Diss. ETH No. 18676

A Modeling Language for Measurement Uncertainty Evaluation

A dissertation submitted to the
Swiss Federal Institute of Technology, Zürich

for the degree of
Doctor of Sciences

presented by
Marco Wolf

Dipl.-Inf. TU-München
born on October 28th, 1975
citizen of Austria

accepted on the recommendation of
Prof. Dr. Walter Gander, examiner
Dr. Matthias Rösslein, co-examiner
Prof. Dr. Bertrand Meyer, co-examiner

2009

ISBN 978-1-4452-1512-9

In memory of my father
Günther Wolf

Zusammenfassung

Die Berechnung der Messunsicherheit ist notwendig, um die Zuverlässigkeit und Genauigkeit einer Messung zusammen mit dem Messwert festlegen zu können. Nur so kann das Resultat auch für weitere Messungen oder für Vergleiche mit anderen Messungen herangezogen werden. Das Ziel dieser Arbeit ist es, Spezialisten aus dem Gebiet der Metrologie ein Simulationssystem zur Verfügung zu stellen, mit dessen Hilfe die Messunsicherheit für sehr detaillierte Messszenarien berechnet werden kann. Eine klare Sprachdefinition, die gebräuchliche Terme aus dem wissenschaftlichen Gebiet verwendet, unterstützt strukturiertes und organisiertes Modellieren von unterschiedlichsten Messszenarien. Dabei wurde die Möglichkeit einer grafischen Benutzeroberfläche von Beginn an berücksichtigt. Außerdem werden unterschiedliche Ansätze zur Optimierung von Messszenarien diskutiert. Eine Methode befasst sich mit der Suche nach guten Sequenzen in Messserien, wobei die Anzahl benötigter Referenzmessungen – und somit auch der Aufwand – minimiert werden soll. Eine zweite Methode extrahiert die wichtigsten Einflussgrößen einer Messung mit Hilfe der Sensitivitätsanalyse. Diverse praktische Beispiele aus Physik und Chemie werden verwendet, um Modellierungskonzepte zu erklären und konkrete Anwendungen des Systems zu veranschaulichen.

Abstract

The evaluation of measurement uncertainty is necessary to report the accuracy of a measurement together with the value of a measurement. Only then the result can be used in further measurements or compared to other measurements. Goal of this project is to provide a simulation framework for specialists in the field of metrology to evaluate the measurement uncertainty for very detailed real-life measurements. A clear language definition using common terms of the scientific field supports structured and organized modeling of measurement scenarios. The possibility of a graphical user interface is considered from the beginning. Besides, different techniques to optimize measurement scenarios are introduces and discussed. One method is concerned with finding good sequences for series of measurements reducing the number of reference measurements and hence, the effort to a minimum. A second method extracts the most important influence quantities to a measurement applying sensitivity analysis. Various practical examples from the fields of physics and chemistry are used to explain modeling concepts and to discuss concrete applications of the system.

Acknowledgments

This work would not have been possible without the help of many people. I want to thank everybody for their support and in particular the people of the Measurement Uncertainty Research Group. Professor Walter Gander gave me the possibility to realize this project at ETH in cooperation with the Empa St. Gallen and working on this interesting, very interdisciplinary thesis. I appreciate very much that Professor Bertrand Meyer, chair of the software engineering group, agreed to be in the committee of my defense. Dr. Matthias Rösslein supported me during more than five years, answered also the hard questions with infinite patience, giving me as computer scientist a deep insight into the field of chemistry. He also raised my motivation when it got too low from time to time. Finally, he read and re-read the thesis chapter after chapter, helping me improving the text up to the current state.

Martin Müller as my office college and friend played a major role in my thesis. He realized the simulation core of the project in a second thesis, concentrating on the mathematical aspects of measurement uncertainty evaluation. The work on the same project was not always easy as the two theses had quite different goals and hence, our priorities were not always the same, but in the end the cooperation was very fruitful and fun. I think the result is a really sophisticated and impressive simulation framework, which we can be proud of.

Harald Krug and Bruno Wampfler provided the necessary backup for the thesis as representatives of the Empa St. Gallen. Sergio Rezzonico and Roman Hedinger worked formerly at the Empa; their work concerning the UncertaintyManager and our discussions helped a lot in understanding the examples from analytical chemistry. Sergio also confirmed that the example of a titration measurement that accompanies the explanation of the language constructs appears maybe not in day-to-day work of every laboratory, but definitely as a special case in metrological laboratories. Thanks for that, too.

The project would not have been possible without an early meeting in Berlin even before my work on the thesis started. There I got to know Nell Sandrask, who was back then head in the Statistical Engineering Division of the National Institute of Standards and Technology (NIST), and Marc Salit, who is nowadays group leader in Biochemical Science Division, especially in multiplexed biomolecular science, also at NIST. My visits at the Statistical Engineering Division at the NIST brought up a lot of new ideas for the project. I have to thank explicitly Blaza Toman for her care during my stays and introducing me to her department.

The meetings with Marko Zeier on several conferences and with his group at METAS in Bern were always interesting and very motivating, pushing me more and more to the problems of interest. Marko has also the important ability to ask the right questions at the right time.

I want to thank Peter Harris and Maurice Cox from NPL in Teddington/UK for the invitation and the interest in my work. Important questions were always answered with very short delay and Peter provided me information for examples and gave me important hints concerning sensitivity analysis.

From ETH I want to thank Hasan Kaharan and Dominik Gabi, for their help in developing very special aspects of the framework during two semester works. Their work really improved usability and acceptance of the simulation framework. I also had very interesting discussions – not always concerning measurement uncertainty – with the former PhD students Erwin Achermann and Oscar Chinellato and current PhD students and colleges of the group Cyril Flaig, Dirk Zimmer, Markus Wittberger, and Pedro Gonnet.

I would like to express my gratitude to Christian Wirthensohn for beeing such a good friends over so many years and for proof-reading the whole text in so little time. I want to thank Andreas Parschalk for his attempt of proof-reading; it's the thought that counts.

I hope that I haven't forgotten anybody who was involved in this project in one way or another. Otherwise, I want to formally apologize and assure that it was not on purpose.

Finally, I want to thank my mother Ria Wolf, my sister Angelika and my brother-in-law Wolfgang for their support and understanding. My two lovely nephews Rene and Alex cheer me up everytime I see them. They really grew up during my time here in Zürich. This is how you recognize how long the project really lasted. Special thanks go to my beloved girlfriend Rebeca Sanchez, who had often a hard stand beside my work even at the weekends; she often complained, but fortunately not too much.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Maybe the best reason to design a new language is because of fun. Designing is fun. But it is also an important discipline for thought. J.R.R. Tolkien, the author of The Lord of the Rings*, in his professional youth as a philologist undertook to design a language he called "High Elvish". He designed the whole language, the lexicon, the syntax, the semantics, the whole bit. Somebody asked him, "Why go through a really vacuous kind of exercise of making up a natural language, High Elvish?" He said, "One doesn't really understand the bones of language until one has tried to design one."*

— Language Design as Design, Frederick P. Brooks, Jr. —

Measurements serve as a basis for decision making. These decisions affect us in professional as well as in private life. Hence, the question of the reliability and accuracy of measurement results is of great importance. Influences on the result of a measurement cannot be determined arbitrarily exactly due to physical and also financial reasons. Unfortunately, such sources of uncertainties can have a considerable impact on the result of a measurement. Thus, it is necessary to have a generalized method to express the confidence in the result. Only if all potential uncertainty sources are considered and quantified a comparison of different results and further use of the current results are reasonable.

Uncountable examples of various measurement techniques appear in many different scientific fields like chemistry, biology, and physics. To start with, I will give a few arbitrarily chosen examples of measurements with an estimated accuracy and possible consequences of deviations in the measurements.

- Current devices for radar speed checks on highways have an accuracy of better than ± 3 %. Law takes this into account to the advantage of the driver and instructs to reduce the measured speed by 3 km/h below 100 km/h or 3 % above 100 km/h. Besides, the tachometer in a car is never allowed to show less velocity than the car actually has. In Germany the upper limit is defined as 10 % + 4 km/h.

- Common kitchen scales have uncertainties between 0.1 g and 1 g, whereas high precision balances in chemical laboratories have accuracies to 0.01 mg; slight changes in temperature or electrostatics can influence such measurements.

- A common alcohol breath tester that measures alcohol consumption has an accuracy of 5 % for 1 per-mil. For blood alcohol analysis a security margin of 1 per-mil is applied to the advantage of the accused, which is usually twice or three times the expected uncertainty of the measurement.

- At the gas station the indicated amount of gasoline is subject to an uncertainty of about ± 1 %. Fluctuations in this range are allowed by the law; the idea is that it averages over all the gas stations and customers. A truck has a reservoir of about 400 liters and more. So if you assume a price of 1.2 Euros/liter you will pay between 475.20 Euros and 484.80 Euros for a 400 liter reservoir.

- The global positioning system (GPS) was developed by the United States Department of Defense and is capable of accuracies up to about 15 meters. The accuracy of devices is constantly improved to get to accuracies of just a few meters, so that navigation devices can not only be used in airplanes, ships and cars, but also for bicycles, hikers and pedestrians. The planned European Galileo system should reach an accuracy of 1 to 3 meters. More expensive GPS devices using for example Differential-GPS for land survey have an accuracy from 5 m down to 0.01 m.

- Devices to measure the consumption of electricity in private houses must have an uncertainty of less than 2 % by law. If you assume that a four-person household uses 4500 kWh per year with a price of 15 Cent/kWh, it would cost 675 Euros. The 2 % result in a maximum difference of 13.50 Euros. The customer is lucky if the device shows 2 % less than it should.

- There are of course more sensitive fields of measurements, where results do not only affect the wallet, but have direct consequences on a person's life. An example is the proof of drug consumption or doping in sports. It requires very accurate and reliable detection of infinitesimal amounts of substances in blood, as the result can decide about the life of people as well as millions of Euros. On 22th of July 2006 Floyd Landis, an American cyclist, finished stage 17 of the Tour the France with an outstanding performance. He won the Tour de France that year. Unfortunately, he was accused of doping as the French national anti doping laboratory found testosterone from an external source in his urine in two separate A and B samples. Doping tests from the earlier 16 stages of the tour were negative. Landis went to court, but lost his case. One of the publicly criticized points among others was that the report of the laboratory did not consider the measurement uncertainty in a proper way. Hence, a very important criterion for the quality of the measurement results is missing.

The selection of examples should give a feeling for the appearance and importance of measurements in various situations. Interesting questions concerning the confidence of measurement results arise from these examples.

- How can the uncertainty of a measurement be described adequately, and is there a – preferably standardized – way to quantify it?
- Can uncertainties of different measurements be compared adequately?
- What are the most important influences on a measurement and its uncertainty?
- Can influences be adjusted or corrected to improve the uncertainty?
- How do measurements behave that are performed over a longer time period in respect to repeatability and measurement uncertainty?

The scientific field to find answers to the questions regarding measurement uncertainty is *metrology*. The *International Bureau of Weights and Measures* (Bureau International des Poids et Mesures, BIPM) was founded to have a basis for a single and coherent system of measurements throughout the world, traceable to the International System of Units (SI). The BIPM defines the field of metrology as follows:

> Metrology is the science of measurement, embracing both experimental and theoretical determinations at any level of uncertainty in any field of science and technology.

The *Joint Committee for Guides of Metrology* (Comité commun pour les guides en métrologie, JCGM) is a working group of the BIPM that elaborated a document on measurement uncertainty evaluation, which answers at least some of our initial questions. The so-called *Guide to the expression of uncertainty in measurements* (GUM) [37] is an internationally accepted and widespread ISO document that describes an evaluation method – the so-called *GUM Uncertainty Framework* (GUF) – to estimate the uncertainty of the result of a measurement. It recommends to identify influences that affect the measurement uncertainty initially and quantify them afterwards in a statistical sense by assigning proper probability density functions (PDF) with suitable parameter settings to describe the uncertainty about the quantity. The functional relationship of the influence quantities are expressed using mathematical equations, the so-called *equation of the measurand.* GUF is appropriate for application for a limited set of measurement scenarios, but actually they appear very often in practice. The method is subject to some major restrictions. One shortcoming is that information about probability distributions used to describe influence quantities is rigorously reduced in only using the first two moments – mean value and standard deviation – of the distributions for evaluation. Another major restriction is that the measurement model has to be linear or must be linearized for evaluation. A consequence is that the measurement model is kept as simple as possible. Hence, it is recommended to use only the five or six most important influences for measurement uncertainty evaluation. Therefore, the decision with regard to which influence quantities can be neglected and which are important is no trivial task.

Because of the limitations for cases in which the classical GUM method could not be applied properly, the authors of the GUM were looking for an alternative evaluation method that could be used for a broader range of measurement scenarios. The result is a Monte Carlo approach and is nowadays the recommended way for calculating the measurement uncertainty for complex measurement scenarios in cases where the GUF method is not applicable or there is doubt with regard to the premises for applying GUF. The document *Supplement 1 to the "Guide to the expression of uncertainty in measurement" – Propagation of distributions using a Monte Carlo method* (GS1) [22] of JCGM has a long history with several drafts and intensive discussion over several years. Ultimately, it was released in its final version in mid-2008. It is not meant to replace the classical GUF approach, but should be applied for measurement uncertainty evaluation, where the use of GUF is limited by its restrictions. It is also recommended to compare and validate results from classical measurement uncertainty evaluation with the new approach in case of doubt.

Nowadays even normal desktop computers are capable of evaluating the measurement uncertainty for complex measurement scenarios with the Monte Carlo method in acceptable time. Moreover, uncertainty evaluation can go a step further to build more realistic, real-world measurement scenarios. The new capabilities of the new evaluation method of GS1 can only be handled by the use of an appropriate simulation framework. Therefore, the goal of this project is to create a simulation framework building up on an efficient core for calculating Monte Carlo simulations and providing a sophisticated modeling language that is easy to learn and understand and is capable of supporting experts from metrology in translating measurement models for uncertainty evaluation in a structured and organized way. Analyzing the importance of individual influence quantities to the measurement uncertainty is a major task as the number of influence quantities is very big for more realistic measurement scenarios. The system should be able to support decision making where to invest more time, money and effort to enhance the results of measurements and improving the quality of the results. Another interesting topic is simulation of different scenarios. This allows to analyze the consequences of a reduction of repeated measurements to get to an appropriate result with a small uncertainty, but still with low effort.

Goals of this thesis

In this section I want to summarize the goals of this thesis. The project goes in many points beyond the scope of existing measurement uncertainty evaluation concepts.

Real-world, high-resoluted modeling Using an advanced evaluation method, i.e. a Monte Carlo simulation system, non-linearities in the model need not be considered as a problem. The equation of the measurand can be used as is. The second advantage is that the full information provided for input quantities can be used for the measurement uncertainty evaluation.

Human-readable modeling language for hierarchical models The advanced models that result from the first point require proper organizing and structuring of the modeling process and models themselves. As we will see later on measurements can be seen as hierarchical structured models in principle. I made use of this concept in defining a domain-specific language which allows a step-by-step development for describing measurement models. The defined modeling language allows structured and well-organized modeling even for complex models with many influence quantities.

Uncertainty analysis with variation of parameter settings The modeling of varying influences should be possible; worsening or improvements with respect to the measurement uncertainty can be detected by the variation of input quantities. Variations of more than one input quantity at a time is possible. Uncertainty analysis is the first step in investigating the measurement in great detail.

Support for optimizing series of measurements In repeated measurements not only the result and measurement uncertainty of one measurement is taken into account, but a set of many individual measurements in sequential or parallel order, where some influence quantities may contribute to different measurements, is analyzed. Also, the number of necessary reference measurements in respect to a stable result and small measurement uncertainty should be analyzed.

Sensitivity analysis to find important influence quantities Different methods exist to extract influence quantities that contribute strongly to the measurement uncertainty. I will present different approaches to analyze the system behavior for individual influence quantities comparing screening factor methods with local and global sensitivity analysis methods.

Framework-related issues To be applicable in practice, a simulation framework for measurement uncertainty evaluation has to provide a possibility to validate the results of the calculation core. Hence, some kind of validation module is necessary that checks the basic functionality. Besides, a graphical user interface is of vital interest as people tend to avoid learning completely new programming languages. Nevertheless, the interface is no direct goal of the work, but should always be in mind during the definition of the system.

I want to stress once more the very interesting, but also very demanding, interdisciplinary approach of this thesis. It uses concepts of various scientific fields, starting naturally from metrology as science of measurements and its applications, which include chemistry, physics, biology and more, to statistics with differences in frequentist and Bayes approaches. Further on, computational and numerical aspects of Monte Carlo simulation and proper handling and analyzing of simulation data is necessary as well as programming expertise and knowledge about software engineering with concepts for domain-specific language definition and project planning.

CHAPTER 2

Measurement uncertainty evaluation

The objective of a measurement is to determine the value of the measurand, that is, the value of the particular quantity to be measured. A measurement therefore begins with an appropriate specification of the measurand, the method of measurement, and the measurement procedure.

— GUM, 1993, Section 3.1.1 —

Before we start a theoretical survey into the world of measurements and measurement uncertainties, I want to give a very practical understanding for what we understand as a measurement in the field of metrology and how we get to values for a measurand of interest. The principle of a single measurement is the comparison of attributes of a sample to a well-known reference to get a value for the measurand as visualized in Figure 2.1. If, for example, the length of a woodblock (sample) is of interest to an accuracy of one millimeter, we can use a measuring tape (measurement instrument) and compare the marks on the tape with the beginning and end of the woodblock to get the length (measurand). The measurement procedure is in this case the comparison of the marks on the tape with the woodblock. Of course the measuring tape, which we use as reference, must have a well-known length. Therefore, it is calibrated using a transfer standard,

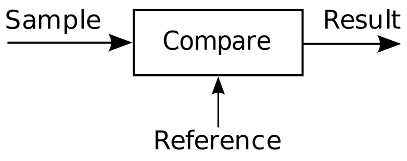


Figure 2.1: Generally speaking, in a measurement we assign a value to an attribute of a sample in comparing the sample with a reference.

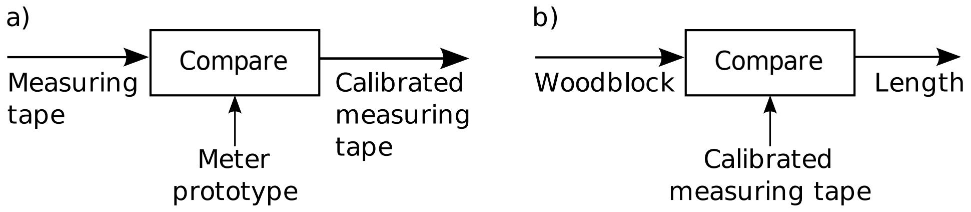


Figure 2.2: a) *reference measurement:* the marks on the measuring tape are compared to a prototype or transfer standard of well-known length; b) *sample measurement:* the measuring tape can then be used to determine the length of a woodblock.

a prototype for the meter, of known length. Figure 2.2 visualizes this concept. Another length-measuring device for verifying the length of the meter may be a caliper with a higher precision than the tape itself, where the caliper is used for comparison with another transfer standard and so on. If we follow the measuring chain for the references all the way up to the top, we would end at the etalon[1] for one or more of the seven *SI base units*[2], defined by the *International System of Units* (SI) [10]. All other physical units can be derived from the SI base units from Table 2.1. The hierarchical concept, summarized in Figure 2.3, is the basis for modeling advanced measurement scenarios for measurement uncertainty evaluation in this work and will help to understand arguments for the modeling language definition and constructs later on. In this figure, the measuring tape as reference for determining the length of a woodblock would be in the lowermost part. The company producing the tape would have calibrated it using own working and reference standards. Their standards are calibrated using secondary standards, secondary with primary standards of a *National Metrology Laboratory* (NML) and finally the primary standards of the NMLs are used for comparison

[1]An etalon is the physical realization of a prototype for a measuring unit, like the standard meter.

[2]Actually, all SI base units are defined using natural constants, except for the mass, which uses a platinum-iridium alloy block as standard reference seated near Paris. There exist six sister copies and official copies of this block for different countries. Currently intensive research [17] is in progress to replace the definition using also natural constants.

Physical quantity	**Symbol**	**Name**	**Measure**
Length	l	meter	m
Mass	m	kilogram	kg
Time	t	second	s
Electric current	I	ampere	A
Thermodynamic temperature	T	kelvin	K
Amount of substance	n	mole	mol
Luminous intensity	I_v	candela	cd

Table 2.1: List of the seven SI base units; all other physical units can be derived from the SI base units.

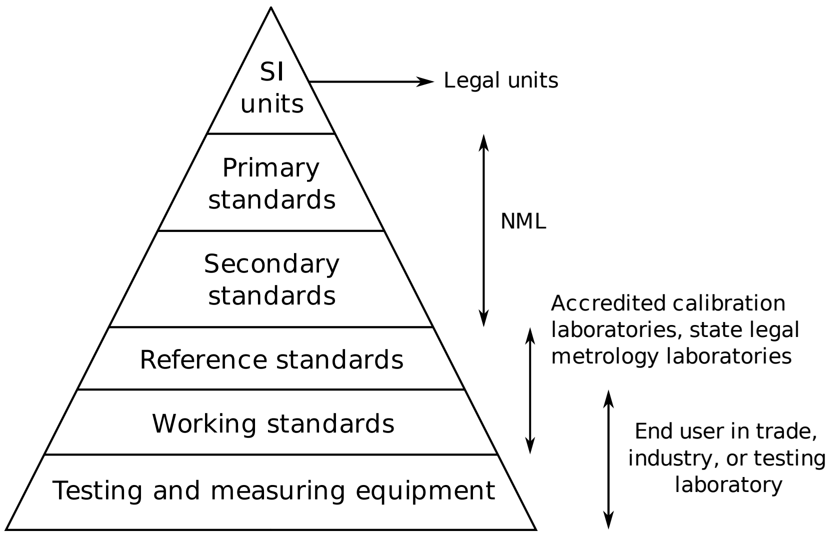

SI units
Legal units
Primary standards
NML
Secondary standards
Accredited calibration laboratories, state legal metrology laboratories
Reference standards
Working standards
End user in trade, industry, or testing laboratory
Testing and measuring equipment

Figure 2.3: Hierarchy of standards and measuring equipment [42]

with the standards of other NMLs in a so-called key comparison procedure.

Figure 2.4 shows a more advanced measurement scenario with n different sample measurements. To keep up with the length measurement, in a real-life scenario this could for example be different saw mills producing wooden bars. Each saw mill has its own working standard of a measuring tape. The bars are used in furniture industry for manufacturing and must have the same length within a given tolerance. In this case a comparison of different measurement results is of interest.

It is very important to see that in general measurements cannot and do not take place at the same location at the same time, thus with strongly varying environmental conditions. This means that we should consider sources, e.g., temperature, pressure, humidity, etc., that vary from measurement to measurement and can af-

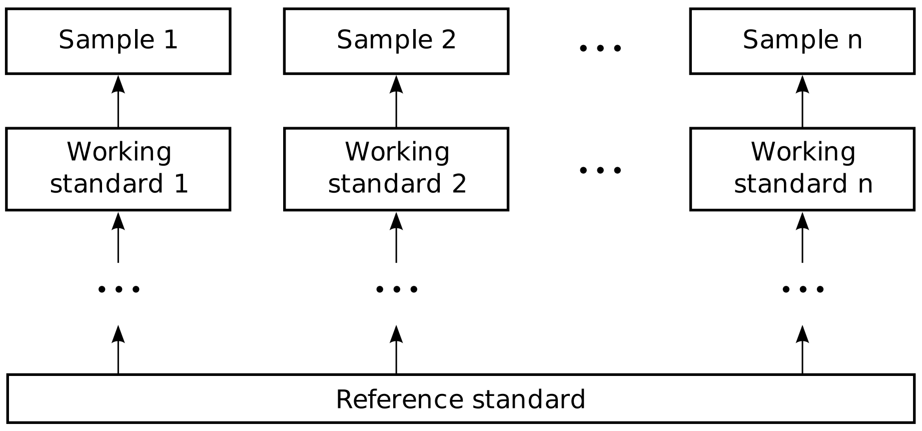


Figure 2.4: *n* measurements in a measurement series using different working standards; at a certain level – in this case the reference standard – the measurements have to rely on the same standard for comparison.

fect the outcome of individual measurements. These effects have to be corrected where possible if they are significant. Nevertheless, there will always remain a variation in the result, because of environmental conditions that are not under full control or because they can only be quantified to a certain precision. One has to be aware of these uncertainty sources and take them into account, state them together with the result of a measurement explicitly and consider them in preceding measurements. This is exactly the idea of the evaluation of *measurement uncertainty*. The definition of measurement uncertainty is given in the Appendix C, but as the concept is so central for this work, I quote it here once more taken from the *International Vocabulary of Metrology* [23, Section 2.26].

> **measurement uncertainty** non-negative parameter characterizing the dispersion of the quantity values being attributed to a measurand, based on the information used.

After we have seen the concepts of measurements and gotten an idea of measurement chains for reference standards using a very practical and simple example, I will introduce in the following section a procedure for measurement uncertainty evaluation following the ISO guidelines.

2.1 Concept of uncertainty evaluation

The fundamental basis of this thesis is an ISO document, the first supplement to the GUM [22]. In combination with the classical GUM document [37] it explains in detail an approach how to quantify measurement uncertainty, and how to calculate it for a given measurement scenario. I want to give a first overview of the concept of measurement uncertainty evaluation according to GS1 [22, Section 5.1.1] and go into more detail in the subsequent sections. There are three main stages, whereas the stages are refined:

1. **Formulation**
   (a) define the output quantity, the quantity intended to be measured (the measurand);
   (b) determine the input quantities upon which the output quantity depends;
   (c) develop a model relating the output quantity to these input quantities;
   (d) on the basis of available knowledge assign probability density functions (PDFs) – Gaussian (normal), rectangular (uniform), etc. – to the input quantities. Assign instead a joint PDF to those input quantities that are not independent;
2. **Propagation**
   propagate the PDFs for the input quantities through the model to obtain the PDF for the output quantity;
3. **Summarizing**
   use the PDF for the output quantity to obtain
   (a) the expectation of that quantity, taken as an estimate of the quantity,
   (b) the standard deviation of that quantity, taken as the standard uncertainty associated with the estimate, and
   (c) a coverage interval containing the output quantity with a specified probability (the coverage probability).

This project mainly concentrates on the first stage, the formulation, to help experts of metrology to model and structure complex measurement scenarios. The second stage deals with the evaluation of the model and is encapsulated in the calculation core. Summarizing in the third step can be supported again by a modeling language as we will see.

2.2 First Stage: Formulation

The formulation phase is where the metrologist in the laboratory has to collect and organize all information related to the measurement and measurement uncertainty.

The first thing to do in 1.(a) is to define the quantity that should be measured and is of interest, the measurand. The next step 1.(b) requires a detailed analysis of the environment and specifies all the influence quantities that contribute to the uncertainty of the measurement. In general, a measurement follows given measurement protocols and/or specifications. This can be used as a basis, but in general protocols only describe the concept of a measurement. Environmental conditions have to be adapted and considered for measurement uncertainty evaluation from measurement to measurement. It is important that the model for measurement uncertainty evaluation is validated and that the measurement is under statistical control.

I will now introduce a nice and simple graphical tool to get an organized and structured overview of the possible sources of uncertainty in a first step, the so-called *cause-and-effect diagrams*. This concept of modeling is not explicitly mentioned in the ISO documents, but used, for example, in the *EURACHEM/CITAC Guide* [19] for the quantification of measurement uncertainty in analytical chemistry.

2.2.1 Visualizing dependencies of influence quantities

Uncertainty sources to measurements can be visualized using so-called cause-and-effect diagrams[3] (CED). CEDs are a common and compact way to show hierarchical dependencies between influence quantities and give information about the structure of measurement scenarios.

The concept is to start with a horizontal arrow, the main bone. Direct influence quantities to the main bone are represented using new arrows pointing to the main bone and labeled with a name or description. This process can then be recursively iterated, adding new arrows or bones and refining the model as far as needed. A practical example from chemistry [19, Section D.4] is shown in Figure 2.5. There a direct determination of the density $d(EtOH)$ of ethanol by weighing a known volume V in a suitable volumetric vessel of tare weight m_{tare} and gross weight including ethanol m_{gross} is shown. In this example the measurand, and therefore the quantity of interest, is used as main bone. The interpretation would be that the three influence quantities affect the outcome of the measurand directly. The shape of a fish is used in this case to visualize the fishbone structure and will be omitted in later examples.

In a second example in Figure 2.6 from physics [32] an alternative view and interpretation of CEDs is given. The measurement is depicted to a measurement

[3]Cause and effect diagrams are also called Ishikawa diagrams or fishbone diagrams, the latter because of their similarity to the shape of a fish skeleton. They have been introduced by Kaoru Ishikawa [36] for management processes and adapted in many different fields to show the causes of a certain event.

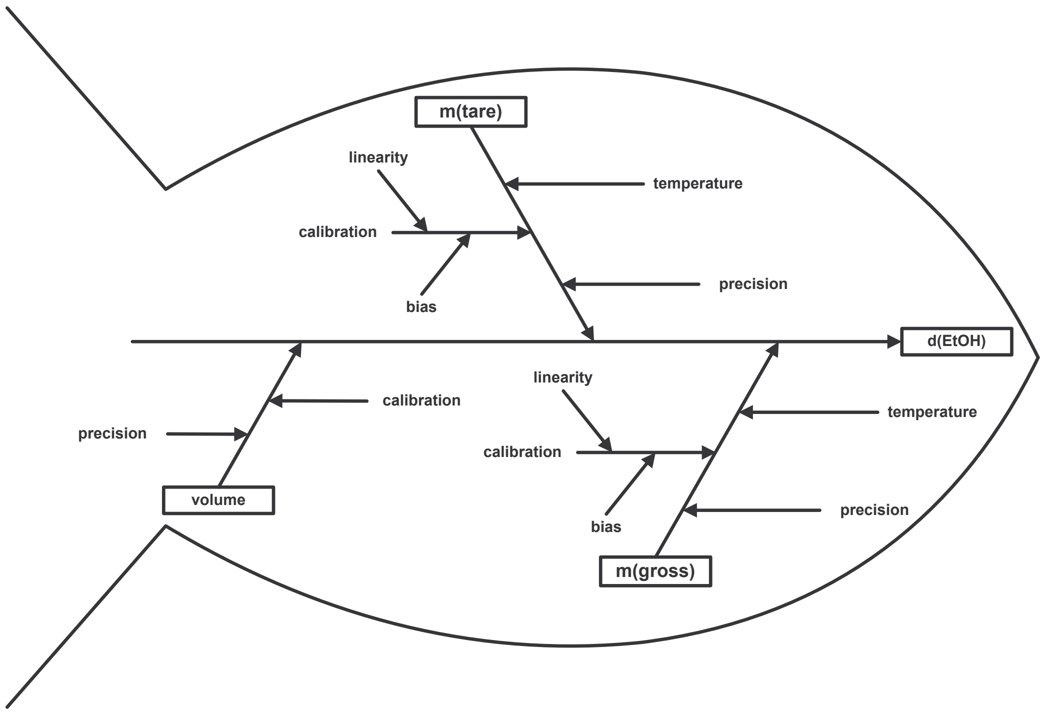


Figure 2.5: Example of a cause and effect diagram from chemistry; it explains the direct determination of the density $d(EtOH)$ of ethanol by weighing a known volume V in a suitable volumetric vessel of tare weight m_{tare} and gross weight including ethanol m_{gross}.

chain visualized by the help of a block diagram. The measurand Δp_{INSTR} itself is embedded in the structure of the diagram. This approach leads to the inverse problem, where the resulting equation has to be converted to extract the measurand. It also shows an iterative refinement of the measurement scenario from a coarse model to a more detailed model.

Although CEDs are practical for modeling measurement scenarios, they suffer from some limitations. The actual importance of influence quantities for the measurement uncertainty is not obvious from the diagrams, and the effect of the structured visualization diminishes with more depth and higher-resolution models. Furthermore, there are some issues if one influence quantity affects more than one other parental influence quantity, which can lead to double counting of certain influence quantities. Some of the limitations can be overcome by using colored diagrams or by storing additional information in the length and thickness of arrows.

In general, CEDs are a good starting point to model measurement scenar-

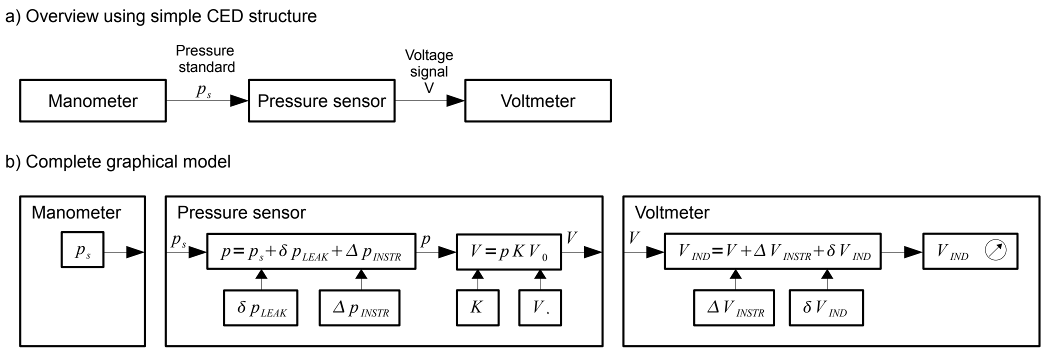


Figure 2.6: Example of a block diagram for a measurement scenario from physics [32]; the measurand Δp_{INSTR} is embedded in the diagram. In a) the basic structure of the measurement is visualized, whereas b) shows the refined dependencies of influences quantities including the mathematical description.

ios from scratch and to get an understanding for the possible uncertainty sources of a measurement. The resolution of CEDs should be as detailed as possible; a stepwise structuring using separate CEDs for different influence quantities seems appropriate. Talking of the importance of uncertainty sources to the measurement, it is not clear in advance if an influence quantity contributes significantly to the measurement uncertainty. Also, the importance may change if parameters or other influence quantities on the measurement vary. This is usually one of the interesting points to analyze in sensitivity analysis, so we encourage metrologists to put as much information into the uncertainty calculation as possible.

I will use CEDs in this work to visualize and clarify the hierarchical structure of measurement scenarios later on where appropriate for modeling and to explain some aspects of the modeling language definition.

2.2.2 Relation of uncertainty sources and measurand

The next step 1.(c) is to develop a mathematical model that describes the functional relationships of all the input quantities to the measurand, the quantity of interest. The result of a measurement is a number of values that can be attributed to the measurand. These values are just an estimations for the real values of the measurand, because uncertainty sources affect the result of every measurement. It is essential to attribute every measurement result with its expected uncertainty. The interaction of the result of a measurement and the uncertainty sources are modeled mathematically by functional relationships that describe the dependencies between the uncertainty sources, i.e. influence quantities, and the measurand. It is the job of metrologists to develop the model equations for the measurand that consider all uncertainty sources and represent the measurement scenario properly.

GUM and GS1 require a single (scalar) measurand Y to be described using the equation of the measurand,

$$Y = f(X_1, \dots, X_N) = f(\mathbf{X}), \tag{2.2.1}$$

where the functional relationship f explains the mathematical dependencies of the distinct input quantities $\mathbf{X} = (X_1, \dots, X_N)$. Each of the input quantities may be affected by an uncertainty and contributes to the measurement uncertainty of the specific measurement. The input quantities $X_1, \dots, X_N$ can depend again on other influence quantities described with an additional functional relationship.

2.2.3 Describing uncertainties for input quantities

In the final step 1.(d) of the formulation phase the input quantities should be analyzed in detail to describe their individual measurement uncertainty. The fundamental principle of GUM and GS1 is the representation of the input quantities $X_1, \dots, X_N$ as random variables. Each random variable has an associated probability distribution that describes the knowledge of the uncertainty of an influence quantity. Additionally, the GUM uncertainty framework, the classical approach of GUM to evaluate the measurement uncertainty, makes a distinction between two types of input quantities:

Type A The probability distribution of values for input quantities of this type is quantified using repeated observations, i.e. repeated measurements. So the probability distribution used for such influence quantities is derived from observed frequency distributions.

Type B Often it is not possible to perform additional measurements to estimate the uncertainty of input quantities explicitly. In this case the probability distribution for the input quantity has to be estimated using prior knowledge or experience, measurement data from earlier observations, information provided by the manufacturer, data provided by calibration, certificates, or handbooks.

The selection of proper probability distributions for input quantities relies on metrologists as additional measurements are necessary, or otherwise a substantial knowledge about the measurement procedure is required and essential to justify the selection of a specific probability distribution to describe an influence quantity. If the equation of the measurand and a quantification of assigning proper probability distributions to the input quantities with well-selected parameters are provided in the formulation phase, the evaluation of the measurement uncertainty can take

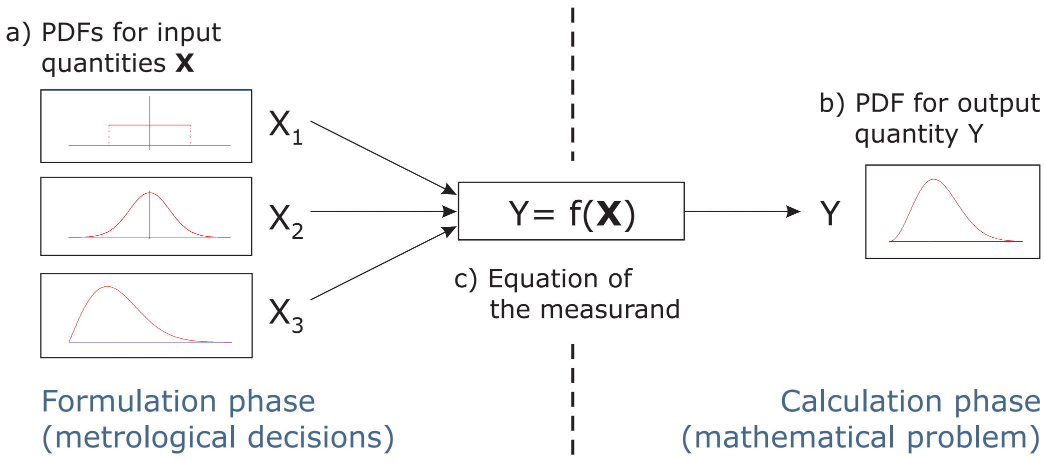

a) PDFs for input quantities **X**
X_1
X_2
X_3
Y= f(**X**)
c) Equation of the measurand
b) PDF for output quantity Y
Y
Formulation phase (metrological decisions)
Calculation phase (mathematical problem)

Figure 2.7: The probability density functions (PDF) in a), which describe the uncertainty of influence quantities, are propagated through the equation of the measurand b) and result in a PDF for the measurand c). This visualization is very common for explaining the concept of the measurement uncertainty evaluation according to GUM and GS1 and appears in several publications [15,55,71,84] in very similar forms.

place in the calculation phase. Figure 2.7 summarizes their concept and visualizes graphically the flow of information from the input quantities described using probability density functions (PDF) to the result for the measurand by propagating the information of the influence quantities through the equation of the measurand.

At this moment we have all the necessary input from the metrologist to start an evaluation for the measurement uncertainty of a specific measurement. In the next section we will examine how the measurement uncertainty of the measurand can be expressed and after that, how the information can be used to derive and calculate the measurement uncertainty.

2.2.4 Standard uncertainty

There are different methods to evaluate the measurement uncertainty using the equation of the measurand and the quantification of the input quantities with properly assigned probability distributions as a basis. Each method comes with its own advantages and limitations [53]. The fundamental issue is that the measurand Y cannot be described by a scalar as we have already seen, but is itself a random variable that has to be described by a distribution function

$$G_Y(\eta) = \int_{-\infty}^{\eta} g_Y(z)dz.$$

The probability density function (PDF) $g_Y(\eta)$ for Y is given as

$$g_Y(\eta) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} g_{\mathbf{X}}(\xi_1,\dots,\xi_N)\delta(\eta - f(\xi_1,\dots,\xi_N))d\xi_N \dots d\xi_1, \qquad (2.2.2)$$

where δ is the Dirac delta function and $\xi_1,\dots,\xi_N$ represent possible values for the input quantities $X_1,\dots,X_N$.

Definition The *standard uncertainty*[4] $u(y)$ is defined as the estimated standard deviation for values that could reasonably be attributed to the measurand Y.

The direct and exact way to evaluate the measurement uncertainty would be to compute the integral 2.2.2 analytically. The problem is that this is only possible in a limited number of cases. So, the evaluation methods presented in this work approximate the measurement uncertainty for the measurand Y in an appropriate way.

Example We use a small example from *Best Practice Guide No. 6* [15, Section 9.2], where the analytical solution of measurement uncertainty evaluation is provided and compared with other approaches. The equation of the measurand

$$Y = f(X_1) = \ln(X_1) \qquad (2.2.3)$$

depends on a single influence quantity $X_1 \sim R(0.1, 1.1)$ using a uniform distribution. The PDF of X_1 is defined as

$$g_{X_1}(\eta) = \begin{cases} 1/(b-a), & a \le \eta \le b \\ 0 & \text{otherwise.} \end{cases}$$

[4]In this document we use the notation of the standard uncertainty as $u(y)$ for a measurand Y from GS1, which differs from the classical GUM, because there it is defined as $u_c(y)$. GUM tries to express that the standard uncertainty results from a combination (c) of influence quantities $X_1,\dots,X_N$. GS1 argues that the subscript c is superfluous, and may even be inappropriate if the result of one measurement uncertainty evaluation is used as input for another evaluation.

We have a single input quantity and our model function $f(X_1) = \ln(X_1)$ is differentiable and strictly monotonic. Hence, we can use

$$g_Y(\eta) = g_X(f^{-1}(\eta))|d(f^{-1}(\eta))/d\eta| \tag{2.2.4}$$

to calculate the PDF $g_Y(\eta)$ for Y according to [64, page 58 ff]. The inverse of our model function is $f^{-1}(\eta) = e^{\eta}$. The first term of Equation 2.2.4 uses the PDF of X_1 with the inverse model function as argument. This leads to

$$g_{X_1}(f^{-1}(\eta)) = g_{X_1}(e^{\eta}) = \begin{cases} 1/(b-a) & a \le e^{\eta} \le b \\ 0 & \text{otherwise.} \end{cases}$$

The derivation for the second term of Equation 2.2.4 is $d(f^{-1}(\eta))/d\eta = e^{\eta}$. Putting all things together, the PDF for Y is given as

$$g_Y(\eta) = \begin{cases} e^{\eta}/(b-a) & \ln(a) \le \eta \le \ln(b) \\ 0 & \text{otherwise.} \end{cases}$$

We can calculate the expectation value y of Y as

$$y = \int_{\ln(a)}^{\ln(b)} \frac{\eta e^{\eta}}{b-a} d\eta = \frac{b(\ln(b)-1) - a(\ln(a)-1)}{b-a}$$

and the standard uncertainty $u(y)$ as

$$u^2(y) = \int_{\ln(a)}^{\ln(b)} \frac{(\eta - y)^2 e^{\eta}}{b-a} d\eta = \frac{b(\ln(b)-y-1)^2 - a(\ln(a)-y-1)^2}{b-a} + 1.$$

If we apply the settings $a = 0.1$ and $b = 1.1$ we obtain an expectation value of $y = -0.665$ and a standard uncertainty of $u(y) = 0.606$. A plot of the result is given in Figure 2.8 on page 27 in comparison with two methods that will be introduced in the next sections as approximations for the standard uncertainty.

Definition The *expanded uncertainty U* is obtained by multiplying the standard uncertainty $u(y)$ by a *coverage factor k*. The rationale is to obtain a coverage or uncertainty interval that should contain a large fraction of the values that can reasonably be attributed to the measurand. Usually the coverage factor k lies between 2 and 3. It has to be reported together with the expanded uncertainty.

I want to point out that the interpretation of the standard deviation as standard uncertainty and a corresponding expanded uncertainty were selected, because in practice it is necessary to express the uncertainty in a compact and convenient form, say in one or two numbers. Having the expectation value and the standard deviation for the measurand, it is obvious to interpret it as parameters of a normal distribution. There exist many examples, where this assumption is not appropriate as we will see in later examples and, therefore, the standard uncertainty should be accompanied with more information about the uncertainty evaluation of the measurand in such cases. This is the case, for example, in highly non-linear scenarios or in all cases, where the result cannot be properly approximated by a normal or Student's *t*-distribution.

2.2.5 Summarizing words to formulation

The formulation phase is essential for a proper measurement uncertainty evaluation. Hence, one major target of the project is to provide support for a structured and organized way of modeling. The developed modeling language allows metrologists from laboratories to develop complex, real-life models in a step-by-step approach, where they start in the beginning with a heap of mathematical equations, experience and expert knowledge, certificates, protocols, specifications, handbooks and observation data, and end up with valid and proper models that can be reused and understood by other metrologists.

The next phase for the measurement uncertainty evaluation is the propagation phase, which defines how calculation should be performed. We have already seen the analytical approach of evaluation of the measurement uncertainty, but unfortunately this is only possible in very simple and rare cases. Hence, we will have a closer look at the advantages and disadvantages of the two recommended methods in the guides and especially at the Monte Carlo method that is implemented in the simulation core for this project.

2.3 Second Stage: Propagation

From this point on the metrologists have provided all the required information and they now have to decide on an evaluation technique. For an overview and comparison of different evaluation methods I want to refer to the PhD thesis of Martin Müller [53]. I will present and summarize the two alternative methods of GUM and GS1 here. The first approach from the classical guide is the GUM uncertainty framework, that relies on an approximated linear model and input quantities that can be properly described using normal distributions. The second approach is a Monte Carlo method introduced in GS1, that does not have the restrictions to a linear model and normal distributed influence quantities, but needs a lot of computational power for the evaluation.

2.3.1 GUM Uncertainty Framework

In 1993 the *Joint Committee for Guides of Metrology* (Comité commun pour les guides en métrologie, JCGM) of the *Bureau International des Poids et Mesures* (BIPM) agreed in the GUM [37] on the *GUM Uncertainty Framework.* It presents a method to approximate the measurement uncertainty for a measurement function $Y = f(X_1,\ldots,X_N)$ and calculate the standard uncertainty using a Taylor series expansion. We can calculate the best estimate of the output quantity Y defined as $y = f(x_1,\ldots,x_N)$ with N best estimates $x_1,x_2,\ldots,x_N$ for the input quantities $X_1,\ldots,X_N$ that are described with appropriate probability distributions. We will have a look now on small deviations of Y about the estimate y in terms of small deviations of X_i about the expectation values x_i, say

$$Y - y = \sum_{i=1}^{N} \frac{\partial f}{\partial X_i}(X_i - x_i).$$

We assume that higher-order terms are negligible for the current case. The square of the deviation $Y - y$ is given as

$$\begin{aligned}(Y-y)^2 &= \left[\sum_{i=1}^{N} \frac{\partial f}{\partial X_i}(X_i - x_i)\right]^2 \\ &= \sum_{i=1}^{N}\left[\frac{\partial f}{\partial X_i}\right]^2 (X_i - x_i)^2 + 2\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \frac{\partial f}{\partial X_i}\frac{\partial f}{\partial X_j}(X_i - x_i)(X_j - x_j).\end{aligned}$$

The expectation of the squared deviation $(Y-y)^2$ is the variance $\sigma_Y^2 = E[(Y-y)^2]$ of Y. The same holds for the quantities X_i, so that $\sigma_i^2 = E[(X_i - x_i)^2]$.

To apply the procedure for measurement uncertainty evaluation, we have to estimate the variances σ_i^2 for influence quantities X_i and plug in the corresponding estimates $u^2(x_i)$. Additionally, we develop the partial derivates $c_i = \partial f/\partial X_i$ using the best estimate x_i of influence quantities X_i and call them the *sensitivity coefficients*[5]. If we put everything together, we obtain formula

$$u^2(y) = \sum_{i=1}^{N} c_i^2 u^2(x_i) + 2\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} c_i c_j u(x_i,x_j) \tag{2.3.1}$$

for the squared standard uncertainty $u^2(y)$, which estimates the variance σ_Y^2 of output quantity Y. The term $u(x_i,x_j)$ is the estimated covariance associated with two influence quantitites X_i and X_j. If the influence quantities are not correlated

[5]The GUM uses the notation $\partial f/\partial x_i = \partial f/\partial X_i$ evaluated at the expectations of X_i.

the term $u(x_i, x_j)$ is zero; in such cases, the formula reduces to

$$u^2(y) = \sum_{i=1}^{N} c_i^2 u^2(x_i) = \sum_{i=1}^{N} u_i^2(y), \tag{2.3.2}$$

where we set $u_i^2(y) = c_i^2 u^2(x_i)$, respectively $u_i(y) = |c_i| u(x_i)$. The method based on first-order Taylor series approximation is called the *law of propagation of uncertainty* in GUM [37, Section 5.1.2]. It relies on an information reduction of the probability distributions of influence quantities to its parameters expectation value and standard deviation. Hence, it is not appropriate for arbitrary probability distributions and, therefore, reduces the number of applicable cases of the approach to evaluate the measurement uncertainty considerably. The second shortcoming is that the model has to behave linear in the range of interest if the uncertainty propagation should be applied. So a lot of initial information about the influence quantities is ignored or lost if applying the law of propagation of uncertainty. For measurement scenarios, where significant non-linearities appear, the guide recommends adding higher-order terms of the Taylor series expansion, like the next-highest-order terms for Equation 2.3.1

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \left[\frac{1}{2} \left(\frac{\partial^2 f}{\partial x_i \partial x_j} \right)^2 + \frac{\partial f}{\partial x_i} \frac{\partial^3 f}{\partial x_i \partial x_j^2} \right] u^2(x_i) u^2(x_j).$$

This leads to further implications and more complex evaluations of the equation of the measurand. Furthermore, the quantification for higher order terms is very complicated and sometimes even not possible in practice.

Besides, if using the GUF approach, it is necessary to estimate the so-called degrees of freedom to get an approximation for an uncertainty interval of confidence p for the measurand Y. The interval $Y = y \pm U_p$ uses the expanded uncertainty $U_p = k_p u(y)$, where the expansion factor $k_p = t_p(\nu_{eff})$ is taken from existing tables for t-distributions with ν_{eff} degrees of freedom. The method of GUM to quantify the degrees of freedom is a rough estimation applying the Welch-Satterthwaite formula [69]

$$\nu_{eff} = \frac{u^4(y)}{\sum_{i=1}^{N} \frac{u_i^4(y)}{\nu_i}},$$

using the estimated variances $u^2(y)$ of Y and $u_i^2(y) = |c_i| u(x_i)$ of influence quantities X_i. The formula was originally thought for influence quantities of Type A, where we have explicitly n_i independent observations for analysis of an quantity X_i and, hence, the degrees of freedom $\nu_i = n_i - 1$ for a quantity X_i are well defined. However, it is not always obvious how to determine or estimate the degrees of freedom for Type B influence quantities, describing, for example, experience or expert knowledge. The guide [37, Section G.4.2] explicitly states the following:

> [...] for a Type B evaluation of standard uncertainty it is a subjective quantity whose value is obtained by scientific judgment based on the pool of available information.

Since the release of the guide the approach to handle influence quantities of Type B with the Welch-Satterthwaite formula is under discussion [40]. The need to estimate degrees of freedom underlines once again that the classical GUM uncertainty framework expects the result to be approximately normal distributed as this estimation is based only on the standard deviation of input quantities and their corresponding degrees of freedom.

Another problem with the classical guide is that there is no explanation of how to quantify the validity of the measurement uncertainty evaluation using the GUM uncertainty framework. This drawback is mentioned in the first supplement [22, Section 8.1], where it is recommended applying a second evaluation method for the evaluation of the measurement uncertainty to prove the results right. I will continue presenting this second evaluation method after a small example.

Example We use the same example [15, Section 9.2] as for the analytical solution using Equation 2.2.3 to evaluate the measurement uncertainty according to the GUF. The measurand Y is given as $Y = \ln(X_1)$. The influence quantity $X_1 \sim R(0.1, 1.1)$ is described using a rectangular distribution. Hence, we have to convert it to a normal distribution to apply the GUF approach. The expectation of X_1 is $x_1 = (a+b)/2$, the standard deviation is $u(x_1) = (b-a)/\sqrt{12}$ according to [37, Section 4.3.7]. The expectation y of Y can be calculated using the expectation x_1 of X_1 as

$$y = \ln(x_1) = \ln((a+b)/2) = -0.511.$$

Finally, the sensitivity coefficient is $c_1 = \partial \ln(X_1)/\partial X_1$ evaluated again at $x_1 = (b-a)/2$, so that $c_1 = 1/x_1 = 2/(a+b)$. Applying the law of propagation of uncertainty from Equation 2.3.2 we get

$$\begin{aligned} u^2(y) &= \sum_{i=1}^{1} u_i^2(y) = \sum_{i=1}^{1} c_i^2 u^2(x_i) \\ &= c_1^2 u^2(x_1) = \left[\frac{2}{a+b}\right]^2 \left[\frac{b-a}{\sqrt{12}}\right]^2 \\ &= 0.2315. \end{aligned}$$

Hence, if we take the root we get the standard uncertainty $u(y) = \sqrt{0.2315} = 0.4811$. The result is plotted in Figure 2.8 on page 27 in comparison with the other methods.

2.3.2 Monte Carlo method

The first supplement to the GUM has introduced a new approach for the evaluation of the measurement uncertainty. The final version of the document has been published in summer of 2008 after a lot of discussion, changes and two drafts in 2004 and 2006. Taking into account the increasing computational power of personal computers and multi-core processor machines as well as distributed computing frameworks the possibility of simulating measurement scenarios, a Monte Carlo technique seems to be a good method to tackle the problem of the measurement uncertainty evaluation. The Monte Carlo method is a straightforward approach to calculate a discrete approximation $\mathbf{G}$ for the density function $G_Y(\eta)$ for the Equation 2.2.1 of the measurand Y with input quantities $X_1,\ldots,X_N \in \mathbf{X}$ that contribute to the uncertainty of the measurand. Each input quantity is randomly varied according to the distribution function in M trials evaluating the equation of the measurand, so that with a large number of evaluations we get a large amount of possible values that can be attributed to the measurand Y according to the underlying model.

For the evaluation of the measurement uncertainty using Monte Carlo simulation according to GS1 the following steps [22, Section 5.9.6] are necessary.

1. Select number M of Monte Carlo trials to be made.
2. Generate M vectors $\mathbf{x}_1,\ldots,\mathbf{x}_M$, by sampling from the assigned PDFs, as realization of the N input quantities $X_1,\ldots,X_N$.
3. For each such vector $\mathbf{x}_i$, from the corresponding model value of Y, yielding M model values $y_i = f(\mathbf{x}_i)$, $i = 1,\ldots,M$.
4. Calculate
$$\tilde{y} = \frac{1}{M}\sum_{r=1}^{M} y_r$$
as an estimate for y of Y and estimate the standard uncertainty $u(y)$ associated with y using
$$u^2(\tilde{y}) = \frac{1}{M-1}\sum_{r=1}^{M}(y_r - \tilde{y})^2.$$
5. Sort the model values $y_1,\ldots,y_m$ to form an appropriate coverage interval for Y, for a stipulated coverage probability p.

The procedure presented here differs a little from the original one in GS1, as we try to avoid sorting all simulation data. This will be explained in the next section, when we concentrate on computational issues of Monte Carlo simulations for measurement uncertainty evaluation.

The number of trials M for a simulation, that has to be provided in the first step of the procedure, depends on the selected probability distributions for the input quantities and the complexity of the equation of the measurand. Hence, in daily life it may be hard before starting the simulation to decide on a number of trials that is computable in acceptable time, but still guarantees an appropriate approximation. GS1 includes a description of an adaptive Monte Carlo approach [22, Section 7.9], where the evaluation continues until the results for the estimated expectation value, standard deviation and a $100p$ % uncertainty interval get stable in a statistical sense. The algorithm analyzes and sorts blocks of data[6], calculates the statistical parameters, averages them and observes if twice the standard deviation of the estimates is within a predefined numerical tolerance.

GS1 mentions the following points as conditions to apply the Monte Carlo method properly [22, Section 5.10].

1. The equation f of the measurand is continuous with respect to the elements $X_1, \ldots, X_N$ of $\mathbf{X}$ in the neighborhood of the best estimates x_i of the X_i, $i = 1, \ldots, N$;
2. the distribution function $G_Y(\eta)$ for Y is continuous and strictly increasing;
3. the PDF for Y is
   (a) continuous over the interval for which this PDF is strictly positive,
   (b) unimodal (single-peaked), and
   (c) strictly increasing (or zero) to the left of the mode and strictly decreasing (or zero) to the right of the mode;
4. The estimate $E(Y)$ and the variance $V(Y)$ exist;
5. a sufficiently large value of M is used.

The requirement for the PDF of the output quantity Y in point 3 is necessary for uniqueness of the shortest coverage interval only. It is not necessary if just the shape of the distribution of the measurand Y is of interest.

[6] The guide recommends 10^4 or more trials for one block.

Computational issues

The principle of the Monte Carlo method is easy to understand and (at least at a first glance) straightforward to implement. Nevertheless, there are some very important issues to consider. One has to take care of efficient code and memory usage, as Monte Carlo simulation needs a lot of computational power for the evaluation of the measurement uncertainty. A suitable memory allocation behavior is required as well as a smart way of handling the huge amount of data generated during simulation. There are algorithms to calculate the arithmetic mean and standard deviation as well as some other statistical parameters iteratively during simulation. However, if simulation data should be saved for analysis after running the simulation, e.g., to calculate coverage intervals or to reproduce the results later on, one has to take care about how data is stored as writing huge amounts of data to a storage volume is a very slow task during simulation. In developing the calculation core used for this project [55, 57] we concentrated on constant memory usage during simulation, linear time-behavior and disk-space usage.

Block evaluation Because of computational reasons it is not satisfying to generate all M vectors $\mathbf{x}_i$ of random numbers for the input quantities $X_1, \ldots, X_N$ in the beginning of a simulation. It would be too memory extensive with a large number of trials for simulation together with a large number of influence quantities. On the other hand, parsing and evaluating all equations for evaluation in M single iterations would be very slow and inefficient. The solution is to divide the calculation into blocks of a defined size[7] 10^n. Then the evaluation of a block of 10^n values for each input quantity takes place simultaneously. Here a good trade-off is necessary in order not to exceed the available physical memory as swapping from memory to hard disk and back is a very costly operation.

Another advantage of the block evaluation is that for the calculation of statistical parameters blocks of smaller size can be analyzed, where the overall parameters are approximated afterwards. Mean and standard deviation can be calculated iteratively for all simulation data. Hence, there is no need for an approximated block evaluation for these two parameters. But to extract a coverage interval with coverage probability p it is necessary in principle to sort all M simulation values for the measurand Y and look for suitable values for the limits of the interval. Details of the applied algorithm follow in the next chapter. As sorting is a slow task[8], an alternative as approximation to the coverage interval can be calculated. The approach only requires sorting blocks of size 10^n values and extracts the coverage interval with coverage probability p. Then we can iteratively calculate mean and

[7]Values of $n = 4$ or $n = 5$ have been found suitable in practice.

[8]Good behavior of a typical sorting algorithm is in $O(n \log n)$ to sort n elements; well-known sorting algorithms are quicksort [46] or introsort [58].

standard deviation for the limit values of the coverage interval of these blocks. The convergence of this approach has been investigated in Martin Müller's and Christian Rink's article [54].

2.3.3 Advantages of the Monte Carlo approach

The Monte Carlo approach circumvents the limitations of the classical GUM uncertainty framework in many respects:

- There is no restriction to linear behavior for the model. The equation of the measurand can be used as is, whereas the application of the classical GUM uncertainty framework needs a linearization of the model equations.
- All information provided as probability distributions for influence quantities of the measurement is used for the evaluation of the measurement uncertainty.
- The standard uncertainty can be approximated from the simulation data directly as the square root of the variance, the standard deviation, of the resulting simulation data.
- Coverage intervals, e.g., the 95 % interval, can be estimated from the result. Therefore, there is no need for determining degrees of freedom to calculate a coverage factor for an expanded standard uncertainty.
- A big advantage is also that dependencies and correlations of uncertainty sources to the measurement can be modeled by redefining the model by using additional influence quantities in different parts of the equations. This is already mentioned in the classical guide, but using arbitrary probability distributions opens up new possibilities.
- Finally, the Monte Carlo method allows a validation if applying the classical GUF approach and comparing the results of the two methods.

Example We have already calculated the measurement uncertainty for the example $Y = \ln(X_1)$ of *Best Practice Guide No. 6* [15, Section 9.2] analytically and with the law of propagation of uncertainties from GUF; the example uses only a single influence quantity $X_1 \sim R(0.1, 1.1)$. Performing a Monte Carlo simulation with 10^7 trials results in the same values as we have got analytically, a mean value of $y = -0.665$ and a standard uncertainty of $u(y) = 0.606$. Figure 2.8 shows very clearly the differences and advantages of the Monte Carlo method compared to the GUF method, which delivers an unsatisfying result in this case. The Monte Carlo method approximates the analytical solution almost perfectly.

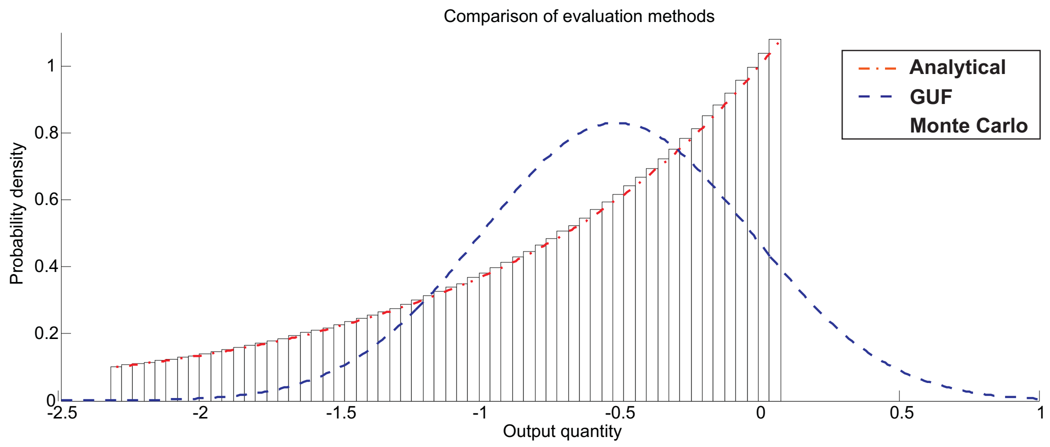


Figure 2.8: Comparison of different methods evaluating measurement uncertainty applied to the very simple example $Y = \ln(X_1)$, where we assume X_1 to be rectangular distributed in the range from 0.1 to 1.1. The result of the Monte Carlo simulation with 10^7 trials approximates very well the analytical solution, whereas the interpretation of the result of the GUF approach assuming a normal distribution is very misleading.

As the Monte Carlo method is currently the most promising and accepted approach [9, 16, 71] in the community of metrologists, the simulation core of the software project presented in this thesis provides an efficient implementation of a Monte Carlo simulation system to evaluate measurement uncertainty for high-resolution and real-life models. The simulation framework is called *MUSE* [55], which is an abbreviation for *Measurement Uncertainty Simulation and Evaluation.*

2.4 Third Stage: Summarizing

In the third and last step to evaluate the measurement uncertainty according to GS1 we have to summarize the collected information about the measurand to a few representative numbers. The expectation value of the quantity is the estimated value for the measurand. As already mentioned before, the standard uncertainty of the measurand is defined as the standard deviation of the resulting data. Additionally, symmetric or shortest coverage or uncertainty intervals are computed usually with coverage probabilities $p \in \{0.95, 0.99, 0.999\}$.

Also in this stage we can analyze the outcome of the simulation with respect to the importance and influence of individual uncertainty sources to the measurement uncertainty of the measurand. The GUM just mentions that so called uncertainty budgets can be reported, stating the expectation value and the standard uncertainty of each uncertainty source. We will go a step further in a later chapter to analyze non-linear models using different approaches to estimate the importance of single influence quantities.

We also want to lay an eye on measurement scenarios where not only one measurement takes place, but sequences of measurements. Such repeated measurements consist of alternating reference and sample measurements. Reference measurements are necessary for controlling environmental conditions and to allow a comparison of sample measurements, e.g., performed in different laboratories. In such cases, where we have more than one output quantity, it is also of interest to analyze the interdependencies of the different results. This goes beyond the scope of the guide and its supplement, but will play an important role in the definition of the modeling language.

Relying on the Monte Carlo method the result of the simulation are one or more huge data files containing possible values for the measurand. A lot of effort was invested to analyze these files properly and extract the most common statistical parameters, like mean value, standard deviation, coverage intervals as well as information for the visualization, like box plots and histograms for a single measurement in an efficient way. We also provide support for fundamental sensitivity and uncertainty analysis of measurement scenarios. As the requirements for analysis depend very strongly on current situations, data files can be stored for additional analysis in third party software, like R [63] or MATLAB®[9]. We decided to implement an analyzing module evaluating the most important statistical parameters and additional summarizing information for visualization after we found limitations of such software packages because of the extensive memory consumption, when analyzing large amounts of simulation data.

2.4.1 Mean and standard deviation

The (arithmetic) mean is defined as the average of the sum of all values

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i.$$

The standard deviation is then defined as

$$s = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2}.$$

[9]MATLAB®, 2008a, The Mathworks, Cambridge MA.

Algorithm 1 Calculation of mean and standard deviation in a single run; input: array *data* with data values

```
$n \leftarrow 0$
$mean \leftarrow 0$
$S \leftarrow 0$
for all $x$ in $data$ do
    $n \leftarrow n+1$
    $delta \leftarrow x - mean$
    $mean \leftarrow mean + delta/n$
    $S \leftarrow S + delta * (x - mean)$
end for
$variance \leftarrow S/(n-1)$
return $mean$ and $variance$
```

A straightforward and direct implementation of the equations as-is would lead to an algorithm with two loops, calculating first the mean value and afterwards the standard deviation.

As *MUSE* usually works on blocks of data, we decided on an efficient algorithm [45, page 232] to calculate the mean value and the standard deviation iteratively within one loop. It is given in pseudo code in Algorithm 1. The idea is to update mean and standard deviation in each step according to

$$\bar{x}_{new} = \frac{n\bar{x}_{old} + x_{new}}{n+1} = \bar{x}_{old} + \frac{x_{new} - \bar{x}_{old}}{n+1}$$

and

$$s^2_{n-1,new} = \frac{(n-1)s^2_{n-1,old} + (x_{new} - \bar{x}_{new})(x_{new} - \bar{x}_{old})}{n},$$

where n is the number of elements before the current iteration, x_{new} the new data value, $\bar{x}_{old}$ and $\bar{x}_{new}$ the old and new mean value.

2.4.2 Uncertainty intervals

Using the Monte Carlo approach, we can distinguish between symmetric and shortest coverage intervals with coverage probability p. For symmetric intervals the probability for a quantity to be below the lower limit must be equal to the probability to be bigger than the upper limit. The shortest interval requires that the upper limit minus the lower limit is minimal for all possible coverage intervals with the same coverage probability p. The shortest and the symmetric coverage intervals are equal for unimodal, symmetric probability distributions. Figure 2.9

Distribution	Cov. int.	95 %	99 %	99.9 %
Exponential	symmetric	[0.025,3.690]	[0.005,5.309]	[0.001,7.707]
	shortest	[0.000,2.992]	[0.000,4.597]	[0.000,6.857]
Beta	symmetric	[0.057,0.543]	[0.023,0.649]	[0.005,0.771]
	shortest	[0.001,0.733]	[0.007,0.609]	[0.037,0.509]

Table 2.2: Values for the coverage intervals for the example of Figure 2.9

shows the comparison of resulting coverage intervals for two non-symmetric probability distributions, a beta and an exponential distribution. Table 2.2 presents the corresponding values.

To estimate a $100p$ % coverage interval $[y_{low}, y_{high}]$ the supplement requires sorting all data. Let then $q = pM$ if pM is an integer; otherwise q is the integer part of $pM + 1/2$ and $r = 1, \ldots, M - q$. A coverage interval is then defined as $y_{low} = y_{(r)}$ and $y_{high} = y_{(r+q)}$. Two of these coverage intervals are of interest.

- The probabilistic symmetric $100p$ % coverage interval can be estimated by taking $r = (M - q)/2$ if the term is an integer, otherwise r should be the integer part of $(M - q + 1)/2$.
- The shortest $100p$ % coverage interval can be estimated by shifting a frame of size q iterating over all r looking for an r^* as a minimum, so that

$$y_{r^*+q} - y_{r^*} \leq y_{r+q} - y_r.$$

This procedure leads to an approximation of the statistical parameters of interest. To prevent sorting all data after simulation, which is in general a very slow task on computers for large data sets, the analyzing module uses the following approach: it calculates statistical parameters again for blocks[10] of simulation data and uses the arithmetic mean as an approximation. In calculating a corresponding standard deviation we gain additional information about the parameters. The convergence of the estimators for this approach is shown in [54].

2.4.3 Histograms

A histogram summarizes information about large data sets. The range from the minimal to the maximal value is equidistantly segmented in k bins. Then each data point falls in one of the bins. The histogram plot shows bars with the height of number of values in each bin. We have already seen histograms, e.g., in Figure 2.8

[10]As this approximation should represent the overall result of the simulation a minimum of 10^5 values are used for evaluation.

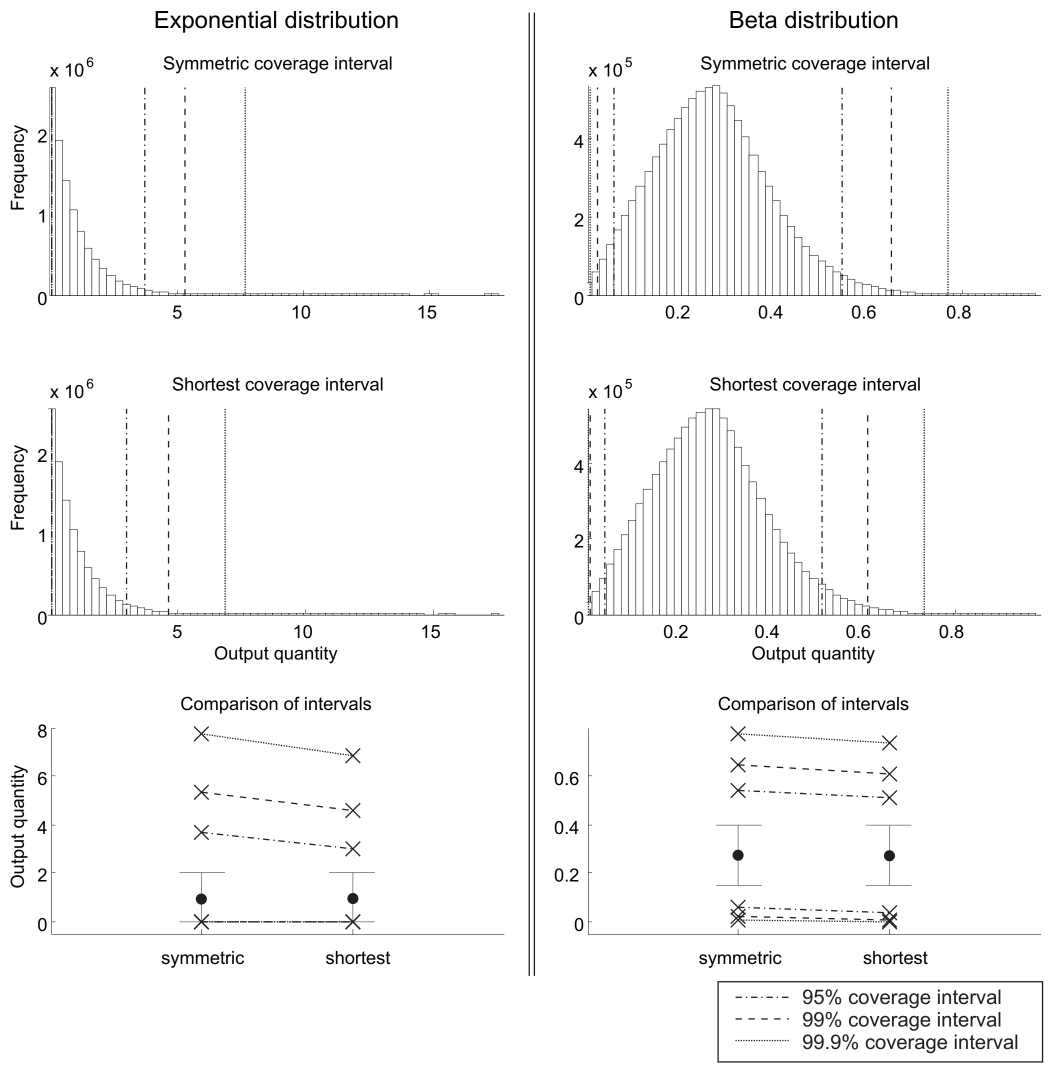


Figure 2.9: The left column shows plots of an exponential distribution with parameter $\lambda = 1$; the right column plots of a beta distribution with parameters $a = 2$ and $b = 5$. In the first and second row there are histograms with symmetric, respectively shortest, uncertainty intervals plotted as dashed lines. The last row uses uncertainty plots for direct comparison. It can clearly be seen that there is a significant difference between the symmetric and shortest coverage intervals. The Monte Carlo simulations were performed with 10^7 trials each.

and 2.9, to visualize the results of Monte Carlo simulations. On the left side of Figure 2.10 the results of three simulations are shown as histograms.

To extract histogram data from data files the number k of bins has to be provided. Then, in an initial run the minimum value x_{min} and maximum value x_{max} of a simulation are examined. The width w_{bin} of one bin is calculated as $w_{bin} = (x_{max} - x_{min})/k$ and the range subdivided in the number k of bins. As blocks of values are sorted for the determination of a p coverage interval, the assignment to a bin can use this additional information about partial order of data and can take place simply in the second run. A two-pass algorithm[11] is given in Algorithm 2.

2.4.4 Box plots

A box plot summarizes information about the simulation data, too. An advantage is that drawing is much more compact and comparing more than one result is easier than with histograms. Of course it relies on information reduction; the summarized data contains less information, e.g., details about the shape of the distribution are lost. Figure 2.10 shows a comparison of three simulations with histograms and box plots. The example is taken from GS1 [22, Section 9.2], the complete model definition is given in Appendix B.1. Instead of the arithmetic mean, the median is used to describe the quantity in a box plot. The box itself defines the lower quartile $\chi_{.25}$ and upper quartile $\chi_{.75}$, each containing exactly 25 % of the values next to the median on the corresponding side. The whiskers indicate the extreme values, the largest and smallest value of the data. If the smallest, respectively largest, value is inside of a range of $1.5IQR$, where the *interquartile range* (*IQR*) is defined as $IQR = \chi_{.75} - \chi_{.25}$, the smallest, respectively largest, value is taken as lower, respectively upper, whisker. If the value is outside of this range, the values below, respectively above, are interpreted as outliers and the range value itself is taken for the whisker. Outliers are usually visualized marked as dots or with an ‘x’ in the plot.

In Monte Carlo simulation very large data files result from simulation[12]. As the number of simulation data is very large, there is a good chance of having many outliers, which would distract from the actual information of the box plots. Therefore, in the current implementation of the analyzing module the outliers are not indicated explicitly as dots, but only counted and annotated below/above the lower/upper limit.

[11] *MUSE* always performs two passes over simulation data to extract the most important statistical parameters; in the end, all the analyzing algorithms presented in this chapter are interwoven in these two passes.

[12] For simulation of examples in this thesis usually 10^7 trials were performed, but at least 10^6.

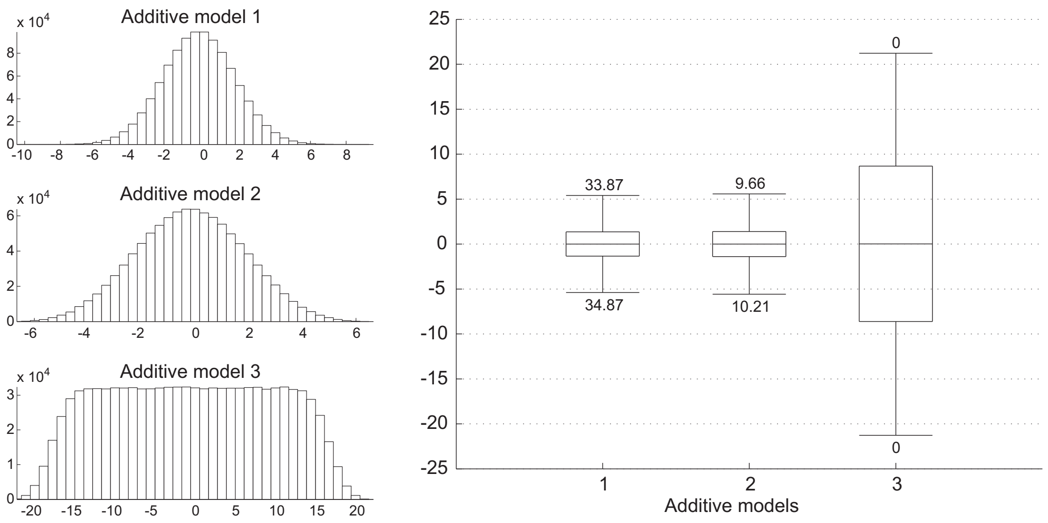


Figure 2.10: Result of three simulations with 10^6 trials each and the equation $Y = X_1 + X_2 + X_3 + X_4$ of the measurand; on the left side are the histograms in comparison, on the right side the corresponding box plot for the same example. In model 1 the quantities $X_1, \ldots, X_4$ are normal distributed $N(0,1)$, in models 2 and 3 they are rectangular distributed with different parameter settings. The annotated numbers at the upper and lower limits of the whiskers are the average number of outliers of analyzed blocks of size 10^5.

Algorithm 2 Two-pass algorithm for determining histogram data. Function $intarray(m, v)$ generates an array of size m initialized with values v. Input: array $data$ with simulation data; block size n; number k of bins

```
for all values $x$ in $data$ do
    if $x < x_{min}$ then
        $x_{min} \leftarrow x$
    end if
    if $x > x_{max}$ then
        $x_{max} \leftarrow x$
    end if
end for
$bins \leftarrow intarray(k, 0)$
$size \leftarrow (x_{max} - x_{min})/k$
for all blocks $b$ of size $n$ in $data$ do
    $cur \leftarrow 0$
    $limit \leftarrow x_{min} + size$
    $b \leftarrow sort(b)$
    for i from 0 to n-1 do
        if $b[i] \geq limit$ then
            $limit \leftarrow limit + size$
            $cur \leftarrow cur + 1$
        end if
        $bins[cur] \leftarrow bins[cur] + 1$
    end for
end for
return $bins$
```

As we avoid sorting the whole data we apply a block-wise evaluation, too, extracting the parameters for each block and averaging afterwards over the set of parameters. Therefore, also the number of outliers does not represent the actual number of outliers of the whole dataset, but only the average number of outliers for the block size used for analyzing.

2.4.5 Standard uncertainty plot

Besides box plots, we will also see the standard uncertainty plot or mean-and-standard-deviation plot in this work. Instead of median and quantiles the statistical parameters mean value (dot) and the difference of the mean value to the standard deviation on both sides (horizontal lines) is plotted. This visualization is more appropriate for classical measurement uncertainty analysis as the same statistical

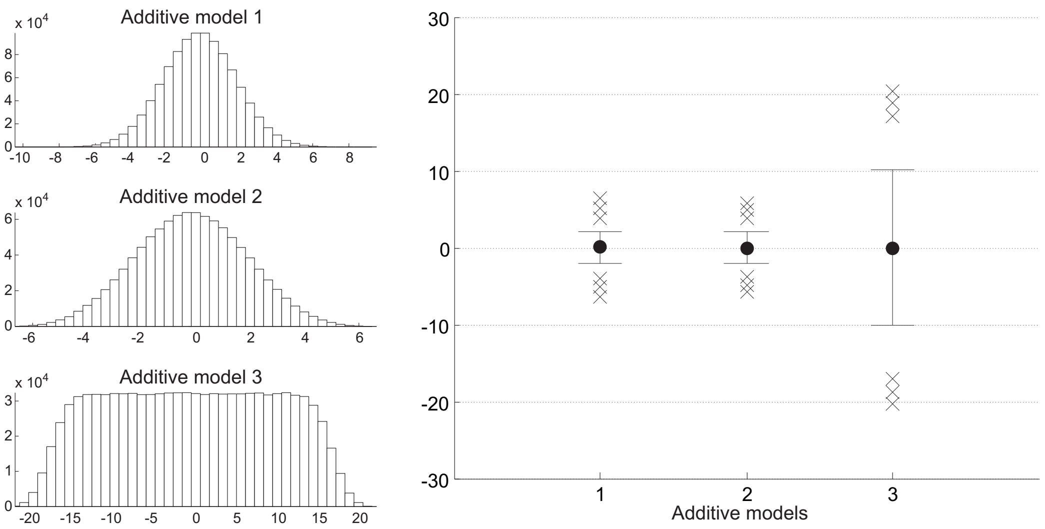


Figure 2.11: Standard uncertainty plot visualizing statistical parameters mean and standard deviation of the same example and simulation data as in Figure 2.10; additionally, the symmetrical uncertainty intervals are plotted.

parameters are used, but is limited as it only shows the symmetrical parameter standard deviation in its basic form. To get a better idea about the resulting distribution, the coverage intervals can be plotted additionally. Figure 2.11 shows such a plot with the same simulation data as the box plots in the last Figure 2.10. For analysis also the 95 %, 99 % and 99.9 % uncertainty intervals are plotted as 'x' marks.

2.5 Summarizing uncertainty evaluation

Figure 2.12 summarizes the necessary steps for the measurement uncertainty evaluation using the Monte Carlo method graphically for a measurement with a single measurand Y.

The formulation stage is the part where uncertainty evaluation software can support experts to model measurement scenarios in a structured way. Metrologists in laboratories have to determine the output quantities, quantify input quantities properly and they also have to provide the mathematical equations that define the functional relationship between the influence quantities of the measurement – the input quantities – and the measurand – the output quantity. It is vital to work in an organized and structured way to get an appropriate and valid model for real-life applications.

After the formulating phase, the propagation phase defines how calculation

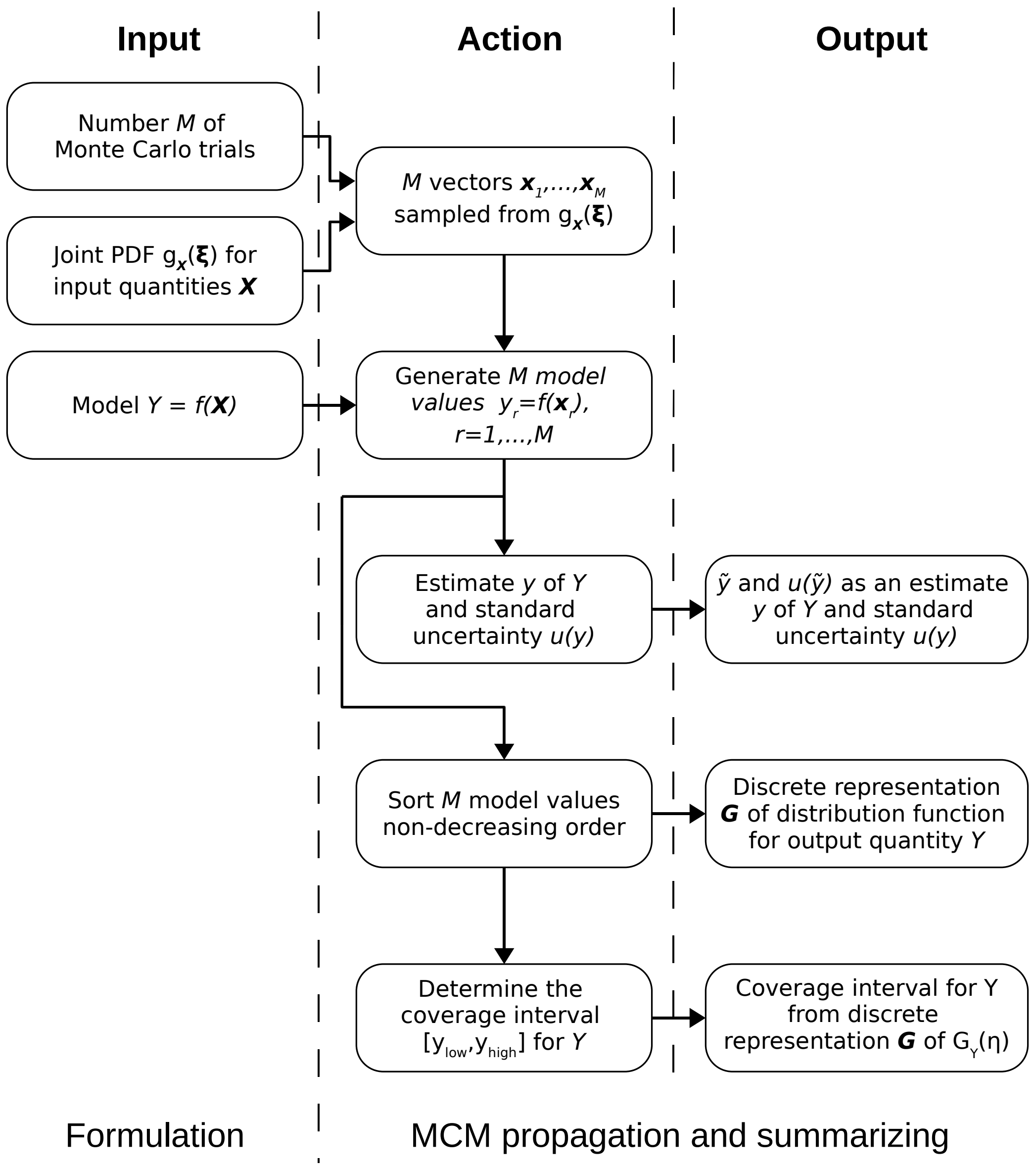

Input
Action
Output
Number M of Monte Carlo trials
Joint PDF g_X(ξ) for input quantities X
Model Y = f(X)
M vectors x_1,...,x_M sampled from g_X(ξ)
Generate M model values y_r=f(x_r), r=1,...,M
Estimate y of Y and standard uncertainty u(y)
Sort M model values non-decreasing order
Determine the coverage interval [y_low,y_high] for Y
ỹ and u(ỹ) as an estimate y of Y and standard uncertainty u(y)
Discrete representation G of distribution function for output quantity Y
Coverage interval for Y from discrete representation G of G_Y(η)
Formulation
MCM propagation and summarizing

Figure 2.12: The necessary steps to evaluate the measurement uncertainty according to the first supplement to the GUM

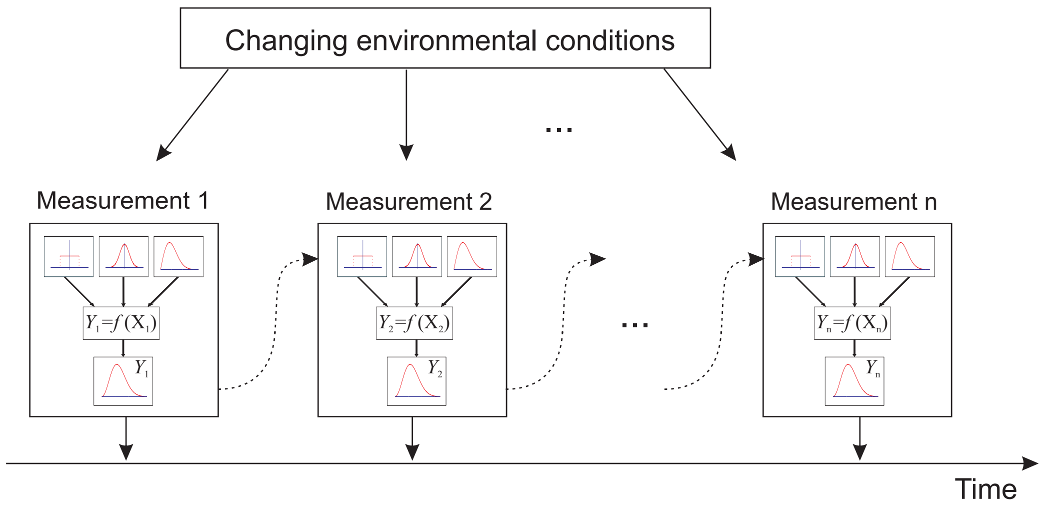


Figure 2.13: Real-life measurement scenario, where the result and measurement uncertainty of one measurement may depend on one or more preceding measurements; environmental conditions change over time and may have a direct influence on the result of measurements.

should be performed. In this chapter I introduced the GUM uncertainty framework as the first possibility to approximate the standard uncertainty. Due to its limitations because of the need of a linear model and the information reduction of probability distributions for influence quantities to their first two moments – expectation value and standard deviation – the Monte Carlo approach of GS1 is more appropriate for application for a larger class of measurement scenarios.

Finally, summarizing simulation data in the third phase is necessary to extract the most important information. Besides the basic statistical parameters, like mean value, standard deviation and symmetric/shortest coverage intervals, information about the shape of the probability distributions, e.g., skewness, or summarized data for histograms and box plots are of interest.

Figure 2.13 shows an advanced measurement scenario, where not the measurement uncertainty of a single independent outcome of a measurement is of interest, but rather the measurement uncertainty of a number of different subsequent measurements that depend on time-varying environmental conditions. This can already appear in very simple scenarios, where the result is the comparison of one or more reference measurements to one or more sample measurements. Current approaches only allow for the definition of a single measurement using conservative estimations to cover a lot of scenarios with the same model and parameter settings. This is sufficient in a lot of cases. Nevertheless, we try to support metrologists in modeling real-life measurements with individual parameter settings. This allows a deeper insight into the measurement system and an analysis of time behavior and is, for example, essential for experts of national and international metrology institutes, as they develop and apply the most accurate measurement standards.

Hence, one major target of the project is to provide a structured way to define the process of modeling. It guides metrologists from laboratories step by step from scratch, where they have a heap of mathematical equations, experience and expert knowledge, specifications, protocols, certificates, handbooks and observation data, to valid and proper models that can also be reused and understood by other metrologists. It is not the goal of this thesis to provide a graphical user interface, but to develop a modeling language that can then be used as the basis for an appropriate interface. I will present a fully functional prototype in a later chapter that allows the definition of simple measurement scenarios and the evaluation of the measurement uncertainty according to GS1.

Before we concentrate on the main part of this thesis in the next chapter, the modeling language of *MUSE*, we will first take a closer look at the advantages and disadvantages of some of the currently available software solutions for measurement uncertainty evaluation.

Difference to existing software solutions There is a lot of professional software [39, 78] using the classical GUM uncertainty framework relying on Taylor series expansion for the evaluation of measurement uncertainty. Of course, there the limitations of the classical GUM approach apply.

Current solutions often rely on spreadsheet applications [8] for the evaluation of measurement uncertainty for simple measurement scenarios. That may be sufficient for cases with a limited number of input quantities, where measurement equipment as influence quantities to the measurement uncertainty are modeled using a limited selection of probability distributions, but are not further specified. The re-usage factor limits itself more or less to copy-paste operations. If measurement devices and equipment should be refined, parts of spreadsheets have to be copied to a new sheet for new evaluations. As the distinction of user and developer diminishes in such cases, the user must have good knowledge and experience with computers and software development. It does not seem appropriate and satisfying that metrologists from laboratories need to be able to master complex implementation tasks. A major point is also that these all-in-one packages provide solutions that are applicable in many cases, but it has to be considered if these black-box solutions really are valid and suffice for an actual situation.

Besides, Erwin Achermann developed in his PhD thesis [4] a framework called MUSAC with the definition language M and a calculation core using the classical GUF approach with uncertainty propagation for the evaluation of the measurement uncertainty. The software was also capable to perform Monte Carlo simulation, but only with a limited number of trials. An extensive graphical user interface [26] was implemented for MUSAC in another project supporting different measurement techniques from analytical chemistry and providing a large database of different measurement items and devices and is a good example for direct support of metrologists not only relying on spreadsheets, but using a more elaborated interface.

As the Monte Carlo method is straightforward and, as already mentioned, a simulation system seems at first glance not to be too complex to implement, it is quite common for metrologists in laboratories to start building their own software solutions. Of course, a lot of problems arise with such individual solutions as requirements get more demanding and functionality is enhanced iteratively over time. If fix coded static models are used in the background, each minor change in the model can generate to a lot of necessary adaptions. Also, issues, as saving large data files efficiently, generating correlated and uncorrelated random numbers and so on, are very important for a proper implementation of a Monte Carlo simulation system.

Often the problem of scaling of simulation systems is underestimated. While coding some software that is only tested with simple scenarios with small numbers of trials, it can lead to problems when one tries to simulate more complex, refined real-world problems afterwards with an appropriate number of trials. Therefore, the goal of this work was to implement a flexible and easy-to-use system that scales well and is optimized for a good trade-off of memory usage and speed. Although parallel computation of simulations is not supported currently, enhancing the simulation system should be straightforward as this was one point we considered from the start of implementing the simulation core as well as for the internal representation of the model.

Beside this project, special Monte Carlo simulation software packages exist. There is, for example, the software *UNCSAM* [34] written in Fortran that concentrates on the evaluation of simulation results. The software provides more than twenty different methods for sensitivity and uncertainty analysis, most of them assuming linear behavior of the model. As one of the key features of the Monte Carlo method is the handling of nonlinear models, I will concentrate more on the analysis not requiring linearities in a later chapter of this thesis.

Another project is EUROMETROS [7], a library of functions, again in Fortran, provided by the *National Physics Laboratory* (NPL), the UK's national metrology laboratory, and EUROMET. The user has to put together functions in Fortran himself, so there is no direct support of the modeling process, but users have to compile self-written source code.

And just to mention another of the many commercial software packages available and giving an example for the broad range of solutions, there is the Microsoft Excel plug-in Oracle Crystal Ball [60]. Originally it was designed as a predictive modeling software, but was found suitable also for Monte Carlo simulation for measurement uncertainty evaluation. It allows a lot of different methods for analyzing simulation data. Most of the commercial products come with advanced user interfaces, but usually they are using spreadsheet layouts as their basis. Many are designed for other areas of interest, like risk assessment and are just adapted for measurement uncertainty evaluation, because of similarities, mainly the Monte Carlo simulation part.

The problem is that for more advanced, high-resolution measurement scenarios with a lot of different equipment, like measurement devices, the flat structure of spreadsheet applications is very limiting. In this project and in the software package *MUSE*, we aim for a concept of a hierarchical, structured model design for advanced and high-resolution measurement scenarios. Concentrating explicitly on problems of metrology allows a more direct approach to assist metrologists in building models for measurement uncertainty evaluation using the common vocabulary in this field. Defining a modeling language does not limit the ability to adapt to given problems, but gives a flexible and improvable structure. More than that, the well-defined language definition used for *MUSE* allows building appropriate graphical user interfaces for parts or the whole functionality of the modeling language and simulation core.

In the next chapter I will first give a short overview of programming and modeling languages and present all the necessary prerequisites for a modeling language to define arbitrarily complex measurement scenarios. Afterwards modeling concepts are introduced step by step with the help of a common real-life measurement from analytical chemistry.

CHAPTER 3

Modeling fundamentals

Derive (if not already available) a model relating a set of output quantities to (input) quantities (raw measurements, suppliers' specifications, etc.) that influence them. Establish distributions for the values of these input quantities. Calculate (in a sense required by context) estimates of the values of the output quantities and evaluate the uncertainty associated with these estimates.

— SSFM Best Practice Guide No. 6, 2004, Page 12 —

A major goal in developing the software project *MUSE* is that the modeling language to describe the measurement scenarios is intuitive and human readable. For a measurement uncertainty evaluation, metrologists have to provide the functional relationships between the influence quantities and the statistical probability distributions with proper parameters to describe the influence quantities, as we already have seen in the last chapter. It is a big advantage for advanced, real-life measurement scenarios, if there is a clear, structured and modular format right from the start as well as a properly defined vocabulary for modeling that is common to people in the laboratories. It should not be necessary for application to learn a complex new programming language. To a certain level this requirement seems – and is indeed – paradox as the simulation system needs all kinds of information from different scientific fields like biology, physics and chemistry mixed with statistical and mathematical information in a structured form, which requires nothing less than a programming or modeling language. Nevertheless, the chosen keywords and the structuring should be intuitive and adequate for the purposes of someone working in a laboratory involved in the evaluation of measurement uncertainties. The modeling language encapsulates the calculation core in a way that there is no direct need for an algorithmic description. Hence, we have chosen a declarative approach. As the modeling language has been designed very specifically for the evaluation of measurement uncertainty in metrology and does not need to support all the structures, like loops and arbitrarily complex conditional statements, of general-purpose programming languages we prefer and use the term domain-specific language or synonymously modeling language as a subset of general-purpose programming languages, like C++, Java, Eiffel etc., henceforth.

Another goal of the project is to always keep in mind that a *graphical user interface* (GUI) is of vital interest in daily-life applications. It is not possible to demand the input of large model and simulation definition files by hand. This means that the domain-specific language should be well-defined and clearly structured using a properly defined grammar, so that a generation of files for simulation is straightforward for other programs. A good example for such a simulation environment is *Modelica* [61], a non-proprietary, object-oriented, equation based language to conveniently model complex physical systems; there exist various commercial and non-commercial approaches to provide graphical user interfaces for the framework. I use in this project a declarative approach, where metrologists provide all information in a well-defined format. The information is then interpreted and transcoded to an internal data structure and finally, the calculation core can start simulating and evaluating the measurement uncertainty. I will show in a later chapter that the modeling language of *MUSE* allows a direct approach to develop a GUI and present a very basic, but already fully functional prototype of a GUI to describe simple measurement scenarios.

In the PhD thesis [4] of Erwin Achermann a domain-specific language called M was developed in combination with a calculation core. It supports evaluation of measurement uncertainty in the daily work of bench chemists in the field of analytical chemistry. The thesis is based on the classical GUM document and hence, the calculation core uses the GUM uncertainty framework to evaluate the model. The project focused mainly on computational issues and strongly used concepts from computer science and procedural programming for the definition of its modeling language M. Additionally, a basic Monte Carlo approach has been implemented, but the program has strong limitations with regard to the number of trials for simulation because of memory management and consumption. Moreover, the modeling language does not support all features needed to apply Monte Carlo for arbitrary measurement scenarios, so that the software does not satisfy all the requirements to apply the Monte Carlo method of GUM supplement 1 properly to more advanced measurement scenarios. For example, the language does only allow defining probability distributions of influence quantities with exactly two input parameters, e.g., for a normal distribution the expectation value and a standard deviation, for a rectangular distribution the mean value and a one-sided width. The following small code fragment in the language M defines the sum $c = a + b$ of two influence quantities, where $a \sim R(3, 0.7)$ is described using a rectangular (`r`) and $b \sim N(2, 0.5)$ a normal or Gaussian distribution (`g`).

```
a = <3 :r 0.7>;
b = <2 :g 0.5>;
c = a + b;
```

This approach is proper and sufficient for applying the GUM uncertainty framework, but restricts the definition of probability distributions too much for the new Monte Carlo approach, which allows using nearly arbitrarily complex functional relationships and probability distributions to describe the behavior of influence quantities for measurement scenarios.

Now, I will continue introducing some ideas concerning the domain-specific language definition and efficiency issues and build up the basis to explain the constructs of the new modeling language afterwards. To show that the language is adequate for our purpose and that it fulfills the requirements for intuitive modeling of complex measurement scenarios, an example from the field of analytical chemistry accompanies the language definition in the subsequent chapters.

3.1 Domain-specific languages

Before we start with the definition of the modeling language for *MUSE*, let us have a short survey and overview of programming language categories. Programming languages can roughly be categorized as follows [51]:

- *First-generation language* (1GL) is a machine-level programming language with no need for an interpreter or compiler.
- *Second-generation language* (2GL) is symbolic assembly language, or in short assembler. It is strongly related to the hardware of the computer.
- *Third-generation language* (3GL) is a “high-level” programming language, such as C, C++, Eiffel or Java. A 3GL is in principle hardware independent, although in praxis portability to different systems is often problematic.
- *Fourth-generation language* (4GL) is designed to be closer to natural language than a 3GL language; an example is the standard query language (SQL) that is used for database queries.

- *Fifth-generation language* (5GL) encodes complex knowledge so that a machine may draw inferences from it. Fields of applications are knowledge-based systems, expert systems, inference engines, etc. An example is the language PROLOG.

In the following we will have a closer look on 4GL. The idea for this class of languages evolved in the late seventies and the beginning of the eighties as application-oriented, specialized task-specific or simply application languages. The important point for this class is what should be accomplished by the program and not how it should be accomplished. In the 1980s the term *4th Generation Language* was introduced by James Martin [50]. 4GL is used to describe high-level programming languages close to natural language, often in the field of database applications. The idea is to provide problem oriented languages for faster and cheaper development of software applications. Programs should be more readable and more transparent to common users with no or only little programming experience. The language should abstract as much as possible from the underlying hardware, software and also algorithmic concepts that are necessary to accomplish the tasks. In the eighties it was an issue that the computational power was rather limited when using 4GL, as the system has to perform complex and advanced tasks to process requests from the user. This is a reason why fourth-generation languages tended to cover only small fields where they could be applied. The probably best-known example for a 4GL application language is the *Standard Query Language* (SQL) [6], where one can build somewhat human-readable English sentences to query relational databases. The language was and still is the standard language for database-related software. A simple example of an SQL query follows to show the clear concept of SQL and its readability:

```
SELECT surname,lastname,address FROM contacts
  WHERE surname LIKE 'Jo%' AND age BETWEEN 23 AND 35
  ORDER BY lastname ASCENDING;
```

This command returns a dataset containing data for the fields *surname*, *lastname* and *address* from table *contacts* of the current database fulfilling the given criteria that the surname starts with the letters 'Jo', like John or Joe and the person is between 23 and 35 years old. The resulting data set is ordered ascending according to the set of returned last names. The command can nearly be read like a common English sentence. This makes it very easy to formulate and understand (at least simple) commands. The language is declarative as programmers just define what they expect as the result, in this case a list of contacts, and provide the information to perform this task. One does not have to care about algorithmic issues in the background, how data is stored or how to get data from the database, where the database itself is an interpretation of one or more files on some storage device. So, 4GLs rely on available functionality and libraries that abstract arbitrarily complex tasks to a simple-to-use interface.

Later the concept of domain-specific languages (DSL) appeared with a very similar approach to 4GL. Usually the term 4GL is used in connection with database-related application languages, whereas the idea of DSLs is to provide human-readable programming languages with proper vocabulary for very specific domains, as the name implies. A good overview of concepts of DSLs and literature is given in [52]. Domain-specific languages are strongly problem driven as already mentioned. Well-known DSLs are, for example, included in MATLAB®, used for computation with vectors and matrices, and the statistical software package R [63] for statistical applications. Also, HTML used for the description of web pages and the language of TEX/LATEX as typesetting system can be seen as domain specific languages. There is a lot of discussion [21, 52, 75] about if and when it is appropriate to develop an own domain-specific language and when to use a general-purpose programming language (GPL). Some of the advantages are that a DSL does not have to be Turing-complete and hence, can reduce its functionality to really necessary domain-specific notations and constructs. The use of a DSL can offer the possibility of analysis, verification, optimization, parallelization, and transformation in terms of DSL constructs. A disadvantage is that the development is challenging in that it requires both extensive domain and language development expertise. It is also a very important question, if a new language is acceptable to the community, requiring support, standardization and maintenance. This is the reason to design the language with very close interaction with people from the community[1].

We decided to develop a new language for the field of metrology, because we think that the need for a well structured way of modeling measurement scenarios for measurement uncertainty evaluation is essential. Measurement uncertainty evaluation is a relatively new field, the first version of the GUM appeared in 1993, the final version of the first supplement has just recently been released in 2008 after two drafts in 2004 and again in 2006. It requires extensive knowledge of the measurement setup and statistical modeling of uncertainty sources. In the beginning, when applying the classical GUM uncertainty framework, only very simple measurements scenarios were of interest, limiting the number of influence quantities to five or six, which were expected as the most important ones. Linearity of the functional relationships was required and interdependencies of influence quantities were omitted where possible. Nowadays, with the new evaluation approach of GS1 a lot of new and additional information can be taken into account in the measurement uncertainty evaluation and a proper refinement of measure-

[1]I want to thank all the people involved in this project very much for a lot of very rich discussions and for spending so much time helping me.

ment models is possible. Hence, the need for a structured and modular modeling concept is bigger than ever.

In this project we introduce a declarative, domain-specific language that allows the definition of equations for the measurand and in more advanced examples multiple measurands. Influence quantities and variables can be defined using a hierarchical concept. The language also provides constructs for analyzing data, a special type of loop for testing different measurement scenarios, and in special cases conditional expressions. The modeling language has to represent the information provided by the user in an adequate form for the simulation core of the system. The language is developed essentially for measurement uncertainty evaluation[2] and only provides structures and constructs of vital interest in application to measurement uncertainty evaluation and the analysis of the results. This allows keeping the language as clean and simple as possible.

Before we start to define the language for the evaluation of measurement uncertainty, we have to take a look at how a computer can process high-level instructions of a DSL in an efficient way. First, I will present XML as a meta-language to define DSLs and afterwards introduce some fundamental design concepts that are necessary to model measurement scenarios.

3.2 XML: Extensible Markup Language

To be able to evaluate measurement uncertainties for arbitrary measurement scenarios with the help of a simulation system, the information about the measurement model and simulation parameters has to be available in computer-readable form. The simulation framework *MUSE* uses documents in a well-defined format corresponding to a given grammar as source. The modeling language is based on *eXtensible Markup Language* (XML). The XML specification [11, 27] is a widespread and accepted open standard of the *World Wide Web Consortium* (W3C). The first recommendation is from February 10, 1998. XML is a simplified subset of *Standard Generalized Markup Language* (SGML), where SGML itself is defined in ISO document 8879 from 1986 [28], a meta-language to describe other markup languages. The application of SGML is mainly for describing documents, where the goal is to separate information and structure from the layout, the way how to display and present the information. A very popular application of SGML using this concept is *HyperText Markup Language* (HTML) that is used in the World Wide Web since the early 1990s to structure and describe the content of web pages and is interpreted by web browsers. Textual information is

[2]I do not like to exclude the application in other areas in general, but I really want to point out that the language was created for this specific domain of metrology.

provided, information about the structure of the document is embedded using so-called markup elements. There is a relatively new standard from 2000 that tries to replace HTML, XHTML. It is a translated version of HTML to the XML standard and syntax.

XML can be used as a meta-language to define domain-specific languages [13, 52,79]. The XML format allows building up a tree of elements and attributes. This structure fits very well to the concept of measurements, where influence quantities can be interpreted as a hierarchical system of other influence quantities as we will see exhaustively in the subsequent chapters. An XML document interpreted as a source file of our DSL contains all information necessary to build the internal data structure required for evaluation of the measurement uncertainty. An initial step before starting to build up an internal representation of the information is to check the correctness according to the XML syntax of the source document and to verify additional rules. There are two levels to prove the correctness of XML documents [27,30].

- **Well-formed** documents follow the syntactical rules of XML.

- **Valid** documents follow additional rules of a grammar that is defined in a separate document or is directly included in an XML document. There exist two different standards for grammar definitions, *XML schema definition* (XSD) or the older *document type definition* (DTD) that is inherited from SGML.

When an XML parser opens an XML document, it will first check for the well-formedness of the document and – if additional DTD or XSD information is provided – the validity according to the grammar will be checked. That means, the usage of existing XML parser solutions, e.g., as for the project *MUSE* from the open source projects *Xalan* [24] and *Xerces* [25] of the *Apache Software Foundation*[3], saves implementation and verification time for own complex parsers for domain-specific languages.

I will not provide a complete description of the XML syntax here[4], but describe only the most important parts in the next section and project-relevant parts in context, where needed. Now we will examine the differences of the two principles of correctness and see how an XML-based domain-specific language looks like and can be defined.

[3]More information about the *Apache Software Foundation*, that supports a lot of open source projects, can be found on their web page `http://www.apache.org/foundation/`.

[4]The complete grammar for the syntax of XML 1.0 is given, for example, in [30, page 318] in extended Backus-Naur form.

3.2.1 XML syntax and well-formed documents

In general, an XML document describes a hierarchical tree structure of elements. A well-formed XML document needs exactly one root element. An element has a descriptive name and uses as markup an opening tag and a closing tag. Tags are enclosed between opening (<) and closing (>) angle brackets. Closing tags have an additional slash (/) after the initial opening angle bracket. In the following first example of an XML document, `flask` is the root element of the document and the parent element of its child elements `volume`, `temperature` and `age`. Each element can have child elements, like `flask`, and/or textual information as content.

```
<flask>
  <volume>100ml</volume>
  <temperature>15°C</temperature>
  <age>1.2years</age>
</flask>
```

Another way to annotate textual content to elements in XML syntax is to use so-called attributes as in the following example.

```
<flask volume="100ml" temperature="15°C" age="1.2years"/>
```

Attributes always accompany an opening tag of an element. They consist of a name followed by an equal sign and a value in pair-wise single or double quotation marks. In the example we additionally use an abbreviation for an "empty" element: if an element does not have any content (textual or child elements), the closing tag can be omitted and the starting tag has to end with an additional slash before the closing angle bracket.

XML documents are case sensitive. This means that the element `<flask>` is not equal to the elements `<Flask>` or `<FLASK>`.

There is a lot of discussion about when to use elements and when attributes [18, Section 5.10]. Major restrictions to attributes are that they can only store textual content and can have no more subsequent elements. Hence, it is not possible to structure information with attributes. The following XML document fragment takes the previous example, and separates information about values and units.

```
<flask>
  <volume>
    <value>100</value>
    <unit>ml</unit>
  </volume>
  <temperature>
    <value>15</value>
    <unit>°C</unit>
  </temperature>
  <age>
```

```
    <value>1.2</value>
    <unit>years</unit>
  </age>
</flask>
```

In an application this separation is very convenient as the document structures information in a way so that the values can be directly interpreted as numbers without the need to parse additionally a sequence of characters for value and unit. A disadvantage is, of course, that this format is somewhat lengthy with extra elements and redundant strings that are introduced with additional opening and closing tags.

In the end it is a subjective decision as to when to use elements and when attributes. In this project I follow a very pragmatic way and use attributes mainly for switches, e.g., on/off or true/false, annotations and settings, e.g., number of histogram bars, type of random number generator, upper and lower limits for values, etc., where appropriate. The following XML document uses, for example, a mixture of attributes and elements and shows how this approach keeps the document easy to read and parse.

```
<flask>
  <volume unit="ml">100</volume>
  <temperature unit="°C">15</temperature>
  <age unit="years">1.2</age>
</flask>
```

Summarizing, we can say that each XML document that suffices the syntactical rules of the XML syntax is well-formed. The following list shows the fundamental syntactical rules of XML.

- An XML document has exactly one root element.
- Empty elements have no content (child elements or textual content); the opening tag of empty elements ends with `/>` or has a closing tag.
- Non-empty elements have an opening and a closing tag.
- Opening and closing tags do not cross.
- All attribute values are quoted with single or double quotes.
- An element cannot have two attributes with the same name.
- Comments always start and end with the character sequences `<!--` and `-->`.

In the first step of parsing XML documents the document is always checked for proper XML syntax. The next (optional) step is then to validate the document according to additional syntactical rules provided as a grammar that describe the allowed structure of an XML document as we will see in the next section.

3.2.2 Valid documents

To specify a user-defined language, the XML standard provides two different ways to describe additional syntactic rules in a grammar to specify the structure of an XML document. The first way is the so-called *Document Type Definition* (DTD), inherited from the former SGML meta-language standard. A DTD mainly consists of definitions for elements and attributes and explains, which elements and attributes are allowed in which context.

XML Schema Definition (XSD) is the second possibility to define rules. This standard is somewhat more complex to read and write, but has big advantages. Besides the definition of elements and attributes, it supports additionally data type declarations. XSD uses 19 different predefined primitive data types, e.g., integers, floating point numbers, Booleans, and strings, and allows deriving more complex data types using restrictions or combinations of the primitive ones. This allows more advanced validity checks for specific textual content of attributes and elements.

Document Type Definition A DTD consists of element and attribute definitions. An element definition is given as

```
<!ELEMENT elementname (contentmodel)>
```

The content model defines if textual content is allowed and/or which child elements are allowed. An excerpt of fundamental rules for the content model follows.

- Required sequence of content models:
  `(contentmodel1,...,contentmodeln)`
- Choice of a content model:
  `(contentmodel1|...|contentmodeln)`
- None or arbitrary many repetitions of content model:
  `(contentmodel)*`
- None or one content model:
  `(contentmodel)?`
- One or arbitrary many repetitions of content model:
  `(contentmodel)+`
- Allowing textual content for element:
  `(#PCDATA)`

Sets of attributes for an element are defined in a single block as follows:

```
<!ATTLIST elementname attributename1  type default
                      attributename2  type default
                      ...
                      attributenamen  type default
>
```

We use mainly the type `CDATA` in this project for attributes. It allows arbitrary textual content for an attribute. Another type is a predefined list of possible values, e.g., for a switch an attribute `mode` with possible values `(on|off)`. Valid definitions for `default` are `#IMPLIED` for optional attributes, `#REQUIRED` for mandatory attributes or a given default value surrounded by quotation marks. Hence, the attribute `mode` for an element `switch` with default mode `off` can be defined as:

```
<!ATTLIST switch mode (on|off) "off">
```

A well-defined XML document is given next, which describes a statistical normal distribution with two parameters, the expectation value μ and the standard deviation $\sigma = 0.2$.

```
<gauss>
  <mu parameter="#mu"/>
  <sigma>0.2</sigma>
</gauss>
```

The following excerpt of a DTD represents the grammar for the example. It defines an element `gauss` with two mandatory child elements `mu` and `sigma` defined as content model. The child elements have an optional – defined by the annotation `#IMPLIED` – attribute `parameter`. The child elements can have textual content, defined by the content model `(#PCDATA)`.

```
<!ELEMENT gauss (mu,sigma)>
  <!ELEMENT mu     (#PCDATA)>
  <!ATTLIST mu     parameter CDATA #IMPLIED>

  <!ELEMENT sigma  (#PCDATA)>
  <!ATTLIST sigma  parameter CDATA #IMPLIED>
```

One problem of DTDs is that element definitions have global scope in the current DTD document and, therefore, element names can only be used once for definition. In the simulation system this is a problem in the definition of probability distributions, as there is, for example, a so-called beta distribution. So the keyword *beta* is used for the definition of such a distribution. But there are also other probability distributions, where the usage of *beta* as parameter is common. Usually such ambiguities should be circumvented. In the given case the different meanings of the same keyword should be clear from the context in practice.

XML Schema Definition DTDs use proprietary syntax, whereas XML schema definition uses valid XML syntax. With XSD it is possible not only to define the structure of an XML document, but also data type definitions are possible for textual content of attributes and elements. For the last example of a document describing a normal distribution a XSD grammar can be defined as follows.

```
<element name="gauss" type="gauss"/>

<complexType name="gauss">
  <sequence>
    <element name="mu" type="attparameter"/>
    <element name="sigma" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="attparameter">
  <simpleContent>
    <extension base="normalizedString">
      <attribute name="parameter" type="string"/>
    </extension>
  </simpleContent>
</complexType>
```

In the example the element `gauss` is defined using a self-defined complex type also named `gauss`. The following type definition contains a sequence with two child elements `mu` and `sigma` of type `attparameter`. The `attparameter` itself is also a complex type definition allowing textual content of the predefined data type `normalizedString` and an attribute named `parameter` of the predefined data type `string`. The complete description of XML Schema can be found in [20]. I only give here in short the corresponding concepts of definitions from the list above for DTD rules.

- Required sequence of content models
  ```
  <sequence>
      contentmodel1
      ...
      contentmodeln
  </sequence>
  ```

- Choice of a content model
  ```
  <choice>
    contentmodel1
  ```

```
    ...
    contentmodeln
  </choice>
```

- None or arbitrary many repetitions of a content model:
  attributes `minOccurs="0"` and `maxOccurs="unbounded"`
- None or one content model:
  attributes `minOccurs="0"` and `maxOccurs="1"`
- One or arbitrary many repetitions of a content model:
  attributes `minOccurs="1"` and `maxOccurs="unbounded"`

The problem of DTDs with global element definitions does not occur in XSDs as element definitions are only local. Hence, different definitions using the same keyword are possible.

Some people criticize DTDs as the syntax of DTDs is not directly XML based, e.g., the element definitions `<!ELEMENT>` do not have corresponding closing tags. Also some limitations occur for the definition of XML-based languages using DTDs, e.g., no data type can be defined for the content of elements or attributes. XSD overcomes some of the shortcomings of DTDs, but is somewhat more complicated to read and understand not only because of its lengthy definitions, but also because the definition can go into much more detail. For the software project *MUSE* we provide DTDs as well as XSDs. I prefer to use the more advanced XSDs as the data type concept prevents more invalid XML documents right from the start of parsing than if using DTDs.

3.2.3 Navigation in XML documents

XML documents describe hierarchical structures of elements and attributes. Using existing parsers reduces the implementation time and prevents failures in an own implementation of a parser. There is also a very sophisticated way to traverse through XML documents, the *XML Path Language* or in short XPath. There is no need for an exhaustive explanation at this point; I just want to show the nice and simple concept of walking down the tree of elements. Let's have a look at a simple example again and assume first, that there is a single flask named `f1` defined in an XML document.

```xml
<flask name="f1">
  <volume>
    <value>100</value>
    <unit>ml</unit>
  </volume>
  <temperature>
```

```
    <value>15</value>
    <unit>C</unit>
  </temperature>
  <age>
    <value>1.2</value>
    <unit>years</unit>
  </age>
</flask>
```

For a traversal of the flask device to the value of the temperature, we can use the path `/flask/volume/value`. The first slash (`/`) tells the system that we want to start with the root element. Then the tags are used for further traversal, separating with the slash again until we reach the element of interest. Now suppose there would be another flask with attribute `name` set to `f2` defined in the same document. Traversal of the path would now lead to two elements in an element list containing both child elements for the volume from both flasks. Because of that reason it is possible to check for specific attribute values in XPath. Assuming that the names of the flask devices in the document are unique, the path to the value of flask `f1` is then given as `/flask[@name='f1']/volume/value`. The constraint in brackets says that the textual content of the attribute (`@`) named `name` must be equal to the string `f1`.

This short overview just scratches the surface of the very powerful and complex XML Path Language. For example, also backward traversal in the tree of elements is possible, and there are a lot of built-in functions to test for specific properties of attributes and elements. The introduction of the basic concepts in this section shows that it is straightforward to analyze and traverse the hierarchical tree structure of documents. This is just another argument for using XML to define our modeling language.

3.3 Concepts for modeling measurements

One of the main goals of the project is to define a domain-specific language that contains all structures to handle the evaluation of measurement uncertainty, even for complex measurement setups. It should also be as close as possible to people who work in laboratories doing measurements, bringing the fields of metrologists, physicist, chemists, biologists and statisticians together. Hence, the modeling language is developed in close interaction with people from metrology.

An important question regarding modeling measurement scenarios is how deep the analysis should go and how detailed the model should be. In [47, Section 5.2] the authors discuss coarse models and advantages of refinement as well as validation of modeling software. The main point for the level of model detail is that it should be consistent with the information and type of data available. They also propagate the idea to start with coarse models, apply sensitivity analysis to identify the most important influences and redefine the model based on this information. I completely agree with this approach and the project provides the tools for metrologists to analyze and refine the model where needed. Robinson [66] also shows that the level of detail is very situation dependent. The conclusion is to keep the model as simple as possible and iteratively refine where necessary and possible. Time required to build an over-complex simulation cannot be reclaimed, but further detail can always be added.

In [73, 74] there is a broad discussion about how to model measurement scenarios from scratch for measurement uncertainty evaluation according to the classical GUM uncertainty framework. The authors suggest an approach, describing how to get from an actual measurement scenario to the mathematical formulation and the expression of dependencies between influence quantities that affect the measurement. They start with an adapted version of cause-and-effect diagrams, as we have seen in Figure 2.6, to identify all uncertainty sources X_i assuming that the measurand itself is embedded in the system of influence quantities. Equations of the form

$$X_1 = h(Y, X_2, \ldots, X_n)$$

are extracted and transformed to get the measurand Y explicitly

$$Y = f(X_1, X_2, \ldots, X_n)$$

if possible. This part of modeling is highly relevant for finding and arguing concepts regarding the modeling language of this project. We start at the point, where a model is given with an explicit expression of the measurand Y, fully according to the GUM and GS1 approach.

Another influence on this thesis and the modeling language is the paper of Dieter Kind [43], where a measurement is split into individual parts. It applies an alternative idea to cause-and-effect diagrams and visualizes the dependencies of more or less independent parts of the measurement. The premise that the individual parts have to be independent is necessary for his approach. For our modeling language we do not make this claim as we will see later on. Heinrich Schwenke [70] examined in his PhD thesis the possibilities to split the measurement process in separate parts, trying to build up a database with basic devices and parts of the equation of the measurand in the background. He already has implemented a Monte Carlo simulation system in the year 1999, five years before the first draft of GS1 was published in 2004. We even want to go a step further than this work in additional stronger structuring of measurement scenarios and not only observing a single measurement, but measurement series. This means that not a single measurement takes place, but we work with various reference and sample measurements and try to analyze dependencies and effects.

From analyzing and comparing various measurement scenarios, we have obtained an idea of structured modeling by using hierarchical models and this is the reason, why I think that XML is a very appropriate tool for the definition of the modeling language of *MUSE*. This is also discussed in article [80]. The key to this idea is that we can depict measurement scenarios into activities and static items as persons, equipment, substances, samples etc. This leads to a hierarchical system of influence quantities and functional relationships. The equation of the measurand can be interpreted as a directed graph of nested influence quantities and processes. Static items are then described in an abstract way and can be used in context as individual instances and parameter settings. In the definition of activities the instances are used for evaluation of the measurement uncertainty for an actual measurement scenario.

Having these concepts in mind, we can now define the very basic fundamentals of the modeling language. Using this basis, I will then start to explain the more advanced concepts by the help of a real-world example first at a coarse modeling level, refining it step-by-step by introducing new modeling concepts and constructs. We have just seen how to parse and traverse complex hierarchical structures using XML and have gotten a basic idea of the hierarchical modeling of measurement scenarios. But for measurement uncertainty evaluation it is also essential to be able to parse mathematical equations efficiently as the dependencies of influence quantities to a measurement are described using functional relationships defined as equations. Another concept introduced in the next section is the modeling of influence quantities with uncertainties using probability distributions. With this prerequisites and the definition of identifiers and numbers in the following paragraph, we can concentrate afterwards on more advanced modeling techniques and structures.

3.3.1 Identifiers and real numbers

Throughout the project description we will assume identifiers provided as names or IDs, concatenated identifiers and real numbers in equations or as parameters to have the format given in extended Backus-Naur Form (EBNF) [1]:

```
con_ident  = {identifier "."},identifier;
identifier = (letter|spec_char),{letter|spec_char|digit};
real       = [sign],{digit},["."],digit,{digit},[exponent];
exponent   = ("e"|"E"),[sign],digit,{digit};
sign       = "+"|"-";
digit      = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9";
spec_char  = "@"|"#"|"_";
letter     = "a"|"b"|"c"|"d"|"e"|"f"|"g"|"h"|"i"|"j"|
"k"|"l"|"m"|"n"|"o"|"p"|"q"|"r"|"s"|"t"|
"u"|"v"|"w"|"x"|"y"|"z"|
"A"|"B"|"C"|"D"|"E"|"F"|"G"|"H"|"I"|"J"|
"K"|"L"|"M"|"N"|"O"|"P"|"Q"|"R"|"S"|"T"|
"U"|"V"|"W"|"X"|"Y"|"Z";
```

Hence, identifiers are allowed to use the special characters `@`, `#` or an underscore `_` and leading letters as first character, followed by an arbitrary order of digits, letters and the special characters defined as `spec_char`. The name `con_ident` is an abbreviation for concatenated identifier and is used for nested models in more complex measurement scenarios. Real numbers can be provided in common scientific format. According to the EBNF-definition, for example, the following numbers are valid numbers:

```
-425
.23
-0.84
174.18e-2
```

The modeling language of *MUSE* is (like XML) case-sensitive, so, for example, the identifiers `flask1` and `Flask1` are not considered as equal.

3.3.2 Parsing formulas

It is important to see that the equation of the measurand – maybe divided in subsequent equations – is a central part in evaluating the measurement uncertainty independent of the evaluation method. The dependencies of influence quantities in a measurement are always described using functional relationships. In principle it would be possible to also use an XML-based representation of formulas. But we think of equations as the central point in modeling that should be provided in infix notation as-is to be better readable and get a better recognition factor for parts of models. Another advantage is that structural changes in the functional relationship can be more easily applied than if the structure of an XML document per se has to be changed.

While the definition of formulas can be provided in the common infix notation, internally we use postfix notation[5] for the evaluation of the equations because of computational reasons. It allows us to use a simple stack data structure for parsing equations without the need of taking operator precedence or parentheses into account. Let's use a small example to explain this important issue and show the advantages of postfix notation to the common infix notation.

[5]Postfix notation is also called *reversed polish notation* and is a modification of the prefix notation or polish notation, invented in the 1920s by Jan Lukasiewicz.

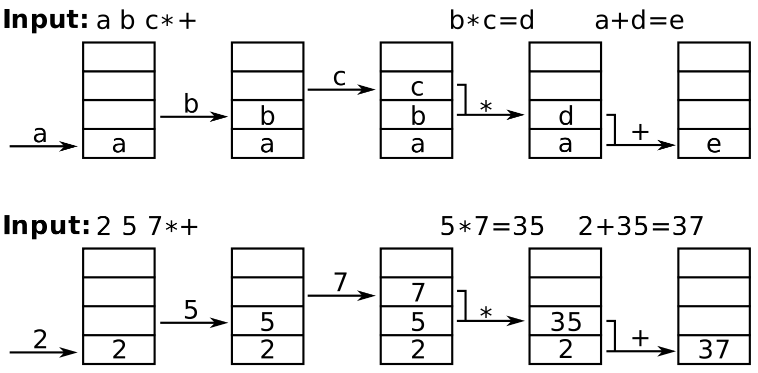


Figure 3.1: Parsing of the formula $a+b\cdot c$, and as example $2+5\cdot 7$, in postfix form. The result is always the first element in the stack.

If a formula in infix notation like

$$a+b\cdot c$$

should be evaluated, a computer has to first build a parse tree internally to see what operation to apply first. It stores the a, reads the plus and continues reading the b. At this point the system has to decide if the operation can be carried out immediately or if it has to parse further on, because of operator precedence. It has to read the multiplication operator and the c to know that it first has to compute the multiplication. This means in practice that before any operation can take place, the system has to parse the whole formula and construct a complete parse tree to derive a sequence how to compute the formula.

In postfix notation the formula from above is

$$abc\cdot+$$

Figure 3.1 visualizes the parsing process, the reading operations and the application of operators. The big advantage is that for parsing a single stack structure and a pointer to the last element of the stack suffice. The parsing process in the example starts with reading the a and pushing it on the stack. It continues with pushing b and c on the stack. Now the multiplication operator follows. As we have postfix notation we know that the last two elements of the stack have to be used as arguments for the operator. These elements are popped from the stack; the result of the multiplication is then pushed on the top of the stack again. Reading the plus sign afterwards, the operation can also be performed immediately; the last two elements on the stack are popped and used as parameters and the result is finally pushed on the stack. So, the result of an evaluation is always the last element on the stack. If more than one element remains on the stack, something went wrong and the simulation system has to act accordingly.

The postfix notation allows a straightforward evaluation internally for the formula parser with a simple stack data structure. A big disadvantage is that formulating equations in the postfix format directly is very inconvenient as it is not that common in practical use[6]. Therefore, a preparser to the simulation system has been developed in the first part of a semester work[7] to convert familiar infix notation to formulas in postfix format for computation. In the work not only the conversion itself was of interest, but also some optimization issues like computing constant expression immediately, elimination of dispensable characters, e.g., tabulators, spaces, line breaks, and keeping the stack size small. For the last task operations should be computed as soon as possible. For example, the suboptimal formula

$$abcde++++$$

needs to store five stack elements before it starts to perform the summing operations. Fortunately, it can be converted to

$$ab+c+d+e+$$

and after conversion it only needs two stack elements. Adding operations are carried out in the latter form as soon as possible.

An important role for efficient evaluation of formulas provided in a character string format is played by a preprocessing routine, replacing multi-character identifiers to identifiers of minimal length. This allows more efficient and faster parsing. Another approach to a faster simulation system is to calculate block wise, performing multiple calculations in one run. Here, parsing the formula once returns a set of results at once. This is implemented and was already explained, when introducing the Monte Carlo simulation approach in the last chapter. Of course, the evaluation of formulas can be further improved in translating the structure in an internal formula tree without the need of repeating string matching operations.

In the rest of the work we will use the more common infix notation for examples, but keep in mind that these formulas are converted, stored and computed in the postfix notation internally for computational reasons.

3.3.3 Probability distributions

Besides mathematical equations and formulas the quantification of uncertainty sources using probability distributions is essential for evaluating the measurement uncertainty. The classical GUM uncertainty framework has major restrictions for

[6]Actually, Hewlett Packard provides a number of different pocket calculators that understand postfix notation, e.g., the HP 50g. New versions support both, postfix and infix notation, as input.

[7]I want to thank Hasan Kaharan at this point once more for his excellent contribution to the project.

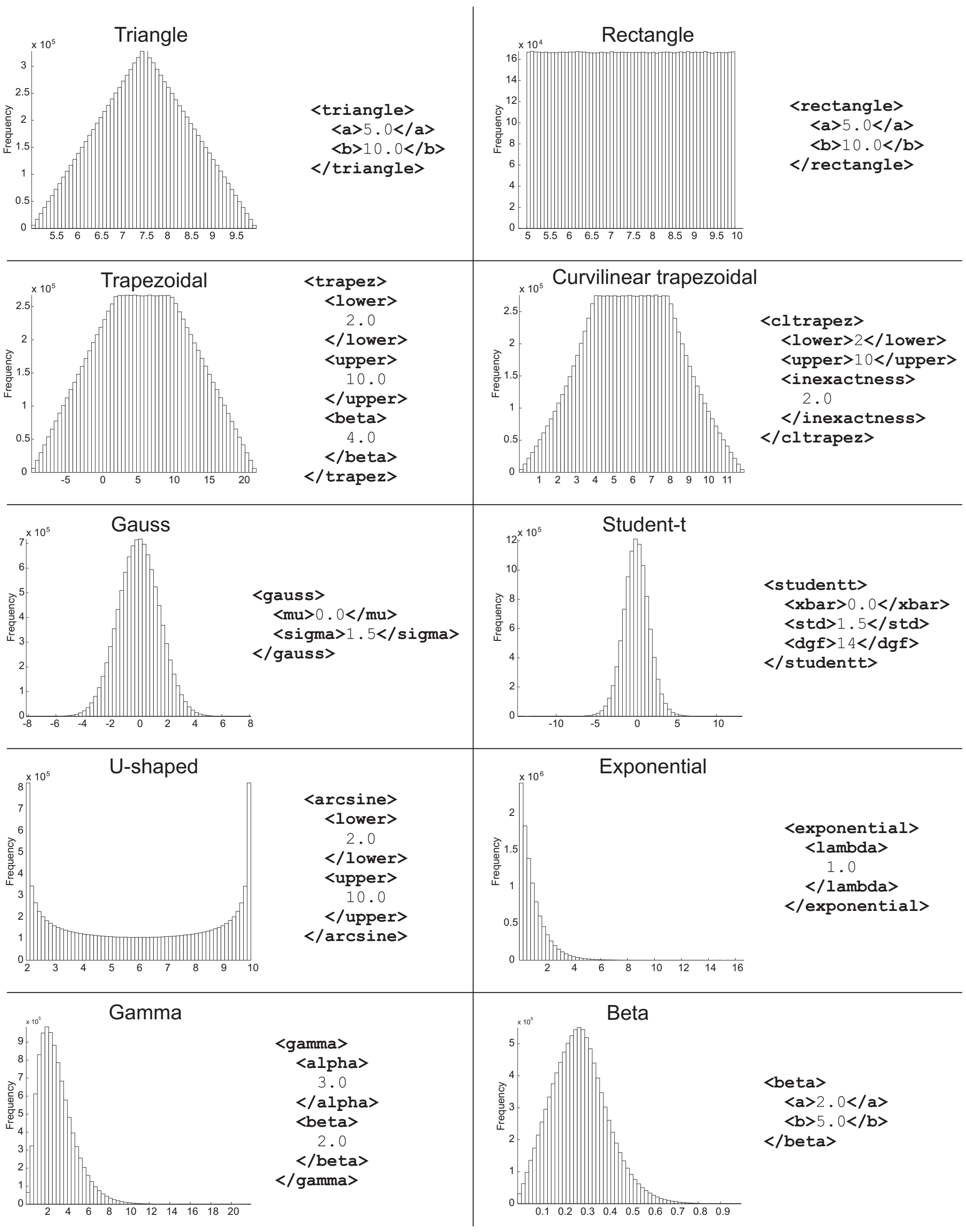
the statistical modeling of input quantities, as already mentioned in Section 2.2.4. One limitation is that information about probability distributions that describe the influence quantities is reduced rigorously to the first two moments, expectation value and standard deviation, for the evaluation of the measurement uncertainty. Hence, modeling languages that apply the classical GUF approach – for example, the system MUSAC [4] with its language M – have a plausible argument to reduce the number of parameters for probability distributions to two.

With the introduction of GS1 and Monte Carlo simulation this limitation is not applicable and appropriate anymore. This means that the modeling language has to be more flexible as arbitrary probability distributions can have varying numbers of parameters. Appendix A.1 presents the XML schema definition for the supported probability distributions of the software package *MUSE* at the time of writing. GUM supplement 1 gives detailed instructions for generating random numbers for some of the most common continuous probability distributions in section 6.4 and additionally in annex C. Figure 3.2 shows histograms resulting from Monte Carlo simulation with 10^7 trials each and the corresponding XML code fragments for the definition of the probability distributions in the modeling language. Additionally, the common beta distribution has been added to the list of continuous probability distributions. The second Figure 3.3 shows discrete probability distributions supported by *MUSE*. The algorithms used to generate random numbers for probability distributions not mentioned in GS1 are provided, for example, in [67, Section 2.4].

Example Student's *t*-distribution To show the capabilities and flexibility of the modeling language of *MUSE*, we use a fragment of an XML document that defines a Student's *t*-distribution with parameters mean $\bar{x} = 2.88$, standard deviation $s = 0.41$ and 9 degrees of freedom.

```
<distribution>
  <studentt>
    <xbar>2.88</xbar>
    <std>0.41</std>
    <dgf>9</dgf>
  </studentt>
</distribution>
```

Probability distribution definitions are encapsulated as the content of an element `distribution`. Afterwards the element definition for the actual probability distribution follows with the according elements to describe the parameters as content. We use full words as names for the elements (as for the probability distributions) in most cases. Abbreviations are only used where very long words appear, e.g., `std` for standard deviation and `dgf` for degrees of freedom, and would decrease readability or the abbreviation is very common in the community, e.g., for the



```
<triangle>
  <a>5.0</a>
  <b>10.0</b>
</triangle>
```

```
<rectangle>
  <a>5.0</a>
  <b>10.0</b>
</rectangle>
```

```
<trapez>
  <lower>
    2.0
  </lower>
  <upper>
    10.0
  </upper>
  <beta>
    4.0
  </beta>
</trapez>
```

```
<cltrapez>
  <lower>2</lower>
  <upper>10</upper>
  <inexactness>
    2.0
  </inexactness>
</cltrapez>
```

```
<gauss>
  <mu>0.0</mu>
  <sigma>1.5</sigma>
</gauss>
```

```
<studentt>
  <xbar>0.0</xbar>
  <std>1.5</std>
  <dgf>14</dgf>
</studentt>
```

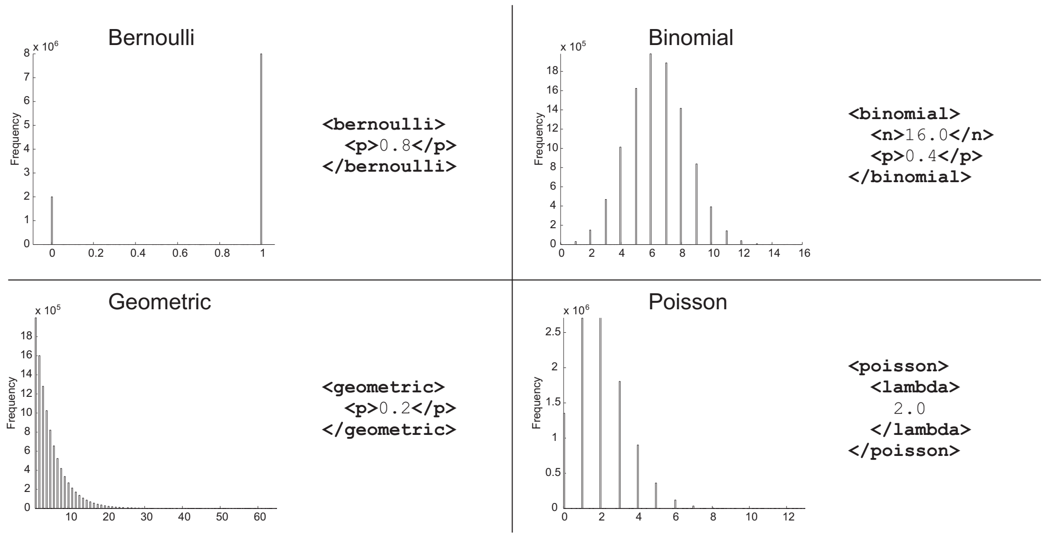
```
<arcsine>
  <lower>
    2.0
  </lower>
  <upper>
    10.0
  </upper>
</arcsine>
```

```
<exponential>
  <lambda>
    1.0
  </lambda>
</exponential>
```

```
<gamma>
  <alpha>
    3.0
  </alpha>
  <beta>
    2.0
  </beta>
</gamma>
```

```
<beta>
  <a>2.0</a>
  <b>5.0</b>
</beta>
```

Figure 3.2: Supported continuous probability distributions given in GS1 and additionally the beta distribution; the histograms result from Monte Carlo simulations with 10^7 trials each.



```
<bernoulli>
  <p>0.8</p>
</bernoulli>
```

```
<binomial>
  <n>16.0</n>
  <p>0.4</p>
</binomial>
```

```
<geometric>
  <p>0.2</p>
</geometric>
```

```
<poisson>
  <lambda>
    2.0
  </lambda>
</poisson>
```

Figure 3.3: Supported discrete probability distributions; the histograms result from Monte Carlo simulations with 10^7 trials each.

parameters of the probability distributions. This enhances the readability of large models as well.

The *t*-distribution is a good example at this point to show the flexibility of the modeling language, as there is a second and even a third possibility for definition of this distribution. The following XML code fragment defines the same *t*-distribution as in the last example, but it states the number *n* of measurements explicitly instead of using the parameter `dgf` for the degrees of freedom.

```
<distribution>
  <studentt>
    <xbar>2.88</xbar>
    <std>0.41</std>
    <n>10</n>
  </studentt>
</distribution>
```

Finally, it is possible for the Student's *t*-distribution to be defined using a list of measured values explicitly. The values can, for example, result from an actual measurement.

```
<distribution>
  <studentt>
    <values>
      <value>2.3</value>
      <value>3.2</value>
      <value>1.9</value>
      <value>3.1</value>
      <value>2.7</value>
      <value>2.6</value>
      <value>3.7</value>
      <value>3.1</value>
      <value>2.3</value>
      <value>3.9</value>
    </values>
  </studentt>
</distribution>
```

Here, the parameters for the mean value $\bar{x}$, the standard deviation s and the degrees of freedom from the first example are calculated after parsing the probability distribution to the internal structure and reading the provided values. The conversion from a specific representation to another one is a trivial task in the case of the Student's t-distribution, but it already saves time when defining models in practice as all of the representations appear in daily life.

Both language definition concepts – DTD and XML schema – give us the necessary flexibility to provide different ways for parameter settings, whereas readability remains very good. Another advantage is that users in laboratories do not have to convert probability distribution parameters from one representation to the other, but just provide present information as is.

Figure 3.4 shows an excerpt of the class diagram of the internal representation of probability distributions. There is one main class *distribution*, handling requests to generate new probability distributions in a static method *createDistribution* and defining the basic interface that has to be implemented for derived classes for specific probability distributions.

Correlated normal distribution A central point in modeling is how to define causal dependencies between different uncertainty sources. The guide recommends [37, Section 5.2.4] to identify the source of the dependency itself and to refine the model by introducing a new uncertainty source that resolves the implicit dependency. Another common way to describe the linear relationship of two or more influence quantities is with an n-dimensional multivariate normal distribution $\mathbf{X} = (X_1, \ldots, X_n)^T$. The definition requires a vector $\mathbf{x} = (x_1, \ldots, x_n)^T$ containing the best estimates for each normal distributed random variable X_i and

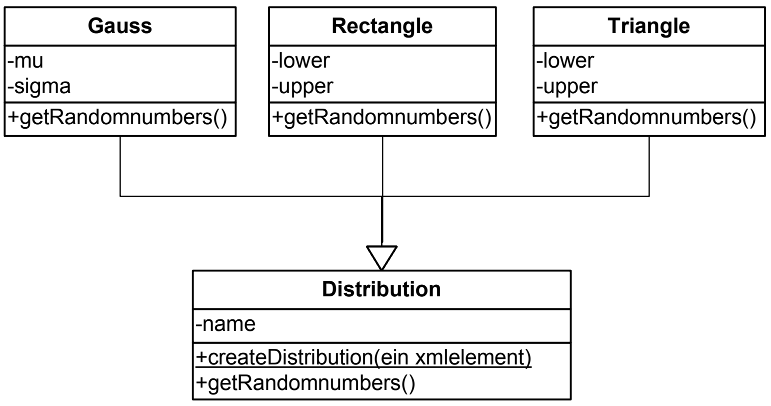


Figure 3.4: Class diagram for rectangular, triangular and normal distributions; for the sake of simplicity, only some of the most important member variables, methods and parameters for methods are shown.

an uncertainty (or covariance[8]) matrix $\mathbf{U_x}$ that contains the covariance coefficients

$$cov(X_i, X_j) = E[X_i X_j] - E[X_i]E[X_j] = E[(X_i - E[X_i])(X_j - E[X_j])]$$

in the corresponding elements of the matrix. For the diagonal this is the squared standard uncertainty, the variance $\sigma^2_{X_i} = u^2(x_i)$ of the corresponding normal distribution of random variable X_i.

GS1 [22, Section C5] and also [67, Section 2.5.2] explain in detail how to draw random numbers from such multivariate normal distributions. The algorithm uses the property of the uncertainty matrix to be strictly positive definite and hence, an existing Cholesky decomposition.

The XML code fragment below describes a two-dimensional correlated normal distribution with estimated expectation values $\mathbf{x} = \begin{pmatrix} 2.0 & 3.0 \end{pmatrix}^T$ and a covariance matrix

$$\mathbf{U_x} = \begin{pmatrix} 2.0 & 1.9 \\ 1.9 & 2.0 \end{pmatrix}.$$

```
<distribution>
  <correlated id="d1" slot="1">
    <muvector length="2">
      <element>2.0</element>
      <element>3.0</element>
    </muvector>
    <uncertaintymatrix size="2">
```

[8]GS1 introduces in section 3.11 and the corresponding note 3 the expression *uncertainty matrix* for the covariance matrix. The modeling language allows both expressions `uncertaintymatrix` and `covariancematrix` for the definition of a covariance matrix.

```
      <row>
        <cell>2.0</cell>
        <cell>1.9</cell>
      </row>
      <row>
        <cell>1.9</cell>
        <cell>2.0</cell>
      </row>
    </uncertaintymatrix>
  </correlated>
</distribution>
```

For larger uncertainty matrices it is possible to define cells using cell indices independently to utilize the symmetry of the matrix. Index `index1` has to be always greater than `index2`; otherwise the system will generate a warning. This prevents redundant definitions of cells. Elements that are not explicitly provided are assumed to be zero. The analogous matrix definition would be as in the following code fragment.

```
    <uncertaintymatrix size="2">
      <cell index1="1" index2="1">2.0</cell>
      <cell index1="1" index2="2">1.9</cell>
      <cell index1="2" index2="2">2.0</cell>
    </uncertaintymatrix>
```

As the multivariate normal distribution is used for the evaluation of different uncertainty sources in different parts of the model, it has a unique identifier. The attribute `slot` is set here to one and defines that the first probability distribution $X_1 \sim N(2,2)$ should be used from the correlated distribution definition.

The following XML fragment builds on the last one and defines a probability distribution referring to the second probability distribution $X_2 \sim N(3,2)$ from the definition above by setting the attribute `slot` to two.

```
<distribution>
  <correlated equals="d1" slot="2"/>
</distribution>
```

Figure 3.5 shows the two resulting histograms again with 10^7 trials per simulation with the multivariate normal distributions and a corresponding scatter plot with the first 1000 paired data points of the result.

The internal class structure of this scenario is visualized in Figure 3.6 showing again only exemplarily methods and member variables of interest. A helper class *correlation* generating random numbers for different instances of the *correlated* class containing all relevant information is introduced.

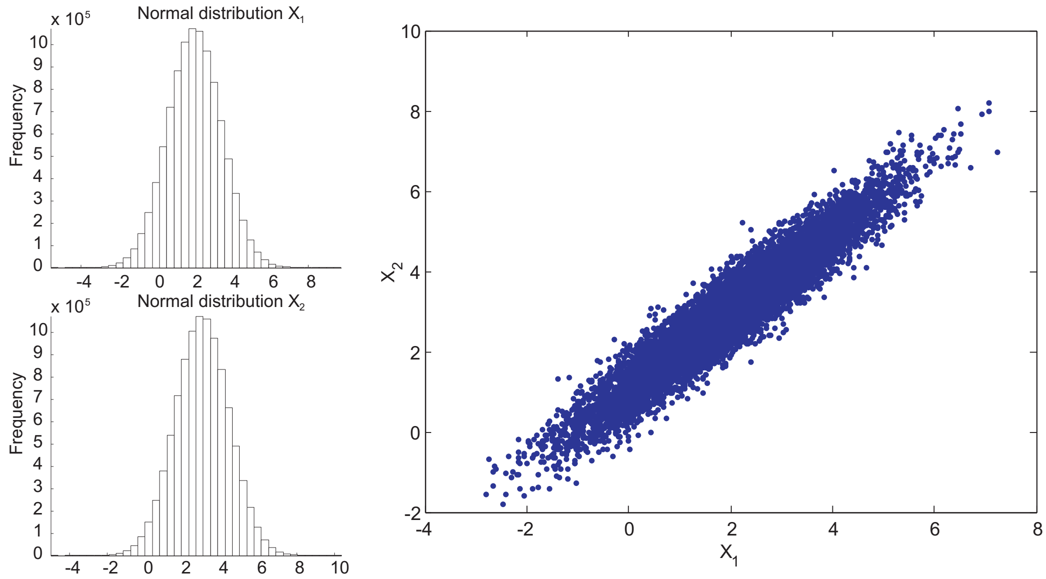


Figure 3.5: The figure shows on the left-hand side the two histograms for the normal distributed quantities $X_1 \sim N(2,2)$ and $X_2 \sim N(3,2)$ with a covariance $cov(X_1, X_2) = 1.9$ in a simulation of 10^7 trials each. The right-hand side shows the corresponding scatter plot using the first 1000 simulation values of the two correlated distributions.

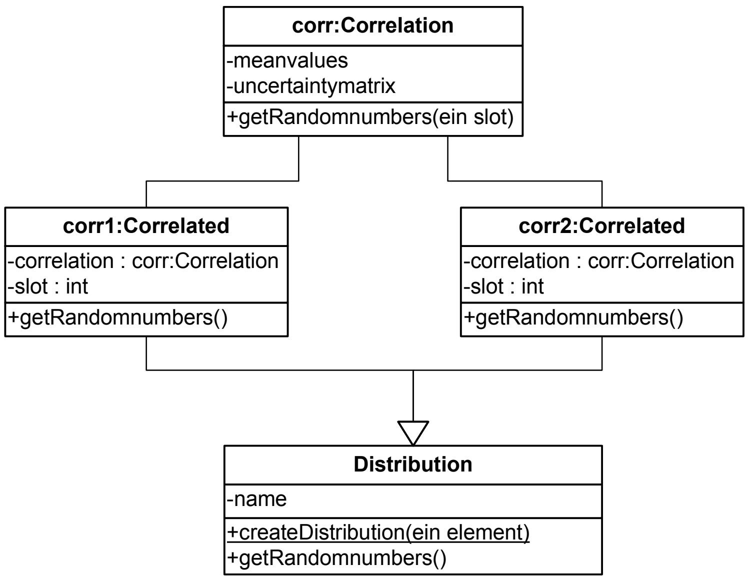


Figure 3.6: Class diagram visualizing the classes and necessary dependencies between them to generate random numbers for a correlated distribution

Parameterizing distributions

If we define complex and higher-resoluted models in later examples it is important that parameters of probability distributions can be set at a global level. The user in a laboratory eventually doesn't have to know exactly where some demanded parameter is used in the model equations or as probability distribution parameter explicitly, but just using predefined models as black boxes. A description of the parameter value should then be sufficient for applying a model for measurement uncertainty evaluation.

The modeling language of *MUSE* provides two distinct approaches to set values for parameters of probability distributions not only to fixed values as in the examples above, but to values defined in another subtree of the model, maybe even in another context.

Explicit parameterization The first way is to parameterize the probability distribution parameters themselves. To accomplish this, a parameter of a probability distribution is annotated with an attribute `parameter` defining an identifier. If during simulation this identifier appears in an internal list of parameter settings, this value is used for the parameter of the probability distribution. The identifiers need not be unique, because two probability distribution parameters using the same identifier will get the same setting or value.

```
<distribution>
  <gauss>
    <mu parameter="#mu"/>
    <sigma parameter="#sigma">0.1</sigma>
  </gauss>
</distribution>
```

It is also possible to set default parameters, e.g., parameter `#sigma`[9] uses the value 0.1 in the example if the provided identifier of the parameter is not found in the list of parameters.

Using variables as parameters The second way is to use variables in the textual content definition of the parameters of the probability distribution. Each probability distribution parameter in *MUSE* cannot only be set to a constant value, but it can also be defined as a formula. The advantage of this form of parameterizing is that variables can be used in formula strings as shown in the next example, where the standard deviation given as a variable *gsigma* is reduced to 70 %. Additionally, variables can consist of more complex structures as we will see later on.

[9]The leading # character of the parameter name is not mandatory, but only recommended; it allows distinguishing between identifiers of parameters and variable, influence and instance names as we will see in later examples.

```
<distribution>
  <gauss>
    <mu>2</mu>
    <sigma>gsigma * 0.7</sigma>
  </gauss>
</distribution>
```

The disadvantage, as always with global variables also in other programming languages, is that one has to know where these variables appear and are used in the model structure and equations. Besides, if two distinct submodels use the same identifier for a global variable and both submodels are used in the same measurement uncertainty evaluation, special care has to be taken if different values should be applied. This can get difficult to manage very quickly even for simple measurement scenarios. Another disadvantage is that the evaluation will fail if the variable is not defined. It is not possible to set default values when variables are used. Hence, we strongly recommend the usage of parameterized probability distribution parameters.

Now we have all the fundamentals to describe measurement scenarios for the evaluation of the measurement uncertainty. Using an XML-based domain-specific language, a practical way of defining functional relationships together with an efficient method to parse the equation of the measurand and the possibility to define arbitrary probability distributions allows an implementation of a simulation system for complex, real-life measurement scenarios with an uncertainty evaluation according to GS1. In the next chapter we will develop a model for a measurement scenario from scratch based on different examples taken from the field of analytical chemistry to introduce the concepts of the modeling language one after another.

CHAPTER 4

Modeling a measurement

Although this Guide *provides a framework for assessing uncertainty, it cannot substitute for critical thinking, intellectual honesty, and professional skill. The evaluation of uncertainty is neither a routine task nor a purely mathematical one; it depends on detailed knowledge of the nature of the measurand and of the measurement. The quality and utility of the uncertainty quoted for the result of a measurement therefore ultimately depend on the understanding, critical analysis, and integrity of those who contribute to the assignment of its value.*

— GUM, 1993, Section 3.4.8 —

Now we have all the prerequisites – possibility to express hierarchical systems in an XML-based domain-specific language, efficient parsing of mathematical expressions and definition of probability distributions to describe influence quantities – to model measurement scenarios for measurement uncertainty evaluation. In the following chapters an actual measurement scenario is built up from scratch, where we apply the preceding concepts. After explaining the principle of the measurement technique, we will start with the most basic elements that are necessary for the evaluation of the measurement uncertainty. With the help of the example it is easy to explain the capabilities of the domain-specific language of *MUSE*. I will explain XML code fragments in detail where necessary and appropriate.

4.1 Basics of a titration measurement

Titration is one of the oldest measurement techniques in analytical chemistry. It is a very common and highly accurate measurement. The articles [29, 65] used for teaching bench chemists from Metrohm[1], the leading manufacturer of titration devices, give a detailed explanation of the titration measurement technique. Example A3 in *EURACHEM/CITAC Guide – Quantifying Uncertainty in Analytical Measurement* [19] shows how to quantify the measurement uncertainty for a titration measurement using the classical GUM uncertainty framework. Our example will be based on this knowledge. Titration is a very precise, cheap, and economic technique to determine the amount of a sample substance in a solution. Therefore, it is widely used in the chemical industry. Some well-known examples for the application of titration measurements are determining the amount

- of calcium and magnesium to specify hardness of water,
- of salt in sea water,
- of vitamin C (ascorbic acid) in medicine and food,
- of calcium in milk, or
- of sulfur in crude oil.

Figure 4.1 explains the concept of an experimental setup for such a titration measurement, Figure 4.2 shows two titration devices from the manufacturer Metrohm that are nowadays used in laboratories. The principle is that titrant solution drops from a burette into a beaker with a solution of a sample or reference material. The solution in the beaker is stirred continuously to mix the drops and the solution. In a concrete example of an acid/base titration the formula for the chemical reaction is

$$RCOOH + Na^+OH^- \longrightarrow RCOO^-Na^+ + H_2O,$$

where $RCOOH$ is a carboxylic acid, Na^+OH^- is sodium hydroxide and the result is an R-acetate $RCOO^-Na^+$ and water H_2O. By replacing R for example with CH_3 the chemical reaction [29, Section II.1.7] changes to

$$CH_3COOH + Na^+OH^- \longrightarrow CH_3COO^-Na^+ + H_2O$$

and describes the reaction of acetic acid found in vinegar with sodium hydroxide to form sodium acetate and water.

[1]Metrohm AG, Oberdorfstrasse 68, 9101 Herisau, Switzerland

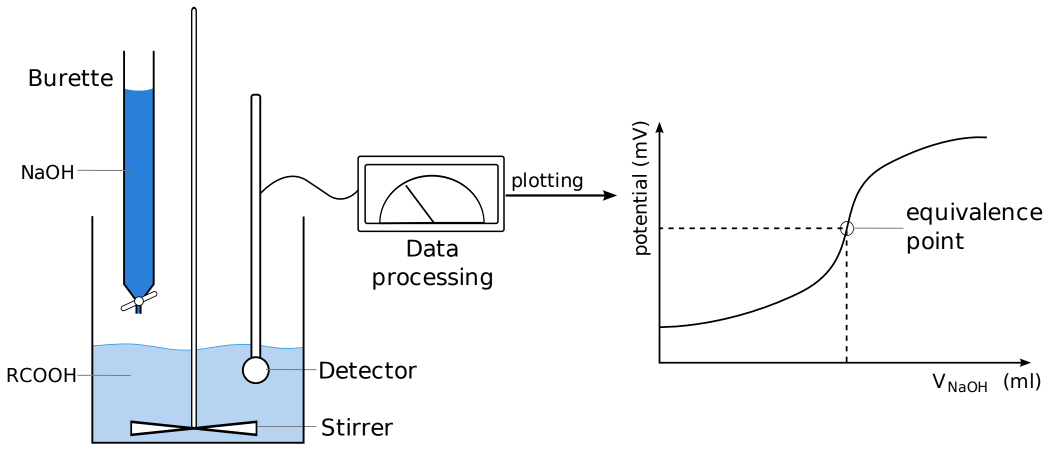


Figure 4.1: Measurement setup for an acid/base titration measurement; the result is a titration curve, where the equivalence point should be determined.

The electric potential and the pH-value of the solution change continuously, while the chemical reaction proceeds during a titration measurement. A detecting electrode measures the potential of the solution as the measurement takes place, until the so-called equivalence point for the potential is reached. This means that the solutions have been mixed in exactly the right proportions.

The solution of a sample substance for titration consists of a specific amount of sample material together with solvent, e.g., highly pure water. The amount of interesting particles, e.g., benzoic acid molecules, in solution of a sample substance can be determined if we know the amount of sample solution from the burette used to get to the endpoint and the result of a second titration measurement with the solution of a reference substance where we know the number of molecules exactly. Hence, the solution of a reference substance consists of reference material with a well-known purity and the same solvent as used for the sample.

We have to know the characteristics of the parameters used to describe the measurement to determine the purity of the substance. The quantities are given in the cause-and-effect diagram in Figure 4.3. They are in detail

- the concentration c_{Tit} as a well-known concentration of the titrant in the burette in (mol/ml),
- the mass m of the substance in solution in (g),
- the molar mass M as a constant ratio of atoms to mass in (g/mol), and
- the volume V_{Tit} of the titrant as the amount of titrant used from the burette in (ml).

Figure 4.2: State-of-the-art titration devices from the manufacturer Metrohm

The next step is to define the functional relationship of the quantities and the purity of the substance. For the example of a titration measurement the purity is given as

$$pur = \frac{c_{Tit} \cdot M \cdot V_{Tit}}{m}. \tag{4.1.1}$$

To determine the actual purity pur_{sam} of the sample substance, we additionally need a reference measurement for comparison with the sample measurement. So, we again use the concept of comparing a well-known attribute of the reference to determine the same unknown attribute of a sample, as was already explained at the beginning of Chapter 2. We know that the concentration of the titrant in the burette is theoretically[2] the same in both measurements. Hence, we can convert Equation 4.1.1 correspondingly and obtain two equations, one for the sample measurement

$$c_{Tit,sam} = \frac{m_{sam} \cdot pur_{sam}}{M_{sam} \cdot V_{Tit,sam}}$$

and one for the reference measurement

$$c_{Tit,ref} = \frac{m_{ref} \cdot pur_{ref}}{M_{ref} \cdot V_{Tit,ref}}.$$

[2]As already explained in earlier chapters, two measurements can practically never take place at the same time with the same environmental conditions. Nevertheless, in practice this is assumed to simplify the measurement. We try to stress this fact in this project and look at the effects of taking this variation into account in later examples.

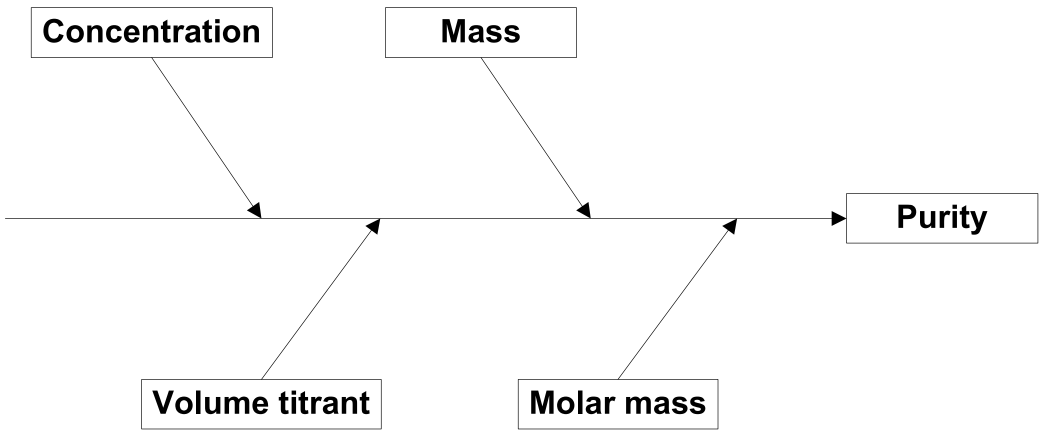


Figure 4.3: Cause and effect diagram showing the influence quantities of a titration measurement.

We assume in a first approximation that the equality $c_{Tit,sam} = c_{Tit,ref}$ of the concentrations holds and get

$$\frac{m_{sam} \cdot pur_{sam}}{M_{sam} \cdot V_{Tit,sam}} = \frac{m_{ref} \cdot pur_{ref}}{M_{ref} \cdot V_{Tit,ref}}.$$

As we are looking for the same substance in sample and reference measurements in the current example, the molar mass is the same on both sides. Thus, we simplify and continue to extract the interesting quantity, the purity pur_{sam}, and get to the measurement model

$$pur_{sam} = \frac{m_{ref} \cdot pur_{ref}}{V_{Tit,ref} \cdot m_{sam}} \cdot V_{Tit,sam}. \tag{4.1.2}$$

Preparing solutions for sample and reference measurements require measurements themselves, e.g., weighing material on a balance. To evaluate the measurement uncertainty for a titration measurement, the uncertainty of this processes in the preparation of samples and references have to be taken into account, too. Each quantity in the measurement model contributes to the measurement uncertainty of the titration. To determine a more accurate and reliable result, the titration is repeated several times for the sample. The reference measurement should be repeated periodically for comparison and verification, too, as visualized in Figure 4.4. Although, in practice the number of reference measurements is usually kept as small as possible, e.g., once every one or two weeks, as one such titration measurement takes between 20 minutes and up to 2 hours and costs money, of course. Thus, we will not end up with a single measurement, but a whole sequence of measurements. It is one goal of this work to be able to model even such advanced series of measurements in an organized and structured fashion.

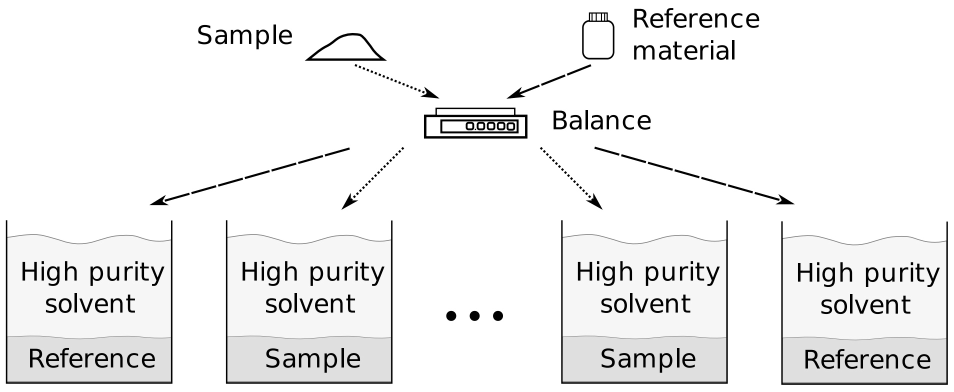


Figure 4.4: To prepare solutions for sample/reference measurements some material from the sample/reference is put into a beaker and then filled with solvent, e.g., highly pure water.

But this is just an outlook to more advanced scenarios, because before we can start to model series of measurements, we have first to concentrate on the proper modeling of a single measurement.

In the following sections we start to model a single titration measurement for measurement uncertainty evaluation introducing necessary basic structures and aspects of the modeling language and enhance as the model gets refined and the resolution is increased. Besides, we show how the concepts of the domain-specific language are then mapped to the internal structure of the simulation system for a straight-forward evaluation of the model.

4.2 Basic influence quantities

To evaluate the measurement uncertainty according to GUM and GS1 the equation of the measurand has to be evaluated, where influence quantities defined as random variables contribute to the measurement uncertainty. The random variables themselves are described using probability distributions. We show in the next section how to depict the measurement model – the equation of the measurand – into separate definitions of measurement items, like measurement devices, substances, etc., and build a hierarchical system of influence quantities for each item individually. This allows a modular and stepwise modeling approach. The system remains manageable and a higher-resolution can be applied to models by incremental refinements.

Please note that the measurement models and assigned values used in the presented examples result mainly from personal communication with experts[3] from

[3] I want to thank here explicitly Matthias Rösslein for his patience, for answering all the ap-

the field of analytical chemistry and therefore, represent real expert knowledge used for uncertainty evaluation in practice. We will start now with the most basic influence quantity from the example, where its uncertainty contribution to the measurement can be described by only a single probability distribution.

4.2.1 Purity of a substance

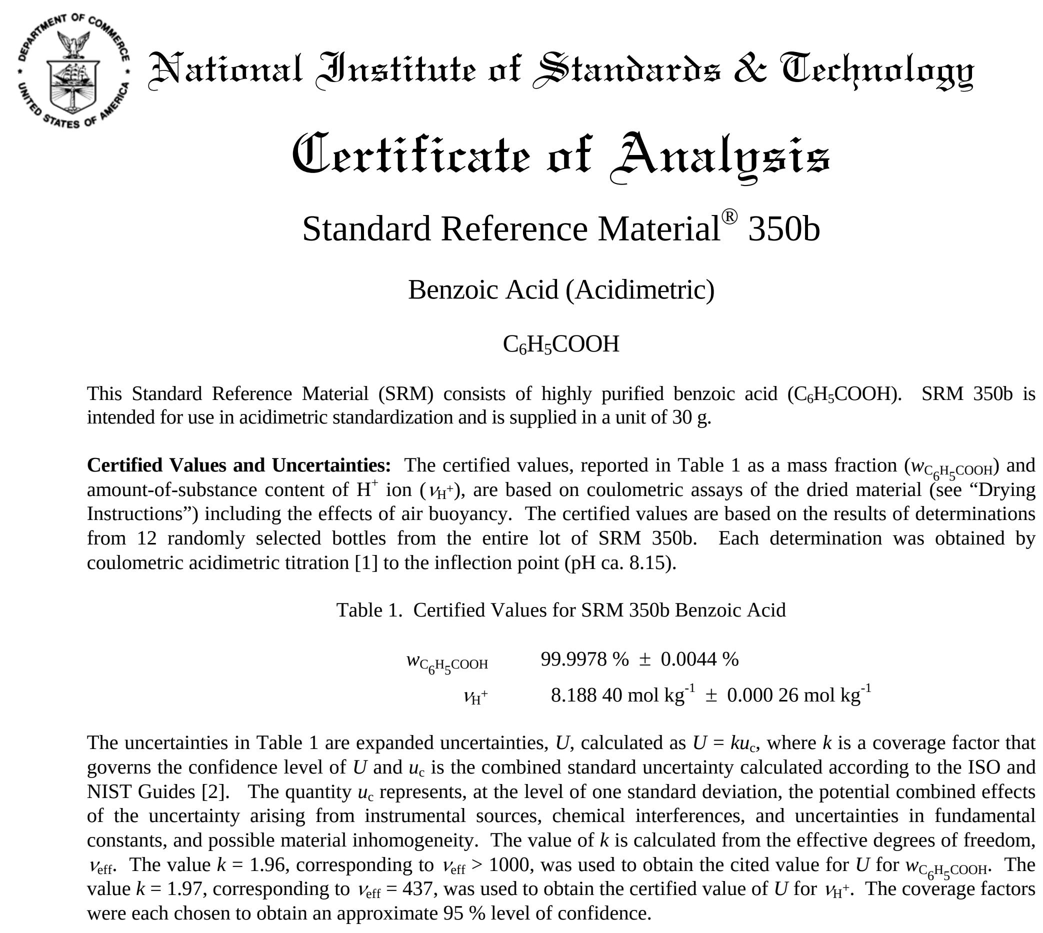
After describing the concepts and the mathematical fundamentals that are necessary for a titration measurement, the influence quantities should be modeled for uncertainty evaluation. We start with the simplest form of models, influence quantities that can be described directly using a common statistical probability distribution. Hence, in the following example we define the model for a standard reference material that can be ordered for example from the National Institute for Standards and Technology (NIST)[4], which consists of highly purified benzoic acid. Figure 4.5 shows the relevant part of the certificate for measurement uncertainty evaluation describing the product. The certificate is available online [59] and reports the purity (mass fraction) of the substance as 99.9978 % with an expanded uncertainty of 0.0044 % and a coverage factor $k = 1.96$ for a 95 % coverage interval. Additionally, it declares the effective degrees of freedom $\nu_{eff} > 1000$. We assume that a normal distribution is adequate in this case, because the t-distribution is practically equal to a normal distribution if we consider the degrees of freedom. As a widely accepted rule of thumb starting at about 30 degrees of freedom the Student's t-distribution can be adequately approximated by a normal distribution.

We will start now with the very basic structure for each model and simulation in *MUSE*. A definition file for the simulation is the basis, starting with the root element `simulation` in the hierarchy. It encapsulates at least an element `calculation`. The calculation section contains parameters relevant to simulation and analysis. It has to contain at least an element `measurand` that defines the output quantity or quantities. The following code fragment evaluates the simple formula $2+5$ in each trial of the simulation.

```
<simulation>
  <calculation>
    <measurand> 2 + 5 </measurand>
  </calculation>
</simulation>
```

pearing questions, and the extensive support during my work.

[4]Measurement Services Division, National Institute of Standards and Technology, 100 Bureau Drive, Stop 2320 Gaithersburg, MD 20899-2320, USA.



National Institute of Standards & Technology

Certificate of Analysis

Standard Reference Material® 350b

Benzoic Acid (Acidimetric)

C_6H_5COOH

This Standard Reference Material (SRM) consists of highly purified benzoic acid (C_6H_5COOH). SRM 350b is intended for use in acidimetric standardization and is supplied in a unit of 30 g.

Certified Values and Uncertainties: The certified values, reported in Table 1 as a mass fraction ($w_{C_6H_5COOH}$) and amount-of-substance content of H^+ ion (ν_{H^+}), are based on coulometric assays of the dried material (see "Drying Instructions") including the effects of air buoyancy. The certified values are based on the results of determinations from 12 randomly selected bottles from the entire lot of SRM 350b. Each determination was obtained by coulometric acidimetric titration [1] to the inflection point (pH ca. 8.15).

Table 1. Certified Values for SRM 350b Benzoic Acid

$w_{C_6H_5COOH}$	99.9978 % ± 0.0044 %
ν_{H^+}	8.188 40 mol kg^{-1} ± 0.000 26 mol kg^{-1}

The uncertainties in Table 1 are expanded uncertainties, U, calculated as $U = ku_c$, where k is a coverage factor that governs the confidence level of U and u_c is the combined standard uncertainty calculated according to the ISO and NIST Guides [2]. The quantity u_c represents, at the level of one standard deviation, the potential combined effects of the uncertainty arising from instrumental sources, chemical interferences, and uncertainties in fundamental constants, and possible material inhomogeneity. The value of k is calculated from the effective degrees of freedom, ν_{eff}. The value k = 1.96, corresponding to ν_{eff} > 1000, was used to obtain the cited value for U for $w_{C_6H_5COOH}$. The value k = 1.97, corresponding to ν_{eff} = 437, was used to obtain the certified value of U for ν_{H^+}. The coverage factors were each chosen to obtain an approximate 95 % level of confidence.

Figure 4.5: First page of a certificate [59] from NIST for highly purified benzoic acid with a purity of 99.9978 % and a corresponding expanded uncertainty of 0.0044 %, coverage factor $k = 1.96$ and effective degrees of freedom $\nu_{eff} > 1000$

Let's return to the example of the benzoic acid substance and model the probability distribution from the certificate. We can simply sample from a normal distribution with the reported parameters. The following code fragment accomplishes this task and additionally introduces the analyzing module.

```
1  <simulation>
2    <calculation>
3      <analyze mode="on" histbars="40"/>
4
5      <variable name="pur" unit="g/g">
6        <distribution>
7          <gauss>
8            <mu>0.999978</mu>
9            <sigma>2.2e-3</sigma>
10         </gauss>
11       </distribution>
12     </variable>
13
14     <measurand>
15       <formula>pur</formula>
16     </measurand>
17   </calculation>
18 </simulation>
```

We still have the very basic structure of the simulation definition starting with the section `simulation` that contains a child element `calculation`. The new element `analyze` in line 3 inside of the calculation section turns on the analyzing module. The system evaluates the resulting data file(s) immediately after simulation and extracts the most common statistical parameters, like arithmetic mean, standard deviation and coverage intervals, as well as information for visualization if turned on. Details of the computation have been discussed in a previous chapter in Section 2.4. Due to this option, there is no immediate need for additional statistical software after simulation, unless if further investigations or additional statistical parameters or tests have to be considered. The attribute `histbars` tells the analyzing module to combine simulation values into 40 histogram bars.

In line 5 the definition of a variable named `pur` as abbreviation for purity is introduced to model the purity of the substance. Variables in the calculation section have global scope, which will be of importance for later constructs, and can be defined using constant values or contain a probability distribution as in the current case. The variable uses a normal distribution $N(0.999978, 2.22 \times 10^{-3})$ with the corresponding expectation value from the certificate. The standard deviation for the purity is determined from the certificate, too, as the expanded measurement uncertainty U is defined as $U = ku_c$, where $u_c = U/k = 2.2 \times 10^{-3}$ is nothing else than the combined standard uncertainty, say the standard deviation.

Figure 4.6 shows the internal representation of the model. The simulation

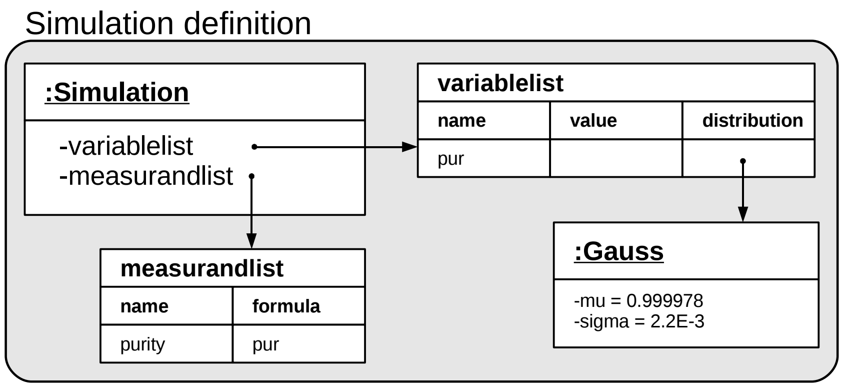


Figure 4.6: Definition of a variable `pur` for influence quantity concentration; variables in calculation section have global scope.

object contains lists of dynamic length for different elements. In the example we introduce two of them. The first one, the variable list, contains identifiers for global variables, a corresponding value or, as in the current case for the variable `pur`, a pointer to an instance of a probability distribution. The second list is the list of formulas for measurands. We will use this list extensively in later examples, where not only one output quantity is of interest, but a list of them. When parsing the formula provided for the evaluation of a measurand, the system compares identifiers with the elements in the list of variables and returns the corresponding values if found.

To visualize the result of a simulation with 10^7 trials, Figure 4.7 shows the corresponding histogram. It appears that the substance can result in a purity above 100 %, which seems impossible at first glance. Actually, in this example it does make sense. The reason is that impurities cannot be excluded in detection of the molecules of the substance. As the measurement does not distinguish between molecules of impurities and wanted molecules, there is a realistic chance to detect a purity with more than 100 % concentration corresponding to the substance. But there are indeed examples, where measurement uncertainty evaluation can lead to doubtful results if influence quantities near physical limits are to be investigated. We will return to this point in a later chapter.

4.2.2 Basic models

It would be a tedious task to manage large models for complex measurement uncertainty evaluations with many influence quantities if every influence quantity had to be defined as a variable in the calculation section of the simulation definition. A large list of influence quantities would not represent the structure of a measurement and dependencies of influence quantities properly. Besides, the

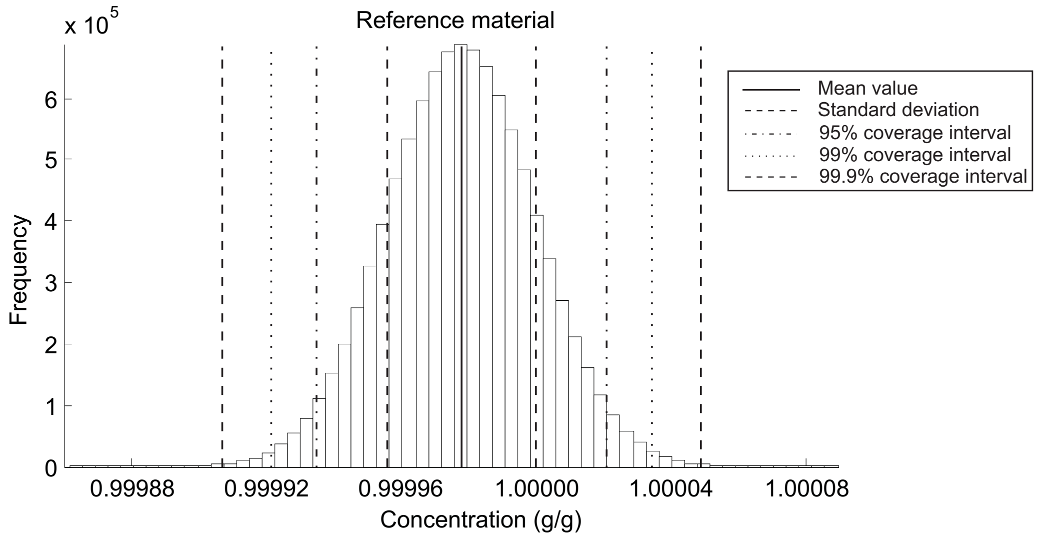


Figure 4.7: Histogram of simulation with 10^7 trials for purity of reference material from NIST; note that values above 100 % for the concentration are allowed in this case.

definition of the equation of the measurand would get lengthy and difficult to read and interpret.

Hence, we introduce so-called *basic models*. Each basic model is defined in a separate XML document. Basic models encapsulate abstractly defined uncertainty sources like measurement items and equipment, e.g., devices and substances. This allows an individual modeling of distinct logical parts of a measurement. The idea is to have a library of such properly designed basic models. Then arbitrarily many independent instances of basic models can be used for a simulation afterwards. Therefore, basic models are also an efficient way to reduce redundant definitions of measurement models or parts of models.

```
<model name="purity" targetname="pur">
  <influence name="pur" comment="purity" unit="g/g">
    <distribution>
        <gauss>
          <mu>0.999978</mu>
          <sigma>2.2e-3</sigma>
        </gauss>
    </distribution>
  </influence>
</model>
```

The XML code for modeling the benzoic acid standard reference material as a basic model consists of an element `model` with an annotated attribute `name`, which corresponds to the file name of the basic model. The second mandatory attribute is the attribute targetname. It identifies an influence quantity as the root element and starting point for parsing. The element model in this example contains a single element influence. An influence quantity can then define a probability distribution for evaluation in the same way as we have already seen for variables in the calculation section. An influence quantity has always at least the two attributes name and comment, which are used to identify and describe the influence quantity. The name has to be unique within a basic model. Furthermore, an influence quantity can be annotated additionally with an attribute for the unit definition.

The mandatory simulation document using an instance of the new basic model now includes a section instances. In the example the new section contains the definition of an instance of the basic model purity with the identifier pur in line 3. This instance replaces the variable definition from the calculation section that defined our purity in the earlier example.

```
1  <simulation>
2    <instances>
3      <instance name="pur" model="purity"/>
4    </instances>
5
6    <calculation>
7      <analyze mode="on" histbars="40"/>
8      <measurand>pur</measurand>
9    </calculation>
10 </simulation>
```

The identifier of an instance, in the example pur, is used for the evaluation of the equation of the measurand just like a variable. An instance is an independent realization of the model structure defined in a basic model with its own scope. The identifier for an instance is visible for each part in the simulation definition, so that the instance can be used for evaluation.

Figure 4.8 shows the internal representation of this scenario. Besides the list of variables, there is now a second object, the model box, containing information about instances of basic models. When evaluating the equation of the measurand, the list of instances of the model box is checked first for suitable identifiers. If the look-up fails the list of variables from the calculation section is checked for a match in a second run.

This modular structure allows the definition of complex and high-resolution models without losing the overview and thus, keeping the models manageable. Basic models avoid redundant modeling, as instances of basic models use the same model definition. We proceed in modeling a more complex influence quantity contributing to the measurement uncertainty of the titration measurement. In the following section we will see that each instance has its own scope and parameter settings, and that hierarchical model structures can be encapsulated in basic model

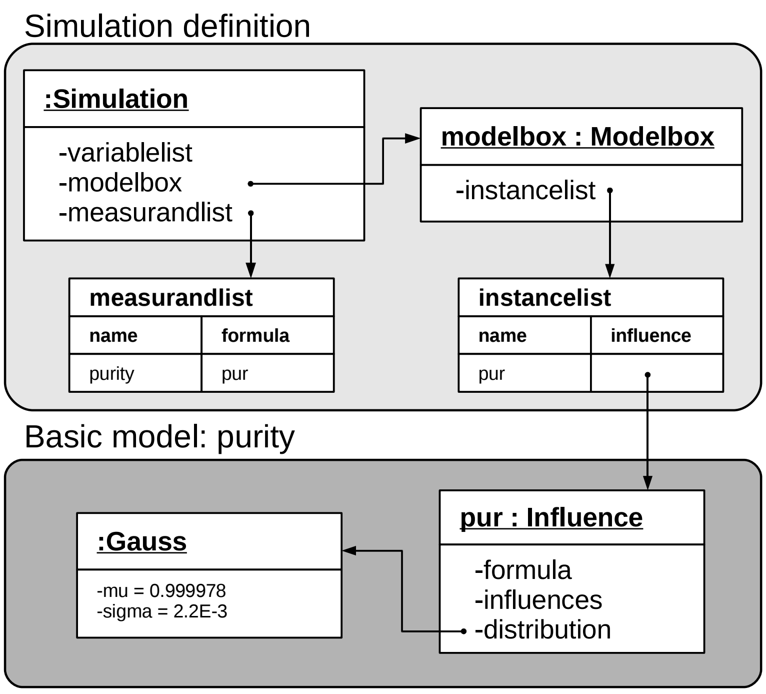


Figure 4.8: Modeling the purity substance for a titration measurement as basic model; basic models keep the definition of measurement scenarios better manageable and prevent redundant definitions.

files.

4.3 Advanced influence quantities

The purity substance from the last section is described using a single probability distribution defined by fixed parameters. In general, influence quantities consist of arbitrarily complex and nested structures, where different probability distributions are combined using functional relationships to describe the dependencies. Hence, advanced influence quantities contain a formula and a list of subsequent influence quantities.

4.3.1 Measurement device balance

We continue with our titration example, where we start to model more advanced influence quantities. The first one is the balance, which is used to measure the amount of substance for a solution. The concept of weighing with a modern balance is described in Figure 4.9. Figure 4.10 shows the corresponding cause-

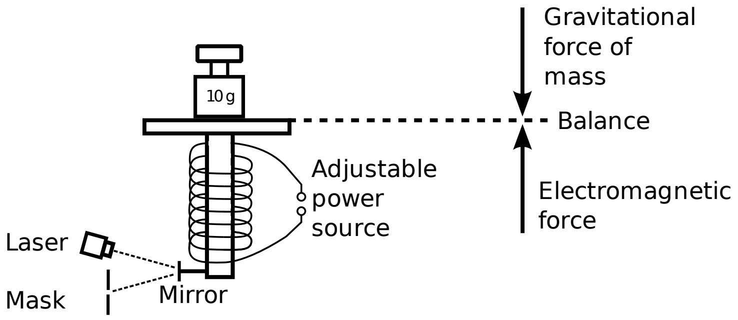


Figure 4.9: If a sample is put on the balance, the power source is regulated until the laser spots to the hole in the mask via the mirror again. This indicates that the electromagnetic force is equilibrated with the gravitational force of the mass.

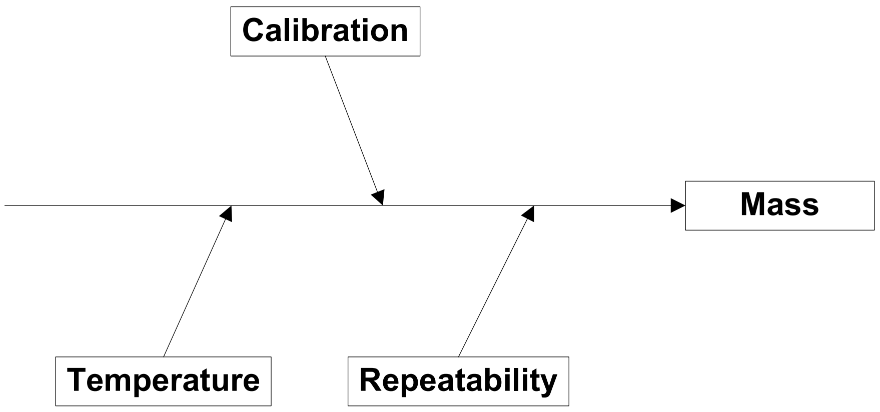


Figure 4.10: The cause-and-effect diagram shows that the influence quantities calibration, temperature and repeatability contribute to the measurement uncertainty of the mass of a sample on the balance.

and-effect diagram with the influence quantities contributing to the measurement uncertainty. We start with this definition and explain the different influence quantities and refine the structure of the model for the balance in the next section. The functional relationship of the influence quantities is additive in this case and defined as

$$m = m_{cal} + m_{rep} + m_{temp}, \tag{4.3.1}$$

where each influence quantity contributes to the measurement uncertainty. All values used to describe the influence quantities for the balance are provided in grams or milligrams as they contribute to the mass m. The quantity m_{cal} is an uncertainty source from calibration of the balance, where the calibration curve is usually provided by the manufacturer. The concept of the calibration is shown in Figure 4.11. The quantity m_{rep} describes, how the system behaves on repeated

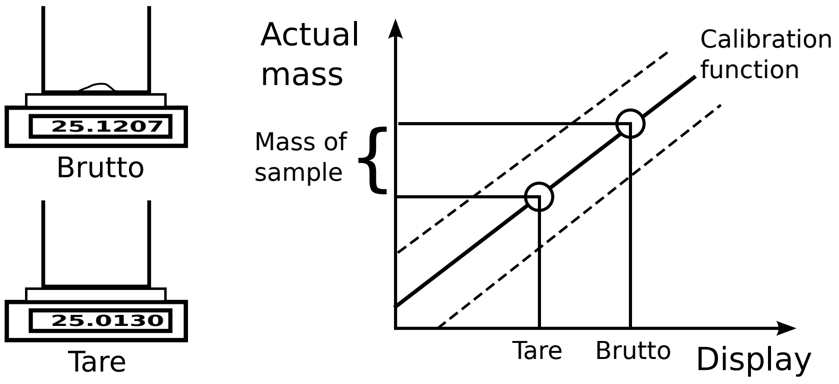


Figure 4.11: Two measurements are necessary to determine the mass of a sample. First the beaker is measured as tare weight; afterwards the beaker including the sample is measured. The difference of the two measurements is the actual mass of the sample.

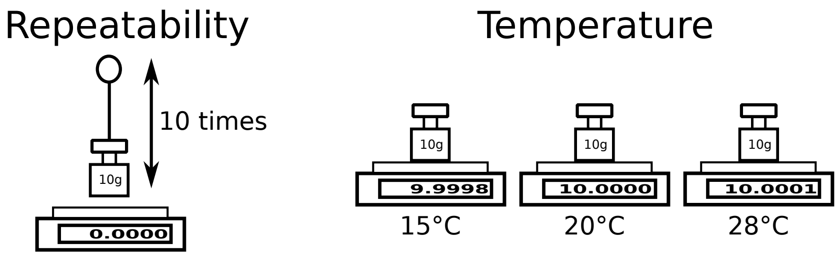


Figure 4.12: Left: to determine the repeatability of the balance a gauge block of well-known mass is measured 10 times; right: expansion and contraction of parts of the balance caused by variation of ambient temperature have to be taken into account.

measurements, whereas m_{temp} expresses the uncertainty related to changes of the ambient temperature. Modern balances have a built-in automatic calibration module. Hence, only an uncertainty source considering this module has to be taken into account. The concepts of this two influence quantities are visualized in Figure 4.12. We encapsulate the balance right from the start in a basic model. The XML code for a first basic model of a balance follows and will be discussed in detail afterwards.

```
<model name="balance" targetname="m">
  <influence comment="balance" name="m" unit="g">
    <formula>mcal + mrep + mtemp</formula>
    <influences>
      <!-- Definition influence quantity calibration -->
      <influence comment="calibration" name="mcal" unit="g">
        <distribution>
```

```
 8           <gauss>
 9             <mu>0.200</mu>
10             <sigma>2.4E-05</sigma>
11           </gauss>
12         <distribution>
13       </influence>
14
15       <!-- Definition influence quantity repeatability -->
16       <influence comment="repeatab." name="mrep" unit="g">
17         <distribution>
18           <gauss>
19             <mu>0</mu>
20             <sigma>5.4E-06</sigma>
21           </gauss>
22         </distribution>
23       </influence>
24
25       <!-- Definition influence quantity temperature -->
26       <influence comment="temperature" name="mtemp" unit="g">
27         <distribution>
28           <gauss>
29             <mu>0</mu>
30             <sigma>8.1E-08</sigma>
31           </gauss>
32         </distribution>
33       </influence>
34     </influences>
35   </influence>
36 </model>
```

This is a first example of the definition of an influence quantity depending on other influence quantities. We use Equation 4.3.1 for the balance in an initial element `formula` in line 3. Afterwards, the influence quantities for the evaluation of the formula are described. For the example we have the definition of three subsequent influence quantities starting at lines 6, 16, and 26 embedded in the section `influences`. All of them are in a first step assumed to be properly represented by normal distributions with suitable parameters for a specific balance device, e.g., an AT200 model from the manufacturer Mettler-Toledo[5]. As we have a very specific device, many of the used values for parameter settings remain the same for all instances of this type of device and can be set to default values. They may be used as-is and there is no need to adapt them for different measurements scenarios. Nevertheless, one parameter will differ in almost any case from measurement to measurement; it is the amount of material on the balance. We apply the method to parametrize the probability distribution parameters in the basic model as has been

[5]Mettler-Toledo GmbH, Laboratory & Weighing Technologies, Greifensee, Switzerland

already explained in Section 3.3.3. So we parametrize the according distribution parameter and replace line 9

```
9             <mu>0.200</mu>
```

with

```
9             <mu parameter="#gross">0.200</mu>
```

This means that we can now set the parameter when instantiating the basic model in the instances section of the simulation definition. If no parameter is provided, the value 0.200 g will be used as default value. The following simulation definition sets the parameter of an instance `m` for a balance to 115 mg. We introduce a new section `parameters` in the instance definition of the balance. Parameters are used individually for the current instance as each instance has its own scope. They are applied while building the internal structure of the basic model.

```
<simulation>
  <instances>
    <instance name="m" model="balance" unit="g">
      <parameters>
        <parameter name="#gross">0.115</parameter>
      </parameters>
    </instance>
  </instances>

  <calculation>
    <measurand>m</measurand>
  </calculation>
</simulation>
```

It is also possible to provide a character string interpreted as a formula instead of a constant value for a parameter. Then, identifiers of other instances as well as global variables can be used in the definition of a parameter. The graphical concept of the internal representation of the simulation definition and the instance of the balance is shown in Figure 4.13. We have an instance `m` of a basic model that uses three subsequent influence quantities to describe the main influence quantity for the evaluation of its formula. Figure 4.14 shows the histogram of a Monte Carlo simulation with 10^7 trials for the evaluation of this measurement scenario when weighing 115 mg.

4.3.2 Shared influence quantities

We can now continue with refining the basic model of the balance. Influence quantities are modeled in a recursive manner, so that the concept that we used for

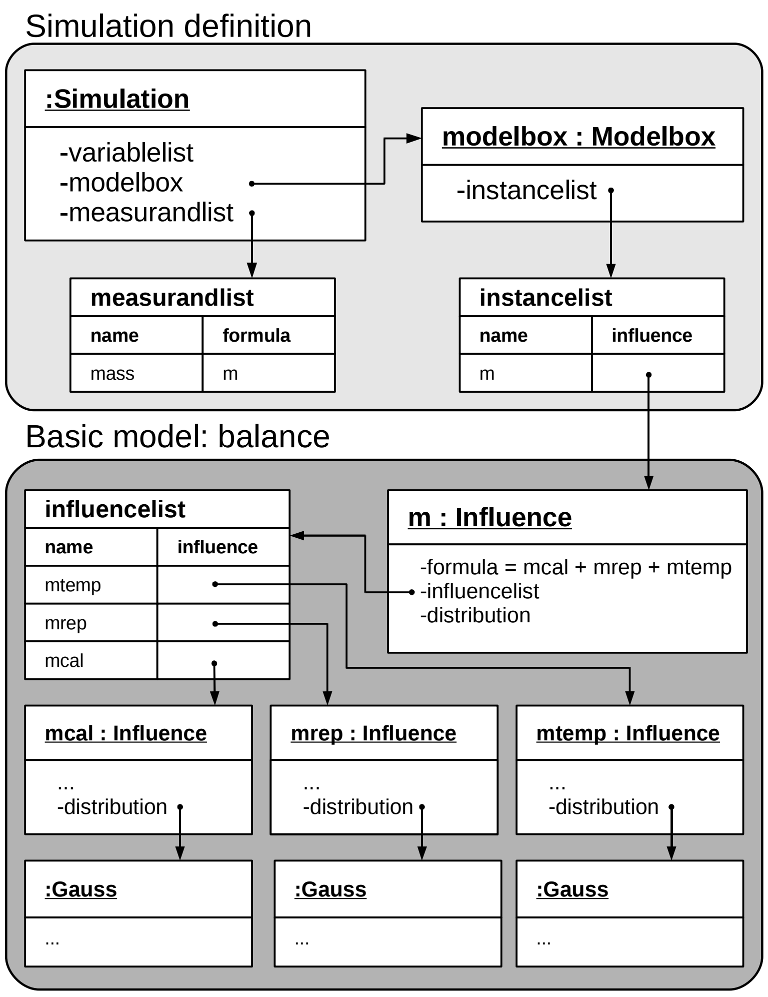


Figure 4.13: Concept of the internal representation of an instance for a model of a balance and a simple corresponding simulation definition.

the model of the balance can be applied directly to the next steps. To begin with, the influence quantity m_{cal} can be subdivided into two distinct uncertainty sources

$$m_{cal} = m_{gross} - m_{tare}. \tag{4.3.2}$$

The influence quantity m_{gross} results from calibration where metallic gauge blocks of well-known masses are used for calibration. The quantity m_{tare} describes the uncertainty of calibrating the balance to the zero point without any mass. The uncertainty source m_{rep} of the repeatability is subdivided into

$$m_{rep} = m_{rep1} + m_{rep2},$$

where m_{rep1} corresponds to an uncertainty source for repeated measurements of the same mass and m_{rep2} represents an uncertainty source for temperature differ-

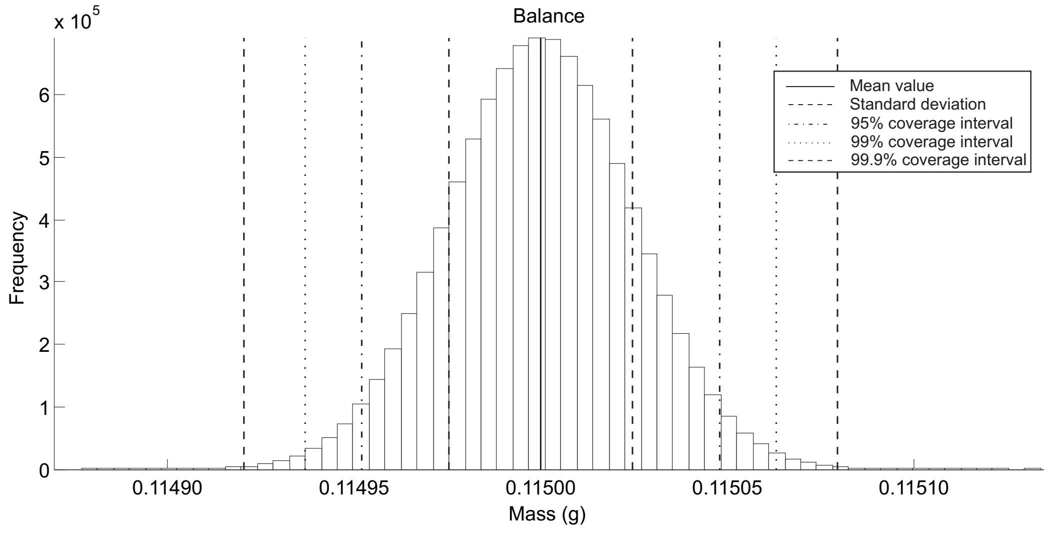


Figure 4.14: The histogram shows the result of a simulation using the basic model for the balance with 10^7 trials.

ences of calibration and sample weighing. In a next step we want to refine the influence quantity temperature of the balance to discuss a special issue. The influence quantities for tare and gross of the balance are uncertainty sources for more than one influence quantity in the same basic model of the balance, so we have to introduce an additional construct for modeling these.

Modeling measurement scenarios leads to a hierarchical structure of influence quantities as we have already seen. Often it seems that at least some of the influence quantities are not independent, but rely on other influence quantities of the measurement. One tries to avoid such causal dependencies and correlations wherever possible as the quantification and analysis of dependencies is a non-trivial task in daily work. We will see different levels of dependencies while developing measurement scenario models. A first possibility for direct dependencies is when an influence quantity is used not only at one point in an uncertainty evaluation but in different functional relationships. This is the case for the influence quantities m_{gross} and m_{tare} in a higher-resolution model of the balance. They are used for the definition of the uncertainty source of calibration as can be seen in Equation 4.3.2, but are also used for the redefinition of the influence quantity m_{temp} to consider uncertainty regarding the temperature. It uses the equation

$$m_{temp} = (m_{gross} - m_{tare}) \cdot temp_R \cdot temp_{DC},$$

where $temp_R$ describes the valid range for the temperature and $temp_{DC}$ is a temperature drift coefficient.

Figure 4.15 shows the enhanced cause-and-effect diagram with the final struc-

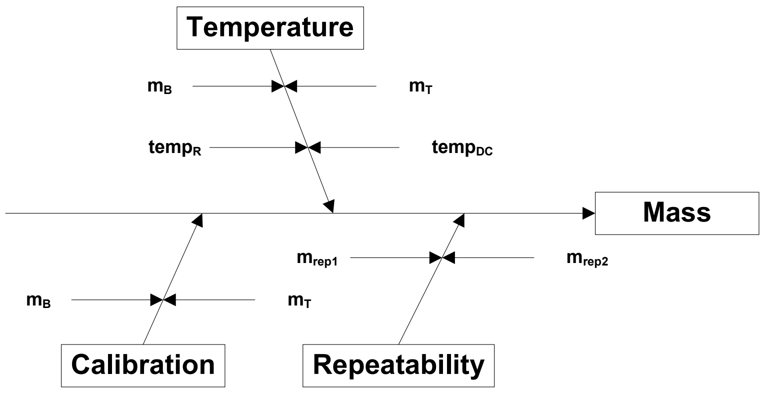


Figure 4.15: The cause-and-effect diagram for a balance with refinements for influence quantities calibration, temperature and repeatability

ture for the basic model representing the balance. The influence quantities for gross and tare appear twice, but it is not clear from the figure if only the identifiers are the same or if the influence quantities are equal. This is one shortcoming of the visualization using cause-and-effect diagrams. To overcome this problem and clarify it, unique identifiers may be assumed and enforced, and/or colors can be used in the CED.

It is not appropriate to define the influence quantities twice in the basic model of the balance, because that would lead to misunderstandings and hence, not reflect the intended dependency. To create a proper model we introduce a construct to tell the system that an influence quantity equals another. The following basic model uses this concept. We discuss the model and its behavior after the listing of the refined model of the balance.

```
<model name="balance" targetname="m">
  <influence comment="balance" name="m" unit="g">
    <formula>mcal + mrep + mtemp</formula>
    <influences>
      <!-- Definition influence calibration -->
      <influence comment="calibration" name="mcal" unit="g">
        <formula>mB - mT</formula>
        <influences>
          <influence comment="mass gross" name="mB" unit="g">
            <distribution>
              <rectangle>
                <mean parameter="#gross">55.13175</mean>
                <width>3e-05</width>
              </rectangle>
            </distribution>
          </influence>
          <influence comment="mass tare" name="mT"  unit="g">
          <distribution>
            <rectangle>
              <mean>55.01675</mean>
              <width>3e-05</width>
            </rectangle>
          </distribution>
        </influence>
      </influences>
    </influence>

    <!-- Definition influence repeatability -->
    <influence comment="repeatability" name="mrep"
        unit="g">
      <formula>mrep1 + mrep2</formula>
      <influences>
        <influence comment="repeatability of balance"
            name="mrep1" unit="g">
          <distribution>
            <gauss>
              <mu>0</mu>
              <sigma>0.5e-05</sigma>
            </gauss>
          </distribution>
        </influence>
        <influence comment="repeatability of sample"
            name="mrep2" unit="g">
          <distribution>
            <rectangle>
              <mean>0</mean>
              <width>0.7e-08</width>
            </rectangle>
          </distribution>
        </influence>
      </influences>
    </influence>

    <!-- Definition influence temperature -->
    <influence comment="temperature" name="mtemp" unit="g">
      <formula>(mB - mT) * tempR * tempDC</formula>
      <influences>
        <!-- Influences mB and mT already defined -->
        <influence comment="mass gross" equals="mB"
            unit="g"/>
        <influence comment="mass tare"  equals="mT"
            unit="g"/>

        <influence name="temperature range" name="tempR"
            unit="°C">
          <distribution>
```

```
66               <rectangle>
67                 <mean>0</mean>
68                 <width>2</width>
69               </rectangle>
70             </distribution>
71           </influence>
72           <influence comment="drift coeff." name="tempDC"
              unit="1/°C">
73             <distribution>
74               <triangle>
75                 <mean>0</mean>
76                 <width>1.5e-06</width>
77               </triangle>
78             </distribution>
79           </influence>
80         </influences>
81       </influence>
82     </influences>
83   </influence>
84 </model>
```

In the example the influence quantities gross `mB` in line 9 and tare `mT` in line 17 are defined for the calculation of the influence quantity `mcal` for calibration in line 7. But the same influence quantities are used again for the calculation of the temperature `mtemp` in line 55. The influence quantities gross `mB` and tare `mT` are said to be equal in structure and parameters to the ones defined first in the example of the balance. There is a convenient way to model this dependency in the modeling language of *MUSE* using the attribute `equals` for an element `influence`.

```
59    <influence comment="mass of gross" equals="mB"
60        unit="g"/>
61    <influence comment="mass of tare"  equals="mT"
62        unit="g"/>
```

During simulation the same influence quantities of the current basic model are used for evaluation of the measurement uncertainty in both equations and, therefore, the same model structure and parameter settings are used for calculation. Despite this equality, it is very important to note that on each call from an influence quantity new random values are drawn for this influence quantity for its current evaluation of the equation, if a Monte Carlo simulation is performed. Influence quantities sharing the same model are only set as equal in values if they appear within the same formula or if they are explicitly marked as equal in value. Otherwise, only structural equality is assumed. In the next section I will show a model for the burette device that delivers exactly the same random value for some equal influence quantities in one trial additionally to the structural equality.

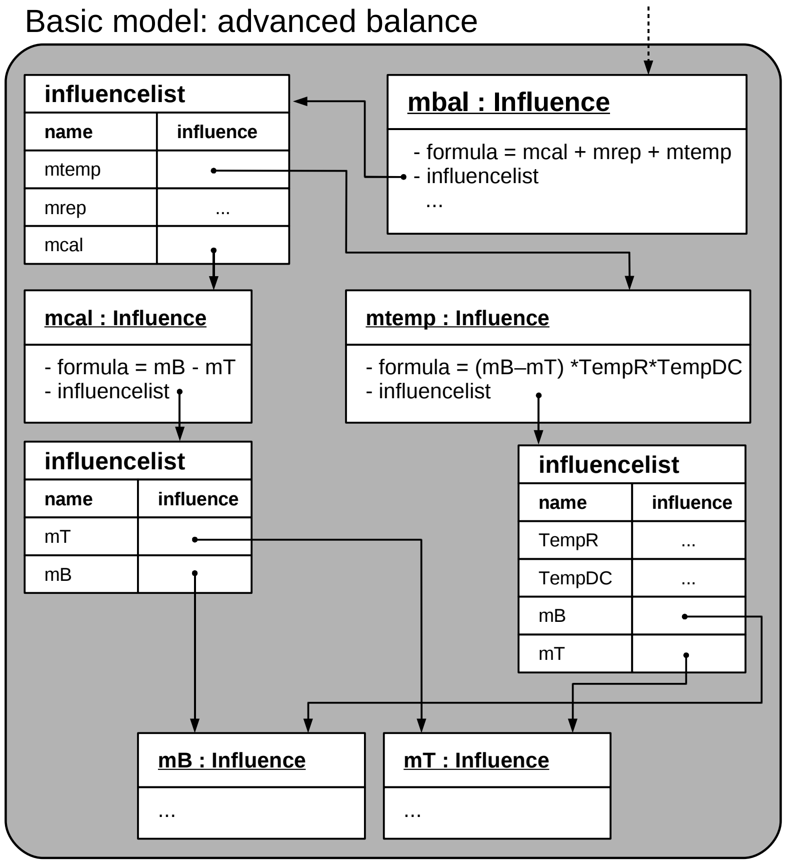


Figure 4.16: Concept of internal structure of a basic model with two influence quantities gross `mB` and tare `mT` that are used from other influence quantities

Figure 4.16 shows the internal representation of the advanced basic model for the balance excluding the simulation definition as it remains the same. There is now a direct link from the influence quantities gross `mB` and tare `mT` of the temperature to the same influence quantities of the calibration.

The histogram of a simulation with 10^7 trials and the higher-resolution basic model again weighing 115 mg is shown in Figure 4.17. The comparison of the histograms of the simple and the higher-resolution balance shows some differences in the shape of the resulting distributions. The behavior of the balance with the simple model can be properly approximated using a normal distribution, whereas the shape of the higher-resolution distribution can be better described by a triangular distribution. The uncertainty intervals are slightly larger using the simple model as can be seen in Table 4.1[6]. In this case the additional information

[6]The standard uncertainty is usually reported with 2 or 3 significant digits; coverage intervals are also reported with 2 or 3 digits considering the mean value; in this case, standard uncertainty

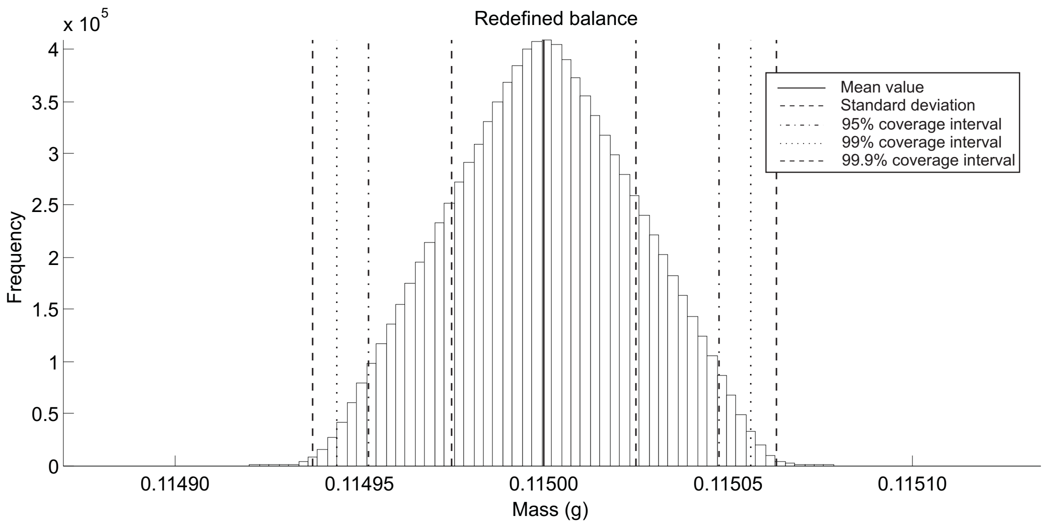


Figure 4.17: The shape of the resulting distribution of the simulation with 10^7 trials using the enhanced model for a balance is approximately triangularly shaped.

about the higher-resolution influence quantities seems to improve the knowledge about the measurement uncertainty, the quality and reliability of the result, only by a little. One reason for this similarity of the results is the selection of proper parameter settings in a first example, where both models describe the properties of the balance accurately. The more advanced model in this example allows a deeper insight into the behavior and structure of the measurement system and a far more detailed analysis, as the shapes of the two resulting distributions differ significantly. So, if more detailed information is already available – this is the case for the presented balance here – higher-resolution models can help in understanding how the measurement uncertainty changes under varying conditions. This point will be of interest at the end of the chapter when we start to compare the results of simulations with different parameter settings.

At this point, we can say that measurement equipment and other uncertainty sources can be implemented in a structured and hierarchical way using a combination of basic models and composed influence quantities. Complex functional relationships can be divided into logical parts to describe parts of the measurement individually where possible and appropriate.

4.3.3 Measurement device burette

The balance device just introduced uses a relatively simple, mainly additive measurement model. Nevertheless, we already have seen differences in the shape of

and coverage intervals are reported with 2 digits.

		Device balance	
		Simple model	**Advanced model**
Mean value		0.11500	0.11500
Standard deviation		2.5×10^{-5}	2.5×10^{-5}
Coverage interval	95 %	[0.11495,0.11505]	[0.11495,0.11505]
	99 %	[0.11494,0.11506]	[0.11494,0.11506]
	99.9 %	[0.11492,0.11508]	[0.11494,0.11506]

Table 4.1: Resulting values for simple and advanced basic model of balance device in direct comparison; simulation with 10^7 trials each; all values are given in (g).

the resulting distribution between a simple and a more refined model because of the parameters chosen for the probability distributions of the influence quantities. The next device to be modeled for the evaluation of the measurement uncertainty for a titration measurement is the burette device V_{bur}. It uses an additive model

$$V_{bur} = V_{cal} + V_{rep} + V_{temp} + V_{diff} + V_{age}$$

in a first step, too, where

- V_{cal} describes an uncertainty contribution of the calibration for the device. The information is provided by the manufacturer and should be defined using a triangular probability distribution for our example.
- V_{rep} describes the dilution behavior for repeated measurements. It is determined for example in 10 repeated measurements of 10 ml diluting. A normal distribution is estimated for modeling.
- V_{diff} describes if there is diffusion[7] between the titrant at the tip of the burette and the solution in the beaker. It is modeled as a curvilinear trapezoidal probability distribution.
- V_{age} represents an uncertainty source because of aging. It is necessary as the glass of the burette may be affected by aggressive substances over time. It uses the sub-model
$$V_{age} = f_{age} \cdot age \cdot V_{cal} \tag{4.3.3}$$
for measurement uncertainty evaluation.

[7]Earlier in the chapter I have described the principle of a classical titration measurement, where the solution of the burette, which is placed above the beaker, drips into the solution in the beaker; however, modern titration devices place the tip of the burette directly in the solution in the beaker, because it allows more exact results.

- V_{temp} describes a factor concerning the temperature difference in the laboratory between the calibration of the device and the actual usage. It uses the sub-model
$$V_{temp} = \gamma \cdot V_{cal} \cdot \delta T. \tag{4.3.4}$$
For further investigation of the influence quantity of the current temperature in the laboratory during measuring, we define a curvilinear trapezoidal distributed temperature difference from usage and calibration in a range of -2.0 °C and 7.0 °C. Supposing that the laboratory is not temperature controlled, we make the very conservative assumption that the calibration took place approximately at 20 °C and the ambient temperature is between 18 °C and 27 °C for the actual sample measurement.

A new and interesting aspect of the burette model is that the influence quantity V_{cal} for the calibration appears in two formulas of the uncertainty source for the age V_{age} in Equation 4.3.3 and the temperature V_{temp} in Equation 4.3.4, similar to the influence quantities gross m_B and tare m_T of the balance in the previous section. The difference is that for multiple usages of the device the calibration should use the same values for evaluation. We will get to this topic after showing the complete XML document of the burette device.

```
<model name="burette" targetname="vb">
  <!-- Influence quantity calibration -->
  <influence comment="calibration" name="vcal" unit="ml"
      mode="static">
    <distribution>
        <triangle>
          <mean parameter="#cal">19</mean>
          <width>3e-3</width>
        </triangle>
    </distribution>
  </influence>

  <!-- Start the definition of burette -->
  <influence comment="burette" name="vb" unit="ml">
    <formula>vcal+vrep+vtemp+vdiff+vage</formula>

    <influences>
      <!-- Influence quantity calibration -->
      <influence comment="calib." equals="vcal" unit="ml"/>

      <!-- Influence quantity repeatability -->
      <influence comment="repeatab." name="vrep" unit="ml">
        <distribution>
            <gauss>
              <mu>0</mu>
              <sigma>7e-04</sigma>
          </gauss>
        </distribution>
      </influence>

      <!-- Influence quantity temperature -->
      <influence comment="temp." name="vtemp" unit="ml">
        <formula>gamma*vcal*deltaT</formula>
        <influences>
          <!-- Definition influence quantity calibration -->
          <influence comment="cal." equals="vcal" unit="ml"/>
          <influence comment="temp.r." name="deltaT"unit="°C">
            <distribution>
                <cltrapez>
                  <lower>-2.0</lower>
                  <upper>7.0</upper>
                  <inexactness>1.0</inexactness>
                </cltrapez>
            </distribution>
          </influence>
          <influence comment="expansion coef." name="gamma"
              unit="1/°C">
            <distribution>
                <cltrapez>
                  <lower>2.40055e-04</lower>
                  <upper>2.29402e-04</upper>
                  <inexactness>1.05e-06</inexactness>
                </cltrapez>
            </distribution>
          </influence>
        </influences>
      </influence>

      <!-- Influence quantity diffusion -->
      <influence comment="diffusion" name="vdiff" unit="ml">
          <distribution>
            <cltrapez>
              <lower>-4.6875e-4</lower>
              <upper>4.6875e-4</upper>
              <inexactness>1.875e-4</inexactness>
            </cltrapez>
        </distribution>
      </influence>

      <!-- Influence quantity aging -->
      <influence comment="aging" name="vage"  mode="static"
          unit="ml">
        <formula>ca*age*vcal</formula>
        <influences>
```

```
72            <influence comment="calibration" equals="vcal"
                  unit="ml"/>
73            <influence comment="age" name="age" unit="years">
74              <distribution>
75                  <rectangle>
76                    <mean>1</mean>
77                    <width>0.3</width>
78                  </rectangle>
79              </distribution>
80            </influence>
81            <influence comment="age factor" name="ca"
82                unit="1/years" lower="0">
83              <distribution>
84                  <cltrapez>
85                    <lower>-1.0e-04</lower>
86                    <upper> 3.8e-04</upper>
87                    <inexactness>2e-05</inexactness>
88                  </cltrapez>
89              </distribution>
90            </influence>
91        </influences>
92      </influence>
93    </influences>
94  </influence>
95 </model>
```

In the listing the influence quantity `vcal` is defined in line 3 and then reused in lines 18 and 35. In this case, not only the model structure and parameter settings should be the same, but also the same random numbers should be used for an evaluation of the two referring sub-formulas during the Monte Carlo simulation. Therefore, we introduce a new attribute `mode`, which has to be set to `static` for the influence quantity `vcal` to accomplish this requirement. To explain the distinct behavior, assume a model containing only one influence quantity $X \sim N(0,1)$ with assigned standard normal distribution.

```
<model name="static" targetname="Y">
  <influence comment="Normal dist." name="X" mode="static">
    <distribution>
      <gauss>
        <mu>0</mu>
        <sigma>1</sigma>
      </gauss>
    </distribution>
  </influence>

  <influence comment="difference" name="Y">
    <formula> X1 - X2 </formula>
    <influences>
```

```
14       <influence comment="Reference 1" name="X1">
15         <formula>X</formula>
16         <influences>
17           <influence comment="Normal dist." equals="X"/>
18         </influences>
19       </influence>
20       <influence comment="Reference 2" name="X2">
21         <formula>X</formula>
22         <influences>
23           <influence comment="Normal dist." equals="X"/>
24         </influences>
25       </influence>
26     </influences>
27   </influence>
28 </model>
```

At a first sight, two equally defined quantities should be canceled out by subtraction. This is true, of course, for classical calculus with constant values. But as we are working with uncertain influence quantities and apply a Monte Carlo simulation to the evaluation of the measurement uncertainty, the result depends on the model, in this case on the modeling of the influence quantity `X`. If we subtract `X1` from `X2` in line 12, where both quantities are set equal to the quantity `X`, but without setting the attribute `mode` to static, the function to generate random values for `X` is called twice for two distinct formulas. Hence, after the first call from `X1` new values are generated in the second call for the quantity `X2` independently of the first draw. In the example the resulting distribution would be a mean value of practically zero, and a non-zero uncertainty of about $u(y) = 1.409$. The keyword `static` of the attribute `mode` annotated to the element definition of the influence quantity `X` tells the simulation system in the example that the value of the quantity has to remain the same for all calls in the current evaluation of the model. The function to draw random numbers for the quantity will then only be called once and the result will be exactly zero with a standard deviation of zero. Hence, the behavior and multiple usage of the influence quantity `vcal` of the burette device are properly modeled in the last listing.

Finally, let's have a look at the influence quantity describing aging of the burette device. The following line defines an influence quantity `ca`.

```
81    <influence comment="Age factor" name="ca"  unit="1/years"
          lower="0">
```

In principle, the probability distribution is a curvilinear trapezoidal distribution. The special issue in this case is that the age cannot become negative. Values below zero do not make any sense for the age. This is no problem for the influence quantities `age` and `cal` as their current definition and parameter settings allow only positive values. But for the quantity `ca` the range of possible values for the influ-

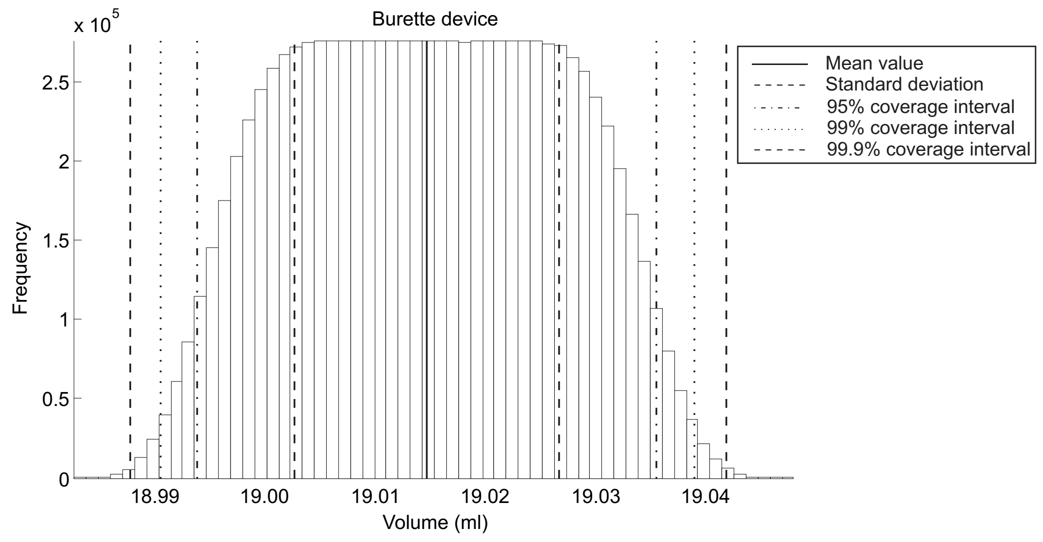


Figure 4.18: Histogram with 10^7 trials of the basic model for the burette device

ence quantities has to be restricted using the attribute `lower` with a limiting value. The same would be possible, of course, for upper limits using the attribute `upper`. For further discussion about measurements near physical limits and measurement uncertainty evaluation considering cut distributions I want to refer to Section 8.1.

Figure 4.18 shows the result of a simulation again with 10^7 trials. It seems that the resulting histogram has a broad flat top, where the flanks gradually decrease. Also interesting is the slight shift of the mean value from 19.00 ml to about 19.015 ml due to the model and parameter settings and a rather large standard uncertainty with 0.012 ml. One reason is that only an estimation based on very little information about the ambient temperature is taken into account, a rough interval of 9 °C around the temperature of calibration.

4.3.4 Enhanced temperature model for burette

We want to investigate the behavior of a burette further and refine the model for the device. The current model is sufficient for daily work in laboratories to report an estimation of the measurement uncertainty with the measurement results. Let's suppose now that the influence of the ambient temperature in the laboratory to the measurement uncertainty of the titration should be determined, because air-conditioning the laboratory seems to be a good and rather inexpensive investment to get more accurate measurement results.

Until now, the influence quantity `deltaT` describes the uncertainty source of the temperature difference from usage and production summarized in one influence quantity, where we used a conservative estimation. For our investigation we redefine this quantity and divide it into an influence quantity for the temperature during calibration and one for the temperature during usage in an actual measurement. This allows a more detailed analysis and simulations using different local ambient temperature settings in the laboratory. The model equation in line 32 is replaced by

```
32  <formula>(Tpro-Tuse)*gamma*vcal</formula>
```

and the influence quantity `deltaT` is replaced by the following code fragment that introduces two new influence quantities `Tpro` for the temperature during production and `Tuse` for the temperature at usage.

```
36  <influence comment="Temp. production" name="Tpro" unit="°C">
37    <distribution>
38      <cltrapez>
39        <lower parameter="#Tplower">20.5</lower>
40        <upper parameter="#Tpupper">21.0</upper>
41        <inexactness>0.1</inexactness>
42      </cltrapez>
43    </distribution>
44  </influence>
45  <influence comment="Temp. usage" name="Tuse" unit="°C">
46    <distribution>
47      <cltrapez>
48        <lower parameter="#Tulower">22.0</lower>
49        <upper parameter="#Tuupper">22.5</upper>
50        <inexactness>0.1</inexactness>
51      </cltrapez>
52    </distribution>
53  </influence>
```

This enhanced model takes the uncertainty of the two distinct temperatures explicitly into account and allows an independent selection of probability distributions. The resulting histogram of a simulation with 10^7 trials is shown in Figure 4.19. The example uses a difference in temperature of about 2 °C as the temperature during calibration is about 20.75 °C and during usage about 22.25 °C. This scenario is very realistic even nowadays as not all laboratories are temperature controlled. Besides, the difference of temperatures is rather moderate in this case, and the calibration of the device with a reference measurement can take place in cool morning hours, whereas the sample measurements themselves take place during the day as an example.

In this case the higher-resolution model results in an approximately normal-shaped distribution. The standard uncertainty decreases significantly as more precise information is used for the evaluation as can be seen in Table 4.2. It is interesting to note the shift of the expectation value. The shift is much more moderate, as not the whole range of possible temperatures has to be taken into

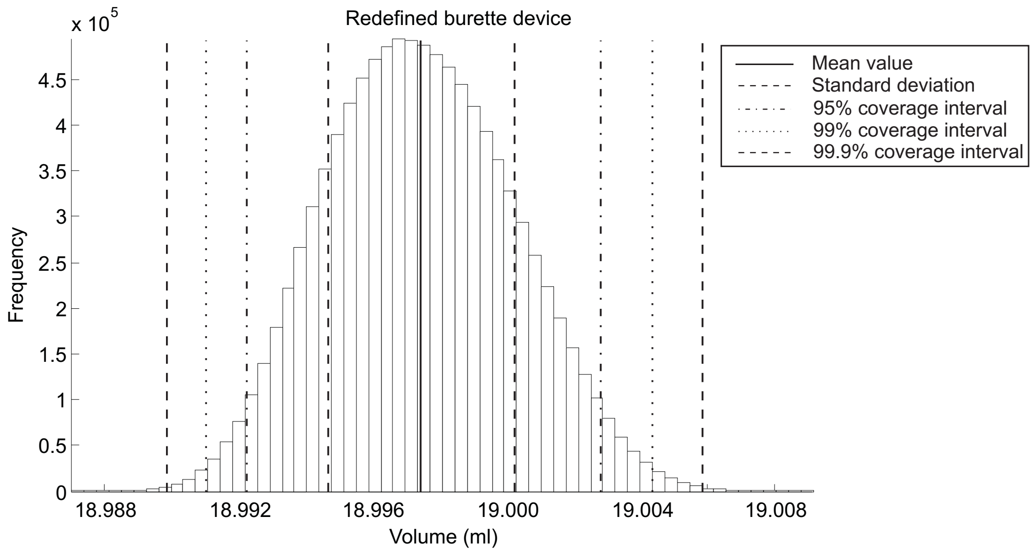


Figure 4.19: Histogram with 10^7 trials of the high-resolution burette model; ambient temperatures of usage and calibration are defined separately.

		Device burette	
		Simple model	**Advanced model**
Mean value		19.0147	18.9974
Standard deviation		1.19×10^{-2}	0.27×10^{-2}
Coverage interval	95 %	[18.9938,19.0356]	[18.9922,19.0028]
	99 %	[18.9906,19.0390]	[18.9910,19.0043]
	99.9 %	[18.9879,19.0420]	[18.9899,19.0058]

Table 4.2: Results of simple and advanced basic model for the burette device in direct comparison; simulation with 10^7 trials each; all values are given in (ml).

account. Before we can examine the behavior of the measurement system with different temperature settings, the volume of the titration measurement has to be introduced based on the burette device. Afterwards we will see if the investment in an air-conditioning system would pay off in the given scenario.

4.3.5 Volume for titration measurement

As we have now the model of the burette device, we can start to describe the model of the influence quantity V_{tit} describing the volume of titrant needed for titration. The quantity is defined as

$$V_{tit} = V_{det} + V_{bur}, \tag{4.3.5}$$

where V_{bur} is the titrimetric volume device – the burette from the last section in our case. The quantity V_{det} is an uncertainty source introduced by detection of the volume. It uses a simple model, where it is described using a normal distribution with an expectation value of zero and a standard deviation of 5×10^{-4}. This formula and the definition of the burette in the previous section allow us to introduce a new concept for modeling, nested basic models. The following code suffices for the definition of the influence quantity V_{tit}.

```
 1 <model name="vtitration" targetname="vtit">
 2   <influence name="vtit" comment="volume titration">
 3     <formula>vdet + vbur</formula>
 4     <influencelist>
 5       <influence name="vdet" comment="detection">
 6         <distribution>
 7           <gauss>
 8             <mu>0.0</mu>
 9             <sigma>5e-4</sigma>
10           </gauss>
11         </distribution>
12       </influence>
13       <influence name="vbur" model="burette"/>
14     </influencelist>
15   </influence>
16 </model>
```

In line 13 of the model document we include the model definition of the burette device. The concept of nested models requires additional syntax definitions for identifiers, as we need to be able to set parameters of the model burette from the simulation definition. Unique identifiers for parameters in arbitrary complex nested models would be a strong limitation and too complicated to handle. Therefore, the path to a specific influence quantity is used in the parameter setting, where the identifiers of nested influence quantities are separated using a point (.). Let's have a look at a short fragment[8] of code from the definition to set the calibration parameter of the burette using the volume model of the titration.

```
<simulation>
  <instances>
    <instance name="v" model="vtit">
      <parameters>
        <!-- Amount of titrant used in ml -->
        <parameter name="vbur.#cal">19</parameter>
      </parameters>
    </instance>
  </instances>
```

[8] Three points (. . .) will be used from now on to shorten equal or repetitive code fragments in the XML code examples.

```
10   ...
11 </simulation>
```

In line 6 the calibration parameter is set to a value of 19 ml using the path `vbur.#cal`, where `vbur` is the name of the influence quantity using the nested basic model of the burette and `#cal` the name of the parameter in this model definition of the burette. The concept of nested models allows very structured modeling and sharing of the same model fragments for different models. Besides, a refinement of such nested models applies immediately to all referencing model definitions without the need of updating each model individually.

4.4 Defining the equation for the measurand

Until now we have defined all devices for the evaluation of the measurement uncertainty for a titration measurement. We proceed by putting all things together using Equation 4.1.2 for the measurand pur_{sam} and obtain the complete measurement model. The corresponding simulation document follows.

```
<simulation>
  <instances>
    <!-- Devices for reference measurement -->
    <instance name="purref" model="purity"/>
    <instance name="mref" model="balance">
      <parameters>
        <parameter name="#gross">0.115</parameter>
      </parameters>
    </instance>
    <instance name="vref" model="vtitration">
      <parameters>
        <parameter name="vbur.#cal">19</parameter>
      </parameters>
    </instance>

    <!-- Devices for sample measurement -->
    <instance name="msam" model="balance">
      <parameters>
        <parameter name="#gross">0.115</parameter>
      </parameters>
    </instance>
    <instance name="vsam" model="vtitration">
      <parameters>
        <parameter name="vbur.#cal">19</parameter>
      </parameters>
    </instance>
  </instances>

```

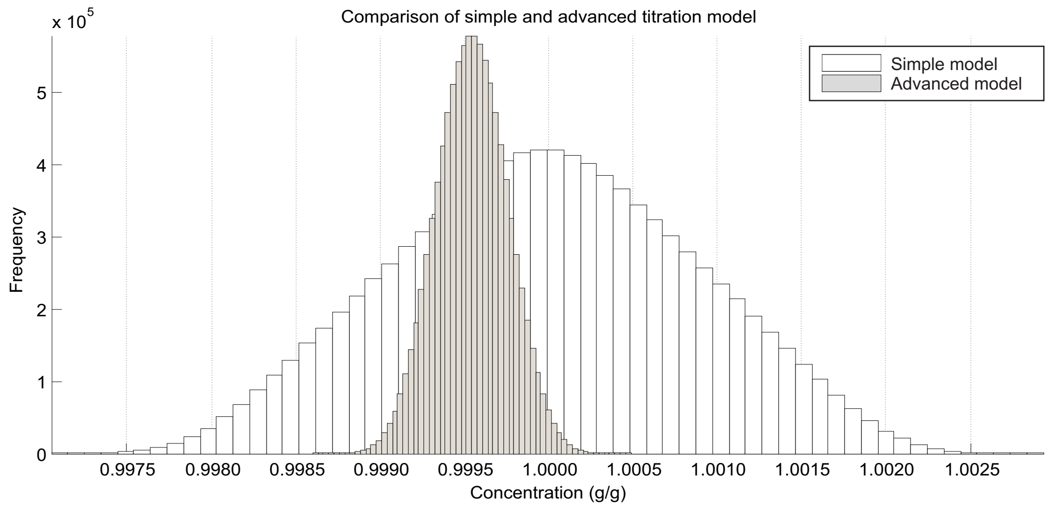


Figure 4.20: Histograms of the simulation of a single titration measurement using the simple model for the burette device and in comparison with the model for the advanced burette device; both simulations were performed with 10^7 trials.

```
  <calculation>
    <measurand>(mref*purref)/(vref*msam) * vsam</measurand>
  </calculation>
</simulation>
```

The result of the simulation is shown in Figure 4.20. We get a purity of about 0.99955 g/g with a standard uncertainty of 2.1×10^{-4} g/g with the given set of parameters for the titration measurement. For comparison also a simulation with the simple model of the burette device has been performed. The mean value then is 0.99998 g/g with a standard uncertainty of 8.9×10^{-4} g/g. This shows that even conservative estimations for environmental conditions, as in this example for the ambient temperature, result in a larger uncertainty.

We assume now that the advanced model describes the measurement properly for our interests. A single simulation cannot answer the question of wether an investment in an air-conditioning system for the laboratory would have an effect on the result and measurement uncertainty of the measurement. A direct approach to test different scenarios would be to change the ambient temperature of the model definition by hand. As this would be a tedious, repetitive and time-consuming task, the next section presents some language features that make life easier and allow testing different parameter settings for measurement scenarios automatically.

4.5 Variation of variables

This is a good moment to introduce a new type of variable in the calculation section of the simulation definition that allows analyzing measurement scenarios in more detail. It is also a first step to a what-if machine that can help experts from the field of metrology to optimize measurements with the goal of a small measurement uncertainty. Simulation of measurement scenarios with different parameter settings is a very cheap and easy way to look for important influences to the measurement if a proper model is available, whereas a lot of time and effort has to be spent to identify and quantify such influences in the laboratory. We have already seen the simple definition of a variable in the calculation section for the initial model of the purity substance of NIST. Instead of defining a single value or a probability distribution for a variable, we now want to define an interval of values, which are used as parameter settings for simulation subsequently. The so-called *variation variable* allows exactly this behavior and is suitable to test a given measurement scenario with different parameter settings. As an example we want to have a look at expected changes of the result of the titration measurement if the ambient temperature in the laboratory increases from 18 °C to 28 °C in steps of 0.5 °C. This range of temperatures is realistic when performing a titration measurement, as there still exist a lot of chemical laboratories that are not temperature-controlled.

Variation variables The first new kind of variable is only allowed in the calculation section of the simulation definition. Variables in the calculation section have global scope as already mentioned. This means they can be used everywhere for the definition of the model for the simulation, but also instances of basic models can access them for evaluation of the measurement uncertainty. The definition of a variation variable is straightforward and similar to usual *for*-loops of common programming languages, except that there is no body for the loop, because only the value of the variable – in the example `Tlab` – is updated and applied in each iteration.

```
<simulation>
  <instances>
    <!-- Devices for reference measurement -->
    <instance name="purref" model="purity"/>
    <instance name="mref" model="balance">...</instance>
    <instance name="vref" model="vtitration">
      <parameters>
        <parameter name="vbur.#cal">19</parameter>
        <!-- temperature for reference measurement is -->
        <parameter name="vbur.#Tulower">20.75</parameter>
        <parameter name="vbur.#Tuupper">21.25</parameter>
      </parameters>
    </instance>

    <!-- Devices for sample measurement -->
    <instance name="msam" model="balance">...</instance>
    <instance name="vsam" model="vtitration">
      <parameters>
        <parameter name="vbur.#cal">19</parameter>
        <!-- temperature for reference measurement was -->
        <parameter name="vbur.#Tplower">20.75</parameter>
        <parameter name="vbur.#Tpupper">21.25</parameter>
        <!-- temperature for sample measurement is --->
        <parameter name="vbur.#Tulower">Tlab-0.25</parameter>
        <parameter name="vbur.#Tuupper">Tlab+0.25</parameter>
      </parameters>
    </instance>
  </instances>

  <calculation>
    <variation name="Tlab" from="18.0" to="28.0" step="0.5"/>
    <measurand>(mref*purref)/(vref*msam) * vsam</measurand>
  </calculation>
</simulation>
```

Figure 4.21 shows the behavior of the mean value and standard deviation with 10^7 trials per simulation graphically. Table 4.3 summarizes the results, when the temperature rises from 18 °C to 28 °C. The mean value decreases as less solution is used for the determination of the concentration due to the expansion of the solution with increasing ambient temperature. Besides, the standard deviation – and hence the standard uncertainty – first decreases slightly until about 21 °C and then increases as the temperature rises again, because we assume the device to be calibrated at about 21 °C. The example also underlines once more that a more detailed view at measurements can enhance the reporting of results significantly as more precision can be achieved and more effort can be put into the significant influence quantities. The conclusion from the result of this concrete example is that an investment in an air-conditioning system would pay off as the measurement is under control in respect to the expectation value and the standard uncertainty. But another question is then how does the temperature affect measurement results during a usual working day. We will analyze this point in the next section.

Variation lists So called *variation lists* allow an explicit definition of values of interest. The following definition of a variation list simulates the varying temperature during a day in a laboratory that is not temperature-controlled. We use a step size of 2 hours for the temperature definitions. The data may result from actual measurements during a working-day, an average of approximately 21 °C over a

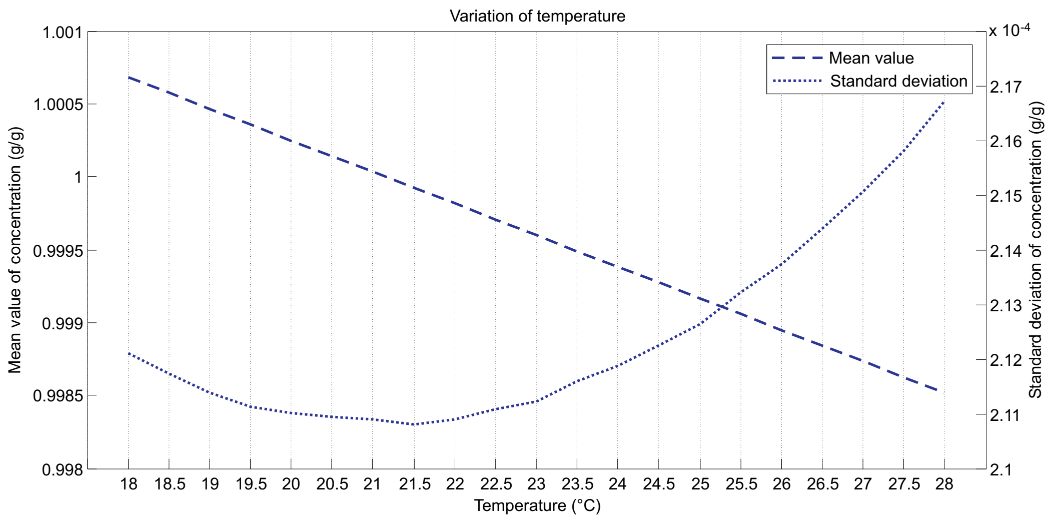


Figure 4.21: Plot of mean value and standard deviation resulting from a simulation of a titration measurement, where the ambient temperature in the laboratory increases from 18.0 °C to 28.0 °C with a step size of 0.5 °C

Temperature	Mean value	Std. uncertainty	95 % cov. interval
18 °C	1.00068	2.12×10^{-4}	[1.00026;1.00109]
19 °C	1.00046	2.11×10^{-4}	[1.00005;1.00087]
20 °C	1.00024	2.11×10^{-4}	[0.99983;1.00065]
21 °C	1.00003	2.11×10^{-4}	[0.99962;1.00044]
22 °C	0.99981	2.11×10^{-4}	[0.99940;1.00022]
23 °C	0.99959	2.11×10^{-4}	[0.99918;1.00000]
24 °C	0.99938	2.12×10^{-4}	[0.99897;0.99979]
25 °C	0.99916	2.13×10^{-4}	[0.99875;0.99957]
26 °C	0.99894	2.14×10^{-4}	[0.99853;0.99936]
27 °C	0.99873	2.15×10^{-4}	[0.99831;0.99915]
28 °C	0.99851	2.17×10^{-4}	[0.99809;0.99893]

Table 4.3: Excerpt of resulting statistical parameters of simulation with varying ambient temperature for titration measurements; values for the titration measurement are given in (g/g).

day is assumed.

```
    <variationlist name="Tlab">
      <value>20.0</value> <!--  0:00 -->
      <value>19.1</value> <!--  2:00 -->
      <value>18.6</value> <!--  4:00 -->
      <value>18.8</value> <!--  6:00 -->
      <value>19.7</value> <!--  8:00 -->
      <value>20.9</value> <!-- 10:00 -->
      <value>22.1</value> <!-- 12:00 -->
      <value>23.0</value> <!-- 14:00 -->
      <value>23.2</value> <!-- 16:00 -->
      <value>22.8</value> <!-- 18:00 -->
      <value>22.1</value> <!-- 20:00 -->
      <value>20.9</value> <!-- 22:00 -->
    </variationlist>
```

In this example the temperature has a range of 4.6 °C during the day. To analyze the difference to an air-conditioned laboratory, where the temperature is under better control, we introduce a new variable `Tc` in the calculation section and use it as coefficient for the ambient temperature definition of the volume device for the sample measurement. The instance definition changes as follows.

```
    <instance name="vsam" model="vtitration">
      <parameters>
        <parameter name="vbur.#cal">19</parameter>

        <parameter name="vbur.#Tplower">20.75</parameter>
        <parameter name="vbur.#Tpupper">21.25</parameter>
        <parameter name="vbur.#Tulower">
          20.75+Tc*(Tlab-21)
        </parameter>
        <parameter name="vbur.#Tuupper">
          21.25+Tc*(Tlab-21)
        </parameter>
      </parameters>
    </instance>
```

The next line shows the definition of the new variable `Tc` in the calculation section. We can use a variation variable for the simulation of the current situation with an ambient temperature range of 4.6 °C and two additional steps reducing the range of temperatures by the factors 0.6 to about 2.8 °C and 0.2 and to 0.9 °C for comparison.

```
   <variation name="Tc" from="1.0" to="0.0" step="-0.4"/>
```

Figure 4.22 shows the result of the simulations. As expected, better temperature control results in a more stable mean value as well as in a smaller standard uncertainty. So, the result underlines once more that an air-conditioning system would pay off in a higher precision for measurements.

From the last definition follow $3 \times 12 = 36$ simulations, as each permutation of the variation variables is used as parameter set. This is the intention in this case, but in general it results in large numbers of simulations, where only specific combinations of parameter settings may be of interest. To handle such scenarios, we introduce a last kind of variation variable.

Variation sets So called *variation sets* can be defined, where explicitly interesting combinations of parameters have to be tested without the need of simulating all possible permutations. Each set of such a construct consists of a given number of variable definitions. The next example should just give an idea of the application of this kind of variable. It defines three sets of volumes to be measured in combination with different ambient temperatures.

```
<calculation>
  <variationset>
    <set>
      <variable name="vcal">15</variable>
      <variable name="Tlab">18.0</variable>
    </set>
    <set>
      <variable name="vcal">17.5</variable>
      <variable name="Tlab">18.0</variable>
    </set>
    <set>
      <variable name="vcal">19</variable>
      <variable name="Tlab">20.0</variable>
    </set>
  </variationset>
  <measurand>(mref*purref)/(vref*msam) * vsam</measurand>
</calculation>
```

The concept of different kinds of variation variables allows an automated simulation of measurement scenarios testing arbitrary combinations of parameter settings. The definition is straightforward and hence, the system delivers all prerequisites to compare and analyze measurement models. Assuming that proper measurement models are used for simulation, this can be used as basis for further analysis and decision making, e.g., if an investment in better equipment would result in more accurate measurement results. There are more advanced techniques to search for important influence quantities that contribute to the measurement uncertainty. We will have a look at these techniques in a later chapter.

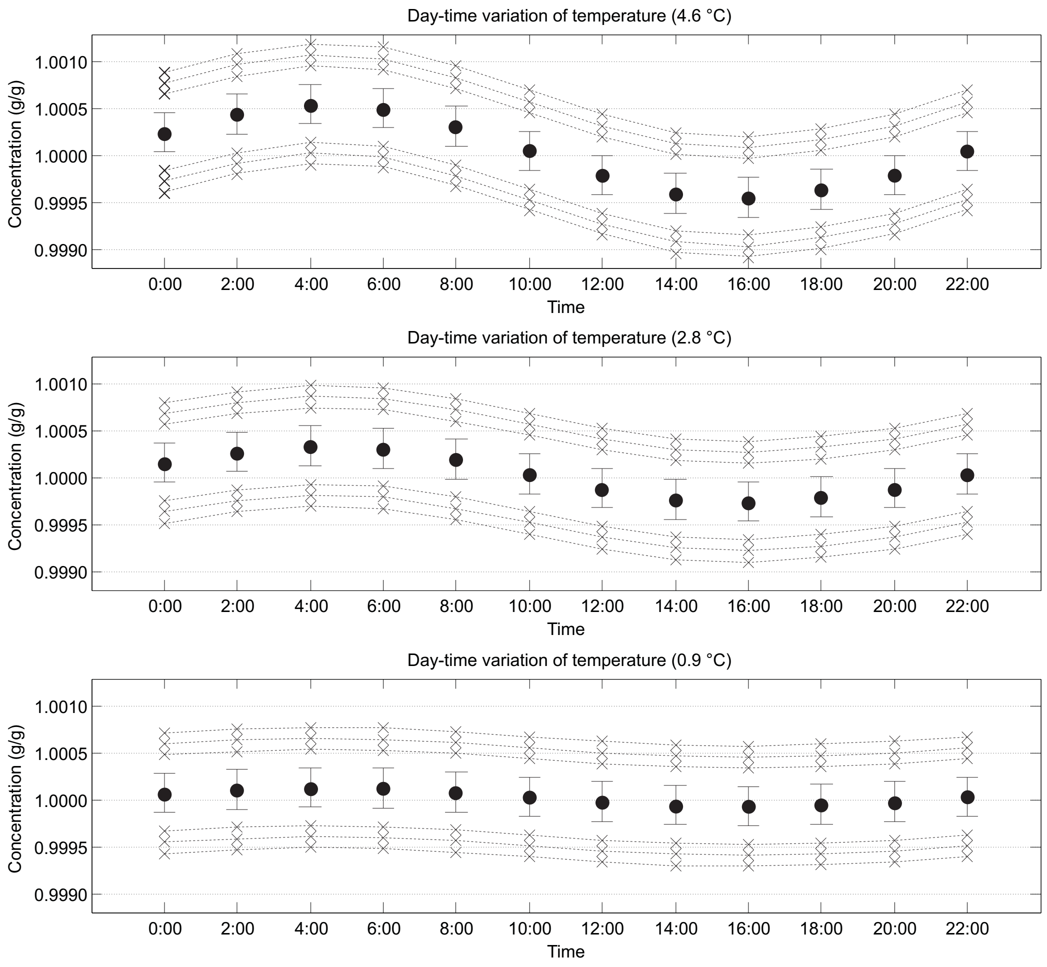


Figure 4.22: Uncertainty plots with varying temperature during a working day; three different scenarios are plotted, where the difference from minimum to maximum temperatures are 4.6 °C, 2.8 °C and in the last line 0.9 °C; all simulations were performed with 10^7 trials. The last scenario would be preferable in a laboratory as only a very small temperature fluctuations takes place during a day.

In this chapter we have modeled a concrete example of the very common titration measurement from scratch introducing and discussing the fundamental as well as some advanced concepts of the domain-specific language of *MUSE*. We have also seen that the variation of variables already allows for an interesting view on the behavior of the measurement results. It can help a lot in understanding the influence of uncertainty sources to the measurement in more detail as we have seen on the basis of a very practical example, asking if investments in an air-conditioning system would enhance measurement results significantly.

In the following chapter we will make a step backwards again and start analyzing the process of the measurement in more detail. Instead of putting the equation of the measurand together in one piece at the end of the definition of the simulation, we will have a look at how and in particular when the different influence quantities of a measurement interact. Thus, we will build a bridge to more sophisticated measurement scenarios, where not only one measurement is under investigation, but series of measurements with dependencies between influence quantities.

CHAPTER 5

Modeling measurement processes

Most of the fundamental ideas of science are essentially simple, and may, as a rule, be expressed in a language comprehensible to everyone.

— The Evolution of Physics, Albert Einstein —

In the previous chapter we have built a measurement scenario from scratch. We have modeled measurement equipment in an abstract way and then refined the models. Afterwards we have instantiated required devices with individual parameter settings and we have seen how the models are mapped to the internal structure of the simulation system for an efficient evaluation. Finally, the measurement model has been put together to evaluate an approximation for the probability distribution of the measurand. This is the most direct way for simple measurement scenarios with a single measurand. For more complex measurement scenarios, where, e.g., measurement equipment is shared between more than one measurement, the definition of the measurement model currently is not flexible enough. Another example, where the approach may not suffice, is series of measurements. There not only one sample measurement relates to a reference measurement, but a whole set of sample measurements. Another point concerning series of measurements is that typically not a single reference measurement is performed to monitor environmental conditions and to validate the results for the sample measurements, but a defined number of reference measurements in (usually) periodic intervals.

This prevents errors and helps to detect drifts and trends in sample measurements. Thus, the approach we follow in this chapter is to split the measurement model once more – as we have now more than one measurement – into logical parts. We attach importance on preserving and explicitly presenting the structure and order of actions performed during a measurement procedure. This parts can then be put together to calculate the result and according measurement uncertainty for the whole measurement scenario, the reference and the sample measurements of sequences of measurements as we will see in the next chapter.

The approach of separating a measurement into distinct parts, which can be modeled individually, is introduced in a similar form in [43], where the measurement is depicted as a measurement chain. The output of one element in the chain is used as input for the next element. The article does not go very much into detail and assumes that the elements of the measurement chain are independent. The GUM uncertainty framework is used for the evaluation of the measurement model. The thesis [70] introduces a more sophisticated system for modeling measurements and an approach for storing parts, mainly measurement equipment, of models similar to basic models as presented in the last chapter. It also describes a modular construction of the measurement model with explicitly defined models for specific problems from the field of physics. The author describes the concepts of the modeling process, but an explanation of how models and parameter settings are stored in detail is missing. Nevertheless, I agree fully with this approach, but we will go a step further and enhance it with a more advanced encapsulation technique making a distinction when modeling measurement equipment using basic models or modeling actions using processes. We work with individual basic models in separate documents, where each one describes the model of measurement equipment. An explicit, well-defined domain-specific language is necessary in practice; the reason is that the system does not rely on specific database engines in the background, models and simulation projects can be exchanged from and to laboratories and persons without any problems and are readable without the need for specific software applications interpreting the data from a database. The language definition also helps in adapting individually tailored graphical user interfaces for special purposes and different scientific fields.

After we have defined the rather static structures of measurement items and equipment with basic models, the next step is to model the interaction of these items and uncertainty sources for more complex measurement scenarios. We stay with the example of the titration measurement from analytical chemistry and enhance it in the following thereby showing that our approach is very close to real-world measurement scenarios and that we meet the requirements of experts from metrology in laboratories. Moreover, with the help of a simple example of diluting a solution for a reference measurement twice, I will explain the concept of static model parts in more details and investigate the consequences. Finally, the concepts of the modeling language introduced so far are summarized with special focus on the scope of variables, processes, instances, and influence quantities in a measurement model.

5.1 Introducing processes

We continue modeling a titration measurement and start again with Equation 4.1.2 for the purity pur_{sam} of the sample of a titration measurement, but now we separate the influence quantities of the measurement models of the reference measurement and the sample measurement and extract the logical order

$$pur_{sam} = \frac{m_{ref} \cdot pur_{ref}}{V_{Tit,ref} \cdot m_{sam}} \cdot V_{Tit,sam} = \underbrace{\boxed{\frac{m_{ref} \cdot pur_{ref}}{V_{Tit,ref}}}}_{\text{reference}} \cdot \underbrace{\boxed{\frac{V_{Tit,sam}}{m_{sam}}}}_{\text{sample}}.$$

To map this structure to our simulation definition, we introduce so-called *processes* in the modeling language. This allows us to define individual actions that take place during a measurement, or to simply divide logical parts into separate process definitions. To clarify the concept and to show the advantages, let us assume that we have two distinct samples to be measured, both referring to the same reference measurement. To model this scenario we introduce one process describing the reference measurement and two separate processes, one for each sample measurement, as shown in Figure 5.1. Processes can access instances of basic models as well as other processes for evaluation of the measurement model. Now we can redefine the document containing the simulation definition from Section 4.4 and depict the measurement model for the measurand. We omit the parameter settings for the instances of the devices in the following code fragment to save some space as they remain the same as before. We adapt the section of the instances adjusting the identifiers of the instances for the first sample measurement and introducing corresponding instances for a second sample measurement.

```
<simulation>
  <instances>
    <!-- Devices for reference measurement -->
    <instance name="purref" model="purity"/>
    <instance name="mref" model="balance">...</instance>
    <instance name="vref" model="vtitration">...</instance>
    <!-- Devices for first sample measurement -->
    <instance name="msam1" model="balance">...</instance>
    <instance name="vsam1" model="vtitration">...</instance>
    <!-- Devices for second sample measurement -->
    <instance name="msam2" model="balance">...</instance>
    <instance name="vsam2" model="vtitration">...</instance>
```

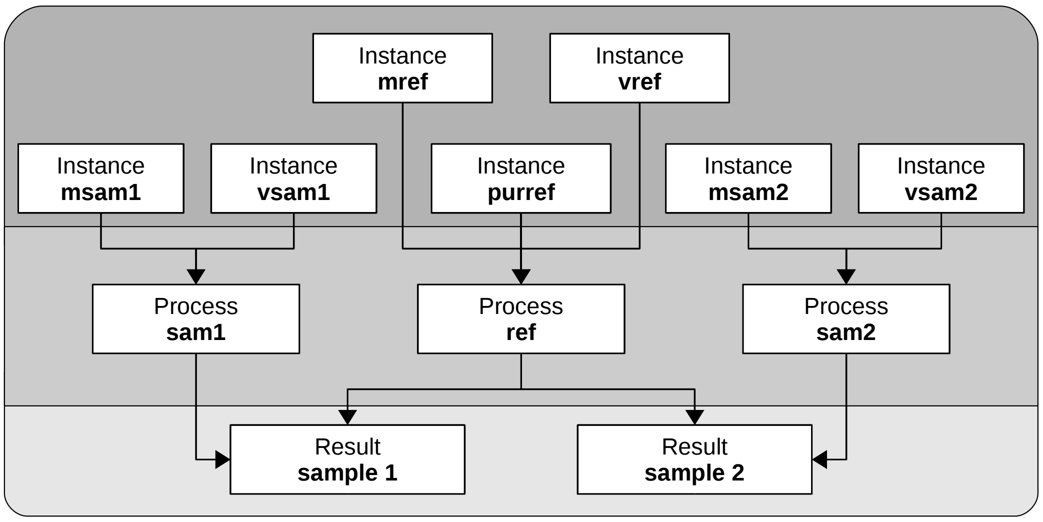


Figure 5.1: Instances of basic models are used in the example to evaluate the measurement models of the processes `sam1`, `sam2` and `ref`; the processes are then used to define the output quantity or – as in this case – output quantities.

```
  </instances>

  <processes>
    <!-- One process for reference measurement -->
    <process name="ref">
      <formula>mref*purref/vref</formula>
    </process>

    <!-- Process for first sample measurement -->
    <process name="sam1">
      <formula>ref * vsam1/msam1</formula>
    </process>

    <!-- Process for second sample measurement -->
    <process name="sam2">
      <formula>ref * vsam2/msam2</formula>
    </process>
  </processes>

  <calculation>
    <measurand>
      <formula name="sample 1"> sam1 </formula>
      <formula name="sample 2"> sam2 </formula>
    </measurand>
  </calculation>
</simulation>
```

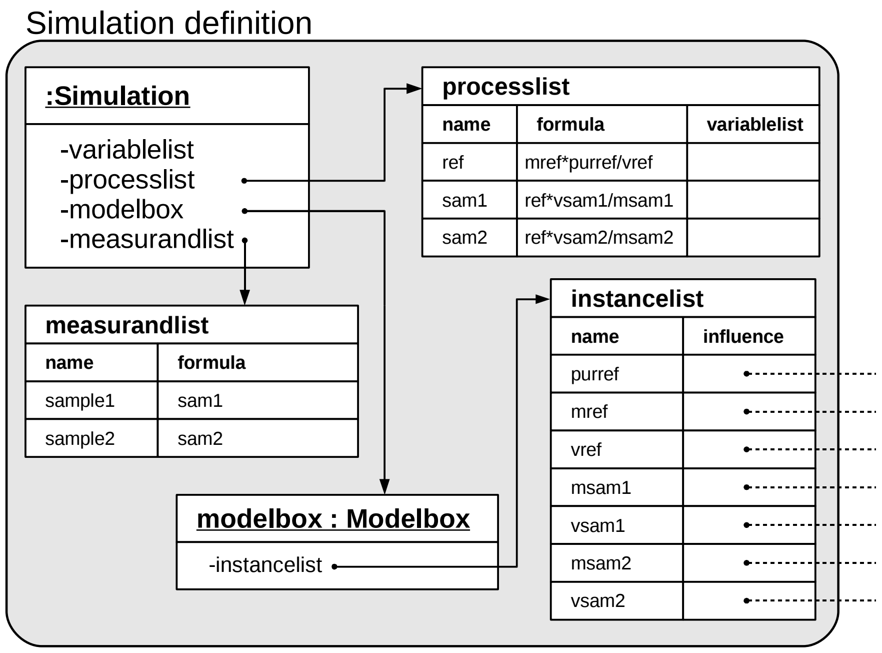


Figure 5.2: Concept of internal structure with new list for processes; the list of measurands contains two formulas in the example.

The only new part is the section `processes` between the instances and calculation section. There we define three independent processes: `ref` for the reference measurement, `sam1` for the first sample measurement and `sam2` for a second sample measurement. Processes contain at least an attribute `name` to define an identifier and an element `formula`.

Furthermore, the calculation section is extended using elements `formula` to define more than one output quantity at once in the section of the measurand. Using this construct, the simulation of more than one measurand can be performed in parallel. The equations for the measurands keep readable and manageable as more complex parts are summarized in individual process definitions. In the example the process `ref` for the reference measurement is used in the two processes `sam1` and `sam2` that define the sample measurements. The big advantage of the concept of processes is the logical and intuitive separation of parts of the measurement model; in this case dividing the measurement scenario into a reference and two sample measurements. Processes also prevent redundant definitions of parts of the model that appear in more than one formula. We will also see in the next chapter that the modeling of series of measurements is now very straightforward with the help of processes.

Figure 5.2 shows the internal representation and structure of the system. The list of the equations of the measurands now contains two distinct entries for evaluation. Additionally, the figure explains how processes are handled. A corresponding list of processes contains for each process its name, a formula for evaluation, and a list of local variables. We will have a closer look at the list of variables after another important issue regarding static parts of the measurement model.

5.2 Static processes, instances and influences

In the last example we had two sample measurements that refer to the same reference measurement. Therefore, we would expect that for the simulation of the reference measurement the same values are used in the evaluation for both sample measurements. That would reduce the calculation effort considerably. As by default new random values are drawn for each usage of an influence quantity or process in distinct formulas during evaluation, this behavior has to be marked explicitly. To avoid the process `ref` to re-evaluate when called the second time, it can be set static using the XML-attribute `mode`. This is the same concept as we have seen in the last chapter for influence quantities in basic models. Values for the quantity are sampled only once in the beginning of the current trial and the quantity – in this case a process – keeps its value for the evaluation without re-sampling.

```
19    <process name="ref" mode="static">
```

With this small change – setting the attribute `mode` to `static` – in line 19 we have reduced the computational effort significantly by modeling the scenario in an appropriate and correct way, but without changing the outcome for the two distinct measurands. The concept of static parts of the measurements can be applied to processes, influence quantities of basic models and whole instances of basic models.

Besides computational reasons, there exit situations, where the behavior of static parts of the model have an essential influence on the result of the simulation. We want to investigate this fact using a concrete example [83] that is strongly related to titration measurements. Suppose a reference solution of some substance has to be established with a specific concentration. The first thing to do is – as in previous examples – to take some amount of the reference substance on a balance and afterwards, put it into a flask and fill it with an appropriate solvent. This will result in a so-called stock solution, which is used for further diluted solutions. We have the equation for the concentration c_{stock} in (g/ml) for the stock solution given as

$$c_{stock} = \frac{m \cdot pur}{V_{f,stock}},$$

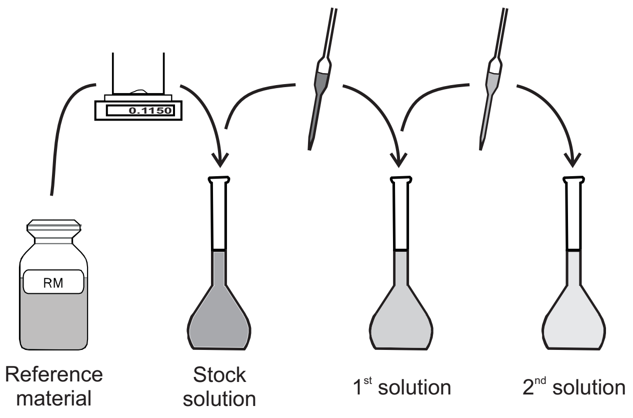


Figure 5.3: Concept of preparing a solution for a reference measurement with two dilution steps

where the mass m in (g), the purity substance pur in (g/g) and the volume device $V_{f,s}$ – in this case a flask – in (ml) are used for the evaluation. We use the stock solution and reduce the concentration in a second diluting step, taking a specific amount of the solution with a pipette V_{p1} in (ml) putting it in a second flask $V_{f,1}$ again in (ml) with solvent and get for the concentration c_1 the equation

$$c_1 = c_{stock} \cdot \frac{V_{p,1}}{V_{f,1}}.$$

In a third dilution step we repeat the reduction of the concentration once again using another, distinct pipette $V_{p,2}$ in (ml) and another, also distinct, flask $V_{f,2}$ in (ml). This leads to equation

$$c_2 = c_1 \cdot \frac{V_{p,2}}{V_{f,2}}$$

for a final concentration c_2. Now we put the equation for the measurand together, and end up with a concentration

$$c = \boxed{\frac{m \cdot pur}{V_{f,stock}}} \cdot \boxed{\frac{V_{p,1}}{V_{f,1}}} \cdot \boxed{\frac{V_{p,2}}{V_{f,2}}}$$

Figure 5.3 shows the concept of the measurement graphically[1]. For the sake of simplicity we use normal distributions for the measurement uncertainty evaluation in this case. So, let

- mass m be $N(0.1, 1 \times 10^{-4})$-distributed,

[1]I want to thank the team of the UncertaintyManager® [26] and especially Roman Hedinger for providing the pictures.

- purity *pur* be $N(0.99, 8 \times 10^{-3})$-distributed,
- flasks $V_{f,stock}$, $V_{f,1}$, $V_{f,2}$ be $N(100, 0.15)$-distributed, and
- pipettes $V_{p,1}$ and $V_{p,2}$ be $N(1.0, 6.1 \times 10^{-2})$-distributed.

The following code fragment introduces three processes that are used in the measurement model for the evaluation of the equation of the measurand. I omit here the definition of the trivial basic models and explicit instantiation of the basic models for devices as the definition is straightforward and should be obvious from earlier examples.

```
...
<processes>
  <!-- Process definition for stock solution -->
  <process name="cstock">
    <formula>(m*pur)/Vfs</formula>
  </process>

  <!-- Process definition for 1st solution -->
  <process name="c1">
    <formula>cstock * Vp1/Vf1</formula>
  </process>

  <!-- Process definition for 2nd solution -->
  <process name="c2">
    <formula>c1 * Vp2/Vf2</formula>
  </process>
</processes>

<calculation>
  <measurand>
    <formula name="c"> c2 </formula>
  </measurand>
</calculation>
```

Our question of interest in this case is, what happens if we use the very same pipette in the two processes `c1` and `c2` instead of two distinct pipettes `Vp1` and `Vp2`? We want to investigate if there is a significant difference in the result if we change the model accordingly. Hence, in a second simulation we use just one pipette `Vp` and we set the mode of the corresponding instance to static telling the system to use the same random values for the device in one trial.

```
<instances>
  ...
  <instance name="Vp" model="pipette" mode="static"/>
  ...
</instances>
```

Number of pipettes	**Mean value**	**Std. uncertainty**	**95 % cov. interval**
1	9.9×10^{-8}	1.2×10^{-8}	$[7.61\times10^{-8};12.3\times10^{-8}]$
2	9.9×10^{-8}	0.9×10^{-8}	$[8.24\times10^{-8};11.5\times10^{-8}]$

Table 5.1: Resulting values of the two distinct measurements preparing solution with two dilution steps; all values are given in (g/ml).

```

<processes>
  ...
  <!-- Process definition for 1st solution -->
  <process name="c1">
    <formula>cstock * Vp/Vf1</formula>
  </process>

  <!-- Process definition for 2nd solution -->
  <process name="c2">
    <formula>c1 * Vp/Vf2</formula>
  </process>
</processes>
...
```

The result of the two distinct simulations, each performed with 10^7 trials, indicates that there is indeed a difference. Figure 5.4 shows the two histograms in one plot; Table 5.1 gives the mean values, the standard deviation and the 95 % coverage interval resulting from the simulation. The conclusion is that using distinct pipettes results in this case in a significantly smaller standard uncertainty. The result shows that a measurement with two distinct pipettes cancels out calibration effects of distinct pipettes for both processes of the preparation of solution, whereas using only one pipette multiplies the contribution to the uncertainty. This simple example already shows that one has to be very careful when modeling measurements for uncertainty evaluation. Even the simple fact that a measurement device is used more than once can change the result of a measurement uncertainty evaluation considerably. In our scenario we set an instance of a basic model to static. In higher-resolution models it has also to be investigated if only parts of basic models describing measurement equipment have to be defined as static for evaluation.

The classical GUM document and its first supplement concentrate on the measurement uncertainty evaluation for a single measurement. This is possible in the presented simulation system applying the introduced concepts of basic models and processes. In the next chapter we will see an advanced and more sophisticated approach, where not only a single measurement is treated, but sequences

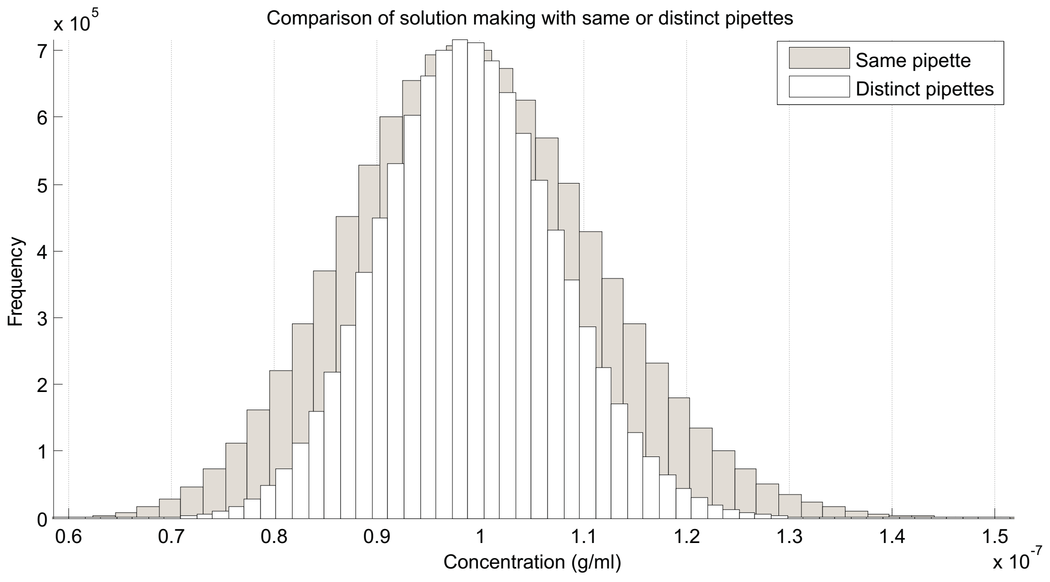


Figure 5.4: Comparison of resulting histograms using one or two pipettes in preparation of solution for measurement

of measurements. Already the results of the simple example here show that the dependencies of influence quantities to the later repeated measurements have to be well considered and modeled properly in such advanced real-life measurement scenarios.

5.3 Processes and variables

We have seen that a process encapsulates specific actions or logical parts of a measurement model. To complete the picture, there is a need for defining only process relevant variables inside a process section to realize individual settings. Hence, it is necessary that each process has its own scope. This allows using different settings, e.g., environmental conditions like ambient temperature, air pressure, or humidity, for the evaluation of distinct processes as the measurements are not performed at the same time in real life, but sequentially. The next example takes a varying ambient temperature during the distinct reference and sample measurements into account, where each measurement is defined by an individual process as before.

```
  <processes>
    <!-- One process for reference measurement -->
    <process name="ref">
      <variable name="Tlab" unit="°C">18.8</variable>
      <formula>mref*purref/vref</formula>
    </process>

    <!-- Process for first sample measurement -->
    <process name="sam1">
      <variable name="Tlab" unit="°C">20.3</variable>
      <formula>ref * vsam1/msam1</formula>
    </process>

    <!-- Process for second sample measurement -->
    <process name="sam2">
      <variable name="Tlab" unit="°C">20.5</variable>
      <formula>ref * vsam2/msam2</formula>
    </process>
  </processes>
```

The current variable `Tlab` will be used for the evaluation of the formula of each process; it will be propagated when evaluating the measurement models of used processes and instances. In this case the scope of process specific variables is of vital interest as the same name for the ambient temperature is used for the reference process as well as for the sample processes. This aspect will be dealt with in the next section; to anticipate it, local variables have a higher priority in general. Hence, the given model properly describes the measurement scenario.

5.4 Scope for measurement models

We now have a very powerful simulation system with a properly defined modeling language to model measurement scenarios with more than one output quantity for the evaluation of the measurement uncertainty. It allows structuring and encapsulating the individual measurements and parts of measurements, each with its own scope and parameter settings. Summarized, the constructs for modeling are

- basic models to define measurement items, like measurement equipment, devices, or substances, in a hierarchical way,
- instances of basic models as realizations of items with individual parameter settings,
- processes that describe individual logical parts of the measurement scenario, and last but not least
- the definition of the measurement model for one or more output quantities.

Figure 5.5 gives an overview of the modeling concept of *MUSE* explaining the single parts once more in a few words each. Let's have a look now at the priority

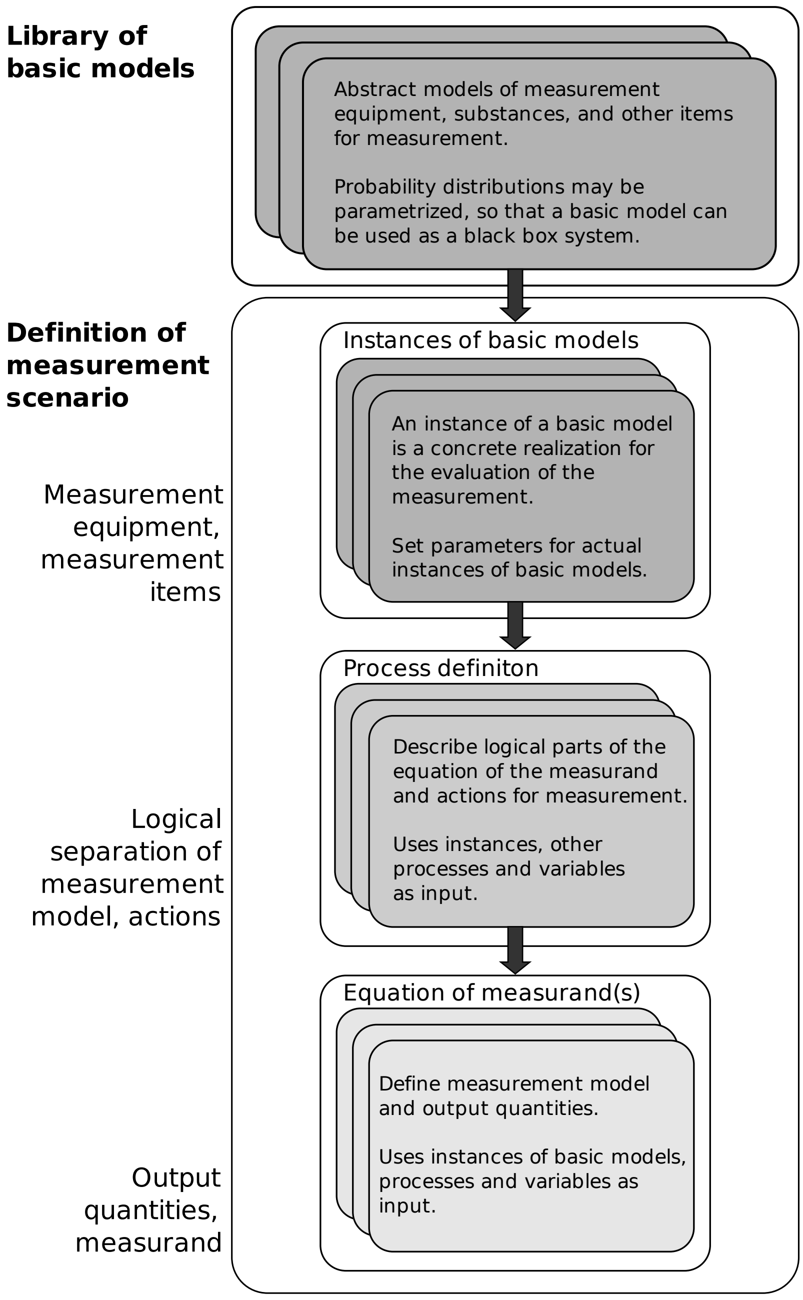

Library of basic models
Abstract models of measurement equipment, substances, and other items for measurement.
Probability distributions may be parametrized, so that a basic model can be used as a black box system.
Definition of measurement scenario
Measurement equipment, measurement items
Instances of basic models
An instance of a basic model is a concrete realization for the evaluation of the measurement.
Set parameters for actual instances of basic models.
Logical separation of measurement model, actions
Process definiton
Describe logical parts of the equation of the measurand and actions for measurement.
Uses instances, other processes and variables as input.
Output quantities, measurand
Equation of measurand(s)
Define measurement model and output quantities.
Uses instances of basic models, processes and variables as input.

Figure 5.5: Hierarchical view of the modeling concept explained graphically from the abstraction of measurement items and equipment building a library of basic models, in the next step the instances of basic models, the logical splitting into individual processes and finally, the description of the equation, or in general the equations of the measurand, respectively measurands.

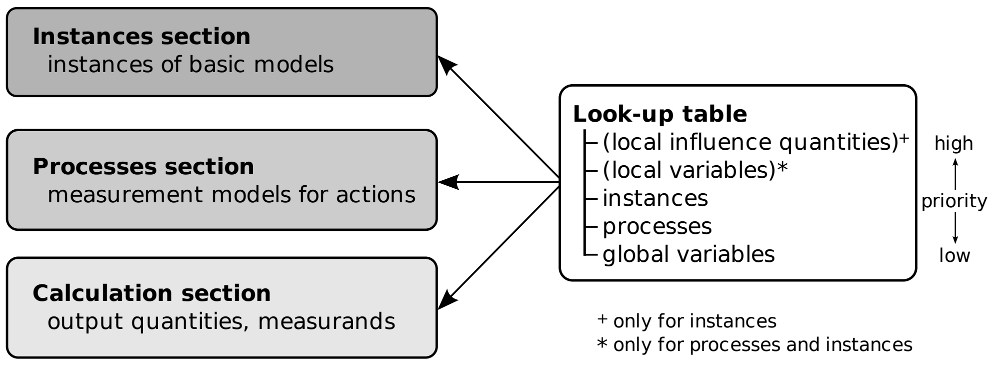


Figure 5.6: There is one global look-up table combining all the local look-up tables. It handles requests to find influence quantities from the lists of identifiers.

of the parts for evaluation, say in which order the lists of identifiers are processed during simulation. All lists containing identifiers are combined to a look-up table internally. In general, we recommend using unique identifiers in the simulation definition as it reduces the risk of misunderstandings and allows a faster adaption of existing measurement scenarios. But for large systems, which may also grow iteratively, this may not be appropriate or possible in any case. We have already seen in the previous section that it might be more comfortable and intuitive to use the same identifiers in distinct parts of the model definition. However, it is important to know in any case, which identifiers have higher priorities than others. Figure 5.6 shows the internal order of processing. For the evaluation of measurement models in the simulation definition instances have highest priority, followed by processes and finally, global variables from the calculation section. Local variables are preferred against the global look-up tables in the definition of processes. For the evaluation of formulas of influence quantities in the model of an instance, first the list of sub-influences is processed, followed by the list of local variables of calling processes. Only if an identifier is not found in these lists, the usual order for look-up follows. As this issue is very central in performing simulations, the system includes in its validation core explicitly defined scenarios considering and checking this priorities.

The next chapter will continue with series of measurements that use arbitrarily many sample and reference measurements. We will look at measurement scenarios with many sample measurements and alternating reference measurements to control and monitor measurements over time. Only lately interest has grown in similar approaches focusing on repeated measurements and considering variations over time due to the new possibilities provided by the usage of the Monte Carlo method to calculate measurement uncertainties. It allows to take dependencies into account in a very direct and intuitive way.

CHAPTER 6

Series of measurements

An uncertainty statement may be inappropriate if it relates to a measurement result that cannot be reproduced over time. A customer is entitled to know the uncertainties associated with the measurement result, regardless of the day or time of year when the measurement was made.

— ISO/TS 21749, 2005, Page 7 —

With basic models and processes we have introduced powerful concepts of the modeling language of *MUSE* that allow defining parts of measurements individually within their own context. Furthermore, we have seen how more than one equation of the measurand can be defined for multiple outcomes of a measurement uncertainty evaluation and direct comparison of results at once. In this chapter I will introduce additional features of the language and the simulation framework including a very compact definition of *series of measurements*, where various sample measurements refer to the same reference measurement or even to more than one reference measurement. It has been one of the main goals of the project to keep the definition of such measurement scenarios as easy and manageable as possible and to reduce redundant definitions of measurement equipment and processes to a minimum. ISO 21749:2005 [2] gives a detailed explanation of how the standard deviation can be evaluated for repeated measurements and nested experiments. The focus of the document lies more on the quantification

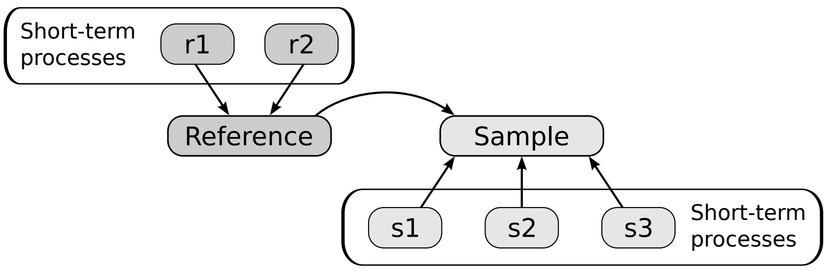


Figure 6.1: The model for a reference measurement can consist of short-term processes; in this case two processes `r1` and `r2` are used. The same holds for repeated sample measurements using the same sample. We use three sample measurements `s1`, `s2`, and `s3` as well as reference measurements for the evaluation of one sample.

of Type A influence quantities using repeated observations, where GUF is used for the evaluation of the measurement uncertainty. I will use the concepts of the ISO document to describe scenarios for measurement series. The document distinguishes three levels of time-dependent fluctuations in Section 5.2:

- short-term fluctuations (repeatability or instrument precision)
- intermediate fluctuations (equipment-to-equipment or operator-to-operator or day-to-day, known as intermediate-precision)
- long-term fluctuations (run-to-run or stability [which may not be a concern for all processes] or intermediate precision)

Short-term fluctuations can be handled using repeated reference measurements over a period of time. It is also possible that a sample is under investigation in more than one measurement. These two cases can be described in the modeling language of *MUSE* using the previously introduced processes as Figure 6.1 details.

To clarify the concept of series of measurements and the different types of fluctuations, I want to explain the issue using a real-life application staying with the titration measurement. Let us take a look at the very practical example of the production of vitamin C effervescent tablets. We are interested in the amount of ascorbine acid in the tablets. We take some of the tablets from production as random samples. The tablets are cut in half and used for different titration measurements to reduce the variability and to check the amount of acid per tablet. For calibration of the titration device a reference material like benzoic acid from the last chapter is used. Reference measurements may also be repeated in a small time

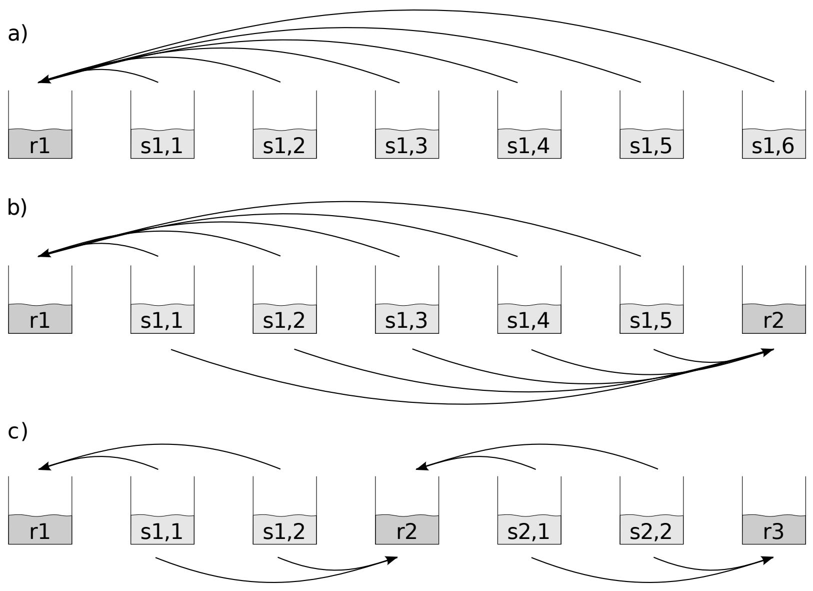


Figure 6.2: Concept of three scenarios with series of measurements, where reference measurements (r) and sample measurements (s) alternate; (a) one reference measurement, six sample measurements; (b) two reference and five sample measurements; (c) three reference and 4 sample measurements

frame to reduce short-term fluctuations. This can directly be modeled using processes as we have seen. In the end, a sequence of measurements is defined, starting with one or more reference measurements followed by a number of sample measurements. To keep the measurements under control over larger time frames, reference measurements have to be repeated in periodic intervals like days, weeks, or even months. This is the intermediate or long-term fluctuation. It is not unusual in everyday work to reduce the number of reference measurements to an absolute minimum thereby keeping the effort and costs as low as possible.

An interesting aspect in measurements with repeated reference measurements is how the sample measurements should refer to the reference measurements. There exist different possibilities; Figure 6.2 shows the concept of three different scenarios for series of titration measurements. In part (a) six sample measurements take place that refer to a single reference measurement that was performed in the beginning; in part (b) the last sample measurement is replaced by a reference measurement to be sure that the environmental conditions are under control and the equipment still works properly at the end of the measurement. In the first scenario this may also be the case using a so called check standard, which is not used for the calculation of the measurement result of the sample. We assume that the check standard measurements do not contribute to the measurement uncertainty in our current scenario and omit them. They should merely be used to control the environmental conditions and the proper working of the measurement equipment beside the actual measurements. Finally, in part (c) four sample and three reference measurements take place, the third sample measurement was again replaced by a reference measurement. The last scenario allows a more detailed investigation of environmental conditions and uncertainty sources. Of course efficiency is reduced – assuming that the system behaves as required for proper measurements – as now only four sample measurements are performed against six in the first scenario. So, we are looking for an efficient trade-off of controlled sample measurements with respect to a stable mean value and a small standard uncertainty to be able to provide accurate measurement results and a minimal number of reference measurements to keep time and costs as low as possible. If we assume that the three measurement scenarios are performed in the same time frame with equal environmental conditions, different questions arise:

- Which scenario is optimal in the sense that the measurement is under control and stable according to the mean value?
- Which scenario has to be preferred for an acceptable standard uncertainty for sample measurements?
- How many reference measurements are necessary for a reasonable result?
- How should the sequence of measurements look like, in which order should sample and reference measurements be performed?

Answering these questions helps in decision making and in setting up concrete measurement scenarios. Of course properly defined measurement models that reflect the outcome of measurements in real-life are necessary and we will and have to take them for granted in the following.

We will concentrate on the intermediate and the long-term fluctuations, where measurements are repeated over days or over runs, maybe separated over months. To start with, I will first introduce the language constructs to model series with a predefined order of measurements in *MUSE* and show different methods how sample measurements can refer to the reference measurements. The presented methods are applied in practice for analyzing measurement data as the example of production of Vitamin C effervescent tablets points out. Afterwards, we will see some approaches to the analysis of such scenarios and discuss an approach to support metrologists in finding an optimal sequence of measurements to keep variability of the mean value and the standard uncertainty small. I want to forestall that the problem of optimizing the order of reference and sample measurements with too many degrees of freedom is very challenging for a universal solution;

results depend mainly on the actual problem. Hence, it is necessary to make some concrete assumptions about the parameters that describe the environmental conditions for concrete examples. The approaches that will be presented and discussed in the next sections are explained using the titration measurement from the last chapter.

6.1 Definition of measurement series

We continue enhancing the example of the titration measurement from analytical chemistry determining the amount of some substance in a solution. Until now we have developed a model to describe a single measurement adequately assuming ideal environmental conditions for a measurement. We have quantified uncertainty sources very conservatively, because variation of influence quantities has to be covered by the model. This is necessary as the quotation at the beginning of this chapter has already pointed out. Customers must obtain reliable results with a proper measurement uncertainty statement that is valid independent of the time or date when the measurement was performed. Nevertheless, we want to have a look at series of measurements, where the just introduced processes are used for the definition of alternating sample and reference measurements considering the variation of the environmental conditions from measurement to measurement explicitly as there are very important and interesting questions with respect to having repeated measurement under control and understanding the behavior of the measurement system. Figure 6.3 shows a titration device from the world market leading manufacturer Metrohm equipped with a sample changer. It is able to perform series of measurements fully automatically, where only the samples and references have to be prepared and put into the sample tubes.

We use the basic models and process definitions from the previous chapters. This allows us to focus on the simulation definition enhancing the processes section where necessary and mainly adding functionality to the calculation section and the definition of the measurand. Until now we have defined the processes `sam1` and `sam2` for sample measurements and `ref` for a reference measurement that is used in combination with the two processes for the samples in the definition of the measurement models. We want to keep the example simple and thus, concentrate only on the first of the two sample measurements `sam1`, renaming it to `sam`. To describe series of measurements we introduce the new section `series` in the section of the measurand in the calculation section. It contains arbitrarily many sections `reference` that contain themselves definitions for sample measurements. The next code fragment using the three process definitions should clarify this concept.

Figure 6.3: Titration device of manufacturer Metrohm equipped with a fully automatic sample changer

```
<processes>
  <!-- One process for reference measurement -->
  <process name="ref">
    ...
    <formula>mref*purref/vref</formula>
  </process>

  <!-- Process sam1 renamed to sam -->
  <process name="sam">
    ...
    <formula>vsam1/msam1</formula>
  </process>
</processes>

<calculation>
  <measurand>
    <series>
      <reference name="rtit" process="ref" number="5">
        <sample name="stit" process="sam" number="5"
             analyze="linlastref"/>
      </reference>
    </series>
  </measurand>
</calculation>
```

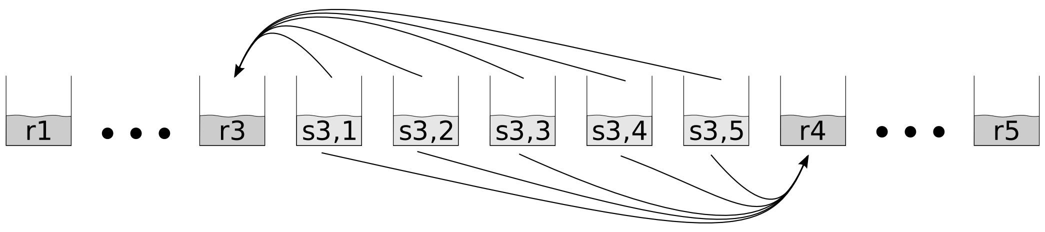


Figure 6.4: Series of titration measurements with five reference measurements; five sample measurements are performed between reference measurements.

The reference and sample measurement definitions must have an identifier and a process used for a single evaluation. The evaluation of the example results in the simulation of five reference measurements; between each of the reference measurements five measurements according to process `sam` are evaluated. Figure 6.4 shows the sequence of measurements. The arrows in the figure indicate that the sample measurements refer to the previous and the following reference measurements. Different methods are applied in practice to refer sample measurements to one or more reference measurements. Let $n = 5$ be the number of reference measurements, $m = 5$ the number of sample measurements between two references, $i = 1, \dots, n$ an index for the last reference measurement and $j = 1, \dots, m$ an index for the current sample measurement. The attribute `analyze` of an element `sample` describes how the j^{th} sample measurement $sam_{i,j}$ should refer to the previous reference measurement ref_i and additional reference measurements. Currently implemented methods are

- `linlastref`: The sample measurement only refers to the previous reference measurement; the equation used is

$$s_{i,j} = \frac{sam_{i,j}}{ref_i}.$$

- `linnextref`: The sample measurement only refers to the next following reference measurement; the equation used is

$$s_{i,j} = \frac{sam_{i,j}}{ref_{i+1}}.$$

- `lin2mean`: The sample measurement refers to the average of both the surrounding reference measurements; the equation used is

$$s_{i,j} = \frac{2}{(ref_i + ref_{i+1})} \cdot sam_{i,j}.$$

- `lin4mean`: The sample measurement refers to the average of the two previous and the two following reference measurements; for $j > 1$ the equation used is
  $$s_{i,j} = \frac{4}{\sum_{k=i-1}^{i+2} ref_k} \cdot sam_{i,j}.$$
  For the sample measurements in the first and last gap of reference measurements the next four closest reference measurements are used.

- `linallmean`: The sample measurement refers to the average of all reference measurements; the equation used is
  $$s_{i,j} = \frac{n_r}{\sum_{k=1}^{n} ref_k} \cdot sam_{i,j}.$$

- `fitline`: This concept does not rely on the average of reference measurements, but fits a line through the result of the previous and next reference measurement and evaluates the corresponding function value for the sample; it assumes that measurements are done continuously in equidistant time slots.

Of course more advanced techniques would be possible at this point using regression analysis and better algorithms that correct the outcome of the measurements instead of relying on a single reference measurement or the average of a number of reference measurements; but we focus in this example on intermediate or long-term fluctuation scenarios. The presented methods are used and are common in daily work in laboratories, where results of series of measurements are often analyzed with minimal effort. If reference measurements are only performed once in two weeks, referring to more than the last and the next reference measurement seems of little use. Moreover, additional assumptions about the behavior of environmental conditions may have to be made that are hard to justify. It also depends strongly on the actual problem if the approaches that rely on the mean value are sufficient. To keep close to real-life applications and to be fit for practical applications, the system currently supports the commonly used methods, but the framework can easily be extended. So, the last simple line fitting method shows a first approach to a slightly improved method. Hence, the system gives a deep insight and helps comparing the different approaches.

Until now the last simulation definition is somewhat limited as the sample and reference process definitions are the same as before. Hence, currently the same distributions result from the evaluation of each of the evaluations for process `sam`. To describe the behavior of the measurements from process to process we have to enhance the variable definition once more, this time especially for the process definition. First, we introduce three global variables for series of measurements.

They are updated automatically by the simulation system. The first global variable _ref is a counter for the current reference measurement; the second global variable _sam is a counter for the current sample measurement since the last reference measurement. Finally, the variable _cur counts iteratively from measurement to measurement regardless if it is a reference or sample measurement. These indexing variables can be used in the definition of variables for processes to distinguish between single measurements. Variable settings in processes can depend on the value of the global variables _cur, _ref, and _sam. It allows individual settings for measurements, e.g., performed in distinct time slots. The next code fragment shows the concept and uses the advanced technique to define variables for processes, where variable values depend on the values of the just introduced global variables _ref and _sam.

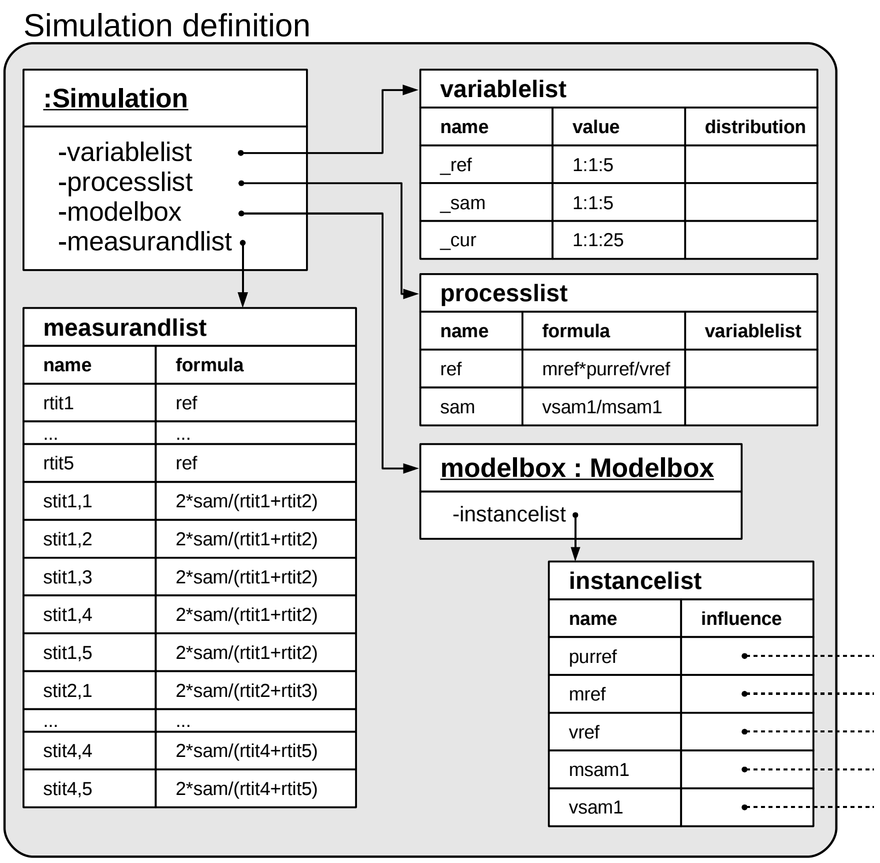
```
...
<processes>
  <!-- One process for reference measurement -->
  <process name="ref">
    <!-- Varying temperature in laboratory -->
    <variable name="Tlab">
      <switch>
        <case refindex="1">20.0</case>
        <case refindex="2">19.1</case>
        ...
        <case refindex="5">20.9</case>
        <default>21.0</default>
      </switch>
    <variable name="Tlab">

    <formula>mref*purref/vref</formula>
  </process>

  <!-- Process for first sample measurement -->
  <process name="sam">
    <!-- Varying temperature in laboratory -->
    <variable name="Tlab">
      <switch>
        <case refindex="1" samindex="1">19.7</case>
        <case refindex="1" samindex="2">19.3</case>
        <case refindex="1" samindex="3">19.1</case>
        <case refindex="1" samindex="4">18.9</case>
        <case refindex="1" samindex="5">19.0</case>
        <case refindex="2" samindex="1">19.6</case>
        ...
        <case refindex="5" samindex="5">20.5</case>
            <!-- 23:20 -->
        <default>21.0</default>
      </switch>
      </variable>
      <formula>vsam1/msam1</formula>
    </process>
  </processes>
  ...
```

A section `switch` allows a variable to change its value depending on the predefined global variables. In the example constant values are used to define cases, but it is also possible to provide ranges of values using the attributes `reffrom`, `refto`, and `refstep` instead of the attribute `refindex`. The same holds for the iteration variables for sample measurements and the global counter. Internally, the definition above would result in a system as shown in Figure 6.5. During simulation the variables `_cur`, `_ref`, and `_sam` are updated automatically and can be used to set individual parameters in processes for the current reference or sample measurement. The value setting for the two variables `_ref` and `_cur` define an iteration variable each, given a starting value, step size, and terminal value separated by a colon, respectively. The list of equations contains all equations for the evaluation of the reference measurements, followed by a list of the corresponding sample measurement definitions. In the example the two closest reference measurements are used for evaluation of the current sample measurement. That would correspond to a setting of attribute `analyze` to the value `lin2mean`. It is important to see that the calculation of the final result for a sample measurement depends on the result of one or more reference measurements. Thus, all reference measurements are evaluated in advance, so that the necessary information can be provided when evaluating formulas of the sample measurements. This is an issue as it requires reference measurements to be independent from the outcome of previous sample measurements. Nevertheless, dependencies can be modeled via instances of basic models or additional process definitions.

Before we start now to analyze repeated measurements in more detail, I want to describe a concrete measurement scenario considering the changes over a usual working day in a metrological laboratory that uses NIST reference material to certify further reference material for end users. We will focus on two varying influence quantities; the first is again the influence of the ambient temperature during measurement as we have just seen in the last code fragment, where the temperature was set individually depending on the two global variables `_ref` and `_sam`. To shorten code and prevent lengthy definitions, I will from now on use the following three equations that were found suitable to represent the fluctuation of the temperature in a laboratory for a working day, where 25 measurements take place. The equations only depend on the value of the global variable `_cur` $= 1, \ldots, 25$ to approximately calculate the temperature difference and variation for three distinct scenarios:



variablelist

name	value	distribution
_ref	1:1:5	
_sam	1:1:5	
_cur	1:1:25	

processlist

name	formula	variablelist
ref	mref*purref/vref	
sam	vsam1/msam1	

measurandlist

name	formula
rtit1	ref
...	...
rtit5	ref
stit1,1	2*sam/(rtit1+rtit2)
stit1,2	2*sam/(rtit1+rtit2)
stit1,3	2*sam/(rtit1+rtit2)
stit1,4	2*sam/(rtit1+rtit2)
stit1,5	2*sam/(rtit1+rtit2)
stit2,1	2*sam/(rtit2+rtit3)
...	...
stit4,4	2*sam/(rtit4+rtit5)
stit4,5	2*sam/(rtit4+rtit5)

instancelist

name	influence
purref	
mref	
vref	
msam1	
vsam1	

Figure 6.5: Internal concept of an explicit series of measurements definition; there are three global variables `_cur`, `_ref`, and `_sam` and an enhanced list of equations for reference measurements and sample measurements afterwards as they depend on the outcome of the reference measurement simulation.

- Temperature deviation over a day without any temperature control is defined as
$$dt_1 = \frac{2 * \sin(x)}{\sqrt{0.5x + 0.2}}$$
with $x = 2\pi * (\texttt{\_cur} - 1)/25$.

- Temperature deviation over a day with air condition, but no temperature control[1], is defined as
$$dt_2 = 25x/10 - 1,$$
where
$$x = \begin{cases} \sin((\texttt{\_cur} - 1) * \pi/20) & \texttt{\_cur} \leq 10 \\ 1.0 & \texttt{\_cur} > 10 \end{cases}$$

- Temperature deviation over a day with a temperature controlled air condition[2]
$$dt_3 = \frac{dt_1}{(1 - (\texttt{\_cur}/25))^2},$$
where the denominator is used for a strong attenuation over time.

The actual temperature is then defined as a curvilinear trapezoidal distribution with parameters $a = 21.0 + dt_i$, $b = 21.5 + dt_i$ with a corresponding $i \in 1,2,3$, and an inexactness of $beta = 0.5$.

The second quantity for variation is the electrode that is used to detect the potential of solution. The electrode does contribute to the measurement uncertainty as the response time of the electrode increases over time. The electrode and its uncertainty contribution are already included in Equation 4.3.5 as influence quantity V_{det}. The effect of the electrode that we will consider here is a worsening of detection over a specific day. The behavior is represented by a continuous increase of uncertainty as well as an increasing overestimation of the volume to be measured. Figure 6.6 shows the plots of mean and standard deviation for the two influence quantities electrode and temperature, where three different approaches are used for the modeling of the ambient temperature in the laboratory.

In Figure 6.7 the scenario with no temperature control in the laboratory is used to visualize the effects of the different referring methods. It can be seen that taking too much references into account using only a simple mean value over some

[1]We assume an air condition that just produces air of the same temperature for the laboratory the whole day; no temperature sensors are used to adjust the actual environmental temperature that may change because of running measurement equipment, computers or entering and leaving staff.

[2]Here we assume that the sensors react rather moderately in adjusting the temperature so that no heavy over- or undershooting takes place.

references is counterproductive, resulting in a series of results for sample measurements without taking the local environmental conditions into account. Hence, a larger standard uncertainty has to be reported. The current scenario favors the line fitting approach with a very stable mean value.

Until now we had a concrete measurement scenario with exactly five reference measurements and five sample measurements between each two reference measurements. In the following we want to concentrate on the question of how many reference measurements are necessary and how they should be distributed if we have a given well-defined measurement scenario. We will have a look on the consequences of reducing the number of reference measurements and the behavior of the system in such a case.

6.2 Optimizing repeated measurements

In the last section we have seen how sequences of measurements with a given order can be defined and simulated with the help of the modeling language. This construct allows to model real-life measurement scenarios containing not only a single measurement, but alternating series of reference and sample measurements. The sequence of sample and reference measurements and environmental conditions for each measurement is well-defined and the simulation system can evaluate the measurement uncertainty and other statistical parameters straightforwardly using the Monte Carlo method.

In this section we go one step further and suggest an approach to look for proper placements of reference measurements in a sequence of reference and sample measurements with respect to stable statistical parameters like the mean and a preferably small standard uncertainty for the sample measurements. The cost function for such a quasi-optimization is not trivial to define and hence, we have to make certain assumptions that hold in practice. Ultimately, we show that the simulation system and the modeling language are able to handle such complex scenarios and deliver all the necessary prerequisites for further analysis. For more advanced analyzing techniques the resulting simulation data can be used in other statistical packages, e.g., the freely available statistical software environment R [63].

6.2.1 Prerequisites for optimization

To be able to suggest a sequence of alternating reference and sample measurements we have to make some assumptions to reduce the complexity of the problem:

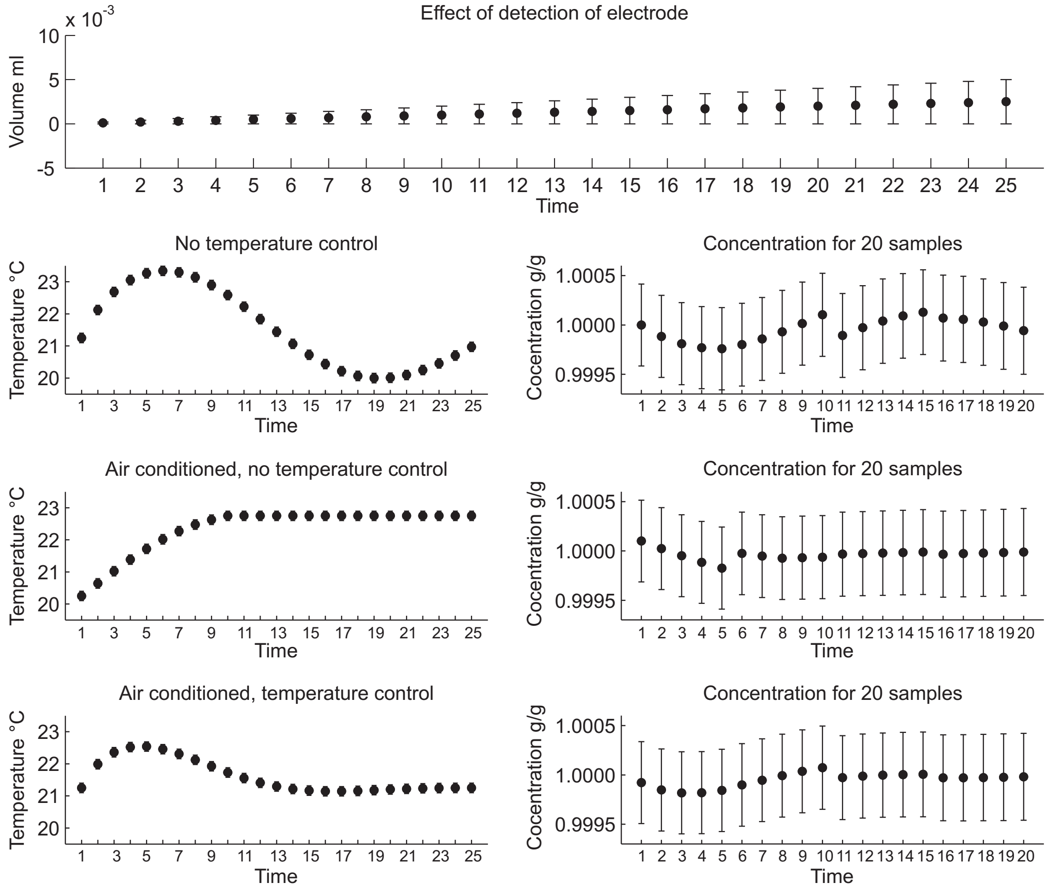


Figure 6.6: The upper plot shows the behavior of the detection electrode for a specific day; the lower part on the left-hand side describes three different scenarios for varying temperatures over a day and on the right-hand side the corresponding resulting concentration from a measurement scenario with 5 reference measurements and 5 sample measurements in each gap; sample measurements use the mean value of the two surrounding reference measurements.

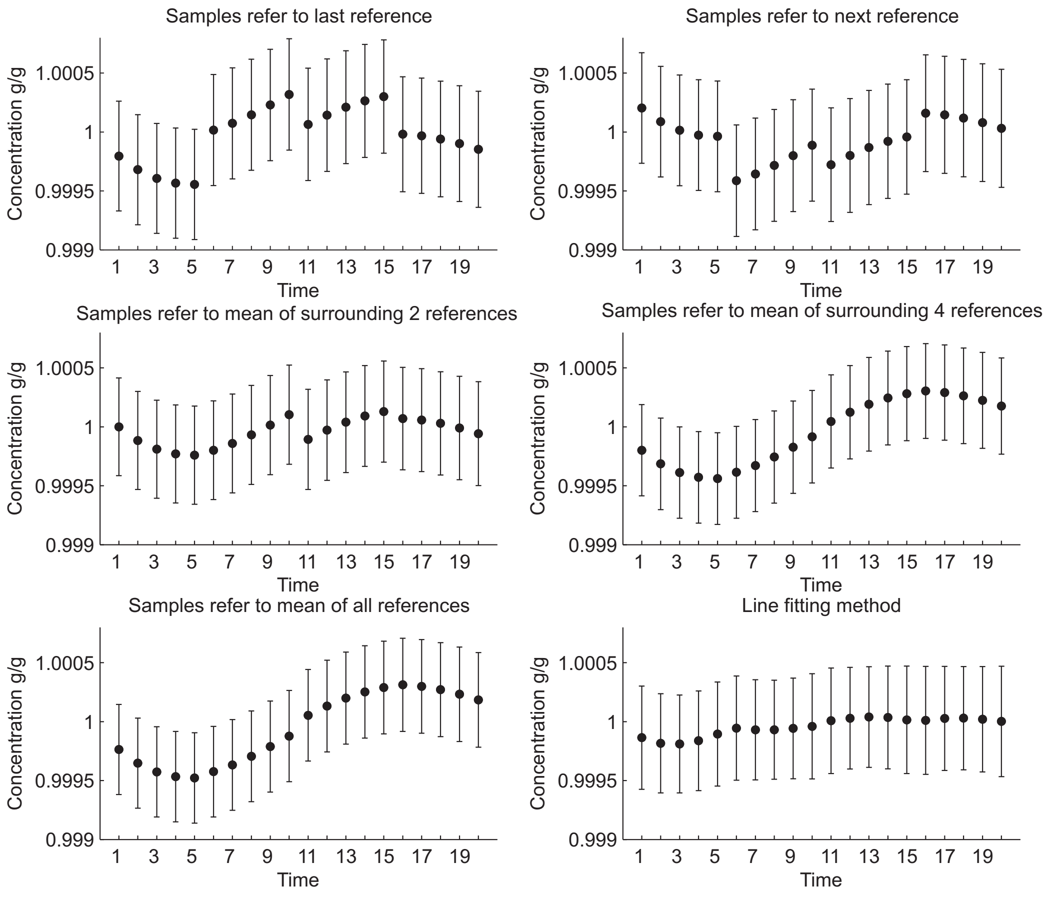


Figure 6.7: The plots show how results differ if different methods are applied to referring sample to reference measurements. The upper two plots only refer to a single reference measurement and hence, the fluctuations are rather strong; the two plots in the middle use the mean value over two, respectively four, reference measurements and hence the fluctuations are much more moderate; using the mean over all reference measurement does not take account of the effect to compensate environmental fluctuations; the best result is obtained applying the line fitting algorithm as the system behaves linear enough between each two reference measurements for this scenario, resulting in a very stable mean value.

- The beginning of the measurement is always at the same time. We cannot expect that in a laboratory one will wait and let some time pass until environmental conditions get better. In fact, more reference measurements will be performed in the beginning to be sure about suitable conditions if necessary.
- The first measurement as well as the last measurement is always a reference measurement. This is also very intuitive as you would like to know that the environmental conditions are suitable in the beginning as well as in the end of the measurement.
- We assume that sample measurements as well as references measurements can each be described with the formerly introduced processes.
- The search space is discretized to so-called time slots and as measurements are done continuously, each subsequent time slot is used; we are looking for a tight chain of measurements without empty time slots. In a laboratory no one would miss two or three possible time slots for measurements to avoid a probably local peek with higher uncertainty as time is money.

The input parameters for the routine to analyze different sequences are the following:

- one model/process for reference measurements
- one model/process for sample measurements
- number n_s of sample measurements
- definition of a method referring sample measurements to the reference measurements
- flag that indicates if unnecessary reference measurements should be removed from the sequence (sequence gets shorter) or if it should be replaced with additional sample measurements (sequence keeps its length).

The maximum size of the measurement series is always $2n_s + 1$ as the initial scenario will always be a one reference by one sample measurement scenario. With the number n_s of sample measurements and the condition of at least two reference measurements, one in the beginning and one at the end of the measurements, we have a minimum size $n_s + 2$ of the measurement series if reference measurements are removed from the series. The length of the measurement series does not change if reference measurements are replaced with additional sample measurements.

6.2.2 Method for optimizing

There are two objectives for optimizing series of measurements and we are looking for a good trade-off between these two distinct positions:

- The best scenario with respect to good statistical parameters like a stable mean value and a small standard uncertainty for sample measurements is if reference and sample measurements alternate one by one, starting and ending with a reference measurement. This is the system of maximum control, monitoring the environmental conditions after each sample measurement.
- The best scenario with respect to low costs and short time frames is if there are as few reference measurements as possible in the sequence of measurements. We assume that the optimal scenario according to cost and time is achieved if only two reference measurements are to be applied, one at the beginning and one at the end of the sample measurements.

The interesting questions are now how many and in which time slots reference measurements should take place for proper behavior and description of the system. We are looking for a good trade-off of these two contrary positions. To start an investigation for important time slots for reference measurements, we begin with an optimal scenario according to the first scenario with alternating reference and sample measurements. We will have a look at how the system worsens if we remove reference measurements one by one from time slots with the smallest impact to the mean value and standard uncertainty. Hence, the process will look as follows:

- Generate an initial one by one sequence
- While there are more than two reference measurements in the sequence do
  - Calculate cost function for each subsequent triple of reference measurements
  - Remove/replace the reference measurement in the middle of the triple with the smallest result according to the cost function
  - Evaluate the new scenario
- Calculate parameters relevant for decision making

An interesting point in the algorithm is the vague formulation of a cost function. Our approach – the current implementation – fits a line through the two limiting reference points r_1 and r_3 and calculates the orthogonal distance $||\mathbf{d}||$ from the point r_2 in the middle to the line. The algorithm first calculates the vectors

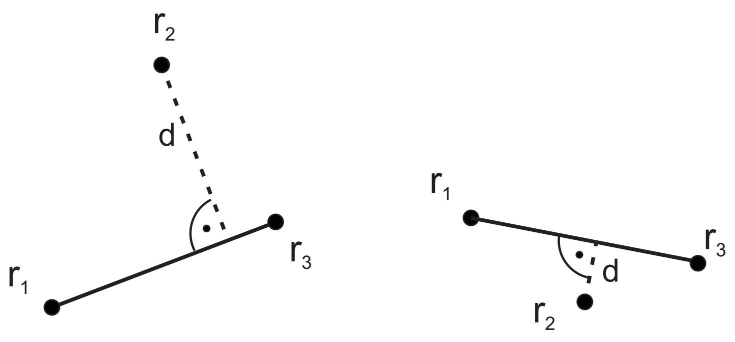


Figure 6.8: We are looking for a triple (r_1, r_2, r_3) of reference measurements where the orthogonal distance $||\mathbf{d}||$ from r_2 to the line from r_1 to r_3 is minimal.

$\mathbf{x} = r_2 - r_1$ and $\mathbf{v} = r_3 - r_2$, and normalizes vector $\mathbf{v}$ to vector $\mathbf{u} = \frac{\mathbf{v}}{||\mathbf{v}||}$. We can then calculate the orthogonal projection $\mathbf{d} = \mathbf{x} - \mathbf{u}\mathbf{u}^T\mathbf{x}$ to the vector $\mathbf{v}$. Two such scenarios are shown in Figure 6.8. The elementary cost function is then used for a decision on which reference measurement should be removed as described in the following algorithm:

- Initialize each reference with a vote counter of 0
- For each block of – e.g., 10^5 – trials
  - Initialize each reference with a summed distance of 0
  - For each triple of references
    - Calculate the distance for each set of values as described and sum it up for the corresponding reference
  - Search for the reference with the minimal summed distance and add one vote for it
- Remove or replace reference with the highest vote.

This algorithm is the key to the optimizing system. As the evaluation method of the simulation core is a Monte Carlo approach, we repeat the above sequence of steps for each trial in every block. This allows us also to bring in the information about the uncertainty of the measurement result. Hence, we do not only get one result, but a whole set of results. To decide which reference measurement should be removed or replaced the distances are summed up for each block, and after the processing of a block we look for the minimal distance and add one vote for it. In the end, the reference with the highest vote is replaced or removed. This method gives us additional information about reference measurements, for example the ranking for removing reference measurements or the standard deviation of the distances.

6.2.3 Application of the method

With all the previous definitions for the evaluation of the measurement uncertainty for a titration measurement it is quite easy to define the scenario for optimization. We first assume that we have an automatic sample changer for exactly 25 samples. Hence, if we have a one by one scenario, 12 sample measurements are the maximum together with 13 reference measurements. The following code fragment of the calculation section suffices as a starting point for the simulation.

```
<calculation>
  <analyze mode="on" histbars="40" datafiles="delete"/>
  <measurand>
    <optimize reference="ref" sample="sam" samples="12"
       mode="replace" analyze="lin2mean"/>
  </measurand>
</calculation>
```

The new element `optimize` is all we need for an optimizing run. The attributes `reference` and `sample` define the processes for reference and sample measurements. Attribute `samples` defines the number of sample measurements for the initial one by one scenario, in our case 12 sample measurements. The number of sample measurements remains constant if reference measurements are removed from the sequence, e.g., if only a constant number of samples is available. The number increases if reference measurements are replaced by additional sample measurements, e.g., if there are enough samples available and the maximum number of samples should be measured in one run. Both cases may occur in daily work and can be represented by the system. Finally, the attribute `analyze` defines how sample measurements should refer to the reference measurements as we already have seen for the definition of explicit measurement scenarios and sequences using a section `series` for the measurand.

Before we analyze the results of such a simulation for sample measurements, Figure 6.9 shows how one after another reference is removed from the measurement sequence. Due to removal of measurements and a shorter measurement sequence in the right column, the worsening of the electrode in respect of a larger measurement uncertainty contributes less to later scenarios. This is an advantage over the right scenario where the electrode fully affects the results for reference measurements.

The algorithm to reduce reference measurements behaves as expected, keeping extreme reference measurements in the measurement sequence as long as possible. The example was performed with 10^7 trials per simulation and the result of the voting algorithm was mostly concordant. The highest discrepancies were obtained for the first removal as well as for the first replacing.

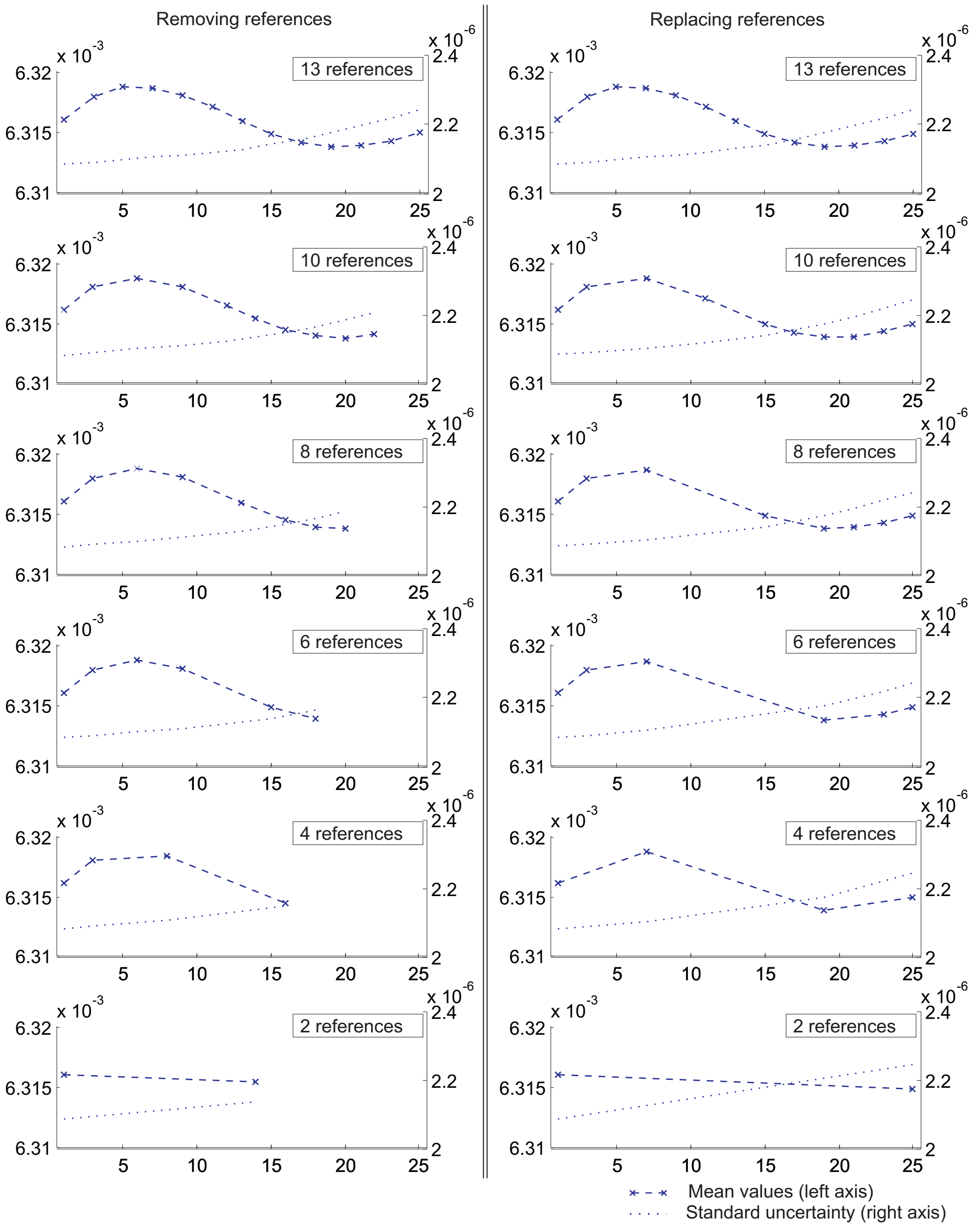


Figure 6.9: Comparison of progress of removal (left column) respectively replacing (right column) reference measurements from a measurement sequence; only every second step is plotted. Both ordinates are given as concentration in g/ml. It can clearly be seen how information is lost with each removal/replacement.

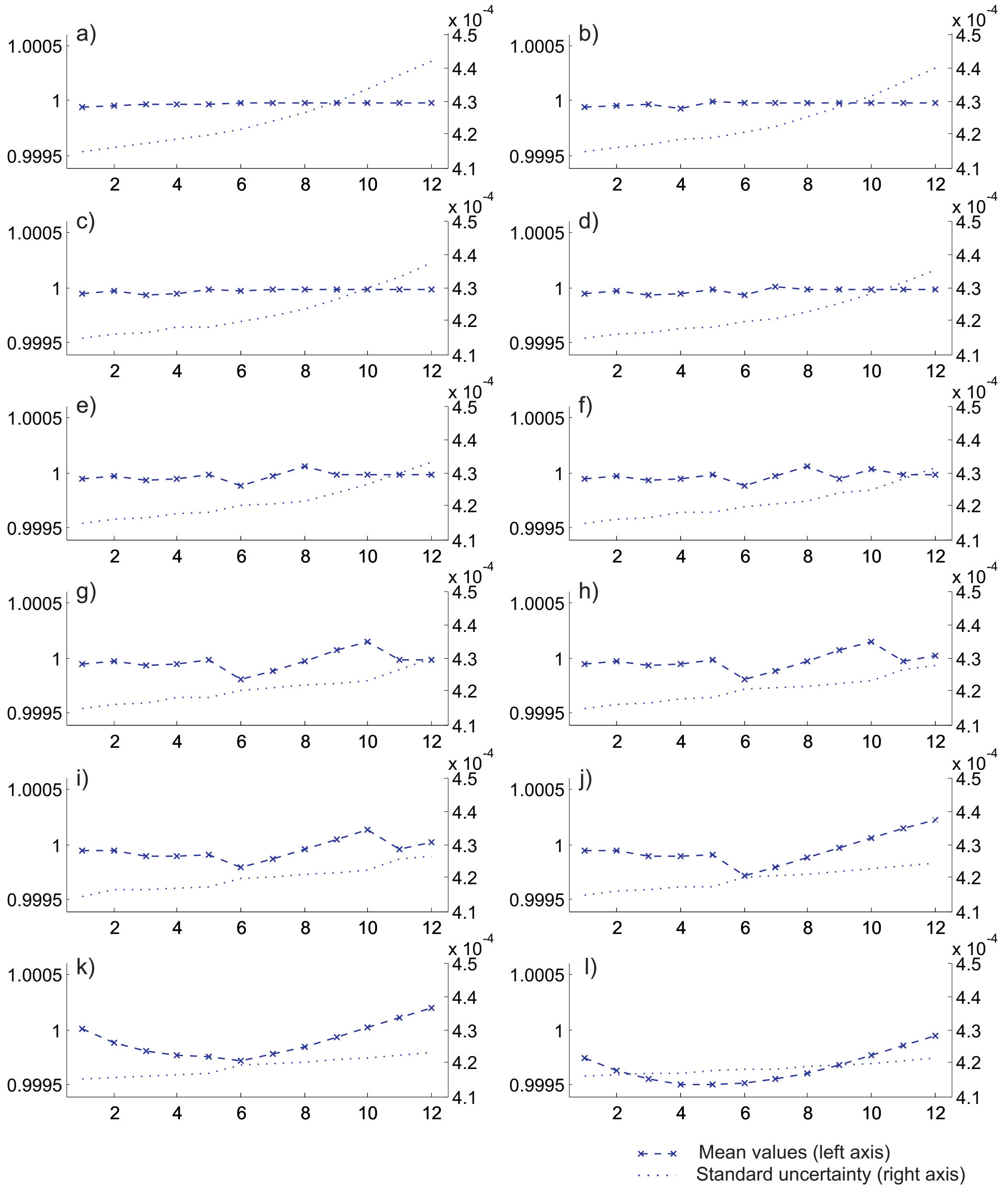


Figure 6.10: Result of an optimizing run starting with a one by one scenario in a) *removing* one reference measurement in every step until only two references are left in l). Hence, the number of overall measurements decreases. The left ordinates for the mean values are all given as concentration in g/g as well as the right ordinates for the standard uncertainty.

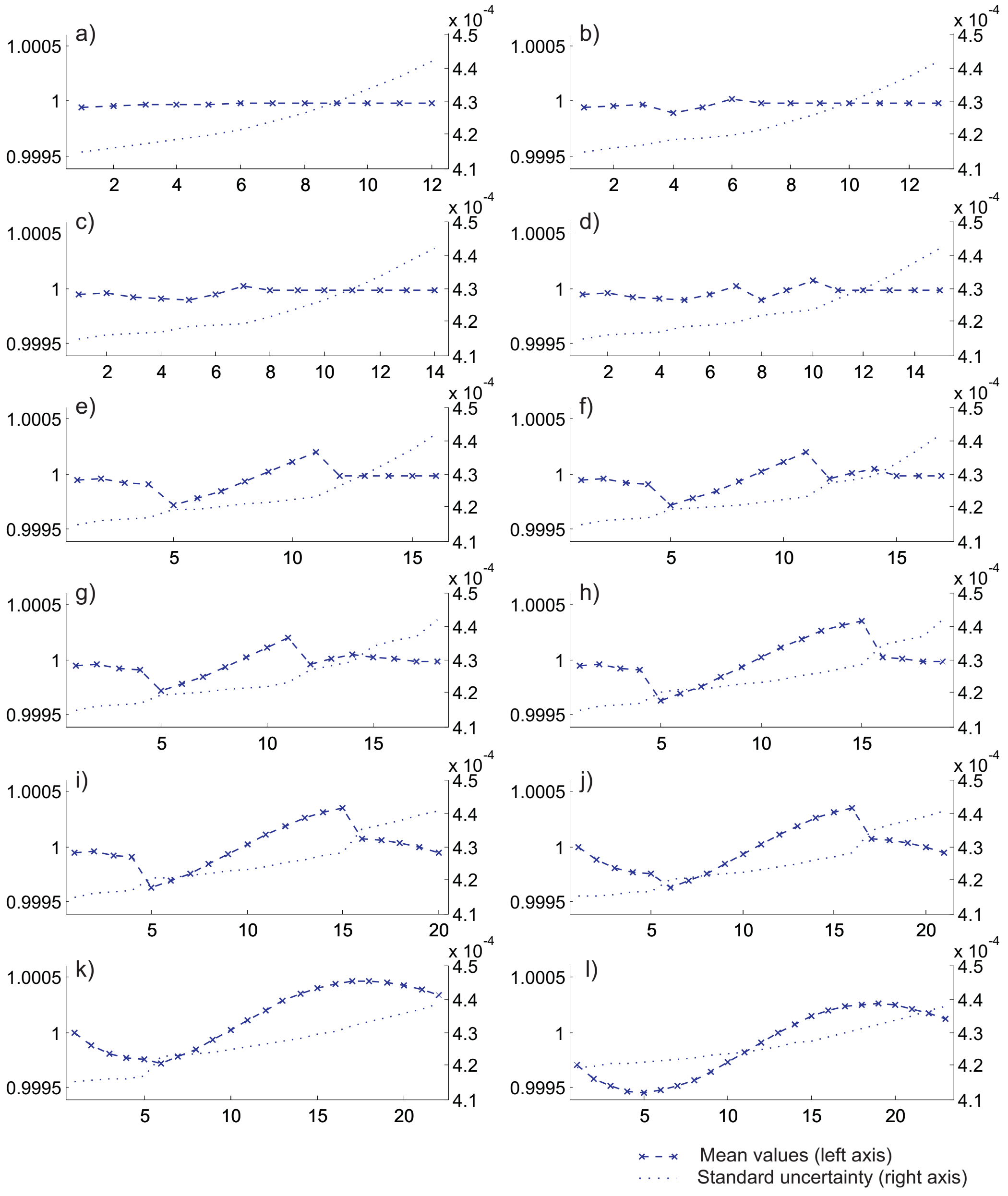


Figure 6.11: Result of an optimizing run starting with a one by one scenario in a), *replacing* a reference measurement with a sample measurement in every step until only two references are left in l) again. The number of overall measurements keeps constant in this case. The left ordinates for the mean values are all given as concentration in g/g as well as the right ordinates for the standard uncertainty.

Figures 6.10 and 6.11 show the results of an optimizing run, each scenario is evaluated with 10^7 trials per simulation[3]. From Figure 6.10, where we remove reference measurements, it can clearly be seen that the stability of the mean value decreases with every removal of a reference measurement. It is interesting that the measurement uncertainty decreases during the measurement if less reference measurements are performed. The explanation for this observation is that more measurements require a larger time frame. As we are working with an electrode that worsens dramatically over the day of interest, we introduce a higher uncertainty for later measurements. However, if we compare the maximal limits for the standard uncertainties of the first (a) and the last (l) scenario in Figure 6.12, we see that the bandwidth for the first scenario is narrower despite the fact that the individual uncertainties are smaller in the last scenario.

In Figure 6.11 the measurement uncertainty is not reduced in later measurements for exactly the same reason. In this figure the fluctuation of the temperature can also be seen in more detail. Performing not enough reference measurements results in an uncorrected result with a strongly varying mean value. Figure 6.12 allows comparing the three extreme cases – the one by one scenario, the shortened and the usual two reference scenario – as a mean and standard uncertainty plot with the corresponding coverage intervals.

Having the resulting plots one might think about suitable parameters for decision making about the number of reference measurements. It is not trivial to formulate a suitable general cost function for all possible cases as we are interested in as less reference measurements as possible, but keeping a small standard uncertainty and a stable mean value. I think that the information provided can help in decision making, but a predefined cost function could lead in a wrong direction. It is up to the expert utilizing the provided framework and analyzing the requirements and priorities for the specific problem.

Finally, I want to conclude that the presented system allows a deep insight into measurement scenarios of high complexity as they appear in real life using an efficient Monte Carlo simulation core. The titration measurement that accompanied us in the previous chapters allowed me to explain and argue the structure and concepts of the domain-specific language of the simulation system. We started with the introduction of identifiers, expressions, and probability distributions, explained the concept of building hierarchical basic models that represent measurement equipment in an abstract way. Afterwards, processes were introduced that allow a more structured modeling of concrete measurement scenarios without changing the structure of basic models but only setting necessary param-

[3]For once I want to mention that the simulation of all scenarios including the calculation of the score and analyzing the data takes about 6 Minutes for each run which leads in our case to a running time of a little more an hour using only a single core with 1.6GHz.

eters. The simulation definition was also extended with variables and different ways to define measurement scenarios with more than one measurand. Using these prerequisites we were able to describe very complex measurement scenarios that depend on many influence quantities and consist of not only single measurements but repeated measurements. In the end we have a simulation framework that allows comparing different scenarios very conveniently. The presented approach to reduce the number of reference measurements is new and a first suggestion; it may not hold in this form for all cases, but for the titration measurement it already delivers appropriate and interesting results, which could be gained in practice only with intensive research and costly measurements. The optimizing routine is also encapsulated in the framework in an own class, which allows easy adaptions and enhancements of the algorithms.

We will now leave the titration example and concentrate again on a measurement with a single outcome in the next chapter. We will have a look on sensitivity analysis and the question of how much a single influence quantity contributes to the standard uncertainty of the whole measurement in a measurement model.

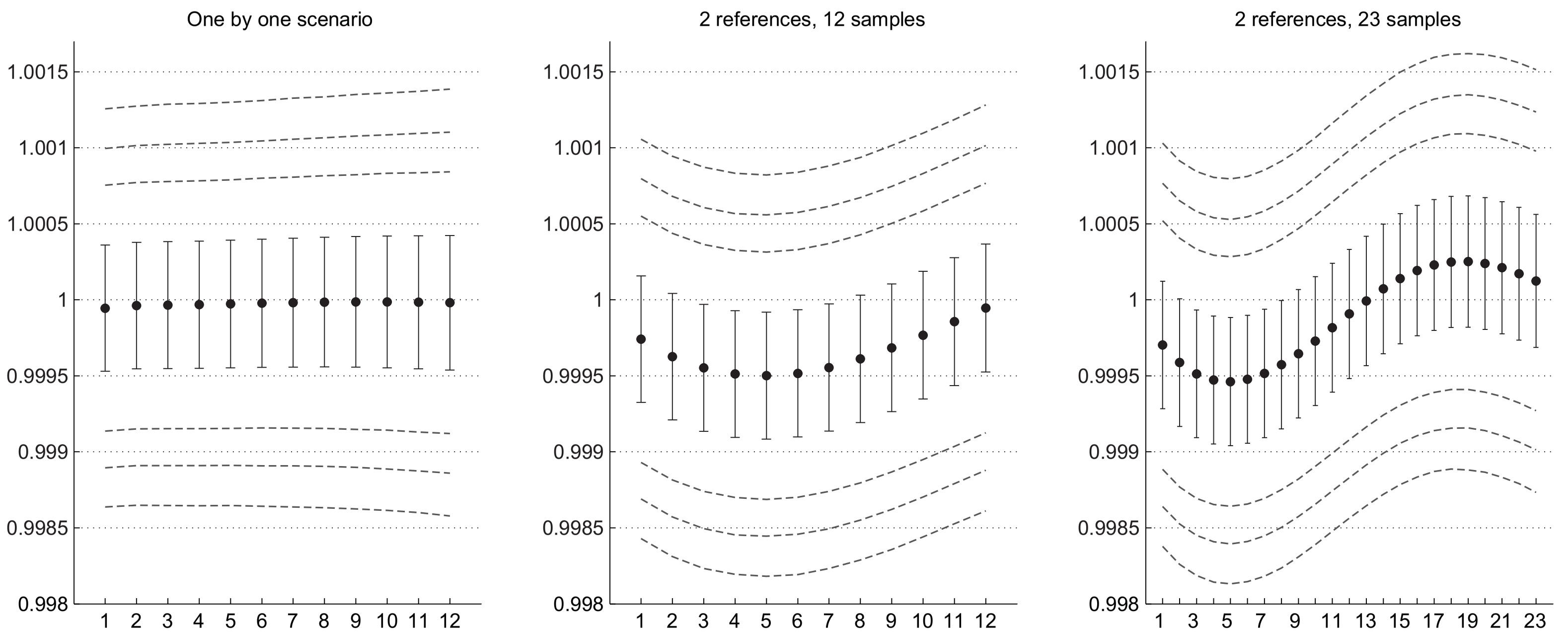


Figure 6.12: The initial one by one scenario in the left plot is the same for both methods of reducing the number of reference measurements. In the middle we see the maximum reduction to two reference measurements if we just remove one reference after another from the measurement series. In the right plot reference measurements are replaced with sample measurements. The maximum range of measurement uncertainty to be reported is – as expected – bigger for both methods of reference reduction. The ordinates show the concentration in g/g again.

CHAPTER 7

Sensitivity analysis

Of course, the computer isn't any of these things. These are all things we were previously familiar with from the real world which we have modeled in the computer, so that we can use the damn thing. Which should tell us something interesting. The computer is actually a modeling device. Once we see that, we ought to realize that we can model anything in it. Not just things we are used to doing in the real world, but the things the real world actually prevents us from doing.

— From Calculator to Typewriter, Douglas Adams, 1997 —

The first supplement to the GUM appeared recently in its final version and people just begin to realize the capabilities of the Monte Carlo method and the advantages of higher-resolution models. For example, in laboratories it is very interesting to find influence quantities with a high potential for reduction of measurement uncertainty. Hence, a measure for the importance of influence quantities or parts of the model according to measurement uncertainty is of high relevance. As the calculation core of the project *MUSE* is given and it implements a very efficient way to evaluate the measurement uncertainty even for very elaborate measurement scenarios, it is obvious to use this ability and flexibility to examine the measurement uncertainty in more detail. There are two related approaches, uncertainty and sensitivity analysis, which are defined as in the following citation [12]:

> Generally speaking, the objective of *sensitivity analysis* (SA) is to quantify the effects of parameter variations on calculated results. Terms such as influence, importance, ranking by importance and dominance are all related to sensitivity analysis. On the other hand, the objective of *uncertainty analysis* is to assess the effects of parameter uncertainties on the uncertainties in calculated results.

Uncertainty analysis analyzes the behavior of a measurement scenario in respect to measurement uncertainty evaluation, where parameters of probability distributions of input quantities are varied directly to examine the effect in the resulting measurement uncertainty of the measurand. The necessary concepts and constructs are provided in the modeling language using variation variables that were already described in Section 4.5.

In the current chapter we will focus on sensitivity analysis, examining the impact or importance of an individual influence quantity or part of the model to the combined measurement uncertainty, also in comparison to the impact of all the other influence quantities. The most direct approach for sensitivity analysis is to fix specific influence quantities to a representative value, e.g., mean or median, to prevent its variation for simulation. Another method is to apply variance analysis that uses properties and capabilities of the Monte Carlo approach, e.g., using Sobol' sensitivity indices [68, 72] as we will see in this chapter. All in all, the goal is to take advantage of the efficient Monte Carlo simulation system and build some kind of a what-if machine that answers interesting questions regarding the measurement uncertainty and allows a deep insight into measurement scenarios and the dependencies of influence quantities on the measurement uncertainty.

I will present selected approaches that show how sensitivity analysis can be applied and what the differences between these approaches are. A good overview and summary of sensitivity analysis methods can be found in [68]. In the article *Hitchhiker's guide to sensitivity analysis* in this book the authors make a distinction between

- *factor screening* to extract influential factors in systems with a large number of input quantities,
- *local sensitivity analysis* to examine local (point) impact of the influence quantities, usually using partial derivatives or analytical methods and
- *global sensitivity analysis* to apportion the uncertainty to the influence quantities. Usually a sampling approach is used here.

Other comparisons of methods for sensitivity and uncertainty analysis can be found in [38, 44]. In this work I decided for a combination of factor screening methods and the *Sobol' sensitivity indices* (SI) as global sensitivity analysis approach[1]. The methods are very intuitive and allow getting a feeling for the behavior of the measurement scenario under investigation. All presented methods rely strongly on the Monte Carlo method and are suitable for linear as well as non-linear models. In the next section I will introduce an example taken from GUM with a manageable number of influence quantities that will help to understand the concepts of sensitivity analysis. Afterwards, I will give a short insight into the GUM uncertainty framework and its implicit local sensitivity analysis, before starting with factor screening and global SA.

7.1 Gauge block calibration

The example of a gauge block calibration describes the measurement uncertainty evaluation for a length measurement of a nominally 50 mm gauge block by comparing it with a well-known reference standard of an equal nominal length of 50 mm. The example is included in GUM [37, Section H.1]. It uses a linearization of the equation of the measurand and limits the probability distributions of the input quantities to normal distributions applying conversion rules for different kinds of probability distributions. In GS1 [22, Section 9.5] as well as in the *Best Practice Guide No. 6* [15, Section 9.15] the example appears again in a slightly different form showing differences in the results of GUF and the new Monte Carlo approach. I decided for this example of sensitivity analysis as it is well known and used in many publications because of the elaborate description and the documented results in the articles. Besides, the example is not too complicated with a limited number of influence quantities, so that effects and results can be explained more easily.

The following list includes all influence quantities with the names used in the model definition and the type of probability distributions including its parameters:

L_s (`ls`), length of the reference standard in mm
t-distribution with $t_{18}(50000623, 2.5 \times 10^{-5})$

D (`dq`), average length difference in nm
t-distribution with $t_{24}(215, 6)$

d_1 (`d1`), random effect of comparator in nm
t-distribution with $t_5(0, 4)$

d_2 (`d2`), systematic effect of comparator in nm
t-distribution with $t_8(0, 7)$

[1]The selection was inspired by private communication with Peter Harris from NPL in 2007 and a talk of Nicolas Fischer in 2008, which is summarized in article [5].

α_s (`as`), thermal expansion coefficient in °C^{-1}
rectangular distribution with $R(9.5\times10^{-6}, 13.5\times10^{-6})$

θ_0 (`tq`), average temperature deviation in °C
normal distribution with $N(-0.1, 0.2)$

Δ (`delta`), effect of cyclic temperature deviation in °C
U-shaped distribution with $U(-0.5, 0.5)$

$\delta\alpha$ (`da`), difference in expansion coefficient in °C^{-1}
curvilinear trapezoidal distribution with
$CTrap(-1.0\times10^{-6}, 1.0\times10^{-6}, 0.1\times10^{-6})$

$\delta\theta$ (`dt`), difference in temperatures in °C
curvilinear trapezoidal distribution with
$CTrap(-5.0\times10^{-2}, 5.0\times10^{-2}, 2.5\times10^{-2})$

For a more detailed explanation of the example, e.g., motivations for the selected probability distributions for the influence quantities, I want to refer to the corresponding documents. The example should help to clarify the concepts of the different approaches and allow a comparison of the results. Figure 7.1 shows the histograms for the influence quantities and the dependencies in the example as a tree. The equation for the measurand of the output quantity, the actual length, L is given as

$$L = \frac{L_s\left[1+\alpha_s(\theta-\delta\theta)\right]+d}{1+(\alpha_s+\delta\alpha)\,\theta}, \tag{7.1.1}$$

where the influence quantity d (`d`) describes the difference in the lengths of the gauge block being calibrated and the reference standard and is defined as

$$d = D + d_1 + d_2.$$

Influence quantity θ (`t`) is the deviation of the temperature from 20 °C of the gauge block and is defined as

$$\theta = \theta_0 + \Delta.$$

We can calculate the deviation δL (`l`) of quantity L from the nominal length $L_{nom} = 50$ mm simply as

$$\delta L = L - L_{nom}.$$

To model this measurement scenario in the modeling language a single basic model realizing equation (7.1.1) and a simple simulation definition suffice. The complete XML definitions are provided in Appendix B.2. We will concentrate on the simulation definition and especially the calculation section, because this is where the sensitivity analysis can be defined. In the following different approaches to sensitivity analysis will be explained and applied to this example of a gauge block calibration.

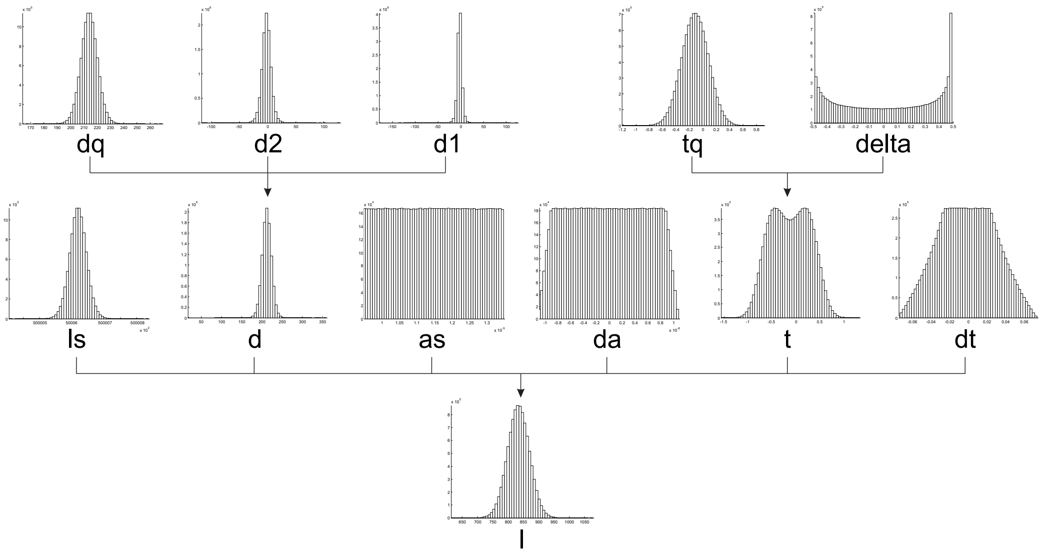


Figure 7.1: Dependencies and histograms of influence quantities in example of gauge block calibration; simulation was performed with 10^7 trials.

7.2 Local SA: Sensitivity coefficients of GUF

The advantage of local sensitivity analysis is that the computational effort is much less than for factor screening or global sensitivity analysis as the method for local SA uses analytical methods and partial derivatives. The classical GUM uncertainty framework can be interpreted as such a method. It linearizes the model and reduces input quantities to their first two moments to evaluate the measurement uncertainty. We initially have the equation of the measurand Y defined as

$$Y = f(X_1, \ldots, X_N).$$

The equation for the combined uncertainty $u(y)$ is then

$$u^2(y) = \sum_{i=1}^{N} c_i^2 u^2(x_i),$$

where the partial derivatives $c_i = \partial f / \partial x_i$ are taken as sensitivity coefficients in the GUM [37, Section 5.1.3] as we have already seen in Section 2.3.1. The coefficients indicate how the output estimate y varies with changes in the value of the input estimates $x_1, ..., x_N$. To have an explicit number that describes the difference in the result because of an influence quantity, the so-called *coefficient of contribution* [41] can be defined as

$$w_i = \frac{u_i^2(x_i)}{u^2(y)} = \frac{c_i^2 u^2(x_i)}{\sum_{j=1}^{N} c_j^2 u^2(x_j)}$$

under the assumption that the input quantities are mutually independent, which is the case for our example. It describes the relation of an influence quantity X_i to the measurand Y.

The first line in Figure 7.7 on page 170 shows bar plots for the gauge block calibration example in comparison with approaches that will be introduced in the next sections. One thing to notice is that due to linearization the influence quantity `t`, the deviation of the temperature from 20 °C of the gauge block being calibrated, has no effect on the combined measurement uncertainty.

7.3 Direct approach: factor screening

In measurement uncertainty evaluation the question arises how much effect an influence quantity has on the combined measurement uncertainty. This allows making a decision on which influence quantities have to be analyzed in more detail or where an enhancement of measurement equipment would pay off. With the Monte Carlo method for the evaluation of measurement uncertainty there exists a very direct and intuitive approach for answering this question using so-called factor screening. Instead of sampling from the probability distribution of an influence quantity, defined quantities return only a specified value, like the mean value, median, or another suitable statistical parameter, on request. There are two ways to apply this method to a measurement scenario. If the behavior of the probability distribution to the model equations is of interest the *one factor at a time* (OAT) approach can be applied, where only the influence quantity of interest generates random values whereas all other values return their specified values. We call the second approach *turn one off* (TOO) as all influences quantities except the one of interest generate random values on request. A measure of the contribution of an influence quantity to the combined standard uncertainty is the difference of an initial simulation with the result of a simulation with constant values. In the following we introduce and compare the two methods OAT and TOO using the gauge block calibration example.

7.3.1 OAT - One factor at a time

A first very direct approach to determine the importance of an influence quantity to the measurement uncertainty is to analyze how the probability distribution of a specific influence quantity propagates through the model equation. The idea is that only a single influence quantity generates random values for such an analysis. All other influence quantities are turned off; say they deliver a constant value such as the mean value or any other statistical parameter on each request. This method is recommended in a similar form in [31, Section 5.8] and is also mentioned in

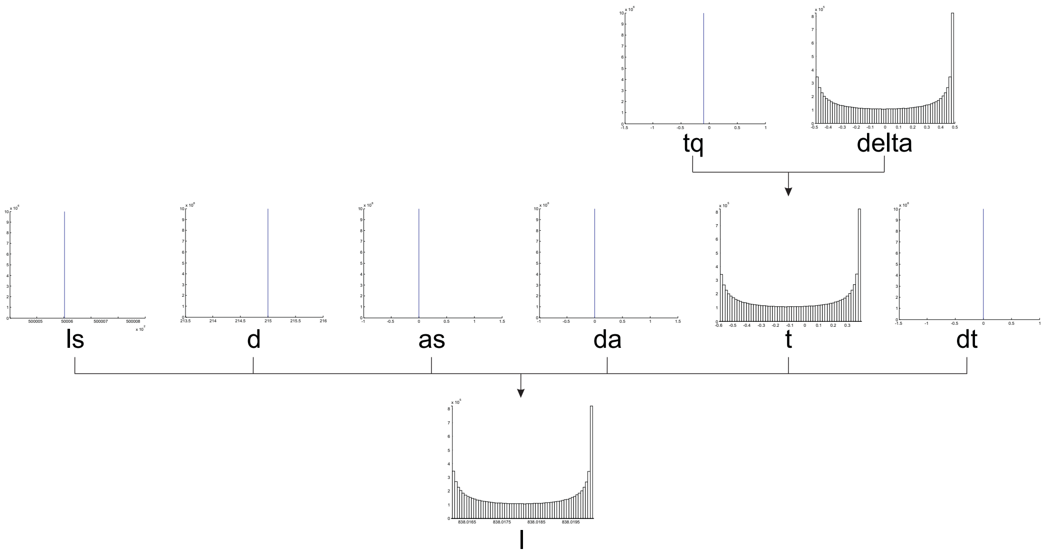


Figure 7.2: Histograms of influence quantities for OAT approach with focus on influence quantity `delta`; only this single influence quantity generates random values for evaluation.

Annex B of GS1 as it delivers something comparable to the sensitivity coefficient of the classical GUF approach and may even give information about the linearity of the model if compared to the classical sensitivity coefficients of GUF.

In a first run the method calculates the measurement uncertainty for the measurement scenario as usual by generating random values for all influence quantities. Internally, each influence quantity updates its mean value, standard deviation as well as other statistical parameters on the fly. There are algorithms to update these parameters for new values iteratively as we have already seen in Section 2.4.1. Starting with a new simulation for each influence quantity of interest, only the current item generates random numbers whereas all other influence quantities use their specified value for calculation. Figure 7.2 illustrates this concept again in a tree-like overview for the influence quantity `delta`, an effect of cyclic temperature variation. The probability distribution of the influence quantity is propagated through the equations and – in this case – does even conserve the shape of the distribution in the final result.

To define such an uncertainty analysis in the domain-specific language one additional line in the calculation section of the simulation definition suffices.

```
<simulation>
  ...
  <calculation>
    <sensitivity mode="all" type="OAT"/>
    <measurand>length</measurand>
  </calculation>
</simulation>
```

The attribute `mode` of the new element `sensitivity` in the calculation section tells the system if all influence quantities should be turned off one after another[2] or if a set of explicitly defined influence quantities is provided as we will see in a later example.

For comparison with the absolute value of “linear” sensitivity coefficients $c_i = \partial f/\partial x_i$ for an influence quantity X_i the “non-linear” coefficient [15, Appendix D]

$$\hat{c}_i = \frac{\hat{u}_i(y)}{u(x_i)}$$

can be used. The term $\hat{u}_i(y)$ is the resulting uncertainty of a Monte Carlo simulation, where only the influence quantity X_i is varied while the other quantities X_j, $j = 1,\ldots,N$ and $j \neq i$, are set to their best estimates. The standard uncertainty $u(x_i)$ of influence quantity X_i can be obtained also by Monte Carlo simulation. When $\hat{c}_i$ is significantly different from c_i, the non-linearity effect may noticeable influence the standard uncertainty $u(y)$.

Figure 7.3 shows the standard deviations and the relative difference to the combined standard uncertainty if OAT is applied to the example of the gauge block calibration. In Figure 7.7 on page 170 the second line shows the bar plots for the OAT approach. The similarity to the GUF sensitivity coefficients is obvious and in this example delivers practically the same result.

A major drawback of this approach is that it only shows a transformation of the randomly drawn values of a single probability distribution according to the equations of the measurement model. It does not preserve any dependencies of other influence quantities, but reduces information rigorously. Thus, we recommend application only for comparison with classical GUF sensitivity coefficients or in combination with other methods for decision making. In general, it is more interesting to analyze dependencies of influence quantities than the outcome of one single probability distribution after propagation through the equations of the measurand. Hence, the next approach tries to keep the simulation as similar as possible to the original one for detection of the impact of one single influence quantity on the uncertainty.

[2]Only single quantities, but no combinations of quantities, are considered. As the modeling language provides constructs for more complex quantities, not only influence quantities are marked for analysis, but also instances of basic models as well as processes.

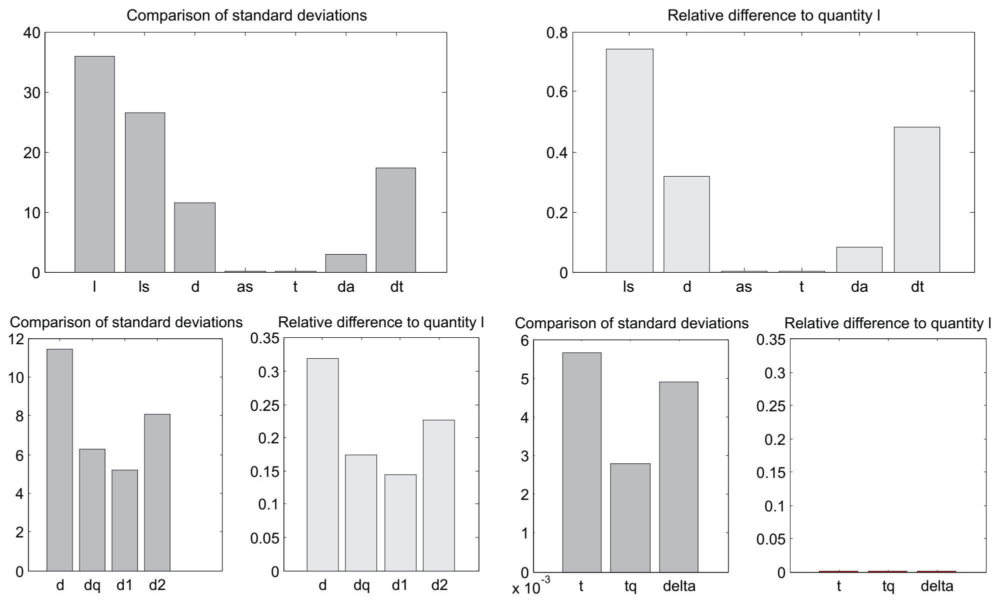


Figure 7.3: Standard deviation and relative difference to combined standard uncertainty if OAT approach is applied for the example of a gauge block calibration.

7.3.2 TOO: Turn one off

We now want to have a look on the measurement scenario assuming that we can optimize individual influence quantities until there is no deviation left. Hence, the quantity can be described using a constant value. The approach reflects what one would try to do in a laboratory reducing the deviation of an individual uncertainty source to a minimum and investigating its importance to the combined measurement uncertainty. This approach is contrary to OAT from the last section. We do not turn off all influence quantities except for one, but we turn only this one influence quantity of interest off. Hence, the TOO approach delivers a measure for what happens if information about a single influence quantity is neglected. In an initial run of the simulation system the statistical parameters to set influence quantities to specific values are calculated. In the subsequent runs all influence quantities generate random values on request except for the one of interest. The definition in the language is the same as for the OAT method except that the value of the attribute `type` changes to `TOO`.

```
    <sensitivity mode="all" type="TOO"/>
```

Figure 7.4 visualizes the concept, again with focus on the influence quantity `delta` in the upper right plot. All influence quantities draw random values from their original distributions except for the one influence quantity `delta` which delivers the same value on each request. The approach allows us to see the effect

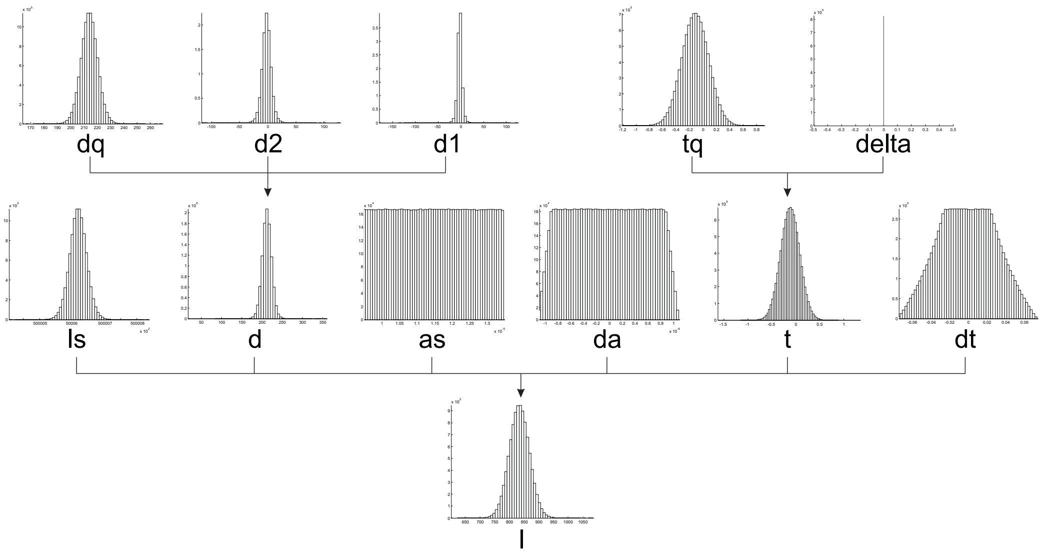


Figure 7.4: Histograms of influence quantities for TOO approach with focus on influence quantity *Delta*; all influence quantities except for *Delta* generate random values for evaluation.

of this one influence quantity on the measurement uncertainty of the measurement scenario. For example, the resulting distribution of influence quantity `t` is no longer double peaked as previously in Figure 7.1 on page 155, but can better be approximated to a normal distribution. The big advantage is that all dependencies of the other influence quantities are preserved; however, the disadvantage is, of course, that the approach requires a greater computational effort than the OAT approach. The number of evaluations of the equations keeps the same, but the number of random number draws is significantly higher, especially for more complex measurement scenarios with many influence quantities. Analysis of each single influence quantity needs a nearly complete additional simulation. However, the effort pays off as a lot of additional information can be extracted from the result. It shows the overall effect of an influence quantity to the combined measurement uncertainty.

Figure 7.5 shows the result if TOO is applied to the example of the gauge block calibration. The most important influence quantities are still `ls`, the length of the reference standard, and `dt`, the difference in temperatures of the gauge block and the reference standard, as already found with the OAT approach, but the effect of influence quantity `t`, the deviation of the temperature of the gauge block being calibrated, seems to be underestimated in the previous two approaches. Applying the TOO approach shows that `t` seems to contribute about as much to the combined standard uncertainty as influence quantities `d` and `da`.

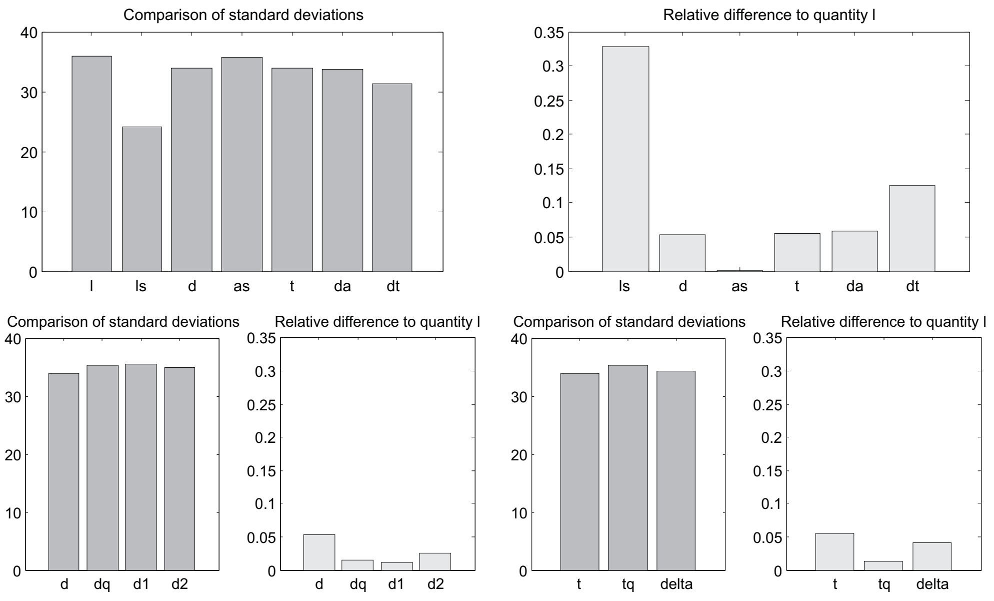


Figure 7.5: Resulting standard deviations of TOO approach and relative differences to the combined standard uncertainty for the example of a gauge block calibration

7.3.3 Critical examination of factor screening

The presented approaches are very straightforward and easy to understand and to implement. As it often seems to be the case, these methods have some restrictions and limitations. The approaches are only applicable to models where the evaluation is fast enough. Let us assume we perform M Monte Carlo runs per simulation. The equation of the measurand has to be evaluated in a first simulation M times to get the statistical parameters needed to calculate the nominal value for n different influence quantities for the following simulations. Then the equation of the measurand has to be evaluated for each influence quantity described by a distribution another M times in an additional simulation to obtain the new measurement uncertainty. All in all, we have to evaluate the equations $M(n+1)$ times, even if we optimize the simulation system and reuse already generated random values. The number n increases if we want to analyze partial results from the equation of the measurand.

The second restriction is that the analysis is very local as we have currently only talked about holding the influence quantities to one specific value. This may be sufficient for a first and very basic understanding of the measurement scenario or a measurement scenario that is almost linear. Important factors can be determined only to a certain level of assurance. If we take, for example, the mean value to fix influence quantities, this may not be the most probable value for each probability distribution under investigation. It may even appear that this value is out of the definition range of the influence quantity. This can easily be seen if a Bernoulli distribution delivering a 0 in 50 % of the cases and 1 for the other 50 % for an influence quantity is assumed. The probability for the long-term mean value 0.5 is zero. A way to enhance the presented procedures would be to do the analysis also for other statistical parameters, like median, mode or even the uncertainty intervals. Anyhow, this would increase the number of necessary simulations and trials and thus, the computational effort significantly.

Finally, the approach may be misleading if correlations between influence quantities appear in the model. As correlated influence quantities may share[3] a probability distribution it is not clear how to evaluate correlated influence quantities. This shortcoming can be handled with an additional functionality that will be introduced in the next section. It allows the definition of sets of influence quantities that can then be analyzed in combination. Currently, it is the only way for proper and intuitive handling of correlated influence quantities.

As the limitations seem to be very restrictive, the practical aspects are far more promising. We want to analyze the behavior of the measurement scenario corresponding to specific influence quantities of the measurement. Hence, it is justifiable to work with – at least initially – rather small numbers M of trials for a simulation and refine results using a bigger number of trials on specific influence quantities that are found to be of interest. The approaches are applicable to analyze the system and extract the influence quantities with the biggest leverage effect. Metrologist can get a feeling for the measurement scenario and observe dependencies and the impact of influence quantities to the overall measurement uncertainty in great detail.

7.3.4 Advanced factor screening approaches

In the last two sections the concepts of two factor screening methods for sensitivity analysis were introduced together with the corresponding language enhancements. The critical examination showed some limitations, but also that the approaches are applicable in daily work. Now we will concentrate on more sophisticated aspects that are promising, because they allow a very detailed analysis of measurement scenarios and dependencies of influence quantities. We use the two approaches OAT and TOO and the same example of the gauge block calibration to explain the relevant issues.

[3]Sharing a distribution in the sense that generated random numbers are used to generate subsequent random numbers for other influence quantities appearing at different places in the model.

Analyzing combinations of influences To analyze the dependencies of influence quantities in the measurement uncertainty analysis it is of major importance to be able to define not only single influence quantities, but also sets of influence quantities. This allows, for example, to analyze sets of correlated influence quantities that have to be analyzed in combination. As Monte Carlo simulations are in general very time consuming and the goal is to handle advanced and high-resolution, real-world models with many influence quantities, it is not feasible to compute all permutations of influence quantities for the analysis per default. Therefore, we rely on information of important sets of influence quantities. So, if sets of influence quantities should be analyzed in combination they have to be defined in the sensitivity section explicitly.

```
<simulation>
  ...
  <calculation>
    <sensitivity mode="set" type="TOO">
      <sensitivityset>
        <sensitivityitem influencename="da"/>
        <sensitivityitem influencename="dt"/>
      </sensitivityset>
      <sensitivityset>
        <sensitivityitem influencename="delta"/>
        <sensitivityitem influencename="dt"/>
      </sensitivityset>
    </sensitivity>
    <measurand>length</measurand>
  </calculation>
</simulation>
```

In this code fragment of the simulation definition, the influence quantities `da` and `dt` are analyzed together in a second simulation and afterwards the combination of the influence quantities `delta` and `dt`. Besides, processes and instances can be used for the definition of sensitivity sets with attributes `processname` and `instancename` of the element `sensitivityitem`. Using the attributes in combination allows analyzing specific appearances of instances or influence quantities in processes and instances.

Fade out to a certain degree In reality it is not possible to reduce the deviation of an influence quantity arbitrarily or let it vanish completely. Hence, sensitivity analysis of influence quantities that are turned off completely is not always satisfying. On the other hand, it is not clear how to reduce the deviation of an influence quantity in general, because there is no general method for arbitrary probability distributions or even combinations of several influence quantities. However, our simulation system calculates the mean value and the standard deviation – say standard uncertainty – for every input quantity in the initial run of the OAT and TOO approach. Thus, we are able to reduce the standard uncertainty to a certain degree if we approximate each influence quantity using a normal distribution. This is an important and major restriction of this approach, because we lose information about the shape of the distribution. To avoid drawing wrong conclusions one should always apply a classical OAT or TOO simulation previously and compare it to the result of reduced uncertainty in an influence quantity afterwards. Only then can one be sure that the approximation using normal distributions can be justified and prevent drawing wrong conclusions.

```
<simulation>
  ...
  <calculation>
    <sensitivity mode="all" type="TOO" stdfactor="0.3"/>
    <measurand>length</measurand>
  </calculation>
</simulation>
```

In the code fragment we reduce the standard deviation of all influence quantities one after another to 30 % of the evaluated standard deviation of the initial run. Figure 7.6 shows a comparison of fading out the standard deviation of each influence quantity of the gauge block calibration example continuously in steps of 20 %. The factor screening method TOO is used in this case for sensitivity analysis. The result shows once more the importance of the influence quantity `ls` to the measurement. Nevertheless, it may pay off to investigate combinations of reduced uncertainties for different influence quantities as it can be cheaper to reduce the uncertainty of two devices to some lesser extent than to focus on one quantity and invest in very expensive measurement equipment.

7.4 Global SA: a variance-based method

Besides the very direct approaches reducing influence quantities to specific values for factor screening or looking at a linearized model with limited information as the local sensitivity analysis does, there exist other techniques using all available information. In our application we are especially interested in the measurement uncertainty, which we estimate using the Monte Carlo method as the standard deviation of the values from evaluating the equation of the measurand M times. Therefore, we will concentrate on variance-based methods; say on methods focusing on the squared standard deviation. An example for such an approach is the method of Sobol' calculating sensitivity coefficients for influence quantities. Additionally, a variation of this approach will be introduced that allows investigating the total effect of an influence quantity to the measurement uncertainty using

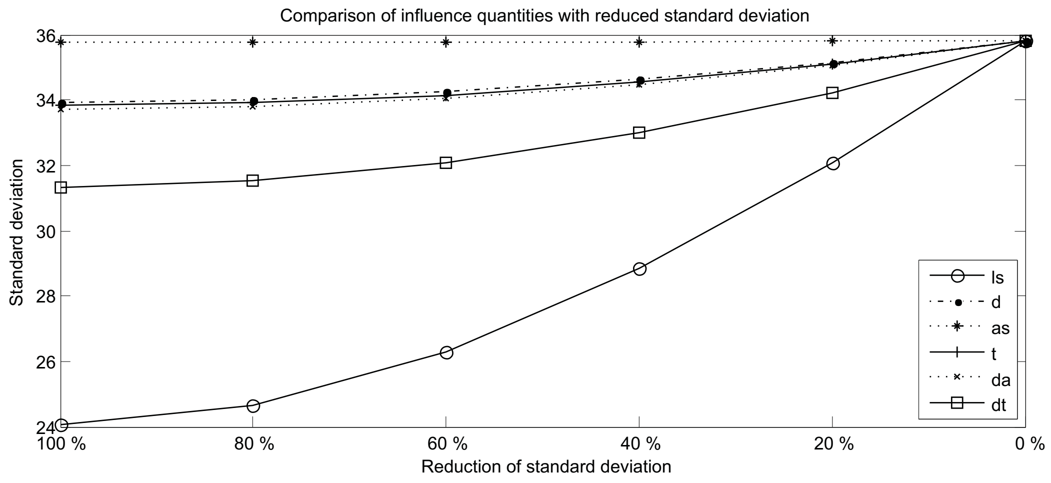


Figure 7.6: Comparison of standard uncertainty reduction of measurand if standard deviation of influence quantities is reduced applying TOO approach

less computational power than the original method requires. The key idea of the methods is quite similar to the one from the factor screening approaches as we will see.

7.4.1 Sobol' sensitivity index

First I want to give a short summary of the theoretical background of Sobol' sensitivity analysis, keeping close to the definitions of the three articles [35,68,72]. To describe Sobol' sensitivity estimate we examine the function $f(\mathbf{x}) = f(x_1,\dots,x_N)$ with

$$K^N = \{\mathbf{x}|0 \le x_i \le 1; i = 1,\dots,N\}$$

as an N-dimensional unit cube. The method of Sobol' relies then on the decomposition

$$f(x_1,\dots,x_N) = f_0 + \sum_{i=1}^{N} f_i(x_i) + \sum_{1\le i<j\le N} f_{ij}(x_i,x_j) + \dots + f_{1,\dots,N}(x_1,\dots,x_N). \tag{7.4.1}$$

Equation 7.4.1 requires that f_0 is constant and the integrals of every summand over any of its own variables must be zero, i.e.

$$\int_0^1 f_{i_1,\dots,i_s}(x_{i_1},\dots,x_{i_s})dx_{i_k} = 0 \quad \text{with } 1 \le k \le s.$$

A consequence is that the terms $f_{i_1,\dots,i_s}(x_{i_1},\dots,x_{i_s})$ are orthogonal, say

$$\int_{K^N} f_{i_1,\dots,i_s}(x_{i_1},\dots,x_{i_s}) f_{j_1,\dots,j_l}(x_{j_1},\dots,x_{j_l}) d\mathbf{x} = 0, \quad (i_1,\dots,i_s) \neq (j_1,\dots,j_l). \tag{7.4.2}$$

Sobol' shows in [72] that the decomposition is unique whenever $f(\mathbf{x})$ is integrable over K^N.

Now we can look at the variance D of $f(x_1,\dots,x_N)$ which is defined as

$$D = \int_{K^N} f^2(\mathbf{x}) d\mathbf{x} - f_0^2.$$

Partial variances can be computed from each of the terms in Equation 7.4.1 as

$$D_{i_1,\dots,i_s} = \int_0^1 \cdots \int_0^1 f^2_{i_1,\dots,i_s}(x_{i_1},\dots,x_{i_s}) dx_{i_1} \dots dx_{i_s}$$

with $1 \le i_1 < \cdots < i_s \le N$ and $s = 1,\dots,N$. If we square and integrate Equation 7.4.1 over K^N and consider Equation 7.4.2, we obtain

$$D = \sum_{i=1}^{N} D_i + \sum_{1 \le i < j \le N} D_{ij} + \ldots + D_{1,\dots,N}.$$

Using this concepts, the sensitivity estimates $S_{i_1,\dots,i_s}$ are defined as

$$S_{i_1,\dots,i_s} = \frac{D_{i_1,\dots,i_s}}{D}.$$

Example To show how the decomposition into summands and the calculation of the sensitivity estimates works, we apply the method of Sobol' [72] to function

$$f(x_1,x_2) = c_0 + c_1 x_1 + c_2 x_2.$$

For the decomposition, we are looking for a representation

$$f(x_1,x_2) = f_0 + f_1(x_1) + f_2(x_2) + f_{1,2}(x_1,x_2).$$

According to Sobol', the first term f_0 of the decomposition is

$$f_0 = \int_{K^N} f(x_1,x_2)\, d\mathbf{x} = \int_0^1 \int_0^1 c_0 + c_1 x_1 + c_2 x_2 \, dx_1 dx_2 = c_0 + \frac{c_1}{2} + \frac{c_2}{2}.$$

Let $d\mathbf{x}_{\sim x_i}$ be the product $dx_1 \cdots dx_N$ except for dx_i with $1 \le i \le N$. According to Sobol' we can use equations

$$g_i(x_i) = \int_0^1 \cdots \int_0^1 f(x_1,\dots,x_N) d\mathbf{x}_{\sim x_i}$$

and

$$g_i(x_i) = f_i(x_i) + f_0$$

to calculate the one-index terms $f_1(x_1)$ and $f_2(x_2)$ for the example using

$$g_1(x_1) = \int_0^1 f(x_1,x_2)dx_2 = c_0 + c_1x_1 + \frac{c_2}{2}$$

and

$$g_2(x_2) = \int_0^1 f(x_1,x_2)dx_1 = c_0 + \frac{c_1}{2} + c_2x_2.$$

This leads to

$$f_1(x_1) = g_1(x_1) - f_0 = c_1x_1 - \frac{c_1}{2},$$

respectively

$$f_2(x_2) = g_2(x_2) - f_0 = c_2x_2 - \frac{c_2}{2}.$$

For the two-index term $f_{1,2}(x_1,x_2)$ we can use formula

$$g_{i,j}(x_i,x_j) = \int_0^1 \cdots \int_0^1 f(x_1,\ldots,x_N)d\mathbf{x}_{\sim x_i,x_j},$$

where $d\mathbf{x}_{\sim x_i,x_j}$ is the product $dx_1 \cdots dx_N$ except for dx_i and dx_j. Further, we use

$$g_{i,j}(x_i,x_j) = f_{i,j}(x_i,x_j) - \sum_{i=1}^{N} f_i(x_i) - f_0.$$

This leads to

$$g_{1,2}(x_1,x_2) = f(x_1,x_2) = c_0 + c_1x_1 + c_2x_2$$

and allows calculating the last missing term

$$f_{1,2}(x_1,x_2) = g_{1,2}(x_1,x_2) - f_1(x_1) - f_2(x_2) - f_0 = 0$$

Now we have all terms for the decomposition of $f(x_1,x_2)$ and can calculate the sensitivity measures. First, we have to evaluate the variance

$$D = \int_0^1 \int_0^1 f^2(x_1,x_2)dx_1dx_2 - f_0^2 = \frac{1}{12}c_1 + \frac{1}{12}c_2.$$

The one-indexed partial variances are

$$D_1 = \int_0^1 f_1(x_1)dx_1 = \frac{1}{12}c_1^2 \quad \text{and} \quad D_2 = \int_0^1 f_2(x_2)dx_2 = \frac{1}{12}c_2^2.$$

The corresponding sensitivity measures are then defined as

$$S_1 = \frac{D_1}{D} = \frac{c_1^2}{c_1^2 + c_2^2} \quad \text{and} \quad S_2 = \frac{D_2}{D} = \frac{c_2^2}{c_1^2 + c_2^2}.$$

Because $f_{1,2}(x_1,x_2)$ is zero, the two-indexed partial variance $D_{1,2}$ and also its sensitivity measure $S_{1,2}$ are zero.

Computational issues Now, in practice we use Monte Carlo simulation with M trials to get a large population of possible values for the input quantities and we can calculate an estimation $\hat{f}_0$ for f_0

$$\hat{f}_0 = \frac{1}{M}\sum_{k=1}^{M} f(\mathbf{x}_k)$$

where $\mathbf{x}_k$ is one of M random draws for all input quantities. Variance D can be estimated as

$$\widehat{D} = \frac{1}{M}\sum_{k=1}^{M} f^2(\mathbf{x}_k) - \hat{f}_0^2$$

using another M random draws $\mathbf{x}_k$ for the input quantities. The estimators for first-order terms $\widehat{D}_i$ are calculated using equation

$$\widehat{D}_i = \frac{1}{M}\sum_{k=1}^{M} f(\mathbf{u}_k, x_{ik}) f(\mathbf{v}_k, x_{ik}) - \hat{f}_0^2,$$

where $\mathbf{u}_k$ and $\mathbf{v}_k$ are two independent draws for all influence quantities except the influence quantity x_i. A value x_{ik} for the quantity x_i is only drawn once and remains the same for both evaluations of the function f.

For estimating second-order terms the equation

$$\widehat{D}_{ij} = \frac{1}{M}\sum_{k=1}^{M} f(\mathbf{r}_k, x_{ik}, x_{jk}) f(\mathbf{s}_k, x_{ik}, x_{jk}) - \hat{f}_0^2 - \widehat{D}_i - \widehat{D}_j$$

is used, where $\mathbf{r}_k$ and $\mathbf{s}_k$ are again draws for all quantities, but this time without the two quantities x_i and x_j. For x_i and x_j the same values x_{ik} and x_{jk} are used for one trial. This schema can then be extended and continued for all higher-order terms recursively. The estimated sensitivity measures are then

$$\widehat{S}_{i_1,\ldots,i_s} = \frac{\widehat{D}_{i_1,\ldots,i_s}}{\widehat{D}}.$$

The problem with this approach is that the number of simulations is 2^N with M trials each for all possible permutations considering all higher-order terms in evaluation, where N is the number of influence quantities. So it is only applicable for terms of limited order or with a very low number of trials per simulation. As stated in [35], higher-order terms are often neglected in sensitivity analysis and usually only the first-order terms are explored. Because of computational reasons this is also true for our system; only first-order terms are evaluated. To define an analysis using Sobol' sensitivity indices again it is sufficient to change line 4 in the code fragment of the factor screening scenario.

```
    <sensitivity mode="all" type="sobol"/>
```

Figure 7.7 shows Sobol' first-order sensitivity indices in the fourth line. The result is very close to the GUF and OAT approach as one expects because of the limitation to first-order dependencies.

7.4.2 Total Effect Indices

A different approach is to compute the so-called *total sensitivity index* (TSI) for each influence quantity according to Homma and Saltelli [35] which is based on the sensitivity indices of Sobol'. It varies not all influence quantities for each evaluation of the function f, but only the one of interest. It takes all terms of the decomposition of function f into account, where the influence quantity x_i appears, and results in equation

$$\widehat{D}_{\sim i} = \frac{1}{M}\sum_{k=1}^{M} f(\mathbf{r}_k, x_{ik}) f(\mathbf{r}_k, x'_{ik}) - \hat{f}_0^2,$$

where $\mathbf{r}_k$ is a single draw of random numbers for all influence quantities except for x_i in the k-th trial. The quantity x_i uses now two draws of random numbers, x_{ik} and x'_{ik}. An estimation for the total sensitivity index is then

$$\widehat{TS}(i) = 1 - \frac{\widehat{D}_{\sim i}}{\widehat{D}}.$$

This measure is a summary considering all effects of an influence quantity x_i to the total output variation. The method requires only $N+1$ simulations, one for each influence quantity and an additional simulation for the initial estimation of $\hat{f}_0$. A drawback is that this method does not describe the system in detail, but only the overall effect and contribution of each influence quantity to the result. Nevertheless, the advantage of a significantly smaller number of trials as well as simulations prevails. For a more detailed analysis of the importance or impact of groups of influence quantities to the measurement uncertainty, the other presented methods can be applied in addition.

In the simulation definition we use the abbreviation TSI to define an analysis with total sensitivity indices.

```
    <sensitivity mode="all" type="TSI"/>
```

Figure 7.7 shows the total sensitivity indices for the gauge block calibration example in the fifth and last line. In comparison to the other approaches the influence quantity `t` seems to be significantly underestimated by the GUF and OAT approach as well as by the first-order Sobol' sensitivity indices.

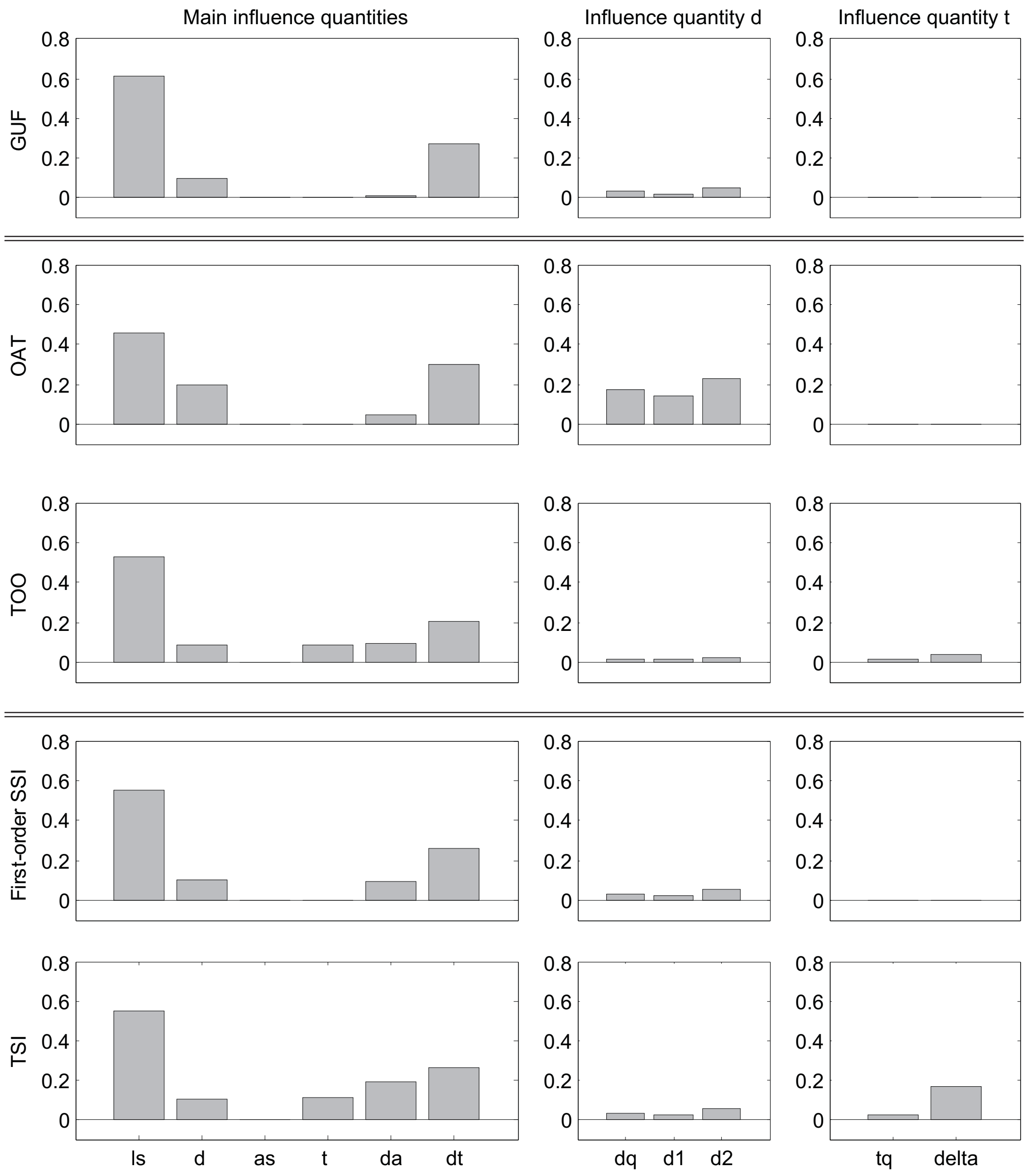


Figure 7.7: Comparison of the different sensitivity analysis approaches, simulation was performed with 10^7 trials. The GUF-based coefficient of contribution, OAT and SSI seem to underestimate influence quantity `t`, because if considering all effects, as methods TOO and TSI do, quantity `t` is approximately as important as `d`.

7.5 Combine factor screening and global SA

The factor screening method OAT and the first-order Sobol' sensitivity indices are similar as only the influence quantities of interest are varied; in the same way the TOO approach is similar to the total sensitivity index as only the influence quantities of interest are neglected in evaluation instead of all others preserving dependencies and effects of other combinations of influence quantities. Hence, in combination the four methods allow a detailed analysis of the measurement system. The factor screening methods focus on the reduction of deviation if fixing influence quantities to specific values, whereas the global sensitivity analysis methods concentrate on the contribution to the variance of individual influence quantities.

In combination the presented methods for factor screening and global sensitivity analysis are a mighty tool to analyze the model of measurement scenarios in great detail. As an efficient Monte Carlo simulation system is provided, the methods make extensive usage of its capabilities. I think it is also of great importance that not only a single method is available, but a set of different approaches. A single method might be misleading in daily work if it were the one method of choice in every case. Providing different approaches requires a deeper knowledge and understanding in practice, but is rewarded with a lot of additional information and a more efficient analysis as specific parts of interest get into focus. The system is also implemented in a modular fashion so that new methods can be added if the provided methods are not sufficient for a specific problem. The results underline once more that the framework with the modeling language is a helpful what-if machine that can support experts in decision making.

CHAPTER 8

Additional modeling issues

Some examples can be regarded as typical of those that arise in metrology. Others are more extreme, in an attempt to indicate the considerations that are necessary when those of "normal circumstances" fail to apply. Perhaps, unfortunately, such adverse circumstances arise more frequently then would be wished, ...

— SSFM Best Practice Guide No. 6, 2004, Page 85 —

Building the exhaustive model of a titration measurement from the most simple measurement item to a complete scenario for a series of measurements showed that the specified language is capable of modeling very complex, real-world measurement scenarios. Besides, the last chapter introduced and compared different methods to extract the most important influence quantities from a given measurement scenario. This helps in decision making and optimizing the measurement setup in respect to the measurement uncertainty. Now, in this chapter we will concentrate on two additional, very specific issues using examples from different sources. They will once again exemplify that the Monte Carlo method is applicable to a broader range of measurement scenarios than the classical GUF approach.

In practice there often appear situations, where measurements are performed near physical limits. Hence, the first issue will be restricting results to a feasible range and investigating the consequences. We will see an example, where sampled values of a probability distribution are restricted explicitly to reflect the physical limits. Afterwards, we will apply limited probability distributions to model a real-world example, where items are classified and sorted in production with respect to defined quality measures.

The second issue is implicit functional relationships, where an influence quantity is embedded in the function. We will see an application in calculating the flow rate in an open channel with uncertainties. Such scenarios require a zero-finding algorithm for evaluation. An adaption of such an algorithm is capable of block-wise evaluation of equations for Monte Carlo simulation.

8.1 Physical limits and feasible ranges

There is a lot of discussion on how to report measurement uncertainty for a measurand near natural limits. In chemical measurements the situation often appears that the symmetrical uncertainty interval overlaps with physically impossible ranges. The *EURACHEM/CITAG Guide* [19] details the problem in Appendix F and states that measurement uncertainty should be reported in any case, even where the result implies an impossible physical situation. In article [14] of Cowen and Ellison different approaches for reporting are analyzed and compared. One approach is to use Bayesian statistics as also proposed by [76]; other approaches are shifting values outside of the feasible range to the limits, truncation of the confidence interval, or data censoring. This leads to further implications as statistical parameters like mean and median are affected and shifted. *Best Practice Guide No. 6* [15, Section 9.5] of the NPL picks up the problem, too. The Monte Carlo method is mentioned as a viable alternative for reporting measurements near natural limits as not only the symmetrical, but also the shortest coverage interval, can be reported. This seems to be an appropriate approach for describing the results of such measurement scenarios properly. Anyhow, currently there is no consensus about reporting measurement uncertainty in such cases and the decision about reporting has to be made by experts in the laboratory from case to case. We don't focus directly on the problem of reporting measurement uncertainty, but we will investigate how a measurement uncertainty with a reported feasible region can be modeled for further measurement uncertainty evaluations in a new measurement scenario. This allows experts analyzing of the consequences of such a measurement uncertainty statement using additional information about its feasible range and will support decision making.

We have built a model for a highly pure substance in the example of a titration measurement in Chapter 4.2.1. There we have seen that physically senseless random values may be drawn during evaluation of the measurement uncertainty, reporting a purity of more than 100 %. Nevertheless, we could argue for the titration measurement to consider sampled values above the 100 % limit in that special case because of the possibility of impurities in the solution. Hence, no special care was required. Another influence quantity with a physical limit for values has been used in the definition of the burette device in Section 4.3.3, restricting the age of the burette to a logical minimum value of zero. In this chapter we want to investigate the consequences of such modeling in greater detail.

8.1.1 Feasible ranges for probability distributions

We start with the most direct approach. It is the restriction of the sampled values from a specific probability distribution to a predefined feasible interval. I presented and discussed a comparison of different methods to restrict values to a feasible range in [82]. The approach seems to be trivial for rectangular probability distributions as the generated random values innately lay between the lower and upper limit, but as we will see there are different ways to limit value ranges of probability distributions; this makes a new concept necessary even for uniformly distributed influence quantities. For other distributions it is not obvious offhand how to limit sampled values to a feasible range. To show the consequences of different ways to restrict sampled values we take initially a very simple example from [14], where a normal distribution with parameters $\mu = 0.01$ and $\sigma = 0.01$ is defined with a natural limit of zero. It represents the result from a trace analysis to detect impurities at a very low level. The following code fragment shows a very simple model with a normal distribution restricted to values equal or above zero in this case.

```
1  <model name="limitgauss" targetname="lg">
2    <influence name="lg" comment="limited normal distribution">
3      <distribution>
4        <gausslimit lower="0.0" limitmode="collect">
5          <mu>0.01</mu>
6          <sigma>0.01</sigma>
7        </gausslimit>
8      </distribution>
9    </influence>
10 </model>
```

The simulation framework includes currently the three special probability distributions `gausslimit`, `trianglelimit`, and `rectanglelimit` with the same parameter definition as their unlimited counterparts. Other probability distributions can be added easily using the same concepts of the language that I will present in this section. In addition to the usual parameter definitions, the attributes `lower` and `upper` can be used to define the feasible range as done in line 4 of the code fragment. In the current example we only use a lower limit. The results of three simulations with 10^7 trials each are compared in Figure 8.1 to show the differences of the settings of attribute `limitmode`. It describes how values outside of the valid range should be handled. The upper histogram is a usual normal distribution with no restrictions. It is clear that the coverage interval includes also physically impossible values. The histogram in the middle uses a mode setting `cut`, which means that values outside of the feasible range are resampled until they are inside of the defined range. Finally, the lower histogram shows the result of a mode setting `collect`, where values outside of the given range are mapped to the closest limiting value. In this case, the resulting probability distribution has a high peak at the lower limit. Table 8.1 summarizes the results.

As we do not like to enforce the usage of special kinds of probability distributions and implementing additional probability distributions is a very inconvenient way to support experts from laboratories, there is another method to restrict sampled values to a valid range. We have already seen its application in Section 4.3.3 for the influence quantity that considers aging of the burette device. There we have defined a lower limit in the definition of an influence quantity instead of restricting sampled values of probability distributions directly. The same attributes – `lower` and `upper` – can be used in combination with the attribute `limitmode` with the settings `cut`, `collect` and `fold`. The additional setting `fold` folds values outside of the defined range back into the range by subtracting the limit and the value and adding or subtracting it from the lower or upper limit. With this construct of restricting values for influence quantities it is not only possible to limit the outcome of drawing random numbers from single probability distributions, but also the result of more complex model parts. With evaluation of the following model we obtain the same results as with the previous model definition.

```xml
<model name="limitgauss" targetname="lg">
  <influence name="lg" comment="limited influence"
      lower="0.0" limitmode="collect">
    <distribution>
      <gauss>
        <mu>0.01</mu>
        <sigma>0.01</sigma>
      </gauss>
    </distribution>
  </influence>
</model>
```

It is important to notice that cutting or resampling sampled values may change the statistical parameters like mean, median and standard deviation significantly as discussed in [14]. Thus, modeling using restricted probability distributions or influence quantities requires a very good understanding and insight into model structures, advanced analyzing techniques and careful parameter settings as the parameter definition of these probability distributions still describes not the restricted, but the original probability distributions.

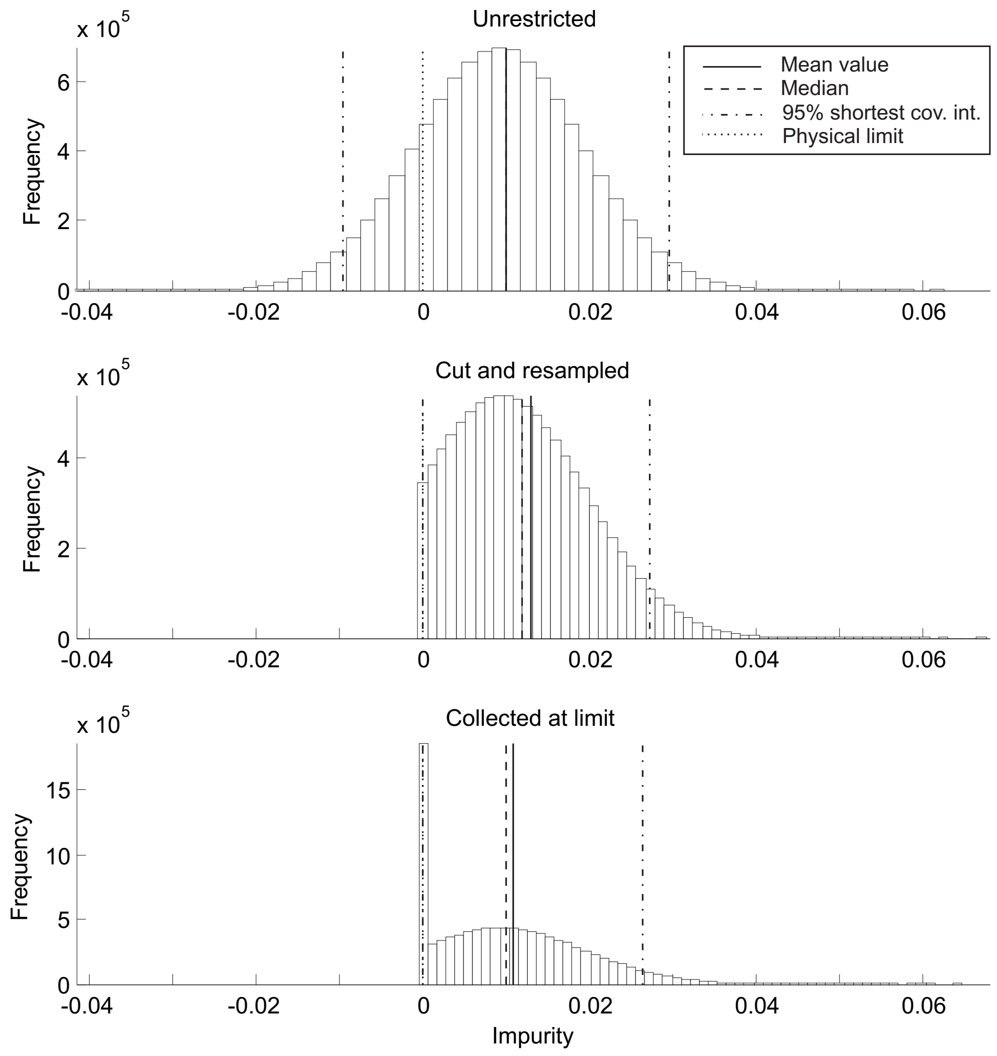


Figure 8.1: Histograms of simulations with 10^7 trials with a normal distribution $N(0.01, 0.01)$ unrestricted, cut at limit, and collected at limit

	Median	**Mean**	**Std.**	**Symmetric 95 %**	**Shortest 95 %**
Unrest.	1.00	1.00	1.00	[-0.96,2.96]	[-0.96,2.95]
Cut	1.20	1.29	0.79	[0.08,3.03]	[0.00,2.73]
Collect	1.00	1.08	0.87	[0.00,2.96]	[0.00,2.64]

Table 8.1: Result for three independent simulations with 10^7 trials each with different behavior (unrestricted, cut, and collect) on natural limit zero of a normal distribution $N(0.01, 0.01)$. The last two columns contain the values of the symmetric and shortest 95 % uncertainty intervals; all values have to be multiplied by 10^{-2}.

Grade	**Range in Ω**
A-grade	[0.99,1.01]
B-grade	[0.95,0.99] or [1.01,1.05]
C-grade	[0.90,0.95] or [1.05,1.10]
Unclassified	all others

Table 8.2: Classes and corresponding value ranges for grading 1 Ω resistors

The modeling language provides all the necessary concepts and hence, the possibility to model and compare different modeling approaches. Most of the current work in the community is related to the question of how to report the measurement uncertainty in such cases, but not how this knowledge should then be used in subsequent measurement uncertainty evaluations. As we allow influence quantities with restricted value ranges as input quantities in measurement models, the consequences of such measurement scenarios including feasible ranges can be further investigated. In the next section we will see that restrictions may even be essential for a proper modeling of specific measurement scenarios.

8.1.2 Quality grading in production

In a second example of the application of limited probability distributions we will see how uncommonly shaped probability distributions can be modeled with existing concepts. The following example about graded resistors is taken from the *Best Practice Guide No. 6* [15, Section 9.2]. A manufacturer produces resistors and reports that a normal distribution with mean 1 Ω and standard deviation 0.04 Ω can be assumed for all produced resistors. The resistors are classified after the production process into four categories A, B, C, or *unclassified.* We suppose that testing of the resistors is fully under control and that uncertainties in the grading process are negligible. Table 8.2 shows the allowed intervals of the four classes for 1 Ω resistors and Figure 8.2 shows the normal distribution segmented to the four classes of resistors.

We now want to analyze circuits using n C-grade resistors connected in series with a nominal resistance of n Ω. The equation of the measurand is in this case just the summation of the individual resistors

$$R = R_1 + \cdots + R_n = \sum_{i=1}^{n} R_i.$$

As already stated in the best practice guide modeling of this problem is straightforward with existing Monte Carlo software. Nevertheless, the modeling aspects concerning our project are quite interesting for this example, as cutting the left and

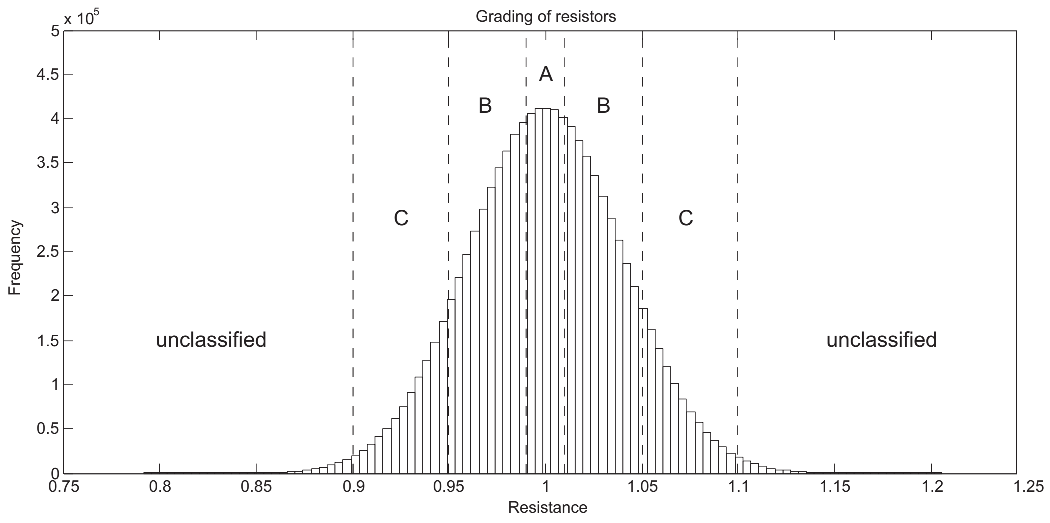


Figure 8.2: The histogram shows the distribution of 1 Ω resistors with a standard deviation of 0.04 Ω (result of a Monte Carlo simulation with 10^7 trials). The four grade classes *A*, *B*, *C*, and *unclassified* are annotated.

right side of normal distributions is not sufficient in this case. Figure 8.3 shows the concept of the model.

```
<model name="resistor" targetname="rc">
 <influence name="rc" comment="class C resistor">
   <!-- Combine distribution from two normals and a
       selection -->
   <formula>s*r1 + (1-s)*r2</formula>

   <influences>
     <influence name="s" comment="Selector">
       <distribution>
         <bernoulli>
           <p>0.5</p>
         </bernoulli>
       </distribution>
     </influence>

     <influence name="r1" comment="resistor lower range"
         unit="ohm">
       <distribution>
         <gausslimit lower="0.90" upper="0.95" mode="cut">
           <mu>1</mu>
           <sigma>0.04</sigma>
         </gausslimit>
       </distribution>
     </influence>
```

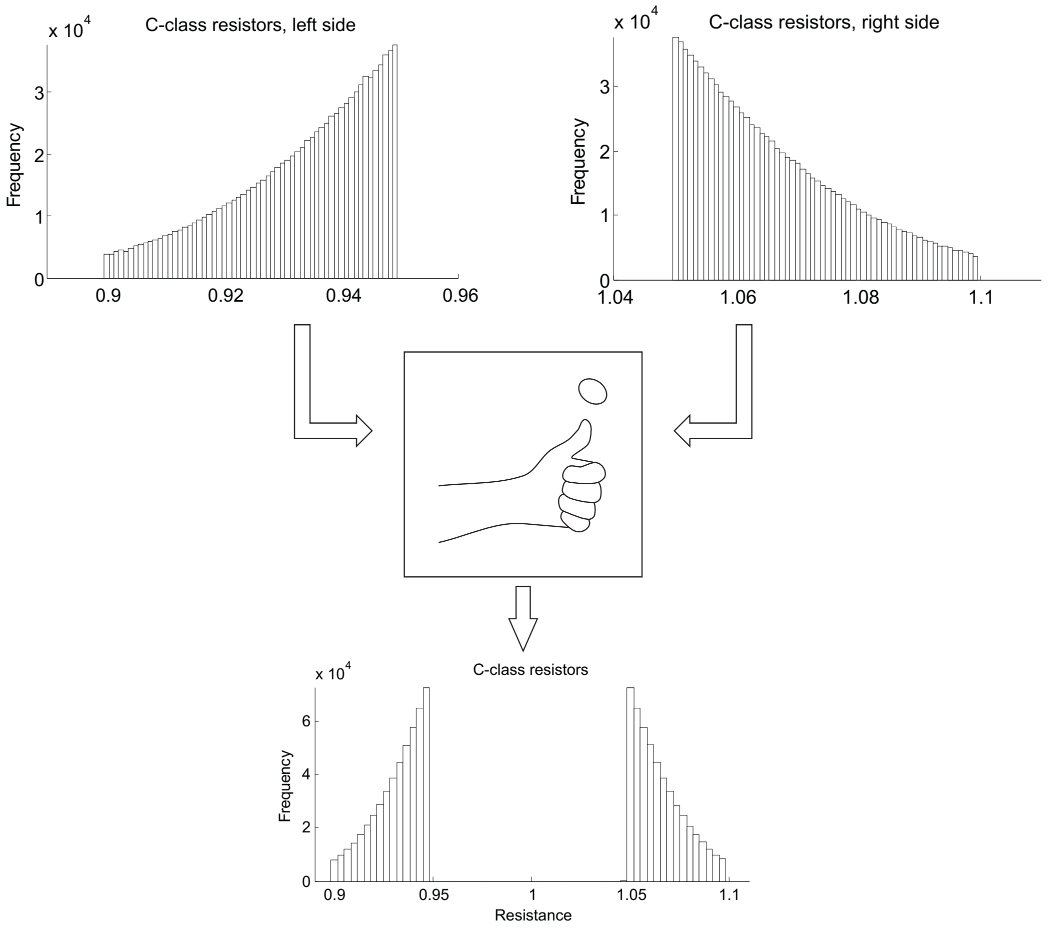


Figure 8.3: To model a class of graded resistors we sample from two limited normal distributions, once for the left part and once for the right part; then we flip a fair coin (Bernoulli experiment) to decide for one of the two sides.

```

      <influence name="r2" comment="resistor upper range"
          unit="ohm">
        <distribution>
          <gausslimit lower="1.05" upper="1.10" mode="cut">
            <mu>1</mu>
            <sigma>0.04</sigma>
          </gausslimit>
        </distribution>
      </influence>
    </influences>
  </influence>
</model>
```

The model and main influence quantity for a resistor uses three random values for one draw:

- Sample a value r_l from lower range of normal distribution $N(1, 0.04)$.
- Sample a value r_u from upper range of normal distribution $N(1, 0.04)$.
- Throw a fair coin and get $s \in \{0, 1\}$ to decide for the lower or upper value.

The decision from the last point is made using a Bernoulli experiment with parameter $p = 0.5$, so that in 50 % of the cases the result is 0, 1 otherwise. The equation for decision is then defined in line 4 of the code fragment as

$$r = sr_l + (1 - s)r_u.$$

Figure 8.4 shows the result of a simulation with 10^7 trials each for circuits of varying numbers of C-grade resistors connected in series. It is interesting that with a small, uneven number n of C-graded resistors the nominal value of n Ω has a small probability. With a rising number of resistors the distribution evolves rapidly to approximately a normal distribution, but still with strong structure.

8.2 Implicit relationship formulas

To finish the line of examples and modeling concepts, we will have a look at a last measurement scenario, where an influence quantity is embedded in a functional relationship. The quantity has to be calculated implicitly. Thus, some additional effort is necessary to evaluate the measurement model. The example of calculating the flow rate in an open channel originated from the National Engineering Laboratory (NEL) in the UK and is again included in the *Best Practice Guide No. 6*[1] [15, Section 9.1]; ISO document 4359-1983 [3] and the book [33] of Reginald Herschy explain the concepts of liquid flow measurements in open channels in great detail. Applications are, for example, the flow rate of water from a river to a cooling system in industrial processes or in water and hydroelectric power industries. We concentrate on the most common type of open channels with a rectangular throated flume. The shape of the channel is sketched in Figure 8.5. The quantities are defined and measured with uncertainty sources as follows:

Approach channel width defined as $B = B_{nom} + \delta B_{cal} + \delta B_{meas}$ with a nominal value $B_{nom} = 2.002$ m, $\delta B_{cal} \sim N(0, 0.00025^2)$, and $\delta B_{meas} \sim t_{6-1}(0, 0.0025^2/6)$

[1] I want to thank Peter Harris from NPL for the additional detailed information to the problem.

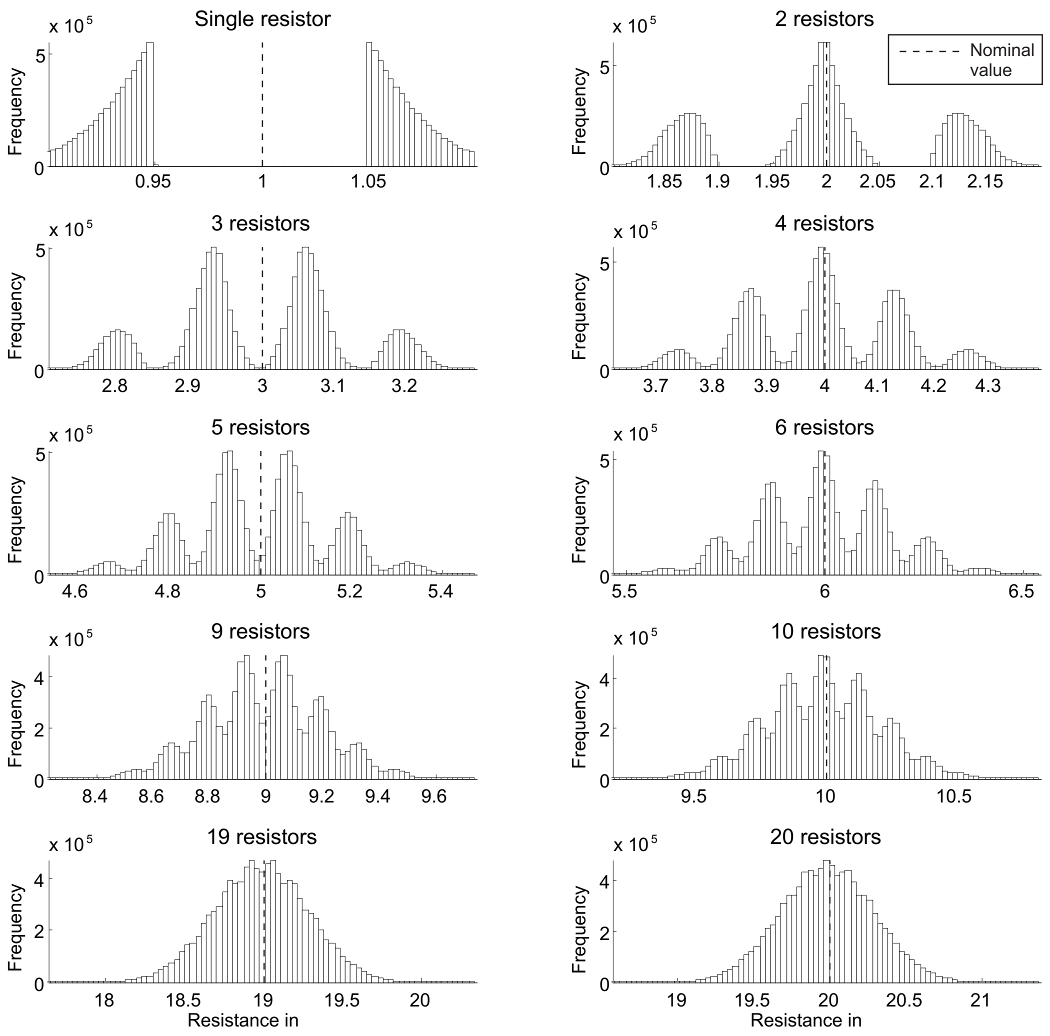


Figure 8.4: The single plots show the resulting distributions of C-grade 1 Ω resistors in serial connection for 1 to 6, 9, 10, 19, and 20 resistors. Even with 19 resistors in series a small dent is observed at the nominal value of 19 Ω.

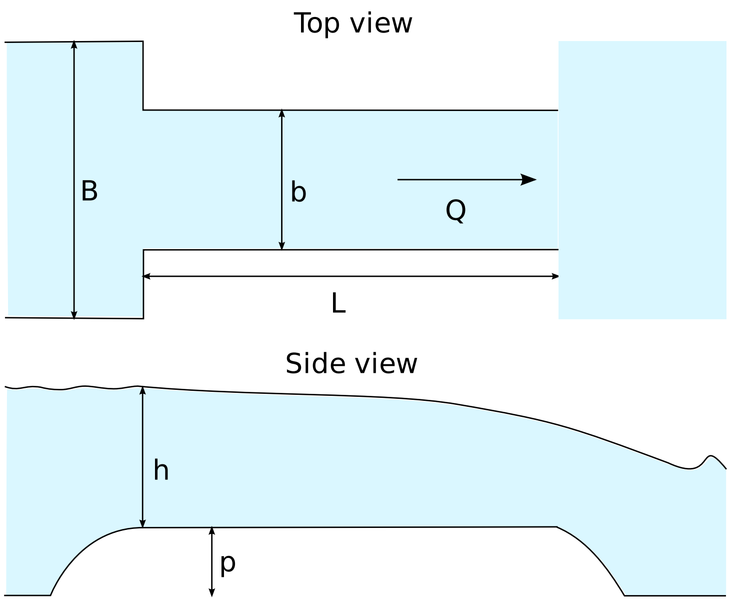


Figure 8.5: Concept and shape of an open flow channel; uncertainties in the measurement of the shape parameters of the channel lead to an uncertainty for the flow rate.

Throat width defined as $b = b_{nom} + \delta b_{cal} + \delta b_{meas}$ with a nominal value $b_{nom} = 0.997$ m, $\delta b_{cal} \sim N(0, 0.00025^2)$, and $\delta b_{meas} \sim t_{9-1}(0, 0.0022^2/9)$

Throat length defined as $L = L_{nom} + \delta L_{cal} + \delta L_{meas}$ with a nominal value $L_{nom} = 3.012$ m, $\delta L_{cal} \sim N(0, 0.00025^2)$, and $\delta L_{meas} \sim t_{9-1}(0, 0.0017^2/9)$

Hump height defined as $p = p_{nom} + \delta p_{cal} + \delta p_{meas}$ with a nominal value $p_{nom} = 0.252$ m, $\delta B_{cal} \sim N(0, 0.00025^2)$, and $\delta B_{meas} \sim t_{9-1}(0, 0.0019^2/9)$

Nominal head defined as $h = h_{nom} + \delta h_{cal} + \delta h_{res} + \delta h_{zero} + \delta h_{meas}$ with a nominal value $h_{nom} = 1.0$ m, $\delta h_{cal} \sim N(0, 0.0015^2)$, $\delta h_{res} \sim R(0, 0.0005^2/3)$, $\delta h_{zero} \sim N(0, 0.0014^2/10)$, and $\delta h_{meas} \sim N(0, 0.0025^2/1)$

Gravitational constant defined as $g = g_{nom} + \delta g_{cal}$ with a nominal value $g_{nom} = 9.812$ m and $\delta g_{cal} \sim N(0, 0.00025^2)$

Length measurements are performed with the same tape rule. So for measurement uncertainty evaluation we have to use the same values for influence quantities of calibration and hence, the condition $\delta B_{cal} = \delta b_{cal} = \delta L_{cal} = \delta p_{cal}$ has

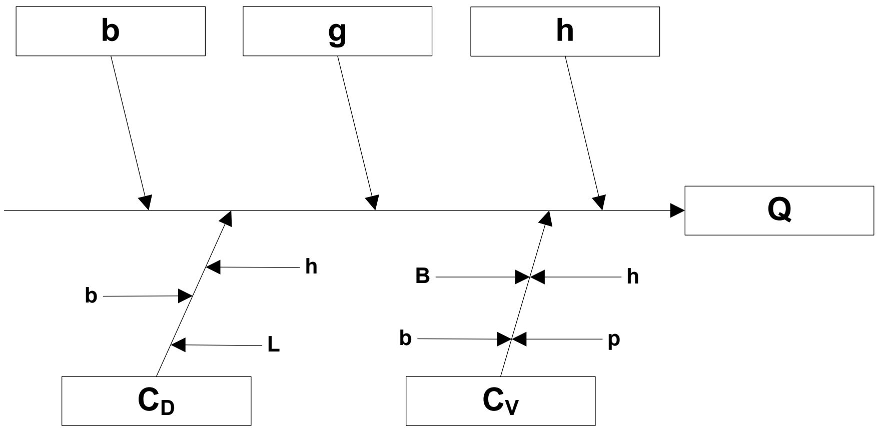


Figure 8.6: Cause and effect diagram for flow rate Q of an open channel

to be satisfied. The head height h of the water is measured with an ultrasonic detector.

The measurand Q of this measurement is the flow rate

$$Q = \left(\frac{2}{3}\right)^{3/2} g^{1/2} C_v C_D b h^{3/2}$$

with the acceleration $g = 9.812\ \mathrm{ms}^{-2}$ due to gravity. C_D is given as

$$C_D = (1 - 0.006L/b)(1 - 0.003L/h)^{3/2}$$

and C_v has to be calculated from the implicit functional relationship

$$4b^2h^2C_v^2 - 27B^2(h+p)^2(C_v^{2/3} - 1) = 0.$$

The modeling of this problem in the simulation framework is straightforward. Again, a single basic model and a simple simulation definition suffice. The complete XML code is provided in Appendix B.3. We want to concentrate on the influence quantity C_v at this point. A code fragment of its definition in the basic model follows.

```
...
<influence name="Cv" comment="Cv">
  <formula target="Cv" from="0" to="5" precision="0.00001">
    pow(2*b*h*Cv,2)-27*pow(B*(h+p),2)*(pow(Cv,2/3)-1)
  </formula>
  ...
</influence>
...
```

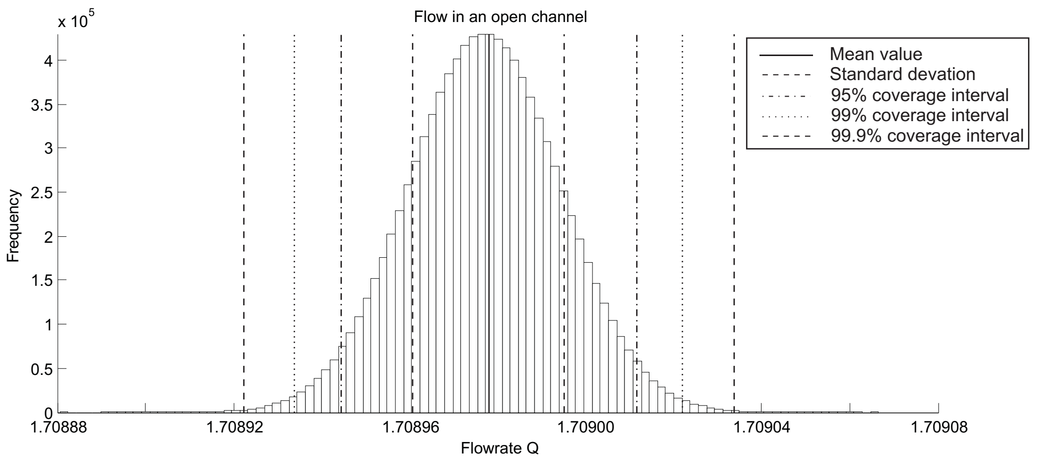


Figure 8.7: Histogram of the simulation with 10^7 trials of the flow rate for an open channel

The interesting new part is that the quantity `Cv` appears in the formula definition itself. Hence, for evaluation a zero-finding algorithm is applied. There are different well-known algorithms available [62]; in our case a simple bisection algorithm is already sufficient to explain the concept. The approach requires an attribute `target` that defines which quantity should be approximated. The two attributes `from` and `to` define the search range. They must have different signs, so that the range includes a root of the function in any case. Finally, the attribute `precision` defines a criterion, when the algorithm should terminate. The implementation of the zero-finding algorithm is interesting in this context, because it is called for each trial, say each set of values, of the Monte Carlo simulation, but also requires new function values to proceed. As the parser evaluates formulas only block wise, the algorithm is adapted to work block wise, too. Hence, new function values for a specific set of parameter values are only generated if the stop criterion has not been satisfied in the last iteration for the set. The algorithm stops when each element of the current block has found a root with the defined precision.

The simulation delivered a flow rate of $Q = 1.71\ \mathrm{m^3 s^{-1}}$ with a standard uncertainty of $u(Q) = 1.72 \times 10^{-5}\ \mathrm{m^3 s^{-1}}$. Figure 8.7 shows the histogram of the evaluation. The result can adequately be approximated by a normal distribution as already stated in *Best Practice Guide No. 6*.

In this chapter two special modeling issues were presented that require special constructs in the language definition. The first one was to restrict random draws from probability distributions or the result of the evaluation of influence quantities to feasible ranges. Besides, the example of graded resistors showed that this concept can be used in combination to model special probability distributions. The second issue was about implicit functional relationships. These problems require an extension of the language and the later one a zero-finding algorithm. All in all, the selected examples have shown that the language definition is proper for these special cases and that the language can quite easily be enhanced and adapted.

Until now we have seen the full language definition as well as additional extensions for special cases. In the next chapter I will present some system-relevant issues like support for validation of the simulation framework. Besides, we will see concepts for graphical user interfaces that realize the ideas of the modeling language. A first prototype allows the definition of basic models. Furthermore, a more advanced approach will be shown that uses an open-source project as basis for an editor that supports all features of the modeling language.

CHAPTER 9

Simulation framework

In the quest for higher accuracy, it is sometimes forgotten that a measurement system will not always be used by the expert who designed it. With modern interactive software, it is possible to make the most complex of measurement systems 'easy' to use. However, reliability can be poor unless the design ensures that operator error is minimised.

SSFM Best Practice Guide No. 1, 2000, Page 60

The project presented in this thesis results in a simulation framework for calculating measurement uncertainty for metrology according to the first supplement of the GUM and furthermore, realizes more advanced concepts. To be applicable in practice some fundamental issues have to be considered. The first one that I will explicitly point out here is the validation of the framework and portability to different operating systems. The language definition and software package include an advanced concept for validation to detect and prevent incorrect behavior on different computers or platforms or in new versions of the package.

The second point is about the acceptance of a completely new modeling language. In general, there is only little time for practitioners to adapt new things in a laboratory. Hence, we put emphasis on keeping as close as possible to the language used in daily work. Furthermore, we do not think that many people will accept a completely new programming language to evaluate measurement uncertainties as the evaluation is only a very small part of daily work. The abstraction from the language to a graphical user interface (GUI) is in our case fundamental. This is the reason why it has been considered right from the beginning of the project. In a first approach presented in this chapter a small – but nevertheless fully functional – prototype of a GUI for rather simple measurement scenarios has been implemented in Java and shows the capability of direct translation from user input to the domain-specific language of *MUSE*. A second approach uses an existing source code basis of an editor which is adapted to the specific requirements of the simulation framework.

We start with the first issue concerning the validation concept of the software package.

9.1 Validation of the simulation system

The presented system calculates statistical parameters from a measurement model. It is clear that a validation concept is vital for application as for example stated in *Best Practice Guide No. 1* [77]. The software package presented in this thesis was implemented in C/C++ and compiled and tested in parallel on different platforms to provide portability to diverse operation systems. This stresses the need for a validation system to be able to verify results on a specific machine. A source code revision system is used, regular code reviews of the programmers took place, and internal system checks – so-called component tests – as well as fix coded test scenarios are implemented to guarantee that no new errors are introduced from version to version. As validation is computationally expensive, the validation module has to be turned on in the calculation section of a simulation definition explicitly.

```
1 <simulation>
2   ...
3   <calculation>
4     ...
5     <validation mode="on" output="on" path="validate"/>
6     <measurand>...</measurand>
7   </calculation>
8 </simulation>
```

In line 5 the element `validation` is defined and turned on by setting attribute `mode` to `on`. The attribute `output` defines if status messages should be printed or if the validation should work in a silent mode and only inform in cases of an error. Finally, the attribute `path` defines an absolute or relative path to external validation scenarios. We will see in the following sections how the validation system works in detail.

9.1.1 Syntax and semantic checks

Concerning the model definition, the grammar definition of the language using the XML formats of DTDs and XML schema as explained in Chapter 3.2 already gives a good basis to prevent users from using wrong definitions and parameter checking. Of course the model has still to be checked for proper settings of parameters. Results of evaluations have to be analyzed for consistency and plausibility.

9.1.2 Component checks

The software itself includes internal system checks for the most important components like

- random number generators,
- probability distributions,
- combinations of data structures for basic models, processes, etc.,
- implementations of block-wise working operators, and
- the formula parser.

Internal test scenarios are applied to these components and compared to known results. If such a test fails detailed information about the error is provided with an option to abort the current evaluation.

9.1.3 Predefined models

The concept of the system validation can be expanded individually which allows regression testing. Predefined models can be included in the validation process and the statistical parameters calculated after evaluation can be compared with predefined values or even ranges of values. This allows a transparent validation of the system and can also be used to check for consistency with later versions of the simulation framework. For the most important modeling concepts predefined model scenarios are provided as listed in Table 9.1. They are evaluated one after another during the validation process and the resulting statistical parameters from the analysis are compared to predefined value ranges. A code fragment of such a validation definition will explain the details.

```
...
<calculation>
 <validate name="data.bin" dimension="1">
   <valmean>
```

Name	Description
valbmstatic.xml	Check static instance of a basic model
valinfnonstatic.xml	Check non-static influence quantity in basic model
valinfstatic.xml	Check static influence quantity in basic model
valprocnonstatic.xml	Check non-static process definition
valprocstatic.xml	Check static process definition
valscope.xml	Check scope of global variables, processes, instances and influence quantities
valswitch.xml	Check switch concept for processes in measurement series

Table 9.1: To validate the functionality of the simulation framework on different platforms model scenarios can be defined for consistency checks. The table shows already included scenarios that cover validating the most important components of the system.

```
39          <lower>-0.1</lower>
40          <upper>0.1</upper>
41        </valmean>
42        <valstd>
43          <lower>1.35</lower>
44          <upper>1.50</upper>
45        </valstd>
46      </validate>
47
48      <measurand>...</measurand>
49    </calculation>
```

Line 38 defines an element `validate` with an attribute `name` as filename with the results of simulation. The second attribute `dimension` is used if multiple results are evaluated. Hence, there may be arbitrary many elements `validate` if more than one result has to be checked for validity. Each element contains an element `valmean` and/or `valstd` defining an upper and lower limit for the mean value or, respectively, the standard deviation for comparison. If such a range check fails during validation of the system the system will ask to proceed or to cancel the current simulation.

9.1.4 Reference test sets

Finally, it is very important to have a set of examples with well-known results. We have to be consistent with the examples from the document this work relies on. Hence, the examples from the first supplement to the GUM are used as reference test set and are always the basis to prove correctness of the functionality of the simulation system. Furthermore, the gauge block calibration example [22, Section 9.5] is used for further investigations and advanced topics presented in this work, because of its clear structure and manageable number of influence quantities. Nevertheless, we think of it as a good practical example and it shows that already quite simple measurement scenarios can lead to interesting insights and results.

9.2 Support for graphical user interfaces

My experience at conferences and meetings [56, 81–83], where the simulation system has been presented, is that many experts from laboratories are a little bit afraid of working with a text-based simulation system. It can be quite an obstacle to start. Modeling of real-life problems is usually no problem if one of our group is present for active support, so the system is capable to solve such problems. Thus, a graphical user interface is of importance for the acceptance of the whole concept of the project and has been considered right from the start. The combination of XML and XML schema for the definition of the domain-specific language of *MUSE* is well suited for the generation of graphical user interfaces with rather low effort. The big advantage in having a clear definition of the language allows a direct interpretation of the grammar to provide a user interface. Only minor adaptations are necessary, e.g., additional validity and consistency checks for values, to get a point and click user interface.

Automated code generation One way to interpret the XML schema language definition is to automatically generate a source code basis for graphically user interfaces as the commercial[1] solution JAXFront [48] does for Java. The resulting code can then be adapted and completed to obtain a fully functional graphical user interface.

Fresh implementation As a small experiment I implemented from scratch a small prototype named *MuseLite* in Java to support metrologists in the definition of basic models for measurement uncertainty evaluation. The result is a fully functional user interface that allows defining arbitrarily complex basic models in a hierarchical structure supporting the probability distributions from GS1 and evaluating a single instance of such a basic model with Monte Carlo simulation. It allows visualizing the results using the tool *MuseView*[2]. Hence, it realizes a

[1]Actually, the company has just recently released a freely available community version, which shows a notification of the origin of the product in derived applications.

[2]The first version of the tool *MuseView* was developed during a semester work by Hasan Kaharan. It has been adapted for the requirements of the prototype of *MuseLite*.

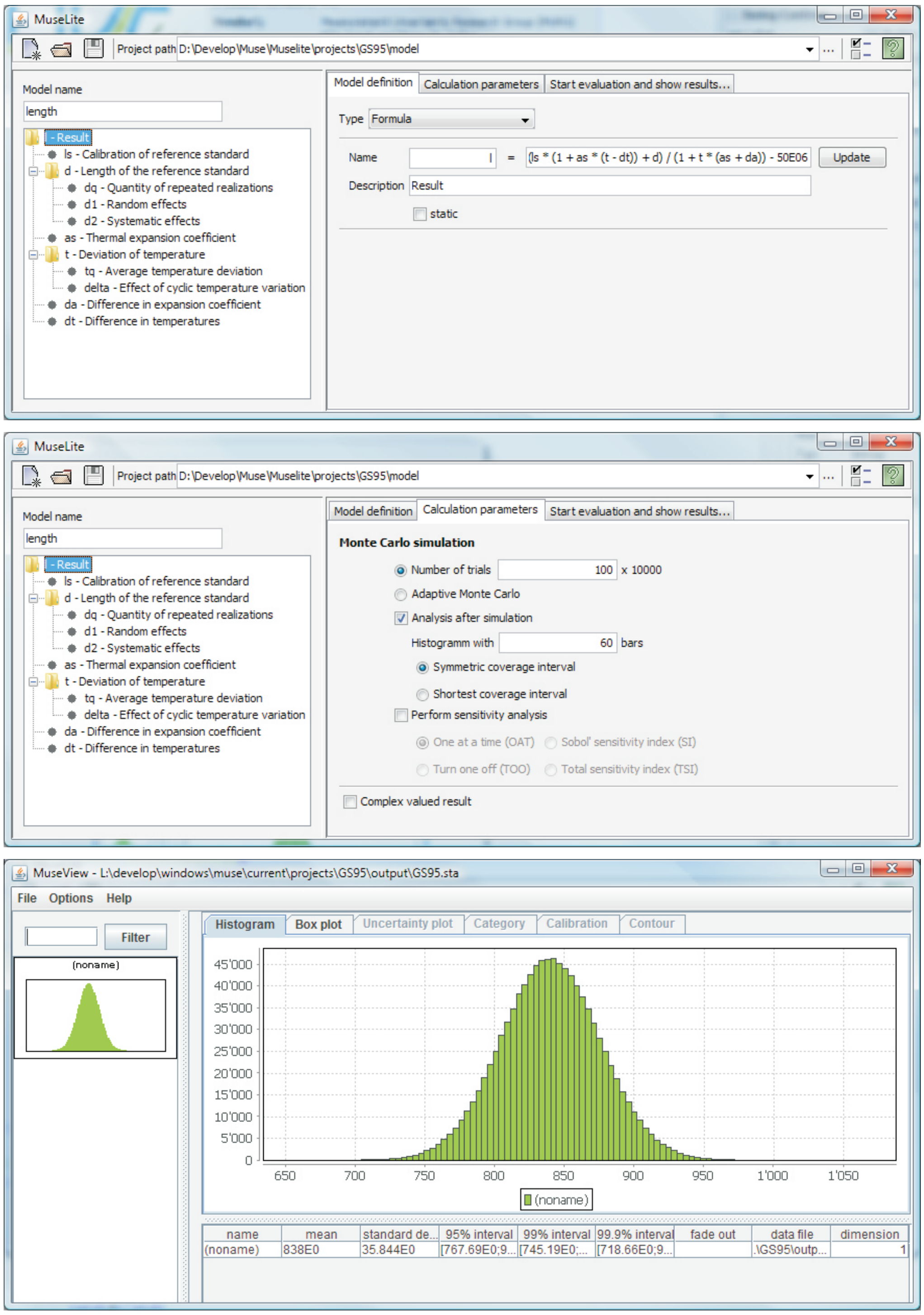
subset of the modeling language presented in this thesis. I experienced that the implementation has been very straightforward, as the concepts of the work due to the language definition can be easily mapped to internal class structures and data representation. Additionally, Java provides extensive support to handle XML files. Figure 9.1 shows a screenshot of the program with the definition of example 9.5 in GS1, the gauge block calibration. On the left of the dialog the hierarchical tree of influence quantities shows the structure of the basic model. In the right part of the upper screenshot there is the formula definition for an influence quantity. In the middle there is the definition of simulation and analyzing parameters. Finally, the third dialog shows the histogram and a table of the results after simulation in the visualization tool *MuseView*.

Building on existing solutions Furthermore, there are already open source solutions for XML editors that directly interpret XML schema and document type definitions; an example is the editor of the Xerlin project [49]. It is currently adapted for the needs of the presented simulation framework as a bachelor thesis by the student Dominik Gabi. It supports the whole language definition and capabilities for measurement uncertainty evaluation. Figure 9.2 shows a screenshot of the editor in a rather early stage of development. In the left sub dialog the simulation definition of the gauge block calibration example is open, on the right side the corresponding model definition.

The prototype of a GUI presented in this chapter supports a restricted subset of concepts for modeling and hence, a part of the features of the modeling language. Nevertheless, the prototype can already be used to calculate simple examples using only a single basic model. Besides, I think of it as a basis for a model builder of a large-scale simulation system that also realizes more advanced concepts like parameterization and multiple instances of basic models, process definitions, and the usage of variables. The experiment shows - as a proof of concept – that the language definition is fit for practice and allows a straightforward approach to user interface development. This has been one of the goals of this thesis.

The adaptation of the open source editor is a second viable approach that leads to a graphical user interface supporting experts from metrology in measurement uncertainty evaluation. It allows direct access to all features of the simulation framework with relatively low effort.

Last but not least, I want to point out once more that the concept of hierarchical modeling outperforms current spreadsheet approaches as hierarchical models reflect the structure and dependencies of the real-world far better than a sheer list of influence quantities.



Project path D:\Develop\Muse\Muselite\projects\GS95\model

Model name: length

- Result
  - ls - Calibration of reference standard
  - d - Length of the reference standard
    - dq - Quantity of repeated realizations
    - d1 - Random effects
    - d2 - Systematic effects
  - as - Thermal expansion coefficient
  - t - Deviation of temperature
    - tq - Average temperature deviation
    - delta - Effect of cyclic temperature variation
  - da - Difference in expansion coefficient
  - dt - Difference in temperatures

Type: Formula

Name: l = (ls * (1 + as * (t - dt)) + d) / (1 + t * (as + da)) - 50E06

Description: Result

☐ static

Monte Carlo simulation

[✓ Number of trials] 100 x 10000

Adaptive Monte Carlo

☑ Analysis after simulation

Histogramm with 60 bars

[✓ Symmetric coverage interval]

Shortest coverage interval

☐ Perform sensitivity analysis

[✓ One at a time (OAT)] Sobol' sensitivity index (SI)

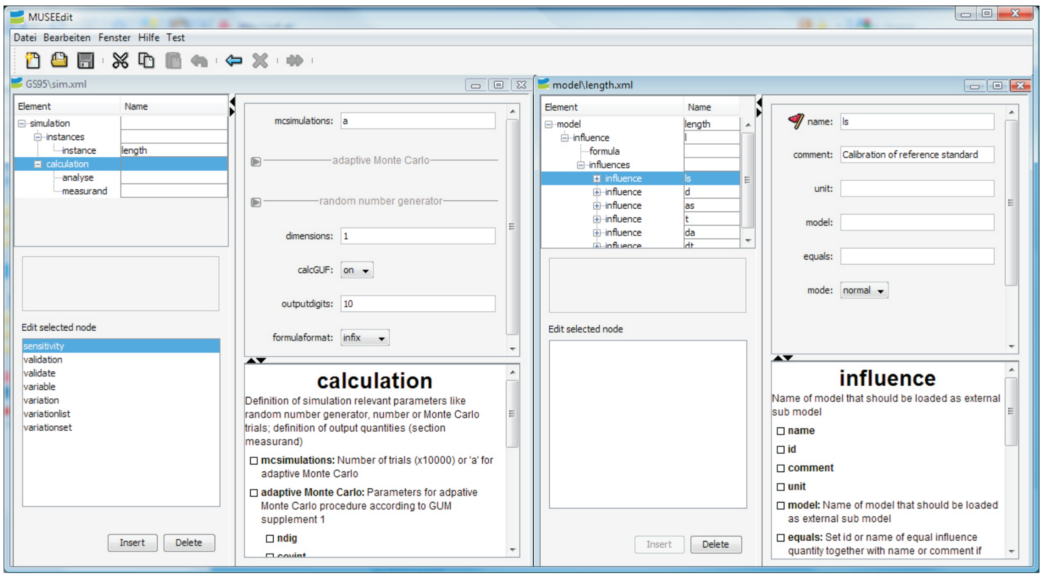
Turn one off (TOO) Total sensitivity index (TSI)

☐ Complex valued result

(noname)

name	mean	standard de...	95% interval	99% interval	99.9% interval	fade out	data file	dimension
(noname)	838E0	35.844E0	[767.69E0;9...	[745.19E0;...	[718.66E0;9...		.\GS95\outp...	1

Figure 9.1: Screenshots of *MuseLite* and *MuseView* with the definition and results from example 9.5 in GS1, the gauge block calibration



GS95\sim.xml

Element	Name
simulation	
instances	
instance	length
calculation	
analyse	
measurand	

Edit selected node

- sensitivity
- validation
- validate
- variable
- variation
- variationlist
- variationset

mcsimulations: a

adaptive Monte Carlo

random number generator

dimensions: 1

calcGUF: on

outputdigits: 10

formulaformat: infix

calculation

Definition of simulation relevant parameters like random number generator, number or Monte Carlo trials; definition of output quantities (section measurand)

- ☐ **mcsimulations:** Number of trials (x10000) or 'a' for adaptive Monte Carlo
- ☐ **adaptive Monte Carlo:** Parameters for adpative Monte Carlo procedure according to GUM supplement 1
  - ☐ **ndig**
  - ☐ **covint**

model\length.xml

Element	Name
model	length
influence	l
formula	
influences	
influence	ls
influence	d
influence	as
influence	t
influence	da
influence	dt

Edit selected node

name: ls

comment: Calibration of reference standard

unit:

model:

equals:

mode: normal

influence

Name of model that should be loaded as external sub model

- ☐ **name**
- ☐ **id**
- ☐ **comment**
- ☐ **unit**
- ☐ **model:** Name of model that should be loaded as external sub model
- ☐ **equals:** Set id or name of equal influence quantity together with name or comment if

Figure 9.2: Screenshot of *MuseEdit* with the simulation definition and the basic model of the gauge block calibration example

CHAPTER 10

Conclusion and summary

An der Akademie hielt Humboldt Vorlesungen über die Leitfähigkeit menschlicher Nerven. Er stand dabei, als im Nieselregen auf ausgetretenem Rasen vor der der Stadt der letzte Abschnitt des Längengrades gemessen wurde, der Paris mit dem Pol verband. Als es vollbracht war, nahmen alle die Hüte ab und schüttelten einander die Hände: Ein Zehnmillionstel der Strecke[1] *würde, in Metall gefaßt, zur Einheit aller künftigen Längenmessungen werden. Man wolle es Meter nennen. Es erfüllte Humboldt stets mit Hochgefühl, wenn etwas gemessen wurde; diesmal war er trunken vor Enthusiasmus. Die Erregung ließ ihn mehrere Nächte nicht schlafen.*

— Die Vermessung der Welt, Daniel Kehlmann, 2005 —

It is essential to complement every result of a measurement with a statement about its certainty respectively uncertainty. Only then measurement results can be communicated properly, used in further measurements, and compared to other measurement results. This is the reason why the GUM and its first supplement are

[1]The text is actually a little bit imprecise as the meter was defined as 10 millionth of the distance from the pole to the equator on the meridian in 1793. This is the reason why the definition of the meter in the text is named Kehlmann-Meter. Nevertheless, we can share the enthusiasm about the situation and the measurement and hence, I interpret it as a little "imprecision" of the author and artistic freedom.

very important documents introducing a standardized method for the evaluation of the measurement uncertainty. The approach is quite new and the two documents are still under discussion and are only recommendations, but they become more and more accepted on a broad basis. Nevertheless, measurement uncertainty evaluation is additional work in practice that takes time and costs a lot of money if seriously evaluated. All the more, it is essential to support metrologists in measurement uncertainty evaluation. In the end, the reward is a very detailed and deep understanding of their measurements, which helps to improve the accuracy of further measurements. An existing model for a measurement scenario together with a what-if machine allows simulating various parameter settings without the need of additional costly measurements, always assuming that the measurement models reflect reality properly.

The result of this project is a very advanced modeling language and a simulation framework that provide a way to formulate measurement scenarios using terms and expression of the specific domain. In the beginning I was a little bit skeptical if the XML-based approach will pay off and will be accepted by the experts from metrology. It appeared that it is quite a hurdle to start modeling and working with the system. Also the redundant format requiring keywords in opening and closing tags of elements is not everyone's cup of tea. Nevertheless, the advantages that resulted in the end justify the approach completely. We were able to use an existing, well-tested and established XML parser, which saved us a lot of implementation work and syntactical checks. The hierarchical definition of elements and attributes turned out to fit perfectly to the requirements of measurement scenario modeling. Language extensions can easily be applied enhancing the grammar in XML schema without the need of additional compilers or language and parser generating tools. The capability of interpreting the grammar directly into a graphical user interface is fantastic; changes and additions are reflected in the interface immediately. Textual descriptions and explanations annotated to elements and attributes can be provided in the grammar and interpreted as documentation. The next step for a complete measurement uncertainty evaluation tool would be the implementation of a report generator. Actually, XML-based models can quite easily be transformed to arbitrary document formats using the so-called *Extensible Stylesheet Language* (XSL). So, this is also a very big advantage, which will be of interest in further work.

For the introduction of the modeling concepts, we started with the very basics of measurement uncertainty evaluation presenting the different approaches in comparison. Afterwards, fundamentals were introduced like

- how the evaluation of the equation of the measurand can be performed in an efficient way using block-wise operations,
- how to model probability distributions, and

- how to analyze the resulting simulation data.

The extensive example of a titration measurement from analytical chemistry was used to develop the concepts of the modeling language step by step and to show that the hierarchical structure fits well. An encapsulation of basic models, processes and a calculation section were introduced. It turned out that the approach is capable of high-resolution, real-life modeling.

Furthermore, the framework helps in actual decision making as we have seen in the chapter about modeling series of measurement. I introduced a first approach for an algorithm to reduce the number of necessary reference measurements in a sequence of a measurement to a theoretical minimum. I think the approach has high potential, but it has to be proven in practice that it lives up to its promises.

Sensitivity analysis is another way of analyzing individual measurement scenarios in great detail determining the impact of a single influence quantity on the overall measurement uncertainty. Again, this approach is relatively new to measurement uncertainty evaluation with the Monte Carlo approach. I compared some ad-hoc approaches and more advanced ones using variance-based methods. It is now necessary to analyze results of the different approaches in practice applying more advanced real life measurement scenarios. Then it would also be interesting to implement and compare more and different approaches.

The additional examples of restricting values of influence quantities to physical limits and feasible ranges as well as the example of the flow in an open channel using implicit functional relationships picked up some special issues and showed that the language can easily be enhanced for these specific applications. We obtained very interesting results. Again, for further investigation it is necessary to fine tune the system, e.g., in a first step implementing more advanced zero-finding algorithms. Nevertheless, we could prove that the framework can be used to evaluate such measurement scenarios.

Finally, I discussed very practical aspects like approaches for graphical user interfaces that provide the concepts of the modeling language and capabilities of the simulation system. The validation of the simulation system on different platforms was another very important point.

In the application of measurement uncertainty evaluation one gains a very deep and detailed knowledge about the behavior and dependencies of measurement systems. For me, working in the field of metrology was and still is a very interesting experience. I discussed with experts from very different scientific fields like chemistry, biology, physics, statistics and mathematics. It was as interesting as demanding to put all information together in a single simulation framework as a computer scientist.

Outlook

As already mentioned, currently a first graphical user interface is under development that provides all features and concepts of the modeling language and the simulation framework. We hope to be able to support experts from the field of metrology with the emerging tool as a combination of the simulation core with an easy-to-use editor and a visualization tool in measurement uncertainty evaluation and in understanding measurement scenarios in great detail.

Furthermore, a very interesting application is currently under investigation concerning quality measures for DNA analysis and next-generation sequencing. The simulation framework already realizes specific requirements for applications like discrete probability distributions. Unfortunately, the scientific field – and hence, the measurement procedures – are quite new and, therefore, a general and accepted method to evaluate measurement uncertainties for such scenarios has yet to be defined.

Besides, different fields of application opened up for the simulation framework recently. In this work I concentrated on measurements from the fields of physics, chemistry, and biology, but of course measurements take also place in various other fields like sociology, economics, and so forth. As uncertainties appear in every measurement, considering the application of the presented method is appropriate.

However, the simulation framework is prepared and ready for action!

APPENDIX A

XML Schema Definitions

The definition files are the central part for the language definition. The version given in this appendix is somewhat shortened. Documentation and most of the comments are omitted to reduce length. The full version comes with each download of the software package *MUSE* from `http://www.muse.ethz.ch`.

A.1 XSD: Probability distributions

The XML schema definition for probability distributions is used for XML documents describing basic models as well as in simulation definitions. Therefore, the schema is included in the two XSDs `model.xsd` and `simulation.xsd`.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!-- ################################### -->
<!-- Definition of Distribution sub node -->
<!-- ################################### -->

<schema xmlns:xs="http://www.w3.org/2001/
      XMLSchema">

<element name="distribution" type="distribution"
      />
  <complexType name="distribution">
    <choice minOccurs="0">
      <element name="constant" type="constant"/>
      <element name="gauss" type="gauss"/>
      <element name="rectangle" type="lowerupper"
            />
      <element name="triangle" type="lowerupper"
            />
      <element name="gamma" type="alphabeta"/>
      <element name="beta" type="alphabeta"/>
      <element name="exponential"
            type="simplelambda"/>
      <element name="trapez" type="trapez"/>
      <element name="cltrapez" type="cltrapez"/>
      <element name="arcsine" type="arcsine"/>
      <element name="studentt" type="studentt"/>

      <element name="gausslimit"
            type="gausslimit"/>
      <element name="rectanglelimit"
            type="lowerupperlimit"/>
      <element name="trianglelimit"
            type="lowerupperlimit"/>

      <element name="bernoulli" type="simplep"/>
      <element name="binomial" type="binomial"/>
      <element name="geometric" type="simplep"/>
      <element name="poisson" type="simplelambda"
            />
    <element name="density" type="density"/>
    <element name="complex" type="complex"/>
    <element name="correlated"
          type="correlated"/>
  </choice>
  <attribute name="name" type="string"/>
  <attribute name="slot" type="string"/>
  <attribute name="equals" type="string"/>
</complexType>

<complexType name="studentt">
  <choice>
    <sequence>
      <element name="xbar" type="attparameter"
            />
      <element name="std" type="attparameter"/>
      <choice>
        <element name="dgf" type="attparameter"
              />
        <element name="n" type="attparameter"/>
      </choice>
    </sequence>

    <element name="values">
      <complexType>
        <sequence>
          <element name="value" type="double"
             minOccurs="3" maxOccurs="unbounded
                   "/>
        </sequence>
      </complexType>
    </element>
  </choice>
</complexType>

<complexType name="gausslimit">
  <sequence>
    <element name="mu" type="attparameter"/>
    <element name="sigma" type="attparameter"/>
  </sequence>
  <attributeGroup ref="attgrouplimit"/>
</complexType>

<complexType name="lowerupperlimit">
  <complexContent>
    <extension base="lowerupper">
      <attributeGroup ref="attgrouplimit"/>
    </extension>
  </complexContent>
</complexType>

<attributeGroup name="attgrouplimit">
  <attribute name="lower" type="double"/>
  <attribute name="upper" type="double"/>
  <attribute name="limitmode" default="cut">
    <simpleType>
      <restriction base="string">
        <enumeration value="cut"/>
        <enumeration value="collect"/>
      </restriction>
    </simpleType>
  </attribute>
</attributeGroup>

<complexType name="constant">
  <sequence>
    <element name="value" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="gauss">
  <sequence>
    <element name="mu" type="attparameter"/>
    <element name="sigma" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="lowerupper">
  <choice>
    <sequence>
      <element name="mean" type="attparameter"/
            >
      <element name="width" type="attparameter"
            />
    </sequence>
    <sequence>
      <element name="lower" type="attparameter"
            />
      <element name="upper" type="attparameter"
            />
    </sequence>
    <sequence>
      <element name="a" type="attparameter"/>
      <element name="b" type="attparameter"/>
    </sequence>
  </choice>
</complexType>

<complexType name="alphabeta">
  <sequence>
    <element name="alpha" type="attparameter"/>
    <element name="beta" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="simplep">
  <sequence>
    <element name="p" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="binomial">
  <sequence>
    <element name="n" type="attparameter"/>
    <element name="p" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="simplelambda">
  <sequence>
    <element name="lambda" type="attparameter"/
          >
  </sequence>
</complexType>

<complexType name="trapez">
  <sequence>
    <element name="lower" type="attparameter"/>
    <element name="upper" type="attparameter"/>
    <element name="beta" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="cltrapez">
  <sequence>
    <element name="lower" type="attparameter"/>
    <element name="upper" type="attparameter"/>
    <element name="inexactness" type="
          attparameter"/>
  </sequence>
</complexType>

<complexType name="arcsine">
  <sequence>
    <element name="lower" type="attparameter"/>
    <element name="upper" type="attparameter"/>
  </sequence>
</complexType>

<complexType name="density">
  <sequence>
    <element name="filename" type="
          normalizedString"/>
  </sequence>
  <attribute name="filetype" default="plain">
    <simpleType>
      <restriction base="string">
        <enumeration value="plain"/>
        <enumeration value="binary"/>
        <enumeration value="linesep"/>
        <enumeration value="openbugs"/>
      </restriction>
    </simpleType>
  </attribute>
```

```
    <attribute name="dimension" type="int"
          default="1"/>
    <attribute name="startvalue" type="
          nonNegativeInteger" default="0"/>
  </complexType>

  <complexType name="complex">
    <choice>
      <element name="distribution" type="
            distribution"
         minOccurs="2" maxOccurs="2"/>
      <sequence>
        <element name="realpart">
          <complexType>
            <sequence>
              <element name="distribution" type="
                    distribution"/>
            </sequence>
          </complexType>
        </element>
        <element name="imaginarypart">
          <complexType>
            <sequence>
              <element name="distribution" type="
                    distribution"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </choice>
  </complexType>

  <complexType name="correlated" mixed="true">
    <choice minOccurs="0">
      <element name="distlist">
        <complexType>
          <sequence>
            <element name="distribution" type="
                  distribution"
               maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>

      <element name="cholesky">
        <complexType>
          <sequence>
            <element name="muvector">
              <complexType>
                <sequence>
                  <element name="element" type="
                        attparameter"
                     minOccurs="1" maxOccurs="
                        unbounded"/>
                </sequence>
                <attribute name="length" type="
                      positiveInteger"
                   use="required"/>
              </complexType>
            </element>
            <choice>
              <!-- Both keywords are possible for
                   covariance matrix -->
              <element name="covariancematrix"
                    type="covariancematrix"/>
              <element name="uncertaintymatrix"
                    type="covariancematrix"/>
            </choice>
          </sequence>
        </complexType>
      </element>
    </choice>
    <attribute name="id" type="string"/>
    <attribute name="slot" type="positiveInteger"
          default="1"/>
    <attribute name="equals" type="string"/>
  </complexType>

  <complexType name="covariancematrix">
    <choice>
      <sequence>
        <element name="cell" minOccurs="1"
              maxOccurs="unbounded">
          <complexType>
            <simpleContent>
              <extension base="attparameter">
                <attribute name="index1"/>
                <attribute name="index2"/>
              </extension>
            </simpleContent>
          </complexType>
        </element>
      </sequence>

      <sequence>
        <element name="row" minOccurs="1"
              maxOccurs="unbounded">
          <complexType>
            <sequence>
              <element name="cell" type="
                    attparameter"
                 minOccurs="1" maxOccurs="
                    unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </choice>
    <attribute name="size" type="positiveInteger"
          use="required"/>
  </complexType>

  <complexType name="attparameter">
    <simpleContent>
    <extension base="normalizedString">
      <attribute name="parameter" type="string"/>
    </extension>
    </simpleContent>
  </complexType>
</schema>
```

A.2 XSD: External libraries

The language definition allows calling external library functions. As basic models and simulation definitions use this concept the XML schema is included in the two XSDs `model.xsd` and +simulation.xsd+.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!-- ################################### -->
<!-- Definition of external functions    -->
<!-- ################################### -->
<schema xmlns:xs="http://www.w3.org/2001/
      XMLSchema">
  <complexType name="externallibrary">
    <sequence>
      <element name="library" minOccurs="0"
            maxOccurs="unbounded">
```

```
        <complexType>
          <sequence>
            <element name="function" minOccurs="0
                  " maxOccurs="unbounded">
              <complexType>
                <attribute name="name" type="
                      normalizedString"/>
                <attribute name="parameters" type
                      ="positiveInteger"/>
              </complexType>
            </element>
          </sequence>
          <attribute name="path" type="
                normalizedString"/>
        </complexType>
      </element>
    </sequence>
  </complexType>
</schema>
```

A.3 XSD: Basic models

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<schema xmlns:xs="http://www.w3.org/2001/
      XMLSchema">
  <!-- Including distributions -->
  <include schemaLocation="distribution.xsd"/>
  <!-- Including external libraries -->
  <include schemaLocation="extern.xsd"/>

  <element name="model">
    <complexType>
      <sequence>
        <element name="extern"
           type="externallibrary"
           minOccurs="0" maxOccurs="1"/>
        <element name="parameters"
           type="parameters" minOccurs="0"/>
        <element name="influence"
           type="influence"  minOccurs="1"
           maxOccurs="unbounded"/>
      </sequence>
      <attribute name="name" type="ID"
         use="required"/>
      <attribute name="targetname"
         type="normalizedString" use="required"/>
    </complexType>
  </element>

  <!-- Define influence type-->
  <complexType name="influence">
    <choice>
      <sequence>
        <choice>
          <element name="formula" minOccurs="0">
            <complexType>
              <simpleContent>
                <extension
                  base="normalizedString">
                  <attribute name="target"
                    type="normalizedString"/>
                  <attribute name="from"
                    type="double"/>
                  <attribute name="to"
                    type="double"/>
                  <attribute name="precision"
                    type="double"/>
                </extension>
              </simpleContent>
            </complexType>
          </element>
        </choice>
        <element name="influences" minOccurs="0">
          <complexType>
            <sequence>
              <element name="influence" type="
                    influence"
                 minOccurs="0" maxOccurs="
                       unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <element name="distribution" type="
            distribution" minOccurs="0"/>
    </choice>
    <attribute name="name"
       type="normalizedString" use="required"/>
    <attribute name="unit"
       type="normalizedString"/>
    <attribute name="comment" type="string"/>
    <attribute name="model"
       type="normalizedString"/>
    <attribute name="equals"
       type="normalizedString"/>
    <attribute name="lower" type="double"/>
    <attribute name="upper" type="double"/>
    <attribute name="limitmode" default="cut">
      <simpleType>
        <restriction base="string">
          <enumeration value="cut"/>
          <enumeration value="collect"/>
          <enumeration value="fold"/>
        </restriction>
      </simpleType>
    </attribute>
    <attribute name="mode" default="normal">
      <simpleType>
        <restriction base="string">
          <enumeration value="normal"/>
          <enumeration value="static"/>
        </restriction>
      </simpleType>
    </attribute>
    <attribute name="log" type="onoff"
       default="off"/>
    <attribute name="logfilename"
       type="normalizedString"/>
  </complexType>

  <simpleType name="onoff">
    <restriction base="string">
      <enumeration value="on"/>
      <enumeration value="off"/>
    </restriction>
  </simpleType>
</schema>
```

A.4 XSD: Simulation definition

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<!-- simulation consists of three parts: -->
<!--   Initialize items                  -->
<!--   Define measurement process        -->
<!--   Define measurand and variables    -->

<schema xmlns:xs="http://www.w3.org/2001/
      XMLSchema">

<!-- Including distributions -->
<include schemaLocation="distribution.xsd"/>
<!-- Including external libraries -->
<include schemaLocation="extern.xsd"/>

<!-- Defintion of root element simulation -->
<element name="simulation">
  <complexType>
    <sequence>
      <element name="extern"
         type="externallibrary"
         minOccurs="0" maxOccurs="1"/>
      <element name="instances" type="instances"
        minOccurs="0"/>
      <element name="processes" type="processes"
        minOccurs="0"/>
      <element name="calculation"
        type="calculation"/>
    </sequence>
  </complexType>
</element>

  <!-- ######################### -->
  <!--  Phase 1: Initialization  -->
  <!-- ######################### -->

  <complexType name="instances">
    <sequence>
      <element name="instance" minOccurs="0"
            maxOccurs="unbounded">
        <complexType>
          <sequence>
            <element name="parameters"
               type="instparameters"
               minOccurs="0"/>
            <element name="overrides"
              type="instoverrides"
              minOccurs="0"/>
            <element name="log" type="instlog"
               minOccurs="0"
               maxOccurs="unbounded"/>
          </sequence>

          <attribute name="model"
            type="normalizedString"
            use="required"/>
          <attribute name="name"
            type="normalizedString"
            use="required"/>
          <attribute name="mode"
            default="normal">
            <simpleType>
              <restriction base="string">
                <enumeration value="normal"/>
                <enumeration value="static"/>
              </restriction>
            </simpleType>
          </attribute>
        </complexType>
      </element>
    </sequence>
  </complexType>

    <complexType name="instparameters">
      <sequence>
        <element name="parameter"
          maxOccurs="unbounded">
          <complexType>
            <attribute name="name"
              type="normalizedString"
              use="required"/>
          </complexType>
        </element>
      </sequence>
    </complexType>

    <complexType name="instoverrides">
      <sequence>
        <element name="override"
          maxOccurs="unbounded">
          <complexType>
            <attribute name="name"
              type="normalizedString"
              use="required"/>
            <attribute name="with"
              type="normalizedString"
              use="required"/>
          </complexType>
        </element>
      </sequence>
    </complexType>

    <complexType name="instlog">
      <attribute name="filename"
         type="normalizedString" use="required"/>
      <attribute name="name"
         type="normalizedString" use="required"/>
      <attribute name="binary" default="on">
        <simpleType>
          <restriction base="string">
            <enumeration value="on"/>
            <enumeration value="off"/>
          </restriction>
        </simpleType>
      </attribute>
    </complexType>

  <!-- ########################### -->
  <!--  Phase 2: Process defintion -->
  <!-- ########################### -->

  <complexType name="processes">
    <sequence>
      <element name="process"
       maxOccurs="unbounded">
        <complexType name="process">
          <sequence>
            <element name="step" minOccurs="0"
              maxOccurs="unbounded">
              <complexType>
                <sequence>
                  <element name="variable"
                     type="variable"
                     maxOccurs="unbounded"/>
                </sequence>
                <attribute name="name"
                   type="normalizedString"
                   use="required"/>
              </complexType>
            </element>

            <element name="variable"
               type="variable"
               minOccurs="0"
               maxOccurs="unbounded"/>
            <element name="formula"
               type="formula"/>
          </sequence>

          <attribute name="name"
             type="normalizedString"
             use="required"/>
          <attribute name="comment"
             type="string"/>
          <attribute name="mode"
            default="normal">
          <simpleType>
            <restriction base="string">
              <enumeration value="normal"/>
              <enumeration value="static"/>
            </restriction>
          </simpleType>
        </attribute>
      </complexType>
    </element>
  </sequence>
</complexType>

<!-- ###################### -->
<!--  Phase 3: Calculation  -->
<!-- ###################### -->

<complexType name="calculation">
  <sequence>
    <element name="sensitivity"
      type="sensitivity" minOccurs="0"/>
    <element name="analyze"
      type="analyze" minOccurs="0"/>
    <choice>
      <element name="validation"
        type="validation" minOccurs="0"/>
      <element name="validate" type="validate"
        minOccurs="0" maxOccurs="unbounded"/>
    </choice>
    <element name="variable" type="variable"
      minOccurs="0" maxOccurs="unbounded"/>
    <element name="variation" type="variation"
      minOccurs="0" maxOccurs="unbounded"/>
    <element name="variationlist"
      type="variationlist"
      minOccurs="0" maxOccurs="unbounded"/>
    <element name="variationset"
      type="variationset"
      minOccurs="0" maxOccurs="unbounded"/>
    <element name="measurand" type="measurand"/
          >
  </sequence>
  <attribute name="calcGUF" type="onoff"
        default="off"/>
  <attribute name="mctrials" type="mctrials"
        default="1000"/>
  <attribute name="ndig" type="positiveInteger"
         default="2"/>
  <attribute name="covint" type="double"
        default="0.95"/>
  <attribute name="covtype" default="shortest">
    <simpleType>
      <restriction base="string">
        <enumeration value="shortest"/>
        <enumeration value="symmetric"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="deltafactor" type="double"
        default="1.0"/>
  <attribute name="outputdigits"
    type="positiveInteger" default="10"/>
  <attribute name="dimensions" type="integer"
        default="1"/>
  <attribute name="rngenerator" default="mt">
    <simpleType>
      <restriction base="string">
        <enumeration value="mt"/>
        <enumeration value="wh"/>
        <enumeration value="dmt"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="staticseed" type="onoff"
        default="off"/>
  <attribute name="formulaformat" default="
        infix">
    <simpleType>
      <restriction base="string">
        <enumeration value="infix"/>
        <enumeration value="postfix"/>
      </restriction>
    </simpleType>
  </attribute>
```

```
</complexType>

<complexType name="sensitivity">
  <choice>
    <element name="sensitivityitem" type="
          sensitivityitem"
       maxOccurs="unbounded"/>
    <element name="sensitivityset" maxOccurs="
          unbounded">
      <complexType>
        <sequence>
          <element name="sensitivityitem"
             type="sensitivityitem" maxOccurs="
                  unbounded"/>
        </sequence>
      </complexType>
    </element>
  </choice>

  <attribute name="mode" default="none">
    <simpleType>
      <restriction base="string">
        <enumeration value="none"/>
        <enumeration value="all"/>
        <enumeration value="set"/>
      </restriction>
    </simpleType>
  </attribute>

  <attribute name="type" default="OAT">
    <simpleType>
      <restriction base="string">
        <enumeration value="OAT"/>
        <enumeration value="TOO"/>
        <enumeration value="sobol"/>
        <enumeration value="TSI"/>
      </restriction>
    </simpleType>
  </attribute>

  <attribute name="stdfactor" type="double"
        default="0.0"/>
</complexType>

<complexType name="sensitivityset">
  <sequence>
    <element name="sensitivityitem"
      type="sensitivityitem" maxOccurs="
          unbounded"/>
  </sequence>
</complexType>

  <complexType name="sensitivityitem">
    <attribute name="mode" default="influence">
      <simpleType>
        <restriction base="string">
          <enumeration value="influence"/>
          <enumeration value="instance"/>
          <enumeration value="process"/>
        </restriction>
      </simpleType>
    </attribute>
    <attribute name="process"
      type="normalizedString"/>
    <attribute name="instance"
      type="normalizedString"/>
    <attribute name="influence"
      type="normalizedString"/>
  </complexType>


<complexType name="analyze">
  <attribute name="mode" type="onoff"
    default="off"/>
  <attribute name="histbars"
    type="positiveInteger" default="40"/>
  <attribute name="datafiles" default="keep">
    <simpleType>
      <restriction base="string">
        <enumeration value="keep"/>
        <enumeration value="delete"/>
      </restriction>
    </simpleType>
    </attribute>
    <attribute name="filename" type="string" />
    <attribute name="cormatrix" type="onoff"
      default="off"/>
    <attribute name="covtype" default="shortest">
      <simpleType>
        <restriction base="string">
          <enumeration value="shortest"/>
          <enumeration value="symmetric"/>
        </restriction>
      </simpleType>
    </attribute>
  </complexType>

  <!-- Validation used for settings for
        validation -->
  <complexType name="validation">
    <attribute name="mode" type="onoff"
     default="off"/>
    <attribute name="output" type="onoff"
     default="off"/>
    <attribute name="continueonfail" type="onoff"
     default="off"/>
    <attribute name="path"
     type="normalizedString" default="validate"/>
  </complexType>

  <simpleType name="onoff">
    <restriction base="string">
      <enumeration value="on"/>
      <enumeration value="off"/>
    </restriction>
  </simpleType>

  <!-- Validate used for actually validating a
        simulation file -->
  <complexType name="validate">
    <sequence>
      <element name="valmean" minOccurs="0">
        <complexType>
          <choice>
            <element name="value" type="
                  attparameter"/>
            <sequence>
              <element name="lower" type="
                    attparameter"/>
              <element name="upper" type="
                    attparameter"/>
            </sequence>
          </choice>
        </complexType>
      </element>
      <element name="valstd" minOccurs="0">
        <complexType>
          <choice>
            <element name="value"
              type="attparameter"/>
            <sequence>
              <element name="lower"
                type="attparameter"/>
              <element name="upper"
                type="attparameter"/>
            </sequence>
          </choice>
        </complexType>
      </element>
    </sequence>
    <attribute name="name"
      type="normalizedString"/>
    <attribute name="dimension"
      type="integer" default="1"/>
  </complexType>

  <!-- List of formulas to evaluate -->
  <complexType name="measurand" mixed="true">
    <choice>
      <element name="formula" type="formula"
         minOccurs="0" maxOccurs="unbounded"/>
      <element name="series" type="series"
         minOccurs="0" maxOccurs="unbounded"/>
      <element name="regions" type="regions"
         minOccurs="0" maxOccurs="unbounded"/>
      <element name="optimize" type="optimize"
         minOccurs="1" maxOccurs="1"/>
    </choice>
  </complexType>

  <complexType name="series">
    <sequence>
      <element name="reference"
        maxOccurs="unbounded">
        <complexType>
          <sequence>
            <element name="sample"
              maxOccurs="unbounded">
              <complexType>
                <attribute name="name"
                   type="normalizedString"
                   use="required"/>
                <attribute name="process"
                   type="normalizedString"
                   use="required"/>
                <attribute name="number"
                   type="positiveInteger"
                   use="required"/>
                <attribute name="mode"
                  default="local">
                  <simpleType>
                    <restriction base="string">
                      <enumeration
                        value="local"/>
                      <enumeration
                        value="continuous"/>
                    </restriction>
                  </simpleType>
                </attribute>
                <attribute name="analyze"
                  default="none">
                  <simpleType>
                    <restriction base="string">
                      <enumeration
                        value="none"/>
                      <enumeration
                        value="linlastref"/>
                      <enumeration
                        value="linnextref"/>
                      <enumeration
                        value="lin2mean"/>
                      <enumeration
                        value="lin4mean"/>
                      <enumeration
                        value="linallmean"/>
                      <enumeration
                        value="fitline"/>
                    </restriction>
                  </simpleType>
                </attribute>
                <attribute name="log"
                  type="onoff" default="on"/>
              </complexType>
            </element>
          </sequence>

          <attribute name="name"
             type="normalizedString"
             use="required"/>
          <attribute name="process"
             type="normalizedString"
             use="required"/>
          <attribute name="number"
             type="positiveInteger"
             use="required"/>
          <attribute name="log"
             type="onoff" default="on"/>
        </complexType>
      </element>
    </sequence>
  </complexType>

  <complexType name="regions">
    <sequence>
      <element name="region"
        maxOccurs="unbounded">
        <complexType>
          <attribute name="name"
            type="normalizedString"
            use="required"/>
          <attribute name="process"
            type="normalizedString"
            use="required"/>
        </complexType>
      </element>
    </sequence>
    <attribute name="from" type="double"
      use="required"/>
    <attribute name="to" type="double"
       use="required"/>
    <attribute name="step" type="double"
       default="1.0"/>
  </complexType>

  <complexType name="optimize">
    <attribute name="reference"
       type="normalizedString" use="required"/>
    <attribute name="sample"
       type="normalizedString" use="required"/>
    <attribute name="samples"
       type="positiveInteger" use="required"/>
    <attribute name="analyze" default="none">
      <simpleType>
      <restriction base="string">
        <enumeration value="none"/>
        <enumeration value="linlastref"/>
        <enumeration value="linnextref"/>
        <enumeration value="lin2mean"/>
        <enumeration value="lin4mean"/>
        <enumeration value="linallmean"/>
        <enumeration value="fitline"/>
      </restriction>
      </simpleType>
    </attribute>
    <attribute name="logreferences" type="onoff"
          default="off"/>
  </complexType>

<!-- ########################### -->
<!--    Variables and variation   -->
<!-- ########################### -->

  <complexType name="formula">
    <simpleContent>
      <attribute name="name"
         type="normalizedString"/>
    </simpleContent>
  </complexType>

  <complexType name="variable" mixed="true">
    <choice>
      <element name="distribution"
         type="distribution" minOccurs="0"/>
      <element name="switch" type="switch"
         minOccurs="0"/>
    </choice>
    <attribute name="name"
       type="normalizedString" use="required"/>
    <attribute name="unit"
       type="normalizedString"/>
    <attribute name="comment" type="string"/>
  </complexType>

  <complexType name="switch">
    <sequence>
      <element name="case" maxOccurs="unbounded">
        <complexType mixed="true">
          <sequence>
            <element name="distribution"
              type="distribution" minOccurs="0"/>
          </sequence>
          <!-- Depending on reference - sample
                combination -->
          <attribute name="samindex" default="-1"
                 type="integer"/>
          <attribute name="samfrom"  default="-1"
                 type="integer"/>
          <attribute name="samto"    default="-1"
                 type="integer"/>
          <attribute name="samstep"  default=" 1"
                 type="positiveInteger"/>
```

```
          <attribute name="refindex" default="-1"
                 type="integer"/>
          <attribute name="reffrom"  default="-1"
                 type="integer"/>
          <attribute name="refto"    default="-1"
                 type="integer"/>
          <attribute name="refstep"  default="1"
                type="positiveInteger"/>
          <!-- Depending on current number of
                measurement -->
          <attribute name="curindex" default="-1"
                 type="integer"/>
          <attribute name="curfrom"  default="-1"
                 type="integer"/>
          <attribute name="curto"    default="-1"
                 type="integer"/>
          <attribute name="curstep"  default=" 0"
                 type="integer"/>
        </complexType>
      </element>
      <element name="default">
        <complexType mixed="true">
          <sequence>
            <element name="distribution"
              type="distribution" minOccurs="0"/>
          </sequence>
        </complexType>
      </element>
    </sequence>
  </complexType>

  <complexType name="variation">
    <attribute name="name"
      type="normalizedString" use="required"/>
    <attribute name="unit"
      type="normalizedString"/>
    <attribute name="from" type="double"
      use="required"/>
    <attribute name="to" type="double"
      use="required"/>
    <attribute name="step" type="double"
      default="1"/>
  </complexType>

  <complexType name="variationlist">
    <sequence>
      <element name="value" type="attparameter"
         maxOccurs="unbounded"/>
    </sequence>
    <attribute name="name"
      type="normalizedString" use="required"/>
    <attribute name="unit"
      type="normalizedString"/>
  </complexType>

  <complexType name="variationset">
    <sequence>
      <element name="set" maxOccurs="unbounded">
        <complexType>
          <sequence>
            <element name="variable"
              type="variable"
              maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>
    </sequence>
    <attribute name="name"
      type="normalizedString" use="required"/>
    <attribute name="unit"
      type="normalizedString"/>
  </complexType>

  <!-- Selecting number of simulations or
        adaptive MC -->
  <simpleType name="mctrials">
    <restriction base="string">
      <pattern value="a|([0-9]*)"/>
    </restriction>
  </simpleType>
</schema>
```

APPENDIX B

Listings of examples

B.1 GUM supplement 1: additive models

This example is taken from GUM supplement 1 [22, Section 9.2].

Listing B.1: Basic model opengauss.xml

```
<model name="opengauss" targetname="og">
  <influence name="og" comment="gauss">
    <distribution>
      <gauss>
        <mu parameter="#mu">0</mu>
        <sigma parameter="#sigma">1</sigma>
      </gauss>
    </distribution>
  </influence>
</model>
```

Listing B.2: Basic model openrect.xml

```
<model name="openrect" targetname="or">
  <influence name="or" comment="rectangle">
    <distribution>
      <rectangle>
        <mean parameter="#mean">0</mean>
        <width parameter="#width">1.73205080756888</width>
      </rectangle>
    </distribution>
  </influence>
</model>
```

Listing B.3: Simulation definition `sim.xml`

```
<simulation>
  <instances>
    <!-- Normal distributions for first example -->
    <instance name="g1" model="opengauss"/>
    <instance name="g2" model="opengauss"/>
    <instance name="g3" model="opengauss"/>
    <instance name="g4" model="opengauss"/>

    <!-- Rectangular distributions for second and third
        example -->
    <instance name="r1" model="openrect"/>
    <instance name="r2" model="openrect"/>
    <instance name="r3" model="openrect"/>
    <instance name="r4" model="openrect"/>

     <!-- Last rectangular is different for third example -->
    <instance name="r5" model="openrect">
      <parameters>
        <parameter name="#width">
           17.32050807568877
        </parameter>
      </parameters>
    </instance>
  </instances>

  <calculation mcsimulations="100" >
    <analyze mode="on" histbars="40"/>
    <validation mode="on" output="on" path="validate"/>
    <measurand>
      <formula name="gauss">g1+g2+g3+g4</formula>
      <formula name="rect1">r1+r2+r3+r4</formula>
      <formula name="rect2">r1+r2+r3+r5</formula>
    </measurand>
  </calculation>
</simulation>
```

B.2 GUM supplement 1: gauge block calibration

This example is taken from GUM supplement 1 [22, Section 9.5]. Simulation definition uses advanced settings for adaptive Monte Carlo approach [22, Section 7.9].

Listing B.4: Basic model `length.xml`

```
<model name="length" targetname="l">
  <influence name="l" comment="Length of gauge block">
    <formula>
      ls*(1+as*(t-dt))+d / (1+t*(as+da)) - 50e6
    </formula>

    <influences>
      <!-- Definition of ls -->
      <influence name="ls" comment="Calibration of reference
          standard">
        <distribution>
          <studentt>
            <xbar>50000623</xbar>
            <std>25</std>
            <dgf>18</dgf>
          </studentt>
        </distribution>
      </influence>

      <!-- Definition of d -->
      <influence name="d" comment="Length of the reference
          standard">
        <formula>dq+d1+d2</formula>
        <influences>
          <influence name="dq" comment="Quantity of repeated
              realizations">
            <distribution>
            <studentt>
              <xbar>215</xbar>
              <std>6</std>
              <dgf>24</dgf>
            </studentt>
            </distribution>
          </influence>
          <influence name="d1" comment="Random effects">
            <distribution>
            <studentt>
              <xbar>0</xbar>
              <std>4</std>
              <dgf>5</dgf>
            </studentt>
            </distribution>
          </influence>
          <influence name="d2" comment="Systematic effects">
            <distribution>
            <studentt>
              <xbar>0</xbar>
              <std>7</std>
```

```xml
            <dgf>8</dgf>
          </studentt>
          </distribution>
        </influence>
      </influences>
    </influence>

    <!-- Definition of alpha_s -->
    <influence name="as" comment="Thermal expansion coefficient">
      <distribution>
        <rectangle>
          <mean>0.0000115</mean>
          <width>0.000002</width>
        </rectangle>
      </distribution>
    </influence>

    <!-- Definition of theta -->
    <influence name="t" comment="Deviation of temperature">
      <formula>tq+delta</formula>
      <influences>
        <influence name="tq" comment="Average temperature deviation">
          <distribution>
            <gauss>
              <mu>-0.1</mu>
              <sigma>0.2</sigma>
            </gauss>
          </distribution>
        </influence>
        <influence name="delta" comment="Effect of cyclic temp. variation">
          <distribution>
            <arcsine>
              <lower>-0.5</lower>
              <upper>0.5</upper>
            </arcsine>
          </distribution>
        </influence>
      </influences>
    </influence>

    <!-- Definition of delta_alpha -->
    <influence name="da" comment="Difference in expansion coefficient">
      <distribution>
        <cltrapez>
          <lower>-0.000001</lower>
```

```
            <upper>0.000001</upper>
            <inexactness>0.0000001</inexactness>
          </cltrapez>
        </distribution>
      </influence>

      <!-- Definition of delta_psi -->
      <influence name="dt" comment="Difference in
          temperatures">
        <distribution>
          <cltrapez>
            <lower>-0.050</lower>
            <upper> 0.050</upper>
            <inexactness>0.025</inexactness>
          </cltrapez>
        </distribution>
      </influence>
    </influences>
  </influence>
</model>
```

Listing B.5: Simulation definition `sim.xml`

```
<simulation>
  <instances>
    <instance name="l" model="length"/>
  </instances>
  <calculation mcsimulations="a" covint="0.99"
      covtype="shortest" ndig="2">
    <analyze mode="on" histbars="40"/>
    <measurand>l</measurand>
  </calculation>
</simulation>
```

B.3 Best Practice Guide No. 6: flow in a channel

This example is taken from *SSFM Best Practice Guide No. 6* [15, Section 9.3] and originated from the National Engineering Laboratory (NEL) in the UK. Additional information was provided as private communication by Peter Harris from the National Physics Laboratory (NPL) in the UK.

Listing B.6: Basic model `flowrate.xml`

```
<model name="flowrate" targetname="Q">
  <!-- calibration of tape rule -->
  <influence name="drcal" comment="calibration"
    mode="static">
    <distribution>
      <gauss>
        <mu>0.0</mu>
        <sigma>0.00025*0.00025</sigma>
      </gauss>
    </distribution>
  </influence>

  <influence name="h" comment="nominal head" unit="m"
    mode="static">
    <formula>1.0 + dhcal + dhres +dhzero + dhmeas</formula>
    <influences>
      <influence name="dhcal" comment="calibration">
        <distribution>
          <gauss>
            <mu>0.0</mu>
            <sigma>0.0015*0.0015</sigma>
          </gauss>
        </distribution>
      </influence>
      <influence name="dhres" comment="resolution">
        <distribution>
          <rectangle>
            <mean>0.0</mean>
            <width>0.0005*0.0005/3</width>
          </rectangle>
        </distribution>
      </influence>
      <influence name="dhzero" comment="zero">
        <distribution>
          <gauss>
            <mu>0.0</mu>
            <sigma>0.0014*0.0014/10</sigma>
          </gauss>
        </distribution>
      </influence>
      <influence name="dhmeas" comment="measurement">
        <distribution>
          <gauss>
            <mu>0.0</mu>
            <sigma>0.0025*0.0025</sigma>
          </gauss>
        </distribution>
      </influence>
    </influences>
  </influence>

  <influence name="b" comment="throat width" unit="m"
    mode="static">
    <formula>0.997+drcal+dbmeas</formula>
    <influences>
      <influence comment="drcal" equals="drcal"/>
      <influence name="dbmeas" comment="measurement">
        <distribution>
          <studentt>
            <xbar>0.0</xbar>
            <std>0.0022*0.0022/9</std>
            <dgf>8</dgf>
          </studentt>
        </distribution>
      </influence>
    </influences>
  </influence>

  <influence name="Q" comment="flow rate">
    <formula>
      pow(2/3,3/2)*sqrt(g)*Cv*Cd*b*pow(h,3/2)
    </formula>

    <influences>
      <influence comment="b" equals="b"/>
      <influence comment="h" equals="h"/>

      <influence name="g" comment="gravity" unit="ms^-2">
        <formula>9.812 + dg</formula>
        <influences>
          <influence name="dg" comment="uncertainty">
            <distribution>
              <gauss>
                <mu>0.0</mu>
                <sigma>0.00025*0.00025</sigma>
              </gauss>
            </distribution>
          </influence>
        </influences>
      </influence>

      <influence name="Cd" comment="Cd">
        <formula>(1-0.006*L/b)*pow(1-0.003*L/h,3/2)</formula>
        <influences>
          <influence comment="b" equals="b"/>
          <influence comment="h" equals="h"/>
          <influence name="L" comment="throat length">
            <formula>3.012+drcal+dLmeas</formula>
            <influences>
              <influence comment="drcal" equals="drcal"/>
              <influence name="dLmeas" comment="measurement">
                <distribution>
                <studentt>
                  <xbar>0.0</xbar>
                  <std>0.0017*0.0017/9</std>
                  <dgf>8</dgf>
                </studentt>
              </distribution>
            </influence>
          </influences>
        </influence>
      </influences>
    </influence>

    <influence name="Cv" comment="Cv">
      <formula target="Cv" from="0" to="5"
          precision="0.00001">
        pow(2*b*h*Cv,2)-27*pow(B*(h+p),2)*(pow(Cv,2/3)-1)
      </formula>
      <influences>
        <influence comment="b" equals="b"/>
        <influence comment="h" equals="h"/>
        <influence name="B" comment="approach channel
            width">
          <formula>2.002+drcal+dBmeas</formula>
          <influences>
            <influence comment="drcal" equals="drcal"/>
            <influence name="dBmeas" comment="measurement">
              <distribution>
                <studentt>
                  <xbar>0.0</xbar>
                  <std>0.0026*0.0026/6</std>
                  <dgf>5</dgf>
                </studentt>
              </distribution>
            </influence>
          </influences>
        </influence>
        <influence name="p" comment="hump height">
          <formula>pnom+drcal+dpmeas</formula>
          <influences>
            <influence name="pnom" comment="Nominal value">
              <distribution>
                <constant>
                  <value parameter="#hh">0.252</value>
                </constant>
              </distribution>
            </influence>
            <influence comment="drcal" equals="drcal"/>
            <influence name="dpmeas" comment="measurement">
              <distribution>
```

```
                  <studentt>
                    <xbar>0.0</xbar>
                    <std>0.0019*0.0019/9</std>
                    <dgf>8</dgf>
                  </studentt>
                </distribution>
              </influence>
            </influences>
          </influence>
        </influences>
      </influence>
    </influences>
  </influence>
</model>
```

Listing B.7: Simulation definition `sim.xml`

```
 <simulation>
  <instances>
    <instance name="Q" model="flowrate"/>
  </instances>

  <calculation>
    <analyze mode="on" histbars="60"/>
    <measurand>
      <formula name="Q">Q</formula>
    </measurand>
  </calculation>
</simulation>
```

APPENDIX C

Definitions and terminology

This work is based on the three documents GUM [37], GS1 [22], and VIM [23]. The vocabulary is used as consistently as possible according to these documents. Although, the work takes its liberties in the language definition in consultation with experts. Different usage is stated on occasion explicitly if it is not clear from context. Definitions in this chapter are annotated if they appear in the same form in one of the documents.

C.1 Metrology

metrology science of measurement and its application (VIM)

measurement set of operations having the object of determining a value of a quantity (GUM); process of experimentally obtaining one or more quantity values that can reasonably be attributed to a quantity (VIM)

measurement model mathematical relation among all quantities known to be involved in a measurement (VIM)

measurement standard realization of the definition of a given quantity, with stated quantity value and associated measurement uncertainty, used as a reference (VIM)

working measurement standard, check standard, control standard measurement standard that is used routinely to calibrate or verify measuring instruments or measuring systems (VIM)

measurement method (method of measurement) logical sequence of operations, described generically, used in the performance of measurements (GUM); generic description of a logical organization of operations used in a measurement (VIM)

measurement scenario actual real-world setting of a measurement taking place in a common, natural environment

series of measurements number of sequential or parallel measurements; alternating reference measurements and sample measurements

repeated measurements number of sequential or parallel measurements with the same sample

measurement procedure set of operations, described specifically, used in the performance of particular measurements according to a given method (GUM); detailed description of a measurement according to one or more measurement principles and to a given measurement method, based on a measurement model and including any calculation to obtain a measurement result (VIM)

measurement process and steps actual realization of a measurement including activities and a time frame; the measurement process may be divided into several steps

influence quantity quantity that is not the measurand but that affects the result of the measurement (GUM)

measurand particular quantity subject to measurement (GUM); quantity intended to be measured (VIM)

equation of the measurand functional relationship of the influence or input quantities; the result is one possible value for the measurand.

input quantity quantity that must be measured, or a quantity, the value of which can be otherwise obtained, in order to calculate a measured quantity value of a measurand (VIM)

influence quantity quantity that, in a direct measurement, does not affect the quantity that is actually measured, but affects the relation between the indication and the measurement result

output quantity quantity to be determined by evaluation of the equation of the measurand (GUM); quantity, the measured value of which is calculated using the values of input quantities in a measurement model (VIM)

(measurement) uncertainty parameter, associated with the result of a measurement, that characterizes the dispersion of the values that could reasonably be attributed to the measurand (GUM); non-negative parameter characterizing the dispersion of the quantity values being attributed to a measurand, based on the information used (VIM)

standard uncertainty measurement uncertainty expressed as a standard deviation (VIM)

expanded measurement uncertainty expanded uncertainty product of a combined standard measurement uncertainty and a factor larger than the number one (VIM)

uncertainty budget statement of a measurement uncertainty, of the components of that measurement uncertainty, and of their calculation and combination (VIM)

GUM uncertainty framework application of the law of propagation of uncertainty and the characterization of the output quantity by a Gaussian distribution or a scaled and shifted t-distribution in order to provide a coverage interval (GS1)

Monte Carlo method method for the propagation of distributions by performing random sampling from probability distributions (GS1)

C.2 Monte Carlo simulation

simulation one performance to evaluate the measurement uncertainty for a measurement scenario; it consists of M trials

trial one evaluation of the equation of the measurand with one set of sampled random numbers for the influence quantities of the equation of the measurand

adaptive Monte Carlo procedure a basic implementation of an adaptive Monte Carlo procedure involves carrying out an increasing number of Monte Carlo trials until the various results of interest have stabilized in a statistical sense (GS1)

C.3 Statistics

probability distribution function giving the probability that a random variable takes any given value or belongs to a given set of values; the probability on the whole set of values of the random variable equals 1 (GUM)

(cumulative) distribution function (CDF) function giving, for every value ξ, the probability that the random variable X be less than or equal to ξ

$$G_X(\xi) = Pr(X \leq \xi)$$

(GUM)

probability density function (PDF) derivative (when it exists) of the distribution function

$$g_X(\xi) = dG_X(\xi)/d\xi.$$

$g_X(\xi)d\xi$ is the "probability element"

$$g_X(\xi) = Pr(\xi < X < \xi + d\xi)$$

(GUM)

coverage interval, uncertainty interval interval containing the value of a quantity with a stated probability, based on the information given (GS1); in this work the term uncertainty interval is used synonymously with coverage interval

coverage probability, level of confidence probability that the value of a quantity is contained within a specified coverage interval (GS1)

shortest coverage interval coverage interval for a quantity with the shortest length (largest value minus smallest value in a coverage interval) among all coverage intervals for that quantity having the same coverage probability (GS1)

C.4 Sensitivity and uncertainty analysis

The definitions for sensitivity and uncertainty analysis are used according to [12] and [68].

uncertainty analysis assess the effects of parameter uncertainties on the uncertainties in calculated results

sensitivity analysis (SA) quantify the effects of parameter variations on calculated results; terms such as influence, importance, ranking by importance and dominance are all related to sensitivity analysis

factor screening extract influential factors in systems with a large number of input quantities

local sensitivity analysis examine local (point) impact of the influence quantities, usually using partial derivatives or analytical methods

global sensitivity analysis apportion the uncertainty to the influence quantities; usually a sampling approach is used

Bibliography

[1] ISO/ IEC 14977:1996. *Information technology — Syntactic metalanguage — Extended BNF*. International Organization for Standardization, Geneva, Switzerland, 1996.

[2] ISO/ TS 21749:2005. *Measurement uncertainty for metrological applications — Repeated measurements and nested experiments*. International Organization for Standardization, Geneva, Switzerland, 2005.

[3] ISO/4359-1983. *Liquid flow measurement in open channels – Rectangular, trapezoidal and U-shaped flumes*. International Organization for Standardization, 1983.

[4] Erwin Achermann. *MUSAC - A Tool for Evaluating Measurement Uncertainty. DISS. ETH Nr. 14573*. Dr. Kovac, 2002.

[5] Alexandre Allard and Nicolas Fischer. Sensitivity analysis in metrology: Study and comparison on different indices for measurement uncertainty. *Advanced Mathematical and Computational Tools in Metrology VIII*, pages 1–6, 2009.

[6] American National Standards Institute. *ANSI X3.135-1992: Information Systems — Database Language — SQL (includes ANSI X3.168-1989)*. American National Standards Institute, 1989.

[7] Robin Barker. *Guide to EUROMETROS: a manual for users, contributors and testers*. Software Support for Metrology, Good Practice Guide No. 5. National Physical Laboratory, Teddington, 2004.

[8] Robin Barker, Peter Harris, and Graeme Parkin. *Development and Testing of Spreadsheet Applications*. Software Support for Metrology, Best Practice Guide No. 7. National Physical Laboratory, Teddington, 2000.

[9] Walter Bich, Maurice G. Cox, and Peter M. Harris. Evolution of the *Guide to the Expression of Uncertainty in Measurement*. *Metrologia*, 43:161–166, 2006.

[10] BIPM. Si base units. Website, 2008. Available online at http://www.bipm.org/en/si/base_units/; visited on June 16th 2008.

[11] Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen, editors. *Extensible Markup Language (XML) 1.0 (Fourth Edition) - W3C Recommendation 16 August 2006*, volume REC-xml-20060816. W3C, 2006.

[12] Dan G. Cacuci. *Sensitivity and uncertainty analysis*, volume 1. Chapman and Hall, London and New York, 2003.

[13] Craig Cleaveland. *Program Generators with XML and Java*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[14] Simon Cowen and Stephen L.R. Ellison. Reporting measurement uncertainty and coverage interval near natural limits. *The Analyst*, 131:710–717, 2006.

[15] Maurice G. Cox and P.M. Harris. *Uncertainty Evaluation*. Software Support for Metrology, Best Practice Guide No. 6. National Physical Laboratory, Teddington, 2004.

[16] Maurice G. Cox and Bernd R. L. Siebert. The use of a Monte Carlo method for evaluating uncertainty and expanded uncertainty. *Metrologia*, 43:178–188, 2006.

[17] Richard Davis. The SI unit of mass. *Metrologia*, 40(6):299–305, 2003.

[18] Steven J. DeRose. *The SGML FAQ Book: Understanding the Foundation of HTML and XML*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[19] Stephen L.R. Ellison, Matthias Rösslein, and Alex Williams, editors. *Eurachem/CITAC Guide: Quantifying Uncertainty in Analytical Measurement*. CITAC Guide number 4. Eurachem, 2000.

[20] David C. Fallside and Priscilla Walmsley, editors. *XML Schema Part 0: Primer Second Edition - W3C Recommendation 28 October 2004*, volume REC-xmlschema-0-20041028. W3C, 2004.

[21] Gerhard Fischer, Elisa Giaccardi, Yunwen Ye, Alistair G. Sutcliffe, and Nikolay D. Mehandjiev. Meta-design: a manifesto for end-user development. *Commun. ACM*, 47(9):33–37, September 2004.

[22] International Organization for Standardization. *Guide to the expression of uncertainty in measurement (GUM)-Supplement 1: Numerical methods for the propagation of distributions*. International Organization for Standardization, Geneva, 2008.

[23] International Organization for Standardization. *International Vocabulary of Metrology – Basic and General Concepts and Associated Terms*. International Organization for Standardization, Geneva, 2008.

[24] The Apache Software Foundation. The Apache Xalan project. Website, 2008. http://xalan.apache.org/.

[25] The Apache Software Foundation. The Apache Xerces project. Website, 2008. http://xerces.apache.org/.

[26] ValiTrace GmbH. Uncertaintymanager. Website, 2009. http://www.uncertaintymanager.org.

[27] Charles F. Goldfarb and Paul Prescod. *Charles F. Goldfarb's XML Handbook, Fifth Edition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2003.

[28] Charles F. Goldfarb and Yuri Rubinsky. *The SGML Handbook*. Oxford University Press, 1990.

[29] Leo Gros, Peter A. Bruttel, and Marcus von Kloeden. *Practical Titration*. Metrohm Ltd., Herisau, Switzerland, 05 2005.

[30] Elliotte Rusty Harold and W. Scott Means. *XML in a Nutshell: A Desktop Quick Reference*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2000.

[31] Peter M. Harris and Maurice Cox. Software Specifications for Uncertainty Evaluation. *Technical Report CMSC 40/04, National Physical Laboratory, Teddington, UK*, March 2004.

[32] Stefan Heidenblut, Rüdiger Kessel, Klaus-Dieter Sommer, and Albert Weckenmann. A modelling concept for practice-oriented evaluation of measurement uncertainty. *Technisches Messen*, 74:494–506, 2007.

[33] Reginald W. Herschy. *Streamflow Measurement - Third edition.* Taylor and Francis, 2009.

[34] P.S.C. Heuberger and P.H.M. Janssen. UNCSAM: A software tool for sensitivity and uncertainty analysis of mathematical models. In J. Grasman and G. van Straten, editors, *Predictability and Nonlinear Modelling in Natural Science and Economics*, pages 362–376. Kluwer Academic Publishers, 1994.

[35] Toshimitsu Homma and Andrea Saltelli. Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17, 1996.

[36] Kaoru Ishikawa. *Introduction to Quality Control.* Productivity Press, New York, January 1990.

[37] ISO, IEC, BIPM, IFCC, IUPAC, IUPAP, and OIML. *Guide to the Expression of Uncertainty in Measurement.* International Organization for Standardization (ISO), Central Secretariat, Geneva, 1 edition, 1993, corrected and reprinted 1995.

[38] P.H.M. Janssen. Assessing sensitivities and uncertainties in models: a critical evaluation. In J. Grasman and G. van Straten, editors, *Predictability and Nonlinear Modelling in Natural Science and Economics*, pages 344–361. Kluwer Academic Publishers, 1994.

[39] Jose Marcos Jurado and Ángela Alcázar. A software package comparison for uncertainty measurement estimation according to GUM. *Accred Qual Assur*, 10(7):373–381, 2005.

[40] Raghu N. Kacker. Bayesian alternative to the ISO-GUM's use of the Welch–Satterthwaite formula. *Metrologia*, 43:1–11, 2006.

[41] Rüdiger Kessel, Raghu Kacker, and Michael Berglund. Coefficient of contribution to the combined standard uncertainty. *Metrologia*, 43(4):S189–S195, 2006.

[42] Shri Krishna Kimothi. *The Uncertainty of Measurements - Physical and Chemical Metrology Impact and Analysis.* ASQ Quality Press, Milwaukee, Wisconsin, 2002.

[43] Dieter Kind. Die Kettenschaltung als Modell zur Berechnung der Messunsicherheit. *PTB Mitteilungen*, Sonderdruck 3/4:66–69, 2001.

[44] Jack P.C. Kleijnen. Sensitivity analysis versus uncertainty analysis: When to use what. In J. Grasman and G. van Straten, editors, *Predictability and Nonlinear Modelling in Natural Science and Economics*, pages 322–331. Kluwer Academic Publishers, 1994.

[45] Donald E. Knuth. *Art of Computer Programming, Volume 2: Seminumerical Algorithms (3rd Edition)*. Addison-Wesley Professional, November 1997.

[46] Donald E. Knuth. *Art of Computer Programming, Volume 3: Sorting and Searching (2nd Edition)*. Addison-Wesley Professional, April 1998.

[47] Averill M. Law. *Simulation, modeling and analysis, 3rd edition*. McGraw-Hill, 2000.

[48] Mike Leber and Stephan Portmann. *JAXFront Developer Manual - Java Client Renderer V2.30*. xentric techology and consulting, Zurich, Switzerland, 2007.

[49] Justin Lipton. Xerlin - opensource extensible xml modeling application. Website, 2008. http://www.xerlin.org and https://sourceforge.net/projects/xerlin/.

[50] James Martin. *Application developement without programming*. Prentice-Hall, Inc., 1982.

[51] James Martin. *Fourth-generation languages. Volume I: principles*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1985.

[52] Marjan Mernik. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344, December 2005.

[53] Martin Müller. *Computational Aspects of Measurement Uncertainty Calculation*. PhD thesis, ETH Zürich, 2009. In preparation.

[54] Martin Müller and Christian Rink. On the convergence of the monte carlo block design. *Metrologia*, 46(5):404–408, 2009.

[55] Martin Müller, Marco Wolf, and Matthias Rösslein. MUSE: computational aspects of a GUM supplement 1 implementation. *Metrologia*, 45(5):586–594, 2008.

[56] Martin Müller, Marco Wolf, and Matthias Rösslein. Measurement uncertainty calculation using *MUSE. Advanced Mathematical and Computational Tools in Metrology VIII*, 2009.

[57] Martin Müller, Marco Wolf, Matthias Rösslein, and Walter Gander. Calculating measurement uncertainties with complex quantities using Monte-Carlo. In *Transverse disciplines in metrology*, Proceeding, 13th International Metrology Congress, pages 603–614. ISTE Ltd., 2007.

[58] David R. Musser. Introspective sorting and selection algorithms. *Software Practice and Experience*, 27:983–993, 1997.

[59] NIST. Certificate srm 350b - benzoic acid (acidimetric). Website, 2009. https://www-s.nist.gov/srmors/view_cert.cfm?srm=350B.

[60] Oracle. Crystal Ball. Website, 2008. http://www.decisioneering.com/.

[61] Martin Otter. Modelica overview, 2009. http://www.modelica.org.

[62] William Press, Saul Teukolsky, William Vetterling, and Brian Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, UK, 2nd edition, 1992.

[63] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.

[64] John A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, Belmont, CA., USA, third edition edition, 2007.

[65] Wolfgang Richter and Ursula Tinner. *Practical aspects of modern titration*. Metrohm, Herisau, Switzerland, 10 2004.

[66] Steward Robinson. Simulation projects: Building the right conceptual model. *Industrial Engineering*, 26(9):34–36, 1994.

[67] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. John Wiley & Sons, Inc., New York, NY, USA, 2008.

[68] Andrea Saltelli, Karen Chan, and E. Marion Scott. *Sensitivity Analysis*. John Wiley & Sons, LTD, West Sussex, England, 2000.

[69] F. E. Satterthwaite. An approximate distribution of estimates of variance components. *Biometrics Bulletin*, 2(6):110–114, 1946.

[70] Heinrich Schwenke. *Abschätzung von Messunsicherheiten durch Simulation an Beispielen aus der Fertigungsmesstechnik*. PhD thesis, Technische Universität Chemnitz, Braunschweig, July 1999.

[71] Bernd R.L. Siebert and Klaus-Dieter Sommer. Weiterentwicklung des GUM und Monte-Carlo-Techniken. *tm - Technisches Messen*, 71(2):67–80, 2004.

[72] Il'ya Meerovich Sobol'. Sensitivity estimates for nonlinear mathematical models. *Mathematical Modeling and Computational Experiments*, 1(4):407–414, 1993.

[73] Klaus-Dieter Sommer and Bernd R.L. Siebert. Praxisgerechtes Bestimmen der Messunsicherheit nach GUM. *Technisches Messen*, 71:52–66, 2003.

[74] Klaus-Dieter Sommer and Bernd R.L. Siebert. Systematische Modellbildung und Grundsätze der Bereichskalibrierung. *Technisches Messen*, 72:258–276, 2005.

[75] Alistair Sutcliffe. Evaluating the costs and benefits of end-user development. In *WEUSE I: Proceedings of the first workshop on End-user software engineering*, pages 1–4, New York, NY, USA, 2005. ACM.

[76] Adriaan M. H. van der Veen. Trends in the certification of reference materials. *Accreditation and Quality Assurance: Journal for Quality, Comparability and Reliability in Chemical Measurement*, 9(4-5):232–236, August 2004.

[77] Brian Wichmann, Robin Barker, Maurice Cox, and Peter Harris. *Measurement System Validation: Validation of Measurement Software*. Software Support for Metrology, Best Practice Guide No. 1. National Physical Laboratory, Teddington, 2000.

[78] Thomas Wiedenhöfer. Softwareunterstützung bei der GUM-konformen Modellierung von Messprozessen. In *VDI - Berichte 1947, Messunsicherheit praxisgerecht bestimmen*, pages 339–346, 2006.

[79] David S. Wile. Supporting the DSL spectrum. *Journal of Computing and Information Technology*, 9(4):263–287, December 2001.

[80] Marco Wolf, Martin Müller, and Matthias Rösslein. Modelling and simulation of complex measurement settings using the Monte-Carlo method. *Technisches Messen*, 74:485–493, 2007.

[81] Marco Wolf, Martin Müller, and Matthias Rösslein. *MUSE* - Messunsicherheit - Simulation und Evaluation basierend auf GS1. In *Messunsicherheit praxisgerecht bestimmen*, 4. Fachtagung Messunsicherheit. VDI Wissensforum GmbH, 2008.

[82] Marco Wolf, Martin Müller, Matthias Rösslein, and Walter Gander. Messunsicherheit – Softwaregestützte Modellierung und Simulation komplexer Messvorgänge. In *VDI – Berichte 1947, Messunsicherheit praxisgerecht bestimmen*, pages 347–356, 2006.

[83] Marco Wolf, Martin Müller, Matthias Rösslein, and Walter Gander. High resolution modeling for evaluation of measurement uncertainty. In *Transverse disciplines in metrology*, Proceeding, 13th International Metrology Congress, pages 615–626. ISTE Ltd., 2007.

[84] Markus Zeier. Der ISO-GUM erhält Verstärkung. *METinfo*, 15(3):9–15, 2008.

Curriculum Vitae

Personal

Name	*Marco Stefan Wolf*
Birth	*October 28, 1975 in Dornbirn, Austria*
Citizenship	*Austrian*

Education

1991-1996	Attending higher technical education institute *Höhere Technische Lehranstalt für Textilbetriebstechnik und Informatik*
1996-2002	Study in computer science at TU-München
2004-2009	Ph.D. student and teaching assistant at the Institute of Computational Science (ICoS) at ETH Zürich

Practical Work

1999-2004	Software developer of UncertaintyManager® at Creasoft AG, Mauren/Liechtenstein
2003-2004	Computer administrator during social service at Caritas Vorarlberg/Austria