Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Federal Institute of Metrology METAS

METAS UncLib - Data Formats V2.8.4

Michael Wollensack

April 2025

Contents

1 XML Schema

The XML format for METAS UncLib is developed by METAS. It's described using an XML schema. See https://www.w3schools.com/xml/schema_intro.asp for more details about XML schemas.

The following listing shows an example of a LinProp uncertainty number which has two dependencies:

```
<?xml version="1.0" encoding="utf-16"?>
<UncNumber xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Value>5</Value>
  <Dependencies>
    <DependsOn>
      <Input>
        <Id>5C-09-F4-58-A1-4A-40-D3-B5-14-13-94-AE-BC-CA-25</Id>
        <Description>side a</Description>
        <Distribution xsi:type="Normal">
          <mu>3</mu>
          <sigma>0.3</sigma>
        </Distribution>
      </Input>
      <Jacobi>0.6</Jacobi>
    </DependsOn>
    <DependsOn>
      <Input>
        <Id>EA-BF-67-4B-28-32-40-84-A4-83-64-30-1D-7C-6A-F5</Id>
        <Description>side b</Description>
        <Distribution xsi:type="Normal">
          <mu>4</mu>
          <sigma>0.4</sigma>
        </Distribution>
      </Input>
      <Jacobi>0.8</Jacobi>
    </DependsOn>
  </Dependencies>
</UncNumber>
```

The 'LinPropUncNumberType' is defined in section 1.4. It contains the value and a list of dependencies, where each dependency contains an input and a sensitivity to that input. The 'InputType' is defined in section 1.3. An input contains an identifier, a description and a distribution. The 'IdType' is defined in section 1.1 and the distribution types are defined in section 1.2.

1.1 ID Type

The following listing shows the XML schema for 'IdType':

```
  <!-- definition of IdType -->
  <xs:simpleType name="IdType">
    <!-- unique identifier -->
    <xs:restriction base="xs:string">
      <xs:pattern value="([0-9a-fA-F][0-9a-fA-F][\-]?)+"/>
    </xs:restriction>
  </xs:simpleType>
```

1.2 Distribution Types

The following listing shows the XML schema for ‘Distribution’:

```
<!-- definition of Distribution -->
<xs:complexType name="Distribution" abstract="true"/>
```

1.2.1 Standard Normal

The following listing shows the XML schema for ‘StandardNormal’:

```
<!-- definition of StandardNormal -->
<xs:complexType name="StandardNormal">
  <!-- mu = 0, sigma = 1 -->
  <!-- mean = mu -->
  <!-- stdunc = sigma -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.2 Normal

The following listing shows the XML schema for ‘Normal’:

```
<!-- definition of Normal -->
<xs:complexType name="Normal">
  <!-- mean = mu -->
  <!-- stdunc = sigma -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="mu" type="xs:double" />
        <xs:element name="sigma" type="xs:double" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.3 Standard Uniform

The following listing shows the XML schema for ‘StandardUniform’:

```
<!-- definition of StandardUniform -->
<xs:complexType name="StandardUniform">
  <!-- a = 0, b = 1 -->
  <!-- mean = (a + b) / 2 -->
  <!-- stdunc = (b - a) / sqrt(12) -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.4 Uniform

The following listing shows the XML schema for ‘Uniform’:

```
  <!-- definition of Uniform -->
  <xs:complexType name="Uniform">
    <!-- mean = (a + b) / 2 -->
    <!-- stdunc = (b - a) / sqrt(12) -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="a" type="xs:double" />
          <xs:element name="b" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.5 Curvilinear Trapezoid

The following listing shows the XML schema for ‘CurvilinearTrapezoid’:

```
  <!-- definition of CurvilinearTrapezoid -->
  <xs:complexType name="CurvilinearTrapezoid">
    <!-- mean = (a + b) / 2 -->
    <!-- stdunc = sqrt((b - a)^2 / 12 + d^2 / 9) -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="a" type="xs:double" />
          <xs:element name="b" type="xs:double" />
          <xs:element name="d" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.6 Trapezoidal

The following listing shows the XML schema for ‘Trapezoidal’:

```
  <!-- definition of Trapezoidal -->
  <xs:complexType name="Trapezoidal">
    <!-- mean = (a + b) / 2 -->
    <!-- stdunc = (b - a) * sqrt((1 + beta^2) / 24) -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="a" type="xs:double" />
          <xs:element name="b" type="xs:double" />
          <xs:element name="beta" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.7 Triangular

The following listing shows the XML schema for ‘Triangular’:

```
  <!-- definition of Triangular -->
  <xs:complexType name="Triangular">
    <!-- mean = (a + b) / 2 -->
    <!-- stdunc = (b - a) / sqrt(24) -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="a" type="xs:double" />
          <xs:element name="b" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.8 Arc Sine

The following listing shows the XML schema for ‘ArcSine’:

```
  <!-- definition of ArcSine -->
  <xs:complexType name="ArcSine">
    <!-- mean = (a + b) / 2 -->
    <!-- stdunc = (b - a) / sqrt(8) -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="a" type="xs:double" />
          <xs:element name="b" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.9 Exponential

The following listing shows the XML schema for ‘Exponential’:

```
  <!-- definition of Exponential -->
  <xs:complexType name="Exponential">
    <!-- mean = mu -->
    <!-- stdunc = mu -->
    <xs:complexContent>
      <xs:extension base="Distribution">
        <xs:sequence>
          <xs:element name="mu" type="xs:double" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

1.2.10 Gamma

The following listing shows the XML schema for ‘Gamma’:

```
<!-- definition of Gamma -->
<xs:complexType name="Gamma">
  <!-- mean = a * b -->
  <!-- stdunc = sqrt(a) * b -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="a" type="xs:double" />
        <xs:element name="b" type="xs:double" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.11 Chi Squared

The following listing shows the XML schema for ‘ChiSquared’:

```
<!-- definition of ChiSquared -->
<xs:complexType name="ChiSquared">
  <!-- mean = k -->
  <!-- stdunc = sqrt(2 * k) -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="k" type="xs:double" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.12 Student T

The following listing shows the XML schema for ‘StudentT’:

```
<!-- definition of StudentT -->
<xs:complexType name="StudentT">
  <!-- mean = mu for n > 2 with n = dof + 1 -->
  <!-- stdunc = sigma * sqrt((n - 1) / (n - 3)) for n > 3 with n = dof + 1 -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="mu" type="xs:double" />
        <xs:element name="sigma" type="xs:double" />
        <xs:element name="dof" type="xs:double" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.13 Student T from Samples

The following listing shows the XML schema for ‘StudentTFromSamples’:

```
<!-- definition of StudentTFromSamples -->
<xs:complexType name="StudentTFromSamples">
  <!-- mean = mean(samples) for n > 2 with n = dof + 1 -->
  <!-- stdunc = sigma * sqrt((n - 1) / (n - 3)) for n > 3 with n =
      length(samples) and sigma = std(samples) / sqrt(n) -->
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="Samples">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="Sample" type="xs:double" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.2.14 Random Choices from Samples

The following listing shows the XML schema for ‘RandomChoicesFromSamples’:

```
<!-- definition of RandomChoicesFromSamples -->
<xs:complexType name="RandomChoicesFromSamples">
  <xs:complexContent>
    <xs:extension base="Distribution">
      <xs:sequence>
        <xs:element name="Seed" type="IdType" />
        <xs:element name="Samples">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="Sample" type="xs:double" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.3 Input Type

The following listing shows the XML schema for ‘InputType’:

```
<!-- definition of InputType -->
<xs:complexType name="InputType">
  <xs:sequence>
    <xs:element name="Id" type="IdType" />
    <xs:element name="Description" type="xs:string" />
    <xs:choice>
      <xs:element name="Distribution" type="Distribution" />
      <!-- obsolete -->
      <xs:element name="IDof" type="xs:double" />
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

The ‘IdType’ is defined in section 1.1 and the distribution types are defined in section 1.2.

1.4 LinProp UncNumber Type

The following listing shows the XML schema for ‘LinPropUncNumberType’:

```
<!-- definition of LinPropUncNumberType -->
<xs:complexType name="LinPropUncNumberType">
  <xs:sequence>
    <xs:element name="Value" type="xs:double" />
    <xs:element name="Dependencies">
      <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded"
              name="DependsOn">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Input" type="InputType" />
                <xs:element name="Jacobi" type="xs:double" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The ‘InputType’ is defined in section 1.3. An input contains an identifier, a description and a distribution.

1.5 MCProp UncNumber Type

The following listing shows the XML schema for ‘MCPropUncNumberType’:

```
<!-- definition of MCPropUncNumberType -->
<xs:complexType name="MCPropUncNumberType">
  <xs:sequence>
    <xs:element name="FunctionValue" type="xs:double" />
    <xs:element name="Values">
      <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="Value"
              type="xs:double" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Dependencies">
      <xs:complexType>
```

```
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="DependsOn">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Input" type="InputType" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

1.6 LinProp ComplexUncNumber Type

The following listing shows the XML schema for ‘LinPropComplexUncNumberType’:

```
<!-- definition of LinPropComplexUncNumberType -->
<xs:complexType name="LinPropComplexUncNumberType">
  <xs:sequence>
    <xs:element name="Real" type="LinPropUncNumberType" />
    <xs:element name="Imag" type="LinPropUncNumberType" />
  </xs:sequence>
</xs:complexType>
```

1.7 MCProp ComplexUncNumber Type

The following listing shows the XML schema for ‘MCPropComplexUncNumberType’:

```
<!-- definition of MCPropComplexUncNumberType -->
<xs:complexType name="MCPropComplexUncNumberType">
  <xs:sequence>
    <xs:element name="Real" type="MCPropUncNumberType" />
    <xs:element name="Imag" type="MCPropUncNumberType" />
  </xs:sequence>
</xs:complexType>
```

1.8 N-Array Types

The following listing shows the XML schema for ‘NArrayType’:

```
<!-- definition of NArrayType -->
<xs:complexType name="NArrayType" abstract="true">
  <xs:sequence>
    <xs:element name="NDims" type="xsd:int" />
    <xs:element name="Size">
      <xs:complexType>
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="int" type="xs:int" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

1.8.1 LinProp Real N-Array Type

The following listing shows the XML schema for ‘LinPropRealNArrayType’:

```
<!-- definition of LinPropRealNArrayType -->
<xs:complexType name="LinPropRealNArrayType">
  <xs:complexContent>
    <xs:extension base="NArrayType">
      <xs:sequence>
        <xs:element name="Data">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="UncNumber" type="LinPropUncNumberType" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.8.2 MCProp Real N-Array Type

The following listing shows the XML schema for ‘MCPropRealNArrayType’:

```
<!-- definition of MCPropRealNArrayType -->
<xs:complexType name="MCPropRealNArrayType">
  <xs:complexContent>
    <xs:extension base="NArrayType">
      <xs:sequence>
        <xs:element name="Data">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="UncNumber" type="MCPropUncNumberType" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.8.3 LinProp Complex N-Array Type

The following listing shows the XML schema for ‘LinPropComplexNArrayType’:

```
<!-- definition of LinPropComplexNArrayType -->
<xs:complexType name="LinPropComplexNArrayType">
  <xs:complexContent>
    <xs:extension base="NArrayType">
```

```
      <xs:sequence>
        <xs:element name="Data">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="ComplexOfUncNumber"
                  type="LinPropComplexUncNumberType" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.8.4 MCProp Complex N-Array Type

The following listing shows the XML schema for ‘MCPropComplexNArrayType’:

```
<!-- definition of MCPropComplexNArrayType -->
<xs:complexType name="MCPropComplexNArrayType">
  <xs:complexContent>
    <xs:extension base="NArrayType">
      <xs:sequence>
        <xs:element name="Data">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="0" maxOccurs="unbounded"
                  name="ComplexOfUncNumber"
                  type="MCPropComplexUncNumberType" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

1.9 Elements

1.9.1 LinProp Elements

The following listing shows the XML schema for LinProp elements:

```
<!-- definition of LinProp elements -->
<xs:element name="LinPropUncNumber" type="LinPropUncNumberType" />
<xs:element name="LinPropComplexOfUncNumber"
    type="LinPropComplexUncNumberType" />
<xs:element name="LinPropRealNArray" type="LinPropRealNArrayType" />
<xs:element name="LinPropComplexNArray" type="LinPropComplexNArrayType"
    />
```

1.9.2 MCProp Elements

The following listing shows the XML schema for MCProp elements:

```
<!-- definition of MCProp elements -->
<xs:element name="MCPropUncNumber" type="MCPropUncNumberType" />
<xs:element name="MCPropComplexOfUncNumber" type="MCPropComplexUncNumberType" />
<xs:element name="MCPropRealNArray" type="MCPropRealNArrayType" />
<xs:element name="MCPropComplexNArray" type="MCPropComplexNArrayType" />
```

1.9.3 Default Elements

The following listing shows the XML schema for default elements:

```
<!-- definition of default elements -->
<xs:element name="UncNumber" type="LinPropUncNumberType" />
<xs:element name="ComplexOfUncNumber" type="LinPropComplexUncNumberType" />
<xs:element name="RealNArray" type="LinPropRealNArrayType" />
<xs:element name="ComplexNArray" type="LinPropComplexNArrayType" />
```

2 Binary LinProp

The version 1 data formats assume a standard normal distribution for the input of each dependencies. The version 2 data formats allow to specify the type of distribution and its parameters for the input of each dependencies.

2.1 UncNumber Version 1

The following enumeration describes the binary structure version 1 of ‘UncNumber’:

1. Version (int32), value: 1
2. Value (double)
3. Version2 (int32), value: 4
4. Number of Dependencies (int32)
5. Dependencies (DependsOn[]), size: number of dependencies

2.1.1 Dependencies

The following enumeration describes the binary structure of ‘DependsOn’:

1. Number of Id Bytes (int32)
2. Input Id (byte[]), size: number of id bytes
3. Input Description (string)
4. Input IDof (double)
5. Jacobi (double)

2.2 UncNumber Version 2

The following enumeration describes the binary structure version 2 of ‘UncNumber’:

1. Version (7-bit encoded int), value: 2
2. Value (double)
3. Number of Dependencies (7-bit encoded int)
4. Dependencies (DependsOn2[]), size: number of dependencies

2.2.1 Dependencies

The following enumeration describes the binary structure of ‘DependsOn2’:

1. Input (InputDistribution)
2. Jacobi (double)

2.2.2 Input Distribution

The following enumeration describes the binary structure of ‘InputDistribution’:

1. Version (7-bit encoded int), value: 2
2. Number of Id Bytes (7-bit encoded int)
3. Id (byte[]), size: number of id bytes
4. Description (string)
5. Distribution (Distribution)

2.2.3 Distributions

The following enumeration describes the binary structure of ‘StandardNormalDistribution’:

1. Type (7-bit encoded int), value: 0

The following enumeration describes the binary structure of ‘NormalDistribution’:

1. Type (7-bit encoded int), value: 1
2. mu (double)
3. sigma (double)

The following enumeration describes the binary structure of ‘StandardUniformDistribution’:

1. Type (7-bit encoded int), value: 2

The following enumeration describes the binary structure of ‘UniformDistribution’:

1. Type (7-bit encoded int), value: 3
2. a (double)
3. b (double)

The following enumeration describes the binary structure of ‘CurvilinearTrapezoidDistribution’:

1. Type (7-bit encoded int), value: 4
2. a (double)
3. b (double)
4. d (double)

The following enumeration describes the binary structure of ‘TrapezoidalDistribution’:

1. Type (7-bit encoded int), value: 5
2. a (double)

3. b (double)
4. beta (double)

The following enumeration describes the binary structure of ‘TriangularDistribution’:

1. Type (7-bit encoded int), value: 6
2. a (double)
3. b (double)

The following enumeration describes the binary structure of ‘ArcSineDistribution’:

1. Type (7-bit encoded int), value: 7
2. a (double)
3. b (double)

The following enumeration describes the binary structure of ‘GammaDistribution’:

1. Type (7-bit encoded int), value: 8
2. a (double)
3. b (double)

The following enumeration describes the binary structure of ‘ChiSquaredDistribution’:

1. Type (7-bit encoded int), value: 9
2. k (int32)

The following enumeration describes the binary structure of ‘StudentTDistribution’:

1. Type (7-bit encoded int), value: 10
2. mu (double)
3. sigma (double)
4. dof (double)

The following enumeration describes the binary structure of ‘StudentTFromSamplesDistribution’:

1. Type (7-bit encoded int), value: 11
2. Version (7-bit encoded int), value: 2
3. Number of Samples (7-bit encoded int)
4. Samples (double[]), size: number of samples

The following enumeration describes the binary structure of ‘RandomChoicesFromSamples’:

1. Type (7-bit encoded int), value: 99
2. Version (7-bit encoded int), value: 2
3. Number of Seed Bytes (7-bit encoded int)
4. Seed (byte[]), size: number of seed bytes
5. Number of Samples (7-bit encoded int)
6. Samples (double[]), size: number of samples

2.3 Complex UncNumber Version 1

The following enumeration describes the binary structure of ‘ComplexUncNumber’:

1. Version (int32), value: 1
2. Real (UncNumber)
3. Imag (UncNumber)

‘UncNumber’ could be either of version 1 or 2.

2.4 Flat Vector of UncNumbers Version 1

The following enumeration describes the binary structure version 1 of ‘FlatVectorUncNumbers’:

1. Version (int32), value: 1
2. Length (7-bit encoded int)
3. Values (double[]), size: length
4. Number of Inputs (7-bit encoded int)
5. Inputs (UncInput[]), size: number of inputs
6. Dependencies (UncDependencies[]), size: length

2.4.1 Input

The following enumeration describes the binary structure of ‘UncInput’:

1. Temp (byte), bit 0: same id size, bit 1: empty description, bit 2: zero idof, bit 3-7: 0
2. Id Size (7-bit encoded int), field only present if not same id size
3. Input Id (byte []), size: id size
4. Input Description (string), field only present if not empty description
5. Input IDof (double), field only present if not zero idof

2.4.2 Dependencies

The following enumeration describes the binary structure of ‘UncDependencies’

1. Number of Dependencies (7-bit encoded int)
2. Dependencies (UncDependency[]), size: number of dependencies, pointer to inputs set to 0.

The following enumeration describes the binary structure of ‘UncDependency’

1. Relative Pointer to Inputs (7-bit encoded int)
2. Jacobi (double)

2.5 Flat Vector of UncNumbers Version 2

The following enumeration describes the binary structure version 2 of ‘FlatVectorUncNumbers’:

1. Version (7-bit encoded int), value: 2
2. Length (7-bit encoded int)
3. Values (double[]), size: length
4. Number of Inputs (7-bit encoded int)
5. Inputs (InputDistribution[]), size: number of inputs
6. Dependencies (UncDependencies[]), size: length

For the definition of ‘InputDistribution’ see section 2.2.2. For the definition of ‘UncDependencies’ see section 2.4.2.

3 Binary MCProp

3.1 UncNumber Version 1

The following enumeration describes the binary structure version 1 of ‘UncNumber’:

1. Version (int32), value: 1
2. Function Value (double)
3. Number of Values (int32)
4. Values (double[]), size: number of values

3.2 UncNumber Version 4

Version 2 and 3 had an bug and they are not used. The following enumeration describes the binary structure version 4 of ‘UncNumber’:

1. Version (int32), value: 4
2. Function Value (double)
3. Number of Values (int32)
4. Values (double[]), size: number of values
5. Number of Dependencies (int32)
6. Dependencies (MCPropDependencies[]), size: number of dependencies

3.2.1 Dependencies

The following enumeration describes the binary structure of ‘MCPropDependencies’:

1. Input (InputDistribution)

For the definition of ‘InputDistribution’ see section 2.2.2.

3.3 Complex UncNumber Version 1

The following enumeration describes the binary structure of ‘ComplexUncNumber’:

1. Version (int32), value: 1
2. Real (UncNumber)
3. Imag (UncNumber)

‘UncNumber’ could be either of version 1 or 4.