nat_gas g
*
resource r
    nat_gas g
    crude_oil o
    uranium u n
*
stocks q
    plutonium p n 10
# in base year 10 ktons on plutonium are available
*
```

distributes the principal energy carriers from the TDB to the levels and gives them the required attributes for the model, like which energy carriers are modelled with load regions, which are nuclear fuels and which can be accumulated over time. Light and heavy oil products are combined into one energy carrier liquid fuels.

Chapter 8

Demand Data

The data defining the demands are constituted by the actual projection of the annual demand per period and the distribution of the demand to the load regions (see sections 28.1.1 and 28.1.2 for the formulation and 21.1.3 for postprocessing).

8.1 Demands: keyword `demand:`

The demand is given as time series for each of the energy forms defined on the demand level (i.e. the first level defined). One line per energy form with demand is read, fuels are identified by the 1-character fuel identifer (`fid`).

Format: a ts

Checks: The fuel identifier has to be defined on the demand level. Each demand identifer can only occur once. The values of the time series have to be nonnegative.

8.2 Load Curve: keyword `loadcurve:`

For each of the fuels on the demand level a distribution of the annual demand to the `NLR` load regions can be given (only if `NLR` > 1) in one line. The default is equal distribution.

Format: a NLR*f

Checks: The fuel identifier has to be defined on the demand level, it has to be defined with load regions and can only occur once. The `NLR` values read have to be nonnegative. They are normalized to 1 during input.

Example for a demand definition in the ADB or LDB

```
demand:
    h 2 100 .995
    e 2  10 1.02
loadcurve:
    h .001 .080 .504 .311 .105
    e .030 .008 .352 .332 .277
```

gives a demand of 100 for heat, declining at 0.5% per year and a demand for electricity

of 10 that grows at 2% per annum. The load curve is defined by the shares of the energy per load region for the 5 load regions of the ordered definition example.

Chapter 9

User-defined Relations

For user-defined relations (see section 30 for the formulation and 21.1.2 for postprocessing) limits, related costs and a description for the report of the input data can be given.

9.1 User-defined Relations: keyword `relations[1/2]:`

There are two types of user-defined relations denoted by the keywords `relations1:` and `relations2:`.

`relname rid rio rlr`

Each user-defined relation is identified by a character string (`relname`) which is read together with the four-character identification (`rid`), a character indicating if the technology coefficients in the relation are related to input (i) or output (o) of the technologies (`rio`)[1] and the (optional) number of the load region or a "+" to indicate relations between periods (`rlr`).

Format: *a 4a a a

Checks: The four-character identification may only occur once (or once for each load region) for type 1 and once for type 2.

After the identification of the relations the keywords shown in table 9.1 are allowed:

Table 9.1: Keywords for user-defined relations

Keyword	REQ	Description
`limit`	O	limit of the relation
`cost`	O	costs related to the relation
`#`	O	comments to the relation

[1]The coefficients are in any case converted during intput to represent the value per unit of main input of the technology.

Keyword limit

`tlim alim`

The type of limit (`tlim:` up/lo/fx) and the values for the limit (`alim:` as time series) are read.

Format: `limit 2a ts *a`

Keyword cost

`acst`

The costs related to the relation are read as time series.

Format: `cost ts`

Keyword

After each relation an optional number of comment lines (each starting with a "#") can follow.

Format: `# 56a`

Keyword *

An asterisk "*" ends the input of this relation, either the next relation or the next keyword comes.

Example for user-defined relations in the ADB or LDB

```
relations1:
base_load_limit base o 1
    limit up 0 0.
# Constraint on maximum share of base load power plants in
# electricity generation in load region 1.
*
investment_limit invl o
    limit up 1 100 100 100 free free free free free
# Investments in power generation are limited in first 3 periods,
# thereafter no limit.
*
relations2:
CO2_emissions CO2 i
    cost 0 15.
# Emissions of carbon dioxide, defined per unit of input.
# Emission taxes are defined as cost.
*
```

defines two relations of type 1 and one of type 2 with limits or costs associated with the relation. CO2 emissions are defined per unit of fuel used, the base-load limit is given per unit of output, while for the investments, which are only applied to capacity variables, it does not matter to what the relation corresponds.

Chapter 10

Conversion Technologies

Conversion technologies are defined by a multitude of keywords, that give all technical and economic performance data, limits and information on the existing structure. The formulation of conversion technologies is desctibed in section 25, postprocessing information in section 21.1.1.

10.1 Conversion Technologies: keyword `systems:`

The description of a conversion technology starts with the `identification` of the technology, which is given without keyword and followed by the data of the technology, which are identified by the keywords given in table 10.1

The identification of the technology is a string of up to 14 characters, which is used to identify the technology. It has to be unique for each technology. The end of the technology is identified by an asterisk (*) as keyword. All technologies are stored after the keyword **systems** and before the keyword **resources** (or the following one(s) if no resources occur in the data bank).

Table 10.2 gives an overview of the formats of the keywords of the data of the conversion technologies.

Identification of energy forms

Identification of energy forms depends on the data bank the technology data are loaded from:

In the TDB the energy carriers are identified by a 20-character string that has to be unique and is also used to establish a connection to the energy carriers defined in section 7.2.

In the ADB and LDB the energy carriers are identified by the ID of the energy carrier and the ID of the level it is defined on:

```
lid-fid [ts]
```

In the ADB and LDB the identification of the energy carrier can be given without changing the data given in the TDB; then no time series is given after the identification.

Table 10.1: Keywords for conversion technologies

keyword	REQ	Description
activity	R	numbering of additional activities
adda	O	number of additional activities
bda	O	bounds on activity
bdc	O	bounds on capacity
cload	O	change of load region distribution
con1a	O	user-defined constraints type 1, activity
con1c	O	user-defined constraints type 1, construction
con2a	O	user-defined constraints type 2, activity
con2c	O	user-defined constraints type 2, construction
corin	O	definition of initial cores (nuclear technologies)
corout	O	definition of final cores (nuclear technologies)
discon	O	definition of distribution of investments over the construction time
fom	O	fix O+M costs
fyear	O	first year of availability
inv	O	investment costs
inp	O	definition of additional inputs
hisc	O	historic capacity
lag	O	lag time in years between input and output
lyear	O	last year of construction
minp	O	definition of main input
cmix	O	unit size for mixed integer applications
mobj	O	identification of objective
moutp	R	definition of main output
mpa	O	market penetration for activity
mpc	O	market penetration for capacity
nonl	O	nonlinearities
outp	O	definition of additional outputs
plf	O	plant factor
pll	O	plant life
prel	R	power relation for alternative activities
tecsw	O	switch for type of equations or identification of objective
vom	O	variable O+M costs
#	O	comment lines
*	R	end of this technology

Table 10.2: Format and occurence of keywords for conversion technologies

keyword	REP	MaxNo	Ex	LR	TS	Format
activity	N	1	A	N	N	n ACTIVITY [a]
adda	N	1	M	N	N	n
bda	Y	2	B	Y	Y	TLR *(aa ts)
bdc	Y	2	M	N	Y	aa ts
cload	N	1	B	N	Y	n [NLR *ts]
con1a	Y	free	B	N	Y	aaaa[:a] ts
con1c	Y	free	M	N	Y	aaaa[:a] ts
con2a	Y	free	B	N	Y	aaaa[:a] ts
con2c	Y	free	M	N	Y	aaaa[:a] ts
corin	Y	free	M	N	Y	a-a ts
corout	Y	free	M	N	Y	a-a ts
discon	N	1	M	N	N	n a *f
fom	N	1	M	N	Y	ts
fyear	N	1	B	N	N	n
inv	N	1	Y	N	Y	f
inp	Y	1	B	N	Y	a-a [ts]
hisc	N	free	B	N	N	f [tsh]
lag	N	1	B	N	N	n
lyear	N	1	B	N	N	n
minp	N	free	B	N	N	a-a [f]
cmix	N	1	M	N	N	f
mobj	N	1	B	N	N	aaaa
moutp	N	1	B	N	Y	a-a [ts]
mpa	Y	2	B	N	Y	aa ts ts
mpc	Y	2	M	N	Y	aa ts ts
nonl	N	1	B	N	N	n
outp	Y	free	B	N	Y	a-a [ts]
plf	N	1	M	N	Y	ts
pll	N	1	M	N	Y	n
prel	N	1	A	N	N	f
tecsw	N	1	M	N	N	aa
vom	N	1	B	N	Y	ts
#	Y	1	B	N	N	65a

The data of a technology can have the following attributes:

REP: keyword and data can be REPeated (Y) or not (N)
MaxNo: maximum number of repetitions of keyword and data possible
Ex: keyword and data exist only for main technology (M), only for additional technologies (A) or for both (B)
LR: if data are defined they have to exist for all load regions
TS: data contain time series or not
Format: short description of format of data expected after keyword

Keyword activity

Identifies the beginning of an additional technology and gives the identifier of the technology (except in the TDB).

Format: n.ACTIVITY [a]

Checks: After a technology with adda $\neq 0$ adda technologies with activity have to come. n is the counter, it has to be less or equal adda+1 and come in ascending order.

Keyword adda

Gives the number of additional technologies linked to the present one. Default: zero.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: adda n

Checks: See chapter 10.1.

Keyword bda

Bounds on activity of a technology: type of bound and values as time series.

Format: bda TLR *(aa [ts])

Checks: The type of bound has to be one of the following: "up", "lo", "fx", "fr". If it is "fr" no time series is read. Bound type "fr" means that the variable can also take negative values and does not have any limits. For the same variable only one bound can be given (with the exception of "up" and "lo", which can be used together). Upper bounds have to be greater or equal the corresponding lower bounds for each time step and load region.

Example: bounds for technology with load regions

```
bda up 0 18 0 9 0 free 0 free 0 free
```

Constant upper bounds in load regions 1 and 2; free variables in the other load regions.

Keyword bdc

Bounds on capacity of a technology: type of bound and values as time series.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: bdc aa [ts]

Checks: The type of bound has to be one of the following: "up", "lo", "fx", "fr", "ui", "bv". If it is "fr" "bv" no time series is read. Bound type "fr" means that the variable can also take negative values and does not have any limits.

Bound type "bv" indicates that the variable is binary (possible values 0 or 1), bound type "ui" defines integer variables, that can take any integer value between 0 and the implicit definition given by the bounds. The bounds are always in units of Power (e.g., MW), the unit size will be used to generate the correct numerical value for the bound. If the investment variables are defined as binary or integer, this unit size of an installation

has to be defined with keyword `cmix`. In addition, mixed integer programming has to be switched on with keyword `mixsw:` in the Gen-file (see section 13).

For the same variable only one bound can be given (with the exception of "up" and "lo", which can be used together). Upper bounds have to be greater or equal the corresponding lower bounds for each time step.

Keyword `cload`

Change load region pattern (and fix it). Switch 2 means base load distribution (no further input), switch 1: one time series per load region. If no `cload` keyword is given, the production pattern of the technology is free.

Only if `TLR = NLR`

Format: `cload n [ TLR *ts ]`

Checks: Each value in the time series has to be nonnegative, the sum of the values over the load regions has to be 1 for each time step.

Example:

```
cload 1 0 .018 0 .009 0 .304 0 .334 0 .334
```

Fixed load region pattern with constant value for each load region.

Keyword `con1a`

Coefficients for activity of technology in user defined relations of type 1: Name of relation followed by the optional :a designing the number of the load region (only if `TLR=NLR`) and a time series for the values.

Format: `con1a` aaaa[:a] ts

Checks: The name of the relation has to be defined in `relations1`. The definition here can either have load regions or not. If the relation is defined with load regions the load region given here has to be defined. Each load region can only be defined once.

Example:

```
con1a base:1 0 1.
```

Constant coefficient 1 in user-defined relation type 1 `base` in load region 1.

Keyword `con1c`

Coefficients for capacity of technology in user defined relations of type 1. Name of relation followed by the optional :a designing the number of the load region (only if `TLR=NLR`) or a "t" (total capacity) and a time series for the values.

Only if the plant life of the technology (`pll`) is positive, i.e. the technology has explicit investment variables.

Format: `con1c` aaaa[:a] ts

Checks: The name of the relation has to be defined in `relations1`. The definition here can either have load regions, a "t" or nothing. If the relation is defined with load regions the load region given here has to be defined. Each load region can only be defined once.

Example:

```
con1c capa:t 0 1.
```

The total installed capacity (instead of the new installations) including historic capacities in the RHS will be inserted in the relation with coefficient 1.

Keyword con2a

Coefficients for activity of technology in user defined relations of type 2: Identical to `con1a`.

Keyword con2c

Coefficients for capacity of technology in user defined relations of type 2. Identical to `con1c`.

Keyword corin

Initial core of nuclear fuel if a new plant goes into operation. Definition of an energy form (first character) and the level it is defined on (third character), followed by the initial core.

Only if the plant life of the technology (`pll`) is positive, i.e. the technology has explicit investment variables.

Format: `corin` a-a ts

Checks: The energy form has to be defined on the given level and be either nuclear or man-made. Each combination of fuel and energy form can only be defined once as `corin`. The core has to be nonnegative.

Keyword copy

Copies all information from a previously defined technology (or alternative operation) to the present one. Should come as first keyword in the technology definition to avoid overwriting of previously given data in this technology definition. Can only be used in LDB or update-files.

Format: `copy *a`

Checks: The identification given has to correspond to one of the technologies already defined. If keyword `copy` is given in the definition of an alternative technology, it can give the number of the alternative technology (e.g. `copy 2` if the data from `2. activity` should be copied.

Example:

```
copy coal_ppl
```

Repeats all definitions of `coal_ppl` in the current technology definition.

Keyword corout

Last core of nuclear fuel if a plant goes out of operation. Definition of an energy form and the level it is defined on, followed by the last core.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: corout a-a ts

Checks: The energy form has to be defined on the given level and be either nuclear or man-made. Each combination of fuel and energy form can only be defined once for keyword corout. The core has to be nonnegative.

Keyword discon

Distribution of investment costs over construction time: Number of years costs are distributed to, switch for kind of function used and parameters. If keyword discon is omitted, the investments are assumed to be payed at the beginning of operation.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

The following functions are implemented:

e equal distribution (no additional input)
p polynomial of the form a + bx + cxx (input: a,b,c)
d free definition (input one value per year, sum has to be 1.)
l logistic function (input eps and delta)

Format: discon n a *f

Checks: The value has to be positive.

Keyword fom

Fix operation and maintainance costs.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: fom ts

Keyword fyear

First year in that the technology was built or can be built in future.

Format: fyear n

Checks: Has to be a positive integer.

Identification

Identification of a technology that is used to select data from data banks. The identification is not identified by a keyword, but by context; the following one-character identification (ID) is used to compose the internal name of the technology, the optional

"fx" thereafter indicates, that the capacity constraints of the technology should be fixed (i.e. the existing capacity has to be used, see section 25.2.1). The ID and "fx" are only applicable in the ADB and LDB.

```
identification tid fx
```

Format: 14a [a [fx]]

Checks: The long identification must not contain a blank and has to be unique in each of the data banks. The short ID has to be an alphanumeric character. It has to exist in the ADB and must not be given in the TDB. The internal name has to be unique for all technologies. If it is not the ID is changed during data input and the new ID reported in the error log of the input routine.

"fx" is only applicable if the technology has a positive plant life. If it is applied to technologies with load regions it will result in full utilization in each load region, i.e. baseload operation.

The internal name of the technology is composed in the following way: (this is also true for the additional technologies, where `tid` is defined by the keyword `ACTIVITY`).

```
Character 1: Identification of level of moutp
Character 2: Identification minp;  if none exists: "."
Character 3: tid of technology
Character 4: Identification moutp
```

Example:

```
coal_ppl c
```

Defines a coal-fired power plant with identification `coal_ppl` and technology identifier `c`.

Keyword `inv`

Investment (capital) cost per unit of main output. A time series switch 0 and a negative value mean that the technology is modelled without capacity variables. If the life of the installation extends beyond the end of the planning horizon, the investment costs will be adapted by a scrap value, see section 32.8.

Format: `inv` ts

Only if the plant life of the technology (`pll`) is positive, i.e. the technology has explicit investment variables.

Keyword `inp`

Definition of additional inputs of the technology and amounts used. If one of the inputs or outputs has load regions the technology is defined to have load regions (`TLR = NLR`, the default for `TLR` is 1, see also section 10.1), unless the keyword `cload` occurs for this technology. If no amount is given the amount from the data bank read before is kept. Keyword `inp` can only be defined if a keyword `minp` is also given.

Format in TDB: `inp` 20a [ts]

Format in ADB and LDB: `inp` a-a [ts]

Checks: The energy form has to be defined on the given level. Each combination of fuel and energy form can only be defined once in the keywords `inp`, `minp`, `outp`, `moutp`. The values of the time series have to be positive.

Keyword `hisc`

Definition of the activity in the base year and–if `pll` is positive–the capacities that were built before the time horizon (`min(fyear-byr-1,pll)` values if switch `n` is 1, two values (total capacity and growth) if switch `n` is 2 and an optional number of pairs of years and values if switch `n` is 4.

Has to be defined if `fyear`$\leq$`byr`.

Format: `hisc` f [n *f]

Checks: All real numbers have to be nonnegative. If switch `n` is 2 the growth has to be positive. Switch `n` can only be 1, 2 or 4. If `fyear` $\leq$ `BYR` the historic activity and/or capacity have to exist.

Example: constant historic growth

```
hisc 100 2 150 1.02
```

Activity in the base year was 100, overall capacity in the base year was 150 and has been growing at 2% per year before.

Example: definition with year/value pairs

```
hisc 100 4 1980 50 1983 70 1985 30
```

Activity in the base year was 100, overall capacity in the base year was 150 and has been installed in 3 specific years.

Keyword `lag`

Lag time in years between inputs and outputs of the technology. If omitted, inputs and outputs refer to the same period.

Only if the technology has at least one input.

Format: `lag` n

Checks: Has to be nonnegative.

Keyword `lyear`

Last year in that the technology can be constructed (default is the end of the horizon).

Format: `lyear` n

Checks: Has to be nonnegative. If it is positive it has to be greater than `fyear` and greater or equal `yr[1]`.

Keyword minp

Definition of main input of the technology and amount used. If one of the fuels has load regions the technology is defined to have load regions (TLR = NLR, the default for TLR is 1, see also section 10.1). If no amount is given the amount from the data bank read before is kept. If no keyword minp is given, the technology does not consume any energy carriers (e.g., wind power plant).

Format in TDB: minp 20a [f]

Format in ADB and LDB: minp a-a [f]

Checks: The energy form has to be defined on the given level. Each combination of fuel and energy form can only be defined once in the keywords inp, minp, outp, moutp. The amount has to be positive.

Keyword cmix

Unit size of new installations for mixed integer applications (either per period or per year, depending on switch n). Integer or binary variables will only be generated if the general mixed integer programmin switch is set via keyword mixsw: in the Gen-file (see section 13). If the new installations of a technology are modeled as integer decisions, the construction bounds (keyword bdc) have to be given as ui or bv (for integer variables or binary variables).

Format: cmix n f

If switch n is eqal 1 the unit size cmix can be built once per period up to a number implicitly defined by the upper limit given as bound. Since this bound is given as annual value, in longer periods more units can be built.

If it is 2 the unit size cmix is built in each year of the period if the investment decision is taken, up to the maximum number implicitly defined by the upper limit given as bound.

Checks: The value has to be positive.

Example: switch 1

```
cmix 1 100
```

if the upper bound is 200 and the period lengths are 1, 1, 2 and 2, the unit size of 100 can be built up to 2 times in periods 1 and 2 and up to 4 times in periods 3 and 4 (i.e. the limits are internlly converted to limits per period). However, it is always possible to build only 1 unit.

Example: switch 2

```
cmix 2 100
```

if the upper bound is 200 and the period lengths are 1, 1, 2 and 2, the unit size of 100 can be built 0, 1 or 2 times in periods 1 and 2 and 0, 2 or 4 times in periods 3 and 4 (i.e. the limits are applied to the installation per period, which are automatically generated as one unit size per year). period).

Keyword mobj

Connects the technology to a specific objective function, that is identified by 4 characters.

Format: mobj aaaa

Keyword moutp

Definition of main output of the technology and the amount produced. If one of the inputs or outputs has load regions the technology is defined to have load regions (TLR = NLR, the default for TLR is 1, see also section 10.1). If no amount is given the amount from the data bank read before is kept.

Format in TDB: moutp 20a [ts]

Format in ADB and LDB: moutp a-a [ts]

Checks: The energy form has to be defined on the given level. Each combination of fuel and energy form can only be defined once in the keywords inp, minp, outp, moutp. The values of the time series have to be positive.

Example:

```
moutp e-x 0 .38
```

Energy carrier e on level x is generated at a constant rate of 0.38 relative to the main input.

Keyword mpa

Dynamic (market penetration) constraint on the activity of the technology. Upper or lower dynamic constraints are defined by time series for the initial value (first time series) and the growth (second time series, see section 25.2.3 and 25.2.2). If a technology has load regions, the constraints is applied to all load regions together.

Format: mpa aa ts ts

Checks: The 2-character identification has to be "up" or "lo". The values of the first time series have to be nonnegative, the second positive.

Example:

```
mpa up 0 0.5 0 1.05
```

The technology output can grow at constatn 5% per year, starting at 0.5.

Keyword mpc

Dynamic (market penetration) constraint on the annual new capacity of the technology. Upper or lower dynamic constraints defined by time series for the initial value (first time series) and the growth (second time series).

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: mpc aa ts ts

Checks: The 2-character identification has to be "up" or "lo". The values of the first time series have to be nonnegative, the second positive.

Keyword nonl

Indicates if nonlinearities are defined for the technology:

0: no nonlinearities for this technology (default),
1: nonlinearities concerning the capacity variables,
2: nonlinearities concerning the activity variables, and
3: nonlinearities concerning the capacity and activity variables.

The corresponding variables are positioned directly after the record with "COLUMNS" in the MPS-file.

Format: nonl n

Checks: Value has to lie between 0 and 3.

Keyword outp

Definition of additional outputs of the technology and the amounts produced. If one of the fuels has load regions the technology is defined to have time series (TLR = NLR, the default for TLR is 1, see also section 10.1). If no amount is given the amount from the data bank read before is kept.

Format in TDB: outp 20a [ts]

Format in ADB and LDB: outp a-a [ts]

Checks: The energy form has to be defined on the given level. Each combination of fuel and energy form can only be defined once in the keywords inp, minp, outp, moutp. The values of the time series have to be positive.

Keyword plf

Plant factor of technology. The meaning of the plant factor depends on the inputs and outputs of the technology and if its load region pattern is fixed. If none of the inputs and outputs of a technology has load regions, the technology does not have load regions and the plant factor is the maximum availability of the technology per year. If one of the in- or outputs has load regions, the technology has load regions and the plant factor is the maximum availability in each load region. If the technology has a fixed production pattern over the load region the plant factor is the maximum availability in the load region with the highest load utilization given by the pattern. The default value is 1.

Only if the plant life of the technology (pll) is positive, i.e. the technology has explicit investment variables.

Format: plf ts

Checks: The plant factor has to be positive and less or equal 1.

Keyword pll

Technical plant life of technology. If no plant life is given or the plant life is set to 0 the technology does not have capacity variables or any of the related characteristics.

Format: `pll` f

Checks: The plant life has to be positive or 0.

Keyword prel

Power relation for alternative technologies: if a technology can be used in two modes (e.g. passout-turbines used to generate only electricity or in co-generation mode) the technology description of the second generation mode is added using the keyword `adda 2` and `2. activity`. In this case the relative maximum production of the main energy output is defined by the keyword `prel`. If, in the above example, electricity generation is reduced by 20% if heat is decoupled in the co-generation mode, `prel` has to be set to 1.2 Only for alternative technologies (i.e. technologies starting with `n. activity`).

Format: `prel` f

Checks: The power relation has to be positive.

Keyword vom

Variable operation and maintainance costs.

Format: `vom` ts

Checks: The variable O+M costs have to be nonnegative.

Keyword

Comment lines for report on technology data.

Format: `#` 65a

Example of a full technology definition

```
gas_ppl c
    fyear 1970
    minp g-r 1
    moutp e-x 0 .4
    inv 0 700
    fom 0 20
    vom 0 30
    pll 0 30
    plf 0 .7
    con2a CO2 0 .482
    con1c inv 0 700
    mpa up 0 0 0 1.06
    hisc 53. 4 1970 30 1975 43 1980 21
    bdc up 1 15 60 free free free free free free
```

*

defines a gas-fired power plant with an efficiency of 40%, a maximum operation time of 70%, CO_2 emissions of .482 per unit of input, an entry in the investment counter, a maximum dynamic constraint on production, historic installations for 3 specific years and upper bounds on construction for the first two periods.

Chapter 11

Resource Data

Resource data are defined on the basis of defining several grades with different costs for the grades and a resource volume for each of the grades. The mathematical formulation is given in section 26.

11.1 Resource Data: keyword `resources:`

Resource data are defined after the (optional) keyword `resources`. Each resource can have various grades, for each of which some definitions can be done. For each resource the keywords given in table 11.1 are allowed.

Table 11.1: Keywords for resources

keyword	REQ	Description
`fuel`	R	definition of energy form and level
`grade`	R	identification of grade(s)
`mpe`	O	market penetration on extraction
`uplim`	O	upper limit on annual extraction
`#`	O	comment lines

For each grade of each resource the keywords given in table 11.2 are allowed:
Table 11.3 gives an overview of the formats of the keywords of the resource data.

Table 11.2: Keywords for grades of resources

keyword	REQ	Description
cost	O	cost per time step
cela	O	relative costs of extraction (per elasticity class)
byrex	O	extraction in base year
lela	O	relative bounds on activity of extraction (per elasticity class)
reflim	O	limit on activity of extraction (for elasticity definition)
resrem	O	maximum share of resource to be extracted
mpe	O	market penetration on extraction per grade
uplim	O	upper limit on annual extraction per grade
volume	R	volume of resource available per grade
*	R	end of grade

Table 11.3: Format and occurence of keywords for resources

keyword	REQ	REP	MaxNo	Ex	EC	TS	Format
cela	O	Y	free	G	Y	Y	n ts
cost	O	N	1	G	N	Y	ts
byrex	O	N	1	G	N	N	f
fuel	R	N	1	R	N	N	a-a-
grade	R	Y	free	G	N	N	a
lela	O	Y	free	G	Y	Y	n ts
reflim	O	N	1	G	N	Y	ts
resrem	O	N	1	G	N	Y	ts
mpe	O	Y	2	B	N	Y	aa ts ts
uplim	O	N	1	B	N	Y	ts
volume	R	N	1	G	N	N	f
#	O	Y	1	R	N	N	65a
*	R	N	1	B	N	N	

The data of a resource can have the following attributes:

REQ: keyword and data are optional for the resource/grade (O) or required (R)
REP: keyword and data can be REPeated within one resource/grade (Y) or not (N)
MaxNo: maximum number of repetitions of keyword and data possible per resource/grade.
Ex: keyword and data exist for resources (R) or for grades (G) or both (B)
EC: keyword and data exist for elasticity classes (Y) or not (N)
TS: data contain time series or not
Format: short description of format of data expected after keyword

Keyword cela

Relative costs of the resource per grade for elasticity classes. This keyword comes after the keyword grade that defines the grade to which the relative costs refer. The number of elasticity class and the factors in relation to the costs of keyword costs are read.

Format: cela n ts

Checks: The data values have to be positive. The keyword cela is required for the same classes as the keyword lela (see section 11.1 on page 56).

Keyword cost

Costs of the resource per grade. This keyword comes after the keyword grade that defines the grade to which the resource costs refer.

Format: cost ts

Checks: The data values have to be nonnegative. The keyword costs is required if elasticity classes are defined. (In this case also the keyword reflim is required). The keyword costs has to come if more than 1 grade is defined.

Keyword byrex

Extraction per grade in the base year (default is 0). This keyword follows the keyword grade that defines the grade to which the extraction refers.

Format: byrex f

Checks: Has to be a nonnegative number.

Keyword fuel

Definition of the fuel (first id) and level (second id).

Format: fuel a-a-

Checks: The fuel has to be defined on the specified level. It must not have load regions. A fuel can be defined once as resource.

Keyword grade

Definition of grades (cost categories) of the fuel. If no grades are defined, the related data can only be defined once per fuel.

Format: grade a

Checks: The one-character identifier has to be unique for each resource.

Keyword lela

Relative limits of the resource per grade for elasticity classes. The number of elasticity class and the factors in relation to the limits of keyword reflim are read. This keyword comes after the keyword grade that defines the grade to which the limits refer.

Format: lela n ts

Checks: The data values have to be positive. The keyword `lela` has to be defined for the same classes as the keyword `cela` (see section 11.1 on page 56).

Keyword `reflim`

Basis of limitations for elasticity classes.

Format: `reflim` ts

Checks: The data values have to be nonnegative. The keyword `reflim` has to be defined if elasticity classes are definied, i.e. if the keywords `cela` (see section 11.1 on page 56) and `lela` (see section 11.1 on page 56) are defined.

Keyword `resrem`

Maximum share of a grade (still existing in a period) that can be extracted annualy in the period. This keyword comes after the keyword `grade` that defines the grade to which the share refers.

Format: `resrem` ts

Checks: The values have to be nonnegative numbers less or equal 1.

Keyword `mpe`

Market penetration on the extraction per grade or per resource. Upper and lower dynamic (market penetration) constraints are defined by a an initial value and the growth. This keyword comes after the keyword `grade` that defines the grade to which the market penetrations refer.

Format: `mpe` aa f f

Checks: The 2-character identification has to be "`up`" or "`lo`". The values of the first time series have to be nonnegative, the second positive.

Keyword `uplim`

Upper limits on annual extraction per grade (only if elasticities are defined) or resource (only if grades or elasticites are defined). This keyword comes after the keyword `grade` that defines the grade to which the limits refer.

Format: `uplim` ts

Checks: If an elasticity class is defined by the first word, also costs (keyword `cost`) for this class have to be given. If no elasticity class is defined, limits on grades are given by `bde`, if no elasticity class and no grade is defined, limits on resources are given by `bde` instead of `uplim`.

Keyword `volume`

Total resource volume per grade or for the resource (if no grades are defined). This keyword comes after the keyword `grade` that defines the grade to which the volume refer.

Format: `volume f`

Checks: Has to be a nonnegative number. Has to come for each grade.

Keyword

Comment lines for report on resource data.

Format: `#` 65a

Example for resource definitions:

```
fuel g-r
    grade a
       volume 100000
       cost   0 150
       byrex  250
       resrem 0 .9
       uplim 0 300
    grade b
       volume 150000
       cost 0 200
# resources of natural gas in two grades (cost categories)
*
```

Constant upper bounds in load regions 1 and 2; free variables in the other load regions.

Chapter 12

Imports and Exports

Imports and exports can be defined for each fuel with most keywords being identical to the resource definition keywords. The mathematical formulation is given in section 27.

12.1 Import/Export Data: keywords `imports`, `exports`:

Import and export data are defined after the (optional) keywords `imports` or `exports`, respectively. For each traded fuel (i.e. imported or exported) the keywords given in table 12.1 are allowed.

Table 12.1: Keywords for imports and exports

keyword	REQ	Description
`byrex`	O	amount in base year
`cela`	O	relative costs per elasticity class
`clr`	O	relative costs per load region
`cost`	O	cost per time step
`fuel`	R	definition of energy form and level
`lela`	O	relative bounds on activity per elasticity class
`llr`	O	limits on activity per load region (share or absolute)
`mpe`	O	market penetration on activity
`reflim`	O	limit on activity of extraction (for elasticity definition)
`uplim`	O	upper limit on import/export
`#`	O	comment lines

Table 12.2 gives an overview of the formats of the keywords of the import and export data.

Table 12.2: Format and occurence of keywords for import and export

keyword	REQ	REP	MaxNo	LR	EC	TS	Format
`cela`	O	Y	free	Y	Y	Y	n n ts
`clr`	O	Y	free	Y	N	Y	n ts
`cost`	O	N	1	N	N	Y	ts
`byrex`	O	N	1	N	N	N	f
`fuel`	R	N	1	N	N	N	a-a-
`lela`	O	Y	free	Y	Y	Y	n n ts
`llr`	O	Y	free	Y	N	Y	n a ts
`reflim`	O	N	1	Y	N	Y	ts
`mpe`	O	Y	2	N	N	Y	aa ts ts
`uplim`	O	N	1	N	N	Y	ts
`#`	O	Y	1	N	N	N	65a
`*`	R	N	1	N	N	N	

The data of imports and exports can have the following attributes:

REQ: keyword and data are optional for the import/export (O) or required (R)
REP: keyword and data can be REPeated within one import/export and country (Y) or not (N)
MaxNo: maximum number of repetitions of keyword and data possible per import/export and country
LR: keyword and data (can) exist for each load region (Y) or only once (N)
EC: keyword and data exist for elasticity classes (Y) or not (N)
TS: data contain time series or not
Format: short description of format of data expected after keyword

Keyword cela

Relative costs of the fuel for elasticity classes per load region. The number of elasticity class and the factors in relation to the costs of keyword costs are read.

Format: cela [n] n ts

Checks: The data values have to be positive. The keyword cela has to be defined for the same classes as the keyword lela (see section 12.1 on page 61). If the fuel has load region the load region identifier has to be the first entry.

Keyword clr

Relative costs of the fuel per load region. The number of load region and the factors in relation to the costs of keyword costs are read.

Format: clr [n] ts

Checks: The data values have to be positive. The keyword cost has to be defined if clr is defined. Can only come if the fuel has load regions.

Keyword cost

Costs of import/gain of export.

Format: cost ts

Checks: The data values have to be nonnegative. The keyword costs has to come if elasticity classes are defined. (In this case also the keyword reflim has to come). The keyword cost has to be defined if clr is defined.

Keyword byrex

Import/export in the base year (default is 0).

Format: byrex f

Checks: Has to be a nonnegative number.

Keyword fuel

Definition of the fuel (first id) and level (second id).

Format: fuel a-a-

Checks: The fuel has to be defined on the specified level. A fuel can only be defined once as import and once as export.

Keyword lela

Relative limits of the import/export per load region for elasticity classes. The number of load region, elasticity class and the factors in relation to the limits of keyword reflim are read.

Format: lela [n] n ts

Checks: The data values have to be positive. The keyword `lela` has to be defined for the same classes as the keyword `cela` (see section 12.1 on page 61).

Keyword `llr`

Relative limits of the fuel per load region. The number of load region, an indication if the limit is the share of the total limit and the values are read.

Format: `llr` n [a] ts

Checks: The data values have to be nonnegative. If the second entry is a % the keyword `uplim` has to be defined for the total amount. Then the values are interpreted as shares of the overall limit.

Keyword `reflim`

Basis of limitations for elasticity classes (for each load region). If the fuel has load regions the first entry is the load region identifier.

Format: `reflim` [n] ts

Checks: The data values have to be nonnegative. The keyword `reflim` has to be defined for the load region if elasticity classes are definied, i.e. if the keywords `cela` (see section 12.1 on page 61) and `lela` (see section 12.1 on page 61) are defined.

Keyword `mpe`

Market penetration on import/export. Upper and lower market penetrations are defined by time series for the initial value (first time series) and the growth (second time series).

Format: `mpe` aa ts ts

Checks: The identification has to be `up` or `lo`. The values of the first time series have to be nonnegative, the second positive.

Keyword `uplim`

Upper limits on annual imports/exports (only if elasticities are defined).

Format: `uplim` ts

Keyword

Comment lines for report on imports or exports.

Format: # 65a

Chapter 13

The General Definition File

The general definition file is an ascii file that contains general information on the model setup. The GEN-file may contain comment lines starting with a semi-colon. All keywords are optional and if no Gen-file is available for a region or other application, the defaults will be used.

Table 13.1: Keywords in the GEN-file

keyword	Format	Description
`ntrun:`	n	number of last period to be in matrix
`mixsw:`	n	mixed integer programming y/n
`mform:`	n [a*]	matrix format: 0 internal / 1 exogenous
`levnam:`	n n*a	number of energy levels and identifiers
`ncap:`	n	capital costs as lumpsum/levellized
`chkacc:`	n	check accounting rows
`imat:`	n	switch for distribution of investments

Keyword `ntrun:`

Number of last year in the matrix. It is possible to run the model for less periods than it is defined for (e.g. for calibrating the base year).

Format: `ntrun:` n

Checks: Has to be greater than `ntu` and less or equal number of periods in model

Default: Number of periods defined in ADB and LDB.

Keyword `mixsw:`

Enable mixed integer programming, i.e. generation of binary variables. If the switch is 0, no binary or integer variables are generated, even if they are defined in the technology section, setting it to 1 enables MIP.

Format: `mixsw:` n

Checks: Has to be 0 or 1

Keyword `mform`:

Format of data in the mps-file. If the switch is 1 a format is read in the same line (including the closing parenthesis, e.g. 'e12.5)', which gives maximum number of digits).

Format: `mform:` n [a*]

Default: e12.6)

Keyword `levnam`:

Identifiers of levels of energy carriers. Reads the number of levels and the respective identifications from resources to demands, last is man-made fuels for stock-pile (has to have id "q"). Useful energy has to have id "u", identification "d" is reserved.

Format: `levnam:` n na

Example:

`levnam: 6 r a x f u q`

Default: `r a x f u q`

Keyword `ncap`:

Capital costs are paid at once at the beginning of the period (switch 0) or levellized over the plant life (switch 1).

Format: `ncap:` n

Default: 0

Keyword `chkacc`:

Switch for checking of additional relations:

0 no checking

1 checking if all relations used are also declared

2 checks 1 and 3

3 checking if repetitions occur

Format: `chkacc:` n

Keyword `imat`:

Switch for generation of cost accounting rows on types of cost (i.e. investments, O+M, resources, imports, exports, costs related to accounting rows, see section 31.1.1).

0 no accounting rows
1 total accounting
2 annual accounting
3 both

Format: `imat:` n

Keyword `imat` can be followed by an optional number of lines giving limits on specific accounting rows, ending with a line with a single asterisk. On each line a switch identifies if the limit corresponds to the annual ("a") or the global accounting row ("t") or if it is to be ignored ("i"), the 4-character identifier of the accounting row and the value for total and a time series for annual cost limits.

Format: a1 a4 ts or a1 a4 f

Accounting rows available for limiting are:

ccur – fix (related to the installed capacity) and variable (related to the production) operation and maintenance costs,
ccap – investment costs; if the investments of a technology are distributed over the previous periods, also the entries to this accounting rows are distributed (if the capital costs are levellized, the total payments in a period can be taken from *cinv*; *ccap* shows the share of investments in the according period, then),
cres – domestic fuel costs,
car1 – costs related to the user defined relations of type 1 (see section 30),
car2 – costs related to the user defined relations of type 2 (see section 30),
cred – costs for reducing demands due to demand elasticities, only related to technologies supplying the demands directly,
cimp – import costs,
cexp – gains for export, and
cinv – total investments (in case of levellized investment costs, see *ccap*)

Example for a Gen-file

```
ntrun: 12          ;  [number of last period to be in matrix]
mixsw: 1           ;  [mixed integer programming initiated]
mform: 1  e12.6)   ;  [matrix is written in format as given below]
levnam: 8 p s N S 0 f u q ;   [energy level names]
chkacc: 2          ;  check for accounting rows
imat:   3          ;  generation of accounting rows
  ;  t: total cost, a: annual cost, i: ignore
    t ccap 600.    ;  limit on total investment volume
    a ccap 0 2000.;  limit on annual investments as time series
    i ccur 0 1000.;  limit on o+m costs ignored
    *          ;  end of bound on cost accounting rows
```

Chapter 14

Control Input for the Matrix Generator

The kind of output produced by the matrix generator (MXG) is controlled by two additonal input files. Two programs of MXG (ROWS and RES) require information on the type of output they have to produce–either generate a matrix or provide the report wiiter (REPO) with the needed information or both. See Table 14.1 for the switches needed (They are read in format I2).

Table 14.1: Output control for the matrix generator.

Code	Switch	Produced Files
ROWS	0	matrix and information for REPO
	1	only information for REPO
RES	0	matrix
	1	information for REPO
	2	matrix and information for REPO

The first line of control input for ROWS contains additionally the name of the LP solver used–this information has to start in column 4, can have up to 8 characters and is forwarded to MKSF (the sorter of the optimization output) as a switch for the format in that the solution has to be read.

If the control program `run_msg` is used the switches are set automatically to the appropriate values.

Chapter 15

Control Input for the Report Writer

The input data of a specific MESSAGE application can be converted into a printable report. Two versions are available, a short and a long report.

The report writer (REPO) reads a control file that contains integer switches in format I5. They control the amount of information produced, and the page control for printing (see Table 15.1 for a detailed description of this control file). It produces two output files, a report and a table of contents (see section 3.5 for an explanation of the files generated).

In the short version of the report, each technology is described by one line, which consists of:

- investment costs per unit of total output,
- fixed O+M costs per unit of total output,
- variable O+M costs per unit of total output,
- overall efficiency,
- plant life,
- plant factor, and
- average cost per unit of total output excluding fuel costs.

The formula used for calculating the average (also called levellized) costs is:

$$\frac{ccap \times \frac{dr\,(1+dr)^{\tau}}{(1+dr)^{\tau}-1} + cfix}{\pi} + ccur\ ,$$

where

$ccap$	is the investment per unit of total output,
$cfix$	is the fixed O+M cost per unit of total outpt,
$ccur$	is the variable O+M cost excluding fuel cost per unit of total output,
dr	is the discount rate,
τ	is the technical plant life, and
π	is the plant factor.

The long report contains a detailed desctiption of all characteristics of a technology, inlcuding a list of time-dependent parameters. It can be used as "Technical Report" for publications. After the first header and before the list of technologies additional information can be added, which will be included in the page control. All pages besides the definition of the energy chain have less than 80 characters per line.

Table 15.1: List and description of control input for the report writer

no	no/yes	description
1.	0/1	no/yes: general information on energy forms, load curve, demands, additional relations, etc.
2.	0/1	no/yes: information on prices and limits on resources, imports and exports
3.	0/1/2	no/short/extensive technology information
4.	0/n	>0: read info from UNIT n and write after the header DESCRIPTION OF TECHNOLOGIES (7 is a free UNIT and could be used)
5.	0/1	no/yes: cost tables for the technologies
6.	0/1	no/yes: short identifiers of energy forms, additional relations and technology
7.	0/1	no/yes: user-defined relations in technology tables
8.	0/1	no/yes: annually built historic capacities
9.	0/1	no/yes: bounds on activity and capacity of technologies
10.	0/n	>0: n = number of page with 1. technology
11.	0/1	no/yes: new page for each technology
12.	0/1	no/yes: check for technology identifier repetition
13.	0/1	1 or 3: additional line of input after switches : (a3,f10.4) currency name, conversion factor to unit in data files 2 or 3: additional line of input after switches : (a3) unit name for variable O+M costs
14.	0/1	0: costs are given relative to total output, 1: relative to main output
15.	nn	maximum number of lines per page (default 60)
16.	-1/0/1	-1: skip next two lines of input 0: do not skip lines 1: read two lines of input with commands for laserprinter: first line to change to landscape before energy form names, second line for change back to portrait mode.

If all additional lines exist, they could have the following form:

```
!R! SPO L; FONT 17; EXIT;
!R! SPO P; FONT 1; EXIT;
US$ 1.
kWa
```

Control of the type of report generated is possible using the flags of `run_msg`, see section 2.6.

Part III

The Post Processor of MESSAGE III

Chapter 16

Introduction

CAP is a Calculator Program designed to operate as post-processor program to the MESSAGE III energy supply model.

The calculating instructions used by CAP follow standard mathematical notation. In addition, some extensions have been made to account for certain model specific requirements. They include the calculation of growth rates or of normalized time series, for example. Moreover, two interpolation routines (natural cubic spline and linear interpolation) are available, permitting the use of files with time periods of different length.

16.1 Introduction to the Application of CAP

Cap requires two input files for controlling the data handling. The first of those files, called Control Variable File thereafter, is used to indicate to the program which of the data files are to be used for a certain application. The second one, called Calculating Instruction File, contains information on how to process the data extracted from one of the data files assigned to CAP. For this purpose a variable name identification scheme was developed that enables the program to identify on which of the data files the indicated data can be found. That scheme is described in detail in Section 21 of this user guide. As CAP was especially designed towards the handling of time series, each of the variable names used in the data handling instructions represents actually a time series. The format can best be seen from the examples given in chapter 23.

Chapter 17

Description of Input Files

17.1 Control Variable File (`xx.tit`)

This file, the so-called title-file `xx.tit` (see section 3.6) contains general information such as switches to be set for plotting, interpolation or extrapolation of time series and ones indicating the input files to be used. The file gives also one line of the header written to an output file, which can be used to indicate e.g. the name of the region or country for which tables and/or plots are to be produced. Usually one of these files can be used together with several calculating instruction files.

The file is divided into three information blocks, each beginning with a new line. Within those blocks free format can be used, i.e.,

- Numbers and strings are separated by at least one blank, and can be written in an optional number of lines;
- comma (,) indicates repetition of the previous string;
- semicolon (;) indicates that any information written behind it is ignored;
- blank lines are ignored.

The first block contains 16 integer numbers. The first 11 yield information on the files to be processed and outputs to be produced. The remaining 5 integers comprise the number of time periods required in the output file, a switch for the output format required, and 3 control integers for plotted outputs. The second block gives labels and lengths of time periods to be considered. The third block contains, if required, the output format statement and the first line of the title of the output file.

Throughout this guide all entries described are identified by E(x,y,z) with

x	number of the logical file containing the entry;
y	number of the block in a file;
z	number of the entry in a block.

17.1.1 Content of blocks

Block 1

E(3,1,1) plotter control;

see also comments on entry E(5,2,2), and chapter 19 Description of Output Files, for possible table formats;

0 for generation of tables only;

1 for generation of tables and an intermediate file containing information to be transferred to the plotting program CAPGRAPH;

–1 as 1, but labeling of plots will be suppressed.

2 produce plots with color information

E(3,1,2) output record length control;

0 record length set to a maximum of 132 columns;

n record length set to a maximum of n columns.

E(3,1,3) error message control;

0 messages are written to unit 6 only;

1 messages are also written to unit 1 (tabular output); messages concerning errors which result in program termination are written to both files in any case.

E(3,1,4) interpolation control;

if the period lengths of the time series read from the data files differ from those given in block 2 of this file, they are interpolated before being used further;

0 no interpolation allowed;

1 for spline interpolation;

2 for linear interpolation.

E(3,1,5) extrapolation control;

if the time horizon of time series read from the data files is shorter than the time horizon required for the output (as indicated in block 2 of this file), they are extrapolated before being used further;

0 no extrapolation allowed;

1 for simple repetition of the last value;

2 to invoke linear extrapolation routine.

E(3,1,6) MEDEE-2 dump file control;

0 if no data are required from the MEDEE-2 dump file;

n to open unit 8 for a file containing n MEDEE-2 dump files.

E(3,1,7) MESSAGE dump file control;

0 if no data are required from a MESSAGE dump file;

1 unused;

2 to open units 10, 14 and 17 for MESSAGE III matrix generator dump files;

E(3,1,8) 0 print all tables (see E(5,2,1))

1 print selected tables only

E(3,1,9) 0 print zerolines in graphics output

1 skip zerolines in graphics output

E(3,1,10) LP solution file control;

0 if no data required from a LP solution file;

n to open n direct access files obtained through the LP solution and read the number and labels of periods from a file on unit 9;

E(3,1,11) supplementary file control;

0 if no supplementary file used;

1 to open unit 14 for a preprocessed supplementary file;

2 to open unit 14 for a preprocessed supplementary file if available, otherwise to open unit 13 for a new supplementary file;

3 to open unit 13 for a new supplementary file;

E(3,1,12) number of time periods in the output file, the maximum number of time steps possible, is to be set prior to compilation of the program.

E(3,1,13) output formatting control;

this switch is only active if E(5,2,2) is set to 3 or –3;

0 tables are produced using the format (';',10a1,/,8(f10.3)) or, if the name of a time series is left blank, with (8(f10.3));

n number of format statements; if the format required is to be read from the first n lines of block 3 of this input file; there can be up to five format statements.

a negative number suppresses ';' and 'nt year' lines in output

E(3,1,14) plot labeling control;

the number given is used to divide the total time horizon, the result gives the number of years between the labels on the time axes of plots.

E(3,1,15) tick mark control;

the number given is used to divide the total time horizon, the result gives the number of years between tick marks on the time axes of plots.

E(3,1,16) time step control;

0 if the time steps required in the tabular output are equidistant, the base year and the step width are to be given in block 2 of this file;

1 if the time steps required are of variable length, the labels for all time steps (i.e. the first year of each period) plus the the last year of the total time horizon have to be given in block 2 of this input file.

Block 2

E(3,2,1) labels of time periods;
if entry E(3,1,16) is set to zero two entries are read, the first one being the label of the base year, and the second one the length of the time steps; if set to one, then E(3,1,12)+1 entries, representing the labels of all periods plus the last year of the total time horizon, are required as input.

Block 3

E(3,3,1) formatting;
if entry E(3,1,13) is set to zero, then no entry is required here and the standard output format will be used;
if entry E(3,1,13) is set to one, the required format statements, each with up to 400 characters in up to 5 lines are read; the format statements are to be written in the usual FORTRAN FORMAT notation and have to be able to process the name of the time series (10a1) and E(3,1,11) real variables.

E(3,3,2) two header lines;
i.e. the first line, restricted to 60 characters, is used as a header line on printed tables, the second one , which may contain a maximum of 40 characters, is used as the first line of a three-line header for tables and plots. The complete title is split into two parts so that one common line can be used for several tables and plots. For example, if the same kinds of calculations are to be carried out for different scenarios only that line has to be changed for each scenario, and this input file can be used together with a number of calculating instruction files as described below.

17.2 Calculating Instructions (xx.cin)

This file, the so-called cin-file **xx.cin** (see section 3.6) contains switches more closely related to a given set of calculating instructions than those described under 18.1 as well as the actual instructions. The file contents are again divided into three blocks.

The first input block in this file contains the second and third lines of the three-line header. (The first is contained in the control variable on file unit 3.) The second block contains switches, information on the units in which tables and/or plots are to be labeled, as well as scaling factors. This block has to be ended by a commercial add (@) written to an extra line, separating in this way the second and the third input blocks from each other. The third block contains the actual calculating instructions.

17.2.1 Content of blocks

Block 1

This block contains two lines serving as second and third lines of the header written on tables and plots that are produced from this input file. Each header line is restricted to 40 characters.

Block 2

E(5,2,1) plotted output control;

the file (on unit 7) containing the information for the plotter software is produced only when entry E(3,1,1) is not set to zero and entry E(5,2,2) is set to an absolute value of 1 or 2; the values transferred to the plotter software are also printed as third table in the tabular output on unit 1 (even if one or both of the plotter control switches are set to zero);

- 0 for normal plotting i.e. all Y-values are plotted relative to the base line;
- 1 for cumulative plotting;
- 2 for plotting the shares of the sum of Y-values;
- +10 to select table for printing (see E(3,1,8))

E(5,2,2) output quantity and format control;

if the absolute value of this entry is less than 3, then entry E(3,1,1) is in control if plots are to be produced or not; in all other cases entry E(3,1,1) has no influence as no plots can be produced; in this context, see also the comments on entry E(3,1,1) and DESCRIPTION OF OUTPUT FILES, for various table formats;

- 0 to generate tables only;
- 1 to generate tables and plots, curves are interpolated;
- -1 as 1, but allows for fixed labeling of the Y-axes of plots; see also the description on entry E(5,2,9) below;

2 as 1, but all curves are plotted as step functions;

-2 as 2, but allows for fixed labeling of the Y-axes of plots; see also the description on entry E(5,2,9) below;

3 to print transposed output matrix to simplify further processing of that file, in this case all lines containing text will be preceded by a semicolon; thus the resulting table can directly be used as input file to MESSAGE III , which interpret a semicolon as the begin of a comment.

-3 as 3, but lines that contain only zeros are not printed;

13 as 3, but adds one line containing the sum over all time series shown in that table;

4 to print the number of the variable in the LP solution file and its status in extra columns;

-4 to print the status of the requested variable in the LP-solution;

E(5,2,3) labeling control;

-1 no labeling of tables and plots;

0 to label outputs 'MESSAGE III ';

1 to label outputs 'MESSAGE III - NL';

2 unused;

3 to label outputs 'MESSAGE III - MIP';

4 to label outputs 'MEDEE-2';

E(5,2,4) table header control;

-2 to suppress certain control characters written to the tabular output file; these control characters are only meaningful if the output file is to be used as input file to CAP later;

-1 to suppress the header lines, date, and time written into the table header, this is useful when the output file produced is to serve as an input file to another program;

0 all labels are written to the tables.

E(5,2,5) units of variables in outputs;

this entry has no influence on the calculations as the program does not know anything about the units of the values read from the various input files, it is up to the user to convert the results to the unit required; the string may have up to 8 characters where blanks have to be padded by a tilde (~); this string appears in the first (or only) table produced and/or as left-hand side label of plots; if no unit name is to be written to the output files, at least one tilde must be present indicating that entry.

E(5,2,6) units of variables in outputs;

this string is written to the second table and as right-hand side label on plots, again this entry has no influence on the calculations; if the first character of this string is a percent sign (%) the second table

contains the shares of each time series, but the plotted output remains unchanged and its right-hand side is labeled with the string following the percent sign immediately; as above, at least one tilde has to be entered to indicate the entry.

E(5,2,7) multiplicative conversion factor between the two units given in the previous entries, (Unit defined in E(5,2,5) * E(5,2,7) = unit defined in E(5,2,6)).

E(5,2,8) multiplicative scaling factor for all values; all time series calculated within CAP are multiplied by that factor.

E(5,2,9) upper label on left-hand side of plots, which is used for fixed labeling if E(5,2,2)=-1; (this entry can remain in the file even if unused); this option helps to produce plots with identical labeling so they can be easier compared with each other.

E(5,2,10) end sign, this is a commercial add (@) which is to be entered in the first column of a separate line.

If E(5,2,2) is set to -2 additional input is required as control for step functions; the first entry is to be the number of curves plotted as step functions the following ones are the numbers of the according curves which are derived from the consecutive number of the mathematical equation used in this file to produce the curve; curves not included in that sequence in are plotted as continuous functions.

Block 3

This block contains the actual calculating instructions applied. The format to be followed when setting up these instructions is explained in the next subsection. An example file illustrating the format specifications is included in Section 23. The number of mathematical expressions to be evaluated is limited to a number which has to be set via the dimensioning program CHDIM prior to compilation of CAP.

17.2.2 Format of mathematical operations

The calculating instructions are to be written as mathematical equations of the format:

X%color%pattern%linetype%symbol% $\leq$ {format} mathematical expression,

where X is the chosen name of the time series (with up to 10 characters) containing the result obtained by the mathematical operation. If more than one format is given (see E(3,1,13) and E(3,3,1)), a "{n}" after the = or < sign can be used to select the n'th format for this specific line. The first format is default. For further information regarding the %-sign-separated graphics instruction see chapter 19.2. The mathematical expression is build up using standard mathematical notation and can be extended to more than one line by typing an ampersand (&) or a backslash (\) as last character in that line. The number of continuation lines possible has to be set in a data statement prior to compilation of the program. The operands are the names of time series as they

appear in the input files used. They are appended by an identifier (:id) defining for the program the input file containing the time series in question and, in cases when the given name refers to a set of data, additionally which of the time series is to be taken from that set. For further instructing the program what operation to perform on a time series, these names can be appended by one ore more operators (in the format ':operator'). These operators can be appended to all mathematical expressions valid in CAP, i.e., time series identifiers, parenthesis, constants or internal time series. For more information on identifiers and operators, see 21 Description of Variables and, in particular, 21.6 Operators.

CAP knows the basic mathematical operators +, -, *, /, ** and parenthesis () as well as their hierarchy. As the mathematical expressions are only evaluated when a closing parenthesis is found or the end of the expression is reached, one has to take care not to use too long expressions without any parenthesis. The maximum number of nested parenthesis and the maximum number of operands enclosed in parenthesis can be set prior to compilation of the program and thus be adapted to the specific requirements of the user.

In addition, both constants and time series can be used by following the format requirements given below,

Constants — can be either used directly in a formula (identified as figures in square brackets []) or by using a stack. The stacking of constants is initiated by a ^ in the first column. Each variable to be stacked has to be written in the format
STOii value
ii being the stack number (up to 20). The end of a stacking sequence has to be indicated by ^ ^. The use of a stacked constant is indicated by RCLii: which is used like a usual variable in a mathematical expression;

Time series — are identified by ^n, with n being the numbers of the time series. This shorthand for 'read n-th time series' is to be used again like a normal variable in a mathematical expression. The time series themselves have to be entered one by one in the lines following the equation where they are used first;
if one of the time series already defined is to be used in another equation it can then be referred to by typing ^n, with n being the number used when that time series was entered; if ^n with an n number used earlier is used again in another equation, the previous time series with that number will be overwritten; the number of entries per time series used must correspond to the number given in entry E(3,1,12); the maximum number of internal time series that can be handled by the program is to be set in a data statement prior to compilation.

17.2.3 Control characters

The program recognizes a few control characters, provided they appear in the first column of a line.

\# — indicates to the program to ignore that line;

"	the rest of the line will be copied to the tabular output preceded by a '#' in the first column. So the output can, if entry E(5,2,2) was set to 3 or –3, be again used as an input file to CAP.
'	this line is copied to the output without any control character in the first column.
@	end of input.
–	after '@' allows to copy the lines following that control character to the end of the tabular output file. The same control characters as described above are to be used to identify the text lines. The text has to be ended by an '@' and one of the control characters described below. If E(5,2,2) is set to +/–3 this option has no effect as any lines preceded by text identification control characters are copied to the output according to their position in the input file. In all other cases however all such lines are, if this option is not used, typed just after the header.
+	after '@' indicates that another calculating instruction file follows on the same input file. The resulting tables are separated by a '1' in the first column (FORTRAN printer control character for new page). The plotting information is also written to a single file and identified accordingly. It is recommended to use this option if more tables should be produced as it saves the repeated loading of the MESSAGE III dump files.
@	after '@' indicates 'END OF FILE'.

Chapter 18

Description of Data Files

18.1 MESSAGE III Technology Name File (`region.tn`)

This file contains the names of and descriptions for all technologies, user defined constraints and energy forms included in the MESSAGE III input file. The information contained in that file is only used when CAP is used in interactive mode.

18.2 LP Solution File (`region_scen.dir`)

The actual LP solution file has to be preprocessed before being used as input to CAP. This preprocessing is performed by program MKSF, that combines the results obtained for each time series into one record that are then sorted alphabetically to allow the use of an efficient search algorithm in CAP. The resulting file is written as an unformatted direct access file. For reasons of comparing different LP solutions or for aggregating various solutions it is possible to use up to 8 different solution files, where the files 2 to 8 are to be assigned to units 18 upwards. See chapter 21.3 on how to extract data from these files.

Besides the data file(s), addition files (`region_scen.int and region_scen.rec`) are needed forwarding information on the content of the direct access file(s) to the program. This file is produced by MKSF during preprocessing of the solution file. If more than one solution is being processed at the same time it is silently assumed that all files are of the same size and contain the same time series names.

18.3 MESSAGE III Dump Files

CAP reads some of the dump files generated by the matrix generator in order to be able to process input data for technology and user-defined relations.

18.4 Supplementary File (region.fil)

The so-called fil-file (region.fil) may contain additional time series, which are either supplied by the user or are obtained from a previous run of CAP with entry E(5,2,2) set to 3 or –3.
If the file is created manually the following format specification have to be followed:
The file is divided into three blocks containing

- comments,
- number and labels of time steps, and
- time series.

The comment block, made up of an optional number of lines, can be used to identify the data contained in that file.
The beginning of the actual data blocks is signaled by the string

```
NT  YEARS
```

which must be written in the format (1x,a2,2x,a5) followed by a line containing the according figures in free format.
The next line can be used to change default values on the file format. This line has to be typed using the format

(1x, a1, 1x, i2, 1x, i3, 1x, a40),

if the default values should be used, then the according entry has to remain blank. To this end, the following four values can be reset:

File content		if this entry is left blank (default) it is assumed that all of the time series names and values are read from the file assigned to the unit number specified by the next entry;
	n	indicates that no names of time series are given explicitly but that they are internally created as natural numbers in increasing order;
	a	indicates that the names of the time series are read continuing on this file but the time series are read from the file connected to the logical unit specified via the next entry.
Unit number		if the time series are not following below on this file then any unit number greater than 15 can be chosen, and the program continues to read from there; this option enables one to process files which are generated by any other computer program, provided the data file used satisfies the only conditions that time series are given row-wise and each one begins with a new line.
File width		the standard number is 80 columns, and the maximum 400.
Input format		the default considered is 'free format' (see description below); if the file to be read follows a defined format this can be specified here, resulting in a faster processing of that input file.

The line following the one used to reset default values is ignored by the program.

When the default input format is used the following rules have to be observed when setting up the file containing the actual set of data.

- The first column is reserved for control characters; number sign (#) indicates that this line is to be ignored and a commercial add (@) indicates the END OF FILE; but can be omitted at the physical END OF FILE. All other characters in the first column are ignored.
- All variable names and values have to begin in or after column two.
- All values are assumed to be represented in real format. If integer numbers are used the input format has to be stated explicitly as Fn.0.
- All entries have to be separated by an optional number of blanks.
- Each time series has to start in a new line, but may continue over an optional number of lines.

Example

```
FIL file for testing

 nt  years
  4 1991 1994 1997 2000


 value1        1.000 1.0 1.0 1.0
 value2        2.000 2.0 2.0 2.0
 value3        3.000 3.0 3.0 3.0
```

18.5 MEDEE-2 Dump File

This dump file holds some general information and all the time series being inputs to or outputs from MEDEE-2. The information is stored in (1+2*NCAL) unformatted FORTRAN records, with NCAL being the number of time steps for which MEDEE-2 was run.

The first record contains the following variables:

REGION	(character*8)	name of region or country;
CASE	(character*4)	number of scenario;
NCAL	(integer)	number of years;
YCAL(1:NCAL)	(integer field)	list of these years;
UNAME	(character*4)	MEDEE-2 output energy units.

Each of the following NCAL blocks consists of two FORTRAN records, the first containing

PARVAR(1:231) (real field) inputs to MEDEE-2; and the second

DERVAR(1:206) (real field) outputs from MEDEE-2.

18.6 Variable Location in the MEDEE-2 Dump File

This file contains the names of all the variables used in MEDEE-2 and an index representing their location in the dump file produced by MEDEE-2. The location is defined by the number of bytes preceding a variable. Variables are ordered in the sequence in which they appear in the common blocks PARVAR (parameter variables) and DERVAR (derived variables) of the MEDEE-2 program. For a list of the variable names see Kahn, A., A. Hölzl; 'Evolution of Future Energy Demands...'.

The names of time series given in the calculating instructions file are compared to those listed in the variable location file; and the given index enables the program to locate the respective time series in the MEDEE-2 dump file. These locations remain the same, unless the MEDEE-2 program is changed.

Retrieval of a well defined group of variables can be speeded up by setting up a new variable location file containing the names and locations of these variables only. This procedure is recommended if certain variables are frequently to be extracted from various dump files.

Chapter 19

Description of Output Files

19.1 Tabular Output (cin_scen.tab)

For E(3,1,1) * E(5,2,2) set to 0, +/-1, or +/-2

Up to three tables are produced. The first one is labeled with the unit given in the calculating instructions file (unit 5) by entry E(5,2,5). The second one gives them in the unit set by entry E(5,2,6) and using the conversion factor given as entry E(5,2,7). The values in the third table are those also used for plotting, following the instructions given by entry E(5,2,1). Tables that would show identical figures are suppressed. The input files for the graphical output (`cin_scen.grf` and `cin_scen.grc`)is produced only if the absolute value of entry E(3,1,1) is set to 1.

For E(5,2,2) set to +/-3

Setting this switch to 3 results in a transposed matrix, which usually simplifies processing of the file produced. (Time series are written as rows rather than as columns.) The format of the file is such that it can again be used as an input file to CAP (see also description of the supplementary file above). Each text line is preceded by a semicolon (;), in this way such a file can also directly be used as an input file to MESSAGE if, e.g., final or secondary energy demands are calculated from MEDEE-2. If the entry is set to -3, then all time series containing only zero values are not printed.

For E(5,2,2) set to 4

This table format can be used to check LP solutions for upper or lower limits on variables or rows and to get the number of the row or variable in question by that it is identified in the LP solution. The format is similar to that of the first table described above, but an additional column containing the number is printed before and one showing the status of the row or variable is printed behind the column containing the values required. The status is expressed as up, (at upper bound), lo (at lower bound), or bs (in basis). If a mathematical expression is used to calculate a time series it shows the number and status of the last variable in that expression.

For E(5,2,2) set to -4

While the format is similar to that obtained through E(5,2,2)=4 the column showing the number of that variable in the LP solution file is not printed in this case.

19.2 Interface to Graphical Output

As the program is set up now it produces (if entries E(3,1,1) and E(5,2,2) are set up accordingly) output files (`cin_scen.grf` and `cin_scen.grc`)containing all relevant information necessary for a graphics program. These files are written in formatted records, in order to allow for device independent postprocessing.

Chapter 20

Description of Intermediate Files

20.1 Interpolated Time Series Storage

This file is a prerequisite for CAP being used with E(3,1,11)=1. If E(3,1,11) is set to 2 and the file has zero length the values of the time series in the supplementary file are interpolated, as necessary, and stored to an intermediary file named `mks`.. If E(3,1,11) is set to 1 it is assumed that the interpolation was performed in a previous run, and any values requested from the supplementary file are taken from this intermediary file instead of the supplementary file. Setting E(3,1,11) to 3 indicates that the supplementary file is to be used as input file and is to be processed as required.

20.2 Input Format Conversion

This file is an intermediate file, used for converting data given in free format to ones written in usual FORTRAN formats. It is named `Cin15`. If errors occur in reading this file, the reason generally lies either in the tit-file or in the cin-file.

Chapter 21

Description of Variables

Generally all variable names used in the calculating instruction file represent names of time series. If the values identified by a variable name is a single constant it is automatically expanded to a constant time series.
All variable names are written in the format below.

VARIABLE:DESCRIPTOR[:OPERATOR].

With VARIABLE representing the time series name to be searched for in one of the input files; DESCRIPTOR identifying the file of residence of the variable, and providing for a more exact variable specification if there exist more time series relating to the same variable (e.g. capital cost, efficiency, etc. related to a specific technology included in MESSAGE). In the cases of certain input files it is also possible to process time series for the same variables contained in different solution files; e.g., for variables read from the MEDEE-2 dump files, and for those read from LP solution files preprocessed by program MKSF. The construction of variable names and their descriptors is explained in some detail in the following sections. The description of OPERATORS is given in Section 21.6.

21.1 Variables Read from the MESSAGE III Dump Files

21.1.1 Technology variables

Format: name:descriptor

name — for conversion technologies:
4 character variable name constructed according to the MESSAGE III user guide; the first character is the energy level identifier; the second one the main energy input identifier; the third one the additional technology identifier; and the fourth character the main output identifier;
for storage technologies:

3-character variable name constructed according to the MESSAGE III user guide plus control characters; the first character is the level identifier, the second one the identifier of the fuel to be stored, the third one the additional technology identifier, the fourth character in this field has to be set to dot (.), the fifth character is to be set to **g** when referring to values related to the input/output part or to **v** to indicate values related to the volume part of the storage device; it remains unused (but can be set to **g**) when a storage device with a fixed relation between generation and volume parts is represented by this technology.

:descriptor

cap	capital cost per unit of total output;
var	variable operation and maintenance cost per unit of total output;
fix	fixed operation and maintenance cost per unit of total output;
ctm	delivers a value representing the multiplicative factor necessary for converting cost per unit of total output to cost per unit of main output (i.e., name:cap * name:ctm gives the capital cost per unit of main output);
pll	technical plant life;
plf	availability factor;
eff	main output to main input efficiency;
lev	calculates levellized cost of technology using the technical plant life as economic lifetime and setting the long term discount rate equal to the interest rate; total cost excluding fuel cost.
dec	parameter describing the decay of storage contents;
rgv	relation between generation and volume capacities of storage devices;

each of the following descriptors is valid for a group of entries and delivers inputs to or outputs from the technology relative to one unit of main input or one unit of installed capacity respectively, the dot (.) has to be substituted by the identifier given in the MESSAGE III input file used to produce the dump file,

ei.	energy input, if the dot (.) is set to star (*) instead of a specific fuel identifier it delivers the sum over all energy inputs;
eo.	energy output, setting dot (.) to star (*) results again in the sum over all outputs;
ri.	nuclear fuel requirement;
ro.	nuclear fuel retirement;
fr.	fraction of total input or output of the technology in the load region indicated (dot (.) set to the number of the load region), when the load duration pattern of a technology was fixed;
$\{^{o}_{c}\}\,\{^{1}_{2}\}$	coefficient in user defined relations, where 'o' or 'c' indicating a relation to one unit of output or one unit of new installed capacity respectively and (1 or 2) stands for the type of the

constraint; the dot is used for identifying the number of loadregion this value is assigned to (set to dot (.) if the coefficient is not defined for a specific load region); the name of the constraint in question has to be added as ':name';

21.1.2 User defined relations

Format: name:descriptor

name		name of the user defined relation
:descriptor	$\{^{r}_{v}\}\,\{^{1}_{2}\}$	coefficient in user defined relations, where 'r' or 'v' indicating the right hand side of the constraints or the value of the objective function coefficient implied on all row entries and (1 or 2) stands for the type of the constraint; the dot is used for identifying the number of loadregion this constraint is defined for (set to dot (.) if the constraint is not defined for a specific load region or to get the sum over all load regions);

21.1.3 Demands

Format: name:loadregion

name	dm.	where the dot has to be set to the identifier of the demand sector in question.
:loadregion	lr.	gives the fraction of the demand falling into the load region indicated, if dot (.) is set to t this results in the total demand.

21.1.4 Other variables

Format: name:

name	lr.	length of load region specified in dot (.) as fraction of one.
	lp.	length of period specified in dot (.) as given in entry E(3,2,2); if the dot is set to a specific period identifier (in this case represented by a letter rather than a number), then the whole time series is set to the value of the length of the period indicated; if the dot is set to zero the time series will contain the set of the period lengths as given in entry E(3,2,2).

21.2 Variables Read from the LP Solution File

Format: name[:sn.]:descriptor

name — first 7 characters of the 8 character time series name in the LP solution file where trailing dots can be omitted; with the character identifying the number of the load region in question (seventh character), set to 0, the program calculates the sum over the load regions; otherwise the time series relating to the load region in question is extracted.

:sn. — is needed if more than one solution is to be processed at the same time for purposes of aggregation, etc.; the number of the file to be chosen is the consecutive number of the file in the order they are assigned to the unit numbers, where the first file is to be assigned to unit 4 and the rest is assigned to units 18 upwards; the dot (.) is to be set to the number of the solution file in question if a time series is to be extracted from one of the files; if it is set to zero, or the entry is omitted altogether, then the sum over the time series in question is calculated using all LP solution files;

:descriptor — the descriptor is used to identify the column in the LP output from that the time series is to be extracted;

act	activity column,
sla	slack (in rows section) or discounted objective function value (in columns-section),
obx	the objective function value is undiscounted relative to the center of the periods (to be used for all variables excluding capacities),
oby	the objective function value is undiscounted relative to the beginning of the periods (to be used for capacity variables),
lol	lower limit,
upl	upper limit,
dis	shadow price discounted,
shp	shadow price undiscounted.

21.3 Variables Read from the Supplementary Input File

Format: name:descriptor

name — variable name as given in the supplementary input file.

:descriptor

fil	has to be appended to the variable name to indicate that the time series is to be read from the supplementary file.

21.4 Variables Read from the MEDEE-2 Dump File

format: name[:sn.]:descriptor

name		variable name as used in the MEDEE-2 program, see Khan A., A. Hölzl; 'Evolution of Future Energy Demands..' for a full list of the variable names.
:sn.		is used if more than one solution is processed at one time for aggregation purposes, etc.; in this case the dump files produced by several of MEDEE-2 runs just have to be combined into a single file; the dot (.) has to be set to the number of the solution file in question if a time series is to be extracted from one of the files; if it is set to zero, or the entry is omitted altogether, then the sum over the time series in question is calculated using all MEDEE-2 solution files available;
:descriptor	med	to be appended to indicate that the time series is contained in a MEDEE-2 dump file.

21.5 Use of Internally Stored Values

The results of a given number of calculations are stacked and can be used in the following calculations by using the left-hand side name of the equation referred to. The number of time series to be stacked is to be set prior to compilation of the program.

Format: name:

The search procedure for the internally stored time series follows the 'last in - first out' principle.

21.6 Operators

Operators can be appended as ':operator' to any type of operand used in a mathematical expression; i.e., variable names of all types described above, internal constants and time series and closing brackets. It is also possible to add more than one operator when separated by colons. They are applied to the time series in their sequential order. Only the final result following the completion of all transformations appears as output. As some of the operators just copy values from one field element to another (e.g. :sh+, :sh-, :ex1) care has to be taken when time series with unequal time steps are used.

Format: :operator

:ind	the time-series is divided by the value of the first period (normalized);
:exp	calculates the exponential value of each member of the time series, (value(i) = exp(value(i));
:grp	calculates the total growth rate per period (the value for the last period is set to zero);
:prp	calculates the growth rate in percent per period (the value for the last period is set to zero);
:gra	calculates the annual average growth rate per period (the value for the last period is set to zero);
:pra	calculates the annual average growth rate in percent per year (the value for the last period is set to zero);

:prd multiplies the values in the time series by the length of the period, thus allowing to calculate e.g., the total new installed capacity per period from the values given as annual averages (as usually the case in time series taken from LP solution files derived from matrices produced from either of the MESSAGE models);

:cum calculates the cumulative value of the time series up to the end of each period, and thus can be used e.g., to calculate the total resource extraction; this operator can for instance also be used to calculate exponential growth functions; e.g., the expression

[1.03] ** ([1.]:cum - lp0:) delivers
1.000 1.159 1.344 1.558 2.094
if entry E(3,2,1) was set to
1975 1980 1985 1990 2000;

:log calculates the natural logarithm of each member of the time series;

:lnp calculates the natural logarithm of the quotient var(t+1)/var(t) (the last value of the resulting time series is set to one);

:lnt calculates the natural logarithm of the quotient var(t+1)/var(1) (the last value of the resulting time series is set to one); these last two options are very useful for calculating elasticities;

:dif calculates the difference between field elements, i.e., 1st element = 2nd element - 1st element, where the last element is set to zero;

:#nn fills the whole time-series with its nn'th element; (e.g.; use '# 2', '#+2' or '#02' when referring to the 2nd element)

:0nn sets the nn'th element of a time series to zero (0), all others remain unaltered;

:1nn sets all elements but the nn'th element of a time series to zero (0);

:sh+ shifts the whole time series one element to the right, the first element is set to zero.

:sh- shifts the whole time series one element to the left, the last element is set to zero.

:ex1 fills the last plus one element of a time series with the last element;

:ex2 fills the last plus one element with a value calculated by linear extrapolation from the last two values; this or the former operator can be used to avoid zeros written per default to the last field element e.g. when differences (:dif) or logarithms of quotients (:lnp, :lnt) are calculated;
(the sequence of operators: :dif:sh+:ex2:sh- can be used to calculate the differences between the values for each time step and to get a linear extrapolation for the last time step, which is set to zero by the operator :dif);

:new this operator allows for a rough calculation of annual new installations on the basis of energy flows; if this operator is used four variables are read in free format from the first line after the formula including this operator; if it is used more than once in one formula a new line is required for each occurrence; these four variables are the plant life (to be given as integer variable), the plant factor, the energy flow in the base year and the historic growth rate of the flow (all to be given as real variables); using this information and the time series of the annual flow for each period, the program then calculates the annual new capacities needed to replace the ones going out of operation and the ones necessary to meet additional demands;

:nii with ii being the plant life, also calculates the annual new installations on the

basis of annual energy flows; the difference to :new being, that here the existing capacity is calculated as the annual flow in the first period divided by the plant life based on the assumption that the build-up of this capacity was equally distributed to the previous years; this leads, in the case of increasing growth rates for new installations during the periods prior to the modelled time frame to an overestimation of the needs for new installations during the first few periods of the modelled time frame; the resulting time series has to be divided by the plant factor explicitly.

To obtain the *total installed capacity* of a technology, it is necessary to calculate it explicitly by adding the slack of the capacity equation to the actual output divided by the length of the load region and dividing the result by the plant factor. See subsection 23.1.2 for examples for technologies with and without load regions.

Chapter 22

Supporting Programs

This section describes how to use the program needed to change the dimensions of CAP to the required size and how to run the programs necessary to convert output files from other programs to a format which can be interpreted by CAP.

22.1 The Dimensioning Program CHDIM

This program is common to all codes related to the MESSAGE III model and is used to adapt the array dimensions of the codes to the specific requirements of an explicit application. All codes handling the same type of information have to contain the same data statements and common blocks with arrays being of equal size for a complete model run. CHDIM works on blocks of information in the FORTRAN codes. The beginning of such a block is identified by three COMMENT lines, where the second line contains a four character block name. The end of the block is signalled by one more empty COMMENT lines. Thus such a block has the following pattern:

```
c
c  name
c

      Block of information to be changed, e.g.; dimension statements,
      common blocks or data statements.

c
```

Program CHDIM reads three switches from the standard input (unit 5). The first switch tells CHDIM under which operating system the programs are running. Currently there are two options, i.e.; U for UNIX or C for Control Data operating systems NOS or NOS/BE, but these options can easily be extended to other operating systems if necessary. The next two switches control the performance of the program where the first one is related to dimensions of arrays common to all programs and the second one to dimensions of arrays contained in CAP only.

If the according switch is set to zero, the program tries to read the required dimension from files assigned to unit 4 (in the case of dimensions relevant to all programs) and unit 3 (in the case of dimensions relevant to CAP only). If no input file is found, then the

program uses default values when writing the new dimension statements. Here only the second input file is described in detail, the description of the first file can be found in Messner S.; 'User's Guide for the Matrix Generator of MESSAGE III '. The input file read from unit 3 contains the following variables (default values are shown in parenthesis):

ntm	–	maximal number of time steps (13);
ncols	–	maximal number of time series stored (30);
nstac	–	maximal number of internal stacks (30);
nzr	–	maximal number of internal time series (10);
ikl	–	maximal number of nested brackets (10);
icont	–	maximal number of continuation lines (4);

The entry ncols, controlling the maximum number of time series stored, at the same time, when entry E(5,2,2,) is set other than 3 or -3, represents the maximum number of time series to be written to the output files; nstac defines the maximum number of operands in a mathematical expression without being enclosed in brackets; nzr refers to the time series defined in the calculating instruction file by ^n as described in section 17.2.2.

If the switch is set to 1, the program writes instead of complete blocks of code, containing dimension and common statements, depending on the setting of the first switch either

include 'com*name*'

in the case of U(NIX), or

*call com*name*

in the case of C(DC). These statements are used to indicate that a file called 'com*name*' is stored outside the program and has to be included during compilation (in the case of UNIX) or to be included when the program is extracted from an UPDATE library in the case of CDC operating systems. In case of other operating systems the write statement can be adapted easily to the specific requirements. In any of these cases (switch set to zero or one) the program to be changed is read from unit 1 and the new code is written to unit 2.

Setting the switch to -4 results in creating the blocks of information on separate files having the required names and format. This performance is required when using the UNIX operating system.

If the switch is set to -5, the program creates one file containing all blocks of information according to the format specifications required by CDC UPDATE libraries (i.e.; each block is preceded by *cd com*name*).

22.2 Sorting the Solution — Program MKSF

As indicated earlier (section 18.2) MKSF is needed to convert a solution file produced by a commercial linear programming package to a format that can be interpreted by CAP. The present version of the program is able to process solution files produces by

MINOS (Stanford University, Department of Operations Research). If other packages are used, then the according changes have to be done in SUBROUTINES FINDR and GETREC as described there. It is assumed that the solution file is written in two parts not containing any blank lines, the first block contains the rows section, the second one the columns section. SUBROUTINE FINDR is used to set the file pointer to the first row (=func), SUBROUTINE GETREC reads one line from the solution file and to set the file pointer to the first line of the column section after the rows section was read.

The program reads the solution file from unit 1. Another input file, created by program ROWS on unit 3, is read from unit 3. This file contains the name of the LP package used, that can be used as a switch to select the according READ or FORMAT STATEMENTS in MKSF. The output file is written to unit 4 and eventual error messages are written to unit 6. Two additional output files are produced. The first one (on unit 9) contains additional information on the solution file to be forwarded to CAP. The second one (on unit 8) contains information on how the file was sorted. That file can be used as an input file on the same unit when a solution containing the same row and column names is to be sorted later on. To indicate that a file is to be sorted according to previous information the switch read from unit 5 has to be set to 'NOSORT', otherwise to 'SORT'.

22.3 Graphical Output

MESSAGE III has no fully integrated graphics routines. A graphics program is available for MS-DOS applications (also under Windows) that visualizes the graphics prepared on files `cin_scen.grf` and `cin_scen.grc`. This program is not documented in this documentation.

22.4 Analyzing the Solution — Program ANA

Program ANA allows to scan an LP solution for all active constraints. This is specially very useful for analyzing a solution if one models long energy chains or uses generally large models. Optionally the output file can show the same values as described in section 21.2. Additionally the status (ll, ul, fx) of the row or column can be shown. The output is produced in a way that all variables and rows concerning one technology, user defined constraint or energy form are contained in one group.

For this purpose ANA processes a number of dump and intermediary files produced during matrix generation and postprocessing the solution file. The output file is generated on file `region_scen.nbf`.

The type of information generated is controlled from the calling routine, `run_msg`. Possible outputs are:

- act: variable/constraint activity (value)
- sla: constraint: slack (difference to limit); variable: objective gradient as it is
- obx: objective gradient, discounting from middle of period assumed
- oby: objective gradient, discounting from beginning of period assumed

- lol: lower limit
- upl: upper limit
- dis: shadow price, discounted
- shp: shadow price, undiscounted
- at: indicate status of variable/constraint (LL/UL/BS)

Chapter 23

How to Get Started

This section shows a few examples of input files to CAP together with the resulting output files. These examples do not cover the full set of options provided, but sketch some basic information needed to get started with CAP.

23.1 Complete Sets of Input Files to CAP

23.1.1 Calculations processing MEDEE-2 results

The Control Variable File

The file shown is typed with one entry per line only, note that this format is, as indicated in section 17.1, not the only possible one, but a number of entries could be typed also in one single line, care has only to be taken, that each of the three blocks of entries described begin with a new line.

```
;Block 1
  1  ;  switch set to allow for plotting
  0  ;  switch reserved for later purposes
  1  ;  error messages are written to the file assigned to unit 6
     ;  and to the tabular output file on unit 1
  2  ;  indicates that linear interpolation is to be used
  0  ;  no extrapolation of time series allowed
  1  ;  to read from a MEDEE-2 dump file on unit 8; this implies
        that
     ;  the file containing the variable locations has to be
        available on unit 2
  0  ;  no variables to be read from MESSAGE III dump file
  0  ;  unused switch
  0  ;  no input to be read from IMPACT output file
  0  ;  no input to be read from a LP solution file
  0  ;  no input from a supplementary input file
  8  ;  number of time steps required in the output file
  1  ;  output formatting control, format is set so it fits onto
        the
     ;  printed page, the format is given in Block 3
  5  ;  plot labeling
 10  ;  tick marks
  1  ;  time steps are given explicitly
;Blocks 2 and 3
;labels of periods, format, and first header line
1975 1980 1985 1990 2000 2010 2020 2030 2040
(';',10a1/5x,8f7.1)
EXAMPLE 1 FOR CAP
```

The calculating instruction file

This file is used to prepare the demand related input file to MESSAGE III . The switch controlling the output format is set so, that the output file can directly be read by the MESSAGE III matrix generator program.

The calculations performed result in the final energy demands for specific uses of certain energy carriers and for useful energy demands that can be met by a number of different final energy forms. The final energy demands are calculated for electricity in the household/service sector and for transportation purposes (ELECTR), specific electricity demand in the industrial sector (ELIND), the demand for coal used in the transportation sector and for steel production (CLSPU) and the demand for feedstocks (FEEDST). The useful energy demands comprise the demands for low and medium temperature process heat (PH/ml) and furnace heat (PH/h) needed in the industrial sector and the demands for low temperature heat in the household/service sector (HHS/l). Parts of the demand for specific uses of electricity have to be onverted from TWhrs/yr to GWyr/yr as required by MESSAGE III .

```
DEMANDS FROM MEDEE 2        2. title line
INPUTS TO MESSAGE III  3. title line
0       ;  not effective in this case
-3      ;  to print transposed matrix and suppress zero lines
1       ;  to label outputs with MEDEE-2
0       ;  to write all headers
GWa     ;  main energy unit
-       ;  second energy unit (not effective in this case)
1.      ;  conversion factor between the units (not effective)
1.      ;  multiplicative factor
@       ;  end of second block

#
# setting of a constant
#
^
sto01 8.76 factor for converting TWhrs to GWyrs
^  ^
#
# begin of calculations
#

ELECTR   = eltr:med + elac:med + (elap:med + elsv:med) / rcl01:
LIQUIDS  = tmftr:med + mfind:med
FEEDST   = feed:med
CLSPU    = coke:med + tcltr:med
ELIND    = elsind:med / rcl01:
PH/ml    = usman(1):med + usman(3):med + ffacm:med * [.65]
PH/h     = usman(2):med
HHS/l    = sh:med + thserv:med + hw:med
```

The variable location file

As the variables used in this calculations are to be extracted from the MEDEE-2 dump file, it is necessary to supply an input file indicating the location of the variables. The variable location file (a part of it is shown below) contains the names of the variables used and their relative location in the dump file. See section 18.1 for a description of the file contents.

```
eltr        1408
elac        1700
elap        1496
elsv        1552
tmftr       1400
mfind       1204
feed        1200
coke        1196
```

The tabular output file

```
;TABLE 1
;MEDEE-2                                    EXAMPLE 1 FOR CAP
;date: 21. 7.1982                           DEMANDS FROM MEDEE-2
;time: 15:18  hrs                           INPUTS TO MESSAGE III

;nt  years
; 8  1975  1980  1985  1990  2000  2010  2020  2030

;GWa
;ELECTR
       62.1   81.5  101.0  125.0  173.0  228.5  287.5  350.0
;LIQUIDS
      423.3  529.8  636.4  740.7  949.4 1145.5 1320.5 1474.3
;FEEDST
      126.4  154.9  183.3  215.2  279.0  338.0  392.5  442.7
;CLSPU
       79.1   94.7  110.2  125.9  157.3  182.8  204.3  221.8
;ELIND
      115.4  141.9  168.4  199.2  260.8  327.4  393.9  460.1
;PH/ml
      134.2  158.3  182.4  207.2  256.7  297.8  336.1  371.5
;PH/h
      134.2  158.3  182.4  207.2  256.7  297.8  336.1  371.5
;HHS/l
      240.4  284.6  328.8  357.5  414.7  454.0  488.0  516.8
```

23.1.2 Calculations processing a MESSAGE III dump and a LP solution file

The control variable file

The control variable file shown here is, as opposed to the one in the previous section, typed using the minimum amount of lines possible.

```
0 , 1 , 2 , 2 0 0 -1 0 6 0 10 5 0
1975 10
EXAMPLE 2 FOR CAP
```

The calculation instruction file

The mathematical expressions used this input file show a number of calculations useful to interpret the results of MESSAGE III . The usual way would be to set up a number of input files; e.g., for identifying where the various primary energy carriers are used, or by which energy forms a given demand is met. Here these calculations appear rather mixed, the aim of the calculations is described in the file. As in the previous example the reader is assumed to be familiar with the set of models and, in this specific case, with MESSAGE III .

```
CALCULATIONS  WITH
MESSAGE III   RESULTS
 1 1 2 0
GWa/a
Quads
0.02989
1.
@
#
# amount of natural gas imported
#
NATGAS   =  iaga:act + iagb:act
#
# amount of electricity produced in hydropower stations
#
HYDRO    = (x.1e..0:act + x.2e..0:act)
#
# total installed capacity of LWRs
#
LWRcap   = (cxuue:sla + xuue:act * xuue:eff) / xuue:plf
#
# total installed capacity of coal power plants
#   (as this plant type is modelled with alternative operation
#    modes, the activities of both have to be added)
#
COALcap  = (cxc1e.1:sla &
         + (xc1e..1:act*xc1e:eff + xc2e..1:act*xc2e:eff) &
         / lr1:) / xc1e:plf
#
# electric power required in the first load region
#
elpow1   = dme:lr1
#
# amount of coal used during each time period
#
coaluse  = (rraa:act + rrab:act):prd
```

The tabular output file

TABLE 1
MESSAGE III EXAMPLE 2 FOR CAP
date: 23. 7.1982 CALCULATIONS WITH
time: 17:17 hrs MESSAGE III RESULTS

GWa/a

	NATGAS	HYDRO	LWRcap	COALcap	elpow1	coaluse
1975	185.67	9.33	0.	1.00	6.14	34.42
1985	199.58	13.28	2.61	8.61	9.82	172.22
1995	111.72	23.16	12.95	38.12	12.80	707.23
2005	78.58	27.61	8.61	44.76	14.32	1115.19
2015	50.77	31.62	8.95	55.31	15.85	1261.45
2025	56.08	29.73	6.97	57.18	17.11	1183.83
2035	53.25	29.79	6.29	57.78	17.50	1200.00

Quads

	NATGAS	HYDRO	LWRcap	COALcap	elpow1	coaluse
1975	5.55	0.28	0.	0.03	0.18	1.03
1985	5.97	0.40	0.08	0.26	0.29	5.15
1995	3.34	0.69	0.39	1.14	0.38	21.14
2005	2.35	0.83	0.26	1.34	0.43	33.33
2015	1.52	0.95	0.27	1.65	0.47	37.70
2025	1.68	0.89	0.21	1.71	0.51	35.38
2035	1.59	0.89	0.19	1.73	0.52	35.87

	NATGAS	HYDRO	LWRcap	COALcap	elpow1	coaluse
GWa/a						
1975	185.67	195.00	195.00	196.00	202.14	236.57
1985	199.58	212.86	215.47	224.08	233.91	406.12
1995	111.72	134.87	147.82	185.94	198.74	905.97
2005	78.58	106.20	114.81	159.57	173.89	1289.08
2015	50.77	82.38	91.33	146.65	162.50	1423.95
2025	56.08	85.81	92.78	149.96	167.07	1350.90
2035	53.25	83.04	89.33	147.11	164.60	1364.60

23.2 Samples for Mathematical Expressions used in CAP

As indicated above, this section will contain only single mathematical expressions followed immediately by the result obtained by CAP. For this examples the output file produced from the first set of input files is used as input file to CAP on unit 13. To indicate how that file was produced the control variable file and the beginning of the calculating instruction file are included.

The control variable file

```
 0 0 1 0 0              ;  entries  E(3,1,1)   to  E(3,1,5)
 0 0 0 0 0 1            ;  entries  E(3,1,6)   to  E(3,1,11)
 5 1 10 5 0             ;  entries  E(3,1,12)  to  E(3,1,16)
 1980 10                ;  entry    E(3,2,1)
 (1x,10a1,5f10.2)
 EXAMPLE 3 FOR CAP
```

The calculating instruction file

```
SAMPLE EXPRESSIONS

 1 3 2 -1
- - 1. 1.
@

'
' USE OF OPERATORS AND STORED TIME SERIES
' ========================================
'
' Total specific electricity demand:
' totel  = ELECTR:fil + ELIND:fil
totel    = ELECTR:fil + ELIND:fil
```

The tabular output file

```
USE OF OPERATORS AND STORED TIME SERIES
=======================================

Total specific electricity demand:
totel    = ELECTR:fil + ELIND:fil
totel         177.50    223.40    269.40    324.20    433.80

Index of electricity demand using the previous result:
elindex  = totel:ind
elindex         1.00      1.26      1.52      1.83      2.44

Growth of industrial process heat demand in percent per year:
PHgr     = (PH/ml:fil + PH/h:fil):pra
PHgr            1.67      1.43      1.28      2.17      0.

Linear extrapolation of previous time series
(the time series is shifted to the right by one element,
 then extrapolated and shifted back):
extr     = PHgr:sh+:ex1:sh-
extr            1.67      1.43      1.28      2.17      3.05

Cumulative use of liquid fuels up to the end of each period:
liqu-cum = LIQUIDS:fil:cum
liqu-cum     4233.00   9531.00  15895.00  23302.00  32796.00


USE OF INTERNAL TIME SERIES AND CONSTANTS
=========================================

Multiplication of the total feedstock demand with a time
series (e.g., share of feedstocks supplied by liquid fuels)
and division by a constant (e.g., the refinery efficiency):
liqu     = FEEDST:fil * ^1 / [.92]
             123.65    165.00    159.39    140.35    136.47

Use of the previously defined time series:
rest     = FEEDST:fil * ( [1.] -  ^1 )
              12.64      3.10     36.66     86.08    153.45
```

23.3 Using CAP to Produce Tables

Although the formatting options are somewhat restricted, it is possible to use CAP to produce tables in a format that can be controlled by the user. The example given here uses again the file produced in the first example.

The control variable file

```
 0 0 1 0 0    ; entries  E(3,1,1)   to  E(3,1,5)
 0 0 0 0 0 1  ; entries  E(3,1,6)   to  E(3,1,11)  (10a1,'—',5f9.1,' —')
 5 1 10 5 0   ; entries  E(3,1,12)  to  E(3,1,16)
 1980 10      ; entry    E(3,2,1)
```

EXAMPLE 4 FOR CAP

The calculating instruction file

```
FILE FOR TABLES
 1 3 2 -1
- - 1. 1.
@
'
'Table xx: Useful energy demands  [GWyr/yr]
'
'----------|-------------------------------------------|
'Demand    |                 Years                     |
sectors    = [1980.] + [1.]:cum - 1p0:
'----------|-------------------------------------------|
ELECTRIC   = ELECTR:fil  +  ELIND:fil
LIQUIDS    = LIQUIDS:fil
FEEDST     = FEEDST:fil
COAL       = CLSPU:fil
'----------|-------------------------------------------|
Final E.   = ELECTRIC: + LIQUIDS: + FEEDST: + COAL:
'----------|-------------------------------------------|
```

The tabular output file

```
Table xx: Useful energy demands  [GWyr/yr]
----------|-------------------------------------------|
Demand    |                 Years                     |
sectors   |   1980.0   1990.0   2000.0   2010.0   2020.0 |
----------|-------------------------------------------|
ELECTRIC  |    177.5    223.4    269.4    324.2    433.8 |
LIQUIDS   |    423.3    529.8    636.4    740.7    949.4 |
FEEDST    |    126.4    154.9    183.3    215.2    279.0 |
COAL      |     79.1     94.7    110.2    125.9    157.3 |
----------|-------------------------------------------|
Final E.  |    806.3   1002.8   1199.3   1406.0   1819.5 |
----------|-------------------------------------------|
```

23.4 Using CAP in Interactive Mode

Using CAP in interactive mode does not differ drastically from using it in batch mode. All input files besides the calculating instructions file on unit 5 are required as described above. Units 1 (output file), 5 (input file) and 6 (error messages) should be assigned to the terminal. To signal interactive mode '$$' has to be typed as first entry. The program then asks for the scaling factor (i.e. E(5,2,8)) and if the printing option should be set. In the following the same input as usually read from unit 5 is required, the results are written to the terminal in the format usually switched by setting E(5,2,2) to 3.

To get information about row or technology names type ')?' to enter the help mode where further information is supplied. To print interesting results (line by line) use the ')p' option and the last result is transferred to the output file on unit 12.

Part IV

The Formulation of MESSAGE III

Chapter 24

Introduction

This part of the document contains the mathematical formulation of MESSAGE III . The computer codes of the matrix generator produce equations according to this formulation, the input data determine the form these equations actually take.

In its general formulation MESSAGE III a dynamic linear programming model with a mixed integer option.[1] This implies that all relations that define the structure of a model are given as linear constraints between continuous variables. The variables of such a model are called "Columns", the equations "Rows". This nomenclature is derived from the usual notation used to write down linear models: in the shape of a matrix.

The variables (columns) of MESSAGE III be grouped into three categories:

1. Energy flow variables representing an annual energy flow quantity. The unit is usually MWyr for small regions and GWyr for bigger areas,
2. Power variables representing the production capacity of a technology (usual unit: MW or GW), and
3. Stock-piles representing the quantity of a fuel being cumulated at a certain point in time (usual unit: MWyr or GWyr).

The constraints (rows) generated by MESSAGE III be grouped into the following categories:

1. Energy flow balances modelling the flow of energy in the energy chain from resource extraction via conversion, transport, distribution up to final utilization,
2. sum or relational constraints limiting aggregate activities on an annual or overall basis, either absolute or in relation to other activities,
3. dynamic constraints setting a relation between the activities of two consecutive periods, and
4. counters that are only used for accounting purposes.

[1] Nonlinear constraints or a nonlinear objective function can only be defined for a specific problem, because the nonlinear functions and, if possible, their first derivatives have to be supplied as FORTRAN subroutines for MINOS.

This manual gives the mathematical formulation of MESSAGE III . It contains a formalized description of all types of variables and equations that the matrix generator generates. The reader of this paper is assumed to be familiar with the theory of linear and mixed integer programming; if he/she wants to apply the nonlinear options, some knowledge about MINOS and access to this code is essential.

Each of the building stones of MESSAGE III handled in a separate chapter, which is again subdivided into sections on columns and rows. The notation used for the variable and equation names is the same as in the MPS-file. Uppercase letters are used to indicate predefined identifiers, while lowercase letters represent characters that are chosen by the user or varied over a set of characters.

In order to keep the notation simple and the mathematical description as short as possible the more complex features are omitted from the description of the rows and described in an additional section (see chapter 32 beginning on page 141). Since practically all parameters of MESSAGE III be defined as time series (i.e. change over the planning horizon), the index for the period is often omitted in the formulation (e.g., for the efficiencies or the plant factors of conversion technologies).

Chapter 25

Conversion Technologies

25.1 Variables

Energy conversion technologies are modelled using two types of variables, that represent

- the amount of energy converted per year in a period (activity variables) and
- the capacity installed annually in a period (capacity variables).

25.1.1 Activities of Energy Conversion Technologies

zsvd.elt

where

z	is the level identifier of the main output of the technology. $z = U$ identifies the end-use level. This level is handled differently to all other levels: It has to be the demand level and technologies with the main output on this level are defined without load regions.
s	is the main energy input of the technology (supply). If the technology has no input s is set to "." (e.g., solar technologies),
v	additional identifier of the conversion technology (used to distinguish technologies with the same input and output),
d	is the main energy output of the technology (demand),
e	is the level of reduction of demand due to own-price elasticities of demands (does only occur on the demand level; otherwise or if this demand has no elasticities $e =$ "."),
l	identifies the load region, $l \in \{1, 2, 3, ...\}$ or $l =$ ".", if the technology is not modelled with load regions, and
t	identifies the period, $t \in \{a, b, c, ...\}$.

The activity variable of an energy conversion technology is an energy flow variable. It represents the annual consumption of this technology of the main input per period. If a technology has no input, the variable represents the annual production of the main output.

If the level of the main output is *not* U and at least one of the energy carriers consumed or supplied is defined with load regions the technology is defined with load regions. In

this case the activity variables are generated separately for each load region, which is indicated by the additional identifier l in position 7. However, this can be changed by fixing the production of the technology over the load regions to a predefined pattern (see section 32.4): one variable is generated for all load regions, the distribution to the load regions is given by the definition of the user (e.g., production pattern of solar power-plants).

If the model is formulated with demand elasticities (see section 32.9), the activity variables of technologies with a demand as main output that is defined with elasticity are generated for each elasticity class (identifier e in position 6).

25.1.2 Capacities of Energy Conversion Technologies

$$Yzsvd..t,$$

where

Y	is the identifier for capacity variables.
z	identifies the level on that the main energy output of the technology is defined,
s	is the identifier of the main energy input of the technology,
v	additional identifier of the conversion technology,
d	is the identifier of the main energy output of the technology, and
t	is the period in that the capacity goes into operation.

The capacity variables are power variables. Technologies can be modelled without capacity variables. In this case no capacity constraints and no dynamic constraints on construction can be included in the model. Capacity variables of energy conversion technologies can be defined as integer variables, if the solution algorithm has a mixed integer option.

If a capacity variable is continuous it represents the annual new installations of the technology in period t, if it is integer it represents either the annual number of installations of a certain size or the number of installations of $1/\Delta t$ times the unit size (depending on the definition; Δt is the length of period t in years).

The capacity is defined in relation to the main output of the technology.

25.2 Constraints

The rows used to model energy conversion technologies limit

- the utilization of a technology in relation to the capacity actually installed (capacity constraint) and
- the activity or construction of a technology in a period in relation to the same variable in the previous period (dynamic constraints).

25.2.1 Capacity Constraints

$$Czsvd.lt,$$

where

C	is the identifier for capacity constraints,
z	identifies the level on that the main energy output of the technology is defined,
s	is the identifier of the main energy input of the technology,
v	additional identifier of the conversion technology,
d	is the identifier of the main energy output of the technology,
l	identifies the load region, $l \in \{1, 2, 3, ...\}$ or $l =$ ".", if the technology is not modelled with load regions, and
t	is the period in that the capacity goes into operation.

For all conversion technologies modelled with capacity variables the capacity constraints will be generated automatically. If the activity variables exist for each load region separately there will be one capacity constraint per load region (see also section 32.4). If the technology is an end-use technology the sum over the elasticity classes will be included in the capacity constraint.

Additionally the activity variables of different technologies can be linked to the same capacity variable, which allows to leave the choice of the activity variable used with a given capacity to the optimization (see section 32.6).

Technologies without Load Regions

For technologies without load regions (i.e. technologies, where no input or output is modelled with load regions) the production is related to the total installed capacity by the plant factor. For these technologies the plant factor has to be given as the fraction they actually operate per year. All end-use technologies (technologies with main output level "U") are modelled in this way.

$$\epsilon_{svd} \times zsvd...t - \sum_{\tau=t-\tau_{svd}}^{min(t,\kappa_{svd})} \Delta(\tau - 1) \times \pi_{svd} \times f_i \times Yzsvd..\tau \leq hc^t_{svd} \times \pi_{svd} .$$

Technologies with Load Regions and "Free" Production Pattern

If a technology has at least one input or output with load regions, the activity variables and capacity constraints will per default be generated separately for each load region. This can be changed by defining the production pattern over the load regions. If the production pattern remains free, the production in each load region is limited in relation to the installed capacity separately for each load region, the capacity is determined by the activity in the load region with the highest requirements. The plant factor has to be given as the fraction the system operates in peak operation mode (in general this is the availability factor).

Maintenance times or minimum operation times could be included by using additional

relations, if required (see section 30).

$$\frac{\epsilon_{svd}}{\lambda_l} \times zsvd..lt - \sum_{\tau=t-\tau_{svd}}^{min(t,\kappa_{svd})} \Delta(\tau-1) \times \pi_{svd} \times f_i \times Yzsvd..\tau \leq hc^t_{svd} \times \pi_{svd} .$$

Technologies with Load Regions and "Fixed" Production Pattern

If a technology has at least one input or output with load regions and the production pattern over the load regions is predefined only one activity variable and one capacity constraint is generated per period. The plant factor has, like for technologies with load regions and free production pattern, to be given for the load region with the highest capacity utilization (i.e. the highest power requirement). The capacity constraint is generated for only this load region.

$$\frac{\epsilon_{svd} \times \pi(l_m, svd)}{\lambda_{l_m}} \times zsvd...t -$$

$$\sum_{\tau=t-\tau_{svd}}^{min(t,\kappa_{svd})} \Delta(\tau-1) \times \pi_{svd} \times f_i \times Yzsvd..\tau \leq hc^t_{svd} \times \pi_{svd} .$$

Technologies with Varying Inputs and Outputs

Many types of energy conversion technologies do not have fix relations between their inputs and outputs. MESSAGE has the option to link several activity variables of conversion technologies into one capacity constraint. For the additional activities linked to a capacity variable a coefficient defines the maximum power available in relation to one power unit of the main activity.

In the following this constraint is only described for technologies without load regions; the other types are constructed in analogy (see also section 32.6).

$$\sum_{\sigma v'\delta} rel^{svd}_{\sigma v'\delta} \times \epsilon_{\sigma v'\delta} \times z\sigma v'\delta...t -$$

$$\sum_{\tau=t-\tau_{svd}}^{min(t,\kappa_{svd})} \Delta(\tau-1) \times \pi_{svd} \times f_i \times Yzsvd..\tau \leq hc^t_{svd} \times \pi_{svd} ,$$

The following notation is used in the above equations:

$zsvd..lt$	is the activity of conversion technology v in period t and, if defined so, load region l (see section 25.1.1),
$Yzsvd..t$	is the capacity variable of conversion technology v (see section 25.1.2).
ϵ_{svd}	is the efficiency of technology v in converting the main energy input, s, into the main energy output, d,
κ_{svd}	is the last period in that technology v can be constructed,

π_{svd} is the "plant factor" of technology v, having different meaning depending on the type of capacity equation applied,
$\Delta\tau$ is the length of period τ in years,
τ_{svd} is the plant life of technology v in periods,
hc^t_{svd} represents the installations built before the time horizon under consideration, that are still in operation in the first year of period t,
f_i is 1. if the capacity variable is continuous, and represents the minimum installed capacity per year (unit size) if the variable is integer,
l_m is the load region with maximum capacity use if the production pattern over the year is fixed,
$\pi(l_m, svd)$ is the share of output in the load region with maximum production,
$rel^{svd}_{\sigma v'\delta}$ is the relative capacity of main output of technology (or operation mode) svd to the capacity of main output of the alternative technology (or operation mode) $\sigma v'\delta$,
λ_l is the length of load region l as fraction of the year, and
λ_{l_m} is the length of load region l_m, the load region with maximum capacity requirements, as fraction of the year.

25.2.2 Upper Dynamic Constraints on Construction Variables

MYzsvd.t

The dynamic capacity constraints relate the amount of annual new installations of a technology in a period to the annual construction during the previous period.

$$Yzsvd..t \; - \; \gamma y^o_{svd,t} \; \times \; Yzsvd..(t-1) \; \leq \; gy^o_{svd,t},$$

where

$\gamma y^o_{svd,t}$ is the maximum growth rate per period for the construction of technology v,
$gy^o_{svd,t}$ is the initial size (increment) that can be given for the introduction of new technologies,
$Yzsvd..t$ is the annual new installation of technology v in period t.

25.2.3 Lower Dynamic Constraints on Construction Variables

LYzsvd.t

$$Yzsvd..t \; - \; \gamma y_{svd,t} \; \times \; Yzsvd..(t-1) \; \geq \; - \; gy_{svd,t},$$

where

$\gamma y_{svd,t}$ is the minimum growth rate per period for the construction of technology v,
$gy_{svd,t}$ is the "last" size (decrement) allowing technologies to go out of the market, and
$Yzsvd..t$ is the annual new installation of technology v in period t.

25.2.4 Upper Dynamic Constraints on Activity Variables

Mzsvd..t

The dynamic production constraints relate the production of a technology in one period to the production in the previous period. If the technology is defined with load regions, the sum over the load regions is included in the constraint.

$$\sum_{l} \epsilon_{svd} \times \left[zsvd..lt - \gamma a^{o}_{svd,t} \times zsvd..l(t-1) \right] \leq ga^{o}_{svd,t},$$

where

$\gamma a^{o}_{svd,t}$ and $ga^{o}_{svd,t}$ are the maximum growth rate and increment as described in section 25.2.2 (the increment is to be given in units of main output), and

$zsvd..lt$ is the activity of technology v in load region l.

If demand elasticities are modelled, the required sums are included for end-use technologies.

25.2.5 Lower Dynamic Constraints on Activity Variables

Lzsvd..t

$$\sum_{l} \epsilon_{svd} \times [zsvd..lt - \gamma a_{svd,t} \times zsvd..l(t-1)] \geq -ga_{svd,t},$$

where

$\gamma a_{svd,t}$ and $ga_{svd,t}$ are the maximum growth rate and increment as described in section 25.2.3, and

$zsvd..lt$ is the activity of technology v in load region l.

Chapter 26

Domestic Resources

26.1 Variables

Extraction of domestic resources is modelled by variables that represent the quantity extracted per year in a period. A subdivision into cost categories (which are called "grades" in the model) and further into elasticity classes can be modelled.

26.1.1 Resource Extraction Variables

Rzrgp..t,

where

R	identifies resource extraction variables,
z	is the level on that the resource is defined (usually = R),
r	is the identifier of the resource being extracted,
g	is the grade (also called cost category) of resource r, $g \in \{a, b, c, ...\}$.
p	is the class of supply elasticity, which is defined for the resource and grade, or "." , if no elasticity is defined for this resource and grade, and
t	identifies the period.

The resource variables are energy flow variables and represent the annual rate of extraction of resource r. If several grades are defined, one variable per grade is generated (identifier g in position 4). Supply elasticities can be defined for resource extraction as described in section 32.10; in this case one variable per elasticity class (identifier p in position 5) is generated.

26.2 Constraints

The overall availability of a resource is limited in the availability constraint per grade, annual resource consumption can be constrained per grade (sum of the elasticity classes) and total. Additionally resource depletion and dynamic resource extraction constraints can be modelled.

26.2.1 Resource Availability per Grade

RRrg.g..

Limits the domestic resource available from one cost category (grade) over the whole time horizon. Total availability of a resource is defined as the sum over the grades.

$$\sum_p \sum_t \Delta t \times RRrgp..t \leq Rrg - \Delta t_0 R_{rg,0} ,$$

where

Rrg	is the total amount of resource r, cost category g, that is available for extraction,
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t,
Δt	is the length of period t.
Δt_0	is the number of years between the base year and the first model year, and
$R_{rg,0}$	is the extraction of resource r, grade g in the base year.

26.2.2 Maximum Annual Resource Extraction

RRr....t

Limits the domestic resources available annually per period over all cost categories.

$$\sum_g \sum_p RRrgp..t \leq Rrt ,$$

where

$Rrgt$	is the maximum amount of resource r, grade g, that can be extracted per year of period t, and
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t.

26.2.3 Resource Depletion Constraints

RRrg.d.t

The extraction of a resource in a period can be constrained in relation to the total amount still existing in that period. For reasons of computerization these constraints can also be generated for imports and exports, although they do not have any relevance there (they could, e.g., be used for specific scenarios in order to stabilize the solution).

$$\Delta t \sum_p RRrgp..t \leq \delta_{rg}^t \left[Rrg - \Delta t_0 R_{rg,0} - \sum_{\tau=1}^{t-1} \Delta\tau \times RRrgp..\tau \right] ,$$

where

Rrg	is the total amount of resource r, cost category g, that is available for extraction,
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t,
δ^t_{rg}	is the maximum fraction of resource r, cost category g, that can be extracted in period t,
Rrg	is the total amount available in the base year,
Δt	is the length of period t in years,
Δt_0	is the number of years between the base year and the first model year, and
$R_{rg,0}$	is the extraction of resource r, grade g in the base year.

26.2.4 Maximum Annual Resource Extraction per Grade

$RRrg.a.t$

Limits the domestic resources available from one cost category per year.

$$\sum_{p} RRrgp..t \leq Rrgt\,.$$

where

Rrg	is the total amount of resource r, cost category g, that is available for extraction, and
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t.

26.2.5 Upper Dynamic Resource Extraction Constraints

$MRRr...t$

The annual extraction level of a resource in a period can be related to the previous one by a growth parameter and an increment of extraction capacity resulting in upper dynamic extraction constraints. For the first period the extraction is related to the activity in the baseyear.

$$\sum_{g,p} RRrgp..t \; - \; \gamma^o_{rt} \sum_{g,p} RRrgp..(t-1) \; \leq \; g^o_{rt}\,,$$

where

γ^o_{rt}	is the maximum growth of extraction of resource r between period $t-1$ and t,
g^o_{rt}	is the initial size (increment) of extraction of resource r in period t, and
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t.

26.2.6 Lower Dynamic Resource Extraction Constraints

LRRr...t

The annual extraction level of a resource in a period can also be related to the previous one by a decrease parameter and a decrement resulting in lower dynamic extraction constraints. For the first period the extraction is related to the activity in the baseyear.

$$\sum_{g,p} RRrgp..t \; - \; \gamma_{rt} \sum_{g,p} RRrgp..(t-1) \; \geq \; - \; g_{rt} \; ,$$

where

γ_{rt}	is the maximum decrease of extraction of resource r between period $t-1$ and t,
g_{rt}	is the "last" size (decrement) of extraction of resource r in period t, and
$RRrgp..t$	is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t.

26.2.7 Dynamic Extraction Constraints per Grade

MRRrg..t, and
LRRrg..t

The same kind of relations as described in sections 26.2.5 and 26.2.6 can be defined per grade of the resource.

Chapter 27

Imports and Exports

27.1 Variables

Imports and exports are modelled by variables that represent the quantity imported per year in a period. A subdivision into countries and further into elasticity classes can be modelled.

27.1.1 Import Variables

Izscp.lt,

where

I	identifies import variables,
z	is the level on that the imported energy form is defined (usually primary energy and secondary energy),
s	identifies the imported energy carrier,
c	is the identifier of the country or region the imports come from,
p	is the class of supply elasticity, which is defined for the energy carrier and country, or ".", if no elasticity is defined for this energy carrier and country,
l	is the load region identifier if *s* is modelled with load regions, otherwise ".", and
t	identifies the period.

The import variables are energy flow variables and represent the annual import of the identified energy carrier from the country or region given. If supply elasticities are defined for the import of this energy carrier and country one variable per elasticity class (identifier *p* in position 5) is generated.

27.1.2 Export Variables

Ezrcp.lt,

where

E	is the identifier for export variables, and
z	is the level on that the exported energy form is defined (usually primary energy and secondary energy),

s	identifies the exported energy carrier,
c	is the identifier of the country or region the exports go to,
p	is the class of supply elasticity, which is defined for the energy carrier and country, or ".", if no elasticity is defined for this energy carrier and country,
l	is the load region identifier if s is modelled with load regions, otherwise ".", and
t	identifies the period.

The export variables are energy flow variables and represent the annual export of the identified energy carrier to the country or region given. If supply elasticities are defined for the export of this energy carrier and country one variable per elasticity class (identifier p in position 5) is generated.

27.2 Constraints

27.2.1 Imports per Country

$Izrc.g..$

Limits the imports of a fuel from a specific country c over the whole horizon.

$$\sum_{p} \sum_{t} \Delta t \times Izrcp..t \leq Irc\,,$$

where

Irc	is the total import limit for r from country c,
$Izrcp..t$	is the annual import of r from country c, elasticity class p in period t, and
Δt	is the length of period t in years.

27.2.2 Maximum Annual Imports

$Izr....t$

Limits the annual imports of a fuel from all countries per period.

$$\sum_{c} \sum_{p} Izrcp..t \leq Irt\,,$$

where

Irt	is the annual import limit for r in period t, and
$Izrcp..t$	is the annual import of r from country c, elasticity class p in period t.

27.2.3 Maximum Annual Imports per Country

$Izrc.a.t$

Limits the imports from one country per year.

$$\sum_{p} Izrcp..t \leq Irct\,,$$

where

$Irct$	is the limit on the annual imports from country c, period t of fuel r, and
$Izrcp..t$	is the annual import of r from country c, elasticity class p in period t.

27.2.4 Upper Dynamic Import Constraints

$MIzr...t$

The annual import level of a fuel in a period can, like the resource extraction, be related to the previous one by a growth parameter and an increment resulting in upper dynamic constraints.

$$\sum_{c,p} Izrcp..t \; - \; \gamma^{o}_{rt} \sum_{c,p} Izrcp..(t-1) \; \leq \; g^{o}_{rt}\,,$$

where

$Izrcp..t$	is the annual import of r from country c, elasticity class p in period t,
γ^{o}_{rt}	is the maximum increase of import of r between period $t-1$ and t, and
g^{o}_{rt}	is the initial size (increment) of import of r in period t.

27.2.5 Lower Dynamic Import Constraints

$LIzr...t$

The annual import level of a fuel in a period can also be related to the previous one by a decrease parameter and a decrement resulting in lower dynamic import constraints.

$$\sum_{c,p} Izrcp..t \; - \; \gamma_{rt} \sum_{c,p} Izrcp..(t-1) \; \geq \; - \, g_{rt}\,,$$

where

$Izrcp..t$	is the annual import of r from country c, elasticity class p in period t,
γ_{rt}	is the maximum decrease of import of r between period $t-1$ and t, and
g_{rt}	is the "last" size (decrement) of import of r in period t.

27.2.6 Dynamic Import Constraints per Country

$MIzrc..t$ and

$LIzrc..t$

The same kind of relations can be defined per country from that the fuel is imported.

27.2.7 Constraints on Exports

The exports of fuels can principally be limited in the same way as the imports. In the identifiers of the variables and constraints the "I" is substituted by an "E".

Chapter 28

Energy Flows

28.1 Constraints

Energy flows are modelled solely by linking the activity variables of the different conversion technologies and the resource extraction, import and export variables in balance constraints. These constraints ensure that only the amounts of energy available are consumed. There are no further variables required to model energy flows.

Energy demands are also modelled as part of a balance constraint: it is the right hand side and defines the amount to be supplied by the technologies in this constraint.

The following description of the energy flow constraints in MESSAGE is given for the following set of level identifiers:

U Useful energy (demand level),

F Final energy (after distribution),

T Final energy (after transmission),

X Secondary energy,

A Primary energy, and

R Energy resources.

The identifier of the demand level (U) which gives it a special meaning (see section 25.1.1) and imports and exports are given for primary energy. Clearly any other combination of technologies is also possible.

28.1.1 Demand Constraints

$$Ud.....t$$

Out of the predefined levels each one can be chosen as demand level. However, level "U" has a special feature. This is related to the fact that useful energy is usually produced on-site, e.g., space heat is produced by a central heating system, and the load variations over the year are all covered by this one system. Thus, an allocation of production technologies to the different areas of the load curve, like the model would set it up

according to the relation between investment and operating costs would ignore the fact that these systems are not located in the same place and are not connected to each other. MESSAGE represents the end-use technologies by one variable per period that produces the required useful energy in the load pattern needed and requires the inputs in the same pattern. For special technologies like, e.g., night storage heating systems, this pattern can be changed to represent the internal storage capability of the system. This representation of end-use technologies has the advantage of reducing the size of the model, because the demand constraints, the activity variables and the capacity constraints of the end-use technologies do not have to be generated for each load region. If another level is chosen as demand level or the demand level is not named "U", all demand constraints for energy carriers that are modelled with load regions are generated for each load region. The general form of the demand constraints is

$$\sum_{svd} \epsilon_{svd} \times \sum_{e=0}^{e_d} k_e \times Usvd.e.t + \sum_{sv\delta} \beta^d_{sv\delta} \times \sum_{e=0}^{e_\delta} k_e \times Usv\delta ue.t \geq Ud.t \,,$$

where

$Ud.t$	is the annual demand for d in period t,
$Usvd.e.t$	is the activity of end-use technology v in period t, elasticity class e and period t (see section 25.1.1),
ϵ_{svd}	is the efficiency of end-use technology v in converting s to d,
$\beta^d_{sv\delta}$	is the efficiency of end-use technology v in producing by-product d from s (δ is the main output of the technology),
e_d	is the number of steps of demand reduction modelled for own-price elasticities of demand d, and
k_e	is the factor giving the relation of total demand for d to the demand reduced to level e due to the demand elasticity. ($k_e \times Usvd.e.t = Usvd.0.t$, $k_0 = 1$, k_e is increasing monotonously.)

28.1.2 Distribution Balance

$Fs....lt$

This constraint, the final energy balance, matches the use of final energy needed in the end-use technologies and the deliveries of the distribution systems. It is generated for each load region, if energy form s is modelled with load regions.

$$\sum_{svs} \epsilon_{svs} \times Fsvs..lt - \sum_{svd} \eta_{d,l} \times \sum_{e=0}^{e_d} Usvd.e.t -$$

$$\sum_{\sigma vd} \beta^s_{\sigma vd} \times \eta_{d,l} \times \sum_{e=0}^{e_d} U\sigma vd.e.t \geq 0 \,,$$

where

$Fsvs..lt$ is the activity of the distribution technology in load region l and period t (see section 25.1.1),
ϵ_{svs} is the efficiency of technology v in distributing s,
$Usvd.e.t$ is the activity of end-use technology v in period t and elasticity class e,
$\beta^s_{\sigma vd}$ is the use of fuel s relative to fuel σ (the main input) by technology v, and
$\eta_{d,l}$ is the fraction of demand for d occurring in load region l.

28.1.3 Transmission or Transportation Balance

$Ts....lt$

This constraint gives the simplest form of an energy balance equation of MESSAGE. It matches the output of transmission to the requirements of distribution systems. The difference to other levels (F, X, A) is not built-in, but emerges from the simplicity of energy transportation (i.e., transportation technologies do usually not have by-products and only one input). Also big industrial consumers that are directly connected to the transmission system would have to be included in this constraint. Like level F it does usually exist for all load regions if they are defined for the fuel.

$$\sum_{svs} \epsilon_{svs} \times Tsvs..lt - \sum_{svs} Fsvs..lt \geq 0 .$$

where
$Tsvs..lt$ is the activity of the transportation technology v (see section 25.1.1), and

all the other entries to the equation are the same as in section 28.1.2.

28.1.4 Central Conversion Balance

$Xs....lt$

In principle the secondary energy balance is built up in the same way as the two previous ones (sections 28.1.2 and 28.1.3). It matches the production of central conversion technologies to the requirements of the transmission systems. Secondary energy imports and exports of secondary energy are usually assigned to level X.

$$\sum_{rvs} \epsilon_{rvs} \times Xrvs..lt + \sum_{rv\sigma} \beta^s_{rv\sigma} \times Xrv\sigma..lt - \sum_{svs} Tsvs..lt +$$

$$\sum_{c,p} IXscp.lt - \sum_{c,p} EXscp.lt \geq 0$$

where

$Xrvs..lt$ is the activity of central conversion technology v in load region l and period t (see section 25.1.1); if the secondary energy form s is not defined with load regions (i.e. $l =$ ".") and the activity of technology v exists for each load region, this equation will contain the sum of the activity variables of technology v over the load regions.

ϵ_{rvs} is the efficiency of technology v in converting energy carrier r into secondary energy form s,

$\beta^s_{rv\sigma}$ is the efficiency of technology v in converting energy carrier r into the by-product s of technology v,

$Tsvs..lt$ is explained in section 28.1.3, and

$IXscp.lt$ and $EXscp.lt$ are the import and export variables explained in sections 27.1.1 and 27.1.2, respectively.

28.1.5 Resource Extraction, Export and Import Balance

$Ar.....t$

This equation matches production and import of primary energy to the requirements of central conversion, transport and for export. In the general case primary energy does not have load regions. Some technologies, like, e.g., nuclear reactors need inventories of primary energy and also leave a last core that is available at the end of the lifetime. It may be necessary to model by-products of extraction technologies, for instance the availability of associated gas at oil production sites.

$$\sum_{rvr} \epsilon_{rvr} \times Arvr...t - \sum_{l} \left[\sum_{rvs} Xrvs..lt + \sum_{\rho vs} \beta^r_{\rho vs} \times X\rho vs..lt \right] + \sum_{c,p} IArcp..t -$$

$$\sum_{c,p} EArcp..t + \sum_{fvs} \left[\frac{\Delta(t-\tau_{fvs})}{\Delta t} \times \rho(fvs,r) \times YXfvs..(t-\tau_{fvs}) - \right.$$

$$\left. \frac{\Delta(t+1)}{\Delta t} \times \iota(fvs,r) \times YXfvs..(t+1) \right] \geq 0\,,$$

where

$Arvr...t$ is the activity of technology v extracting resource r,

ϵ_{rvr} is the efficiency of technology v in extracting fuel r (this is usually 1.),

$\beta^r_{\rho vs}$ is the efficiency of technology v in producing secondary energy form s from the by-input ρ,

$IArcp..t$ and $EArcp..t$ are the import and export variables described in section 27.1.1 and 27.1.2, respectively,

τ_{fvs} is the plant life of technology v in periods (depending on the lengths of the periods covered),

$YXfvs..t$ is the annual new installation of technology v in period t (see section 25.1.2),

$\iota(fvs,r)$ is the amount of fuel r that is needed when technology v goes into operation (usually this is the first core of a reactor). It has to be available in the period before technology v goes into operation, the normal unit is kWyr/kW,

$\rho(fvs, r)$ is the amount of fuel r that becomes available after technology v goes out of operation (for a reactor this is the last core that goes to reprocessing). The unit is the same as for $\iota(fvs, r)$, and

Δt is the length of period t in years.

28.1.6 Resource Consumption

Rr.....t

The resources produced by the extraction technologies in a period can come from different cost categories (also called grades), which can, e.g., represent the different effort to reach certain resources. Short-term variations in price due to steeply increasing demand can be represented by an elasticity approach (see section 32.10).

$$\sum_{g,p} RRrgp..t \; - \; \sum_{rvr} Arvr...t \; \geq \; 0 \, ,$$

where

RRrgp..t is the annual extraction of resource r, cost category (grade) g and elasticity class p in period t, and

Arvr...t is the activity of extraction technology v in period t (as described in section 25.1.1).

Chapter 29

Stock-piles

29.1 Variables

Generally MESSAGE III does not generate any variables related to an energy carrier alone. However, in the case of man-made fuels, that are accumulated over time, a variable that shifts the quantities to be transferred from one period to the other is necessary.

29.1.1 Stock-pile Variables

$$Qfb....t,$$

where

Q	identifies stock-pile variables,
f	identifies the fuel with stock-pile,
b	distinguishes the variable from the equation, and
t	is the period identifier.

The stock-pile variables represent the amount of fuel f that is transferred from period t into period $t + 1$. Note that these variables do not represent annual quantities, they refer to the period as a whole. These variables are a special type of storage, that just transfers the quantity of an energy carrier available in one period into the next period. Stock-piles are defined as a separate level. For all other energy carriers any overproduction that occurs in a period is lost.

29.2 Constraints

29.2.1 Stock-piling Constraints

$$Qf.....t$$

Q is a special level on that energy forms can be defined that are accumulated over time and consumed in later periods. One example is the accumulation of plutonium and later use in fast breeder reactors.

The general form of this constraint is:

$$Qfb....t \; - \; Qfb....(t-1) \; + \; \sum_{v} \left[\sum_{l} \Delta t \; \times \; (\; zfvd..lt \; + \; \beta^{f}_{\phi vd} \; \times \; z\phi vd..lt \; -$$

$$\epsilon_{svf} \; \times \; zsvfu.lt \; - \; \beta^{f}_{sv\phi} \; \times \; zsv\phi..lt \;) \; + \; \Delta t \; \times \; \iota(svd, f) \; \times \; Yzsvd..(t+1) \; -$$

$$\Delta(t - \tau_{svd} - 1) \; \times \; \rho(svd, f) \; \times \; Yzsvd..(t - \tau_{svd}) \;] \; = \; 0 \, ,$$

where

f	is the identifier of the man-made fuel (e.g. plutonium, U_{233}),
τ_{svd}	is the plant life of technology v in periods,
$\iota(svd, f)$	is the "first inventory" of technology v of f (relative to capacity of main output),
$\rho(svd, f)$	is the "last core" of f in technology v, see also section 28.1.5,
Δt	is the length of period t in years,
$zfvd..lt$	is the annual input of technology v of fuel f in load region l and period t (l is "." if v does not have load regions), and
$Yzfvd..t$	is the annual new installation of technology v in period t.

Chapter 30

User-defined Relations

30.1 Constraints

The user-defined relations allow the user to construct constraints that are not included in the basic set of constraints. For each technology the user can specify coefficients with that either the production variables (see section25.1.1), the annual new installation variables (see section 25.1.2) or the total capacity in a year (like it is used in the capacity constraints, see section 25.2.1) can be included in the relation. The relations can be defined with and without load regions, have a lower, upper or fix right hand side or remain free (non-binding) and be related to an entry in the objective function, i.e., all entries to this relation are also entered to the objective function with the appropriate discount factor. There are two types of user-defined constraints, for which the entries to the objective function–without discounting–are summed up under the cost accounting rows $CAR1$ and $CAR2$ (see chapter 31).

The formulation of the user-defined relations is given for relations, that are related to the main output of the technologies. It is also possible (e.g., for emissions) to relate the constraint to the main input of the technology, i.e. the amount of fuel used. In this case the efficiencies (ϵ) would be omitted from the formulation.

30.1.1 Relation without Load Regions

$Nm.....t$ or $Pm.....t$

Relations without load regions just sum up the activities (multiplied with the given coefficients) of all variables defined to be in this constraint. If a technology has load regions, the activity variables for all load regions of this technology are included. If the total capacity of a technology is included, all new capacities from previous periods still operating are included, if new capacities are included, the annual new installation of the current period is taken.

$$\sum_{svd} \left[ro^{mt}_{svd} \times \sum_{e=0}^{e_d} Usvd.e.t \times \epsilon_{svd} + \sum_{\tau=t-ip}^{t} rc^{mt}_{svd} \times YUsvd..\tau \right] +$$

$$\sum_{rvs} \left[ro^{mlt}_{rvs} \times \sum_{l} zrvs..lt \times \epsilon_{rvs} + ro^{mt}_{rvs} \times zrvs...t \times \epsilon_{rvs} + \right.$$

$$\left.\sum_{\tau=t-ip}^{t} rc_{rvs}^{mt} \times Yzrvs..\tau\right] \left\{\begin{array}{l} free \\ \geq rhs_m^t \\ = rhs_m^t \\ \leq rhs_m^t \end{array}\right. ,$$

where

$Usvd.e.t$ and $YUsvd..t$ are the activity and capacity variables of the end-use technologies,

$zrvs..lt$, $zrvs...t$ and $Yzrvs..t$ are the activity variables of technologies with and without load regions and the capacity variables of the technologies,

ϵ_{rvs} and ϵ_{svd} are the efficiencies of the technologies; they are included by the code,

ro_{svd}^{mt} is the relative factor per unit of output of technology v (coefficient) for relational constraint m,

rc_{svd}^{mt} is the same per unit of new built capacity,

ro_{rvs}^{mlt} is the relative factor per unit of output of technology v (coefficient) for relational constraint m, load region l,

rc_{rvs}^{mlt} is the same per unit of new built capacity,

tl is 1 for relations to construction and $\Delta\tau$ for relations to total capacity,

ip is 1 for accounting during construction and the plant life on periods for accounting of total capacity, and

rhs_m^t is the right hand side of the constraint.

30.1.2 Relation with Load Regions

$Nm....lt$ or $Pm....lt$

The user defined relations can be defined with load regions. Then all entries of activities of technologies with load regions are divided by the length of the according load region resulting in a representation of the utilized power.

$$\sum_{svd}\left[ro_{svd}^{mlt} \times \sum_{e=0}^{e_d} Usvd.e.t \times \epsilon_{svd} + \sum_{\tau=t-ip}^{t} rc_{svd}^{mlt} \times YUsvd..\tau\right] +$$

$$\sum_{rvs}\left[\frac{ro_{rvs}^{mlt}}{\lambda_l} \times zrvs..lt \times \epsilon_{rvs} + ro_{rvs}^{mt} \times zrvs...t \times \epsilon_{rvs} +\right.$$

$$\left.\sum_{\tau=t-ip}^{t} rc_{rvs}^{mlt} \times tl \times Yzrvs..\tau\right] \left\{\begin{array}{l} free \\ \geq rhs_{ml}^t \\ = rhs_{ml}^t \\ < rhs_{ml}^t \end{array}\right. ,$$

where

$Usvd.e.t$ and $YUsvd..t$ are the activity and capacity variables of the end-use technologies,

$zrvs..lt$, $zrvs...t$ and $Yzrvs..t$ are the activity variables of technologies with and without load regions and the capacity variables of the technologies,

ϵ_{rvs} and ϵ_{svd} are the efficiencies of the technologies; they are included by the code,

ro_{svd}^{mlt} is the relative factor per unit of utilized capacity of technology v (coefficient) for relational constraint m in load region l, period t (this constraint is adapted to represent the utilized power, as stated above),

rc_{svd}^{mlt} is the same per unit of new built or installed capacity,

ro_{rvs}^{mlt} is the relative factor per unit of output of technology v (coefficient) for relational constraint m, load region l,

rc_{rvs}^{mlt} is the same per unit of new built capacity,

tl is 1 for relations to construction and $\Delta\tau$ for relations to total capacity,

ip is 1 for accounting during construction and the plant life on periods for accounting of total capacity, and

rhs_{ml}^{t} and is the right hand side of the constraint.

30.1.3 Construction of Relations between Periods

$Nm.....t$ or $Pm.....t$

The change of activities over time can either be limited or included in the objective by constructing relations between periods: The relations expresses the difference between the annual activity in a period and the following period. This difference can either be limited or included in the objective function.

$$\sum_{svd} \left[ro_{svd}^{mt} \times \sum_{e=0}^{e_d} Usvd.e.t \times \epsilon_{svd} - ro_{svd}^{m(t-1)} \times \right.$$

$$\left. \sum_{e=0}^{e_d} Usvd.e.(t-1) \times \epsilon_{svd} \right] + \sum_{rvs} \left[ro_{rvs}^{mt} \times zrvs...t \times \epsilon_{rvs} - ro_{rvs}^{m(t-1)} \times \right.$$

$$\left. zrvs...(t-1) \times \epsilon_{rvs} \right] + \sum_{rvs} \left[ro_{rvs}^{mlt} \times \sum_{l} zrvs..lt \times \epsilon_{rvs} - ro_{rvs}^{ml(t-1)} \times \right.$$

$$\left. \sum_{l} zrvs..l(t-1) \times \epsilon_{rvs} \right] \begin{cases} free \\ \geq rhs_m^t \\ = rhs_m^t \\ < rhs_m^t \end{cases},$$

where

$Usvd.e.t$	is the activity variable of the end-use technologies,
$zrvs..lt$	and $zrvs...t$ are the activity variables of technologies with and without load regions,
ϵ_{rvs}	and ϵ_{svd} are the efficiencies of the technologies; they are included by the code,
ro^{mt}_{svd}	is the relative factor per unit of output of technology v (coefficient) for relational constraint m, period t,
ro^{mlt}_{rvs}	is the relative factor per unit of output of technology v (coefficient) for relational constraint m, load region l, and
rhs^t_m	and is the right hand side of the constraint.

For this type of constraints only the ro-coefficients have to be supplied by the user, the rest is included by the model. It can be defined with and without load regions.

30.1.4 Special Handling of Demand Elasticities

$Pm.....t$

The second type of user defined relations differs from the first one in the fact that the activity of the end-use technologies is multiplied by k_e and therefore represents the production without reduction by demand elasticities.

Thus this constraint can be applied to force a certain reduction level due to the elasticities reached in one period to be also reached in the following period, allowing the interpretation of the reduction as investments in saving. The coefficient of the technologies supplying a demand have to be the inverse of this demand in the current period, then. This constraint has the following form:

$$\sum_{sv} \sum_{e=0}^{e_d} Usvd.e.t \times \epsilon_{svd} \times \frac{\kappa_e}{Ud.t} -$$

$$\sum_{sv} \sum_{e=0}^{e_d} Usvd.e.(t-1) \times \epsilon_{svd} \times \frac{\kappa_e}{Ud.(t-1)} \leq 0 ,$$

where

the coefficients are supplied by MESSAGE. The user can additionally define multiplicative factors for these coefficients.

Chapter 31

Objective and Cost Counters

31.1 Constraints

31.1.1 Cost Accounting Rows

The different types of costs (i.e. entries for the objective function) can be accumulated over all technologies in built-in accounting rows. These rows can be generated per period or for the whole horizon and contain the sum of the undiscounted costs. They can also be limited. The implemented types are:

$CCUR$ – fix (related to the installed capacity) and variable (related to the production) operation and maintenance costs,
$CCAP$ – investment costs; if the investments of a technology are distributed over the previous periods, also the entries to this accounting rows are distributed (if the capital costs are levellized, the total payments in a period can be taken from $CINV$; $CCAP$ shows the share of investments in the according period, then),
$CRES$ – domestic fuel costs,
$CAR1$ – costs related to the user defined relations of type 1 (see section 30),
$CAR2$ – costs related to the user defined relations of type 2 (see section 30),
$CRED$ – costs for reducing demands due to demand elasticities, only related to technologies supplying the demands directly,
$CIMP$ – import costs,
$CEXP$ – gains for export, and
$CINV$ – total investments (in case of levellized investment costs, see $CCAP$)

31.1.2 The Objective Function

$FUNC$

In its usual form the objective function contains the sum of all discounted costs, i.e. all kinds of costs that can be accounted for. All costs related to operation (i.e. resource use, operation costs, costs of demand elasticities,...) are discounted from the middle of the current period to the first year. Costs related to construction are by default discounted from the beginning of the current period to the first year. By using the facility of distributing the investments or accounting during construction these costs can

be distributed over some periods before or equal to the current one (see section 32.2). This distribution can also be performed for user defined relations.

The objective function has the following general form:

$$\sum_t \left[\beta_m^t \, \Delta t \left\{ \sum_{svd} \sum_l zsvd..lt \times \epsilon_{svd} \times \left[ccur(svd,t) + \sum_i \sum_m ro_{svd}^{mlt} \times cari(ml,t) \right] + \right.\right.$$

$$\sum_{svd} \epsilon_{svd} \times \sum_{e=0}^{e_d} Usvd.e.t \times \epsilon_{svd} \times \left[\kappa_e \times (ccur(svd,t) + \sum_m ro_{svd}^{mt} \times car2(m,t)) + \right.$$

$$\left. cred(d,e) + \sum_m ro_{svd}^{mt} \times car1(m,t) \right] + \sum_{svd} \sum_{\tau=t-\tau_{svd}}^{t} \Delta\tau \times Yzsvd..\tau \times cfix(svd,\tau) +$$

$$\sum_r \left[\sum_g \sum_l \sum_p Rzrgp.lt \times cres(rgpl,t) + \right.$$

$$\left.\left. \sum_c \sum_l \sum_p Izrcp.lt \times cimp(rcpl,t) - \sum_c \sum_l \sum_p Ezrcp.lt \times cexp(rcpl,t) \right] \right\} +$$

$$\beta_b^t \times \left\{ \sum_{svd} \sum_{\tau=t}^{t+t_d} \Delta(t-1) \times Yzsvd..\tau \times \left[ccap(svd,\tau) \times fri_{svd}^{t_d-\tau} + \right.\right.$$

$$\left.\left. \sum_i \sum_m rc_{svd}^{mt} \times cari(m,t) \times fra_{svd,m}^{t_d-\tau} \right] \right\} \Bigg] ,$$

where

Δt is the length of period t in years,

$$\beta_b^t = \prod_{i=1}^{t-1} \left[\frac{1}{1+\frac{dr(i)}{100}} \right]^{\Delta i} ,$$

$$\beta_m^t = \beta_b^t \times \left[\frac{1}{1+\frac{dr(t)}{100}} \right]^{\frac{\Delta t}{2}} ,$$

$dr(i)$ is the discount rate in period i in percent,

$zsvd..lt$ is the annual consumption of technology v of fuel s load region l and period t; if v has no load regions, $l =$ ".".

ϵ_{svd} is the efficiency of technology v in converting s to d,

$ccur(svd,t)$ are the variable operation and maintenance costs of technology v (per unit of main output) in period t,

ro_{svd}^{mlt} is the relative factor per unit of output of technology v for relational constraint m in period t, load region l,

$car1(m,t)$ and $car2(m,t)$ are the coefficients for the objective function, that are related to the user defined relation m in period t,

$car1(ml,t)$ and $car2(ml,t)$ are the same for load region l, if relation m has load regions,

$Usvd.e.t$ is the annual consumption of fuel s of end-use technology v in period t and elasticity class e,

κ_e is the factor giving the relation of total demand for d to the demand reduced due to the elasticity to level e,

ro^{mt}_{svd} is the relative factor per unit of output of technology v for relational constraint m in period t,

$cred(d,e)$ is the cost associated with reducing the demand for d to elasticity level e,

$Yzsvd..t$ is the annual new built capacity of technology v in period t,

$cfix(svd,t)$ are the fix operation and maintenance cost of technology v that was built in period t,

$ccap(svd,t)$ is the specific investment cost of technology v in period t (given per unit of main output),

fri^n_{svd} is the share of this investment that has to be paid n periods before the first year of operation,

rc^{mt}_{svd} is the relative factor per unit of new built capacity of technology v for user defined relation m in period t,

$fra^n_{svd,m}$ is the share of the relative amount of the user defined relation m that occurs n periods before the first year of operation (this can, e.g., be used to account for the use of steel in the construction of solar towers over the time of construction),

$Rzrgp.lt$ is the annual consumption of resource r, grade g, elasticity class p in load region l and period t,

$cres(rgpl,t)$ is the cost of extracting resource r, grade g, elasticity class p in period t and load region l (this should only be given, if the extraction is not modelled explicitly),

$Izrcp.lt$ is the annual import of fuel r from country c in load region l, period t and elasticity class p; if r has no load regions l=".",

$cimp(rcpl,t)$ is the cost of importing r in period t from country c in load region l and elasticity class p,

$Ezrcp.lt$ is the annual export of fuel r to country c in load region l, period t and elasticity class p; if r has no load regions l=".", and

$cexp(rcpl,t)$ is the gain for exporting r in period t to country c in load region l and elasticity class p.

Chapter 32

Special Features of the Matrix Generator

The mathematical formulation of MESSAGE as presented in the previous sections shows the structure of all constraints as the matrix generator builds them up. The background of the more complicated features is given here for a better understanding.

32.1 The Time Horizon–Discounting the Costs

The whole time horizon of the calculations is divided into periods of optional length. All variables of MESSAGE are represented as average over the period they represent, resulting in a step-function. All entries in the objective function are discounted from the middle of the respective period to the first year, if they relate to energy flow variables and from the beginning of that period if they represent power variables. The function to discount the costs has the following form:

$$c_t = \frac{C_t^r}{\prod_{k=1}^{t-1} (1 + \frac{dr_k}{100})^{\Delta k} \times f_i} ,$$

where

C_t^r is the cost figure to be discounted,

c_t is the objective function coefficient in period t,

$$f_i = \begin{cases} 1 & \text{for costs connected to investments,} \\ (1 + \frac{dr_t}{100})^{\frac{\Delta t}{2}} & \text{else, and} \end{cases}$$

dr_k is the discount rate in period k.

32.2 Distributions of Investments

In order to support short term applications of MESSAGE the possibility to distribute the investments for a new built technology over several periods was implemented. The

same type of distributions can be applied to entries in user defined relations if they relate to construction. The distribution of investments can be performed in several ways. There is one common parameter that is needed for all of these possibilities, the construction time of the technology [*ct*].

The implemented possiblilities are: 1. Explicit definition of the different shares of investments for the years of construction. The input are *ct* figures that will be normalized to 1 internally. 2. The investment distribution is given as a polynomium of 2nd degree, the input consists of the three coefficients:

$$y = a + bx + cx^2 \quad , x = 1(1)ct,$$

where *ct* is the construction time. The values of the function are internally normalized to 1, taking into account the construction time. 3. Equal distribution of the investments over the construction period. 4. A distribution function based on a logistic function of the type

$$f = \frac{100}{1 + e^{-\alpha(x - x_0)}} \ ,$$

where

$$x_0 = \tfrac{ct}{2} \ ,$$

and

$$\alpha = \tfrac{2}{ct} \ln \left(\tfrac{100}{\epsilon} - 1\right) .$$

This function is expanded to a normalized distribution function of the following type:

$$g = \left[\frac{100}{1 + e^{-\frac{ln(\frac{100}{\epsilon} - 1)(x-50)}{50}}} - \epsilon \right] \times \frac{1}{1 - \frac{\epsilon}{50}} \ .$$

g gives the accumulated investment at the time x, x is given in percent of the construction time. The parameter ϵ describes the difference of the investment in the different years. ϵ near to 50 results nearly in equal distribution, an ϵ close to 0 indicates high concentration of the expenditures in the middle of the construction period.
In order to shift the peak of costs away from the middle of the construction period the function is transformed by a polynomium:

$$x = az^2 + bz \quad , 0 < z < 100 \, ,$$

where

$$b = \tfrac{5000 - d^2}{100d - d^2} \quad , 0 < d < 100 \, ,$$

and

$$a = \frac{1-b}{100} .$$

d denotes the time at that the peak of expenditures occurs in percent of ct. This kind of investment function was taken from [9].

The distribution of these yearly shares of investments is done starting in the first period of operation with a one years share, the expenditures of the remaining $ct - 1$ years are distributed to the previous periods.

The coefficients of the capacity variables of a technology in a relational constraint can be distributed like the investments.

32.3 The Load Curve

The years representing a period can be subdivided into so-called load regions. This can be done by either ordering the whole year according to the power requirements for the most important energy carriers like, e.g., electricity, or by grouping the year into load regions with similar characteristics (hereafter called characteristic loads), like, e.g., winter days and nights and summer days and nights. The first option results in an interpolation of the usual representation of the load curve by a step function (see Figure 32.1), the second one in a step-function where the time is still ordered in a historic way.

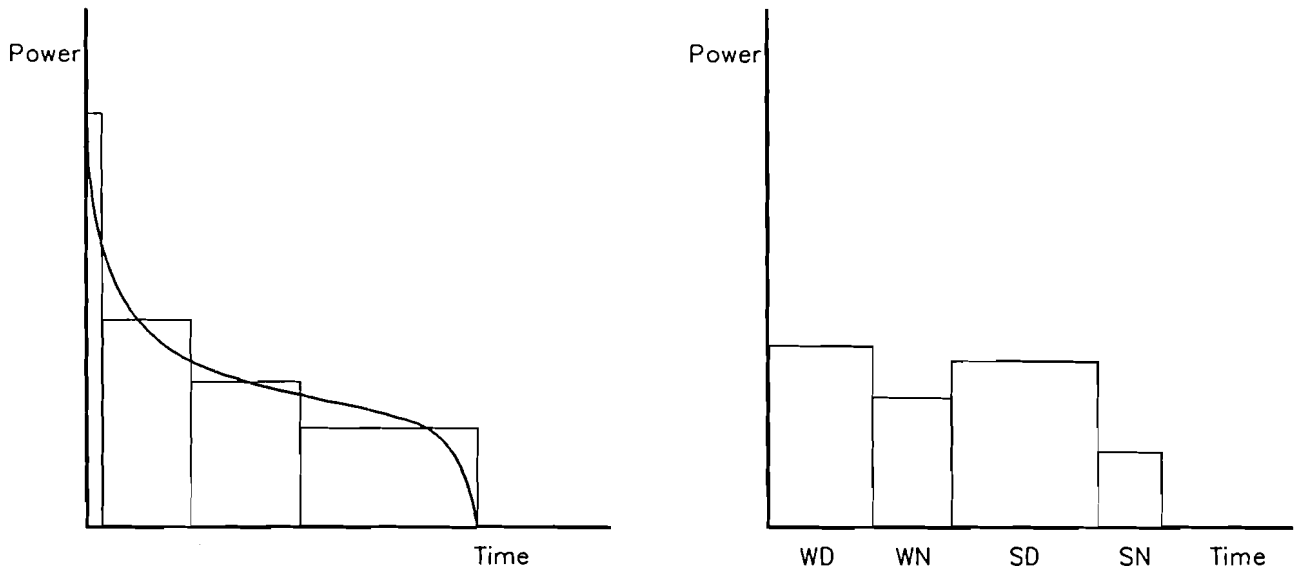

Figure 32.1: Example of an ordered (left) and a semi-ordered load curve (right) (WD stands fpr winter day, WN for winter night, SD for summer day and SN for summer night.)

32.4 Consideration of Load Variations in Conversion Technologies

The activity of a conversion technology is generated for each load region, if the main in- or output energy form is defined to have load regions. In this case the relation of these

activities between the load regions is freely chosen by the model. The relations can be fixed by the user to reflect a certain fixed production pattern. In this case the activity will only be generated once and written to the energy flow balances with coefficients reflecting the chosen pattern. A powerplant operating in baseload mode would for instance have the shares of the load regions in the year as coefficients in the balances of energy forms with load regions.

For end-use technologies (output level "U") the production is assumed to meet the demand pattern, the input of the technology is fixed to reflect the according demand variations. This can also be changed into to a different pattern. This would, e.g., model night storage heating systems that meet the heat demand of a houshold, but generate a final electricity demand with a different load distribution, namely at night.

32.5 Lag Times Between Input and Output of a Technology

Since MESSAGE can be used for very short time steps, even for steps of 1 year per period, the implementation of lag-times between input and output of a conversion technology seemed to be appropriate. One possible application are the reprocessing units for nuclear fuels, which usually keep the fuels for several years.

The lag time for a technology (keyword `lag`) is given in years and the period in which the output is available is calculated beginning from the middle of the period when the input is required.

32.6 Variable Inputs and Outputs

A lot of power plants can use different fuels for electricity generation, the highest variability occurs between oil products and natural gas as fuel. This can be modeled by having two or even more energy conversion variables with different inputs, efficiencies and variable operation and maintainance costs linked to one capacity in one capacity equation (definition using keywords `adda` and `activity`; see also section 25.2.1).

The same link of different conversion activities can be used to model co-generation of electricity and heat with a variable output pattern. In this case one of the conversions would be to electricity (with an efficiency ϵ_e) and the other one producing a mix of electricity and the maximal possible share of district heat (producing ϵ_c electricity and δ_c district heat from one unit of input). In the latter case the efficiency to electricity (ϵ_c) is lower than in the first case (ϵ_e), but the overall efficiency is naturally much higher. The two conversion variables have to be related to the same capacity by a factor giving the relative production of the main product possible with one unit of installed capacity, which is always related to the first operation mode. In the terms used above this would mean that the plant can produce ϵ_e electricity in the first operation mode, while it can produce ϵ_c with the same capacity in the second operation mode. For the model this means that the electric capacity is not utilized fully in the second mode, the relation has to be defined by the user. (It would be ϵ_e / ϵ_c in the described case, but could also be independent from the efficiencies for other technologies.)

32.7 The Contribution of Capacities Existing in the Base Year

The possible contribution of an installation that exists in the base year is kept track of over time. There are two possibilities to give the necessary information to MESSAGE.

1. Define the capacities that were built in the years $iyr, ..., iyr - \tau + 1$, with iyr = base year and τ = plant life in years explicitly. These capacities are then distributed to historic periods of the length ν.

2. Define the total capacity, c_0, that exists in iyr and the rate at that it grew in the last τ years, γ. This information is then converted to one similar to 1. by using the function:

$$y_0 = c_0 \frac{\gamma^{-\nu} - 1}{\nu(\gamma^{-\tau} - 1)} ,$$

$$y_t = y_0 \gamma^{-t \times \nu} , t = 1(1)\frac{\tau}{\nu} ,$$

where

y_t	is the annual construction in period $-t$, (0 = base year),
γ	is the annual growth of new installations before the base year,
c_0	is the total capacity in the base year,
τ	is the plant life, and
ν	is the length of the periods in that the time before the base year is divided.

The right hand sides in the capacity constraints are derived by summing up all the old capacities that still exist in a certain period (according to the plant life). If the life of a technology expires within a period, MESSAGE takes the average production capacity in this period as installed capacity (this represents a linear interpolation between the starting points of this and the following period).

32.8 Capacities which Operate Longer than the Time Horizon

If a capacity of a technology is built in one of the last periods its life time can exceed the calculation horizon. This fact is taken care of by reducing the investment costs by the following formula:

$$C_t^r = C_t \times \frac{\sum_{k=1}^{\tau_p - \nu} \prod_{\tau=t}^{t+k-1} \frac{1}{1+dr_\tau}}{\sum_{k=1}^{\tau_p} \prod_{\tau=t}^{t+k-1} \frac{1}{1+dr_\tau}}$$

To reduce the demand to the level q_i the supply has to have the cost

$$c(q_i) = \int_{q_i}^{1} q^{\frac{1}{\epsilon}}\, dq = \frac{1}{1+\frac{1}{\epsilon}} \times \left[1 - q_i^{1+\frac{1}{\epsilon}}\right],$$

a function increasing monotonously with decreasing q_i (see also Figure 32.2). In absolute terms this means that the cost would be higher by an absolute value of

$$R(Q_i) = c(\frac{Q_i}{Q_r}) \times Q_r \times P_r$$

compared to the cost at the reference level.

The step-function is then defined by choosing certain levels of demands and prices (Q_i, P_i), $i = 1(1)n$ with $Q_i < Q_r$, that fulfil the elasticity function. The code can choose, which demand level it supplies, but if it supplies a level $Q_i < Q_r$ it has to pay addititionally $R(Q_i)$, the cost of reducing the demand to level i.

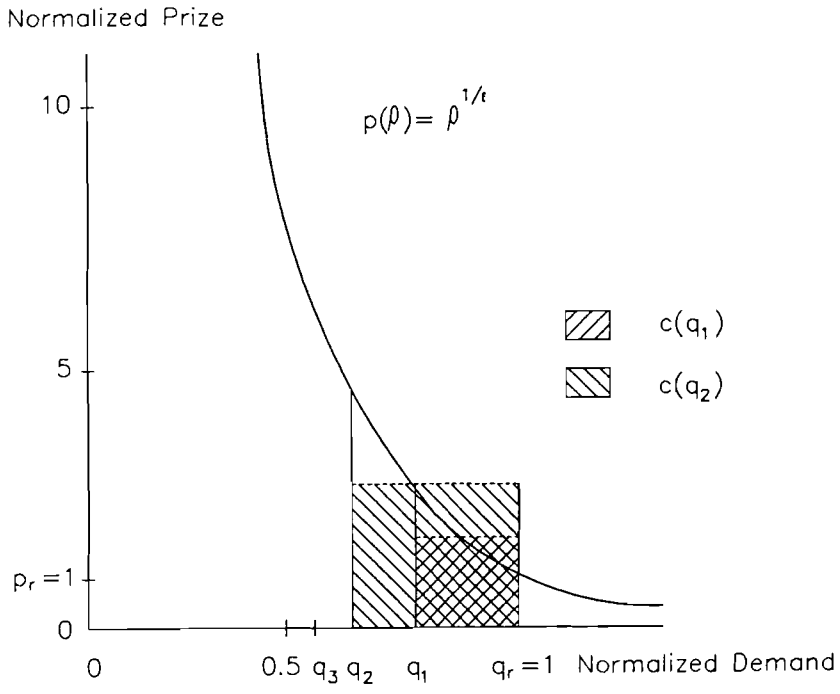

Figure 32.2: Representation of Demand Elasticities

32.10 Supply Elasticities

The reaction of the market prices to changes in demand can be expressed as elasticities:

$$\frac{P}{P_r} = \left[\frac{S}{S_r}\right]^{\alpha},$$

where

ν is the number of years the technology exists after the end of the calculation horizon,

dr_τ is the discount rate for year τ,

τ_p is the plant life in years,

C_t is the investment cost in year t, and

C_t^r is the reduced investment.

32.9 Own-Price Elasticities of Demand

Own-price elasticities of demand can be interpreted either as short-term elasticities resulting in reduced demand due to sharp price increases (they have to relate to a reference price- and demand level and represent renunciation of services) or as long-term elasticities reached by substituting capital for energy. In the latter case the user has to asure that the relatively decreased demand level is maintained over the calculation horizon by applying user-defined relations (see section 30). The costs and levels of demand reduction can be derived from the investments and savings that are associated to certain additional installations, like, e.g., three-glass windows to save in space heating.
The form of the own price elasticity function of demand is

$$\frac{Q}{Q_r} = \left[\frac{P}{P_r}\right]^\epsilon ,$$

where

Q_r is the reference demand level,

P_r is the reference price level, and

ϵ is the elasticity, (assumed to be < 0).

It says that the demand will decrease by a factor of x^ϵ if the price rises by x. This function is approximated by a step-function of the following form:
The demands (Q) and prices (P) are normalized to the reference levels:

$$Q = q \times Q_r ,$$

and

$$P = p \times P_r ,$$

the normalized values follow the function

$$q = p^\epsilon ,$$

or

$$p(q) = q^{\frac{1}{\epsilon}} .$$

where

P_r	is the reference price level,
S_r	the reference supply level, and
α	the elasticity.

The normalized form of this equation is

$$c = s^{\alpha},$$

where

$$c = \frac{P}{P_r},$$

and

$$s = \frac{S}{S_r}.$$

The relationship is converted to a step-function with n steps, which is shown in Figure 32.3. $f(s_1)$ is the cost of supplying amount s_1 relative to supplying s_r, while $f(s_1) + (s_2)$ ist the relative cost of supplying the amount s_2. The marginal costs are then defined as

$$\mu(s) = \frac{\int_{s_{i-1}}^{s_i} \sigma^{\alpha}\, d\sigma}{s_i - s_{i-1}},$$

where

$s_{i-1} < s \leq s_i$

According to the normalized function the total price of buying the amount s is then

$$tc(s) = \sum_{j=1}^{i-1} \mu(s_j) \times (s_j - s_{j-1}) + \mu(s_i) \times (s - s_{i-1}).$$

The price of the amount S, $S_{i-1} < S < S_i$, is defined as

$$TC(S) = P_r \times S_r \times tc(\frac{S}{S_r}).$$

In the matrix this function is implemented as $n + 1$ additive elasticity classes for resources and imports ($R_0 = S_r$, $R_i = S_i - S_{i-1}$, $i = 1(1)n$), which have increasing costs. The code takes these classes as supply one after the other and has to pay incrasing prices, then.

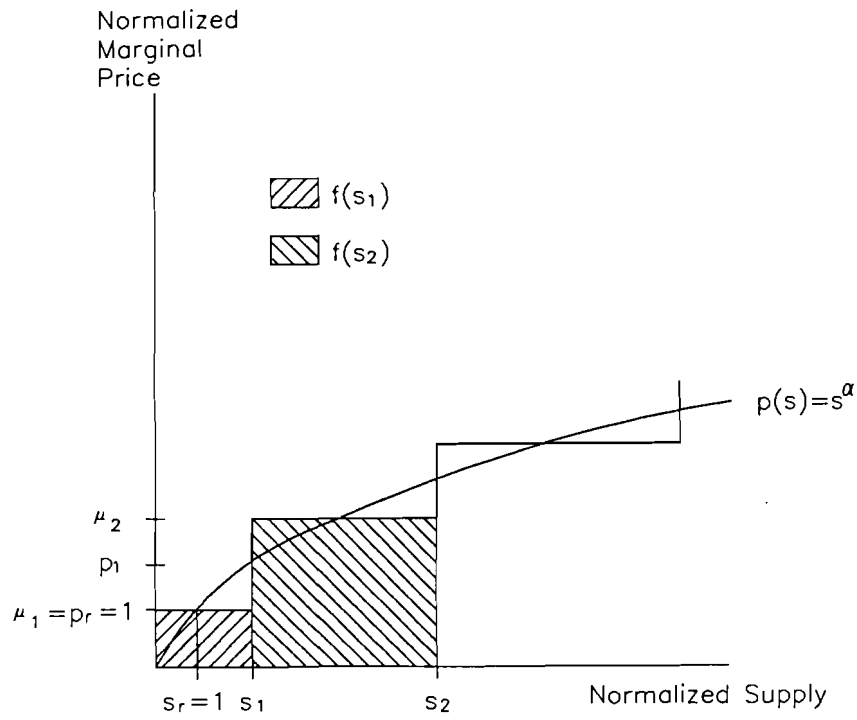

Figure 32.3: Representation of Supply Elasticities

32.11 The Mixed Integer Option

If the LP-package used to solve a problem formulated by MESSAGE has the capability to solve mixed integer problems, this can be used to improve the quality of the formulated problems, especially for applications to small regions.

The improvement consists in a definition of unit sizes for certain technologies that can only be built in large units. This avoids for instance the installation of a 10 kW nuclear reactor in the model of the energy system of a city or small region (it can only be built in units of e.g., 700 MW). Additionally this option allows to take care of the "economies of scale" of certain technologies.

This option is implemented for a technology by simply defining the unit size chosen for this technology (keyword `cmix`). The according capacity variable is then generated as integer in the matrix, its value is the installation of one powerplant of unit size.

If a problem is formulated as mixed integer it can be applied without this option by changing just one switch in the general definition file (keyword `mixsw`). Then all capacity variables are generated as real variables.

32.12 Nonlinearitites

In combination with MINOS [7] MESSAGE can be applied to problems with a partly nonlinear objective function or with nonlinear constraints. The requirements are that the functions are differentiable and convex with respect to the solution space.

In order to use a nonlinear objective or nonlinear constraints the user has to identify the variables that are to be included with nonlinear coefficients in the input file (keyword `nonl`; they will be written to the matrix as first entries in the columns section–as

required by MINOS) and to supply MINOS with an additional subroutine (`Funcon` for nonlinear constraints and `Funobj` for nonlinear objective gradients), which yields the nonlinear part of the constraints or objective and the first derivatives as required by MINOS.

In order to start a nonlinear problem it can be solved as linear problem in the beginning. The nonlinear variables can be fixed to user-defined estimates by specifying "initial bounds" in the bounds section.

The order in that the nonlinear variables appear in the input file is essential, because the same order is used for identifying them in `Funcon` and in `Funobj`. MESSAGE generates the activity variables first, then the capacity variables (both of them in the order in that the technologies appear in the input file). The loops in producing the columns are nested in the following order:

- load regions,
- demand elasticity classes, and
- time periods.

32.13 Multiobjective Optimization

MESSAGE is capable of handling two types of multiobjective optimization: It can generate weights on different types of activities or prepare the MPS-file for using the Reference Trajectory Optimization Method.

Weights on Activities

The common way to optimize several objectives at the same time is to define weights for the different types of activities in the model. MESSAGE provides an easy way to do this: It is possible to define costs that are added to the objective gradients of all technologies that have coefficients in a specific additional relation (see chapter 30), e.g. all technologies emitting SO_2 could get some addition to the objective gradient, if this addition is defined for the relation accounting for the emissions of SO_2.

Alternatively additive and multiplicative weights for all activities considered in the "Cost Accounting Rows" (see section 31.1.1) can be defined. As an example, additional costs (taxes) put on energy imports could be imposed this way.

The Reference Trajectory Method

The "Reference Trajectory Optimization Method"[1] is an approach to optimize more than one objective function for a problem in a way that circumvents the necessity to define weights on the single objectives. It allows to define *reference trajectories* for all objectives; the solution will lie on the "pareto"-optimal border of the feasible region and be as close as possible to all reference trajectories.

[1] This method is based on the "Reference Point Optimization Method"[3], that has been developed at IIASA and is implemented in the DIDASS system.

The way in which the pareto-optimal border is approached can either be problem oriented with an egalitarian approach between the objectives, or aspiration oriented, meaning that objectives with a reference point that is closer to the overall optimum (the UTOPIA[2] point), get higher influence on the solution.

The objectives can be grouped to nodes, for each of which a multiobjective approach is taken. The nodes are just summed up in the objective function. This feature is useful if interconnected models with separate objectives are depicted in one physical model.

Mathematically the objective functions are summed into a single function that minimizes the maximum difference between the reference trajectory and the actual value of the function for each time step. The difference is calculated using the Chebychev norm of the two points (reference point and actual value). The time steps are usually handled like the nodes, i.e. each point in time has a single objective. Alternatively the time steps can be included in one objective, which means that the compromise solution is searched over all objectives and time steps at once. This algorithm may lead to unrealistic results, since the dynamics of the model may not be handled adequately.

The aggregated objective function has the following general form:

$$\min \sum_n \sum_t \left[\left(\max_{j \in J_n} \sigma_j \frac{y_{j,t} - \bar{y}_{j,t}}{\alpha_{j,t}} \right) + \epsilon \sum_{j \in J_n} \sigma_j \frac{y_{j,t} - \bar{y}_{j,t}}{\alpha_{j,t}} \right]$$

where

n	is the index for the nodes,
t	is the index for the time steps,
J_n	is the set of objectives in node n,
j	is the index for the objectives,
σ_j	is a scaling factor to improve numerical stability (all objectives should have the same order of magnitude to avoid rounding-off errors),
$y_{j,t}$	is the actual value of objective j in time step t,
$\hat{y}_{j,t}$	is the reference point for objective j in time step t,
$\alpha_{j,t}$	is the scaling factor for objective j in time step t, it represents the way the pareto-optimal border is approached, and
ϵ	is a small number to drive th solution algorithm in the right direction.

If all time steps are to be aggregated into one objective, the sum over t is added to the sums over the variables instead of being outside the maximum.

The scaling factors are generated depending on a criterion regarding the way in which the absolutely optimal (and probably infeasible) point is approached:

Problem oriented scaling: $\alpha_{i,t} = \frac{1.}{\bar{y}_{i.t}}$

Aspiration oriented scaling: $\alpha_{i,t} = \frac{1.}{u_{i,t} - \bar{y}_{i.t}}$

where

$u_{j,t}$	is the optimal value for objective j in time step t with single-objective optimization. The respective values for all objective constitute the UTOPIA point.

[2]The UTOPIA point is the combination of all single-objective optima into one overall optimum, which is usually not in the feasible region.

Bibliography

[1] Agnew M., Schrattenholzer L., Voss A.: User's Guide for the MESSAGE Computer Program, IIASA, RM-78-21, 1978.

[2] Messner S.: User's Guide for the Matrix Generator of MESSAGE II , IIASA, WP-84-71, 1984.

[3] Grauer M., Lewandowski A., Schrattenholzer L.: Use of the reference level approach for the generation of efficient energy supply strategies. IIASA, WP-82-19, 1982.

[4] Cplex Optimization, Inc: Using the CPLEX Callable Library. Incline Village, NV 89451-9436, 1989-1994.

[5] IBM: Optimization Subroutine Library, Guide and Reference. Kingston, NY 12401, July 1991.

[6] U. Suhl: MOPS — Mathematical OPtimization System, Benutzerhandbuch. Freie Universität Berlin, 1989-1994.

[7] Murtagh B. A., Saunders M. A.: MINOS 5.0 USER'S Guide, Technical Report SOL 83-20, Systems Optimization Laboratory, Department of Operations Research, Stanford University, 1983.

[8] Gondzio, J and M. Makowski: HOPDM, Modular Solver for LP Problems. User's Guide to version 2.12. IIASA, WP-95-50, June 1995.

[9] Codoni R., Fritsch B. (Eds.): Capital Requirements of Alternative Energy Strategies, ETH Zürich, Institut für Wirtschaftsforschung, 1980.

Index

abbreviations in formats, 26
ADB: application data bank, 22

BYR — base year, 25, 28

Cin15: intermedate file for converting tit/cin-file, 87
cin: calculating instructions, 76
co-generation, 144
constraints
  exports, 126
continuation lines, 23
conversion technologies, 113

directory structure, 11
discounting the costs, 141

elasticities: demand, 146
elasticities: supply, 147
energy forms, 31
environment variable MMS_PRO, 11
equations
  user-defined relations, 134
  annual imports, 124, 125
  annual resource extraction, 120
  annual resource extraction per grade, 121
  central conversion balance, 129
  cost accounting rows, 138
  demand constraints, 127
  dynamic constraints on construction variables, 117
  dynamic constraints on imports, 125, 126
  dynamic constraints on resource extraction, 121, 122
  dynamic constraints on technology activity, 118
  energy distribution balance, 128
  energy transmission and transport balance, 129
  import per country, 124
  objective function, 138
  primary energy balance, 130
  resource consumption, 131
  resource depletion, 120
  resources per grade, 120
  stock-piling of fuels, 132
  technology capacity, 114
existing capacities, 145

fil: file with additional time series, 82

GEN-file, 22
glob.dir: directory file, 13
glob.reg: definition of regions, 12

input files, 21
integer variables, 49, 63, 149
investments distribution, 141
isw — input switch for time series, 26

Keyword
  *, 32
  #, 32, 38, 52, 58, 62
  , 38
  activity, 43
  ADB, 27
  ADB:, 24
  adda, 43
  bda, 43
  bdc, 43
  byrex, 56, 61
  cela, 56, 61
  chkacc, 64
  cload, 44
  clr, 61
  cmix, 49
  con1a, 44
  con1c, 44
  con2a, 45
  con2c, 45
  copy, 45
  corin, 45
  corout, 46
  cost, 38, 56, 61

demand:, 24, 35
description:, 23, 24, 28
discon, 46
drate, 28
drate:/vdrate:, 24
endata, 23
energyforms:, 24, 31
exports:, 24
fom, 46
fuel, 56, 61
fuelidentification:, 23, 31
fyear, 46
grade, 56
hisc, 48
imat, 64
imports:, 24
inp, 47
inv, 47
lag, 48
LDB, 27
lela, 56, 61
length, 29
levnam, 64
limit, 38
llr, 62
loadcurve:, 24, 35
loadregions:, 24, 29
ltype, 29
lyear, 48
mform, 64
minp, 49
mixsw, 63
mobj, 50
moutp, 50
mpa, 50
mpc, 50
mpe, 57, 62
ncap, 64
nonl, 51
ntrun, 63
outp, 51
plf, 51
pll, 52
prel, 52
problem:, 24, 28
reflim, 57, 62
relations1:, 24, 37
relations2:, 24, 37
resources:, 24
resrem, 57
systems:, 23, 24
TDB, 27
TDB:, 23, 24
timesteps:, 24, 28
update-file, 23
uplim, 57, 62
volume, 57
vom, 52
keywords for input, 23

lag times, 144
LDB: loacal data bank, 22
levellized costs, 67
load curve, 143
load variations in technologies, 143
LP[NT] — period lengths, 25, 28
LPO: length of historic periods, 29

matrix generator MXG, 15
mixed integer option, 149
mms.def: general definitions file, 11
mms.pro: the profile, 11
MMS_PRO: environment variable, 11, 21
multiobjective optimization, 150

NLR — number of load regions, 25
nonlinear objective function or constraint, 149
NR — number of time steps, 25

objective function, 138
optimizer: input and output files, 15
optimizers supported, 11

postprocessor CAP, 16

reference trajectory method, 150
region definition, 12
region.dir: region directory file, 14
relations between periods in user-defined relations, 37, 136
report writer: REPO, 16

scrap value implementation, 145
solution analyzer: ANA, 17
solution sorting: MKSF, 16

TDB: technology data bank, 22
time series: format and input switches, 26
tit: title-file, 72

TLR — number of load regions for technology, 25

variables
export, 123
import, 123
resource extraction, 119
stock-piling of fuels, 132
technology activity, 113
technology construction, 114

YR[NT+1] — first year for each period, 25, 28