



METAS UncLib C# - User Reference V2.5.3

Michael Wollensack

February 2022

Contents

1	Introduction	2
2	Global uncertainty settings	2
2.1	Using Metas.UncLib	2
2.2	Additional global settings	2
3	Create an uncertainty object	3
4	Calculations with uncertainty objects	4
4.1	Math functions	4
4.2	Linear algebra	4
4.3	Numerical routines	5
5	Get properties of an uncertainty object	5
6	Storage functions	6
6.1	Store a computed uncertainty object	6
6.2	Reload a stored uncertainty object	6



METAS UncLib C# - User Reference V2.5.3

1 Introduction

This document is a quick reference sheet. For more details refer to the [METAS UncLib](#) help with the installation of the software.

The [METAS UncLib C#](#) library supports creation of uncertainty objects and subsequent calculation with them as well as storage of the results. It's able to handle complex-valued and multivariate quantities. It has been developed with Microsoft Visual Studio 2013 and it requires the .NET Frame-work V4.5. There are three namespaces for uncertainty propagation: [LinProp](#), [DistProp](#) and [MCProp](#).

LinProp supports linear uncertainty propagation $\mathbf{V}_{out} = \mathbf{J}\mathbf{V}_{in}\mathbf{J}'$.

DistProp supports higher order uncertainty propagation, i.e. higher order terms of the Taylor expansion of the measurement equation are taken into account.¹

MCProp supports Monte Carlo propagation.¹

2 Global uncertainty settings

2.1 Using Metas.UncLib

`using Metas.UncLib.Core;` Use the linear uncertainty propagation.
`using Metas.UncLib.LinProp;`

`using Metas.UncLib.Core;` Use the higher order uncertainty propagation.
`using Metas.UncLib.DistProp;`

`using Metas.UncLib.Core;` Use the Monte Carlo uncertainty propagation.
`using Metas.UncLib.MCProp;`

2.2 Additional global settings

`DistProp.Misc.Global.MaxLevel` Set the higher order uncertainty propagation maximum level. Default value: 1 (1 corresponds to [LinProp](#))

`MCProp.Misc.Global.n` Set the Monte Carlo uncertainty propagation sample size. Default value: 100000

¹preliminary implementation



3 Create an uncertainty object

Square brackets indicate vector or matrix.

`UncNumber x = new UncNumber(value);` Creates a new uncertain number without uncertainties.

`UncNumber x = new UncNumber(value, standard_unc, (idof));` Creates a new real uncertain number with value, standard uncertainty and inverse degrees of freedom (optional).

`UncNumber x = Unc.RealUncNumber(value, standard_unc, (idof));` Creates a new real uncertain number with value, standard uncertainty and inverse degrees of freedom (optional).

`Complex<UncNumber> x = Unc.ComplexUncNumber(value, [covariance], (idof));` Creates a new complex uncertain number. Covariance size: 2×2

`UncNumber[] x = Unc.RealUncArray([value], [covariance], (idof));` Creates a new real uncertain array. Covariance size: $N \times N$

`Complex<UncNumber>[] x = Unc.ComplexUncArray([value], [covariance], (idof));`
Creates a new complex uncertain array. Covariance size: $2N \times 2N$

`UncNumber x = Unc.RealUncNumberFromSamples([samples], (desc), (p));` Creates a new real uncertain number from samples with a description (optional) and a probability (optional).

`Complex<UncNumber> x = Unc.ComplexUncNumberFromSamples([samples], (desc), (p));`
Creates a new complex uncertain number from samples with a description (optional) and a probability (optional). The complex uncertain number contains the correlation between real and imaginary parts.

`UncNumber[] x = Unc.RealUncArrayFromSamples([samples], (desc), (p));` Creates a new real uncertain array from samples with a description (optional) and a probability (optional). The real uncertain array contains the correlation between the different entries.

`Complex<UncNumber>[] x = Unc.ComplexUncArrayFromSamples([samples], (desc), (p));`
Creates a new complex uncertain array from samples with a description (optional) and a probability (optional). The complex uncertain array contains the correlation between real and the imaginary parts and the different entries.

`UncNumber x = new UncNumber(); x.Init(value, [sys_inputs], [sys_sensitivities]);`
Creates a new uncertain number by setting sensitivities with respect to uncertain inputs.²

²[LinProp](#) uncertainty objects only



4 Calculations with uncertainty objects

4.1 Math functions

- $x + y$
- $x - y$
- $\text{Math.Sqrt}(x)$
- $\text{Math.Exp}(x)$
- $\text{Math.Log}(x)$
- $\text{Math.Log10}(x)$
- $\text{Math.Log}(x, y)$
- $\text{Math.Pow}(x, y)$
- $x * y$
- x / y
- $\text{Math.Sin}(x)$
- $\text{Math.Cos}(x)$
- $\text{Math.Tan}(x)$
- $\text{Math.Asin}(x)$
- $\text{Math.Acos}(x)$
- $\text{Math.Atan}(x)$
- $-x$
- $\text{Math.Sinh}(x)$
- $\text{Math.Cosh}(x)$
- $\text{Math.Tanh}(x)$
- $\text{Math.Asinh}(x)$
- $\text{Math.Acosh}(x)$
- $\text{Math.Atanh}(x)$
- $\text{Math.Sign}(x)$
- $\text{Math.Real}(x)$
- $\text{Math.Imag}(x)$
- $\text{Math.Abs}(x)$
- $\text{Math.Angle}(x)$
- $\text{Math.Conj}(x)$
- $\text{Math.Atan2}(x, y)$

4.2 Linear algebra

`LinAlg.Dot(M1, M2)` Matrix multiplication of matrix M_1 and M_2

`LinAlg.Lu(M)` LU decomposition of matrix M

`LinAlg.Det(M)` Determinate of matrix M

`LinAlg.Inv(M)` Matrix inverse of M

`LinAlg.Solve(A, Y)` Solve linear equation system: $Ax = y$

`LinAlg.LstSqrSolve(A, Y)` Least square solve over determined equation system

`LinAlg.WeightedLstSqrSolve(A, Y, W)` Weighted least square solve over determined equation system

`UncLinAlg.NonLinearEig(A)` Non-linear eigenvalue problem²: $A_0V + A_1VD + A_2VD^2 + \dots + A_{(n-1)}VD^{(n-1)} = 0$

²`LinProp` uncertainty objects only



4.3 Numerical routines

`NumLib.PolyFit(x, y, n)` Fit polynom to data

`NumLib.PolyVal(p, x)` Evaluate polynom

`NumLib.Interpolation(x, y, n, xx)` Interpolation

`NumLib.Interpolation2(x, y, n, xx)` Interpolation with linear uncertainty propagation

`NumLib.SplineInterpolation(x, y, xx, boundaries)` Spline interpolation

`NumLib.SplineInterpolation2(x, y, xx, boundaries)` Spline interpolation with linear uncertainty propagation

`NumLib.Integrate(x, y, n)` Integrate

`NumLib.SplineIntegrate(x, y, n)` Spline integrate

`NumLib.Fft(v)` Fast Fourier transformation

`NumLib.Ifft(v)` Inverse Fast Fourier transformation

5 Get properties of an uncertainty object

`Unc.GetValue(y)` Returns the expected value.

`Unc.GetFcnValue(y)` Returns the function value.

`Unc.GetStdUnc(y)` Computes the standard uncertainty.

`Unc.GetCoverageInterval(y, p)` Computes the coverage interval.

`Unc.GetMoment(y, n)` Computes the n-th central moment.

`Unc.GetCorrelation([y1 y2 ...])` Computes the correlation matrix.

`Unc.GetCovariance([y1 y2 ...])` Computes the covariance matrix.

`Unc.GetIDof(y)` Computes the inverse degrees of freedom.²

`1.0 / Unc.GetIDof(y)` Computes the degrees of freedom.²

`Unc.GetJacobi(y)` Returns the sensitivities to the virtual base inputs (with value 0 and uncertainty 1).²

`Unc.GetJacobi2(y, x)` Computes the sensitivities of y to the intermediate results x.²

`Unc.GetUncComponent(y, x)` Computes the uncertainty components of y with respect to x.²

²`LinProp` uncertainty objects only



6 Storage functions

6.1 Store a computed uncertainty object

`y.BinarySerialize(filepath)` Binary serializes uncertainty object `y` to file.

`Misc.Storage.BinarySerialize(y, filepath)` Binary serializes uncertainty object `y` to file.

`y.BinarySerializeToByteArray()` Binary serializes uncertainty object `y` to byte array.

`Misc.Storage.BinarySerializeToByteArray(y)` Binary serializes uncertainty object `y` to byte array.

`y.XmlSerialize(filepath)` XML serializes uncertainty object `y` to file.

`Misc.Storage.XmlSerialize(y, filepath)` XML serializes uncertainty object `y` to file.

`y.XmlSerializeToString()` XML serializes uncertainty object `y` to string.

`Misc.Storage.XmlSerializeToString(y)` XML serializes uncertainty object `y` to string.

6.2 Reload a stored uncertainty object

`y.BinaryDeserialize(filepath)` Reloads uncertainty object from binary file.

`Misc.Storage.BinaryDeserialize<T>(filepath)` Reloads uncertainty object from binary file.

`y.BinaryDeserializeFromByteArray(d)` Reloads uncertainty object from byte array.

`Misc.Storage.BinaryDeserializeFromByteArray<T>(d)` Reloads uncertainty object from byte array.

`y.XmlDeserialize(filepath)` Reloads uncertainty object from XML file.

`Misc.Storage.XmlDeserialize<T>(filepath)` Reloads uncertainty object from XML file.

`y.XmlDeSerializeFromString(s)` Reloads uncertainty object from XML string.

`Misc.Storage.XmlDeSerializeFromString<T>(s)` Reloads uncertainty object from XML string.