
AIMMS Language Reference - Mixed Complementarity Problems

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com or order your hard-copy at www.lulu.com/aimms.

Copyright © 1993–2011 by Paragon Decision Technology B.V. All rights reserved.

Paragon Decision Technology B.V.	Paragon Decision Technology Inc.	Paragon Decision Technology Pte.
Schipholweg 1	500 108th Avenue NE	Ltd.
2034 LS Haarlem	Ste. # 1085	80 Raffles Place
The Netherlands	Bellevue, WA 98004	UOB Plaza 1, Level 36-01
Tel.: +31 23 5511512	USA	Singapore 048624
Fax: +31 23 5511517	Tel.: +1 425 458 4024	Tel.: +65 9640 4182
	Fax: +1 425 458 4025	

Email: info@aimms.com
WWW: www.aimms.com

AIMMS is a registered trademark of Paragon Decision Technology B.V. IBM ILOG CPLEX and sc CPLEX is a registered trademark of IBM Corporation. GUROBI is a registered trademark of Gurobi Optimization, Inc. KNITRO is a registered trademark of Ziena Optimization, Inc. XPRESS-MP is a registered trademark of FICO Fair Isaac Corporation. MOSEK is a registered trademark of Mosek ApS. WINDOWS and EXCEL are registered trademarks of Microsoft Corporation. $\text{T}_{\text{E}}\text{X}$, $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$, and $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ are trademarks of the American Mathematical Society. LUCIDA is a registered trademark of Bigelow & Holmes Inc. ACROBAT is a registered trademark of Adobe Systems Inc. Other brands and their products are trademarks of their respective holders.

Information in this document is subject to change without notice and does not represent a commitment on the part of Paragon Decision Technology B.V. The software described in this document is furnished under a license agreement and may only be used and copied in accordance with the terms of the agreement. The documentation may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Paragon Decision Technology B.V.

Paragon Decision Technology B.V. makes no representation or warranty with respect to the adequacy of this documentation or the programs which it describes for any particular purpose or with respect to its adequacy to produce any particular result. In no event shall Paragon Decision Technology B.V., its employees, its contractors or the authors of this documentation be liable for special, direct, indirect or consequential damages, losses, costs, charges, claims, demands, or claims for lost profits, fees or expenses of any nature or kind.

In addition to the foregoing, users should recognize that all complex software systems and their documentation contain errors and omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors shall not be responsible under any circumstances for providing information or corrections to errors and omissions discovered at any time in this book or the software it describes, whether or not they are aware of the errors or omissions. The authors, Paragon Decision Technology B.V. and its employees, and its contractors do not recommend the use of the software described in this book for applications in which errors or omissions could threaten life, injury or significant loss.

This documentation was typeset by Paragon Decision Technology B.V. using $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ and the LUCIDA font family.

Chapter 18

Mixed Complementarity Problems

This chapter discusses the special identifier types and language constructs that AIMMS offers to allow you to formulate mixed complementarity problems. Although mixed complementarity problems do not involve optimization, they are specified through variables which are linked to constraints in terms of these variables, and thus fall into the common framework of a MATHEMATICAL PROGRAM. AIMMS also supports nonlinear optimization problems with additional complementarity constraints (also known as MPCC or MPEC problems)

This chapter

18.1 Complementarity problems

Complementarity relations arise in a variety of engineering and economics applications, most commonly to express an equilibrium of quantities such as forces or prices. One standard application in engineering arises in contact mechanics, where complementarity expresses the fact that friction occurs only when two bodies are in contact. Other applications are found in structural mechanics, structural design, traffic equilibrium and optimal control.

Complementarity problems

Interest among economists in solving complementarity problems is due in part to increased use of computational general equilibrium models, where for instance complementarity is used to express Walras' Law, and in part to the equivalence of various games to complementarity problems.

Economic models

Some generalizations of nonlinear programming, such as multi-level optimization—in which auxiliary objectives are to be minimized—may be reformulated as problems with complementarity conditions. Also, by formulating the Kuhn-Tucker conditions of a nonlinear optimization model one obtains a complementarity problem, which could be solved by a complementarity solver. In the latter case, however, one requires second-order derivative information of all constraints in the original optimization model.

Nonlinear optimization

Complementarity problems are, in general, systems of nonlinear constraints where variables in the system are linked to constraints in the form of complementarity conditions. There are two forms of complementarity conditions, the classical complementarity condition, and its generalization, the mixed complementarity condition.

Complementarity conditions

The classical form of a complementarity condition involves a nonnegative variable x_i and an associated function $f_i(x)$. It requires that

Classical complementarity conditions

$$\begin{aligned} x_i = 0 & \quad \text{and} \quad f_i(x) \geq 0, \text{ or} \\ x_i > 0 & \quad \text{and} \quad f_i(x) = 0. \end{aligned}$$

This condition states that of both inequalities, at least one must reach its bound. Alternatively, one can formulate this complementarity condition as $x_i \geq 0$, $f_i(x) \geq 0$ and $x_i \cdot f_i(x) = 0$.

The mixed form of a complementarity condition involves a bounded variable $l_i \leq x_i \leq u_i$ with an associated function $f_i(x)$. It requires that

Mixed complementarity condition

$$\begin{aligned} x_i = l_i & \quad \text{and} \quad f_i(x) \geq 0, \text{ or} \\ x_i = u_i & \quad \text{and} \quad f_i(x) \leq 0, \text{ or} \\ l_i < x_i < u_i & \quad \text{and} \quad f_i(x) = 0. \end{aligned}$$

This condition states that either x_i must reach one of its bounds, or the function $f_i(x)$ must be zero. A mixed complementarity condition can be split into two classical complementarity conditions (albeit by introducing auxiliary variables). The classical complementarity condition, on the other hand, is a special case of the mixed complementarity condition by choosing $l_i = 0$ and $u_i = \infty$.

By choosing $l_i = -\infty$ and $u_i = \infty$, the mixed complementarity condition reduces to the special case

Special cases

$$x_i \text{ is "free" and } f_i(x) = 0.$$

Another special case is obtained when $l_i = u_i$. The mixed complementarity condition then reduces to

$$x_i = l_i \text{ and } f_i(x) \text{ is "free"}$$

All complementarity conditions described above can be represented by associating a variable with a single constraint, which will form the basis for representing complementarity conditions in AIMMS. Consider the inequalities

Supported complementarity conditions in AIMMS

$$l_{x_i} \leq x_i \leq u_{x_i} \quad \text{and} \quad l_{f_i} \leq f_i(x) \leq u_{f_i}$$

where $f_i(x)$ is a nonlinear expression, and *exactly* two of the constants l_{x_i} , u_{x_i} , l_{f_i} and u_{f_i} are finite. The six possible cases are enumerated in Table 18.1, and are discussed below.

Case	l_{x_i}	u_{x_i}	l_{f_i}	u_{f_i}
1	finite	finite	$-\infty$	∞
2	finite	∞	finite	∞
3	finite	∞	$-\infty$	finite
4	$-\infty$	finite	finite	∞
5	$-\infty$	finite	$-\infty$	finite
6	$-\infty$	∞	finite	finite

Table 18.1: Six allowed cases with exactly two finite bounds

The case $l_{x_i} \leq x_i \leq u_{x_i}$ and $-\infty \leq f_i(x) \leq \infty$ corresponds to the mixed complementarity condition already discussed above: Case 1

$$\begin{aligned} x_i &= l_{x_i} \quad \text{and} \quad f_i(x) \geq 0, \text{ or} \\ x_i &= u_{x_i} \quad \text{and} \quad f_i(x) \leq 0, \text{ or} \\ l_{x_i} < x_i < u_{x_i} \quad \text{and} \quad f_i(x) &= 0. \end{aligned}$$

The case $l_{x_i} \leq x_i \leq \infty$ and $l_{f_i} \leq f_i(x) \leq \infty$ corresponds to the classical complementarity condition Case 2

$$\begin{aligned} \hat{x}_i &= 0 \quad \text{and} \quad \hat{f}_i(x) \geq 0, \text{ or} \\ \hat{x}_i &> 0 \quad \text{and} \quad \hat{f}_i(x) = 0. \end{aligned}$$

where $\hat{x}_i = x_i - l_{x_i}$ and $\hat{f}_i(x) = f_i(x) - l_{f_i}$.

The case $l_{x_i} \leq x_i \leq \infty$ and $-\infty \leq f_i(x) \leq u_{f_i}$ corresponds to the classical complementarity condition Case 3

$$\begin{aligned} \hat{x}_i &= 0 \quad \text{and} \quad \hat{f}_i(x) \geq 0, \text{ or} \\ \hat{x}_i &> 0 \quad \text{and} \quad \hat{f}_i(x) = 0. \end{aligned}$$

where $\hat{x}_i = x_i - l_{x_i}$ and $\hat{f}_i(x) = u_{f_i} - f_i(x)$.

The case $-\infty \leq x_i \leq u_{x_i}$ and $l_{f_i} \leq f_i(x) \leq \infty$ corresponds to the classical complementarity condition Case 4

$$\begin{aligned} \hat{x}_i &= 0 \quad \text{and} \quad \hat{f}_i(x) \geq 0, \text{ or} \\ \hat{x}_i &> 0 \quad \text{and} \quad \hat{f}_i(x) = 0. \end{aligned}$$

where $\hat{x}_i = u_{x_i} - x_i$ and $\hat{f}_i(x) = f_i(x) - l_{f_i}$.

The case $-\infty \leq x_i \leq u_{x_i}$ and $-\infty \leq f_i(x) \leq u_{f_i}$ corresponds to the classical complementarity condition Case 5

$$\begin{aligned} \hat{x}_i &= 0 \quad \text{and} \quad \hat{f}_i(x) \geq 0, \text{ or} \\ \hat{x}_i &> 0 \quad \text{and} \quad \hat{f}_i(x) = 0. \end{aligned}$$

where $\hat{x}_i = u_{x_i} - x_i$ and $\hat{f}_i(x) = u_{f_i} - f_i(x)$.

The case $-\infty \leq x_i \leq \infty$ and $l_{f_i} \leq f_i(x) \leq u_{f_i}$ with $l_{f_i} = u_{f_i}$ corresponds to the first special case of the mixed complementarity condition Case 6: $l_{f_i} = u_{f_i}$

$$x_i \text{ is "free" and } \hat{f}_i(x) = 0.$$

where $\hat{f}_i(x) = f_i(x) - l_{f_i}$.

After the introduction of variables $x_i^+, x_i^- \geq 0$ and functions Case 6: $l_{f_i} < u_{f_i}$

$$\begin{aligned} f_i^x(x) &= x_i - x_i^+ - x_i^- \\ f_i^+(x) &= f_i(x) - l_{f_i} \\ f_i^-(x) &= u_{f_i} - f_i(x) \end{aligned}$$

the case $-\infty \leq x_i \leq \infty$ and $l_{f_i} \leq f_i(x) \leq u_{f_i}$ with $l_{f_i} < u_{f_i}$ corresponds to a system of three simultaneous complementarity conditions

$$x_i \text{ is "free" and } f_i^x(x) = 0$$

$$\begin{aligned} x_i^+ &= 0 \quad \text{and} \quad f_i^+(x) \geq 0, \text{ or} \\ x_i^+ &> 0 \quad \text{and} \quad f_i^+(x) = 0 \end{aligned}$$

$$\begin{aligned} x_i^- &= 0 \quad \text{and} \quad f_i^-(x) \geq 0, \text{ or} \\ x_i^- &> 0 \quad \text{and} \quad f_i^-(x) = 0. \end{aligned}$$

AIMMS supports the variable-constraint couples with two finite bounds, as discussed above, through the special COMPLEMENTARITY VARIABLE data type. The declaration and attributes of this data type are discussed in the next section, while section 18.3 describes the declaration of mixed complementarity models through the common MATHEMATICAL PROGRAM declaration. AIMMS support

Like with nonlinear optimization models, not all mixed complementarity systems that can be formulated are well-behaved. For instance, a variable $x \geq 0$ with an associated constraint $1 - x \geq 0$, only admits the solutions 0 and 1, which would destroy the continuous character of complementarity problems. For systems of complementarity conditions that are not well-behaved, the solution process may produce no, or unexpected results. Well-behaved systems

18.2 COMPLEMENTARITY VARIABLE declaration and attributes

To support you in formulating a complementarity model, AIMMS provides a special type of variable, the COMPLEMENTARITY VARIABLE. The attributes of a complementarity variable allow you to declare an (indexed) class of variables in a complementarity model along with their associated constraints. The attributes of a COMPLEMENTARITY VARIABLE are listed in Table 18.2.

Complementarity variables

By construction, this new variable type automatically ensures that every variable in a complementarity model is associated with a single constraint. Also, when AIMMS detects that the total number of (finite) bounds on both the complementarity variable and its associated constraint is not equal to two (as required above), a compilation error will result. Thus, COMPLEMENTARITY VARIABLES will help to reduce the most common declaration errors for this type of model.

Automatic sanity checks

Attribute	Value-type	See also page
INDEX DOMAIN	<i>index-domain</i>	42, 207, 207
RANGE	<i>range</i>	207
UNIT	<i>unit-valued expression</i>	45, 210
TEXT	<i>string</i>	19, 45
COMMENT	<i>comment string</i>	19
COMPLEMENT	<i>expression</i>	216
NONVAR STATUS	<i>reference</i>	211
PROPERTY	NoSave, Complement	

Table 18.2: COMPLEMENTARITY VARIABLE attributes

Through the INDEX DOMAIN attribute of a complementarity variable you can specify domain of tuples for which you want AIMMS to generate a variable and its associated constraint. During generation, AIMMS will only generate a variable for all tuples that satisfy all domain restrictions that you have imposed on the domain.

The INDEX DOMAIN attribute

In the RANGE attribute you can specify the lower and upper bound of a complementarity variable, in a similar manner for ordinary VARIABLES (see also Section 14.1). During generation, AIMMS will perform a runtime check, for every individual tuple in the index domain, whether the number of finite bounds specified here, plus the number of finite bounds in the constraint specified in the COMPLEMENT attribute, exactly equals two.

The RANGE attribute

The COMPLEMENT attribute allows you to specify the constraint that must be associated with the complementarity variable at hand. With $f(x, \dots)$ a general nonlinear function, the following types of expressions are allowed

The COMPLEMENT attribute

- $f(x, \dots) \geq a$ (variable must have a single-sided RANGE),
- $f(x, \dots) \leq a$ (variable must have a single-sided RANGE),
- $a \leq f(x, \dots) \leq b$ (variable must be free),
- $f(x, \dots) = a$ (variable must be free), or
- $f(x, \dots)$ (variable must be bounded).

In addition, the COMPLEMENT attribute can hold a reference to an existing CONSTRAINT in your model, which then should hold a definition as one of the cases above. The COMPLEMENT attribute can also hold a scalar element parameter into the set ALLConstraints, which offers the possibility to assign different constraints to the complementarity variable in sequential solves.

In the constraint listing, the constraints associated with a complementarity variable will be listed with a generated name consisting of the name of the COMPLEMENTARITY VARIABLE with an additional suffix “_complement”.

Constraint listing

With the NONVAR STATUS attribute you can indicate for which tuples you want AIMMS to consider the complementarity variable as a parameter, i.e. with the lower and upper bound set equal to the level value prior to solving the model (see also Section 14.1.1). From the mixed complementarity condition it follows that the function in the corresponding constraint is then allowed to assume arbitrary values, whence there is no strict need to generate the variable and constraint for the solver.

The NONVAR STATUS attribute

The value of the NONVAR STATUS attribute must be an expression in some or all of the indices in the index list of the variable, allowing you to change the non-variable status of individual elements or groups of elements at once. When the NONVAR status assumes a positive value, AIMMS will not generate the variable and its associated constraint. For negative values, the variable and constraint will be generated, but reduces to the second special case of the mixed complementarity condition

Positive and negative values

$$\hat{x}_i = x_i - x_i^0 = 0 \quad \text{and} \quad f_i(x) \text{ is “free”,}$$

i.e. the function in the constraint will be allowed to assume arbitrary values.

Providing a UNIT for a complementarity variable will help you in a number of ways.

The UNIT attribute

- AIMMS will help you to check the consistency of all the constraints and assignments in your model (including the expression in the COMPLEMENT attribute), and
- AIMMS will use the units to scale the model that is sent to the solver.

Proper scaling of a model will generally result in a more accurate and robust solution process. You can find more information on the definition and use of units to scale mathematical programs in Chapter 30.

Complementarity variables support the properties `NoSave` and `Complement`. With the property `NoSave` you indicate that you do not want to store data associated with this variable in a case. The `Complement` property indicates that you are interested in the level values of the constraint defined in the `COMPLEMENT` attribute. When this property is set, AIMMS will make the level value of this constraint available through the `.Complement` suffix of the complementarity variable at hand.

The PROPERTY attribute

The declaration of the complementarity variable `MembraneHeight` expresses a complementarity condition for the height of a membrane in a rectangular (x, y) -grid, with a uniform external force acting on each cell in the grid.

Example

```
COMPLEMENTARY VARIABLE:
  identifier   : MembraneHeight
  index domain : (x,y)
  range       : [MembraneLowerBound(x,y), MembraneUpperBound(x,y)]
  complement  : 4*MembraneHeight(x,y)
                - MembraneHeight(x+1,y) - MembraneHeight(x-1,y)
                - MembraneHeight(x,y+1) - MembraneHeight(x,y-1)
                - CellForce ;
```

The complementarity condition expresses that either the membrane reaches one its given bounds (for instance, an obstacle placed in the way of the membrane), or the external force on the cell must be equal to the internal forces acting on the cell caused by differences in height with neighboring cells.

18.3 Declaration of mixed complementarity models

To define a pure mixed complementarity model, you must declare a `MATHEMATICAL PROGRAM` (see also Section 15.1) and specify `mcp` as the `TYPE` attribute of the `MATHEMATICAL PROGRAM`. In the `VARIABLES` attribute you can specify a subset of the set of all `COMPLEMENTARITY VARIABLES` to be included in the mixed complementarity model at hand. Based on this specification, AIMMS will automatically generate all constraints associated with these complementarity variables, resulting in a square system.

Mixed complementarity models

In addition, AIMMS allows you to add ordinary variables to the `VARIABLES` attribute, and to specify additional constraints in the `CONSTRAINTS` attribute of the `MATHEMATICAL PROGRAM` that must be satisfied as well. If the solver used to solve the mixed complementarity model requires a square system, AIMMS will automatically add auxiliary constraints or variables to the generated system,

Additional variables and constraints

and provide the linkages with the ordinary variables and constraints you have added to the system.

For a mixed complementarity problem you should not specify the `OBJECTIVE` and `DIRECTION` attributes, as a complementarity solver will only compute a feasible solution that satisfies all the complementarity conditions specified. If these attributes are not empty, AIMMS will produce a runtime error when you apply the `SOLVE` statement the corresponding `MATHEMATICAL PROGRAM` (see also Section 15.3).

No optimization

A mixed complementarity model containing the declaration of the complementarity variable `MembraneHeight` declared in the previous section, is defined by the following declaration.

Example

```
MATHEMATICAL PROGRAM:
  identifier : Membrane
  variables  : AllVariables
  type       : mcp ;
```

As usual, you can solve the `Membrane` through the statement

```
solve Membrane;
```

which will generate the mixed complementarity model and invoke a suitable solver for `mcp` problem type.

18.4 Declaration of MPCC models

Through the `KNITRO` solver, AIMMS also supports mathematical programs with complementarity constraints (MPCC models). MPCC models are also more commonly denoted by other modeling languages as MPEC models, which form a more general, and much more difficult, class of optimization problems. A MPCC model is an ordinary NLP model with additional complementarity constraints that have to be satisfied.

MPCC models

To define a MPCC model, you must declare a `MATHEMATICAL PROGRAM` (see also Section 15.1) and specify `mpcc` as the `TYPE` attribute of the `MATHEMATICAL PROGRAM`. The variable set of a `MATHEMATICAL PROGRAM` of a MPCC model can contain ordinary variables as well as complementarity variables. Contrary to pure mixed complementary models, a MPCC model has an objective function.

Declaring MPCC models

To solve MPCC models in AIMMS, you need a license for the KNITRO solver. If you do not have a license for the KNITRO solver, AIMMS will return an error that it has no suitable solver available for the `mcc` class, whenever you try to solve a MPCC model. The KNITRO solver can also be used for solving pure mixed complementarity problems, but is, in general, far less efficient in that case than dedicated `mcp` solvers.

*Solving MPCC
models*