

---

**AIMMS Function Reference - Arithmetic Functions**

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](http://www.aimms.com)

**Part I**

---

**Function Reference**

# Arithmetic Functions

AIMMS supports the following arithmetic functions:

- Abs
- ArcCosh
- ArcCos
- ArcSin
- ArcSinh
- ArcTanh
- ArcTan
- Ceil
- Cos
- Cosh
- Cube
- Degrees
- Div
- ErrorF
- Exp
- Floor
- Log
- Log10
- MapVal
- Max
- Min
- Mod
- Power
- Precision
- Radians
- Round
- ScalarValue
- Sign
- Sin
- Sinh
- Sqr
- Sqrt
- Tan
- Tanh
- Trunc

- Val

---

**Abs**

```
Abs(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Abs returns the absolute value of x.

**Remarks:**

The function Abs can be used in constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems if the argument assumes values around 0.

**See also:**

Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## ArcCos

```
ArcCos(  
    x          ! (input) numerical expression  
)
```

### Arguments:

`x`  
A scalar numerical expression in the range  $[-1, 1]$ .

### Return value:

The ArcCos function returns the arccosine of  $x$  in the range  $0$  to  $\pi$  radians.

### Remarks:

- A run-time error results if  $x$  is outside the range  $[-1, 1]$ .
- The function ArcCos can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcSin](#), [ArcTan](#), [Cos](#). Arithmetic functions are discussed in full detail in [Section 6.1.4](#) of the Language Reference.

---

## ArcCosh

```
ArcCosh(  
    x          ! (input) numerical expression  
)
```

### Arguments:

x

A scalar numerical expression in the range  $[1, \infty)$ .

### Return value:

The ArcCosh function returns the inverse hyperbolic cosine of  $x$  in the range from 0 to  $\infty$ .

### Remarks:

- A run-time error results if  $x$  is outside the range  $[1, \infty]$ .
- The function ArcCosh can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcSinh](#), [ArcTanh](#), [Cosh](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## ArcSin

```
ArcSin(  
    x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression in the range  $[-1, 1]$ .

### Return value:

The ArcSin function returns the arcsine of  $x$  in the range  $-\pi/2$  to  $\pi/2$  radians.

### Remarks:

- A run-time error results if  $x$  is outside the range  $[-1, 1]$ .
- The function ArcSin can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcCos](#), [ArcTan](#), [Sin](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## ArcSinh

```
ArcSinh(  
    x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The ArcSinh function returns the inverse hyperbolic sine of  $x$  in the range from  $-\infty$  to  $\infty$ .

### Remarks:

The function ArcSinh can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcCosh](#), [ArcTanh](#), [Sinh](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## ArcTan

```
ArcTan(  
    x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The ArcTan function returns the arctangent of  $x$  in the range  $-\pi/2$  to  $\pi/2$  radians.

### Remarks:

The function ArcTan can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcSin](#), [ArcCos](#), [Tan](#). Arithmetic functions are discussed in full detail in [Section 6.1.4](#) of the Language Reference.

---

## ArcTanh

```
ArcTanh(  
    x          ! (input) numerical expression  
)
```

### Arguments:

$x$   
A scalar numerical expression in the range  $(-1, 1)$ .

### Return value:

The ArcTanh function returns the inverse hyperbolic tangent of  $x$ .

### Remarks:

- A run-time error results if  $x$  is outside the range  $(-1, 1)$ .
- The function ArcTanh can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [ArcCosh](#), [ArcSinh](#), [Tanh](#). Arithmetic functions are discussed in full detail in [Section 6.1.4](#) of the Language Reference.

---

## Ceil

```
Ceil(  
  x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The function `Ceil` returns the smallest integer value  $\geq x$ .

### Remarks:

- The function `Ceil` will round to the nearest integer, if it lies within the equality tolerances `equality_absolute_tolerance` and `equality_relative_tolerance`.
- The function `Ceil` can be used in the constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems around integer values.
- When the numerical expression contains a unit, the function `Ceil` will first convert the expression to the corresponding base unit, before evaluating the function itself.

### See also:

The functions `Floor`, `Round`, `Precision`, `Trunc`. Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference. Numeric tolerances are discussed in Section 6.2.2 of the Language Reference.

---

**Cos**

```
Cos(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The Cos function returns the cosine of x in the range  $-1$  to  $1$ .

**Remarks:**

The function Cos can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Sin](#), [Tan](#), [ArcCos](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## Cosh

```
Cosh(  
  x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The Cosh function returns the hyperbolic cosine of  $x$  in the range 1 to  $\infty$ .

### Remarks:

The function Cosh can be used in constraints of nonlinear mathematical programs.

### See also:

The functions [Sinh](#), [Tanh](#), [ArcCosh](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Cube**

```
Cube(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Cube returns  $x^3$ .

**Remarks:**

The function Cube can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Power](#), [Sqr](#), and [Sqrt](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## Degrees

```
Degrees(  
  x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The function `Degrees` returns the value of `x` converted from radians to degrees.

### Remarks:

The function `Degrees` can be used in constraints of linear and nonlinear mathematical programs.

### See also:

The function [Radians](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Div**

```
Div(  
  x,      ! (input) numerical expression  
  y      ! (input) numerical expression  
)
```

**Arguments:**

- $x$   
A scalar numerical expression.
- $y$   
A scalar numerical expression unequal to 0.

**Return value:**

The function `Div` returns  $x$  divided by  $y$  rounded down to an integer.

**Remarks:**

A run-time error results if  $y$  equals 0.

**See also:**

Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**ErrorF**

```
ErrorF(  
    x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function ErrorF returns the error function value  $\frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{t^2}{2}} dt$ .

**Remarks:**

The function ErrorF can be used in constraints of nonlinear mathematical programs.

**See also:**

Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Exp**

```
Exp(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Exp returns the exponential value  $e^x$ .

**Remarks:**

The function Exp can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Log](#), [Log10](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Floor**

```
Floor(  
    x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Floor returns the largest integer value  $\leq x$ .

**Remarks:**

- The function Floor will round to the nearest integer, if it lies within the equality tolerances `equality_absolute_tolerance` and `equality_relative_tolerance`.
- The function Floor can be used in the constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems around integer values.
- When the numerical expression contains a unit, the function Floor will first convert the expression to the corresponding base unit, before evaluating the function itself.

**See also:**

The functions [Ceil](#), [Round](#), [Precision](#), [Trunc](#). Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference. Numeric tolerances are discussed in Section 6.2.2 of the Language Reference.

---

**Log**

```
Log(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

`x`  
A scalar numerical expression in the range  $(0, \infty)$ .

**Return value:**

The function `Log` returns the natural logarithm  $\ln(x)$ .

**Remarks:**

- A run-time error results if `x` is outside the range  $(0, \infty)$ .
- The function `Log` can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Exp](#), [Log10](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Log10**

```
Log10(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

$x$

A scalar numerical expression in the range  $(0, \infty)$ .

**Return value:**

The function `Log10` returns the base-10 logarithm of  $x$ .

**Remarks:**

- A run-time error results if  $x$  is outside the range  $(0, \infty)$ .
- The function `Log10` can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions `Exp`, `Log`. Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference.

---

## MapVal

```
MapVal(
  x          ! (input) numerical expression
)
```

### Arguments:

*x*  
A scalar numerical expression.

### Return value:

The function `MapVal` returns the (integer) mapping value of any real or special number *x*, according to the following table.

Value <i>x</i>	Description	MapVal value
<i>number</i>	any valid real number	0
UNDF	undefined (result of an arithmetic error)	4
NA	not available	5
INF	$+\infty$	6
-INF	$-\infty$	7
ZERO	numerically indistinguishable from zero, but has the logical value of one.	8

### See also:

Special numbers in AIMMS and the `MapVal` function are discussed in full detail in Section [6.1.1](#) of the Language Reference.

---

**Max**

```
Max(  
  x1,      ! (input) numerical expression  
  x2,      ! (input) numerical expression  
  ..  
)
```

**Arguments:**

$x1, x2, \dots$   
Multiple numerical expressions.

**Return value:**

The function Max returns the largest number among  $x1, x2, \dots$ .

**Remarks:**

The function Max can be used in constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems if the first order derivatives of two arguments between which the Max function switches are discontinuous.

**See also:**

The function [Min](#). Arithmetic functions are discussed in full detail in [Section 6.1.4](#) of the Language Reference.

---

**Min**

```
Min(  
  x1,      ! (input) numerical expression  
  x2,      ! (input) numerical expression  
  ..  
)
```

**Arguments:**

$x1, x2, \dots$   
Multiple numerical expressions.

**Return value:**

The function `Min` returns the smallest number among  $x1, x2, \dots$ .

**Remarks:**

The function `Min` can be used in constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems if the first order derivatives of two arguments between which the `Min` function switches are discontinuous.

**See also:**

The function `Max`. Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Mod**

```
Mod(  
  x,      ! (input) numerical expression  
  y      ! (input) numerical expression  
)
```

**Arguments:**

- $x$   
A scalar numerical expression.
- $y$   
A scalar numerical expression unequal to 0.

**Return value:**

The function Mod returns the remainder of  $x$  after division by  $|y|$ . For  $y > 0$ , the result is an integer in the range  $0, \dots, y - 1$  if both  $x$  and  $y$  are integers, or in the interval  $[0, y)$  otherwise. For  $y < 0$ , the result is an integer in the range  $y - 1, \dots, 0$  if both  $x$  and  $y$  are integers, or in the interval  $(y, 0]$  otherwise.

**Remarks:**

- A run-time error results if  $y$  equals 0.
- The function Mod can be used in constraints of mathematical programs. However, nonlinear solver may experience convergence problems if  $x$  assumes values around multiples of  $y$ .

**See also:**

Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Power**

```
Power(  
  x,      ! (input) numerical expression  
  y      ! (input) numerical expression  
)
```

**Arguments:**

$x$   
A scalar numerical expression.

$y$   
A scalar numerical expression.

**Return value:**

The function `Power` returns  $x$  raised to the power  $y$ .

**Remarks:**

- The following combination of arguments is allowed:
  - $x > 0$
  - $x = 0$  and  $y > 0$
  - $x < 0$  and  $y$  integerIn all other cases a run-time error will result.
- The function can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions `Cube`, `Sqr`, and `Sqrt`. Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## Precision

```
Precision(  
  x,          ! (input) numerical expression  
  y          ! (input) integer expression  
)
```

### Arguments:

*x*  
A scalar numerical expression.

*y*  
An integer expression.

### Return value:

The function `Precision` returns *x* rounded to *y* significant digits.

### Remarks:

- The function `Precision` can be used in constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems around the discontinuities of the `Precision` function.
- When the numerical expression contains a unit, the function `Precision` will first convert the expression to the corresponding base unit, before evaluating the function itself.

### See also:

The functions `Round`, `Ceil`, `Floor`, `Trunc`. Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference.

---

## Radians

```
Radians(  
    x          ! (input) numerical expression  
)
```

### Arguments:

x  
A scalar numerical expression.

### Return value:

The function Radians returns the value of  $x$  converted from degrees to radians.

### Remarks:

The function Radians can be used in constraints of linear and nonlinear mathematical programs.

### See also:

The function [Degrees](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

## Round

```
Round(  
  x,          ! (input) numerical expression  
  decimals    ! (optional) integer expression  
)
```

### Arguments:

*x*  
A scalar numerical expression.

*decimals (optional)*  
An integer expression.

### Return value:

The function `Round` returns the integer value nearest to  $x$ . In the presence of the optional argument  $n$  the function `Round` returns the value of  $x$  rounded to  $n$  decimal places left (*decimals* < 0) or right (*decimals* > 0) of the decimal point.

### Remarks:

- The function `Round` can be used in constraints of nonlinear mathematical programs. However, nonlinear solvers may experience convergence problems around the discontinuities of the `Round` function.
- When the numerical expression contains a unit, the function `Round` will first convert the expression to that unit, before evaluating the function itself. See also the option `rounding compatibility` in the option category `backward compatibility`.

### See also:

The functions `Precision`, `Ceil`, `Floor`, `Trunc`. Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference.

---

## ScalarValue

```
ScalarValue(  
  identifier, ! (input) element expression into AllIdentifiers  
  suffix      ! (optional) element expression into AllSuffixNames  
)
```

### Arguments:

*identifier*

A scalar element expression into AllIdentifiers

*suffix*

A scalar element expression into AllIdentifiers

### Return value:

The function `ScalarValue` returns the value contained in the scalar identifier *identifier* or scalar reference *identifier.suffix*.

### Remarks:

When *identifier* or *identifier.suffix* is not a scalar numerical valued reference, the function `ScalarValue` returns 0.0.

### See also:

The function `Val`.

The `ScalarValue` function is a function that operates on subsets of AllIdentifiers. Other functions that operate on subsets of AllIdentifiers are referenced in Section [23.4](#) of the Language Reference.

---

**Sign**

```
Sign(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function `Sign` returns  $+1$  if  $x > 0$ ,  $-1$  if  $x < 0$  and  $0$  if  $x = 0$ .

**Remarks:**

The function `Sign` can be used in constraints of nonlinear mathematical programs. However, nonlinear solver may experience convergence problems round 0.

**See also:**

The function `Abs`. Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Sin**

```
Sin(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The Sin function returns the sine of  $x$  in the range  $-1$  to  $1$ .

**Remarks:**

The function Sin can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Cos](#), [Tan](#), [ArcSin](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Sinh**

```
Sinh(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The Sinh function returns the hyperbolic sine of x in the range  $-\infty$  to  $\infty$ .

**Remarks:**

The function Sinh can be used in the constraints of nonlinear mathematical programs.

**See also:**

The functions [Cosh](#), [Tanh](#), [ArcSinh](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Sqr**

```
Sqr(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Sqr returns  $x^2$ .

**Remarks:**

The function Sqr can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Power](#), [Cube](#), and [Sqrt](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Sqrt**

```
Sqrt(  
  x          ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression in the range  $[0, \infty)$ .

**Return value:**

The function Sqrt returns the  $\sqrt{x}$ .

**Remarks:**

- A run-time error results if  $x$  is outside the range  $[0, \infty)$ .
- The function Sqrt can be used in the constraints of nonlinear mathematical programs.

**See also:**

The functions [Power](#), [Cube](#), and [Sqr](#). Arithmetic functions are discussed in full detail in [Section 6.1.4](#) of the Language Reference.

---

**Tan**

```
Tan(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The Tan function returns the tangent of  $x$  in the range  $-\infty$  to  $\infty$ .

**Remarks:**

The function Tan can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Cos](#), [Sin](#), [ArcTan](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Tanh**

```
Tanh(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The Tanh function returns the hyperbolic tangent of  $x$  in the range  $-1$  to  $1$ .

**Remarks:**

The function Tanh can be used in constraints of nonlinear mathematical programs.

**See also:**

The functions [Cosh](#), [Sinh](#), [ArcTanh](#). Arithmetic functions are discussed in full detail in Section [6.1.4](#) of the Language Reference.

---

**Trunc**

```
Trunc(  
  x      ! (input) numerical expression  
)
```

**Arguments:**

x  
A scalar numerical expression.

**Return value:**

The function Trunc returns the truncated value of x:  $\text{Sign}(x) \cdot \text{Floor}(\text{Abs}(x))$ .

**Remarks:**

- The function Trunc will round to the nearest integer, if it lies within the equality tolerances `equality_absolute_tolerance` and `equality_relative_tolerance`.
- The function Trunc can be used in the constraints of nonlinear mathematical programs. However, nonlinear solver may experience convergence problems around integer argument values.
- When the numerical expression contains a unit, the function Trunc will first convert the expression to the corresponding base unit, before evaluating the function itself.

**See also:**

The functions [Ceil](#), [Floor](#), [Round](#), [Precision](#). Arithmetic functions are discussed in full detail in Section 6.1.4 of the Language Reference. Numeric tolerances are discussed in Section 6.2.2 of the Language Reference.

---

**Val**

```
Val(  
    str    ! (input) string or element expression  
)
```

**Arguments:**

*str*  
A scalar string or element expression.

**Return value:**

The function `Val` returns the numerical value represented by the string or element *str*.

**Remarks:**

If *str* cannot be interpreted as a numerical value, a runtime error may occur, see option `suppress error messages` of `val` function.

**See also:**

The `Val` function is discussed in full detail in Section [5.2.2](#) of the Language Reference.