
AIMMS Function Reference - Matrix Manipulation

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

Matrix Manipulation Functions

AIMMS supports the following matrix manipulation functions:

- `MatrixActivateRow`
- `MatrixAddColumn`
- `MatrixAddRow`
- `MatrixDeactivateRow`
- `MatrixFreezeColumn`
- `MatrixGenerate`
- `MatrixModifyCoefficient`
- `MatrixModifyColumnType`
- `MatrixModifyDirection`
- `MatrixModifyLeftHandSide`
- `MatrixModifyLowerBound`
- `MatrixModifyQuadraticCoefficient`
- `MatrixModifyRightHandSide`
- `MatrixModifyRowType`
- `MatrixModifyType`
- `MatrixModifyUpperBound`
- `MatrixRegenerateRow`
- `MatrixRestoreState`
- `MatrixSaveState`
- `MatrixSolve`
- `MatrixUnfreezeColumn`

In addition, the following function can be used to add cuts during the solution process of a mixed integer program:

- `GenerateCut`

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP functions.

MatrixActivateRow

The procedure `MatrixActivateRow` activates a row in the matrix that was previously deactivated.

```
MatrixActivateRow(  
    MP,          ! (input) a mathematical program  
    row         ! (input) a scalar value  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row in the matrix; this can not be the objective row.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixDeactivateRow`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixAddColumn

The procedure `MatrixAddColumn` adds a column to the matrix.

```
MatrixAddColumn(  
  MP,           ! (input) a mathematical program  
  column       ! (input) a scalar value  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

column

A scalar reference to an existing column name in the model.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- Coefficients for this column can be added to the matrix by using the procedure `MatrixModifyCoefficient`. After calling `MatrixAddColumn` the type and the lower and upper bound of the column are set according to the definition of the corresponding symbolic variable. By using the procedures `MatrixModifyColumnType`, `MatrixModifyLowerBound` and `MatrixModifyUpperBound` the column type and the lower and upper bound can be changed.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixModifyCoefficient`, `MatrixModifyColumnType`, `MatrixModifyLowerBound`, `MatrixModifyUpperBound`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixAddRow

The procedure `MatrixAddRow` adds a row to the matrix.

```
MatrixAddRow(
    MP,          ! (input) a mathematical program
    row         ! (input) a scalar value
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row name in the model.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- Initially, the row will added with zero coefficients, regardless of whether the symbolic AIMMS constraint has a definition or not. Regeneration of all of the coefficients of the row according to its definition can be achieved through the procedure `MatrixRegenerateRow`. Individual coefficients of the row can be added by calling the procedure `MatrixModifyCoefficient`.
- After calling `MatrixAddRow` the type of the row is set to '`<=`' and the right-hand-side value to INF (the left-hand-side value is set to -INF). By using the procedures `MatrixModifyRowType` and `MatrixModifyRightHandSide` the row type and right-hand-side value can be changed.
- After a call to `MatrixAddRow` or `MatrixRegenerateRow` for a certain row it is not allowed to do another call to `MatrixAddRow` for that row.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixModifyCoefficient`, `MatrixModifyLeftHandSide`, `MatrixModifyRightHandSide`, `MatrixModifyRowType`, `MatrixRegenerateRow`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixDeactivateRow

The procedure `MatrixDeactivateRow` deactivates a row in the matrix. The row will be ignored by the solver until it is activated again.

```
MatrixDeactivateRow(  
    MP,          ! (input) a mathematical program  
    row         ! (input) a scalar value  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row in the matrix; this can not be the objective row.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- Deactivating a row results in changing the type of that row into '`<`' and the right hand side value into `INF` (the row coefficients do not change).
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the `GMP` library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding `GMP` procedures.

See also:

The procedure `MatrixActivateRow`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixFreezeColumn

The procedure `MatrixFreezeColumn` fixes the value of a column in the model. The column can be freed by using `MatrixUnfreezeColumn`.

```
MatrixFreezeColumn(  
    MP,          ! (input) a mathematical program  
    column,     ! (input) a scalar value  
    value       ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing column in the matrix.

value

The value to which the column should be fixed.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- Fixing a column to a certain value has the same effect as changing the lower and upper bound into that value.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixModifyLowerBound`, `MatrixModifyUpperBound`, `MatrixUnfreezeColumn`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixGenerate

The procedure `MatrixGenerate` instructs AIMMS to generate a mathematical program without actually solving it.

```
MatrixGenerate(  
    MP                ! (input) a mathematical program  
)
```

Arguments:

MP

A mathematical program to be generated. The mathematical program should be a linear, mixed-integer linear or quadratic programming model.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- The procedure `MatrixGenerate` can be used to generate a mathematical program, if your algorithm does not call the `SOLVE` statement to solve it initially, prior using the matrix manipulation routines.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the `GMP` library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding `GMP` procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyCoefficient

The procedure `MatrixModifyCoefficient` changes a coefficient in the matrix. This procedure can also be used to modify a coefficient in the objective row. The value for the coefficient can be equal to 0.0 prior to calling this procedure.

```
MatrixModifyCoefficient(  
    MP,           ! (input) a mathematical program  
    row,         ! (input) a scalar value  
    column,     ! (input) a scalar value  
    value       ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row in the matrix; this might be the objective row.

column

A scalar reference to an existing column in the matrix.

value

The new value that should be assigned to the coefficient corresponding to *row* and *column* in the matrix. This value should be unequal to NA, INF, -INF and UNDF.

Return value:

The procedure returns 1 on success, and 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyColumnType

The procedure `MatrixModifyColumnType` changes the type of a column in the matrix into either 'continuous' or 'integer'.

```
MatrixModifyColumnType(  
  MP,          ! (input) a mathematical program  
  column,     ! (input) a scalar value  
  type        ! (input) a column type  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

column

A scalar reference to an existing column in the matrix.

type

One of the column types 'continuous' or 'integer' that should be assigned to the column.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyDirection

The procedure `MatrixModifyDirection` changes the direction of a mathematical program to 'maximize', 'minimize' or 'none'. The direction 'none' is the instruction to the solver to find a feasible solution. If the type of the mathematical program is 'MIP' then the solver will try to find an integer feasible solution.

```
MatrixModifyDirection(  
    MP,           ! (input) a mathematical program  
    direction     ! (input) a direction  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

direction

One of the directions 'maximize', 'minimize' or 'none'.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The set `AllMatrixManipulationDirections`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyLeftHandSide

The procedure `MatrixModifyLeftHandSide` changes the left-hand-side of a row in the matrix.

```
MatrixModifyLeftHandSide(  
  MP,          ! (input) a mathematical program  
  row,        ! (input) a scalar value  
  value       ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing ranged row in the matrix.

value

The new value that should be assigned to the left-hand-side of the row. This value should be unequal to NA, UNDF and INF (but might be -INF).

Remarks:

- After a call to `MatrixSolve` AIMMS checks for each modified ranged row whether or not the left-hand-side value is valid, that is, the left-hand-side value should be unequal to INF.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixModifyRightHandSide`, `MatrixSolve`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyLowerBound

The procedure `MatrixModifyLowerBound` changes the lower bound of a column in the matrix.

```
MatrixModifyLowerBound(  
  MP,           ! (input) a mathematical program  
  column,      ! (input) a scalar value  
  value        ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing column in the matrix.

value

The new value that should be assigned to the lower bound of the column.

Return value:

The procedure returns 1 on success, and 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixModifyUpperBound`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyQuadraticCoefficient

The procedure `MatrixModifyQuadraticCoefficient` changes a quadratic coefficient in the objective row of a quadratic mathematical program. The value for the coefficient can be equal to 0.0 prior to calling this procedure.

```
MatrixModifyQuadraticCoefficient(  
  MP,           ! (input) a mathematical program  
  col1,         ! (input) a scalar value  
  col2,         ! (input) a scalar value  
  value        ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a quadratic programming model.

col1

A scalar reference to an existing column.

col2

A scalar reference to an existing column.

value

The new value that should be assigned to the quadratic coefficient corresponding to *col1* and *col2* in the objective row. This value should be unequal to NA, INF, -INF and UNDF.

Return value:

The procedure returns 1 on success, and 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyRightHandSide

The procedure `MatrixModifyRightHandSide` changes the right-hand-side of a row in the matrix.

```
MatrixModifyRightHandSide(
  MP,          ! (input) a mathematical program
  row,        ! (input) a scalar value
  value       ! (input) a numerical expression
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row in the matrix; this can not be the objective row.

value

The new value that should be assigned to the right-hand-side of the row. This value should be unequal to NA and UNDF (but might be INF or -INF).

Remarks:

- If you assign INF to the right-hand-side value of a row with type '=', `MatrixModifyRightHandSide` will not produce an error, since you might want to change the type of this row into '<=' (using `MatrixModifyRowType`) immediately thereafter.
- After a call to `MatrixSolve` AIMMS checks for each modified row whether or not the right-hand-side value is valid for the current row type. If the row type is '=' then the right-hand-side value should be unequal to INF and -INF; if the row type is '<=' or 'ranged' then it should be unequal to -INF.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixModifyLeftHandSide`, `MatrixModifyRowType`, `MatrixSolve`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyRowType

The procedure `MatrixModifyRowType` changes the type of a row in the matrix.

```
MatrixModifyRowType(
  MP,          ! (input) a mathematical program
  row,        ! (input) a scalar value
  type        ! (input) a row type
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row in the matrix; this can not be the objective row.

type

One of the row types '`<=`', '`=`', '`>=`' or '`ranged`' that should be assigned to the row.

Remarks:

- The following examples show what happens if we change the row type into '`ranged`':

```
a(x) <= 3   modified into 'ranged' results in  -inf <= a(x) <= 3
a(x) >= 3   modified into 'ranged' results in    3 <= a(x) <= inf
a(x) = 3    modified into 'ranged' results in    3 <= a(x) <= 3
```

The next examples show what happens if we change the row type of a '`ranged`' row:

```
2 <= a(x) <= 4   modified into '<=' results in  a(x) <= 4
2 <= a(x) <= 4   modified into '>=' results in  a(x) >= 2
2 <= a(x) <= 4   modified into '=' results in  a(x) = 4
```

- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyType

The procedure `MatrixModifyType` changes the type of a mathematical program from 'MIP' into 'RMIP', or vice versa.

```
MatrixModifyType(  
    MP,          ! (input) a mathematical program  
    type        ! (input) a mathematical programming type  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

type

One of the types 'MIP' or 'RMIP'.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixModifyUpperBound

The procedure `MatrixModifyUpperBound` changes the upper bound of a column in the matrix.

```
MatrixModifyUpperBound(  
  MP,           ! (input) a mathematical program  
  column,      ! (input) a scalar value  
  value        ! (input) a numerical expression  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

column

A scalar reference to an existing column in the matrix.

value

The new value that should be assigned to the upper bound of the column.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixModifyLowerBound`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixRegenerateRow

The procedure `MatrixRegenerateRow` regenerates the coefficients of a row according to the definition of its associated symbolic constraint in the model.

```
MatrixRegenerateRow(  
  MP,           ! (input) a mathematical program  
  row          ! (input) a scalar value  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

row

A scalar reference to an existing row name in the model.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- If the row does not exist yet, it will be automatically added to the matrix before generating its coefficients.
- Before regenerating the row, the procedure first removes all existing matrix coefficients.
- This procedure will automatically add columns that are not in the matrix.
- The row type and the right-hand-side value (and, if the row type is 'ranged', the left-hand-side value) are set according to the constraint definition.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixAddRow`, `MatrixModifyCoefficient`, `MatrixModifyLeftHandSide`, `MatrixModifyRightHandSide`, `MatrixModifyRowType`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixRestoreState

With procedure `MatrixRestoreState` you can restore the state of your mathematical program as it was on the moment that you called `MatrixSaveState`.

```
MatrixRestoreState(  
    MP,          ! (input) a mathematical program  
    state       ! (input) an integer scalar parameter  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

state

The value corresponding to a state that you want to restore.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixSaveState`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixSaveState

With the procedure `MatrixSaveState` you can save the current state of a mathematical program. Later on, after manipulating the mathematical program, you can restore this state by calling `MatrixRestoreState`.

```
MatrixSaveState(  
    MP,           ! (input) a mathematical program  
    state        ! (output) an integer scalar parameter  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

state

On return, contains a positive integer value assigned to the state.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- States are numbered from 1 upwards by AIMMS.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixRestoreState`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixSolve

The procedure `MatrixSolve` instructs the solver to solve a mathematical program in its current state after being modified by using several matrix manipulation procedures.

```
MatrixSolve(  
    MP          ! (input) a mathematical program  
)
```

Arguments:

MP

A mathematical program that was previously solved or generated. The mathematical program should be a linear, mixed-integer linear or quadratic programming model.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- After a call to `MatrixSolve` AIMMS will first check if all modifications performed by calling matrix manipulation procedures are all valid, before actually calling the solver. Most errors, however, will be filtered out by the matrix manipulation procedures themselves.
- As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedure `MatrixGenerate`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

MatrixUnfreezeColumn

The procedure `MatrixUnfreezeColumn` frees a column that was fixed with `MatrixFreezeColumn`. After calling `MatrixUnfreezeColumn` the value of the column can vary again between its lower and upper bound.

```
MatrixUnfreezeColumn(  
  MP,           ! (input) a mathematical program  
  column       ! (input) a scalar value  
)
```

Arguments:

MP

A mathematical program that was previously solved. The mathematical program should be a linear or mixed-integer linear programming model.

column

A scalar reference to an existing fixed column in the matrix.

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

As of AIMMS release 3.5, the matrix manipulation procedures have become deprecated. New projects should use the GMP library instead. Please refer to Table 21.6 of the Language Reference for a mapping of the matrix manipulation procedures to corresponding GMP procedures.

See also:

The procedures `MatrixFreezeColumn`, `MatrixModifyLowerBound`, `MatrixModifyUpperBound`. Matrix manipulation routines are discussed in more detail in Chapter 21 of the Language Reference.

GenerateCut

The procedure `GenerateCut` adds a row to the matrix during the solution process of a mixed integer program.

```
GenerateCut(  
    Arow,          ! (input) a scalar value  
    [local]       ! (optional, default 1) a scalar binary expression  
)
```

Arguments:

Arow

A scalar reference to an existing row name in the model.

local

A scalar binary value to indicate whether the cut is valid for the local problem (i.e. the problem corresponding to the current node in the solution process and all its descendant nodes) only (value 1) or for the global problem (value 0).

Return value:

The procedure returns 1 on success, or 0 otherwise.

Remarks:

- This procedure can only be called from within a `CallbackAddCut` callback procedure.
- A `CallbackAddCut` callback procedure will only be called when solving mixed integer programs with CPLEX 8.0 or higher, or with GUROBI 2.0 or higher.

See also:

See Section [15.2](#) of the Language Reference for more details on how to install a callback procedure to add cuts.