Chapter 2

Mathematical Concepts

- Definition of measurement
- Mathematical formulae used to model *known* influences
- Normal distribution modeling *unknown* or *unquantifiable* influences
- Standard deviation is a measure for uncertainty

MATHEMATICAL concepts behind a measurement are described very formally in [2] the so-called GUM. This chapter rephrases some of the statements in the GUM, comments them and draws several consequences and implications on the automated system MUSAC.

2.1 What is a Measurement

In this section we summarize and comment the basic definitions, rules and assumptions on which the GUM [2, p. 31f.] is founded.

2.1.1 Definitions

The GUM defines several terms very precisely. Let's cite some of them here:

(measurable) quantity attribute of phenomenon, body or substance that may be distinguished qualitatively and determined quantitatively.

NOTES

1 The term quantity may refer to a **quantity in a general sense** [see example a)] or to a **particular quantity** [see example b)]

EXAMPLES

a) quantities in a general sense: length, time, mass, temperature, electrical resistance, amount-of-substance concentration;

b) particular quantities:
- length of a rod
- electrical resistance of a given specimen of wire
- amount-of-substance concentration of ethanol in a given sample of wine.

2 Quantities that can be placed in order of magnitude together into **categories of quantities**, for example:
- work, heat, energy
- thickness, circumference, wavelength.

3 **Symbols for quantities** are given in ISO 31.

value (of a quantity) magnitude of a particular quantity generally expressed as a unit of measurement multiplied by a number [. . .]

NOTES

1 The value of a quantity may be positive, negative or zero.

2 The value of a quantity may be expressed in more than one way.

3 The values of quantities of dimension one are generally expressed as pure numbers.

4 A quantity that cannot be expressed as a unit of measurement multiplied by a number may be expressed by reference to a conventional reference scale or to a measurement procedure or to both.

measurement set of operations having the object of determining a value of a quantity.

NOTE: The operations may be performed automatically.

measurement procedure set of operations, described specifically, used in the performance of particular measurements according to a given method

NOTE: A measurement procedure is usually recorded in a document that is sometimes called a "measurement procedure" (or a **measurement method**) and is usually in sufficient detail to enable an operator to carry out a measurement without additional information.

measurand particular quantity subject to measurement

EXAMPLE: vapour pressure of given sample of water at 20 °C.

NOTE: The specification of a measurand may require statements about quantities such as time, temperature and pressure.

result of a measurement value attributed to a measurand, obtained by a measurement

NOTES

1 When a result is given, it should be made clear whether it refers to:

- the indication
- the uncorrected result
- the corrected result

and whether several values are averaged.

2 A complete statement of the result of a measurement includes information about the uncertainty of measurement.

uncertainty (of a measurement) parameter, associated with the result of a measurement, that characterizes the dispersion of the values that could reasonably be attributed to the measurand

NOTES

1 The parameter may be, for example, a standard deviation (or a given multiple of it), or the half-width of an interval having a stated level of confidence.

2 Uncertainty of measurement comprises, in general, many components. Some of these components may be evaluated from the statistical distribution of the result of series of measurements and can be characterized by experimental standard deviations. The other components which can also be characterized by standard deviations, are

evaluated from assumed probability distributions based on experience or other information.

3 It is understood that the result of the measurement is the best estimate of the value of the measurand, and that all components of uncertainty, including those arising from systematic effects, such as components associated with corrections and reference standards, contribute to the dispersion.

These definitions are not undisputed: E.g. Feller [20] and others criticize the definition of a measurable quantity having two different meanings: a quantity in a *general sense* is a very abstract notion of physical phenomena such as length, time, energy which are difficult to define themselves. Richard Feynman explained the concept of energy in [21]: "... there is a certain quantity, which we call energy, that does not change in the manifold changes which nature undergoes. That is a most abstract idea, because it is a mathematical principle... It is not a description of a mechanism, or anything concrete; it is just a strange fact that we can calculate some number and when we finish watching nature going through her tricks and calculate the number again, it is the same. ...It is important to realize that in physics today, we have no knowledge of what energy *is*." On the other hand, the term quantity may also refer to a *particular quantity* such as the "electrical resistance of a given specimen of wire" according to the metrological definition. This is a very different level of abstraction. Unfortunately, the observable phenomenon and its abstract principle are not clearly distinguished. This may sometimes lead to confusion: The length of two particular sticks are two different quantities which in turn are different to the quantity "length".

However, one is able to understand the crucial and central message of the GUM [2, § 3.1.2] with the definitions cited above:

> In general, the result of a measurement is only an approximation or estimate of the value of the measurand and thus is complete only when accompanied by a statement of the uncertainty of that estimate.

In other words, the result of a measurement relates to an estimated mean and the uncertainty of the measurement refers to the estimated variance of a random variable. A comment on terminology: In this text the shorter term "measurement" is often used instead of the correct but clumsy "result of a measurement". The context will (hopefully) always clarify the meaning.

2.2 Different Approaches

Several methods are proposed in order to tackle the problem of *how to compute measurement uncertainty effectively and accurately*. They can be classified into two groups of different approaches. The one proposed by the GUM can roughly be summarized as *linearizing the problem* whereas the second approach can be subsumed as *simulating the problem*. The simulating approach is not explicitly mentioned in the first edition of the GUM and currently there are no plans for a second edition. But there are plans for supplemental guides. One of which will be concerned with the propagation of distributions. It will recommend the *Monte Carlo Simulation*, a computer-intensive process that samples many times from the input distributions, and, by evaluating the model for each sample, provides a distribution for the output [15]. This approach of simulating the input has become feasible only in modern times thanks to the tremendous improvement of computer speed.

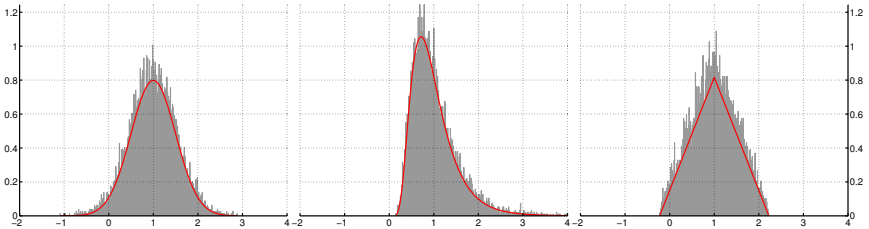
The following sections explain the reasoning and mathematics behind the linearizing approach, except section 2.2.5 which explains the Monte Carlo Simulation approach.

2.2.1 Random variables

Every single result of a measurement is unique and is subject to unique circumstances. Therefore no two executions of a measurement procedure yield the exact same result. One can imagine this fact as if infinitely many influences alter the result. This is why a result of a measurement can always be viewed as a *random variable*. A random variable is a statistical concept for describing situations such as the one above. Because there is no single value representing the "value of a random variable" one is interested in statistical statements on a random variable such as what is the *mean*, the *variance* or the *distribution* of its observed values. We will denote random variables by capital symbols such as X and Y. Instances of observed values of a random variable will be denoted by x_i or y_i. Conceptually, a random variable contains infinitely many observed values but for practical reasons there is often a finite number n of instances available. Think of a random variable as a basket containing n observed values.

Figure 1 shows different histograms of observed values. Observe that each of the three samples has the same mean $\mu = 1.0$ and the same vari-

Figure 1 Sampled random variables with different distributions. The mean andthe variance are in each case the same $\mu = 1.0$ and $\sigma^2 = 0.25$.



a) Normal b) Lognormal c) Triangular

ance $\sigma^2 = 0.25$. The diagrams are generated using 10000 observed random values which are depicted in a histogram of 500 columns. The height of the columns is normalized such that the area over all columns is 1. That is, as the number of observed values and the number of histogram columns grow big, these normalized histograms approach the *probability density function* of the underlying distribution which are also drawn in the histograms of Figure 1.

Mean The *mean* of a random variable X is a number describing the average of all observed values x_i. It is also called the *expected value* of X and it is denoted by $\mathrm{E}[X] = \mu_X$. It is a single number that represents the average of infinitely many instances x_i. If we are observing n instances of X then the best we can do is to estimate the mean μ_X by

$$\mathrm{E}[X] = \mu_X \approx \bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{2.1}$$

With this definition the following properties of the expected value are obvious. They will be used in the following sections without any further reference:

$$\mathrm{E}[\alpha] = \alpha \qquad \text{and} \qquad \mathrm{E}[\alpha X + Y] = \alpha\,\mathrm{E}[X] + \mathrm{E}[Y]\,,$$

where $\alpha \in \mathbb{R}$ and X and Y are arbitrary random variables.

Variance The *variance* $\mathrm{Var}(X) = \sigma_X^2$ of a random variable X is a number describing the scattering of the observed values around the mean μ_X. The

larger the variance σ_X^2 is, the wider the instances are distributed around μ_X. The variance is the expected value of a random variable S defined as $S = \big(X - \mu_X\big)^2$. In other words: it is the average of the squared distance from the mean.

$$\sigma_X^2 = \mathrm{Var}(X) = \mathrm{E}\Big[\big(X - \mu_X\big)^2\Big] \tag{2.2}$$

Note the following important property:

$$\begin{aligned}
\mathrm{Var}(X) &= \mathrm{E}\Big[\big(X - \mu_X\big)^2\Big] \\
&= \mathrm{E}\Big[X^2 - 2X\mu_X + \mu_X^2\Big] \\
&= \mathrm{E}\Big[X^2\Big] - 2\mu_X\,\mathrm{E}[X] + \mu_X^2 \\
&= \mathrm{E}\Big[X^2\Big] - 2\mu_X^2 + \mu_X^2 \\
\mathrm{Var}(X) &= \mathrm{E}\Big[X^2\Big] - \mu_X^2. \qquad (2.3)
\end{aligned}$$

If we are given n observations of X an estimate for σ_X^2 can be found by applying Equation (2.2) directly to the true mean μ_X if it is known: $\mathrm{Var}(X) \approx {}^1\!/_n \sum_{i=1}^n \big(x_i - \mu_X\big)^2$. But the true mean is almost never known, rather we can compute an estimate for the mean $\mu_X \approx \bar{X}$ using Equation (2.1). In that case the best estimate for the variance is found by:

$$\mathrm{Var}(X) = \sigma_X^2 \approx s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}\big(x_i - \bar{X}\big)^2. \tag{2.4}$$

The division by $n-1$ compensates for the fact that $\bar{X}$ is an estimate rather than the true mean [17, 33]. Or one can apply Property (2.3). Compensating for the difference between true and estimated mean then again leads to

$$\mathrm{Var}(X) = \sigma_X^2 \approx s_x^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2\right). \tag{2.5}$$

Covariance The *covariance* between two random variables X and Y expresses the relationship between the two of them. It is defined by

$$\mathrm{Cov}(X, Y) = \mathrm{E}\big[\big(X - \mu_X\big)\big(Y - \mu_Y\big)\big].$$

Observe that $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$ and $\mathrm{Cov}(X,Y) = \mathrm{Cov}(Y,X)$. Independent random variables have a covariance equal to zero. The *correlation* $\rho_{X,Y}$ expresses the same relationship, but it is normalized such that it takes values between -1 and 1:

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y}. \tag{2.6}$$

The bigger the correlation's absolute value is, the stronger is the relationship between the two random variables. If the correlation, and therefore the covariance, is negative, then any two corresponding observed values are likely to lie on opposite sides of the respective mean. If the correlation is positive, then two corresponding observed values are probably on the same side of their mean.

For an estimation of the covariance we observe the following property:

$$\mathrm{E}\big[\big(X - \mu_X\big)\big(Y - \mu_Y\big)\big] = \mathrm{E}[XY - \mu_X \mu_Y] = \mathrm{E}[XY] - \mu_X \mu_Y \tag{2.7}$$

This means that given two samples for two random variables of size n their covariance can be estimated either by

$$\mathrm{Cov}(X,Y) = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i y_i - n\bar{X}\bar{Y}\right) \tag{2.8}$$

or by

$$\mathrm{Cov}(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}\big(x_i - \bar{X}\big)\big(y_i - \bar{Y}\big). \tag{2.9}$$

As before, the division by $n-1$ instead of by n is the best correction for the fact that we are using estimated instead of the true means.

There is, of course, a lot more to be said on random variables, statistics and so on. But I would rather point the interested reader to some good literature like [17, 28, 29, 33, 40].

Accurate computation of mean, variance and covariance In order to calculate the estimated mean a first approach would naïvely iterate over the data and add each summand to the partial sum $T_{i,j} = \sum_{k=i}^{j} x_k$ finally computing the total sum $T = T_{1,n}$. This means, especially for large i, that at step i adding x_{i+1} to $T_{1,i}$ eventually is like adding a drop of water to the sea. The relative error bound for $T = T_{1,n}$ computed in this way is basically $n\varepsilon$,

n being the number of summands to add, and ε being a small error due to floating point representation if nothing else, the so called *machine epsilon*. In other words, the error for the sum grows with the number of summands involved. A good idea would be to sort the data and start summing up with the smallest numbers. The idea of this approach is to keep the intermediate sum small and thereby minimizing the absolute error per step. But still the partial sum eventually might grow so big that the addition of one single number would not affect the result, again due to round off. This is a fundamental problem when building a sum of many summands. William Kahan proposed Algorithm 1 to compute a sum of many summands. Its

Algorithm 1 Kahan's Summation Formula

function $T = \mathsf{Sum}(\boldsymbol{x}, n)$
  — computation of $T = \sum_{i=1}^{n} x_i$ —
  $T \leftarrow 0$
  $c \leftarrow 0$
  forall $1 \leq i \leq n$ **do**
    $y \leftarrow x_i - c$ — *add the previous cut off error* —
    $T_{\text{old}} \leftarrow T$
    $T \leftarrow T + y$
    $c \leftarrow (T - T_{\text{old}}) - y$ — *fetch the (negative) cut off error* —
  end forall

idea is to store the round off error in each step and add it to the subsequent summand. This reduces the relative error bound for T to about 2ε which is a dramatic improvement compared to the naïve algorithm. See [26] and [30] for a thorough discussion of Kahan's summation formula.

In order to calculate the variance by Equation (2.4) the data must be traversed twice: once for finding $\bar{X}$ and a second time for computing $\sum\left(x_i - \bar{X}\right)^2$. In order to avoid these two passes statistical textbooks propose to reorganize it into Equation (2.5), building $\sum x_i$ and $\sum x_i^2$ in one pass. But this is deprecated, because of the possibly catastrophic numerical cancellation happening in the subtraction operation. Chan, Golub and LeVeque [14] propose to calculate the variance and mean in a recursive manner as shown in Algorithm 2. It is based on the so called *pairwise summation* — it deserves its name because pairs of similar magnitude are added at each recursion

level — and the *updating formula* by Youngs and Cramer [44]

$$\begin{aligned} T_{1,j} &= T_{1,j-1} + x_j \\ S_{1,j} &= S_{1,j-1} + \frac{1}{j(j-1)}(jx_j - T_{1,j})^2 \end{aligned} \tag{2.10}$$

with $T_{1,1} = x_1$ and $S_{1,1} = 0$ where $S_{ij} = \sum_{k=i}^{j}\left(x_k - {}^1/_{(j-i+1)}T_{ij}\right)^2$ is the sum of squares in analogy to T_{ij} as above. Chan, Golub and LeVeque show that this approach is stable moreover they conjecture that the error bound is $\kappa\log(n)\varepsilon$ whereas Equation (2.4) is stable with an error bound $\kappa^2 n\varepsilon$ and Equation (2.5) is even unstable. Barlow [7] proves that the error bound of Algorithm 2 is indeed $\kappa\log(n)\varepsilon$ as conjectured, where $\kappa = \|\boldsymbol{x}\|_2/\sqrt{S}$ is the condition number as defined in [14] for the calculation of the sample variance and $S = \sum(x_i - \bar{X})^2$.

In the paragraph above we showed very briefly the problem of calculating large sums i.e. the mean and the variance of a sample $\boldsymbol{x}$ along with some suggested solutions and refinements. In the context of MUSAC the goal is to have a fast and accurate algorithm, where memory and data availability is no problem. Several experiments showed that choosing the best algorithm is not obvious. Table 1 lists several criteria of various tested algorithms. The algorithms compared each perform the following four tasks:

1. find $x_{\min} = \min_i x_i$
2. find $x_{\max} = \max_i x_i$
3. compute $T = \sum_i x_i$
4. compute $S = \sum_i (x_i - \bar{X})^2$

The performance is given in milli seconds on a Pentium III 750 MHz. The two columns show different performances. The first column shows the time used if the executable code is not optimized by the compiler, the second column shows the time used, if all speed relevant optimizations of the compiler are turned on. Care must be taken not to "over-optimize" the executable code which can render Kahan's summation formula futile. The last column shows the relative error of $|(S_{\text{sort}} - S)/S_{\text{sort}}|$, where S_{sort} is the exact result obtained by using a two pass algorithm sorting the data in each pass and using Kahan's summation. S is the result of the examined algorithm. Finally, the

Algorithm 2 Recursive Calculation of Variance and Mean

function $\langle m, v\rangle = \mathsf{var}(\boldsymbol{x})$
— $\boldsymbol{x}$: sample $\{x_i\}$, m: $\mathrm{E}[\boldsymbol{x}]$, v: $\mathrm{Var}(\boldsymbol{x})$ —
$n \leftarrow \mathsf{length}(\boldsymbol{x})$
$\langle T, S\rangle \leftarrow \mathsf{var}_{\mathsf{rec}}(\boldsymbol{x}, 1, n)$ — call the recursive working horse —
$m \leftarrow T/n$
$v \leftarrow S/(n-1)$

function $\langle T, S\rangle = \mathsf{var}_{\mathsf{rec}}(\boldsymbol{x}, i, j)$
— T: $\sum_{k=i}^{j} x_k$, S: variance of subsample $\sum_{k=i}^{j}(x_k - T/(j-i+1))^2$ —
if $i = j$ **then**
$T \leftarrow x_i$
$S \leftarrow 0$
else
$k \leftarrow \lfloor (i+j)/2 \rfloor$
$m \leftarrow k - i + 1$
$n \leftarrow j - k$
$\langle T_1, S_1\rangle \leftarrow \mathsf{var}_{\mathsf{rec}}(\boldsymbol{x}, i, k)$
$\langle T_2, S_2\rangle \leftarrow \mathsf{var}_{\mathsf{rec}}(\boldsymbol{x}, k+1, j)$
$T \leftarrow T_1 + T_2$
if $m = n$ **then**
$S \leftarrow S_1 + S_2 + (T_1 - T_2)^2/2m$ — *save a few flops* —
else
$S \leftarrow S_1 + S_2 + \frac{m(n/mT_1 - T_2)^2}{n(m+n)}$
end if
end if

data for these experiments were a sample of 10^6 normally distributed random numbers with mean 1 and $\sigma = 10^{-8}$. Admittedly, this example is fairly extreme but it makes the differences more obvious. In more moderate situations (bigger σ) the relative error of S generally becomes smaller. The gist of

Table 1 Algorithms for computing a large sum

Algorithm	relative error bound for T	relative error bound for S	performance not	performance opt	relative error of S
one pass, Equation (2.5)	$n\varepsilon$	$n\kappa^2\varepsilon$	451	130	∞
recursive pairwise summation	$\log(n)\varepsilon$	$\log(n)\kappa\varepsilon$	3184	932	3.1×10^{-11}
iterative pairwise summation	$\log(n)\varepsilon$	$\log(n)\kappa\varepsilon$	2153	501	8.6×10^{-12}
simple updating using (2.10)	$n\varepsilon$	$n\kappa\varepsilon$	470	361	1.5×10^{-9}
updating with Kahan's summation	2ε	$2\kappa\varepsilon$	581	450	7.6×10^{-12}
simple two pass	$n\varepsilon$	$n\varepsilon + n^2\kappa^2\varepsilon^2$	461	220	7.0×10^{-12}
two pass with Kahan's summation	2ε	$2\varepsilon + \mathcal{O}(\kappa^2\varepsilon^2)$	561	310	0
two pass, sorting, Kahan's summation	1ε	1ε	89058	6049	–

the experiments shown in Table 1 is that the two pass algorithm is not only the most accurate but also quicker than any other comparably accurate algorithm. It is quite noteworthy that the updating algorithms are slower than the two pass algorithms. This is attributed to the fact that the two pass algorithm uses two but particularly simple and thus very fast loops over the data rendering the two pass algorithm even faster than the updating one pass algorithm. Note, that the result for S obtained by the school book one pass algorithm according to Equation (2.5) was negative which is senseless for a

variance. Also note that the recursive and the iterative implementation of the *pairwise summation* yield slightly different results. This is due to the fact that the iterative implementation does not build the partial sums in *exactly* the same manner as the recursive implementation does. But it is very obvious that the overhead needed for storing and managing the partial sums is quite expensive. Therefore, Algorithm 3 is used in the MUSAC-system for finding the four characteristic numbers as stated on page 18 in Monte Carlo Simulations.

Algorithm 3 Mean, Variance, Minimum and Maximum of a Sample

function $\langle m, v, a, b\rangle = \mathsf{update}(\boldsymbol{x})$
— m : mean, v: variance, a :minimum, b: maximum of sample $\boldsymbol{x}$ —
$a \leftarrow x_1, \quad b \leftarrow x_1, \quad T \leftarrow 0, \quad S \leftarrow 0$ — *initializations* —
$c \leftarrow 0, \quad n \leftarrow \mathsf{length}(\boldsymbol{x})$ — *temporary variables* —
forall $1 \leq i \leq n$ **do**
  $y \leftarrow x_i$
  $a \leftarrow \min(a, y), \quad b \leftarrow \max(b, y)$
  $y \leftarrow y - c$ — *Kahan's summation* —
  $T_{\text{old}} \leftarrow T$
  $T \leftarrow T + y$
  $c \leftarrow (T - T_{\text{old}}) - y$ — *do* not *"optimize" this statement* —
end forall
$m = {}^{T}/_{n}$
$c \leftarrow 0$
forall $1 \leq i \leq n$ **do**
  $y \leftarrow x_i - m, \quad y \leftarrow y^2$ — *simple and quick instructions* —
  $y \leftarrow y - c$ — *Kahan's summation again* —
  $T_{\text{old}} \leftarrow T$
  $T \leftarrow T + y$
  $c \leftarrow (T - T_{\text{old}}) - y$
end forall
$v \leftarrow {}^{S}/_{(n-1)}$ — *c.f. Equation* (2.4) —

For computing the covariance Equation (2.8) is numerically unstable for the same reason that Equation (2.5) is unstable whereas Equation (2.9) is stable. The experiments for a quick and accurate computation of the variance

showed that the two pass algorithm is not only very accurate but also comfortably quick. Thus we apply the same technique to the calculation of the covariance. So in the first pass we calculate the means of the two samples.

Note, that since the means are calculated in a first pass, we could shift the data by their respective mean rendering the body of the double loop as simple as $S \leftarrow x_i y_i$. But experiments showed that the access to the additional memory for storing the shifted data is by far more time consuming than executing the subtraction operation within the loop. Thus Algorithm 4 turned out to be the fastest as well as accurate for computing the covariance.

Algorithm 4 Calculation of the Covariance

function $c = \mathsf{cov}(\boldsymbol{x}, \boldsymbol{y})$
  $c \leftarrow 0, \quad n \leftarrow \mathsf{length}(\boldsymbol{x})$ — *initialization* —
  $\bar{X} \leftarrow \mathsf{mean}(\boldsymbol{x})$ — *first pass to find means* —
  $\bar{Y} \leftarrow \mathsf{mean}(\boldsymbol{y})$
  forall $1 \leq i < n$ **do**
    $c \leftarrow c + (x_i - \bar{X})(y_i - \bar{Y})$ — *using Kahan's summation once more* —
  end forall
  $c \leftarrow {}^{c}/_{n-1}$

2.2.2 Mathematical Model

To repeat the execution of a measurement n times is in most cases infeasible and often it is even impossible. Additionally, the result of a measurement Y most often depends on many other quantities X_i. The GUM assumes that this dependence can be modeled using mathematical functions. Thus any measurement can be expressed through a functional relationship

$$Y = f(X_1, X_2, \cdots, X_n), \tag{2.11}$$

where the X_i are all quantities that influence the measurement Y. It is important that the effects of the influences X_i are known or at least can be modeled by a mathematical function as well.

Law of Propagation of Uncertainty Consider a quantity Y depending on n influences as in Equation (2.11). We expand f around the expected value of X_i i.e. $\mathrm{E}[X_i] = \mu_i$ in a Taylor series. This yields

$$Y = f(\mu_1, \mu_2, \ldots, \mu_n) + \sum_{i=1}^{n} \frac{\partial f}{\partial X_i}(X_i - \mu_i) + \mathcal{O}\left(\sum (X_i - \mu_i)^2\right), \quad (2.12)$$

and for

$$\begin{aligned}\mathrm{E}[Y] &\approx \mathrm{E}\left[f(\mu_1, \mu_2, \ldots, \mu_n) + \sum_{i=1}^{n} \frac{\partial f}{\partial X_i}(X_i - \mu_i)\right] \\ &= \mathrm{E}[f(\mu_1, \mu_2, \ldots, \mu_n)] + \sum_{i=1}^{n} \frac{\partial f}{\partial X_i}\mathrm{E}\left[(X_i - \mu_i)\right] \\ &= f(\mu_1, \mu_2, \ldots, \mu_n),\end{aligned}$$

where $\partial f / \partial X_i$ is the derivative of f with respect to X_i evaluated at $\mu_1, \mu_2, \ldots, \mu_n$. The GUM assumes that higher order terms are mostly negligible. The first order approximation for $\mathrm{E}[Y]$ is

$$\mathrm{E}[Y] = \mu_Y \approx \bar{Y} = f(\mu_1, \mu_2, \ldots, \mu_n). \quad (2.13)$$

In Equation (2.12) we cut the Taylor series expansion after the linear term. The next following term of the expansion is:

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial^2 f}{\partial X_i \partial X_j}(X_i - \mu_i)(X_j - \mu_j) + \mathcal{O}\left(\sum (X_i - \mu_i)^3\right). \quad (2.14)$$

We now get an additional term which in turn will lead to a better approximation for the expected value of $\mathrm{E}[Y]$ as follows:

$$\begin{aligned}\mathrm{E}[Y] &= \mu_Y \\ &\approx \bar{Y} = \mathrm{E}[f(\mu_1, \mu_2, \ldots, \mu_n)] + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial^2 f}{\partial X_i \partial X_j}\mathrm{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right] \\ &= f(\mu_1, \mu_2, \ldots, \mu_n) + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial^2 f}{\partial X_i \partial X_j}\mathrm{Cov}(X_i, X_j) \quad (2.15)\end{aligned}$$

Thus we get a second order approximation for the expectation of Y. The additional term of Equation (2.15) may be split into two sums. One sum is

taken over the variances and the other is taken over the covariances of X_i, this leads to the following expression for the second order approximation:

$$\mathrm{E}[Y] \approx f(\mu_1, \mu_2, \ldots, \mu_n) + \frac{1}{2}\sum_{i=1}^{n} \frac{\partial^2 f}{\partial X_i^2} \operatorname{Var}(X_i) + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \frac{\partial^2 f}{\partial X_i \partial X_j} \operatorname{Cov}\big(X_i, X_j\big) .$$

In the same way we need to estimate $\operatorname{Var}(Y)$ the variance of Y, with Y again given as in Equation (2.12). Substituting the Taylor Series Expansion of f (2.12) in Property (2.3) yields

$$\begin{aligned}
\operatorname{Var}(Y) &= \mathrm{E}\Big[\big(Y - \mu_Y\big)^2\Big] \\
&\approx \mathrm{E}\left[\left(f(\mu_1, \mu_2, \ldots, \mu_n) + \sum_{i=1}^{n} \frac{\partial f}{\partial X_i}\big(X_i - \mu_i\big) - \mu_Y\right)^2\right] \\
&\approx \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial f}{\partial X_i}\frac{\partial f}{\partial X_j} \mathrm{E}\big[\big(X_i - \mu_i\big)\big(X_j - \mu_j\big)\big] ,
\end{aligned}$$

which — after splitting the sum into variance and covariance terms — leads to the following approximation for $\operatorname{Var}(Y)$ as stated in [2, §5.2.2]:

$$\operatorname{Var}(Y) \approx \sum_{i=1}^{n} \left(\frac{\partial f}{\partial X_i}\right)^2 \operatorname{Var}(X_i) + 2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \frac{\partial f}{\partial X_i}\frac{\partial f}{\partial X_j} \operatorname{Cov}\big(X_i, X_j\big) . \tag{2.16}$$

Expanding the Taylor series of Y^2 up to quadratic order leads to (now using the quadratic order approximation (2.15) for $\mathrm{E}[Y]$) the following additional term to be added to Equation (2.16).

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \left(\frac{1}{2}\left(\frac{\partial^2 f}{\partial X_i \partial X_j}\right)^2 + \frac{\partial f}{\partial X_i}\frac{\partial^3 f}{\partial X_i \partial X_j^2}\right) \operatorname{Var}(X_i) \operatorname{Var}\big(X_j\big) . \tag{2.17}$$

The GUM assumes that first order terms are in most cases sufficient. But it is not entirely clear whether or not higher order terms should be used. On page 19, § 5.1.2 the GUM [2] reads: "When the nonlinearity of f is significant, higher-order terms in the Taylor series expansion must be included in the expression for $\operatorname{Var}(Y)$ ...". Unfortunately the GUM does not say what exactly a *significant* nonlinearity of f really is. But it states, that "...when the distribution of X_i is symmetric about its mean, the most im-

portant terms of next highest [higher] order to be added to terms of Equation (2.16) are (2.17)". Under the constraints of symmetric distributions the second order estimate for $\mathrm{Var}(Y)$ is therefor the sum of (2.16) and (2.17).

Occasionally, one is interested in the covariance of two random variables whose relation is known. Suppose we are looking for the covariance of two random variables that are constructed by different combinations of one random variable, i.e. $\mathrm{Cov}\big(f(X), g(X)\big)$, or we might be asking for the covariance between the random variables X and $Z = h(X, Y)$ which is constructed using X and Y with a known covariance $\mathrm{Cov}(X, Y) = c$. In the next two paragraphs we will derive expressions for the two cases in order to show how these questions can be answered mechanically. For simplicity we will stick to the first order approximations but expressions for the second order approximation can be found analogously. As a first step, let us write down the first order approximations of the functions involved:

$$\begin{aligned} F = f(X) &\approx \bar{f} + \frac{df}{dX}\big(X - \bar{X}\big) \\ G = g(X) &\approx \bar{g} + \frac{dg}{dX}\big(X - \bar{X}\big) \\ H = h(X, Y) &\approx \bar{h} + \frac{\partial h}{\partial X}\big(X - \bar{X}\big) + \frac{\partial h}{\partial Y}\big(Y - \bar{Y}\big), \end{aligned}$$

where $\bar{f} = f(\bar{X})$ and so forth. In the first example we are looking for

$$\begin{aligned} \mathrm{Cov}\big(F, G\big) &= \mathrm{E}[FG] - \mathrm{E}[F]\,\mathrm{E}[G] \\ &\approx \mathrm{E}\left[\left(\bar{f} + \frac{df}{dX}\big(X - \bar{X}\big)\right)\left(\bar{g} + \frac{dg}{dX}\big(X - \bar{X}\big)\right)\right] - \bar{f}\bar{g} \\ &= \mathrm{E}\left[\bar{f}\bar{g} + \bar{g}\frac{df}{dX}\big(X - \bar{X}\big) + \bar{f}\frac{dg}{dX}\big(X - \bar{X}\big) + \frac{df}{dX}\frac{dg}{dX}\big(X - \bar{X}\big)^2\right] - \bar{f}\bar{g} \\ &= \bar{f}\bar{g} + \big(\bar{f}\frac{dg}{dX} + \bar{g}\frac{df}{dX}\big)\,\mathrm{E}\big[\big(X - \bar{X}\big)\big] + \frac{df}{dX}\frac{dg}{dX}\,\mathrm{E}\Big[\big(X - \bar{X}\big)^2\Big] - \bar{f}\bar{g} \\ &= \frac{df}{dX}\frac{dg}{dX}\,\mathrm{Var}\big(X\big). \end{aligned}$$

We only used Property (2.3) and the fact that $\mathrm{E}\big[\big(X - \bar{X}\big)\big] = 0$. In the second

example we make additional use of Property (2.7) to derive

$$\begin{aligned}
\operatorname{Cov}\big(X,H\big) &= \operatorname{E}[XH] - \operatorname{E}[X]\operatorname{E}[H] \\
&\approx \operatorname{E}\left[X\Big(\bar{h} + \frac{\partial h}{\partial x}(X-\bar{X}) + \frac{\partial h}{\partial y}(Y-\bar{Y})\Big)\right] - \bar{X}\bar{h} \\
&= \operatorname{E}\left[\bar{h}X + \frac{\partial h}{\partial x}X(X-\bar{X}) + \frac{\partial h}{\partial y}X(Y-\bar{Y})\right] - \bar{X}\bar{h} \\
&= \bar{X}\bar{h} + \frac{\partial h}{\partial x}\operatorname{E}\big[X(X-\bar{X})\big] + \frac{\partial h}{\partial y}\operatorname{E}\big[X(Y-\bar{Y})\big] - \bar{X}\bar{h} \\
&= \frac{\partial h}{\partial x}\Big(\operatorname{E}\Big[X^2\Big] - \bar{X}^2\Big) + \frac{\partial h}{\partial y}\Big(\operatorname{E}[XY] - \bar{X}\bar{Y}\Big) \\
&= \frac{\partial h}{\partial x}\operatorname{Var}(X) + \frac{\partial h}{\partial y}\operatorname{Cov}(X,Y)\,.
\end{aligned}$$

In the general case the following holds

$$\operatorname{Cov}\big(f(x_1,\ldots,x_n), g(y_1,\ldots,y_m)\big) \approx \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{\partial f}{\partial x_i}\frac{\partial g}{\partial y_j}\operatorname{Cov}\big(x_i,y_j\big)\,. \quad (2.18)$$

Fundamental Steps Equipped with this technique the general procedure for evaluating uncertainty according to the GUM is obvious:

- Build a mathematical model for the measurement using all possible (i.e. quantifiable) influences on the result.
- Determine the uncertainty of the result by applying the law of uncertainty propagation as shown in the previous sections.

The EURACHEM Guide [18, p. 12] expands them into the following four steps to be executed for every measurement:

1. Specify the measurement
2. Identify uncertainty sources
3. Quantify all influences
4. Determine the result of measurement and the measurement uncertainty

2.2.3 ISO Guide: First Order Approximation

The GUM proposes to apply the *law of propagation of uncertainty* in order to determine the measurement uncertainty of any measurement. It is based on various assumptions on the distributions of the random variables. Furthermore it assumes that the functions involved are sufficiently linear, that is, it assumes that nonlinearities can safely be neglected. Once these assumptions are accepted, the process for estimating the measurement uncertainty is indeed straightforward. Given the function

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

one applies Equation (2.14) for the estimation of $\mathrm{E}[Y]$ and Equation (2.16) for the estimation of the variance $\mathrm{Var}(Y)$ both up to first order. This process is quite a tedious and mechanical task, even more so in the case of second order approximations, c.f. Equations (2.15) and (2.17). Still, it should be mentioned that apart from these simplifying assumptions the main problem is the process of constructing an appropriate function $f(\cdot)$ that describes the physical laws of the measurement to the level of detail needed for the purpose of the measurement.

2.2.4 Simple Rules

After considering a variety of measurements it turns out that there is often no need to calculate the derivatives of the measurement function explicitly. The GUM [2, § 5.1.3, § 5.1.6] proposes the application of the following simple rules:

$$\text{if} \quad Z = X \pm Y \quad \text{then} \quad \begin{cases} \mathrm{E}[Z] \leftarrow \mathrm{E}[X] \pm \mathrm{E}[Y] \\ \mathrm{Var}(Z) \leftarrow \mathrm{Var}(X) + \mathrm{Var}(Y) \end{cases}$$

$$\text{if} \quad Z = X \cdot Y \quad \text{then} \quad \begin{cases} \mathrm{E}[Z] \leftarrow \mathrm{E}[X] \cdot \mathrm{E}[Y] \\ \mathrm{Var}(Z) \leftarrow \bar{Y}^2 \,\mathrm{Var}(X) + \bar{X}^2 \,\mathrm{Var}(Y) \end{cases}$$

In other words, the variance and the mean can be calculated almost simultaneously. *But* these rules are only correct in the sense of first order approximation if

1. there are no covariances between the input variables, *and*

2. the input variables affect the measurement in exactly one distinct way only.

As an example for the second point, consider the case of $Z = X \cdot X$. The simple rules would not take into account the correlation between X and X which happens to be as strong a correlation as can be. As a rule of thumb these very simple rules cannot be applied wherever one input variable appears at more than once in the model formula and it cannot be applied if covariances between input variables are considered. And finally these simple rules cannot be applied when functions such as $\exp(\cdot)$, $\log(\cdot)$, $\sqrt{\cdot}$, ... are used.

2.2.5 Monte Carlo Simulation

In some cases the assumptions of Section 2.2.3 are too restrictive. In such cases other techniques must be used to reduce the errors induced by the assumptions. One way would be to employ higher order approximations. This approach would reduce errors due to nonlinearity in the model function $f(\cdot)$, but it still relies on the fact that distributions of the input variables would be symmetric. It turns out that most measurements are deliberately tuned to be as linear as possible, so that the need for higher order approximation due to nonlinearities is practically nonexistent. The reason for unacceptable errors produced by the method of Section 2.2.3 are due to violations of the assumed symmetric distributions. They appear in situations where a variable refers to a number for example close to 0% or close to 100%. E.g. it does not makes sense to talk about a "normally distributed concentration with an average of 0.1% and a variance of 0.05%" because this would imply that with some considerable and non vanishing probability the "observed values" of that concentration are negative which is physical nonsense. Still, it might be the case that the concentration has indeed an observed mean of 0.1% and an observed variance of 0.05%, but the distribution of that concentration cannot be normal. By physical common sense it must be somehow skewed; as a matter of fact the most probable value for the concentration practically never coincides with the expected value as is

visualized in Figure 1 on page 14.

The expression *Monte Carlo Simulation* stands for *simulating true values*. The Monte Carlo Simulation is as simple as effective. It is based on the assumption that each *observed value* of a random variable can be treated as if it were the *true value*. The simulation therefore proceeds in the following steps:

- Generate n random values for each input variable
- Execute the model function $f(\cdots)$ for each observed value assuming it to be the "true value", thus generating n "true values" for Y which according to this assumption are interpreted as observed values of Y.

Now, in order to estimate the mean $\mathrm{E}[Y]$ and the variance $\mathrm{Var}(Y)$ Equations (2.1) and (2.4) can be applied. This method is indeed very simple in principle.

But let's see how big n must be in order to yield reasonably accurate estimates! The expression for $\bar{X}$ in Equation (2.1) is itself a random variable since we pick at random n values out of the infinitely many observable values of X. The variance of the random variable $\bar{X}$ is proportional to $1/\sqrt{n}$ as explained in any handbook on statistics [19]. This means increasing the number of observed values n by a factor of 100 decreases the expected difference $\bar{X} - \mu_X$ by a factor of 10. This factor is equally bad for the estimation of the variance since the variance of the variance estimated according to Equation (2.4) is proportional to $1/\sqrt{n}$ as well, see [17]. Thus, the number n of observed values needed quickly grows very big. Even for a modest requirement of three correct digits one needs to generate observed values in the order of $10'000'000$ for each input variable. Executing this method by hand is beyond discussion, but executing the same function n times by a computer is very attractive, since it is very simple to implement and computers are becoming ever faster. Therefore waiting is ever less an issue. Despite the simplicity of the principle of Monte Carlo simulations, it is very interesting to implement the handling of such large random variable samples efficiently. Section 5.3.3 on page 118 describes the techniques applied in some more details.

2.3 Calibration

This section was developed together with Oscar Chinellato [6]. I thank him for his time, his enthusiasm and the fun we had.

The following sections apply the principle of Section 2.2.2 thoroughly to the *calibration process* or a *general regression*. These are metrological and mathematical terms for the same thing: fitting a curve through some given points.

2.3.1 Calibration and Measurement

Most analytical methods are relative to some references and therefore need calibration. Thus calibration measurements are normally performed based on *reference material* or *calibration standards*. Usually least-squares methods ignoring the uncertainties associated with calibration standards are used. But different authors [12, 5] have shown that taking into account the uncertainties associated with calibration standards leads to better approximations of the model. In [12, 5] this is called a *Maximum Likelihood (fitting of a) Functional Relationship (MLFR)*. It is also shown in a terse way how the computation of MLFR can be executed. We present here two slightly different approaches. The first method, the so-called XIP-FIT, is very similar to the MLFR but it allows to take covariances between input data into account. The second method, the so-called P-FIT, yields results — at least in our tests — close to the *effective variance* approach as described in [39], but again it allows to take covariances among input data into account.

Since measurement instruments obey physical laws, they can be modeled by mathematical functions. Unless all free parameters $\boldsymbol{p}$ are quantified these functions describe only the qualitative behavior. Calibration is the process of quantifying these parameters. For example, consider the measurement of a weight using a spring balance.

Here we denote weights by $x \in \mathbb{R}$ and stretches of the spring by $y \in \mathbb{R}$. Let x_{cs} be a known weight of a calibration standard and y_{cs} be the stretch of the spring balance for this calibration standard. For a given calibrated balance $\boldsymbol{p}$ the following holds: $y_{\text{cs}} = f(x_{\text{cs}}, \boldsymbol{p})$. The function f is called *calibration function*. The art of calibrating consists of adjusting the parameters $\boldsymbol{p}$(e.g. the spring constant) using several calibrating pairs x_i and y_i such that $y_i \approx f(x_i, \boldsymbol{p})$ holds for all i as well as possible.

For an unknown mass $\hat{x}$, *weighing with a calibrated spring balance* requires the observation of a spring stretch $\hat{y}$ and subsequent application of the *measurement function* $g(\hat{y}, \boldsymbol{p})$. In other words, $\hat{x} \approx g(\hat{y}, \boldsymbol{p})$. Note that g is the inverse function of f.

In the case that the pairs (x_i, y_i) were exact this would be a standard problem. But in practice every measurement is subject to uncertainties. Therefore, not only the calibrating pairs (x_i, y_i) but also their respective uncertainties (α_i, β_i) must be known and taken into consideration. Throughout this section the error random variables $e_i^{(x)}$ and $e_i^{(y)}$ are treated as $e_i^{(x)} \sim \mathcal{N}(0, \alpha_i)$ and $e_i^{(y)} \sim \mathcal{N}(0, \beta_i)$ respectively. They represent the distribution of the true values. This is in full compliance with the "Guide to the Expression of Uncertainty in Measurement" (GUM) [2].

With respect to our spring balance example, this implies that the real calibration weights are not x_i but $x_i + e_i^{(x)}$. Analogously the correct values for the stretch are no longer y_i but $y_i + e_i^{(y)}$.

Note that the uncertainties of x and y are uncorrelated. That is, the error of the calibration standard is independent of the error of the reading.

2.3.2 Calibration Models

Before we start considering calibration models, it is necessary to agree on a number of assumptions.

The measurement tool is expected to work according to the function $f(\cdot, \cdot)$ in use. This means that there exists a set of parameters which describes the apparatus correctly.

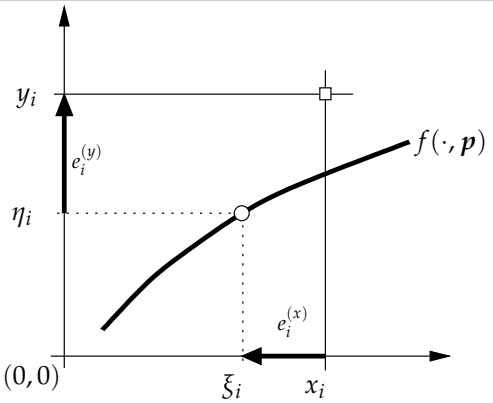
Every error *random vector* $\boldsymbol{u}$ is assumed to be normally distributed and is denoted by $\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$ where $\boldsymbol{\mu} \in \mathbb{R}^n$ is the *mean vector* and $\boldsymbol{C} \in \mathbb{R}^{n \times n}$ is the *covariance matrix* of the random variable $\boldsymbol{u} \in \mathbb{R}^n$. The multivariate *probability density function* (pdf) is

$$\mathrm{pdf}(\boldsymbol{u}) = \mathrm{err}\,(\boldsymbol{u} - \boldsymbol{\mu}, \boldsymbol{C}) = (2\pi)^{-\frac{n}{2}} \frac{1}{\sqrt{\det(\boldsymbol{C})}} \exp\left(-\frac{1}{2} (\boldsymbol{u} - \boldsymbol{\mu})^\top \boldsymbol{C}^{-1} (\boldsymbol{u} - \boldsymbol{\mu}) \right).$$

A crucial point for the further development of calibration models is the exact specification of the calibration process. A reasonable assumption for one calibration measurement is the following (the process is schematically depicted in Figure 2):

- Choose the i-th calibration standard x_i (whose real but unknown value ξ_i differs from x_i by $e_i^{(x)}$).
- Let $\boldsymbol{p}$ be the real but unknown set of parameters that describes the tool exactly. Then η_i shall denote the true but (again) unknown value $f(\xi_i, \boldsymbol{p})$.
- Finally, the reading mechanism of the tool generally imposes an additional error $e_i^{(y)}$. This leads to $y_i = \eta_i + e_i^{(y)}$.

Figure 2 Process of a Calibration Measurement



Berkson's Model The *Berkson Model* [9, 13, 22, 31] is a statistical model that handles exactly the process described above and shown in Figure 2. It behaves as follows

$$\begin{aligned} \xi_i &= x_i + e_i^{(x)} \\ y_i &= f(\xi_i, \boldsymbol{p}) + e_i^{(y)} \end{aligned} \tag{2.19}$$

where x_i is a known and constant value and the errors $e_i^{(x)}$ and $e_i^{(y)}$ are independent and normally distributed with means 0 and variances α_i^2 and β_i^2 respectively.

A *Taylor series expansion* of $f(\xi_i, \boldsymbol{p})$ around x_i yields an expression that reveals the connection between ξ_i and y_i

$$y_i \approx f(x_i, \boldsymbol{p}) + \frac{\partial}{\partial x} f(x_i, \boldsymbol{p}) \cdot e_i^{(x)} + e_i^{(y)} = f^{(i)} + f_x^{(i)} e_i^{(x)} + e_i^{(y)}, \quad (2.20)$$

where $f^{(i)} = f(x_i, \boldsymbol{p})$ and $f_x^{(i)} = \frac{\partial}{\partial x} f(x_i, \boldsymbol{p})$. Equations (2.19) and (2.20) make the following relations obvious:

$$\begin{aligned} \operatorname{Var}(\xi_i) &= \operatorname{Var}(x_i + e_i^{(x)}) &&= \alpha_i^2 \\ \operatorname{Var}(y_i) &\approx \operatorname{Var}\left(f^{(i)} + f_x^{(i)} e_i^{(x)} + e_i^{(y)}\right) &&= {f_x^{(i)}}^2 \alpha_i^2 + \beta_i^2 \\ \operatorname{Cov}(\xi_i, y_i) &\approx \operatorname{E}\left[e_i^{(x)} \left(f_x^{(i)} e_i^{(x)} + e_i^{(y)}\right)\right] &&= f_x^{(i)} \alpha_i^2. \end{aligned}$$

Fitting of ξ and p (XIP-FIT) As can be seen in Section 2.3.2, the pdf of (ξ_i, y_i) can be written as a *bivariate normal distribution*

$$\operatorname{pdf}(\xi_i, y_i) = \operatorname{err}(\boldsymbol{z}_i, \boldsymbol{C}_i) = \frac{1}{2\pi} \frac{1}{\sqrt{\det(\boldsymbol{C}_i)}} \exp\left(-\frac{1}{2} \boldsymbol{z}_i^\top \boldsymbol{C}_i^{-1} \boldsymbol{z}_i\right),$$

where $\boldsymbol{z}_i$ and $\boldsymbol{C}_i$ are defined as follows:

$$\boldsymbol{z}_i = \begin{pmatrix} \xi_i - x_i \\ y_i - f^{(i)} \end{pmatrix} \quad \text{and} \quad \boldsymbol{C}_i = \begin{pmatrix} \alpha_i^2 & f_x^{(i)} \alpha_i^2 \\ f_x^{(i)} \alpha_i^2 & {f_x^{(i)}}^2 \alpha_i^2 + \beta_i^2 \end{pmatrix}.$$

We generalize to all n measurements, and allow additional correlations among the errors $e_i^{(x)}$ and $e_i^{(y)}$ denoted by $\boldsymbol{C}_x$ and $\boldsymbol{C}_y$ respectively. This leads to the

$$\operatorname{pdf}(\boldsymbol{\xi}, \boldsymbol{y}) = \operatorname{err}(\boldsymbol{z}, \boldsymbol{C}) = (2\pi)^{-n} \frac{1}{\sqrt{\det(\boldsymbol{C})}} \exp\left(-\frac{1}{2} \boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z}\right),$$

where $\boldsymbol{z}$ and $\boldsymbol{C}$ now are defined as

$$\boldsymbol{z} = \begin{pmatrix} \boldsymbol{\xi} - \boldsymbol{x} \\ \boldsymbol{y} - \boldsymbol{f} \end{pmatrix} \quad \text{and} \quad \boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_x & \boldsymbol{C}_x \boldsymbol{F}_x \\ \boldsymbol{F}_x \boldsymbol{C}_x & \boldsymbol{F}_x \boldsymbol{C}_x \boldsymbol{F}_x + \boldsymbol{C}_y \end{pmatrix},$$

and analogously,

$$\boldsymbol{f} = \left(f^{(1)}, \ldots, f^{(n)}\right)^\top \quad \text{and} \quad \boldsymbol{F}_x = \operatorname{diag}\left(f_x^{(1)}, \ldots, f_x^{(n)}\right).$$

Thus we know the value of the pdf at $(\xi, \boldsymbol{p})$ given $\boldsymbol{x}$ and $\boldsymbol{y}$. The maximum likelihood principle aims at maximizing this value. Hence

$$\max_{\xi,p} \left((2\pi)^{-n} \frac{1}{\sqrt{\det(\boldsymbol{C})}} \exp\left(-\frac{1}{2} \boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z} \right) \right) \tag{2.21}$$

must be solved for $\boldsymbol{p}$ and ξ. See Section 2.3.3 for computational details.

Fitting of p (P-FIT) In the previous section we fitted for both parameters $\boldsymbol{p}$ and ξ. Hence we found the most probable pairing of $\boldsymbol{p}$ and ξ, i.e. an estimation for the true shape of f and simultaneously the true values of the calibration standards $\boldsymbol{x}$. However, it is often the case that one is not interested in an estimate for the calibration standard values, but rather one is interested in the most probable shape of the calibration function f regardless of any estimate for $\boldsymbol{x}$.

Consequently, we are looking for the pdf of $\boldsymbol{y}$ only. This is computed by the *law of total probability* [38]. We show the derivation of it for one calibration pair (x, y). We use $f = f(x, \boldsymbol{p})$ and $f_x = \frac{\partial}{\partial x} f(x, \boldsymbol{p})$ as abbreviations.

$$\begin{aligned}
\mathrm{pdf}(y) &= \int_{-\infty}^{\infty} \mathrm{pdf}\left(y \middle| \xi\right) \mathrm{pdf}\left(\xi\right)\, d\xi \\
&= \int_{-\infty}^{\infty} \mathrm{err}\big(y - f(\xi, \boldsymbol{p}), \beta\big)\, \mathrm{err}\left(\xi - x, \alpha\right)\, d\xi \\
&\approx \int_{-\infty}^{\infty} \mathrm{err}\big(y - f - f_x(\xi - x), \beta\big)\, \mathrm{err}\left(\xi - x, \alpha\right)\, d\xi \\
&= \frac{1}{2\pi} \frac{1}{\alpha\beta} \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2} \frac{\big(y - f - f_x(\xi - x)\big)^2}{\beta^2} \right) \exp\left(-\frac{1}{2} \frac{(\xi - x)^2}{\alpha^2} \right) d\xi \\
&= \frac{1}{\sqrt{2\pi}} \frac{1}{\sqrt{\alpha^2 f_x^2 + \beta^2}} \exp\left(-\frac{1}{2} \frac{(y - f)^2}{\alpha^2 f_x^2 + \beta^2} \right)
\end{aligned}$$

$$\implies \mathrm{pdf}(y) \approx \mathrm{err}\left(y - f, \sqrt{\alpha^2 f_x^2 + \beta^2} \right)$$

The generalization to the n-dimensional case is now straightforward:

$$\mathrm{pdf}(\boldsymbol{y}) \approx \mathrm{err}\,(\boldsymbol{z}, \boldsymbol{C}) = (2\pi)^{-\frac{n}{2}} \frac{1}{\sqrt{\det(\boldsymbol{C})}} \exp\left(-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{C}^{-1}\boldsymbol{z}\right)$$

with

$$\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{f} \qquad \text{and} \qquad \boldsymbol{C} = \boldsymbol{F}_x \boldsymbol{C}_x \boldsymbol{F}_x + \boldsymbol{C}_y.$$

Hence we know the value of the pdf at $\boldsymbol{p}$ given $\boldsymbol{x}$ and $\boldsymbol{y}$. Again using the principle of maximum likelihood, we are left with the maximization problem

$$\max_{\boldsymbol{p}} \left((2\pi)^{-\frac{n}{2}} \frac{1}{\sqrt{\det(\boldsymbol{C})}} \exp\left(-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{C}^{-1}\boldsymbol{z}\right)\right). \tag{2.22}$$

Superficially, this looks equivalent to Equation (2.21), but recall that $\boldsymbol{C}$ and $\boldsymbol{z}$ are defined differently. See Section 2.3.3 for the computation of $\boldsymbol{p}$.

Model proposed in ISO 6143 The method proposed by ISO 6143 [5] is again a maximum likelihood fit:

$$\max_{\xi,\boldsymbol{p}} \left(K \exp\left(-\frac{1}{2}(\xi - \boldsymbol{x})^\top \boldsymbol{C}_x^{-1}(\xi - \boldsymbol{x}) - \frac{1}{2}(\boldsymbol{f}(\xi, \boldsymbol{p}) - \boldsymbol{y})^\top \boldsymbol{C}_y^{-1}\big(\boldsymbol{f}(\xi, \boldsymbol{p}) - \boldsymbol{y}\big)\right)\right) \tag{2.23}$$

with $K = (2\pi)^{-n}\big(\det(\boldsymbol{C}_x)\det\big(\boldsymbol{C}_y\big)\big)^{-1/2}$ constant. Comparing this formula to Equation (2.21), one can see the that (2.21) is almost a linearized form of (2.23). But in (2.21) the covariances between calibration standards lead to additional terms that are ignored by (2.23). On the other hand, our model (2.21) uses only linear approximations for the $\boldsymbol{f}(\xi, \boldsymbol{p})$-terms. Both forms have their advantages: the linearized version (2.21) can handle covariances and can be used to compute the probability density function for the P-FIT, whereas the non linearized variant (2.23) is too complicated to proceed on that path. The disadvantage is that one is forced to linearize the calibration function around the calibration standards. The ISO 6143 model is more accurate in cases where the calibration function shows strong non linear behavior in the neighborhood of the calibration standards, this is at the prize of ignoring covariances completely.

In our opinion the benefit of considering covariances is more impor-

tant since in practical cases one strives for almost linear measurement functions. Moreover, $\boldsymbol{C}_x$ and $\boldsymbol{C}_y$ cannot be assumed to be diagonal as pretended by (2.23) (see [5, p. 22]). However, using this assumption anyhow Equation (2.23) leads to the minimization problem

$$\min_{\xi,\boldsymbol{p}} \left(\sum_{i=1}^{n} \left(\frac{(\xi_i - x_i)^2}{\alpha_i^2} + \frac{\big(f(\xi_i, \boldsymbol{p}) - y_i\big)^2}{\beta_i^2} \right) \right). \tag{2.24}$$

This is the minimization problem described in [5, p. 19]. It belongs to the class of *weighted orthogonal distance regression* (Weighted ODR) problems that are thoroughly studied in [11] and [43].

2.3.3 Computations

In this section we show how to solve the maximization problems (2.21) and (2.22). For quantifying the measurement uncertainty when using the calibrated tool we need the covariances of the parameters $\boldsymbol{p}$, i.e. the covariance matrix $\boldsymbol{C}_{\boldsymbol{p}}$. Its use will be demonstrated in Section 4.4. We explain the computation of the covariance matrix $\boldsymbol{C}_{\boldsymbol{p}}$ at the end of each subsection.

Notation We use the following symbols, notations and abbreviations for readability: $\boldsymbol{M}^{(i)}$ denotes the i-th column vector of matrix $\boldsymbol{M}$. Additionally, ∂_k abbreviates $\frac{\partial}{\partial \boldsymbol{p}_k}$, i.e. the element-wise derivative with respect to $\boldsymbol{p}_k$.

$$\begin{aligned} \boldsymbol{J}_{\boldsymbol{p}}^{\top} &= \left(\tfrac{\partial}{\partial \boldsymbol{p}} f^{(1)}, \ldots, \tfrac{\partial}{\partial \boldsymbol{p}} f^{(n)} \right) \\ \boldsymbol{J}_{x\boldsymbol{p}}^{\top} &= \left(\tfrac{\partial}{\partial \boldsymbol{p}} f_x^{(1)}, \ldots, \tfrac{\partial}{\partial \boldsymbol{p}} f_x^{(n)} \right) \\ \boldsymbol{F}_{x\boldsymbol{p}_k} &= \partial_k \boldsymbol{F}_x \end{aligned}$$

Computing ξ and $\boldsymbol{p}$ Consider the maximization problem (2.21). This is equivalent (after taking the logarithm) to the following minimization problem:

$$\min_{\xi,\boldsymbol{p}} \left(n \log(2\pi) + \frac{1}{2} \log\big(\det(\boldsymbol{C})\big) + \frac{1}{2} \boldsymbol{z}^{\top} \boldsymbol{C}^{-1} \boldsymbol{z} \right).$$

Constant terms can now be dropped . Note that $\det(\boldsymbol{C})$ is also constant. This can be seen by applying a *Cholesky factorization* to $\boldsymbol{C}$. Note: There exists a Cholesky factorization for any symmetric positive semi-definite matrix

which, by definition, is the case for all covariance matrices.

$$\boldsymbol{C} = \boldsymbol{R}^\top \boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_x^\top & \boldsymbol{0} \\ \boldsymbol{F}_x \boldsymbol{R}_x^\top & \boldsymbol{R}_y^\top \end{pmatrix} \begin{pmatrix} \boldsymbol{R}_x & \boldsymbol{R}_x \boldsymbol{F}_x \\ \boldsymbol{0} & \boldsymbol{R}_y \end{pmatrix},$$

where $\boldsymbol{C}_x = \boldsymbol{R}_x^\top \boldsymbol{R}_x$ and $\boldsymbol{C}_y = \boldsymbol{R}_y^\top \boldsymbol{R}_y$. Note the fact that $\det(\boldsymbol{C}) = \det\left(\boldsymbol{R}_x^\top\right) \det(\boldsymbol{R}_x) \det\left(\boldsymbol{R}_y^\top\right) \det(\boldsymbol{R}_y) = \det\left(\boldsymbol{R}_x^\top \boldsymbol{R}_x\right) \det\left(\boldsymbol{R}_y^\top \boldsymbol{R}_y\right) = \det(\boldsymbol{C}_x) \det\left(\boldsymbol{C}_y\right)$ and therefore $\det(\boldsymbol{C})$ is independent of ξ and $\boldsymbol{p}$and thus constant. Lucky us! We are left with the following minimization problem

$$\min_{\xi, \boldsymbol{p}} \left(\boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z} \right).$$

Solving this is equivalent to solving a *nonlinear least squares problem*:

$$\begin{aligned} \min_{\xi, \boldsymbol{p}} \left(\boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z} \right) &= \min_{\xi, \boldsymbol{p}} \left(\boldsymbol{z}^\top \boldsymbol{R}^{-1} \boldsymbol{R}^{-\top} \boldsymbol{z} \right) \\ &= \min_{\xi, \boldsymbol{p}} \left(\boldsymbol{r}^\top \boldsymbol{r} \right) \qquad \text{with } \boldsymbol{r} = \boldsymbol{R}^{-\top} \boldsymbol{z} \\ &= \min_{\xi, \boldsymbol{p}} \|\boldsymbol{r}\|_2^2 = \min_{\xi, \boldsymbol{p}} \mathcal{S}\left(\xi, \boldsymbol{p}\right). \end{aligned} \tag{2.25}$$

There are different standard solvers for such problems, e.g. the *Gauss-Newton* or the further developed *Levenberg-Marquardt* method, other trust region methods, the *Newton* method, the *spectral decomposition* method, etc. [10, 24, 25, 36, 42]. Section 2.3.5 describes the development of the algorithm used in the MUSAC-system to some detail. The following condition holds for the computed minimum

$$\boldsymbol{J}^\top \boldsymbol{r} = \boldsymbol{0}, \tag{2.26}$$

where $\boldsymbol{J}$ denotes the *Jacobian* of $\boldsymbol{r}$ with respect to ξ and $\boldsymbol{p}$. Its structure can be seen in Equation (2.28). The statement above can be verified as follows: we define $\boldsymbol{w} = (\xi, \boldsymbol{p})^\top$. Linearizing the function $\boldsymbol{r}(\boldsymbol{w} + \Delta\boldsymbol{w}) \approx \boldsymbol{r}(\boldsymbol{w}) + \boldsymbol{J}\Delta\boldsymbol{w}$ we get $\mathcal{S}\left(\boldsymbol{w} + \Delta\boldsymbol{w}\right) \approx \mathcal{S}\left(\boldsymbol{w}\right) + 2\Delta\boldsymbol{w}^\top \boldsymbol{J}^\top \boldsymbol{r} + \mathcal{O}(\|\Delta\boldsymbol{w}\|_2^2)$. Thus Equation (2.26) obviously holds at a minimum.

The Computation of $\boldsymbol{C}_p$ For a good estimation of the measurement error we need the covariance matrix $\boldsymbol{C}_p$ of $\boldsymbol{p}$. We therefore rewrite Equation (2.26) as

$$\boldsymbol{h}(\boldsymbol{w}, \boldsymbol{y}) = \boldsymbol{J}^\top \boldsymbol{r} = \boldsymbol{0}. \tag{2.27}$$

The function $\boldsymbol{h}$ is an implicit description of the maximum likelihood criterion. Thus, $\boldsymbol{w}$ is determined by $\boldsymbol{y}$ only (and of course $\boldsymbol{x}$ which is constant for a given set of calibration standards). Since $\boldsymbol{y}$ is a random variable with a known covariance matrix, we are now able to compute the covariance matrix of $\boldsymbol{w}$ and thus of $\boldsymbol{\xi}$ and $\boldsymbol{p}$.

We define $\bar{\boldsymbol{w}} = \left(\bar{\boldsymbol{\xi}}, \bar{\boldsymbol{p}}_{\text{est}}\right)^\top = \mathrm{E}[\boldsymbol{w}]$ and $\bar{\boldsymbol{y}} = \mathrm{E}[\boldsymbol{y}]$ to be the means of the respective random variables. Note that the mean of several estimated parameter vectors $\bar{\boldsymbol{p}}_{\text{est}}$ is not necessarily the true and unknown parameter vector. Then

$$\boldsymbol{w} = \bar{\boldsymbol{w}} + \Delta\boldsymbol{w} \quad \text{and} \quad \boldsymbol{y} = \bar{\boldsymbol{y}} + \Delta\boldsymbol{y}.$$

By definition, we may write

$$\boldsymbol{h}(\bar{\boldsymbol{w}}, \bar{\boldsymbol{y}}) = \boldsymbol{0}$$

and after linearizing $\boldsymbol{h}$ and inserting the equations above we get

$$\begin{aligned} \boldsymbol{0} &\approx \frac{\partial}{\partial \boldsymbol{w}} \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{y}) \Delta\boldsymbol{w} + \frac{\partial}{\partial \boldsymbol{y}} \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{y}) \Delta\boldsymbol{y} \\ &= \boldsymbol{H}_w \Delta\boldsymbol{w} + \boldsymbol{H}_y \Delta\boldsymbol{y} \end{aligned}$$

Solving for $\Delta\boldsymbol{w}$ and rearranging the expressions we get

$$\begin{aligned} \Delta\boldsymbol{w} &\approx -\boldsymbol{H}_w^{-1} \boldsymbol{H}_y \Delta\boldsymbol{y} \\ \implies \quad \mathrm{Cov}(\boldsymbol{w}, \boldsymbol{w}) &= \mathrm{E}\left[\Delta\boldsymbol{w} \Delta\boldsymbol{w}^\top\right] \\ &\approx \mathrm{E}\left[\boldsymbol{H}_w^{-1} \boldsymbol{H}_y \Delta\boldsymbol{y} \Delta\boldsymbol{y}^\top \boldsymbol{H}_y^\top \boldsymbol{H}_w^{-\top}\right] \\ &= \boldsymbol{H}_w^{-1} \boldsymbol{H}_y \left(\boldsymbol{F}_x \boldsymbol{C}_x \boldsymbol{F}_x + \boldsymbol{C}_y\right) \boldsymbol{H}_y^\top \boldsymbol{H}_w^{-\top}. \end{aligned}$$

Note that $\boldsymbol{H}_w$ is the Hessian of $\mathcal{S}$ and that its inverse exists if $\boldsymbol{H}_w$ is positive definite which implies the existence of a unique minimum. The covariance matrix $\boldsymbol{C}_p$ corresponds to the lower right $m \times m$ block of $\mathrm{Cov}(\boldsymbol{w}, \boldsymbol{w})$.

Finally we must show how to compute the matrices $\boldsymbol{H}_w$ and $\boldsymbol{H}_y$. We start by expanding the expression (2.27) using the variables introduced throughout Section 2.3.2 and 2.3.3. On the one hand, we have

$$\begin{aligned} \boldsymbol{H}_y &= \frac{\partial}{\partial \boldsymbol{y}} \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{y}) \\ &= \left(\frac{\partial}{\partial \boldsymbol{y}} \boldsymbol{J}^\top\right) \boldsymbol{r} + \boldsymbol{J}^\top \left(\frac{\partial}{\partial \boldsymbol{y}} \boldsymbol{r}\right) \end{aligned}$$

$$
\begin{aligned}
\boldsymbol{H}_{\boldsymbol{y}} &= \mathbf{0} \cdot \boldsymbol{r} + \boldsymbol{J}^{\top} \boldsymbol{R}^{-\top} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{I} \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{R}_x^{-\top} & \mathbf{0} \\ -\boldsymbol{R}_y^{-\top} \boldsymbol{F}_x & -\boldsymbol{R}_y^{-\top} \left(\operatorname{diag}(\xi - \boldsymbol{x})\, \boldsymbol{J}_{xp} - \boldsymbol{J}_p \right) \end{pmatrix}^{\top} \boldsymbol{R}^{-\top} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{I} \end{pmatrix} \qquad (2.28) \\
&= \begin{pmatrix} -\boldsymbol{F}_x \boldsymbol{C}_y^{-1} \\ -\left(\boldsymbol{J}_{xp}^{\top} \operatorname{diag}(\xi - \boldsymbol{x}) - \boldsymbol{J}_p^{\top} \right) \boldsymbol{C}_y^{-1} \end{pmatrix}.
\end{aligned}
$$

On the other hand, for the i-th column $\boldsymbol{H}_{\boldsymbol{w}}^{(i)}$, for $1 \le i \le n$, i.e. the derivatives with respect to ξ_i, we have

$$
\begin{aligned}
\boldsymbol{H}_{\boldsymbol{w}}^{(i)} &= \left(\frac{\partial}{\partial \xi_i} \boldsymbol{J}^{\top} \right) \boldsymbol{r} + \boldsymbol{J}^{\top} \left(\frac{\partial}{\partial \xi_i} \boldsymbol{r} \right) \\
&= \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\boldsymbol{J}_{xp}^{\top} \operatorname{diag}(\boldsymbol{e}_i)\, \boldsymbol{R}_y^{-1} \end{pmatrix} \boldsymbol{r} + \boldsymbol{J}^{\top} \boldsymbol{R}^{-\top} \begin{pmatrix} \boldsymbol{e}_i \\ \mathbf{0} \end{pmatrix},
\end{aligned}
$$

and for $1 \le k \le m$, i.e. the derivatives with respect to $\boldsymbol{p}_k$, we have

$$
\begin{aligned}
\boldsymbol{H}_{\boldsymbol{w}}^{(n+k)} &= \left(\partial_k \boldsymbol{J}^{\top} \right) \boldsymbol{r} + \boldsymbol{J}^{\top} \left(\partial_k \boldsymbol{r} \right) \\
&= \begin{pmatrix} \mathbf{0} & -\boldsymbol{F}_{x\boldsymbol{p}_i} \boldsymbol{R}_y^{-1} \\ \mathbf{0} & -\left(\boldsymbol{J}_{x\boldsymbol{p}\boldsymbol{p}_i}^{\top} \operatorname{diag}(\xi - \boldsymbol{x}) - \boldsymbol{J}_{\boldsymbol{p}\boldsymbol{p}_i}^{\top} \right) \boldsymbol{R}_y^{-1} \end{pmatrix} \boldsymbol{r} \\
&\quad + \boldsymbol{J}^{\top} \left(\begin{pmatrix} \mathbf{0} & \mathbf{0} \\ -\boldsymbol{R}_y^{-\top} \boldsymbol{F}_{x\boldsymbol{p}_i} & \mathbf{0} \end{pmatrix} \boldsymbol{z} + \boldsymbol{R}^{-\top} \begin{pmatrix} \mathbf{0} \\ -\boldsymbol{J}_{\boldsymbol{p}}^{(i)} \end{pmatrix} \right),
\end{aligned}
$$

where $\boldsymbol{e}_i$ is the i-th unit vector.

Computing p only Consider the maximization problem (2.22). This is equivalent to the following minimization problem (after taking logarithms again):

$$
\min_{\boldsymbol{p}} \left(\frac{1}{2} n \log(2\pi) + \frac{1}{2} \log\big(\det(\boldsymbol{C}) \big) + \frac{1}{2} \boldsymbol{z}^{\top} \boldsymbol{C}^{-1} \boldsymbol{z} \right).
$$

Note that the matrix $\boldsymbol{C}$ used here is different from the one in Section 2.3.3. $\boldsymbol{R}$ is again the upper triangular matrix resulting from the Cholesky factorization of the new $\boldsymbol{C}$. It is easy to see that $\det(\boldsymbol{C})$ is no longer constant in $\boldsymbol{p}$ and cannot be neglected. The only constant term negligible is $\frac{1}{2}n\log(2\pi)$. Thus, we are not so lucky anymore! We are now left with the following general minimization problem

$$\min_{\boldsymbol{p}} \left(\log\big(\det(\boldsymbol{C})\big) + \boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z} \right).$$

Solving (2.29) is no longer equivalent to solving a nonlinear least squares problem, as can be seen by the following transformation

$$\begin{aligned} &\min_{\boldsymbol{p}} \left(\log\big(\det(\boldsymbol{C})\big) + \boldsymbol{z}^\top \boldsymbol{C}^{-1} \boldsymbol{z} \right) \\ = &\min_{\boldsymbol{p}} \left(\log\left(\det\big(\boldsymbol{R}^\top \boldsymbol{R}\big) \right) + \boldsymbol{z}^\top \boldsymbol{R}^{-1} \boldsymbol{R}^{-\top} \boldsymbol{z} \right) \\ = &\min_{\boldsymbol{p}} \left(2\log\big(\det(\boldsymbol{R})\big) + \boldsymbol{r}^\top \boldsymbol{r} \right) = \min_{\boldsymbol{p}} \mathcal{S}\left(\boldsymbol{p}\right) \end{aligned} \tag{2.29}$$

Nonetheless the standard techniques mentioned in Section 2.3.3 can be applied to this problem. However, solving Problem (2.29) requires more effort, as the construction of the Jacobian is more difficult (see Section 2.3.3). At the minimum the following condition holds:

$$\boldsymbol{q} + \boldsymbol{J}^\top \boldsymbol{r} = \boldsymbol{0}$$

Here $\boldsymbol{q} = \frac{\partial}{\partial \boldsymbol{p}} \log\big(\det(\boldsymbol{R})\big)$ and $\boldsymbol{J}$ denotes the Jacobian of $\boldsymbol{r}$ with respect to $\boldsymbol{p}$. Again, as in Equation (2.26), this means that the first derivative with respect to $\boldsymbol{p}$ is zero. We show in the following how to compute these entities.

Computing of $\boldsymbol{C}_p$ In the first step we derive an expression for

$$\begin{aligned} \boldsymbol{q} &= \frac{\partial}{\partial \boldsymbol{p}} \log\big(\det(\boldsymbol{R})\big) \\ &= \frac{\partial}{\partial \boldsymbol{p}} \log\left(\prod_{i=1}^{n} \boldsymbol{R}_{i,i} \right) \\ &= \frac{\partial}{\partial \boldsymbol{p}} \sum_{i=1}^{n} \log\left(\boldsymbol{R}_{i,i}\right). \end{aligned}$$

For the k-th entry of $\boldsymbol{q}$, we get

$$\begin{aligned}\boldsymbol{q}_k &= \partial_k \sum_{i=1}^{n} \log\left(\boldsymbol{R}_{i,i}\right) \\ &= \sum_{i=1}^{n} \frac{\partial_k \boldsymbol{R}_{i,i}}{\boldsymbol{R}_{i,i}}.\end{aligned}$$

What remains is to show how $\partial_k \boldsymbol{R}_{i,i}$ can be computed. We begin by observing that

$$\begin{aligned}\partial_k \boldsymbol{C} &= \left(\partial_k \boldsymbol{F}_x\right) \boldsymbol{C}_x \boldsymbol{F}_x + \boldsymbol{F}_x \boldsymbol{C}_x \left(\partial_k \boldsymbol{F}_x\right) \\ &= \boldsymbol{F}_{x\boldsymbol{p}_k} \boldsymbol{C}_x \boldsymbol{F}_x + \boldsymbol{F}_x \boldsymbol{C}_x \boldsymbol{F}_{x\boldsymbol{p}_k}.\end{aligned}$$

With $\partial_k \boldsymbol{C}$ and with the derivation rules applied to $\boldsymbol{C} = \boldsymbol{R}^\top \boldsymbol{R}$ we get

$$\partial_k \boldsymbol{C} = \left(\partial_k \boldsymbol{R}^\top\right) \boldsymbol{R} + \boldsymbol{R}^\top \left(\partial_k \boldsymbol{R}\right).$$

This can easily be solved for $\partial_k \boldsymbol{R}$ and its diagonal elements are the wanted $\partial_k \boldsymbol{R}_{i,i}$. As in Section 2.3.3 we introduce again an implicit function

$$\boldsymbol{h}(\boldsymbol{p}, \boldsymbol{y}) = \boldsymbol{q} + \boldsymbol{J}^\top \boldsymbol{r} = \boldsymbol{0}.$$

Due to the complicated structure of $\boldsymbol{C}$ we abstain from giving an explicit expression for $\boldsymbol{J}$, rather we show its computation in a column-wise manner. We write the k-th column of $\boldsymbol{J}$ by $\boldsymbol{J}^{(k)}$, for $1 \leq k \leq m$. Then we have

$$\begin{aligned}\boldsymbol{J}^{(k)} &= \partial_k \boldsymbol{r} \\ &= \left(\partial_k \boldsymbol{R}^{-\top}\right) \boldsymbol{r} + \boldsymbol{R}^{-\top} \left(\partial_k \boldsymbol{r}\right) \\ &= \left(\partial_k \boldsymbol{R}^{-\top}\right) \boldsymbol{r} - \boldsymbol{R}^{-\top} \boldsymbol{J}_{\boldsymbol{p}}^{(k)}.\end{aligned}$$

This leaves us with the task of computing $\partial_k \boldsymbol{R}^{-\top}$. We have $\boldsymbol{I} = \boldsymbol{R}^{-1}\boldsymbol{R}$. Thus, after taking derivatives on both sides:

$$\begin{aligned}\boldsymbol{0} &= \left(\partial_k \boldsymbol{R}^{-1}\right)\boldsymbol{R} + \boldsymbol{R}^{-1} \left(\partial_k \boldsymbol{R}\right) \\ \Longrightarrow \quad \partial_k \boldsymbol{R}^{-1} &= -\boldsymbol{R}^{-1} \left(\partial_k \boldsymbol{R}\right) \boldsymbol{R}^{-1}\end{aligned}$$

Again, as in Section 2.3.3, we need to compute $\boldsymbol{H}_{\boldsymbol{p}}$ and $\boldsymbol{H}_{\boldsymbol{y}}$. Let $\boldsymbol{H}_{\boldsymbol{y}}^{(i)}$ denote the i-th column of $\boldsymbol{H}_{\boldsymbol{y}}$. For $1 \leq i \leq n$, we get

$$\begin{aligned}
\boldsymbol{H}_{\boldsymbol{y}}^{(i)} &= \frac{\partial}{\partial y_i}\left(\boldsymbol{q} + \boldsymbol{J}^\top \boldsymbol{r}\right) \\
&= \mathbf{0} + \left(\frac{\partial}{\partial y_i}\boldsymbol{J}^\top\right)\boldsymbol{r} + \boldsymbol{J}^\top\left(\frac{\partial}{\partial y_i}\boldsymbol{r}\right) \\
&= \left(\frac{\partial}{\partial y_i}\left(\boldsymbol{J}^{(1)}, \ldots, \boldsymbol{J}^{(m)}\right)^\top\right)\boldsymbol{r} + \boldsymbol{J}^\top \boldsymbol{R}^{-\top}\boldsymbol{e}_i \\
&= \left(\left(\partial_1 \boldsymbol{R}^{-\top}\right)\boldsymbol{R}^{-\top}\boldsymbol{e}_i, \ldots, \left(\partial_m \boldsymbol{R}^{-\top}\right)\boldsymbol{R}^{-\top}\boldsymbol{e}_i\right)^\top \boldsymbol{r} + \boldsymbol{J}^\top \boldsymbol{R}^{-\top}\boldsymbol{e}_i.
\end{aligned}$$

The derivation for $\boldsymbol{H}_{\boldsymbol{p}}$ is exceedingly odd. Nevertheless here it is: First we have

$$\partial_l \boldsymbol{q}_k = \sum_{i=1}^{n} \frac{\left(\partial_l \partial_k \boldsymbol{R}_{i,i}\right)\boldsymbol{R}_{i,i} - \left(\partial_k \boldsymbol{R}_{i,i}\right)\left(\partial_l \boldsymbol{R}_{i,i}\right)}{\boldsymbol{R}_{i,i}^2},$$

$\left(\partial_l \partial_k \boldsymbol{R}_{i,i}\right)$ can be found by solving

$$\partial_l \partial_k \boldsymbol{C} = \left(\partial_l \partial_k \boldsymbol{R}^\top\right)\boldsymbol{R} + \left(\partial_l \boldsymbol{R}^\top\right)\left(\partial_k \boldsymbol{R}\right) + \left(\partial_k \boldsymbol{R}^\top\right)\left(\partial_l \boldsymbol{R}\right) + \boldsymbol{R}^\top\left(\partial_l \partial_k \boldsymbol{R}\right).$$

Further we need

$$\begin{aligned}
\partial_l \boldsymbol{J}^\top \boldsymbol{r} &= \left(\partial_l \boldsymbol{J}\right)^\top \boldsymbol{r} + \boldsymbol{J}^\top\left(\partial_l \boldsymbol{r}\right) \\
&= \left(\partial_l \boldsymbol{J}^{(1)}, \ldots, \partial_l \boldsymbol{J}^{(m)}\right)^\top \boldsymbol{r} + \boldsymbol{J}^\top\left(\partial_l \boldsymbol{r}\right) \\
\partial_l \boldsymbol{r} &= \left(\partial_l \boldsymbol{R}^{-\top}\right)\boldsymbol{z} - \boldsymbol{R}^{-\top}\boldsymbol{J}_{\boldsymbol{p}}^{(l)} \\
\partial_l \boldsymbol{J}^{(k)} &= \left(\partial_l \partial_k \boldsymbol{R}^{-\top}\right)\boldsymbol{r} + \left(\partial_k \boldsymbol{R}^{-\top}\right)\left(\partial_l \boldsymbol{r}\right) - \left(\partial_l \boldsymbol{R}^{-\top}\right)\boldsymbol{J}_{\boldsymbol{p}}^{(k)} - \boldsymbol{R}^{-\top}\left(\partial_l \boldsymbol{J}_{\boldsymbol{p}}^{(k)}\right).
\end{aligned}$$

Putting things together yields

$$\boldsymbol{H}_{\boldsymbol{p}}^{(l)} = \partial_l \boldsymbol{q} + \partial_l \boldsymbol{J}^\top \boldsymbol{r},$$

where $\partial_l \boldsymbol{q}$ is of course the vector with elements $\partial_l \boldsymbol{q}_k$, for $k = 1, \ldots, m$.

2.3.4 Measurement

One goal of calibration is to compute a good set of parameters to quantify the measurement tool's behavior. An equally important goal is to estimate the measurement uncertainty when using the calibrated device.

Assume that the measuring tool is calibrated. Thus we have a set of estimated parameters $\hat{\boldsymbol{p}}$ together with an estimated covariance matrix $\hat{\boldsymbol{C}}_{\boldsymbol{p}} = \mathrm{Cov}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}})$.

Let $\boldsymbol{y} \in \mathbb{R}^s$ be a measurement vector whose elements are readings taken from the measurement tool. The readings are subject to error of the form $\boldsymbol{y} - \bar{\boldsymbol{y}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{E})$, where $\bar{\boldsymbol{y}}$ is the correct but unknown reading vector and $\boldsymbol{E} = \mathrm{Cov}(\boldsymbol{y}, \boldsymbol{y})$.

For $1 \leq i \leq s$ compute $\hat{x}_i$ by applying $\hat{x}_i = g(y_i, \hat{\boldsymbol{p}})$ or by solving $y_i = f(\hat{x}_i, \hat{\boldsymbol{p}})$. For arbitrary functions $f(x, \hat{\boldsymbol{p}})$, differentiable at least once, we propose the use of a standard solver such as *bisection*, *regula falsi*, *Newton's method*, whichever you prefer.

The computation of the measurement uncertainty is performed as follows. We represent the physical law by $\bar{y}_i = f(\bar{x}_i, \boldsymbol{p}_{\mathrm{ex}})$, where $\bar{x}_i$, $\boldsymbol{p}_{\mathrm{ex}}$ and $\bar{y}_i$ are the exact but unknown "true values". The given values are $\bar{x}_i + \Delta x_i = \hat{x}_i$, $\bar{y}_i + \Delta y_i = y_i$ and $\boldsymbol{p}_{\mathrm{ex}} + \Delta \boldsymbol{p} = \hat{\boldsymbol{p}}$. Substituting and linearizing these equations leads to

$$\Delta y_i \approx f_x^{(i)} \Delta x_i + \nabla_{\boldsymbol{p}} f^{(i)^\top} \Delta \boldsymbol{p},$$

where $f^{(i)} = f(\hat{x}_i, \hat{\boldsymbol{p}})$ and $f_x^{(i)} = f_x(\hat{x}_i, \hat{\boldsymbol{p}})$. Extending this approximation to s readings $\boldsymbol{y}$ and solving for $\Delta \boldsymbol{x}$, we get

$$\Delta \boldsymbol{x} \approx \boldsymbol{F}_x^{-1} \Delta \boldsymbol{y} - \boldsymbol{F}_x^{-1} \boldsymbol{J}_{\boldsymbol{p}} \Delta \boldsymbol{p}$$

and furthermore

$$\begin{aligned} \boldsymbol{C}_x \;=\; & \mathrm{E}\left[\Delta \boldsymbol{x} \Delta \boldsymbol{x}^\top\right] \\ \approx \quad & \boldsymbol{F}_x^{-1} \mathrm{Cov}(\boldsymbol{y}, \boldsymbol{y})\, \boldsymbol{F}_x^{-\top} \\ & + \boldsymbol{F}_x^{-1} \boldsymbol{J}_{\boldsymbol{p}} \mathrm{Cov}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}})\, \boldsymbol{J}_{\boldsymbol{p}}^\top \boldsymbol{F}_x^{-\top} \\ & - \boldsymbol{F}_x^{-1} \left(\mathrm{Cov}(\boldsymbol{y}, \hat{\boldsymbol{p}})\, \boldsymbol{J}_{\boldsymbol{p}}^\top + \boldsymbol{J}_{\boldsymbol{p}} \mathrm{Cov}(\hat{\boldsymbol{p}}, \boldsymbol{y}) \right) \boldsymbol{F}_x^{-\top}. \end{aligned}$$

There are two things to note. First, $\mathrm{Cov}(\boldsymbol{y}, \hat{\boldsymbol{p}})$ can safely be assumed to be $\boldsymbol{0}$, since the uncertainties of the reading for the s measurements are indepen-

dent of $\hat{\boldsymbol{p}}$ found during calibration. Second, it is obvious that not a single $f_x^{(i)}$ is allowed to be 0. In practice this does not represent a problem, since no-one would use a measurement device within a range where the slope of the calibration function is even close to 0. So we end up with the calculation for

$$\boldsymbol{C}_x \approx \boldsymbol{F}_x^{-1} \left(\mathrm{Cov}(\boldsymbol{y}, \boldsymbol{y}) + \boldsymbol{J}_p \, \mathrm{Cov}(\hat{\boldsymbol{p}}, \hat{\boldsymbol{p}}) \, \boldsymbol{J}_p^\top \right) \boldsymbol{F}_x^{-1}. \tag{2.30}$$

2.3.5 Truncated SVD Solver

In a first approach we applied an ordinary *Gauss-Newton* method to solve the Equations (2.24), (2.25) or (2.29). But we soon realized that the non-linearities of the problems were strong enough to render the reliability of the plain Gauss-Newton method insufficient. We then applied a *Levenberg-Marquardt* method (see Algorithm 5) which stabilizes the Gauss-Newton method in the sense that it converges more reliably at the price of more iterations [42]. The effect of the λ introduced is to reduce the norm of the correction vector $\boldsymbol{d}$, such that the norm of the residual $\boldsymbol{r}$ decreased in every step. With increasing λ the norm of $\boldsymbol{d}$ diminishes. If $\lambda = 0$ then Algorithm 5 corresponds to exactly the Gauss-Newton method. There are uncountable proposals for a strategy to adapt λ. None of them (at least the ones we experimented with) was satisfactory in the range of problems we were facing in the MUSAC-project. It turned out that problems stemming from calibration have surprisingly often a badly conditioned Jacobian $\boldsymbol{J}$. This can lead to bad numerical results for $\boldsymbol{d}$, and therefore the whole algorithm is disappointingly unstable, numerically. The Levenberg-Marquardt step solves $\boldsymbol{J}\boldsymbol{d} \approx -\boldsymbol{r}$ with some constraints on $\|\boldsymbol{d}\|_2$.

Alternatively we can compute an unconstrained step $\boldsymbol{d}$ using the *Singular Value Decomposition* (SVD). Then the following steps are executed (see e.g. [27]):

$$\begin{aligned}
\boldsymbol{J} &= \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top && \boldsymbol{U} \text{ and } \boldsymbol{V} \text{ orthogonal, } \boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_n) \\
\tilde{\boldsymbol{r}} &= \boldsymbol{U}^\top \boldsymbol{r} && \text{apply left singular vectors} \\
\forall i : \sigma_i \neq 0 : \tilde{\boldsymbol{d}}_i &= \tilde{r}_i / \sigma_i && \text{compute the solution} \\
\boldsymbol{d} &= \boldsymbol{V}\tilde{\boldsymbol{d}} && \text{apply right singular vectors}
\end{aligned}$$

This procedure uses the computationally expensive SVD, but it discloses useful structural information about the system of equations. The procedure

Algorithm 5 Gauss-Newton Loop with Levenberg-Marquardt Method for Solving Equation (2.25)

function $\langle \xi, p \rangle = \mathsf{LM\_solver}\big(x, y, R_y, R_x\big)$
— Find ξ and p minimizing (2.25) or similar —
$\big\langle \xi, p, \lambda \big\rangle \leftarrow \mathsf{Initialize}(x, y)$
do — *Gauss-Newton-Loop* —
$z \leftarrow [\xi - x\,;\, y - f(\xi, p)]$ — *set up* z, r *and* J —
$r \leftarrow R^{-\top} z$
$J \leftarrow \begin{pmatrix} R_x^{-\top} & 0 \\ -R_y^{-\top} F_x & -R_y^{-\top}\left(\mathrm{diag}(\xi - x)\, J_{xp} - J_p\right) \end{pmatrix}$
$d \leftarrow -(J^\top J + \lambda I)^{-1}(J^\top r)$ — *Levenberg-Marquardt step and adaption of* λ *according to some heuristics, to be replaced by Algorithm 6* —
$\lambda \leftarrow \mathsf{Adapt}(\lambda)$
$[\xi\,;\, p] \leftarrow [\xi\,;\, p] + d$ — *update* ξ *and* p —
until $\|Jr\|_2$ has converged

does not impose any constraints on d. But note that the division by the singular values σ_i can lead to ridiculously large entries in $\tilde{d}$ depending on the respective entry in r and the magnitude of σ_i. Even worse, if any σ_i happens to be 0 this procedure would lead to a crash. There is, again, a whole bunch of proposals on how to *truncate* the SVD in this context to ensure a good convergence for Algorithm 5. A good survey on trust region methods is [35]. We experimented with some of them. In the end, we came up with a truncation that we haven't seen anywhere else before. It seems astonishingly well suited for the class of problems stemming from calibration. It is shown in Algorithm 6 on the following page. The key idea of Algorithm 6 is that the norm of the correction vector d is bounded by a fraction λ of the current solution's norm $\|z\|_2$. Note that $\|\tilde{d}\|_2 = \|d\|_2$ due to the orthogonality of V. Therefore after the loop over all i, the number n is equal to $\|d\|_2^2$. Within that loop, we observe whether $\|d\|_2 > \lambda\|z\|_2$. If this is the case then the boolean vector b stores the entries of $\tilde{d}$ responsible for the norm exceeding the trust region. After the loop, if any of b's elements were marked, the factor α is computed to be the factor by which to reduce the exceeding entries in $\tilde{d}$, such that $\|\tilde{d}\|_2 \leq \lambda\|z\|_2$ holds.

Algorithm 6 *Truncated SVD Method*

function $\langle \boldsymbol{d}, \lambda \rangle = \mathsf{TSVD}\left(\boldsymbol{J}, \boldsymbol{r}, \boldsymbol{z}, \lambda\right)$
— Find correction $\boldsymbol{d}$ for solution so far $\boldsymbol{z}$ and adapt λ —
$\langle \boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V} \rangle \leftarrow \mathsf{svd}(\boldsymbol{J})$ — *apply SVD on* $\boldsymbol{J}$ —
$\tilde{\boldsymbol{r}} \leftarrow \boldsymbol{U}^\top \boldsymbol{r}$ — *apply left singular vectors* —
$n \leftarrow 0, \;\; \boldsymbol{b} \leftarrow \boldsymbol{0}$
for all i **do**
  if $\sigma_i \neq 0$ **then**
    $\tilde{\boldsymbol{d}}_i \leftarrow \tilde{\boldsymbol{r}}_i / \sigma_i$
    $n \leftarrow n + \tilde{\boldsymbol{d}}_i^2$ — *update the norm of* $\boldsymbol{d}$ —
    if $n > \lambda^2 \|\boldsymbol{z}\|_2^2$ **then**
      $\boldsymbol{b}_i \leftarrow 1$
    end if
  else
    $\tilde{\boldsymbol{d}}_i \leftarrow 0$
  end if
end do
if $\mathsf{any}(\boldsymbol{b} = \boldsymbol{1})$ **then**
  $\alpha \leftarrow \dfrac{\lambda \|\boldsymbol{z}\|_2 - \|(1 - \boldsymbol{b}_i)\tilde{\boldsymbol{d}}_i\|_2}{\|\boldsymbol{b}_i \tilde{\boldsymbol{d}}_i\|_2}$ — *elementwise multiplications* —
  $\tilde{\boldsymbol{d}}_i \leftarrow (1 - (1 - \alpha)\boldsymbol{b}_i)\tilde{\boldsymbol{d}}_i$ — *scale down the entries of* $\tilde{\boldsymbol{d}}$ *that exceed the allowed norm of* $\boldsymbol{d}$ —
end if
$\boldsymbol{d} \leftarrow \boldsymbol{V}\tilde{\boldsymbol{d}}$ — *apply right singular vectors* —
$\lambda \leftarrow \beta\lambda$ — *our heuristic to adapt* λ*:* $\beta < 1$ —

Many variants of TSVD propose to ignore singular values below some heuristically adapted threshold, few propose to consider the quotient r_i/σ_i and ignore it if some heuristic criteria are met. We do not ignore any of the information gained from the SVD, but we scale down the seemingly over exaggerated contributions of either $\boldsymbol{r}_i$ and/or σ_i.

Often people propose to reduce the length of the correction so that the norm really diminishes in every iteration of the Gauss-Newton-Loop. We do not require this. Since $\boldsymbol{d}$ computed by Algorithm 6 is not even guaranteed to be a descending direction, despite the fact that it is certainly an approximation of an descending direction. By reducing λ monotonically, we ensure that the correction eventually gets so small, that the entire $\tilde{\boldsymbol{d}}$ gets scaled, instead of only parts of it. Now $\alpha \boldsymbol{V}\tilde{\boldsymbol{d}}$ is indeed a true descending direction, and therefore (again when λ and thus α is small enough) the minimizing Gauss-Newton-Loop is really progressing towards the next local minimum (at least).

In the MUSAC-system we initialize λ to 0.75 and the reduction factor β to 0.991. This yields 4044 iterations until $\lambda < 10^{-16}$. If this many iterations are used without $\|\boldsymbol{J}\boldsymbol{r}\|_2$ ever getting sufficiently close to 0 then a warning about non-convergence is issued and the Gauss-Newton-Loop is aborted. It is difficult to compare such heuristics thoroughly. Because any comparison depends strongly on the input given to the solver. Therefore the following comparison of the TSVD-solver with the Levenberg-Marquardt solver (with a fixed λ) is far from complete. It is a data set of an arbitrary calibration, as it may appear in the HPLC:

$$\begin{aligned}
\boldsymbol{x} &= (100, 10, 1, 0.1)^\top \\
\boldsymbol{y} &= (10000, 1000, 100, 10)^\top \\
\boldsymbol{C}_x &= \begin{pmatrix} 133.333 & 13.3333 & 1.33333 & 0.133333 \\ 13.3333 & 1.33348 & 0.133348 & 0.0133348 \\ 1.33333 & 0.133348 & 0.0133362 & 0.00133362 \\ 0.133333 & 0.0133348 & 0.00133362 & 0.000133376 \end{pmatrix} \\
\boldsymbol{C}_y &= \operatorname{diag}(10000, 100, 1, 0.0001)
\end{aligned} \tag{2.31}$$

This calibration input data was used to fit a line $f(x) = p_0 + p_1 x$ with the *Classical Model* which is used in the Monte Carlo Simulation, and to *Berkson's Model*, the XIP-FIT employed in the Iso-Evaluation. Table 2 shows the average of iterations used by each of the solvers to reach a solution within the tolerance of 10^{-8}. The starting parameter vector $\boldsymbol{p}_0$ was varied from

$\boldsymbol{p} = (p_0, p_1) \in \{-100, -10, 0, 1, 10, 100\}^2$. The correct $\boldsymbol{p}$ depends on the model, but in both cases it is very close to $\boldsymbol{p} = (0.0, 100.0)^\top$. Table 2 shows that in terms of iterations until convergence the two solver are equally fast. But for Berkson's Model which is intrinsically worse conditioned, the TSVD solver is faster, tremendously so. Further experiments show also that the TSVD solver is much more robust against bad starting points. These are two advantages of the proposed TSVD solver that outweigh the computationally more expensive singular value decomposition by far.

Table 2 Average Iterations for TSVD-Solver and Levenberg-Marquardt Solver on the Input Data (2.31) at a Tolerance of 10^{-8}

	TSVD	Levenberg-Marquardt
Classical Model	$\mu = 7.0, \sigma = 1.4$	$\mu = 7.0, \sigma = 1.4$
Berkson's Model	$\mu = 4.9, \sigma = 1.6$	$\mu = 3678, \sigma = 1074$

2.3.6 Numerical Comparisons and Simulations

In the following section we compare four different calibration functions. We fit a linear, an exponential, a power and a sigmoidal function. In these experiments we compare the data to a "God given", known and exact function. The calibration data was generated randomly according to Figure 2 on page 32 using the true parameters $\boldsymbol{p}_{\text{ex}}$. The covariances in x and y are assumed known as well. The variance in y at measurement time was chosen quite arbitrarily.

Simply put, we are trying to recover the true function $f(x, \boldsymbol{p}_{\text{ex}})$. One would expect that the chances for recovering the true function are bad using only a handful of calibration points. But the parameters recovered or fitted by the different methods are so close that, in order to make the differences visible at all, we had to apply big variances for the random errors and we selected the calibration points at unfavorable x-positions for the entire calibration process.

In each case the figure on the left hand side shows the fitted functions using XIP-FIT (thick line), P-FIT (thin line), the classical least squares fit (dash-dotted line) and the exact line as a fine dotted line. The two dashed lines

build an approximated 95%–confidence region for the XIP-FIT. The confidence region is computed by constructing 1000 random instances of the fitted parameters within their 95%-confidence interval. The dashed line is then the hull of all curves that are produced by each of these 1000 instances.

The figure on the right shows the simulated measurement uncertainty for the XIP-FIT and the P-FIT (dashed lines) compared to the estimated uncertainties (solid lines) according to Equation (2.30). The simulated uncertainty is computed as follows: Each of $\texttt{Cals} = 200$ random instances of $f^{-1}(y, \boldsymbol{p})$ are applied to $\texttt{Sims} = 20$ random instances of y-values. This yields $\texttt{Cals} \times \texttt{Sims} = 4000$ measured x-values whose variance is plotted at the mean position of those x-values. This process is done for 20 y-positions within the range that should according to $\boldsymbol{p}_{\text{ex}}$ lay in the x-range of the calibration standards

Linear Fit The calibration function is $f(x, \boldsymbol{p}) = p_1 + p_2 x$. The calibration data for Figure 3 is

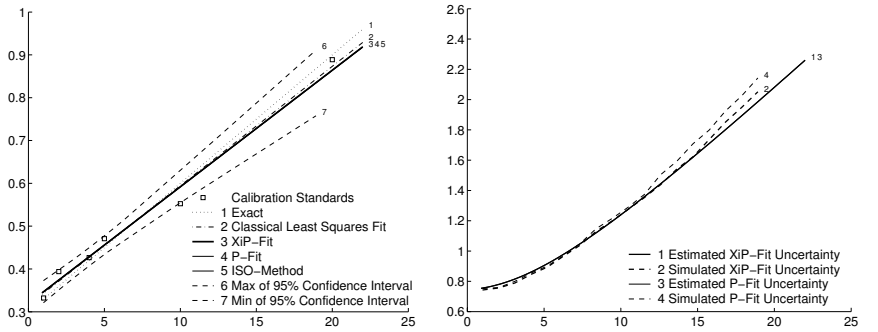
x	1.0000	2.0000	4.0000	5.0000	10.0000	20.0000
y	0.332295	0.393916	0.426571	0.470667	0.552436	0.888801

The covariance in x is given by the following matrix:

$$\boldsymbol{C}_x = \begin{pmatrix} 0.0012 & 0.0016 & 0.0016 & & & \\ 0.0016 & 0.0032 & 0.0032 & & & \\ 0.0016 & 0.0032 & 0.0064 & & & \\ & & & 0.0100 & & \\ & & & & 0.0400 & \\ & & & & & 0.1600 \end{pmatrix}$$

In other words, their relative standard deviation is around 2% and the first three x-values are correlated. The covariance in y is $\boldsymbol{C}_y = 10^{-4}\,\mathrm{diag}(2.7225, 3.24, 4.41, 5.0625, 9, 20.25)$. In other words, they are not correlated and their relative standard deviation is in the order of 5%. Finally the fitted parameters $\boldsymbol{p}$ are as follows:

Figure 3 Fitting a linear function



(a) Fitting a linear function with different methods

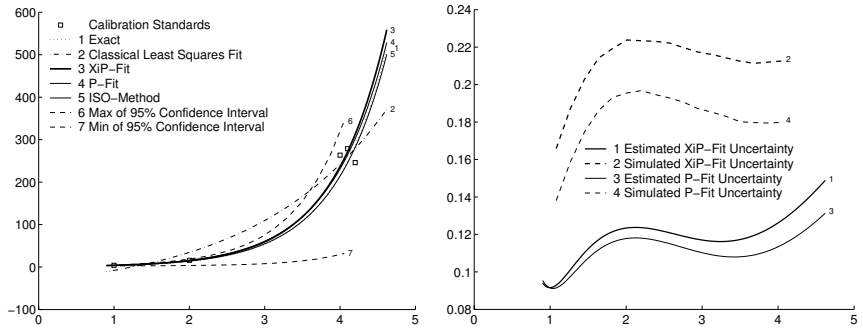
(b) Estimated and simulated measurement uncertainty

	p_1	p_2
Exact	0.3	0.03
Classical LS	0.31558	0.027886
XIP-FIT	0.31971	0.027226
P-FIT	0.31971	0.027204
ISO-Method	0.31972	0.027226

The ISO-Method as sketched on page 35 is implemented in `B-Least`, a software available at BAM.

Remarks The matching between estimated and simulated measurement uncertainty is quite nice. The difference between the compared methods is not very big, in fact for linear functions and if the x-values are not correlated then the XIP-FIT and the ISO-Method can be considered equivalent. The difference in the first parameter is entirely due to the correlation of the x-values. The variation of the second parameter, the slope p_2 between the P-FIT and XIP-FIT is due to the different model assumptions. Note the big uncertainties involved in this example: the y-values for the calibration standards are all assumed to have a standard deviation of 5%. For a measurement of $y = 0.4$, $u_y = 5\%$ this yields an estimated $x = 2.94917$ with $u_x = 0.809 = 27.5\%$! Example 18 on page 89 shows the M input using the data of this example.

Figure 4 Fitting an Exponential Function



(a) Fitting an exponential function with different methods

(b) Estimated and simulated measurement uncertainty

Exponential Fit The calibration function is $f(x, \boldsymbol{p}) = p_1 + p_2 \exp(p_3 x)$. The data for Figure 4 is

x	1.0000	2.0000	4.0000	4.1000	4.2000
y	4.1682	15.9362	263.3923	278.6612	245.8233

The covariance in x is given by the following matrix:

$$\boldsymbol{C}_x = \begin{pmatrix} 0.0075 & 0.0100 & 0.0100 & & \\ 0.0100 & 0.0200 & 0.0200 & & \\ 0.0100 & 0.0200 & 0.0400 & & \\ & & & 0.0420 & \\ & & & & 0.0441 \end{pmatrix}$$

The covariance in y is $\boldsymbol{C}_y = \mathrm{diag}(0.0280, 0.4393, 117.1169, 154.9422, 204.9871)$. Finally the fitted parameters $\boldsymbol{p}$ are as follows:

	$\boldsymbol{p}_1$	$\boldsymbol{p}_2$	$\boldsymbol{p}_3$
Exact	0.1	0.8	1.4
Classical LS	-62.9821	31.2164	0.5699
XIP-FIT	0.39654	0.91511	1.3883
P-FIT	0.47711	0.88251	1.3724
ISO-Method	-0.060929	1.1143	1.3338

Remarks The uncertainties in the calibration and the measurement data are 5%. This is quite considerable. The errors induced by linearization now become obvious. We presume that the simulated uncertainties are therefore bigger but qualitatively they are still acceptable. Note also that the classical least squares fit is way out of bounds. The ISO-Method, despite the different parameter numbers, crowds the picture in between the XIP-FIT and the P-FIT. There was a warning of the `B-Least` software about a division by zero but the output still seems reasonable. The picture on the right was produced with considerably more simulations as in the linear case above. (`Cals` $= 1000$, `Sims` $= 10$).

Power Function Fit The calibration function is $f(x, \boldsymbol{p}) = p_1 x^{p_2}$. Some of the data for Figure 5 is taken from NIST[1].

x	1.3000	1.4700	1.4900	1.5700	1.6000	1.6800
y	2.3093	2.6972	3.9351	3.8846	4.9255	6.2209

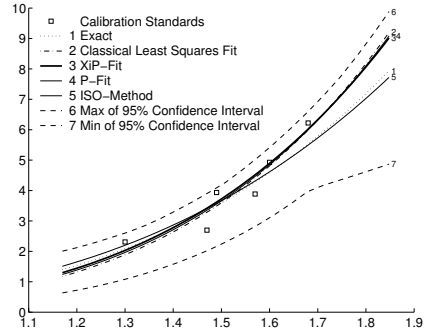
The covariance in x is $\boldsymbol{C}_x = \text{diag}(0.0027, 0.0035, 0.0036, 0.0039, 0.0041, 0.0045)$. The covariance in y is $\boldsymbol{C}_y = \text{diag}(0.0070, 0.0177, 0.0196, 0.0292, 0.0338, 0.0489)$. Finally the fitted parameters $\boldsymbol{p}$ are as follows:

	p_1	p_2
Exact	0.7700	3.8000
Classical LS	0.5801	4.5005
XIP-FIT	0.6676	4.2387
P-FIT	0.8618	3.5703
ISO-Method	0.6264	4.3545

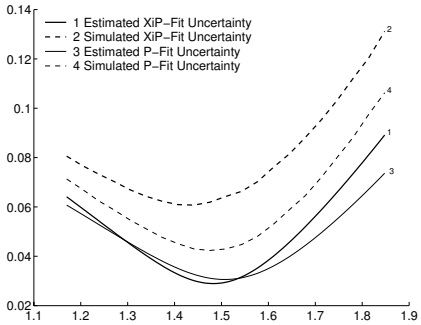
Remarks The uncertainties in the calibration and the measurement data are 4%. Again the linearization induces a shift between the estimated and the simulated measurement uncertainty (`Cals` $= 1000$, `Sims` $= 10$). Note that the P-FIT is visually better in the sense that it is closer to the exact function. This is the case for most of the experiments we have studied. The computation for the ISO-method is done by our own MATLAB-implementation because the `B-Least` software does not offer this type

[1] `http://www.itl.nist.gov/div898/strd/index.html` → Nonlinear Regression → Dan Wood

Figure 5 Fitting a Power Function



(a) Fitting a linear function with different methods



(b) Estimated and simulated measurement uncertainty

of function to be fitted. Despite numerous trials we couldn't find an input data set which would have shown more obvious differences between the methods. Our conclusion is that for fitting a power function the XIP-FIT and the ISO-method are practically equal, whereas the P-FIT seems to be slightly better recovering the true function.

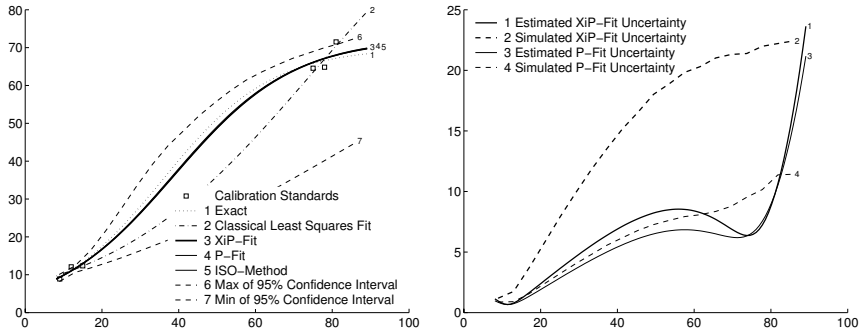
Sigmoid Function Fit The calibration function is $f(x, \boldsymbol{p}) = p_1/(1+\exp(p_2 - p_3 x))$. Some of the data for Figure 6 is taken from NIST[2]. They assign a "higher level of difficulty" to this problem.

x	9	12	15	75	78	81
y	8.8916	12.0857	12.3492	64.5600	64.8196	71.5168

The covariance in x is $\boldsymbol{C}_x = \text{diag}(0.0729, 0.1296, 0.2025, 5.0625, 5.4756, 5.9049)$. The covariance in y is $\boldsymbol{C}_y = \text{diag}(0.0786, 0.1126, 0.1592, 3.8960, 3.9862, 4.0616)$. The fitted parameters $\boldsymbol{p}$ are as follows:

[2] `http://www.itl.nist.gov/div898/strd/index.html` → Nonlinear Regression → Rat42

Figure 6 Fitting a Sigmoid Function



(a) Fitting a linear function with different methods

(b) Estimated and simulated measurement uncertainty

	p_1	p_2	p_3
Exact	70.0000	2.5000	0.0700
Classical LS	124.6656	2.7761	0.0375
XIP-FIT	72.5830	2.4889	0.0641
P-FIT	72.2162	2.4931	0.0650
ISO-Method	72.6064	2.4900	0.0641

Remarks The uncertainties in the calibration and the measurement data are 3%. As opposed to the other examples, the estimated and the simulated measurement uncertainties for the P-FIT are very close. We attribute this to the fact that the uncertainties in the calibration data are the same and relatively small. On the other hand the XIP-FIT seems to yield uncertainty estimates which are too optimistic. For this plot we used `Cals` $= 500$ and `Sims` $= 10$. Note how the measurement uncertainty seems to explode as soon as the calibration function's slope approaches zero.

2.3.7 Final remarks on Regression

Taking the covariance of calibration data into account is important for cases with very strong correlation or with relatively large uncertainties. However,

there is no sound reason to ignore it, since it does not seriously affect the runtime behavior.

The P-FIT leads to slightly higher measurement uncertainties, but it recovers the "exact" parameters generally better — at least in our experiments. This is explicable through its property of "integrating over all possible true ξ-values". On the other hand it is computationally more expensive. The XIP-FIT yields the same results for linear regressions as the ISO-Method as long as no correlations among the x-values are involved. When the function to be fitted is not linear, then the XIP-FIT and the ISO method differ due to the fact that the XIP-FIT linearizes the function taking covariances into account and the ISO method does not linearize the function but ignores covariances. Classical least squares fitting is not generally suitable for calibration tasks. There may be exceptions to this rule in peculiar situations where one knows and/or is well aware about negligible variances.