UPTEC STS08 029
Examensarbete 30 hp
Oktober 2008

To calculate with averages

What is a good result in the absence of the correct?

Niklas Molin

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

To calculate with averages

Niklas Molin

This thesis specifies the foundation for a computer system that operates on averages of measured values provided by automatic measurements systems. This consists of specifying a data object for the average of measured values and fusion methods for that object. The data object should, apart from the value, contain a time and quality description for the average. The fusion methods considered are elementary arithmetic operations to combine different objects and aggregates to yield summarized information on homogenous objects.

The quality of the data objects are described by three attributes; uncertainty, confidence and completeness. The study provides propagation methods for all quality attributes but focuses on the uncertainty of the value. Three different methods for uncertainty propagation are tested. The first method is based on the framework of the corresponding ISO-standard. The others are standard Monte Carlo and quasi-Monte Carlo simulations.

The findings can be used directly as material for an implementation of a system that operates on averages or as material for further discussion. Some of the issues concerning such a system remain unresolved, while others might be optimized.

Handledare: Mats Holmberg
Ämnesgranskare: Torsten Söderström
Examinator: Elísabet Andrésdóttir
ISSN: 1650-8319, UPTEC STS08 029
Sponsor: ÅF Engineering, Nynäshamn

Populärvetenskaplig beskrivning

Dagens industri använder ofta avancerade automatiska mätsystem och detta gäller i synnerhet för processindustrier. Dessa mätsystem nyttjas ofta till styrning och kontroll av processer men kan även utgöra en datakälla för extern användning av mätresultaten. Detta arbete specificerar grunderna för ett datasystem som genomför beräkningar på medelvärden av mätresultat. Med grunderna avses ett medelvärdesobjekt samt beräkningar med denna typ av objekt. Medelvärdesobjektet innehåller förutom själva medelvärdet, en tids- och en kvalitetsbeskrivning av värdet.

Ett av användningsområdena för ett sådant datasystem är att beräkna processegenskaper som är svåra att mäta. Ett aktuellt exempel som presenteras i denna rapport är beräkningen av det totala koldioxidutsläppet från en processindustri. Koldioxidutsläppet är intressant eftersom processindustrier är ålagda att redovisa storleken på sina utsläpp samt köpa utsläppsrätter som motsvarar utsläppet. För att beräkna koldioxidutsläppet i det presenterade exemplet används över hundra uppmätta mätvärden och självklart eftersöks ett så korrekt värde som möjligt. Ett annat användningsområde är kvalitetsäkring av data. Modeller av samband mellan kvantiteter kan användas för att kontrollera de uppmätta värdenas rimlighet och eventuellt även korrigera felaktigheter.

Medelvärdesbildning används eftersom många mätningar genomförs väldigt ofta och därigenom genererar stora mängder data. Tyvärr innebär medelvärdesbildningen en informationsförlust som försvårar specifikationsarbetet. Exempelvis resulterar en multiplikation av två uppmätta storheters medelvärden inte i samma värde som medelvärdet av en produkt av de individuella mätningarna på samma storheter.

En stor del av arbetet ägnas åt att definera en lämplig kvalitetsbeskrivning av data. Kvalitetsbeskrivningen beskriver möjliga skillnader mellan det registrerade medelvärdet och det sanna medelvärdet av den storhet som medelvärdesobjektet representerar. Tre aspekter av datakvaliteten behandlas: osäkerheten i medelvärdena orsakade av precisionsbrist i mätningarna, fel som införts av medelvärdesbildningen och beräkningarna, samt hur stor del av de underliggande mätningarna som anses lyckade. När olika beräkningar genomförs på medelvärdena måste lämpliga metoder användas för att generera en adekvat kvalitetsbeskrivning av resultatet. Dessa metoder behandlas mest utförligt för den osäkerhet i värdet som orsakats av precisionsbrist i mätningarna. Tre olika metoder för denna osäkerhet presenteras och deras lämplighet för denna specifika tillämpning utvärderades med en mindre testapplikation. Den slutgiltiga specifikationen innefattar:

- Ett dataobjekt som innehåller medelvärdet för en variabel tidsperiod, en beskrivning av tidsperioden och en kvalitetsbeskrivning.
- Definitioner av hur dessa dataobjekt kan kombineras enligt de fyra räknesätten, samt regler för att definiera ytterligare matematiska operationer.
- Beskrivningar av ett antal väl valda aggregeringar för dataobjekten.
- En beskrivning av hur medelvärdsobjekt för samma storhet som härrör från olika källor kan kombineras för att öka noggrannheten.

CONTENTS

1 Introduction

Industrial processes often routinely collect large amounts of data. This is done by computers collecting measurements on numerous process variables such as flows, pressures and temperatures. In addition to the process variables, several quality and productivity variables are usually measured. (MacGregor, 1997)

These measurement systems most often store the data and provide an interface for communicating data concerning the measured quantities of the process. This data provided by modern measurements systems isn't restricted to raw data values[1]; such systems often provide fused values such as average over time, standard deviation and the value accumulated over a time period.

Some of the measured variables are thereafter used in activities like reporting, for example, production or greenhouse gas emission reports. But the entities used in reporting don't always correspond to values obtained by measurements. Not all characteristics of a process can be measured. Some are too complicated, others too expensive. Those entities can often be calculated from a combination of measured values.

The basic idea behind this thesis is that a great increase in information value can be gained by constructing a computer system that combines knowledge about the process and the data from the measurement system. This would allow for non-measured process variables to be determined in terms of their relations to measured variables and thereby continuously assigned values. Multiple ways of evaluating a process variable could be defined, thus improving on the accuracy in the values and lowering the amount of missing data points for process variables. Known relationships such as mass and energy balances can be used to verify and correct values and thereby to increase the reliability of the data.

Such a computer system could provide an organization with the information it needs about its production processes. All desired calculation could be performed within the system and thereby eliminate the need for further post processing of the values. An instant application is that it would eliminate the need for the large spreadsheet calculations that's often used to produce the specific entities used in reports and planning activities.

The creation of such a system would require several different building blocks. For example, a database is needed for storage. Another crucial part would be the application that performs the defined calculations. This thesis will focus on the mathematics of that application, i.e. how to treat the data mathematically. The implementation of the application won't be discussed further. But the concepts of his thesis should be applicable regardless whether the application is implemented as stored procedures within the database or as standalone software.

1.1 Aim of this study

The aim of this work is to specify the foundation for a computer based system for historical process data[2] that combines the masses of process data with a mathematical model of the process and thereby increases the information gained by performing

[1] A raw data values is simply the data retrieved from a measurement.
[2] The term historical implies simply that the data originates from prior timestamp.

operations on the measured values. This consists of specifying a data object and how to perform operations on instances of that object. The data objects should, in addition to the values gained from the measurement, contain a quality description of those values. This quality description must include measurement uncertainty as one of its dimension. Measurement uncertainty is a well defined concept and it is often desired to have the uncertainty in measured values stated explicitly. This work's minimum criterion is that the uncertainty evaluation must conform to the directives concerning the reporting of CO_2-emission in Sweden. This specific criterion was chosen since it's an activity that most Swedish process industries must conduct. The activity was also the underlying reason for the decision to stipulate the inclusion of measurement uncertainty in the quality description at this early stage.

The two most prominent features that complicate these specifications are the time requirements and the incorporation of measurement uncertainty into the data object:

- The system should operate on averages valid for specified time intervals rather than discrete data points. Some arguments for this are: sampling rates might differ whereas handling values specified on a common time interval simplifies the system, save storage space and computational time by not performing large calculations and communicating data at an unnecessary high rate.
- The value of a process quantity might be gained from one of several possibilities for each time interval. The uncertainty in the value of a process quantity might therefore vary between time intervals.

This work won't cover algorithms of higher conceptual level, i.e. algorithms that use the process model to verify or improve on the measurements values. For example, the mass and energy balances found in many processes provide interesting opportunities for the construction of such algorithms. But the exploration of those opportunities and development of such algorithms don't fit within the timeframe of this thesis.

The core of the above stated aim can be summarized and rephrased into a couple of questions:

- How can the quality of the data be assured when averaging is used to diminish the data quantity?
- How can the quality of the result be assured when calculations are performed on averaged data?

These two questions will hopefully be answered implicitly by the treatment in the subsequent chapters.

1.2 A Note on the methodology

The contemplated methodology of this study was to conduct a literature review of previous work regarding the subject and thereby identify the most popular methods. The differences in the characteristics of those methods could be used to evaluate the methods and deduce suitable specifications from those results. This structure was abandoned at an early stage due to the poor results of the literature review. Lots of studies have been conducted on most of the notions found this thesis, for example: data quality, measurements systems and measurement uncertainty. Less information was found on the actual problem of this thesis: assuring the quality whilst using averages to diminish the quantity of data.

The lack of input changed the scope of this work slightly; focus was slightly shifted towards finding **a** framework for calculations with averages of measured values (in contrast to finding the optimal framework). A drawback of this approach is the difficulty to evaluate the adequacy and performance of the proposed framework, no evaluation in relative terms can be completed due to the lack of references. The specification was therefore developed using a *best known option* criteria, drawing inspiration and input from related concepts. This methodological shortcoming gives this work a descriptive orientation, resulting in a proposed framework that doesn't claim to be optimal but at least usable.

One aspect of the specification that allows numerical evaluation of the adequacy is measurement uncertainty. These trials and the concept of the measurement uncertainty were therefore given a prominent role in this thesis. The methodology of these trials is described in the chapter 7.

1.2 Disposition

The first part of this thesis, the chapter called background, contains short introductions to three concepts that are important to the remaining parts: measurement systems, data quality and signal sampling. The aim of the chapter isn't to provide a review of the latest research within the corresponding fields but rather to provide the reader with an insight on how the concepts affected and were regarded throughout this work. The second chapter introduces a data object and specifies operations on these objects. The concept of measurement uncertainty is incorporated in the data object but presented in the subsequent chapter. The slight focusing on uncertainty has been made since the concept constitutes an interesting topic; it's subject to requirements, standards and extensive research. Chapter five contains the last part of the specifications, how multiple value sources of the same physical quantity can be combined to provide a more accurate value. It constitutes a separate chapter since it utilizes notions from both preceding specification chapters and therefore must be presented subsequent to them.

Arguments for the design choices and the final specifications are mixed to provide a clearer view of the theoretical foundation for each choice. This mixture of results and theoretical foundation might make it more difficult to distinguish between the results of previous studies and concepts that are innovated in this work. A rule to distinguish the two categories is, as always, that the results of previous studies are referenced to a source presenting those results.

The methods proposed in the chapters on uncertainty and combining multiple sources were subject to testing. The application used and the test cases are presented in chapter six; test case and implementation. The tests, their results and discussions of the results are presented in chapter seven. The thesis is wrapped up with the concluding discussion in chapter eight.

2 Background

This chapter consists of three independent sections that serve the purpose of providing a short introduction to three different concepts. The first section discusses briefly the measurement systems that a system based on the specifications of this thesis would interact with. The second section is a short introduction to the concept of data quality; practical issues of this concept will run through this entire thesis. The third section contains a discussion of how the sampling process affects and constraint the treatment of measured values. No deeper understandings of these concepts are needed to grasp the remaining content of this thesis but the chapter will hopefully provide the uninitiated reader with a better understanding of the remaining chapters.

2.1 Measurement systems

Industrial processes are often equipped with a measurement system that performs measurements at different points in the process. This can be illustrated as in Figure 1, where the points $X_1,\ldots,X_{10}$ signify sensors conducting measurements.

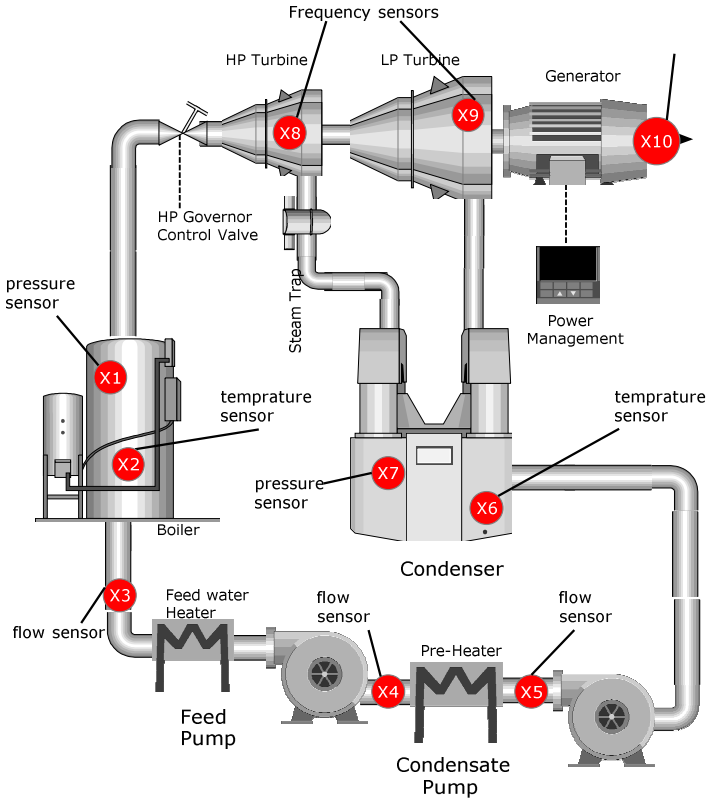

Figure 1 - Sensors in a process unit

The measured quantities marked in the picture will in this thesis be referred to as the process variables (PVs) $X_1,..,X_n$ and the values from measurements conducted with the sensors are the observations $x_1,..,x_n$ of those PVs.

An example of a class of automatic measurement systems are the SCADA systems described in section 2.1.1 below. The PI system by OSIsoft constitutes another, more

concrete, example (OSIsoft, 2008). The section 2.1.2 provides some background on a standard interface for communication of historical data. It's actually that interface a system built on the specifications of this thesis would have to recognize and not the actual measurement system.

2.1.1 SCADA Systems

SCADA is an acronym for supervisory control and data acquisition. As the acronym suggests, these are computer systems for gathering and analyzing real time data. They are used to monitor and control a plant, equipment in industries or other types of complex activities such as a municipal water system (Webopedia: SCADA). There is some confusion around the definition and the distinction towards distributed control systems. A distinction is sometimes introduced by stating that SCADA systems coordinate rather than control processes (Wikipedia: SCADA).

The system usually consists of four types of subsystems (Office of the manager; National communications system, 2004):

- A Master terminal unit (MTU), which gathers data on and sends commands out to the process, the central unit that processes and stores the data.
- Remote terminal units (RTU), the units that actually communicate with the process, for example, a programmable logic controller (PLC) that amongst other things acquires samples of the process from sensors.
- HMI, a human-machine interface that communicates information from the system to its operators, for example, notifying the operators about alarms, visualizing real time data and plotting trends.
- Communication infrastructure; connecting the parts of a real time systems can be nontrivial when they sometimes cover 7000 km of pipelines (Case Study: OPC connects large SCADA Gas Transmission System to Customer Systems).

So the SCADA systems are often quite large and complicated. But this thesis won't require any details about these systems since the MTU of most SCADA systems includes a historical data server (also called historian). The historian logs the data from the SCADA system and supplies the data for external use. The interface provided by the historian could be used in the construction of a system that uses various measured values from the process and therafter performs operations using those values.

2.1.2 OPC foundation

The OPC foundation (the name OPC is an abbreviation for Ole for Process Control) is an organization that produces standards aiming to provide interoperability in industrial automation and enterprise systems that support industry (OPC about). According to themselves, their members include *nearly all of the world's major providers of control systems, instrumentation, and process control systems* (OPC about).Their standard Historical Data Access (OPC HDA) specification (currently at version 1.20) specifies an interface for communicating historical data. An implementation of this interface would allow a system built on the specifications of this thesis to communicate with a wide range of data providers and facilitate other applications communication to the system. The specification will be used as a guideline throughout this work. It will be a guideline in the sense that it provides information about what the system sketched by these specifications can expect when retrieving data from historians. It also provides a list of and information about standard aggregates that should be implemented.

2.2 Data quality

Using sensor based automatic measurements systems restricts the data quality. The sensors themselves have restrictions, often physical, that limit the numerical precision in the measurements. Failures and malfunctions in sensors are commonly occurring, returning erroneous or no values at all. Data processing is likely to amplify the initial error, for example, by the sampling process. Operations like aggregation, combination and evaluation are performed to reduce the data quantity or to extract information. This may summarize the inherent errors but also introduce new ones. Knowledge about the quality of the data used is therefore essential since the data often is used in reporting, decision making and other activities where the output of activity is likely to inherent deficiencies from input data (Klein, 2007).

The notion of data quality is a multidimensional concept. A literature review in the mid 90's showed that a multitude of dimensions were proposed with no consensus of a suitable set nor exact definitions for the proposed dimensions (Wang,1995). Examples of dimensions that were frequently mentioned are; consistency, completeness, accuracy and timeliness.

But there are good examples of sets that aim to give a generic and complete description of the data quality. For example, Wand and Wang (1996) performs in *Anchoring data quality dimensions in ontological foundations* an analysis of data quality based on inconformity between the two views of the real world system; the view obtained by direct observation and the view inferred from the information system. Their analysis generated a set of four data quality dimensions, whether the data can be: complete, unambiguous, meaningful and correct. Their usage of inconformity to discuss dimensions does, apart from aid in the deduction of a reasonable set of dimension, provide useful interpretations of the dimensions. Unfortunately they, as so many others, fail to discuss how the dimensions can be operationalized. Theoretical definitions cannot be directly implemented in methods that assess the data quality, not without operationalization and methods for propagation. The operationalization and propagation methods are likely to be problematic and this work will therefore use notions that give sufficient information about the data quality and focus on the operationalization and propagation of those notions.

Someone who has treated the more practical issues of the quality of data from a continuous stream of samples generated by an automatic measurement system is Anja Klein. She identifies in *Incorporating quality aspects in sensor data streams* the problem with unreliable sensors as a data source for business applications and the need for reliable knowledge about the quality of the data. Her solution is to stipulate that three dimensions are needed for a good description of the quality: accuracy, confidence and completeness. The different dimensions are defined as:

- **Accuracy,** the data quality dimension that describe the numerical precision of measurement data (the errors that the measurement processes introduce).
- **Confidence**, the data quality dimension that signifies the errors that the sampling operators introduce.
- **Completeness**, the data quality dimension that addresses the problem of missing or bad data values due to failures or malfunctions in the measurement process.

Klein also defines methods for assignation of numerical values of those dimensions to specific data and how to propagate those quality measures through different operations.

She provides a whole framework for handling data quality in an application that receives continuous data, exactly what is needed in these specifications.

Her work served as source of inspiration when developing the data object and operations presented in the subsequent chapters. The theoretical discussion in her work may be limping and the theoretical foundation weak but she actually provides a neat framework for working with the three dimensions. Details on this framework and additional information can be found in the original paper. More on how the treatment of data quality in this work was influenced by results found in the literature can be found in the sections concerning the data object and operations on that object.

2.3 Signal Sampling

This work isn't directly concerned with the signal sampling but working with averages of samples or the samples themselves are, of course, related concepts. Two aspects of signal sampling that must be considered in this work are:

- The systems sources of measured data might use different sampling rates.
- There are firm limits on the density needed for datasets to represent a continuous signal accurately; lowering the number of data points or performing operations on a data set might therefore result in information loss.

The aim of a system that operates on averages is, of course, to use averages over longer time intervals than the most frequent sampling in the system while retaining the accuracy and validity of the values.

The Nyquist sampling theorem provides limits for how often an analog signal must be sampled in order to enable exact signal reconstruction. The theorem states that the sampling frequency must be greater than twice the bandwidth of the signal (Glad & Ljung, 1981). Hence to get all sought information from a signal with the highest frequency of interest at 1 Hz, the signal has to be sampled more than twice a second. The sampling process in practical applications does often operate on a significantly higher sampling frequency than twice the bandwidth (Wikipedia: Sampling).

The sampling rates of a measurement system are hopefully set in a correct manner, resulting in data sets that represent the measured variables well enough. But the specification of a system that performs calculations on the measured data is facilitated if all PVs have data points available at corresponding time stamps. This desired functionality can always be fulfilled since upwards sampling rate conversion can be applied to the PV that are not the most frequently sampled. The upwards sampling rate conversion is achieved by interpolation (John Watkinson, 2002). In the most straightforward case, where the sample rate is doubled, points are added halfway between the original samples and their value gain by a suitable interpolation method (Wikipedia: Sample rate conversion).

There are normally restrictions to the rate (or density) of data points when calculating and storing numerous variables. A variable described by a data point per second would need approximately one MB per day in storage space in an OPC Historian (Kirrmann, 2005). Hence a historian logging measured values of 1000 variables would need one GB of storage space per day and 365 GB per year. A reasonable representation in Java, representing the value by a double and the timestamp by a long, corresponds to a usage of 1.3824 MB per variable/day for variables stored once per second. Storing these amounts of data wouldn't constitute a problem. But a slower rate might be preferable if

each data point has to be stored, communicated through shared networks and used in calculations.

The other, theoretical, option to obtain a dataset where all variables have points at the same points in time would be to downsample (downwards sample rate conversion) the more frequent sampled variables. In digital signal processing, downsampling (or subsampling) should ideally be carried out with respect to the Nyqvist sampling theorem.

The operational frequency of the system can be lowered by introducing averages. Averaging can be regarded as a sort of subsampling but changes the situation slightly. Averages formed from a sufficient set of data points, are indeed valid average values for those PVs. But what happens when two averages are combined to constitute an average of a different PV? Can such operations be considered valid? Table 1 illustrates two such operations; addition and multiplication. The example in the table indicates that the linear operation addition produces a correct average value for the output even when the two underlying datasets contains variation. The nonlinear operation multiplication seems to be more problematic, multiplication of the two averages of X_1 and X_2 results in 18.24 which differs from the result gained by multiplying the individual samples.

Sample nr	X_1	X_2	X_1+X_2	X_1*X_2
1	2	6	8	12
2	5	4	9	20
3	2	4	6	8
4	1	7	8	7
5	7	9	16	63
6	3	2	5	6
7	6	3	9	18
8	1	7	8	7
9	4	4	8	16
10	7	2	9	14
Average	3.8	4.8	8.6	17.1

Table 1 - Combining and averaging of two PVs

The error for the multiplication of the averages in Table 1 can be explained in terms of the Nyquist sampling theorem. If the intervals were chosen too long, i.e. the true values of the actual quantities measured vary within the interval and then nonlinear operations on the averages will produce erroneous results. Hence, choosing the time interval for the averages too long can be considered synonymous with choosing the sampling rate to low. If the intervals were chosen adequately, then the measurements were affected by some erroneous effects, for example, signals of higher frequency than those of interest to the measurement. Then multiplying the averages probably provides a more accurate value and the difference against multiplying each pair of samples is due to cancellation of the erroneous effects.

3 Object and operations specification

This chapter specifies the data object, analogues of elementary arithmetic operations for the data objects and slightly modified implementations of the functions specified as standard aggregates in the OPC HDA.

3.1 Data object

The requirement on the specifications to consider average values rather than discrete data points simplifies the specifications in many aspects. It also brings on problems due to the loss of information caused by substituting several raw values for their average. The problem of choosing suitable time intervals for the averages will be treated by introducing the notion of unit time and the concept of active time.

Unit time is the length of the common time interval for the averages of PVs. The length of the unit time should ideally be set in accordance with the bandwidth of the measured PVs, the operations performed on the PVs and the aim of the system. The average value of a PV over a unit time interval incorporated in the data object of this section will be denoted a **unit time average**.

The information lost when averaging is slightly reduced by giving the data objects an attribute corresponding to the length of the subinterval where the measured quantity has been an active part of the system treated. The length of this subinterval will be noted the **active time** of a unit time average. This active time can, of course, also be used as a switch indicating when a PV should be included in calculation and so on, this by setting the active time to zero.

The description of the quality of the data will include two measures of the accuracy and one of the completeness. The accuracy of the data will be divided into two categories to make a distinction between both the origin and nature of the imperfections. The categories are, as in the work of Anja Klein, imperfections in the measurement of a single data value that lowers the accuracy and the errors introduces by the sampling process. The first category will be called **uncertainty** and the second **confidence**. Measurement uncertainty is a well defined concept and some activities, such as reporting CO_2-emission, require that the concept of uncertainty is used to establish the quality of the data used to calculate the emission. Using the concept of uncertainty and conforming to the standards and regulations concerning the concept is therefore beneficial for systems that perform operations on automatically measured process variables.

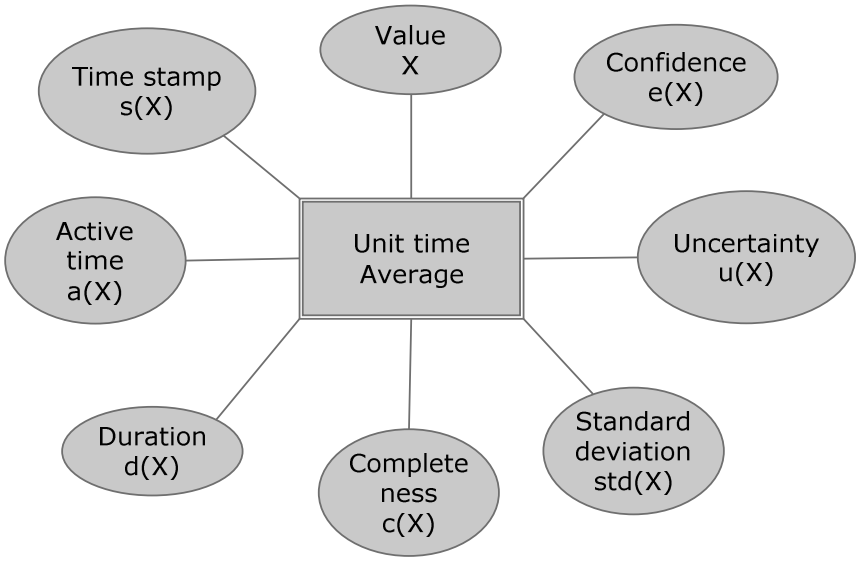

Figure 2 - Data object

A suitable object to represent the unit time averages is presented in Figure 2. There is a self-explanatory need for attributes containing the average value and the time period for which the average value is valid. The object proposed in Figure 2 uses a slightly more detailed time description. Active time has been added to the basic description of a time interval consisting of start time and duration. This inclusion aims to reduce the erroneous effect of processes not even being active for certain parts of the time period, as mentioned above.

The three attributes completeness, confidence and uncertainty can be regarded as different attributes describing the quality of the data.

The standard deviation is included since some value of the spread might be desired. No other attribute offers any information about the variation within the time periods. An application would be to verify the measured values, for example, unexpected spread in the values or constant values can indicate corrupt measurements. It's also used in the propagation of confidence. All the attributes of Figure 2 are described more elaborately in separate sections below.

3.1.1 Value

The average value of a data object consists of a measured or calculated value that is valid for the time period specified in the data object. The time period is the interval starting at the time stamp and ending the number of time units specified as duration after the time stamp. The average aggregate specified in OPC HDA can be a regarded as a specification on how to compute the average values and as an example of a possible interface towards a source providing average values to the system. The OPC HDA specifies the average aggregate to be the arithmetic average of all good raw values in the interval. The section on completeness contains a discussion on the significance of a value being of good quality.

3.1.2 Uncertainty

Uncertainty is an attribute that expresses the error in the value that was caused by imprecision in the measurements. This concept will be clarified in the chapter with the same name.

3.1.3 Confidence

Klein defines confidence as a parameter $\varepsilon(X)$ such as the interval $[x-\varepsilon(X), x+\varepsilon(X)]$ constitutes a 95%-confidence interval for the PV *X* for which *x* is a random sample from the time interval specified by time stamp and duration (Klein, 2007). A normal interpretation is that there is a 95 percent chance that the true value of the variable *X* lays within the interval (Klein, 2007). The interpretation and usage here will be slightly different; the confidence will be used to indicate undesired variance and erroneous effects due to the treatment unit time averages derived from varying data points. The intervals produced by applying the proposed formulas aren't guaranteed to hold a 95 percent confidence level. Both uncertainty and confidence should be regarded as attributes that penalize the accuracy in the value. But confidence is a less exact concept and shouldn't be combined with uncertainty to produce a common confidence interval. Having two different attributes that describe the accuracy won't constitute a problem in systems where the unit time has been chosen carefully. Those systems will have low confidence on the unit time averages and the confidence will decline further when the PVs are aggregated over time.

A numerical value for the confidence of a unit time average where the input consists of raw samples can be calculated in accordance with the samples estimate as

$$\varepsilon(X) = \left(\frac{d^2 \sigma^2(X)}{n} \right)^{1/2} \tag{3.1}$$

where σ^2 is the variance of *X* in the interval, *n* is number of all data points in the interval and d is the coverage factor that expands the interval to a 95 percent confidence level (Klein 2007). The coverage factor should be derived from the student's t-distribution with *n-1* degrees of freedom but the value two can be used as an approximation for systems where n has a minimal value around 20. Hence the confidence will be zero for unit time averages computed from intervals containing only non-bad data points (non-bad data points will be defined under completeness below). Klein introduced this notion to penalize a data stream that has been subject to downsampling. Missing data points can be considered as unintentional downsampling and boosting the confidence of intervals containing bad data is therefore appropriate.

If any of the inputs used to construct the unit time averages has been subject to any other downsampling technique and therefore been assigned a confidence value, then those confidence values has to be incorporated in $\varepsilon(X)$. The new expression can be constructed as

$$\varepsilon_N = \frac{\sqrt{\sum_{i=1}^{n} \varepsilon^2(X_i)}}{n} + \left(\frac{d^2 \sigma^2(Y)}{n} \right)^{1/2} \quad (3.2)$$

where $\sum \varepsilon^2(X_i)$ is the sum of confidence of the inputs and all parameters have the same significance as in equation (3.1) (Klein, 2007).

The confidence is important for variables that are constructed from problematic combinations such as multiplication; an explanation of problematic combinations can be found in the section sampling of signals. The importance is guarded when the variable is aggregated or combined further. This will hopefully be fully clarified in the description of the operations and aggregations, where the confidence will be reduced for linear operations and aggregations and increased for nonlinear ditto. Appendix A contains a more elaborate description of the concept and motivations of the proposed formulas.

3.1.4 Completeness

The completeness of a unit time average will be defined as nothing else but the percentage of samples that were included in the calculation of the average (it assumed that the averages where calculated over an interval filled with data points separated by a fixed distance). There is no absolute interpretation of this quality parameter such as those describing the accuracy, but it provides information on both the amount of substance there is behind the numerical values and something about the success and thereby reliability of the automatic measurement system. Hence, a low completeness implies the direct effect of less reliable numerical values due to fewer values used in the computation. It also implies an indirect effect on the reliability due to the closeness in time between the values used in the calculation and malfunctions in the measurement system, which implies that the values used might have been affected by erroneous effects (this stipulated indirect effect would be avoided by a system that could identify all malfunctions in the measurement process).

OPC has three categories to indicate the quality of data; good, uncertain and bad (Data Access Custom Interface Standard, OPC). The exact definition of the categories is server dependent. But assigned an identifier representing the quality of each OPC data object is mandatory.

According to the specification, all OPC-HDA aggregates should omit bad data from the calculation. Whether uncertain values should be included or not is server dependent. If all data points in the time period are included in the calculation, then the quality is good. If any values are omitted from the calculation then the quality of the aggregates is uncertain.

OPC specifies that only good raw values should be used to calculate certain aggregates, when can a unit time average be consider good? Extrapolating from the propagation of quality in the OPC specification would imply that only unit time averages with 100 percent completeness should be considered good. No exact definition will be given here and the section below will use the term good quality but this should be interpreted as unit time averages with sufficient completeness. A system built after these specifications should define an exact level of completeness for values to be considered of good quality.

3.1.5 Time Stamp

The time stamp is the point of time (expressed in GMT) that initiates the time period for which the value is valid.

3.1.6 Duration

The duration is the length of the time period expressed in unit time for which the value is valid.

3.1.7 Active Time

Active time represents the length of the time period where the PV has been an active part of the production process. The length of this time interval should be expressed in unit time.

3.1.8 Standard Deviation

The standard deviation of the raw data points is defined in the OPC HDA version 1.2 and that definition will be reused here. Values representing a time period of one time unit derived from good raw data have the deviation from the mean of those good data points. A constant or a single raw data point has a standard deviation of zero. OPC specifies the formula to be used as (3.23).

3.2 Basic operations

Since PVs of interest have some relationship to the physical reality there are also given relationships between process variables in the system. These relationships can be used to combine different measured values to obtain values of nonmeasured PVs, alternatively values for measured PVs or quantities that aren't found in the process. The set of operations on PVs presented in this section consists of the elementary arithmetic operations: addition, subtraction, multiplication and division. But there is no logical need for this restriction; the set can easily be expanded to contain other mathematical functions. The elementary arithmetic operations were chosen since they are assumed to be the most commonly occurring operations and constitute intuitive examples. A general note for all kinds of operations is that there has been a loss of information when creating the averaged values representing a time unit. How this affects the value was discussed in the section sampling of signals but it also influences other attributes.

The active time represents the time a PV has been an active part of the process within the specified interval for which the process variable is valid, but if the active time is smaller than the interval there is no way of telling where in the interval the PV has been active. The general rule used to define these operations is: *two process variables being combined are assumed to be active at the same points in time to the largest extent possible.* There is no mathematical foundation for this rule, but it can be argued to be a good assumption in relation to reality and won't constitute a problem in a well defined system were the unit time is chosen appropriately.

For what time interval is the output valid? This is another impossible problem, there is no way of knowing unless the real time interval of the output is explicitly defined in the system. One rule can be stated, an effect cannot occur earlier in time than its cause. Therefore, to be consistent with the assumption made about active time, the timestamp plus the duration of the output cannot be earlier than the timestamp of any of its inputs having active time. The solution to this problem is to give the output the timestamp and

duration of the, in respect to time, latest of its inputs. This is to be considered as the general rule but it causes some special cases that will be treated in the definitions below (for example, no PV should have a greater active time than its duration). An implementation with a strict usage of unit time and common timestamps for all PVs largely evades this problem.

The quality parameter confidence is designed to describe the loss of information due to variations in the sampling rate. These deviations described by the confidence will be assumed to be completely random which implies that the confidence parameters of two data object can be assumed independent. The confidence parameters will therefore be combined by Gaussian error propagation as in (3.3). (Klein, 2007)

$$\varepsilon(Y) = \sqrt{\sum \left(\frac{\partial Y}{\partial X_i} \right)^2 \varepsilon^2(X_i)} \tag{3.3}$$

Nonlinear operations will add further terms to the equation to compensate for the error they might introduce. The general form for confidence propagation for nonlinear operations:

$$\varepsilon(Y) = 2\sqrt{\left(\frac{\partial Y}{\partial X_1} \right)^2 \varepsilon^2(X_1) + \left(\frac{\partial Y}{\partial X_2} \right)^2 \varepsilon^2(X_2)} \tag{3.4}$$

with the same denotation as in equation (3.2). This penalization can be described as the error generated when using unit time averages instead of the data points that used to calculate the averages, assuming that there was no covariance between the inputs.

The last problematic parameter is the standard deviation. Since the correlation between the two process variables most probably will be unknown there is no way of calculating the standard deviation for the output correctly. The calculation will therefore be performed equally to how the operations would be performed on two independent PVs. Hence the similarity between the standard deviation and confidence continues.

A couple of numerical examples of operations between two PV are presented in appendix B. The first operation in the example is the addition presented in section 3.2.1 below. The second is the multiplication specified in section 3.2.3.

3.2.1 Addition

The addition of to process variables is not simply the addition of their value parameters. Since they represent average values of physical parameters, the active time (denoted $a(X)$) of that measurement has to be considered for the calculated output to be a correct average value over the new active time. The output will therefore be calculated as the accumulated sum of the inputs and thereafter divided by active time of the output. Hence, the value of the output *Y* will be:

$$Y = \frac{a(X_1)\cdot X_1 + a(X_2)\cdot X_2}{\max(a(X_1), a(X_2))} \tag{3.5}$$

The timestamp of Y will be set to the earliest of the inputs timestamps and the duration according to:

$$d(Y) = \begin{cases} \max(d(X_1), s(X_2)) & if \quad s(Y) = s(X_1) \vee s(Y) = s(X_1) = s(X_2) \\ \max(d(X_2), s(X_1)) & if \quad s(Y) = s(X_2) \end{cases} \tag{3.6}$$

Since the two inputs are assumed to have their active time in common to the largest extent possible and addition implies that the output is active when either of the inputs is, the active time of the output will be the largest active time of the inputs.

$$a(Y) = \max(a(X_1), a(X_2)) \tag{3.7}$$

The standard deviation, calculated as for independent variables (Råde & Westergren, 1998):

$$\begin{aligned} \mathrm{std(Y)} &= \sqrt{E[(\frac{a(X_1)\cdot X_1 + a(X_2)\cdot X_2}{\max(a(X_1), a(X_2))})^2] - \left(E\left[(\frac{a(X_1)\cdot X_1 + a(X_2)\cdot X_2}{\max(a(X_1), a(X_2))}) \right] \right)^2} \\ &= \sqrt{\frac{a(X_1) std^2(X_1) + a(X_2) std^2(X_2)}{\max(a(X_1), a(X_2))}} \end{aligned} \tag{3.8}$$

The confidence, according to equation (3.3):

$$\varepsilon(Y) = \sqrt{\left(\frac{\partial Y}{\partial X_1}\right)^2 \varepsilon^2(X_1) + \left(\frac{\partial Y}{\partial X_2}\right)^2 \varepsilon^2(X_2)} \tag{3.9}$$

where $\left(\frac{\partial Y}{\partial X_1}\right)^2 = \left(\frac{a(X_1)}{\max(a(X_1), a(X_2))}\right)^2$ *and* $\left(\frac{\partial Y}{\partial X_2}\right)^2 = \left(\frac{a(X_2)}{\max(a(X_1), a(X_2))}\right)^2$.

3.2.2 Subtraction

Subtraction of two process variables is almost like addition. The value will, of course, be calculated slightly different. But apart from that, the logic and assumptions are the

same as for addition and therefore most of the parameters will be defined in a similar manner. The numerical Value is calculated as:

$$Y = \frac{a(X_1)\cdot X_1 - a(X_2)\cdot X_2}{\max(a(X_1), a(X_2))} \tag{3.10}$$

Everything else can be calculated according to the rules of addition.

3.2.3 Multiplication

Multiplication suffers from insufficient information about both activity and variation in the inputs. The value, ignoring possible correlation between the inputs, is calculated as:

$$Y = X_1 \cdot X_2 \tag{3.11}$$

Active time, calculated under the assumption of the largest possible common active time:

$$\min(a(X_1), a(X_2)) \tag{3.12}$$

Standard deviation, calculated as[3]:

$$std(Y) = \sqrt{\overline{X}_2^{\,2} std^2(X_1) + \overline{X}_1^{\,2} std^2(X_2) + std^2(X_1) std^2(X_2)} \tag{3.13}$$

where $\overline{X}_i$ represents the observed unit time average and is used as an estimator of the expectation value. The confidence, penalized, according to (3.4), to compensate for variation within the unit time average inputs:

$$\varepsilon(Y) = 2\sqrt{\overline{X}_2^{\,2}\varepsilon^2(X_1) + \overline{X}_1^{\,2}\varepsilon^2(X_2)} \tag{3.14}$$

---

[3] If X and Y are independent, so are X^2 and Y^2 .Hence $E\left((XY)^2\right) = E\left((X)^2\right)E\left((Y)^2\right)$
and:

$$\mathrm{std}^2(Y) = E[(X_1X_2)^2] - (E[(X_1X_2)])^2 = E[X_1^{\,2}]E[X_2^{\,2}] - (E[X_1])^2(E[X_2])^2 + E[X_1^{\,2}](E[X_2])^2 - E[X_1^{\,2}](E[X_2])^2$$
$$= E[X_2]^2 std^2(X_1) + E[X_1^{\,2}] std^2(X_2) + E[X_1]^2 std^2(X_2) - E[X_1]^2 std^2(X_2)$$
$$= E[X_2]^2 std^2(X_1) + std^2(X_1) std^2(X_2) + E[X_1]^2 std^2(X_2)$$

3.2.4 Division

Is basically the same operation as multiplication, a division can be converted into a multiplication with the valid substation $X_3 = X_2^{-1}$ and vice versa. This affects the calculation of some attributes, for example, the numerical Value:

$$Y = \frac{X_1}{X_2} \tag{3.15}$$

Standard deviation:

$$\mathrm{std}(Y) = \sqrt{\frac{1}{\bar{X}_2^{\,2}} std^2(X_1) + std^2(X_1) std^2\left(\frac{1}{X_2}\right) + \bar{X}_1^{\,2} std^2\left(\frac{1}{X_2}\right)} \tag{3.16}$$

Confidence, according to (3.4):

$$\varepsilon(Y) = 2\sqrt{\frac{1}{\bar{X}_2^{\,2}} \varepsilon^2(X_1) + \frac{\bar{X}_1^{\,2}}{\bar{X}_2^{\,4}} \varepsilon^2(X_2)} \tag{3.17}$$

3.3 Aggregation

The aggregations that are to be considered here are those specified as standard aggregates in OPCs specification of historical data access version 1.2 (OPC HDA). The standard aggregates of the specification contain, for example: arithmetic average, min, max and count. The idea is not to provide strict implementations for the standard aggregates, rather to provide implementations of meaningful counterparts of them. The standard aggregates won't always have meaningful interpretations since these specifications are concerned with unit time averages rather than raw data points. However, the list of standard aggregates provided in the OPC standard is used because it's assumed to contain commonly occurring aggregates whose counterparts probably are a desired functionality of a system that provides values and aggregates of PVs.

An incremental structure in the calculations of the aggregates is desirable. An incremental structure where each data object is used once doesn't require the whole data set of the aggregation to be kept in memory throughout the calculation (or alternatively, thrown away and thereafter retrieved, once again, from the data source) (Holmberg, 2008).

Formulas that calculate the aggregates incrementally will therefore be identified whenever possible. This will be achieved by dividing the standard aggregates into parts that can be calculated incrementally and those parts summarized in the aggregate object. The specification of an aggregate object that contains all of the incremental parts is mostly for illustrative reasons but can facilitate implementations where all standard aggregates of PV are calculated automatically since many of the incremental parts

reoccur in several standard aggregates. Predefined automatic calculation of aggregates is motivated by large computational time for aggregates of some PVs. A common approach is to automatically calculate aggregates over predefined time periods such as days, weeks, months and years (Holmberg, 2008). The incremental parts needed to calculate the aggregates will be identified in the sections describing each and every one of the standard aggregates.

3.3.1 OPC standard aggregates

All the definitions concerning the standard aggregates in this section can be found in the OPC HDA. The OPC standard aggregates will be regarded, as mentioned in chapter 2, as nothing more than a suitable list of mathematical aggregation functions to implement. The definitions will therefore be modified to fit a system where the basic data are averages. All references to the OPC specification in this section should be regarded as proposals, possibilities or expected functionality of the measurement systems that constitute the contemplated data sources.

The specification states that each OPC standard aggregate must be specified by three time parameters; start time, end time and sampling interval. The specification defines what data points are to be used and how to inter-/extrapolate when data points are missing for different combinations of the inputs. (OPC HDA)

But there is no logical need for implementations of these aggregates to restrict the specification of the data included to the three time parameters of the OPC HDA. Any other attribute of the data objects can be used to constrain the data included. For example, active time can be used to exclude time periods where the PV hasn't been an active part of the process or a minimum limit on the completeness used to select only reliable data. No specification of suitable methods for constraining the data will be presented here. The formulas and object presented in the sections below are specified for a set of unit time averages as input and aren't affected by the selection of this set.

The specifications of the standard aggregates use OPCs concept of data quality. The quality of the aggregates is specified as: good, uncertain or bad (OPC HDA). Most standard aggregates are considered to be of good quality if all raw values in the specified time interval are of good quality (OPC HDA). The quality of the aggregate is uncertain/subnormal if any of the raw values in the time interval are of uncertain or bad quality (OPC HDA). Exceptions to these rules will be described in the corresponding section below. This thesis interpretation of good quality was discussed in the section 3.1.4 and applies to the aggregates in the same manner.

The sections below won't specify the completeness or any other quality measure for each single aggregate but the overall quality of the aggregate object will be described in terms of the three quality parameters: uncertainty, confidence and completeness.

3.3.1.1 Time average or weighted average

The definition of the standard aggregate **time average** is a function that draws straight lines between raw data points, thereafter calculates the area under the lines and divides the areas by the length of the corresponding time interval (OPC HDA). This definition must be slightly modified since this system utilizes active time as a basis for time-dependent calculations. **Time average** will therefore be defined as the sum of unit time averages weighted against their active time. The interpolative approach of drawing straight lines between the data points is redundant since the unit time averages constitutes values valid for intervals of time rather than instants. This is calculated as:

$$\text{time average}(X) = \frac{\sum_{i=1}^{n} a(x_i)x_i}{\sum_{j=1}^{n} a(x_j)} \tag{3.18}$$

where *n* is the number of unit time averages with start time in the time interval.

The definition of the time average can easily be extended to a more versatile aggregate, the **weighted average**. The **time average** is a useful concept when dealing with accumulating quantities. But time average lacks meaningful interpretation for non-accumulating quantities such as density; it wouldn't give any information about the average density for the corresponding mass. For the average of density over a time period to make sense it has to be weighted against the volume flow over the time period. Thus the **weighted average**:

$$\text{weighted average}(X) = \frac{\sum_{i=1}^{n} c_i x_i}{\sum_{j=1}^{n} c_i} \tag{3.19}$$

where each c_i represents the weight associated with the unit time averages x_i in the interval.

These two formulas can be divided into the following incremental parts:

$\sum_{i=1}^{n} c_i x_i$, $\sum_{i=1}^{n} c_i$, $\sum_{i=1}^{n} a(x_i)x_i$ and $\sum_{i=1}^{n} a(x_i)$. Calculating the two aggregates from these incremental parts should be self-evident from the formulas above.

3.3.1.2 Total

Total is defined by OPC as the product between time average and the length of the time interval (OPC HDA). The time average is the result from a time average call with the parameters inherited from the total call and the interval length should be expressed in seconds (OPC HDA). The total aggregate is only meaningful for accumulating process quantities:

$$total(X) = time\ average(X) \cdot \sum_{i=1}^{n} a(x_i) \tag{3.20}$$

No new incremental parts have to be added to the aggregation object to calculate (3.20).

3.3.1.3 Average

The OPC definition of the **average** aggregate differs from time average by not considering the length of subintervals. Instead, this function simply adds up the values from all good data points in the interval and divides by the number of good data points.

$$average(X) = \frac{\sum_{i=1}^{n} x_i}{n} \tag{3.21}$$

Two new incremental parts has to be added to the aggregate object; $\sum_{i=1}^{n} x_i$ and the number n of good unit time average in the interval.

3.3.1.4 Count

Count returns the number of good raw data points within an interval. If any data points are non-good they are excluded from the count and this also lowers the quality of the aggregate to uncertain/subnormal. The equivalent for unit time averages is self-evident and no new incremental part is needed.

3.3.1.5 Standard deviation

In the OPC specification the standard deviation is defined as

$$Std(X) = \begin{cases} \sqrt{\frac{\sum_{i=1}^{n} (x_i - Average(X))^2}{n-1}} & n = 2,3,.....\infty \\ 0 & n = 1 \end{cases} \tag{3.22}$$

where X are the n good data points in the interval. If $n = 1$ the functions should return zero. This can be rewritten as

$$Std(X) = \begin{cases} \sqrt{\frac{\sum_{i=1}^{n} x_i^{\,2} - 2Ave(X)\sum_{i=1}^{n} x_i + nAve(X)^2}{n-1}} = \sqrt{\frac{\sum_{i=1}^{n} x_i^{\,2} - nAve(X)^2}{n-1}} & n = 2,3...\infty \\ 0 & n = 1 \end{cases} \tag{3.23}$$

implying that the standard deviation is constructed from three blocks that can be incrementally calculated over the interval, the average value and number of good unit time averages is already identified above, the only new part is $\sum_{i=1}^{n} x_i^{\,2}$, the sum of squares.

3.3.1.6 Variance

The variance is the square of the standard deviation and inherits therefore the behavior of the standard deviation aggregate.

3.3.1.7 Regression coefficient and constant

A regression line is, according to the specification, a “line-of-best-fit” over the interval.

A least square-estimate of a line $\hat{x}_i = a \cdot o_i + b$ is given by

$$\hat{\theta} = \left[\sum_{t=1}^{N} \psi(t)\psi^T(t) \right]^{-1} \sum_{t=1}^{N} \psi(t) x_t \tag{3.24}$$

assuming that the matrix

$$R_N = \sum_{t=1}^{N} \psi(t)\psi^T(t) \tag{3.25}$$

is invertible. $\hat{\theta}$ is an approximation of the $\theta = \begin{bmatrix} b & a \end{bmatrix}^T$ that minimizes the sum of squares

$$\sum_{t=1}^{N} (x_t - \hat{x}_t)^2 \tag{3.26}$$

and $\psi(t) = \begin{bmatrix} 1 & o_t \end{bmatrix}^T$. The o_t corresponds to the time offset from the aggregates start time to the start of the unit time average with index t. The two sums in the expression can be divided into parts during the calculations. The parts needed for the final evaluation would be: $\sum_{t=1}^{N} o_t$, $\sum_{t=1}^{N} {o_t}^2$, $\sum_{t=1}^{N} x_t$ and $\sum_{t=1}^{N} o_t x_t$ for all good unit time averages in the interval.

The incremental parts can be inserted into (3.24) and evaluated as:

$$\hat{\theta} = \left[\sum_{t=1}^{N} \begin{bmatrix} 1 \\ o_t \end{bmatrix} \begin{bmatrix} 1 & o_t \end{bmatrix} \right]^{-1} \sum_{t=1}^{N} \begin{bmatrix} 1 \\ o_t \end{bmatrix} x_t = \frac{1}{N\sum_{t=1}^{N} {o_t}^2 - \sum_{t=1}^{N} o_t \sum_{t=1}^{N} o_t} \begin{bmatrix} \sum_{t=1}^{N} {o_t}^2 \sum_{t=1}^{N} x_t - \sum_{t=1}^{N} o_t \sum_{t=1}^{N} o_t x_t \\ N\sum_{t=1}^{N} o_t x_t - \sum_{t=1}^{N} x_t \sum_{t=1}^{N} o_t \end{bmatrix} \tag{3.27}$$

All the products of sums in (3.27) might look complicated at a first glance, but once the sums are computed, evaluating (3.27) constitutes of a few elementary operations.

3.3.1.8 Minimum/Maximum And Minimum/Maximum actual time

This OPC aggregates minimum/maximum should simply return the minimum alternatively maximum of all good raw values (OPC HDA). The equivalent here is the aggregate that returns the minimum/maximum of the unit average values. The minimum/maximum actual time aggregates returns the same value but it also provides the timestamp of the minimum/maximum values occurrence (OPC HDA). The oldest

minimum/maximum should be returned if there is more than one occurrence of the minimum/maximum value within the interval (OPC HDA). The quality of the aggregate is subnormal/uncertain if any non-good value is lower/higher than the minimum/maximum good value (OPC HDA). The min/max value and their timestamps, though not incremental in structure, have to be stored in the aggregate object.

3.3.1.9 Start/end

The start aggregate must return the value of the first raw value, and the end should return the last (OPC HDA). The corresponding aggregate for unit time averages is self-evident and OPC defines the quality of the aggregate as the quality as that of the raw value returned.

3.3.1.10 Delta

Delta is defined as the difference between the first good raw value and the last good raw value (OPC HDA). Hence the corresponding aggregate for unit time averages returns the difference between the first and last unit time average. The quality is subnormal/uncertain if any non-good values exist before the first good or after the last good value (OPC HDA). The first and last good unit time averages have to be stored continuously in the aggregate object to enable incremental calculation of this aggregate.

3.3.1.11 Duration good, percent good, duration bad and percent bad

These aggregates correspond to the duration and percentage of the total duration of good/bad raw values in the interval (OPC HDA). The equivalent for unit time averages can be formed by simply substituting the raw values for the unit time averages. The other possible interpretation of these aggregates, considering the percentage of duration in the values used to calculate the unit time averages, is covered by the completeness.

3.3.1.12 Worst quality

The OPCs definition specifies that worst quality aggregates must return the worst quality found amongst the raw data in the interval (OPC HDA). The value of this aggregate can be improved by returning the lowest completeness of all unit time averages of the interval. Hence the lowest completeness has to be stored in the aggregate and updated when it's appropriate.

3.3.2 The aggregation object

The quality of the aggregates should be described by the same parameters as the unit time averages. Klein, in her treatment of data streams, specifies the completeness of an aggregate to be the average completeness of all incoming tuples, indifferent of the aggregate operator used (Klein, 2007). This specification inherited the concept of completeness from her work and will therefore perform the aggregation equivalently, calculating the completeness as the average completeness of all unit time averages in the aggregation. But defining the uncertainty and confidence is problematic, the interpretation of the parameters must remain the same and some aggregates, for example, duration good, doesn't even relate to the numerical value of the PVs. The propagation of confidence should be conducted according to (3.3) or (3.4), with a slight modification of the formula to consider different unit time averages for the same PV rather than combining different PVs (Klein, 2007). This formula applied to the weighted average will be considered as an adequate confidence measure for the entire aggregate object. The specification of confidence for the entire aggregate object is a good

substitute for specifying the propagation of this quality parameter for each of the standard aggregates. How to propagate the uncertainty is treated in chapter 4.

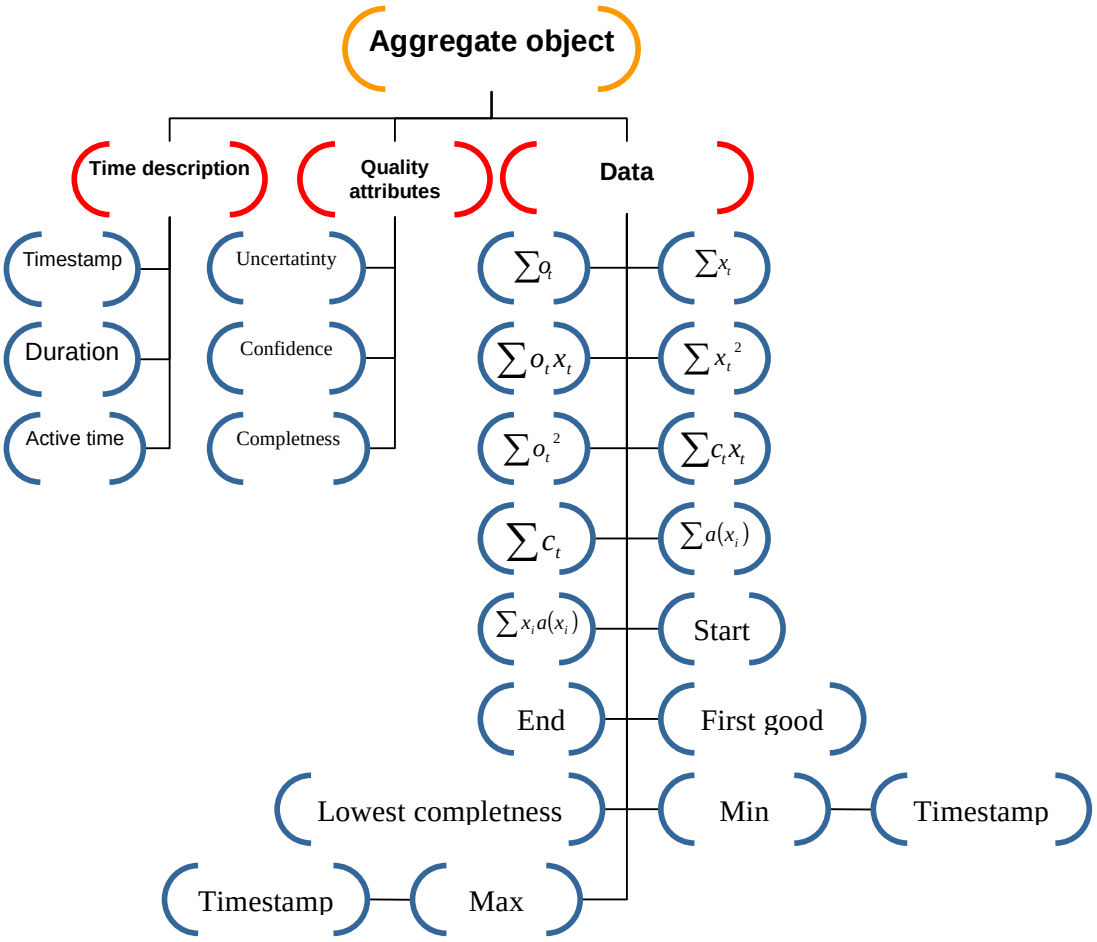

Figure 3 - The aggregate object

The aggregate object and all its attributes are shown in Figure 3. The introduction in section 3.3 mentioned that the usage of a single aggregate object isn't a necessity but might save some storage space and computational time. The attributes assigned to the aggregate object aren't either a fixed concept. Other aggregates requiring other incremental parts are wanted can easily be incorporated into the concept.

4 Uncertainty

This chapter on uncertainty serves three different purposes: to define the perspective on measurement uncertainty of this thesis, to give a brief summary of the requirements on uncertainty evaluation for carbon dioxide emission reports to control bodies and finally to introduce methods that can be used for automatic computerized uncertainty evaluation.

The theoretical description ventilates uncertainty in measured values and doesn't explicitly describe how uncertainty in averages should be treated. But the extension from single values to averages can be deduced from the theory and will be briefly discussed in the section propagating uncertainty of averages.

4.1 Definition of measurement uncertainty

Every measurement of interest is associated with some error. If a measurement is known to be free of error, then the true value of the measured quantity must be known, consequently the measurement doesn't provide us with any new knowledge and is therefore off less interest to us. The sources of error in measurements are normally divided into two categories; bias and uncertainty. Bias are errors spawned from imprecisely determined contextual conditions or imperfect models, leading to systematic errors in the measurements. Uncertainty errors appear at random and have zero mean. (Gleser, 1998)

Other definitions of uncertainty do exist and some might be more conceptually intuitive. Jan Lindskog introduces, in *Mätvärdesbehandling och rapportering av mätvärden*, a definition where the uncertainty is the lack of focus in a measurement (Lindskog, 2006). The ISO guide *International Vocabulary of Basic and General Terms in Metrology (VIM)* considers the uncertainty to be a parameter, associated with the result of a measurement, which characterizes the dispersion of the values that could reasonably be attributed to the measured quantity (cited in NIST, 2000).

The concepts are fundamentally different in that the definitions of VIM and Gleser regard uncertainties as entities with certain characteristics whereas Lindskog's definition is based on the lack of a notion. Nevertheless they all treat uncertainty by the same mathematical methods and those methods are the major concern here.

A widely adopted set of guidelines for evaluating and expressing uncertainty is given by the ISO standard *Guide to the Expression of Uncertainty in Measurement* (GUM) (see Lindskog, Gleser, NIST et al). The approach presented in this and the following section aims to be consistent with the GUM.

4.2 Uncertainty expressed by statistics

Uncertainty in measurements is commonly expressed in statistical terms. A measurement of a PV X can be written as $\hat{\hat{X}} = x + \sum v_i + \sum e_i$ where $\hat{\hat{X}}$ is the true value of X, x the observed value, v_i represents biases and e_i uncertainties. The uncertainties e_i can be expressed in terms of a probability distribution (PD) (Cox & Harris, 2006). One standard deviation of $\sum e_i$ will be denoted the standard uncertainty and $u(X)$ is the notation of the standard uncertainty for an observation of X.

The methods presented below don't treat bias, only how to evaluate the uncertainty. The value of a measurement should be corrected for the bias, to the largest extent possible, before the uncertainty is evaluated. Note that most corrections will themselves contribute to the uncertainty of the measurement.

4.2.1 The two types

Measurement uncertainties are normally categorised into two types, type A and type B. Their classification is not based upon their nature but on how they are derived. The type A uncertainties are derived by statistical analysis of repeated measurements of a stationary quantity. The variation of measurement values around the average will, by the central limit theorem, result in a normal distributed approximation of the uncertainty. (Lindskog, 2006)

The type B uncertainties are uncertainties that are derived by other methods. E.g. these can be uncertainties specified in instrument specifications, manuals, calibration certificates or other publications. But any other method of assessment, for example, a manual assessment conducted by an expert, will fall into this category. (Lindskog, 2006)

The GUM and other publications do often mix their usage of this categorization. They sometimes refer to uncertainties of type A or B, whereas it is not the uncertainty itself but the method used to identify the magnitude of the uncertainty that is of type A or B. Most uncertainties can be identified with the type A method. The reason for this mixed usage is convenience. The type A uncertainties will be described by a normal distribution whereas the type B might be any statistical distribution. It is this distinction that will be of importance when treating uncertainties. (Lindskog, 2006)

4.2.2 Combing uncertainty

The output Y_{sys} for which the uncertainty is desired might be a function of several inputs, where $Y_{sys} = f_{sys}(x_1,...., x_n)$ would be an estimator for the PV Y_{sys} and each x_i is an observation of the corresponding PV X_i. A method is needed to compute a combined standard uncertainty for the output Y_{sys} from the known standard uncertainties of the inputs X_i. In the GUM, if an approximately linear relation exists between the inputs and output, this is achieved by defining the combined standard uncertainty $u(Y_{sys})$ as:

$$u(Y_{sys}) = \sqrt{\sum_{k=1}^{n} (c_k u(x_k))^2 + 2\sum_{i=1}^{n-1} \sum_{j=1+i}^{n} c_i c_j u(x_i) u(x_j) r(x_i, x_j)} \qquad (4.1)$$

where c_i is the ith sensitivity coefficient and $r(x_i, x_j)$ is the correlation coefficient between the ith and jth input (Gregory, Bibbo & Pattison, 2005). The correlation coefficient is defined as

$$r(x_i, x_j) \equiv \frac{u(x_i, x_j)}{u(x_i) u(x_j)} \qquad (4.2)$$

where $u(x_i, x_j)$ is the covariance. As a result of the definition, the correlation coefficients will be 0 for uncorrelated, 1 for fully positively correlated and -1 for fully negatively correlated (anti-correlated) variables.

The sensitivity coefficient c_i is the partial derivate of f with respect to the i[th] input x_i, i.e. $c_i = \frac{\partial f}{\partial x_i}$.

(4.1) can be used to calculate the combined uncertainty when the condition of approximate linearity between the output and the inputs of the function f is fulfilled. If non-linearity is present, then the first order terms of the Taylor series that constitute equation (4.1) won't provide enough detail to describe the relation between the input and output satisfactory. This can be remedied by adding higher order terms of the Taylor expansion. For example, adding the next term to (4.1) would give:

$$u^2(Y_{sys}) = \sum_{k=1}^{n}\left(\frac{\partial f}{\partial x_k}u(x_k)\right)^2 + 2\sum_{i=1}^{n-1}\sum_{j=1+i}^{n}\frac{\partial f}{\partial x_i}\frac{\partial f}{\partial x_j}u(x_i)u(x_j)r(x_i, x_j) + \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\frac{1}{2}\left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)^2 + \frac{\partial f}{\partial x_i}\frac{\partial^3 f}{\partial x_i \partial x_j^2}\right]u^2(x_i)u^2(x_j)r^2(x_i, x_j) \quad (4.3)$$

where the standard uncertainty $u(Y_{sys})$ is the positive square root of $u^2(Y_{sys})$. The GUM can be consulted for more detailed information on when non-linearity must be considered Gregory, Bibbo & Pattison, 2005).

The output distributions u_{sys} are often assumed to have approximate normal distribution (Taylor & Kuyatt, 1994). However if the effective degrees of freedom for the combined standard uncertainty is known then the PD can be treated mathematically like a t-distribution. The effective degrees of freedom can be obtained by the Welch-Satterthwaite formula (Taylor & Kuyatt, 1994):

$$v_{eff} = \frac{u_{sys}^4}{\sum_{i=1}^{n}\left[\frac{(c_i u_i)^4}{v_i}\right]} \quad (4.4)$$

where v_i is the degrees of freedom of i[th] contribution uncertainty and v_{eff} is the effective degrees of freedom of the combined standard uncertainty. The degrees of freedom for type A uncertainties are given by the identification process and the type B has to be approximated. If the type B uncertainties are specified with max and min limits, these are set so the probability that a value would lie outside those limits is very small, then the degrees of freedom can be taken to be infinite (Taylor & Kuyatt, 1994). If this criterion is not met however, then a more elaborate evaluation is needed, a description of such a method is found in the GUM.

To further complicate and show that the use of effective degrees of freedom is an approximate approach, the formula (4.4) gives inconsistent results if any quantities

$x_i, i \in \{1,2.....n\}$ for which u_i is the standard uncertainty, has mutual parts (Willink, 2008). The two quantities x_1 and x_2 have mutual parts if both can be regarded as intermediate quantities defined by their inputs and they have any input in common (Willink, 2008).

Treating uncertainty in the form of standard uncertainty is practical whilst working with the uncertainty but it is not suitable entity for accurate reporting. For a normal distributed uncertainty, the expression $x \pm u(x)$ constitutes a 68% confidence interval, i.e. there is a 68% chance that the true value of X lays within the interval $x \pm u(x)$. But if the uncertainty is represented by a uniform distribution, then the same interval would hold 57.7% confidence. To make the result easier to interpret the GUM defines the expanded uncertainty U such as $x \pm U(x)$ constitutes a 95% confidence interval. The constant k such as $U(x) = ku(x)$ is called the coverage factor.

Determining exact confidence-levels and coverage-factors for combined uncertainties is difficult when the contributing uncertainties originating from different distributions. Especially since the information and knowledge about the contributing uncertainties most often are limited and based upon assumptions. Therefore even a correct analytic solution might be considered flawed. The practical solution to this problem is to identify the situations where the combined uncertainty's distribution function can be approximated by a normal distribution function. (Lindskog, 2006)

The uncertainty distribution function can be approximated by a normal distribution if these criteria are fulfilled:

- A PV Y is based upon a functional relationship with n input PVs X_i and number n is sufficiently large.
- Almost all uncertainties of the inputs are described by reasonably nice distribution functions such as a normal or uniform distribution.
- The standard uncertainties from the type A and the type B evaluations contribute to the combined uncertainty in comparable amounts.
- The uncertainty contributions from the different inputs X_i can be considered independent.
- There are enough effective degrees of freedom ($\nu_{eff} > 20$).

The above defined criteria might seem strict but are actually quite often satisfied. (Lindskog, 2006)

4.3 Uncertainty in CO_2-reporting

This section outlines some of the demands placed by the European Union and the Swedish government on how activities must report their greenhouse gas (GHG) emission and in particular the carbon dioxide (CO_2) emission. These demands shall be regarded as a minimum criterion. Methods for uncertainty calculations, proposed by this thesis, must fulfill this criterion. Drawing requirements from the particular activity, reporting of CO_2 emission, is considered suitable since it's an activity that concerns all Swedish process industries.

The document governing how GHG emissions must be reported within the European Union, is the European Union Commission decision 2007/589/EG of the 18 July 2007. The decision defines how monitoring and reporting of GHG emissions in accordance

with their directive 2003/87/EG of the 13 October that outlines the system for trade with emission licenses.

The European Union Commission decision concerning reporting of GHG emission has, in Sweden, been implemented by Naturvårdsverket. Their result can be found in the regulation NFS 2007:5. According to **15 §** in this regulation should:

All emission and utilization, included in the law ((2004):1 199) concerning the trade of emission allowance, shall be supervised. The supervision shall be concluded by calculations or by continuous measurements in a flue gas channel[4].

But in **24 §** they've added a way out for activities that would have a hard time supervising according to **15 §.** It states that the activity might use an alternative method for the supervision if the method specified in **15 §** isn't *technically possible or involves unreasonable costs.* The alternative method must be applied with at least surveillance level 1 for almost all fuels/materials; the exceptions are fuels/materials that contribute very little to the total emission[5].

A system where a model of the process is used to calculate the carbon dioxide emission would fall under the category alternative methods specified in **24 §.** If an alternative method is used, then a full uncertainty evaluation of the emission value has to be performed and the combined uncertainty has to be within the limits, showed in Table 2, for the corresponding type of activity[6].

Typ of facility	Max uncertainty in the reported yearly CO_2 Emission
Category I	± 7.5 %
Category II	± 5.0 %
Category III	± 2.5 %

Table 2 - max uncertainty in the reported CO_2 emission

4.3.1 The uncertainty evaluation

The first appendix of the NFS 2007:5 explicitly states what the uncertainty evaluation must include. The activities must have knowledge about the uncertainty concerning the equipment utilized to perform the measurements. The assessment of the equipment should consider:

- The uncertainty for all components in the system.
- The uncertainty contributed by the calibration or lack of calibration of the equipment used.
- Possible further uncertainty depending on how the equipment is used in practice.

The uncertainty specified by the equipment's supplier shall be used whenever possible. A complete analysis of the equipment's uncertainty must be performed if no such specification is available. Calculations must in both cases consider necessary

[4] Translated freely by the author
[5] To be more specific, it is those fuels/material that together accounts for 1000 tons or less of a facility's carbon dioxide emission per year that can be excepted. Alternatively a group of fuels/materials that together represents less than 2%, up to a maximum of 20 000 tons per year, of the facility's total emission
[6] For a description of the categories, consult NFS 2007:5, the purpose of the table is to hint the magnitude of allowed uncertainties.

corrections due to effects of the actual utilization of the equipment. These effects can be the result of calibration, the physical environment around the equipment, the age of the equipment or maintenance. (NFS 2007:5)

The reported uncertainty must be expressed as a combined uncertainty for all components in the measurement system. Furthermore, the uncertainty must be expressed as relative uncertainty, a percentage of the reported value at a confidence level of 95%. (NFS 2007:5)

Calculations that combine sources of uncertainty must be performed according to the propagation law specified in several alternative sources, one of the sources being the GUM[7]. The regulation provides an exact interpretation of the propagation law, i.e. specific formulas for the contemplated cases. These are products and sums of mutually uncorrelated uncertainties and their correlated counterparts. The proposed formulas for propagation of correlated sums and products are of a conservative nature, they correspond to the worst case scenario which is fully positively correlated variables. Uncertainties should be regarded as correlated if there is any reason to suspect correlations. For example, measurements that are conducted by the same model of sensors and in the same environment provide reason to suspect correlation. The four formulas are summarized in Table 3 (U_i and U_{total} are the uncertainties expressed as percentages at a 95 percent confidence level).

Independent sums	$U_{total} = \frac{\sqrt{(U_1 x_1)^2 + (U_2 x_2)^2 + ... + (U_n x_n)^2}}{\lvert x_1 + x_2 + ... + x_n \rvert}$
Dependent sums	$U_{total} = \frac{(U_1 x_1) + (U_2 x_2) + ... + (U_n x_n)}{\lvert x_1 + x_2 + ... + x_n \rvert}$
Independent products	$U_{total} = \sqrt{(U_1 x_1)^2 + (U_2 x_2)^2 + ... + (U_n x_n)^2}$
Dependent products	$U_{total} = (U_1 x_1) + (U_2 x_2) + ... + (U_n x_n)$

Table 3 – the formulas for propagation of uncertainties presented in NFS 2007:5

4.3.2 A more elaborate interpretation

The simplified formulas for calculating uncertainty supplied in NFS 2007:5 have both advantages and disadvantages. An obvious advantage is the simplicity of the calculations. Their application becomes especially simple if no considerable modification has been done in the system during the time period for which we are interested in the uncertainty of the measurements. For example, an activity that is about to report of the yearly emission of GHG for a non modified system the uncertainty calculation would consist of solving one equation, the equation being a combination of the formulas in Table 3.

[7] Alt. ISO-5168:2005 Measurement of fluid flow – Procedures for the evaluation of uncertainties
Annex A of *Good Practice Guidance and Uncertainty Management in National Greenhouse Gas Inventories*
annex A of *Revised 1996 IPCC Guidelines for National Greenhouse Gas Inventories*

The major disadvantage of this simplified approach is the penalty the calculation imposes on dependent variables in most cases, especially since the worst case scenario is quite uncommon in reality.

NFS 2007:5 states that the calculations used shouldn't be more detailed than necessary. The directive brings out the conservative propagation laws, but a more detailed approach can be argued necessary in some cases, correlated errors are severely penalized and their tolerance limits are quite low.

NFS 2007:5 refers to several guides when stating that the propagation law should be used to combine uncertainties. But these guidelines are not entirely consistent. For example, the IPCC states implicitly that the uncertainties should be expressed as percentages and thereafter combined as the square root of the sum of squares[8].

A statement that all these guidelines have in common is that they bring out the option to calculate the uncertainty by any method of choice, that being if the method is valid and the reporting activity presents good documentation the methods used. Monte Carlo simulations are promoted as a suitable method by both the IPCC and the GUM. (IPCC 1, 1996)

The sections above constitute some of the arguments for more detailed methods than the propagation law. The propagation law produces an unnecessary large uncertainty in some scenarios, the presentation in NFS 2007:5 is somewhat flawed and the reference cited by the directive all bring out more elaborate methods.

4.4 Propagating the uncertainty of averages

The methods presented in this chapter are normally used to propagate uncertainties of single measurements. Propagating uncertainties in averages is an equivalent operation and the methods can therefore be applied to calculate averages without modification. The only requirement is that the PDs and the corresponding standard uncertainties describe the uncertainty of averages values.

The uncertainty of the aggregation objects is a special case of averaging. Theoretically, if the uncertainty evaluation is completely exhausting and all bias is eliminated by corrections, then the uncertainty of the weighted average could be calculated as:

$$u\left(\overline{X}\right) = \frac{1}{\sum_{i=1}^{n} c_i} \sqrt{\sum_{k=1}^{n} {c_k}^2 u^2\left(X_k\right)} \tag{4.5}$$

where c_i are the normalized weights. A more pragmatic approach would stress that the measurements errors described by uncertainty are likely to be autocorrelated. These autocorrelations have been accounted for in NFS 2007:5, stating that the uncertainties of equivalently measured observations of a PV must be considered fully positively correlated. This corresponds to the formula:

[8] As long as none of the individual uncertainties are greater that 60 percent of the sum of squares of all individual uncertainties.

$$u(\overline{X}) = \frac{\sum_{k=1}^{n} c_k u(X_k)}{\sum_{i=1}^{n} c_i} \tag{4.6}$$

4.5 GUM framework algorithm

The theory presented in the sections 4.2 and 4.4 can be used to construct an algorithm that propagates uncertainty between interconnected components in the system. One such algorithm is outlined in *Propagating uncertainty in instrument systems* by B.D. Hall. The basic algorithm presented in this section is the same as Hall's algorithm and his work should be consulted for a more elaborate presentation.

In section 4.2, a quantity Y_{sys} was expressed as a function $Y_{sys} = f_{sys}(x_1,...., x_n)$ where the inputs also might consist of quantities that might contribute some uncertainty. This implicates the more general representation

$$x_j = f_j(\Lambda_j) \tag{4.7}$$

where Λ_j is a set of input quantities, stating that each quantity can be considered as a composition of a set of simpler functions. This represents a recursive structure where the inputs of the composite quantity x_i also can be stated on the form of equation (4.7). The basic inputs or base cases of the recursion would be represented by the case where Λ_j is an empty set. For example:

$$y_{sys} = f_{sys}(x_{1,.....,}x_6) = \frac{x_1x_2 - x_3x_4}{x_5 + x_6} \tag{4.8}$$

can be rewritten as:

$$y_{sys} = f_{sys}(x_{9,}x_{10}) = \frac{x_{10}}{x_9} \tag{4.9}$$

where

$$x_9 = f_9(x_5, x_6) = x_5 + x_6 \tag{4.10}$$

$$x_{10} = f_{10}(x_7, x_8) = x_7 - x_8 \tag{4.11}$$

and

$$x_7 = f_7(x_1, x_2) = x_1x_2 \tag{4.12}$$

$$x_8 = f_8(x_3, x_4) = x_3x_4 \tag{4.13}$$

The example illustrates how functions of several inputs can be divided into a set of simpler functions. It should be obvious from this example that this division into simpler functions would allow for complicated formulas to be calculated recursively by defining as small set of basic functions.

This notation has a natural extension to handle uncertainty. Let's first define a module. A module m_j would be an entity that consists of an output value x_j and x_j's components of uncertainty. A component of uncertainty is the sum of products between each uncertainty of the input and the components sensitivity to that input.

$$u_i(x_j) = \sum_{x_k \in \Lambda_j} \frac{\partial f_j}{\partial x_k} u_i(x_k) \qquad j = 1,\ldots,m \tag{4.14}$$

These components of uncertainty must remain parameterized on the inputs, i.e. the components of uncertainty of the inputs $u_i(x_k), \quad x_k \in \Lambda_j$ should not be evaluated nor substituted with their numerical values. The index i in $u_i(x_k)$ is meant to signify this distinction between the standard uncertainty of $u(x_k)$ which has an explicit value and the component of uncertainty $u_i(x_k)$ that should be evaluated in regard to its context.

Combining modules into larger ones follow the same scheme as introduced above. The modules being combined into a new module are to be considered as inputs and the new module is the output-variable. It's to provide this opportunity that the components of uncertainty must remain parameterized at each module and thereby account for all correlation when evaluating the uncertainty for the final output. Evaluating the uncertainty for this final output x_j consists of combining all the components of uncertainty at the module m_j as in the formula (4.1).

The discussion of linearity and the addition of higher order terms of the Taylor series apply here as well. In this particular presentation this would consist of adding higher order terms of the Taylor series to (4.14).

4.6 Monte Carlo

Another approach to calculating the uncertainty in measurements is by Monte Carlo simulations. The Monte Carlo methods can solve many of the problems where the assumptions of the GUM framework do not apply. It's also recommended as a tool for validating the calculations made according to the GUM framework (Cox & Harris, 2003). Another advantage is that the results of a Monte Carlo simulation provide more information than that of a GUM framework calculation. The Monte Carlo simulation provides a discrete approximation of the probability density function (PDF) whilst the GUM framework calculation provides the standard deviation of some unknown PDF.

This increased amount of information can be used for lots of different purposes, some aspects that are interesting to this work are: an exact coverage factor can be calculated for symmetric PDFs, a confidence interval for asymmetric ditto and information is gained on whether the output PDF has become skewed or biased due to correlations (Papadopoulos 2001).

This increased amount of information does of course come with a price. The convergence rate of Monte Carlo simulation is slow, $n^{-0.5}$ to be precise (L'Ecuyer, 2003). This has shown to be problematic for high dimensional problems since both the generation of pseudo-random numbers and solving the equation consumes computation time (L'Ecuyer, 2003).

4.6.1 MC theoretical description

A Monte Carlo (MC) method for evaluating the uncertainty of a quantity $Y = f(X)$ where X is a vector of input variables $X = (X_1, X_2, ..., X_N)$ consist of repeating the two steps; adding random error terms to observed values of the input $x = (x_1, x_2, ..., x_N)$ and thereafter calculating the randomized output y from the randomized observations. If the random error terms are drawn from the PDFs $g_{X_i}(\xi_i) \quad \forall i \in \{1,2,....,N\}$ of the inputs then the randomized outputs y will be observations from the PDF $g_Y(\eta)$ corresponding to observations from the uncertainty of y's PDF with the expectation-value shifted by the un-randomized output y. As the number of iterations of the two steps in the procedure goes to infinity, the randomized outputs converge to a discrete representation of $g_y(\eta)$.

Interesting characteristics of the Monte Carlo method is that the function f might be any mathematical function and that the PDFs $g_{X_i}(\xi_i)$ might be any PDFs, the inputs can be correlated and can even originate from the same multivariate PDF.

The number M of simulations carried out can be fixed, decided before the start of the iteration, or halted by some stop condition for when enough accuracy has been achieved. Using a stop condition might be the more intuitive approach in uncertainty evaluation. This stop condition can be based upon: the expectation value, standard uncertainty or limits of the confidence interval. These are all parameters of interest and are therefore suitable candidates for the stop criteria.

An example of an adaptive method using stop conditions: if the values $y^{(h)}$, $u(y^{(h)})$, $y_{low}^{(h)}$, $y_{high}^{(h)}$ are calculated for each N^{th} iteration, the index h signifies the number of N iterations that has been performed. The $y^{(h)}$ is the approximated expectation-value of the last N iterations, $u(y^{(h)})$ is the approximation of the standard uncertainty, $y_{low}^{(h)}$ the lower and $y_{high}^{(h)}$ the higher limit of the approximated 95% confidence-interval. After each N iterations (apart from the first), the arithmetic mean of the intermediate values $y^{(h)}$ are calculated. The standard deviation s_y associated with the arithmetic mean is formed as a measure of degree to which the calculation has stabilized. The counterparts of the arithmetic mean and standard deviation are determined for $u(y^{(h)})$, $y_{low}^{(h)}$ and $y_{high}^{(h)}$. Then if the largest of $2s_y$, $2s_{u(y)}$, $2s_{ylow}$ and $2s_{yhigh}$ is smaller than the degree of accuracy needed in u(y) the calculation can be regarded as stabilized. (Cox & Harris, 2006)

The algorithm used for the MC simulations in this work:

While (The accuracy is not achieved or N <2){
Repeat M times{
Generate the vector x^j from the inputs and sampling from the PDFs of the inputs
Evaluate the model $y^{NM+j} = f(x^j)$

}

Calculate the estimate of the output for the M iterations, $\hat{y}^M = \frac{1}{M-1}\sum_{j=1}^{M} y^j$

Calculate the associated standard deviation $s_y^M = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}\left(y^j - \hat{y}\right)^2}$

Update the overall $\hat{y}$ **and** s_y **by weighting in the values of** $\hat{y}^M$ **and** s_y^M

Increment N

}

Sort the values $\mathbf{y^j}$**,** $j \in \{1,2,.....,NM\}$**, in non-decreasing order, the sorted values then represent a discrete representation of the approximation of the output PDF.**

The above algorithm outlines one particular MC implementation. The details can differ a lot between implementations.

4.7 Quasi Monte Carlo

As previously mentioned an issue with MC simulations is the slow convergence rate. This property has lead to the development of quasi-MC (QMC) methods. In QMC the pseudo-random numbers used in MC has been replaced by quasi-random numbers. The notation quasi-random numbers is somewhat misleading as they are normally deterministically distributed point sets and not random at all but. If strictly quasi-random numbers are used in a simulation it might result in a biased result. This can be prevented by introducing random shifts between repeated simulations with the same deterministically distributed point set.

4.7.1 Low discrepancy sequences

A design criterion of pseudo-random numbers is that they are uncorrelated, leading to uncorrelated dimensions in the noise sets. This often results in overlapping and clustering of data points, leading to slow exploration of the space (L'Ecuyer, 2006). A more efficient approach is to explore the space deterministically. The gain in efficiency does come with a cost, as a deterministic exploration of the space defies the founding assumptions of the MC methods since it is associated with a deterministic error. To acquire some of the efficiency of deterministic distribution but to keep the required randomness, predetermined sequences are used to distribute some points whereupon the sequences receive a random shift on each dimension to form the remaining points.

The most basic implementation can be described as follows: the predetermined sequences are used to distribute N points in a unit-hypercube of same dimension as the input dataset. The model is then evaluated for those N points. The points correspond to the random errors of the models inputs and since these might not be uniformly distributed over the interval $[0,1)$, the inverse of their PDFs must be used to calculate the sought value. Afterwards, another point from the same unit-hypercube is drawn randomly. The value of this point is added module 1, coordinate by coordinate, to each of the N points to construct the next N points that can be used to evaluate the model. Since the shift is random in each dimension, the N original points and the N points generated will be uncorrelated and the method has therefore revived the randomness that guarantees convergence to a non-biased value.

4.7.2 Niederreiter Sequences

As mentioned above QMC methods use predetermined sequences, more specifically low-discrepancy sequences. The most commonly used low-discrepancy sequences are the (t,s)-sequences, where s is the dimensionality and t can be regarded as a quality measure of the sequence (Niederreiter, 1988). The sequence to be used here is a Niederreiter sequence in base 2, chosen partly for practical reasons. The Niederreiter sequence in base 2 have a better t-value than the commonly cited Halton, Faure and Sobol sequences (Niederreiter, 1988). But the Niederreiter-Xing sequences often improves on the t-value of the Niederreiter sequence in base 2 (Dick & Niederreiter, 2008).

Does the use of the low discrepancy sequences improve the convergence rate of the standard MC method with pseudo-random numbers? A theoretical bound for the convergence rate can be derived from the koksma-hlawka inequality[9]. It has been shown that it's possible to create (t,s)-sequences with a discrepancy $D(P_n)$ (and also convergence rate) of

$$\mathrm{O}(D(P_n)) = \mathrm{O}\left(\frac{\log^d n}{n}\right) \tag{4.15}$$

where P_n is a sequence of length n and dimension d (Niederreiter, 1988). Many common sequences such as Halton, Faure, Sobol and Niederreiter do converge at this rate. Their convergence rates differ by a constant and the sequences are presented in an order of decreasing size for this constant (Niederreiter, 1988). These convergence rates beats the $\mathrm{O}(n^{-1/2})$ of standard MC methods asymptotically (L'Ecuyer, 2006). But the dimensionality has a large impact on the convergence rate of QMC-methods for more practical values of N. If $N \leq 10^9$ then the theoretical convergence rate of QMC beats that of MC only for low dimensionalities, the breakpoint is about 7 or 8 (L'Ecuyer, 2006).

Fortunately, the theoretical bound on the convergence rates doesn't fully reflect the practical aspects of the methods. QMC has successfully been shown to beat MC for problems with over 1000 dimensions (L'Ecuyer, 2006). This can to a large extent be explained by the notion of effective dimensions. Effective dimension can in general terms be described as the number of dimensions needed to express the function f. More distinct definitions are based on the sense the function f can be said to have effective dimension s. For example, f is said to have effective dimension s in the superposition sense if f is well approximated by a sum of s dimensional functions (Owen, 2002). Put in more common terms; if f is a function of n variables that can be well approximated

[9] Let f be the function from the inputs to the output, μ be the expectation-value of output and Q_n the Monte Carlo estimator for μ. Then consider the Banach space F of functions, where $\|f - \mu\|$ measures the variability of f and $D(P_n)$ is a measure of the discrepancy of the point set P_n. If $D(P_n)$ is chosen so that $|Q_n - \mu| \leq \|f - \mu\| D(P_n)$ holds for all $f \in F$, then the error will converge at least as fast as $D(P_n)$ if the variability of f is bounded (L'Ecuyer, 2006).

by a function of $n/2$ variables, then f is said to have an effective dimension. For more strict definitions and descriptions see the references.

5 Using a process model to improve/replace variables

If multiple measurements are available for a PV they can be combined to improve the accuracy of the estimate and thereby lower the uncertainty (Kessel, Berglund & Wellum, 2008). In this system, multiple measurements would consist of different ways of evaluating the PV. For example, the value of a PV A might be measured directly by a sensor and calculated from sensors measuring other PVs. Calculation from sensors measuring other PVs is possible in the presence of a model that specifies their physical relationship with A. Two issues concerning the use of multiple measurements will be addressed below. Firstly, under what conditions can two or more approximate value sources of a PV be combined to improve accuracy? Secondly, how should the values be combined to provide the lowest uncertainty?

5.1 Consistency

Averaging of several complete measurements of a PV can be used to achieve a better accuracy. The term complete measurement aims at distinguishing these replicated measurements consisting of evaluations for both value and uncertainty from the type A evaluations that can be used to evaluate the uncertainty for their average (Kessel, Berglund & Wellum, 2008).

This can be illustrated by an example using the GUM approach to uncertainty described above. Let Y be the arithmetic average of the two normally distributed variables X_1 and X_2, then the expanded uncertainty of Y, according to (4.1), will be:

$$U(Y) = k\sqrt{\frac{u^2(X_1)}{4} + \frac{u^2(X_2)}{4} + \frac{u(X_1, X_2)}{2}} \tag{5.1}$$

where k is the coverage factor. If k is chosen as 2 then $Y \pm U(Y)$ is a confidence interval at approximately 95 percent confidence level. If x_1=100 and x_2=80 are observations of X_1 and X_2 with $u(x_1)=u(x_2)=1$, then $y = \frac{1}{2}x_1 + \frac{1}{2}x_2 = 90$ constitutes an observation of Y with the expanded uncertainty:

$$U(y) = 2\sqrt{\frac{u^2(x_1)}{4} + \frac{u^2(x_2)}{4} + \frac{u(x_1, x_2)}{2}} = \sqrt{2} \tag{5.2}$$

for the observations x_1, x_2 with a coverage factor k=2. The large differences between the measurements in the example, together with their relatively small uncertainties raise the question whether they can be considered to be good approximations of the same physical quantity. If the probability is 0.95 that the true values of X_1,X_2 and Y lie within their respective confidence interval, shouldn't these intervals be almost the same or at least overlap?

The scenario in the example wouldn't occur if all measurements had been conducted successfully, all the uncertainties associates with the measurements had been identified

and all calculations on the results of the measurements were based on exact relationships.

But a system, built after the ideas presented here, includes lots of assumptions that may be invalid. The concept of consistent measurements will therefore be used to test these assumptions and improve the accuracy in values of the PVs.

Measurements are said to be inconsistent if the results of multiple replicates of the same measurement are not equivalent. One way of verifying the consistency is therefore to perform an equivalence test. The test is to check, for each Y_i in the weighted sum:

$$\bar{Y} = \sum_{i=1}^{N} c_i Y_i \tag{5.3}$$

that the relationship $\left|Y_i - \bar{Y}\right| \leq k \cdot u\left(Y_i - \bar{Y}\right)$ hold. A coverage factor $k = 2$ corresponds approximately to a 95% confidence level if all sources of uncertainty are normal distributed. (Kessel, Berglund & Wellum, 2008)

The exact value of the coverage factor for different confidence levels probably won't be known unless all input uncertainties are normal distributed. But the coverage factor of the normal distribution can be used as an approximate value of the coverage factor for most combined uncertainties (Kessel, Berglund & Wellum, 2008). However, if the approximate value of the coverage factor is used without regard to the specific inputs, it lowers the quality of the test since it changes the corresponding confidence level. This approximate approach of handling the coverage factor will be used in lack of better options.

If the consistency check fails then the measured/calculated values cannot be combined to produce a more reliable value for the process variable (Kessel, Berglund & Wellum, 2008). Two options are available to produce a reliable value; removing values from the combination according to some reliability-criteria or by manipulating the values into being consistent.

The later alternative would consist of penalizing the uncertainty of the measurements. One way of achieving this is to add small terms of zero mean noise with equal variance to each measurement. Thereafter check for consistency again and if the measurements still are not consistent, then the procedure can be repeated until they are. (Kessel, Berglund & Wellum, 2008)

5.2 Weighting

If different complete measurements can be combined to lower the uncertainty in the approximation of a PV, then how should these measurements be combined to produce the best possible result?

Fusion methods, methods for combining sources to achieve higher performance than of the individual sources have been studied for centuries. Obviously the problem is easily solved if all error distributions and their correlations are known, but this is often not the case. Early fusion rules assumed independence between the sensors errors whereas the problem becomes simplified. Other fusion rules not depending on this assumption have

been developed. For example, Rao provides in *generic sensor fusion problem: classification and function estimation* a method, not based upon the assumption of independence, but using data from several samples. (Rao, 2004)

The problem of combining different measurements to produce the most accurate value of a PV corresponds to find the best weights c_i for the weighted sum of N variables:

$$Y = \sum_{i=1}^{N} c_i X_i \tag{5.3}$$

where

$$\sum_{i=1}^{N} c_i = 1 \tag{5.4}$$

The optimal weights would normally be the ones that minimize *u(Y)* and thereby produce the most accurate value of *Y*. But the problem is harder in this case since the values of the process variables are required to be consistent, the full optimization problem is to find, if they exist, the weights that minimizes the objective function in (5.5).

$$\begin{cases} Min\ u(Y) \\ Y = \sum_{i=1}^{N} c_i X_i \\ 0 \le c_i \le 1 \quad \forall i \in \langle 1,2....,N \rangle \\ \sum_{i=1}^{N} c_i = 1 \\ |X_i - Y| \le 2 \cdot u(X_i - Y) \end{cases} \tag{5.5}$$

Solving (5.5) for each process variable at each time interval would be too computationally heavy; it's a nonlinear optimization problem that can include a large number of variables.

An alternative approach, promoted by Lindskog (2006), would be to weight according to the uncertainties assuming mutual independence. This method might not produce the optimal weights but allow for the weights to be produced with only few elementary arithmetic operations per component of uncertainty. The standard uncertainty for the output *Y*, calculated according to (4.1) with zero correlation, is:

$$u^2(Y) = \sum_{i=1}^{N} \left(\frac{\partial f}{\partial X_i} \right)^2 u^2(X_i) \tag{5.6}$$

for inputs X_i with uncorrelated uncertainties. If the output Y is a sum weighted by a normalized set of weights c_i, such as:

$$Y = \sum_{i=1}^{N} c_i X_i \tag{5.7}$$

then

$$c_i = \frac{u^2(X_i)}{u^2(Y)} \tag{5.8}$$

are the normalized weights that minimize the uncertainty in Y (Lindskog, 2006). But for inputs X_i that are to be combined into a process variable there is also the notion of consistency. The weights can only be used if the output produced is consistent with the inputs. This will be handled by checking the consistency and if the test fails, then throw away the most uncertain input and thereafter recalculate weights and redo the consistency check. A small test and illustration of the consistency check and weighting is presented in section 7.2.

6 Test case and Implementation

6.1 The java implementation

A small java based application was created to illustrate the definitions above and to evaluate some of the proposed methods. The time span of this project didn't allow for the development of a full blown system based upon the ideas presented above. A test application that allows for an assessment of the uncertainty evaluation methods and combination of consistent measurements seemed suitable.

6.1.1 Basic description

The input of the application is limited compared to a full blown system; it takes a XML-document as input rather that continuously produced unit time averages. The XML-document must specify an expression consisting of a combination of any number of inputs and the values for the inputs. The combination must consist of the basic operations specified in the section with the same name. These inputs are converted a tree structure that corresponds to the expression, nothing is stored and all test and evaluation are performed on the tree structure.

A large part of the application consists of an object called Variable and its subobjects. These are used to construct the tree structures mentioned above, thus acting as both memory and supplier of rules for how the structure is evaluated. The object hierarchy is shown in Figure 4.

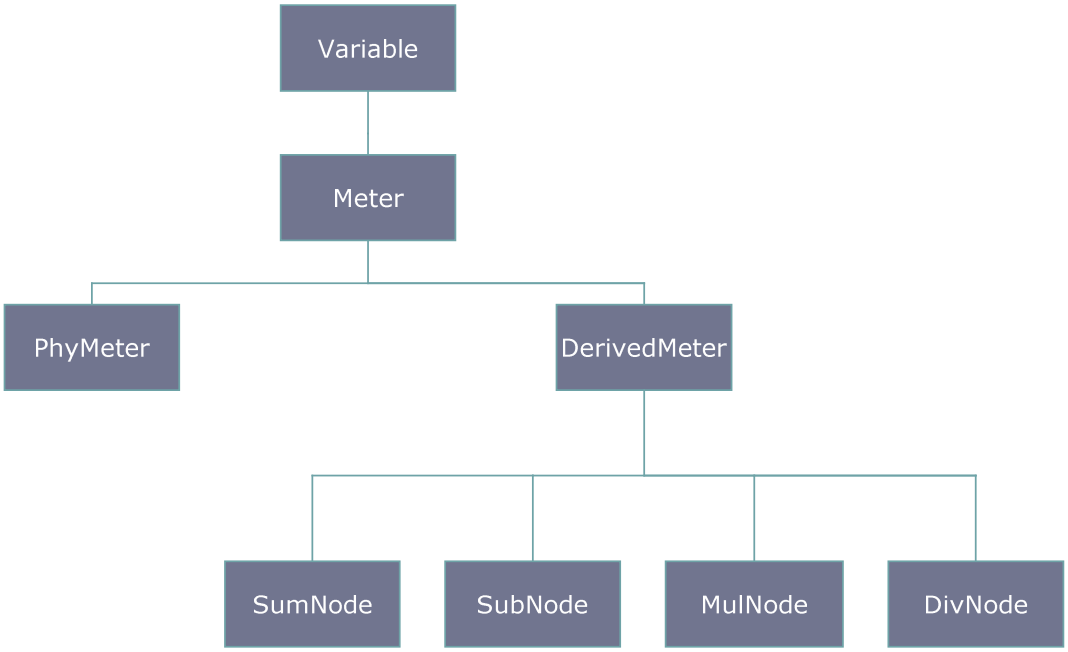

Figure 4 - the object hierarchy

6.1.2 The Monte Carlo implementation

This section aims to clarify the implementation of the MC methods, to show how the results in the uncertainty evaluation below were obtained. The evaluation of the MC

methods uses the tree structure but the methods themselves are implemented in the class Monte Carlo that can be found in appendix C. The code is included to represent the exact structure of the algorithms and won't be explained further. Three important details in the implementation that aren't explained by the code is the pseudo-random number generator, the Niederreiter sequence generation and the random multivariate normal distribution generator. They are described in the subsequent sections.

6.1.2.1 Random number generator

The generation of pseudo random numbers is a crucial part of MC-simulations and there are differences between different generators. Many of the commonly used random number generators fail to show satisfying randomness.

Mersenne Twister algorithm was used for the generation of pseudo random numbers. This particular implementation was provided as a part of the Math-uncommons library (Uncommons Maths).

The generator has a prime period of 2^{19937}-1 and occupies a working area of 624 words (As a comparison, the rand-function has a period of ~2^{31} and use a working area of 1 word)[10]. It generates numbers as fast as most modern generators and is almost as fast as the rand-generator included in java. It has also shown promising statistical properties and passed several tests including Marsaglias diehard test. (Matsumoto & Nishimura, 1998)

Tests were performed to ensure that the large working area of the Mersenne Twister didn't affect the computational time in this particular implementation. The tests concluded that the time consumed by the Mersenne Twister generator was approximately equal to the time needed by rand to generate the same amount of numbers, regardless of the magnitude of that amount. Nor did the larger working space affect the computational time needed to perform MC simulations.

6.1.2.3 Generation of Niederreiter sequences

This functionality is supplied by a library called SSJ developed at the Département d'Informatique et de Recherche Opérationnelle, at the Université de Montréal (designed and supervised by Pierre L'Ecuyer). The Niederreiter sequence in base 2 is implemented in a class named *NiedSequenceBase2*. The class allows the parameters of the sequence to be specified; the number of points in the sequence, the number of output digits and the dimension. (SSJ API Specification)

6.1.2.2 Random Multivariate normal distribution generator

Another difficult part of the Monte Carlo is the extension from pseudo random number generation to generation of correlated pseudo random numbers. This functionality is also supplied by the SSJ library. The class used in this implementation uses a method based on Cholesky decomposition of the covariance matrix. The decomposition is defined as $\boldsymbol{\Sigma} = \mathbf{A}\mathbf{A}^t$ where $\boldsymbol{\Sigma}$ is the covariance matrix and $\mathbf{A}$ is a lower-triangular matrix. Observations of the multivariate normal distribution defined by $\boldsymbol{\Sigma}$ and the expectation vector $\boldsymbol{\mu}$ can be calculated as $\mathbf{X} = \boldsymbol{\mu} + \mathbf{A}\mathbf{Z}$, where $\mathbf{Z}$ is a vector of independent standard normal distributed variables. The class uses an external source for the generation of the

[10] The working area can be regarded as a non-specified type of memory and a word is a number of bytes that a processor considers as a unit of data. Hence, the normal wordsize today would be 32 bits.

vector **Z**, the methods for number generation presented in the section above are used for this purpose. (SSJ API Specification)

6.2 The test cases

Three different cases were used to verify the implementation and to test the proposed methods. They serve different purposes and have therefore been selected by different criterions. The first denoted M1 is a small fictional model which is simple enough for the value and uncertainty to be verified manually. It also serves as a contrast to the much larger model M2 in the evaluation of methods to calculate the uncertainties in the values.

These models can be illustrated as trees where each node is an object from the hierarchy presented in Figure 4. M1 is illustrated in Figure 5, the left tree represents the structure of the model and the right illustrates the values at every node after evaluation. The inputs numerical values presented in the figure are used for every test that involves M1.

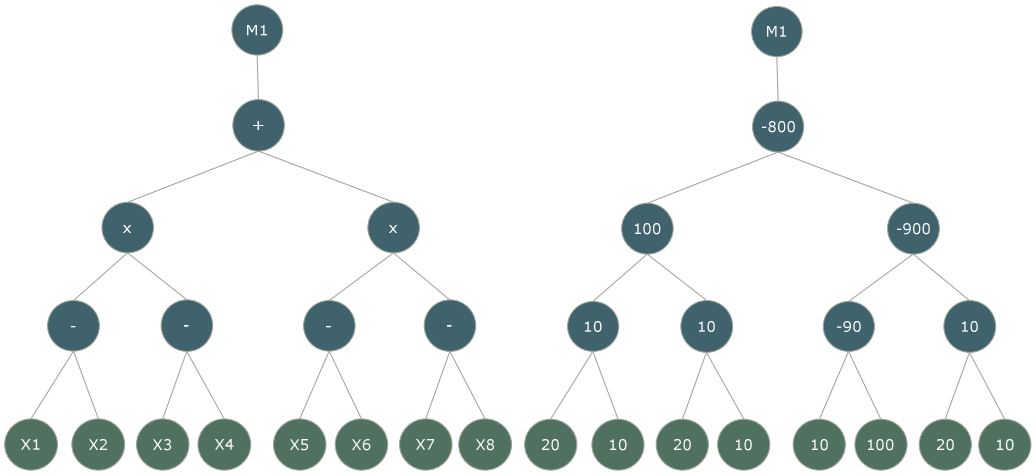

Figure 5 - Illustrations of M1

The second case M2 is a model of a real Swedish process industry's CO_2 emission into the atmosphere. It serves the purpose of testing how the method for propagating measurement uncertainty handles models that include a large number of inputs and operations. Table 4 shows the exact number of input and operations in M2. This is presented as a table instead of a tree structure simply because the model is too large. The number nodes can be regarded as an approximate of the trees size; the number of nodes are equal to the sum of all occurrences in Table 4, except the row *PVs* corresponding to the dimensionality of the model. The data used for the evaluation of this model is average values from one hour of production, the specific hour was chosen at random.

Node type	Nr. of occurrences
Constants	184
PVs	102
Occurrences of PVs	302
Sum	249
Sub	10
Mul	240
Div	41

Table 4 - Occurrences of node types in M2

The last test case M3 is a small model that is included to illustrate combination of consistent measured values. The specific model used here was not chosen on any specific criteria except that it was two small models of the same physical quantity. The values used in the evaluation are the 24 hourly averages that constitute a day of production. M3 is illustrated in Figure 6 below.

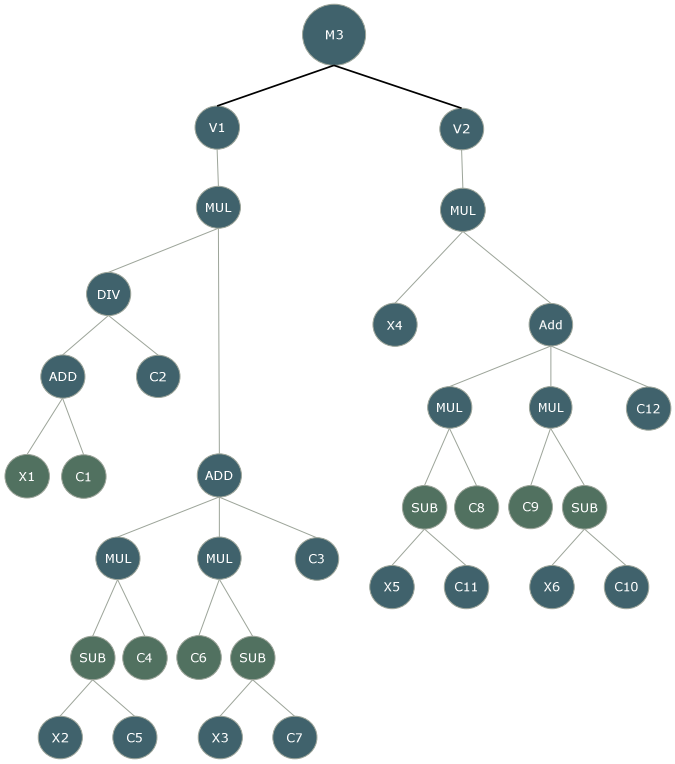

Figure 6 - illustration of M3

7 Results

7.1 Uncertainty

The three different methods for automatic evaluation of measurement uncertainty were tested with the two test models M1 and M2. The test serves two purposes in the selection process of suitable method to handle uncertainty. The first purpose is to verify that the methods provide reasonable and corresponding answers in, at least, these particular cases. The second is to provide a hint on how fast the different methods are and if they might be fast enough to be used for this purpose.

The stop condition used in the MC algorithms is based only on the relative improvement in the value and standard deviation over a series of iterations. Note that other stop conditions have been proposed and the conditions presented in the theoretical section about uncertainty also incorporate the standard uncertainty, upper and lower limits of the confidence interval. A rather large number of iterations are run before and between each time the stop condition is checked. This is supposed to reinforce the credibility of the stop condition by reducing the probability that the method stops after only a few "lucky" numbers. The stop condition is verified after each multiple of 1000 iterations for the standard MC method, 8192 for the simulation of M2 with QMC method and 1024 for M1s QMC simulation. There is also a limit to the maximum number of iterations before the simulation is stopped and considered nonconvergent. This limit was set to 100000 for the standard MC and the multiple below 100000 for the QMC simulations. Longer runs will show to serve little purpose since the computational time would be too long and this particular implementation and the test computer limits the number of data points to slightly above 100 000 due to the restricted amount of available memory. The relative improvement in the value and standard deviation is calculated as the difference between value/standard deviation calculated using all iterations performed and the value/standard deviation calculated at the last verification of the stop criteria (hence the improvement gain using another n iterations, n being 1000, 8192 or 1024 as described above). This improvement will be referred to as accuracy in the results below, thus the accuracy should be interpreted as a relative rather than absolute measure, describing how far the computation has converged. A statement of the accuracy in absolute terms and more elaborate stop conditions is desirable for real applications of MC methods but serves little purpose for these tests.

An uncertainty was assigned to all PV that corresponds to basic measurements in the models (denoted inputs in section 5.2). These uncertainties were normal distributed and with a standard deviation set to one percent of the respective measurements value. Strictly normal distributed uncertainties were used for no other reason than to facilitate the implementation of these tests. An extension of the application to include uncertainties identified by the type B method shouldn't change the results drastically and wouldn't pose any problems to implement[11]. It would be possible to extend the application to handle uncertainties described by any continuous probability distribution (Cox & Harris, 2006). The covariance between the different uncertainties was specified by the correlation matrix in Figure 7. Additional results for M2 were produced for

[11] Since the implementation is based upon the generation of pseudo- and quasi-random numbers where upon the inverse of the probability density function is used to calculate the normal distributed numbers. So all that is needed to make the algorithms support other probability distributions is to implement the inverse of their probability density functions.

completely uncorrelated uncertainties, e.g. a covariance matrix with the diagonal consisting of the uncertainties variances and all other elements set to zero. Most of the results were produced with correlated inputs since it constitutes a more interesting case where the output PDF might become asymmetric and biased.

$$\begin{bmatrix} 1 & 0.5 & 0 & \cdots & \cdots & \cdots & 0 \\ 0.5 & 1 & 0.5 & \ddots & & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & 0.5 & 1 & 0.5 \\ 0 & \cdots & \cdots & \cdots & 0 & 0.5 & 1 \end{bmatrix}$$

Figure 7 – the correlation matrix

7.1.1 Time and uncertainty for M1

Results of the GUM algorithm for M1	
Calculated value	-800
Standard uncertainty	9.55249
Time to evaluate the uncertainty	10 (ms)

Table 5- results of the GUM algorithm for M1

Tables 5-7 show that the uncertainties calculated by the three methods were about equal[12] and this result was concluded by a manual evaluation. Their estimation of the value followed the same pattern. The largest distinction is a slight difference found in the third significant digits of the standard uncertainties. Errors of this magnitude can be expected whit the stop conditions used and would be insignificant in practical applications since measurement uncertainty is an approximate concept.

Table 7 shows that the standard MC method converges for the lowest accuracy, converges sometimes for the mean but not for the highest. It also shows expected characteristics over most of the properties. The convergence rate cannot be fully evaluated due to the minimum and maximum constraint on the number of iterations.

The theoretical convergence rate is $n^{-½}$, implying that a 100 times more iterations are needed to improve the accuracy by one decimal. The results in Table 7 don't fully comply with the theory. The difference between a relative accuracy of four and five decimals is only 3-4 times the number of iterations. The highest accuracy doesn't provide any further information on this clash between theory and practice since it didn't converge within the maximum number of iterations.

The result of the QMC simulations doesn't follow the same pattern as those of standard MC. The simulations converged within the maximum number of iterations every single run for the two lower requirements on the accuracy and sometimes for the highest. Table 6 show that the average number of iterations needed to converge was slightly more than doubled from the lowest to the middle accuracy. No such relationship can be

[12] All values are rounded to five decimals

deduced between the middle and highest accuracy due to the non-convergence of some of the trials with the highest accuracy.

The difference in computational time between the two methods was negligible. Both methods completed the simulations for the lowest accuracies in approximately one fourth of a second and the highest in one second (though some runs stopped at the maximum number of iterations). The GUM framework algorithm was much faster than the two MC methods; the computational time required was only 10 ms.

M1 computed with QMC			
Required accuracy	10^-6	10^-5	10^-4
Converged	Sometimes	Yes	Yes
Value	-799.99997	-799.99637	-799.99622
Standard uncertainty	9.55276	9.55274	9.55363
Coverage factor	1.959275	1.96228	1.96187
Skewness	-0.13437	-0.123097	-0.11723
Value error	2.06157E-5	0.00362	0.00378
Time elapsed while iterating	736.1 (ms)	489.4 (ms)	199.8 (ms)
Time elapsed while sorting	110.0 (ms)	63.0 (ms)	19.0 (ms)
Number of iterations	79462	52428	21299

Table 6 – Results of QMC for M1

The numerical values for the coverage factor found in Tables 6 and 7 correspond well to the coverage factor of a normal distributed variable. However, the numerical value of the coverage factor is calculated under the assumption that the PDF of the uncertainty is symmetric. Some information about the symmetry of the distribution can be deduced from the skewness, the calculated skewness coefficients for these simulations are all between 0.10-0.15. An approximate test for symmetry is to compare the skewness to the standard error of skewness[13] (SES), skewness greater than two times the SES indicates asymmetry.

M1 computed with MC			
Required accuracy	10^-6	10^-5	10^-4
Converged	No	Sometimes	Yes
Value	-799.99724	-799.98250	-800.00905
standard deviation	9.54547	9.56298	9.57205
Coverage factor	1.95997	1.95948	1.96333
Skewness	-0.14689	-0.12817	-0.14627
Value error	0.00276	0.01749	-0.0090
Time elapsed while iterating	1006.0 (ms)	727.4 (ms)	201.7 (ms)
Time elapsed while sorting	115.8 (ms)	82.7 (ms)	16.0 (ms)
Number of iterations	100000	72900	19700

Table 7 - Results of MC for M1

[13] Skewness coefficients can be considered significant if their absolute values are greater or equal to 2 times the standard errors for skewness (Tabachnick & Fidell, 1996). The standard error for skewness (SES) can be calculated as $SES = (6/n)^{1/2}$

The greatest SES for these simulations is 0.017 and all the uncertainty PDFs should therefore be considered asymmetric. But how does this asymmetry affect the results? The Monte Carlo methods allow numerical computation of confidence interval and these proved to be fairly symmetric in this particular case.

7.1.2 Time and uncertainty for M2

Results of the GUM algorithm for M2	
Calculated Value	196.77797
Standard uncertainty	1.679746
Time to evaluate uncertainty	50 (ms)

Table 8 – Results of the GUM algorithm for M2

The results of the MC simulations of M2 are presented in the Tables 9 and 10. The three levels of accuracy required were lowered by a digit due to increased dimensionality. Not much can be deduced from the numerical value of the parameters; value, standard deviation and coverage factor. There is slight difference on the third decimal, as in the simulations of M1.

The iterations consumed by standard MC to simulate M2 were fairly consistent with results received while applying the same method to M1. Three times the iterations were required to improve the accuracy by one decimal. The results of QMC for M2 presented here were gained using a longer sequence than in the result on M1 above. This is simply because the longer sequence generated much better results in terms of consistency between runs. QMC succeeded to converge some of the 10 trials for the lowest and middle accuracy.

The successful trials of the standard MC method still show a large weakness: the computational time. About 5, 13 and 42 seconds, respectively were needed to arrive at the specified levels of accuracy. The GUM algorithm completed the task in 50 ms which is rather significantly lower than 42 s.

The skewness of the uncertainty PDF is smaller than M1's equivalent. A quick verification of the SES provides varying results. The SES for the simulation with lowest accuracy is about 0.028 and the PDF could therefore be considered symmetric. The simulations requiring the highest accuracy had a SES about 0.01 implying that the PDF is asymmetric. The numerically calculated confidence intervals, as in the simulations of M1, were practically symmetric.

Results of MC for M2			
Required Accuracy	10^-5	10^-4	10^-3
Converged	Yes	Yes	Yes
Value	196.77585	196.77009	196.77553
Standard deviation	1.67773	1.68042	1.68222
Coverage factor	1.95951	1.95818	1.95990
Skewness	0.03224	0.05163	0.03921
value error	-0.00211	-0.00787	-0.00243
Time elapsed while iterating	42142.3 (ms)	12885.6 (ms)	4859.8 (ms)
Time elapsed while sorting	140.0 (ms)	20.1 (ms)	5.0 (ms)
Number of iterations	69300	21900	8000

Table 9 - Results of MC for M2

Results of QMC for M2			
Required Accuracy	10^-5	10^-4	10^-3
Converged	No	rarely	Rarely
Value	196.77409	196.77421	196.77422
standard deviation	1.66945	1.67440	1.67667
Coverage factor	1.95966	1.96095	1.96245
Skewness	0.04029	0.04568	0.04064
value error	-0.00387	-0.00377	-0.00367
Time elapsed while iterating	58438.3 (ms)	54367.3 (ms)	53952.6 (ms)
Time elapsed while sorting	144.6 (ms)	126.6 (ms)	120.0 (ms)
Number of iterations	98304	95027	91750

Table 10 - Result of QMC for M2

Table 11 presents simulations of M2 whereas the uncertainties in the inputs were considered completely uncorrelated (i.e. the correlation matrix was set to the identity matrix with a dimensionality equal to the number of basic inputs). These results are provided to show that the number of iterations needed and computational time consumed remain approximately the same for uncorrelated input uncertainties. The results show nothing remarkable, the skewness present can be explained by several of the inputs reoccurring in the model. The approximate equality in computational time consumed using correlated and uncorrelated inputs provides reason to believe that the random multivariate normal distribution generator used can operate approximately at the same speed regardless of the provided correlation matrix structure. This is important since profiling[14] reveal that over 90 percent of the computational time consumed by simulations was spent by the multivariate normal distribution generator. A comparison between the simulations with correlated and uncorrelated input does also indicate that the method used to correlate the inputs didn't corrupt the discrepancy of the Niederreiter sequences.

[14] Conducted with the profiler incorporated in NetBeans IDE 6.1

Results of MC & QMC of M1 with Uncorrelated Uncertainties and accuracy 10^{-3}		
Method	MC	QMC
Converged	Yes	Sometimes
Value	196.77072	196.77704
standard deviation	1.43967	1.44959
Coverage factor	1.97407	1.95717
Skewness	0.01989	0.02825
Value error	-2.85995E-4	9.31199E-4
Time elapsed while iterating	4676.1 (ms)	56014.6 (ms)
Time elapsed while sorting	7.0 (ms)	130.9 (ms)
Number of iterations	7400	91750

Table 11 - Results of MC & QMC of M1 with Uncorrelated Uncertainties and accuracy 10-3

7.1.3 Discussion of the uncertainty results

All three methods can obviously be used to approximate the value of a process variable and the uncertainty in that value. The approximation of the value coincides over the first four significant digits. Four digits should be acceptable in most practical applications even though a higher accuracy might be desired. The accuracy in absolute terms is difficult to estimate but the values of Table 5 and 8 are exact.

The accuracy in the standard uncertainty is a bit worse, only two significant digits coinciding. Two digits might not look like much but it's a satisfying result since it implies that all three methods successfully evaluated the standard uncertainty. And two significant digits are usually enough since the whole procedure of identifying sources of uncertainty is of an approximate nature.

Another interesting result is the number of iterations needed to satisfy the stop condition. The results of the standard MC method does deviate from the theoretical bounds, the results indicate a much faster convergence rate. The QMC doesn't improve on the convergence rate of standard MC. The exact convergence rate of the QMC simulations is clouded by the minimum and maximum limitations, but the performance is worse than standard MC for the M2 and better for M1. The difference in performance on M1 was small, QMC requires slightly fewer iterations. The difference in performance on M2 was much more significant. The standard MC performed better than expected whereas QMC rarely succeeded to converge with the maximum number of iterations. Whether QMC's failure for M2 is a result of the problem having high effective dimensionality or if there is a mismatch between this specific implementation and the problems, cannot be deduced from these trials. QMC would be unsuitable as a general method for uncertainty evaluations if the bad performance can be explained by the problem having high effective dimension (since M2 is a veritable model were the uncertainty is of interest). This study provides no support for QMC as an effective general method for propagating uncertainty, regardless of the explanation.

Closely coupled with the number of iterations needed to achieve certain accuracy is the time consumed by the calculation. This time measure isn't a generalizable result for the

methods; it depends on loads of different factors such as the technologies used in the implementation and the specific computer used. The MC methods needed approximately five seconds to complete 8000 iterations on a large model. This indicates that the methods probably would be too slow for online implementations of automatic uncertainty evaluation. The time consumed by the GUM algorithm is negligible in comparison and seems to be a more suiting option for online implementations of automatic uncertainty evaluations.

The trials above focused mainly on how fast the methods completed the task of propagating the uncertainty through a model. There are, as mentioned in the chapter named uncertainty, other aspects to regard. A couple of examples, briefly treated in the tests, are the coverage factor and symmetry. Other examples are nonlinearity in models, scale vs. frequency and the volatility of the system. The GUM algorithm has to treat non-linearity by approximations and rely on assumptions for the coverage factor and symmetry. The specifications in chapter 2 and 3 aimed to be as general as possible, but the system and processes using automatic measurement system have variant characteristics. Even the GUM algorithm might be too slow for a very large system (scale) requiring a short unit time (frequent). System with varying ways (volatility) to evaluate each PV might have greater use for automatic and frequent uncertainty evaluation.

An alternative or complement to automatic uncertainty evaluation with the GUM algorithm would be an offline tool that does the evaluation on demand. The time consumed by the MC methods to achieve sufficient accuracy does allow for usage of such tools. An offline MC tools could also be used to validate the calculations and assumptions made in online evaluation with the GUM method. Such a tool could be used to verify sufficiently often that the standard uncertainty has been correctly evaluated. But it could also be used to calculate a coverage factor and the symmetry of the PDF. For example, if a coverage factor is calculated every twentieth unit time average and remains close to constant over time, then that coverage factor could be assumed valid for unit time averages within the interval where the coverage factor remained almost constant.

7.2 Result of weighting

M3 was used in order to verify and get some practical sense of consistency checking and accuracy gain by combining measurement results. The model was chosen because it is an actual case of where there are two ways of evaluation a PV and yet simple enough for the results to be interpretable.

A normal distributed uncertainty was added to each measured result, just as in the evaluation of uncertainty methods above. The standard deviations of these uncertainties were defined relatively to the magnitude of the corresponding measured result. Test were performed at three levels of the uncertainties standard deviations; 1, 1.5 and 2 percent of corresponding measured result (in this chapter the standard deviation of the uncertainties will be specified in percentages, which is to be read as percentages of the corresponding measured result), to illustrate the common effect of implementing both consistency checking and weighted averaging. All three levels of the standard deviation were run for the uncertainties set to be uncorrelated, but also correlated according to the correlation matrix in fig 7 from the evaluation of uncertainty methods.

The results are presented in the Tables 12 and 13, compartmentalized on whether the uncertainties were considered correlated or not. The numbers shown are mean values of

24 different sets of measured values; each set represents the average over one hour, so together the mean values presented represents the average for a day of production (rounded to two decimals precision). The column mode indicates how the value was evaluated; v1 means that the value was taken from v1 in M3, v2 as with v1, average that the value is the arithmetic average of the inputs and optimal is the optimal weighting described in chapter 6. *Nr of consistent* is the column that denotes how many of the hourly averages passed the consistency test. For each hour where the consistency test failed, the average or optimal value was replaced with the most accurate value of V1 or V2.

Perhaps the most obvious result of the tests is that uncertainties with a standard deviation of one percent did not result in any consistent measurements whereas almost all measurements when the standard deviation was set to two percent of the measurements value passed the consistency check. This result is, unfortunately, only valid in this particular case and doesn't allow for any general conclusions to be drawn.

It can be deduced from Table 12 that uncorrelated input uncertainties with a standard deviation of 1.5 percent gave a total uncertainty of 1.27 percent of the value using the average method and 1.38 percent using the optimal method. In comparison with the best of the two basic values, v2 with a standard uncertainty of 1.5 percent, these methods improved standard deviation of the uncertainties by 23 respectively 12 percent. For input uncertainties with a standard uncertainty of 2 percent; the average method resulted in a standard uncertainty of 1.48 percent and the optimal method 1.51 percent. So even in this case where the optimal method actually produced optimal weights due to the non-existing correlation, the method didn't succeed to produce an optimal result. Even a simple arithmetic average provided better results, simply because more of the 24 hour averages could be considered consistent.

All the weights generated by the optimal method were 0.56±0.02 for v1and 0.44±0.02 for v2, i.e. no weight deviated more than 0.08 from the 0.5-weighting used by the average method. Hence the difference in the result gained from the two methods for an hour where the measured values were found consistent, would be small.

Weighting results for uncorrelated uncertainties

Uncertainty level	Mode	Nr of consistent	Value	Standard uncertainty
1 procent	Actual	0	152.80	1.66
	Pred	0	146.40	1.46
	Average	0	146.40	1.46
	Optimal	0	146.40	1.46
1,5 procent	Actual	0	152.80	2.49
	Pred	0	146.40	2.12
	Average	14	148.12	1.88
	Optimal	7	147.08	2.03
2 procent	Actual	0	152.80	3.31
	Pred	0	146.40	2.92
	Average	24	149.60	2.21
	Optimal	22	148.91	2.25

Table 12 -Weighting results for uncorrelated uncertainties

The results changed slightly when correlations were introduced into the tests. The correlation matrix used in the uncertainty test was recycled and used to define the

structure and magnitude of the correlations. A comparison between the uncertainties in Tables 12 and 13 show that the total uncertainty of v1 increased due to these correlations whilst that of v2 remained unchanged. It can also be read from Table 13 that using the arithmetic average still outperforms the optimal weights which now only showed seven consistent hour averages. But also the effectiveness of the arithmetic average method was severely penalized by the correlations, the improvement in standard uncertainty was only 0.08 percent in comparison with the best simple value v2, this with a standard deviation of the basic uncertainties at 2 percent.

Weighting results for correlated uncertainties				
Uncertainty level	Mode	Nr of consistent	Value	Standard uncertainty
1 procent	Actual	0	152.80	1.94
	Pred	0	146.40	1.46
	Average	0	146.40	1.46
	Optimal	0	146.40	1.46
1,5 procent	Actual	0	152.80	2.90
	Pred	0	146.40	2.20
	Average	7	147.17	2.18
	Optimal	0	146.40	2.20
2 procent	Actual	0	152.80	3.87
	Pred	0	146.40	2.93
	Average	22	149.27	2.85
	Optimal	7	146.97	2.87

Table 13 - Weighting results for correlated uncertainties

7.2.1 Discussion of weighting

The absolute results presented in Tables 12 and 13 cannot be generalized, but they show that the methods can be used successfully. The gain in accuracy was quite small in several cases but better for the largest uncertainties. These larger uncertainties could be argued to constitute a more realistic case since a large proportion of those hourly averages could be considered consistent. Successful simultaneous measurements of the same physical quantity should be consistent, so cases with few consistent hourly averages should be considered less realistic or as realizations where one measurement failed.

The better results of average method than of the optimal-method can also be explained by the by trueness of the case, the optimal method would, of course, produce better results for uncorrelated uncertainties where all measurements could be considered consistent. So the optimal method would produce the best results in a fine tuned reliable system even if the results above display the opposite.

The lessons learned from the experiment is that combining consistent measurements is indeed useful for the calculation of trustworthy values of process variables and that the choice of method for producing the weights might depend slightly on the particular system. The optimal method should be preferred for most sufficiently fine tuned reliable systems, but as the quality declines and not all measurements can be considered consistent for the optimal method, then the usage of the average method might produce better results.

8 Concluding discussion

The purpose of constructing a system that operates on averages is to lower the amount of data that needs to be stored and processed. The implicit aim was to specify methods that allowed averaging over large time periods and didn't require extensive knowledge or dubious assumptions. Two attributes were introduced to tackle these issues; confidence and active time. The active time allows shorter time periods to be considered in the calculations, but still requires an assumption of where the active periods are placed within the time unit. The assumed rule induces calculations that result in a total value equal to the result gained if all averages were expanded to be valid for the entire time unit. Explicitly stating the active time in a separate attribute is still worthwhile since the source for active time doesn't have to be the same as for the average value. The confidence, describing error due to the averaging, assumes that there's no covariance between PVs (of the variance within a unit time average). The quality description would be incorrect if PVs with covariance within the unit time are combined by nonlinear operations. The exact gain of incorporating active time is difficult to assess but the incorporation confidence substantially lowers the assumptions required. Lowering the requirements from a resolution that eliminates systematic variation within the unit time averages to a resolution where no nonlinear operations are performed on unit time averages that exhibits common systematic variation. This allows the system to use a greater unit time than otherwise while still retaining an accurate data description.

Three different methods to propagate uncertainty were evaluated; an algorithm based on the GUM framework, a standard Monte Carlo method and a Quasi-Monte Carlo method. The family of Monte Carlo methods is known to be slow for multidimensional problems. Tests were performed to verify if they could be fast enough for automatic uncertainty evaluation. Test of the time consumed to solve various problems with Monte Carlo simulations has been conducted before. But the computational speed of standard computers increases all the time and uncertainty, being an approximate concept, doesn't require high accuracy in the simulations. This provided reason to believe that the method might be fast enough for this particular application. The five seconds required by standard Monte Carlo to provide an accuracy 10^{-3} for M2, confirmed that the methods still would be too slow for automatic uncertainty evaluation of most systems. For example, if the unit time was set to one hour and the system contained a thousand PVs with about the same complexity as M2, then the total time required for the simulations would have been 5000 seconds[15]. This must be considered too slow since one hour still corresponds to 3600 seconds. The quasi-Monte Carlo failed to shorten the time consumed by the simulation. A successfully test wouldn't have implied that QMC is suitable as a general method (since the convergence rate is closely coupled with the effective dimension of the problem). But if at least one real high dimensional measurement uncertainty problem proved to gain substantially from the use of QMC, then it would have been of interest to produce methods of assessing the problems effective dimension and thereafter identify the MC method to use. The overall conclusion from the test is that automatic uncertainty evaluation must be conducted with the GUM-algorithm. A Monte Carlo method, implemented as an offline tool, constitute a good compliment to verify the results produced by the GUM-algorithm and calculate

[15] A unit time of one hour and a thousand PVs isn't an unreasonable case, A large process industry contains lots of PVs and some, for example refineries, have stable production that changes slowly.

coverage factors. Another alternative for PVs, whose value sources remain basically unchanged, is to use an MC method and evaluate the uncertainty for every N^{th} unit time average.

Another argument for the use a GUM algorithm is that it would be much simpler, as compared to MC methods, to combine with consistency checking. Only consistent values of a PV should be combined and thereby produce more accurate values. The description in section 5.2 showed that optimization of the weights used to combine the different sources is a difficult problem. Two different alternatives were proposed to avoid solving the optimization problem: the arithmetic average and optimal weights under assumed independence. The results of the tests in section 7.2 showed that both methods where about equal in performance. The slight difference, as can be predicted from their mathematical description, was that the arithmetic average produced more accurate results for doubtful descriptions of the uncertainty and the optimal method was a bit better when both methods produced a consistent combination. Both methods should produce consistent combinations in a finely tuned system, where the uncertainty of the values is correctly described. The arithmetic average is probably the best overall choice since it aids doubtful uncertainty descriptions better and the doubtful descriptions can be assumed to be in greatest need of improvement. Using the arithmetic average would demand a small test on whether the average does improve on the inputs. Any of the inputs might constitute a less uncertain value than the arithmetic average.

The methods proposed in this thesis provide a good foundation for a system operating on averages of PVs. They provide an adequate description of the data, need only a few assumptions and an implementation would be straightforward. The methodological issues of this work, as described in the introduction, restrict the claim of the hereby proposed specifications to being a *good* foundation. There might be a more suitable set of data quality dimensions, a more clever time specification, an algorithm to find the optimal weights etc. These subjects should all be up for debate and this work can, at least, serve as both a contribution and foundation for such a debate.

9 References

Cacciatore, V., Carlo, A., Paris, M. & A. Allan, (2006). Uncertainty Effects of Data Compression in Measurement Applications, Instrumentation and Measurement Technology Conference, 2006. Proceedings of the IEEE, pp. 1462-1467

Cox, M. G. & Harris, P. M., (2006). Software Support for Metrology, Best Practice Guide No. 6: Uncertainty Evaluation, National Physics Laboratory. Available at: http://publications.npl.co.uk/npl_web/pdf/dem_es11.pdf (2008-08-02).

Dick, J. & Niederreiter, H.,(2008). *On the exact t-value of Niederreiter and Sobol' sequences*, Journal of Complexity, ***In Press, Corrected Proof***, *Available online 24 May 2008*

Glad, T. & Ljung, L, (1981), *Reglerteknik grundläggande teori*, 4th ed., Studentlitteratur, Lund

Gleser, L. J, (1998). *Assessing Uncertainty in Measurement*, Statistical Science, volume 13, number 3, pp. 277-290

Gregory, K., Bibbo, G. & Pattison, J.E., (2005). *A standard approach to measurements uncertainties for scientists and engineers in medicine*, Australasian physical & engineering sciences in medicine, volume 28, issue 2, pp.131-139

Haas, P., (1997). *Large sample and deterministic confidence intervals for online aggregation*, Proceedings of the Ninth International Conference on Scientific and Statistical Database Management, IEEE Computer Society, Washington, pp. 51 - 63

Hall, B.D., (2005). *Propagating uncertainty in instrument systems*, IEEE Transactions on Instrumentation and Measurement, Volume 54, Issue 6, pp. 2376 - 2380

Holmberg, M., (2008), Engineer at ÅF Engineering in Nynäshamn, continuous talks throughout the work

Hässelbarth, W. & Bremser, W., (1998),*Derived measurement standards of reduced uncertatiny- A contradiction?*, Accreditation and Quality Assurance: Journal for Quality, Comparability and Reliability in Chemical Measurement, Berlin, volume 3, number 9, pp. 337-339

IPCC 1, (1996). *Revised 1996 IPCC Guidelines for National Greenhouse Gas Inventories: Reporting Instructions,* International panel on climate change.

Available at:
http://cd4cdm.org/Asia/Philippines/Training%20Workshop/day3/public/gl/invs4.htm (2008-08-03)

IPCC (2000), *Good Practice Guidance and Uncertainty Management in National Greenhouse Gas Inventories,* IPCC

Kessel, R., Berglund, M. & Wellum, R., (2008), *Application of consistency checking to evaluation of uncertatinty in multiple replicate measurements,* Accreditation and Quality Assurance: Journal for Quality, Comparability and Reliability in Chemical Measurement, volume 13, number 6, pp. 293-298.

Kirrman, H., (2005). *4 Access to devices*, slideshow created by Prof. Dr. H. Kirrmann, ABB Research Centre, Baden

Klein, A. (2007): *Incorporating quality aspects in sensor data streams*, Proceedings of the ACM first Ph. D. workshop in CIKM 2007, session 3, ACM, New York, pp. 77-84

L'Ecuyer, P., (2003). *Quasi-Monte Carlo Methods for Simulation,* Proceedings of the 2003 Winter Simulation Conference, session: advanced tutorials, Winter Simulation Conference, pp. 81-89

Lindeskog, J., (2006), *Mätvärdesbehandling och rapportering av mätresultat*, 3rd ed., Studentlittertur, Lund

MacGregor, J. F., (1997). *Using On-Line Process Data to improve Quality: Challenges for statisticians,* International Statistical Review, volume 65, pp. 310-316

Matsumoto, M. & Nishimura, T., (1998). *Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator* , ACM Trans. on Modeling and Computer Simulation Vol. 8, No. 1, pp.3-30

NFS 2007:5, (2007). *Naturvårdsverkets föreskrifter och allmänna råd om utsläppsrätter för koldioxid,* Naturvårdsverkets författningssamling, Naturvårdsverket, Stockholm

Niederreiter, H., (1988), *Low-Discreapance and Low-dispersion sequences*, Journal of Number Theory, volume 30, 51-70

NIST, (2000). *The Nist Reference on constants, units and uncertainty; uncertainty of measurement results*, National institute of standards and technology, http://physics.nist.gov/cuu/Uncertainty/index.html (2008-07-05)

Office of the manager; National communications system, (2004). Supervisory Control and Data Acquisition (SCADA) Systems (NCS TIB 04-1).

available at: http://www.ncs.gov/library/tech_bulletins/2004/tib_04-1.pdf#search='Future%20Trends%20SCADA' (2008-07-23)

OPC About. *About OPC*, available at: http://www.opcfoundation.org/Default.aspx/01_about/01_whatis.asp?MID=AboutOPC# open (2008-08-13)

OPC HDA, (2003). *OPC Historical data access version 1.20*, OPC foundation, available to members at: http://www.opcfoundation.org/Downloads.aspx?CM=1&CN=KEY&CI=283 (2008-08-05)

OSIsoft, (2008). *A Universal Platform that Improves Operations Across any Industry*, available at: http://www.osisoft.com/Products/PI%20System/ (2008-08-05)

Owen, A. B., (2002), *Necessity of low effective dimension*, Stanford University report, available at: http://www-stat.stanford.edu/~owen/reports/necessity.pdf, (2008-07-05).

Papadopoulos, C, E. & Yeung, H., (2001). *Uncertainty estimation and Monte Carlo simulation method*, Flow Measurement and Instrumentation, volume 12, issue 4, pp. 291-298

Rao, N. S. V., (2004). *A generic sensor fusion problem: classification and function estimation*, Multiple classifier systems, volume 3077 of Lecture notes in computer science, Springer, Berlin, pp. 16-30

Råde, L. & Westergren, B., (1998), *Mathematics handbook for science and engineering*, 4th ed., Studentlitteratur, Lund

SSJ API Specification, *SSJ : A Java library for a stochastic simulation API specification*, available at: http://www.iro.umontreal.ca/~simardr/ssj/doc/html/overview-summary.html (2008-08-15)

Tabachnick, B. G. & Fidell, L. S. (1996). *Using multivariate statistics*, 3rd ed., Harper Collins, New York

Taylor, B. N. & Kuyatt, C. E., (1994). *Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results, NIST Technical Note 12971994 Ed* , National institute of standards and technology, Gaithersburg.
Available at: http://physics.nist.gov/Pubs/guidelines/TN1297/tn1297s.pdf (2008-09-01)

Uncommons Maths, *Uncommons Maths: Random number generators and other mathematical utilities,* https://uncommons-maths.dev.java.net (2008-07-15)

Wand, Y., & Wang, R. Y., (1996), *Anchoring data quality dimensions in ontological foundations,* Communications of the ACM, Vol. 39, No. 11

Wang, R.Y., Storey, V.C., and Firth, C.P., (1995), *A framework for analysis of data quality research,* IEEE Trans. on Knowl. Data Eng., volume 7, issue 4, 623–640

Watkinson, J., (2002). *An Introduction to Digital Audio*, 2nd ed., Focal Press, Oxford

Webopedia: Scada, (modified: 2007-05-30), *SCADA,* http://www.webopedia.com/TERM/S/SCADA.html, (2008-08-20)

Wikipedia: sampling, (modified: 2008-09-04). *Sampling (signal processing),* http://en.wikipedia.org/wiki/Sampling_(signal_processing) (2008-08-16)

Wikipedia: sample rate conversion, (modified: 2008-08-15). *Sample rate conversion*, http://en.wikipedia.org/wiki/Sample_rate_conversion (2008-08-19)

Willink, R., (2008). An inconsistency in uncertainty analysis relating to effective degrees of freedom, Metrologia, volume 45**,** 63–67

Appendix A

The unbiased sample standard deviation is defined as:

$$\sigma(X) = \left(\frac{\sum_{i=1}^{n} (x_i - \overline{X})^2}{n-1} \right)^{1/2} \quad \text{(A.1)}$$

where n is the sample size, x_i the individual samples and $\overline{X}$ the sample average.

The concept of confidence as explained by Klein uses the large-sample confidence intervals described by Haas (Klein, 2007; Haas, 1997). These large-sample confidence intervals can be erroneous and whole concept of confidence should therefore be regarded as approximate (Haas, 1997). The construction of these intervals relies on the assumption that the observations used can be regarded as a random sample large enough for the central limit theorem to apply (Haas, 1997).

The confidence of an average aggregate of a variable *X* is, under these assumptions, defined as:

$$\varepsilon(X) = \left(\frac{d^2 \sigma^2(X)}{n} \right)^{1/2} \quad \text{(A.2)}$$

where d is the coverage factor that expands the confidence level of the interval to 95 percent. The logic of this definition should be clear; the confidence interval for an observation of the population is $x_i \pm d\sigma(X)$ and the errors are combined by Gaussian error propagation to form the interval of the average. The Gaussian error propagation assumes independence between errors and is normal in scientific applications (Klein,2007; Lindskog, 2006). The Gaussian error propagation is based on a first order Taylor approximation of the variance of a real function of independent variables. The approximation:

$$Var(g(X_1,....X_n)) \approx \sum_{i=1}^{n} \left(\frac{\partial g}{\partial x_i} \right)^2 Var(g(X_i)) \quad \text{(A.3)}$$

is exact for linear functions (Råde & Westergren, 1998). The assumed independence should be regarded as a compromise; some variables might have a positive covariance that would increase the total variance and others a negative covariance lowering the total variance (Lindskog, 2006).

The situation changes slightly if the samples X_i were subjected to some downsampling technique before the averaging. Each X_i would contribute their confidence and those confidences should, again assuming independence, be combined by Gaussian error propagation:

$$\varepsilon\left(f\left(X_1,....X_n\right)\right)=\sqrt{\sum_{i=1}^{n}\left(\frac{\partial f}{dX_i}\right)^2\varepsilon^2\left(X_i\right)} \tag{A.4}$$

But the averaging would also introduce another error on the form of (A.2). Klein imposes that this new error and those of the individual components X_i should be considered dependent and therefore combined by linear addition (as for fully positively dependent variables):

$$\varepsilon_N=\frac{\sqrt{\sum_{i=1}^{N}\varepsilon^2\left(X_i\right)}}{N}+\left(\frac{d^2\sigma^2\left(Y\right)}{n}\right)^{1/2} \tag{A.5}$$

where each $\varepsilon\left(X_i\right)$ corresponds to the confidence of the individual input X_i and Y represents the average aggregate (Klein, 2007). This corresponds to the formula (3.2).

The confidence of a result gained by performing linear operations on unit time averages, such as addition or subtraction, is calculated according to (A.3). This should be self-evident since the operations don't introduce any new error apart from the initial sampling process. The situation is slightly different for the operations denoted nonlinear. For example, a new error is introduced by calculating a the average of a PV Y that is defined as a multiplication of two other PVs X_1, X_2 as $\overline{Y}=\overline{X}_1\overline{X}_2$. This will be handled by consider $\overline{X}_1\pm d\sigma\left(X_1\right)$ to be a 95 percent confidence interval for each value used to form $\overline{X}_1$. This would imply, using Gaussian error propagation that:

$$\overline{X}_1\overline{X}_2\pm\sqrt{\left(\frac{\partial(x_1x_2)}{\partial x_2}\right)^2 d^2\sigma\left(X_1\right)^2+\left(\frac{\partial(x_1x_2)}{\partial x_1}\right)^2 d^2\sigma\left(X_2\right)^2} \tag{A.6}$$

constitutes a 95 percent confidence interval for each individual observation of X_1X_2. Averaging of n such observation would therefore have to consider two sources of errors in the average: those of the individual observation according to (A.5) and those of the averaging itself according to (A.2). These two sources must, of course, be considered dependent and the errors combined by linear addition. The resulting formula for multiplication is:

$$\begin{aligned}\varepsilon\left(average\left(X_1X_2\right)\right)&=\sqrt{\frac{\overline{X}_2{}^2\varepsilon^2\left(X_1\right)}{n_1}+\frac{\overline{X}_1{}^2\varepsilon^2\left(X_2\right)}{n_2}}+\sqrt{\frac{\overline{X}_2{}^2d^2\sigma^2\left(X_1\right)}{n_1}+\frac{\overline{X}_1{}^2d^2\sigma^2\left(X_2\right)}{n_2}}=\\&=2\sqrt{\overline{X}_2{}^2\varepsilon^2\left(X_1\right)+\overline{X}_1{}^2\varepsilon^2\left(X_2\right)}\end{aligned} \tag{A.7}$$

where each first partial derivate has been substituted with the calculated average (this would, strictly formally, introduce another small error). The derivation of formulas for other nonlinear operations can be conducted similarly.

Appendix B

This appendix provides a numerical example of the basic operations multiplication and addition. The example in Table 1 of section 2.3 will be reused to provide the numerical PVs. Those numbers are reproduced in Table 14 below.

Sample nr	X_1	X_2	X_1+X_2	X_1*X_2
1	2	6	8	12
2	5	4	9	20
3	2	4	6	8
4	1	7	8	7
5	7	9	16	63
6	3	2	5	6
7	6	3	9	18
8	1	7	8	7
9	4	4	8	16
10	7	2	9	14
Average	3.8	4.8	8.6	17.1

Table 14 - The example

Some more information is needed to construct unit time averages for X_1 and X_2. The time description, the completeness and standard deviation are parameters that are normally provided by the measurement system. The measurement uncertainty is usually provided by some other external source. But this is a fictional example without real data sources and these values have to be assigned manually. The standard deviation can be calculated from the values of the individual samples. Calculating the unbiased standard deviation estimator of formula (3.22) results in:

$$Std\ dev(X_1) = \sqrt{\frac{(2-3.8)^2 + (5-3.8)^2 + (2-3.8)^2 + (1-3.8)^2 + (7-3.8)^2 + (3-3.8)^2 + (6-3.8)^2 + (1-3.8)^2 + (4-3.8)^2 + (7-3.8)^2}{9}} = 2.347576 \quad \text{(B.1)}$$

for X_1 and

$$Std\ dev(X_2) = \sqrt{\frac{(6-4.8)^2 + (4-4.8)^2 + (4-4.8)^2 + (7-4.8)^2 + (9-4.8)^2 + (2-4.8)^2 + (3-4.8)^2 + (7-4.8)^2 + (4-4.8)^2 + (2-4.8)^2}{9}} = 2.347576 \quad \text{(B.2)}$$

for X_2. The other parameters have to be stipulated. Let these be chosen according to:

- Completeness = 100 percent
- Timestamp = 2008-09-18 12:00
- Duration = 1
- Active time = 1
- Standard uncertainty = 1 percent of the PV's value
- The uncertainties of the inputs are mutually independent

for both PVs. The confidence is calculated according to equation (3.1) and taking the coverage factor as 2 yields:

$$\varepsilon(X_i) = \left(\frac{2^2 \sigma^2(X_i)}{10} \right)^{1/2} = 1.484737 \quad \text{(B.3)}$$

for both i=1 and i=2. Choosing the coverage factor as 2 is a slight simplification. An exact coverage factor for a large sample confidence interval with nine degrees of freedom is slightly larger. The resulting complete unit time averages are shown in Table 15 and 16.

PV	X1
Value	3.8
Timestamp	2008-09-18 12:00
Active Time	1
Duration	1
Standard Deviation	2.347586
Standard Uncertainty	0.038
Confidence	1.484737
Completeness	100

Table 15 - X1

PV	X2
Value	4.8
Timestamp	2008-09-18 12:00
Active Time	1
Duration	1
Standard Deviation	2.347586
Standard Uncertainty	0.048
Confidence	1.484737
Completeness	100

Table 16 - X2

Let's first consider the case when these two PV are combined by addition. The specifications for this operation were given in section 3.2.1. The value of the result is specified in equation (3.5):

$$Y = \frac{a(X_1) \cdot X_1 + a(X_2) \cdot X_2}{\max(a(X_1), a(X_2))} = \frac{1 \cdot 3.8 + 1 \cdot 4.8}{1} = 8.6 \quad \text{(B.4)}$$

where Y is the result of the addition. The timestamp of Y will be that of X_1, the duration specified in formula (3.6) will be 1 and equation (3.7) yields an active time of 1. The completeness will remain at 100 percent since it's calculated as the average of the completeness of all input PVs. The standard deviation of Y is specified in equation (3.8):

$$\text{std}(Y) = \sqrt{\frac{a(X_1) std^2(X_1) + a(X_2) std^2(X_2)}{\max(a(X_1), a(X_2))}} = \sqrt{(2.347586)^2 + (2.347586)^2} \quad \text{(B.5)}$$
$$= 3.319973$$

and the confidence specified in equation (3.9) is calculated according to (B.6).

$$\varepsilon(Y)=\sqrt{\left(\frac{\partial Y}{\partial X_1}\right)^2\varepsilon^2(X_1)+\left(\frac{\partial Y}{\partial X_2}\right)^2\varepsilon^2(X_2)}=\sqrt{(1)^2(1.484737)^2+(1)^2(1.484737)^2}$$
$$=2.099735 \qquad \text{(B.6)}$$

The uncertainty in this example will be propagated according to the GUM framework. The reason for this choice is that it would be impossible to illustrate a Monte Carlo simulation. Addition of two variables is propagated according to (4.1):

$$u^2(Y)=\sum_{k=1}^{2}(c_k u(X_k))^2+2c_1c_2u(X_1)u(X_2)r(X_1,X_2) \qquad \text{(B.7)}$$

where $r(x_i,x_j)=0$ due to the mutual independence of the input uncertainties,

$c_1=\dfrac{a(X_1)}{\max(a(X_1),a(X_2))}=1$ and $c_2=\dfrac{a(X_2)}{\max(a(X_1),a(X_2))}=1$.

So the numerical value of the standard uncertainty is:

$$u(Y)=\sqrt{u(X_1)^2+u(X_2)^2}=\sqrt{(0.038)^2+(0.048)^2}=0.061221. \qquad \text{(B.8)}$$

Table 17 contains a summary of numerical unit time average. The confidence is quite large but this is an expected result if both inputs have such large standard deviations.

PV	X1+X2
Value	8.6
Timestamp	2008-09-18 12:00
Active Time	1
Duration	1
Standard Deviation	3.319973
Standard Uncertainty	0.061221
Confidence	2.099735
Completeness	100

Table 17 - Addition of X1 and X2

An example of a nonlinear operation is the multiplication specified in section 3.2.3. The unit time averages specified in Table 15 and 16 will be reused for this illustration. The time specification of the result becomes as that of the addition. The value:

$$Y=X_1X_2=3.8\cdot 4.8=18.24 \qquad \text{(B.9)}$$

which is 1.14 larger than the result presented in Table 14. The standard deviation calculated according to equation (3.13) is calculated in formula (B.10).

$$std(Y)=\sqrt{\overline{X}_2{}^2std^2(X_1)+\overline{X}_1{}^2std^2(X_2)+std^2(X_1)std^2(X_2)}$$
$$=\sqrt{4.8^2(2.347586)^2+3.8^2(2.347586)^2+(2.347586)^2(2.347586)^2}=14.37207 \qquad \text{(B.10)}$$

The confidence shown in formulae (B.11) is calculated according to equation (3.14).

$$\varepsilon(Y) = 2\sqrt{\left(\frac{\partial Y}{\partial X_1}\right)^2 \varepsilon^2(X_1) + \left(\frac{\partial Y}{\partial X_2}\right)^2 \varepsilon^2(X_2)} = 2\sqrt{(4.8)^2(1.484737)^2 + (3.8)^2(1.484737)^2}$$
$$= 18.17939 \tag{B.11}$$

The uncertainty propagated with the GUM framework is calculated according to equation (4.1):

$$u^2(Y) = \sum_{k=1}^{2} (c_k u(X_k))^2 + 2c_1 c_2 u(X_1) u(X_2) r(X_1, X_2) \tag{B.12}$$

where $r(x_i, x_j) = 0$, $c_1 = \overline{X}_2 = 4.8$ and $c_1 = \overline{X}_1 = 3.8$. Insertion of the numerical values yields the result shown in formulae (B.13).

$$u(Y) = \sqrt{4.8^2 u(X_1)^2 + 3.8^2 u(X_2)^2} = \sqrt{4.8^2(0.038)^2 + 3.8^2(0.048)^2}$$
$$= 0.257953 \tag{B.13}$$

The result of the multiplication is summarized in Table 18.

PV	X1*X2
Value	18.24
Timestamp	2008-09-18 12:00
Active Time	1
Duration	1
Standard Deviation	14.37207
Standard Uncertainty	0.257953
Confidence	18.17939
Completeness	100

Table 18 - Multiplication of X1 and X2

Appendix C

The Monte Carlo implementation used in the tests. The class has been stripped of all irrelevant content such as get/set- methods. The code presented below is the code used in the actual simulations.

```
package performance;
import ProcessEntity.Ucomp;
import ProcessEntity.Variable;
import ProcessEntity.Meter;
import cern.colt.matrix.impl.DenseDoubleMatrix2D;
import cern.colt.matrix.linalg.EigenvalueDecomposition;
import umontreal.iro.lecuyer.randvar.NormalGen;
import umontreal.iro.lecuyer.randvarmulti.MultinormalCholeskyGen;
import umontreal.iro.lecuyer.probdist.NormalDist;
import umontreal.iro.lecuyer.hups.NiedSequenceBase2;
import umontreal.iro.lecuyer.hups.PointSetIterator;
import java.util.Vector;
import java.util.Collections;
import cern.colt.matrix.linalg.Algebra;
/**
 * Class that takes a variable as an argument, indexes all the
phyMeters found in
 * the tree of the variable.
 * For a moment the class creates a fake covariance Matrix.
 * @author Niklas Molin
 */
public class MonteCarlo
{
	private Vector v;//Vektor where each PhyMeter-object will be
indexed
	private final int it = 10000;
	private double relAccuracy = 0.00001;
	private final int maxit = 100001; // Cannot be much higher for
MCp & qMCp metohds or heap space overflow
	private double[] mu; //Vektor med väntevärde för
feldistributionerna till mätarna
	private DenseDoubleMatrix2D sigma;//Covariance-Matrix
	double[] point;//where each point value will be saved
	double res = 0; //approximation of the variance
	double res1 = 0; //approximation of y
	double old = 1;
	double old1 = 1;
	int nr = 1;
	RandomGen RndStream;
	NormalDist ND;
	NormalGen NDG;
	MultinormalCholeskyGen MNDG;
	private Variable m;
	//RESULT DATA-----------------------------------------------------
	private double y, ymin, ymax, skewness;
	private double itTime, sortTime;
	private double error, std;
	private String method = "";
	private int nrOfIt = 0;
	//----------------------------------------------------------------
	public MonteCarlo(Variable m)
	{
		this.m = m;
```

```
y = m.getValue();
v = new Vector();//Vektor where each PhyMeter-object will
be indexed
getList(m);//Indexed all the Phymeters in the vector
// System.out.print("Antalet mätare med osäkerhet: " +
      v.size());
Meter m1;// Variable used to save references to meters
tepomorary
sigma = new DenseDoubleMatrix2D(v.size(), v.size());
//sigma = new double[v.size()][v.size()];
mu = new double[v.size()];
point = new double[mu.length];//where each point value
will be saved
//The loop is supposed to load the covariance and variance
of the meters
//into the covariance-Matrix

// for (int i = 0; i<
v.size();i++)System.out.println(((Ucomp)((Meter)
v.elementAt(i)).getU().elementAt(0)).getId());
for (int i = 0; i < v.size(); i++)
{
      m1 = (Meter)v.elementAt(i);
      mu[i] = 0;

      for (int j = 0; j < v.size(); j++)
      {
            if (j == i)
            {
                  try
                  {
                  sigma.setQuick(i, j,
                  Math.pow(((Ucomp)m1.getU().elementAt(0))
                  .getValue(), 2));

                  }
                  catch (Exception ex)
                  {
                        System.out.println(i + "   " +
                        m1.getClass() + "  " +
                        m1.getName());
                  }

            }
            else if (j - 1 == i)
            {

                  sigma.setQuick(i, j, 0.5 *
                  ((Ucomp)m1.getU().elementAt(0)).getValue
                  () *
                  ((Ucomp)((Meter)v.elementAt(j)).getU().e
                  lementAt(0)).getValue());
            }
            else if (j + 1 == i)
            {

                  sigma.setQuick(i, j, 0.5 *
                  ((Ucomp)m1.getU().elementAt(0)).getValue
                  () *
                  ((Ucomp)((Meter)v.elementAt(j)).getU().e
                  lementAt(0)).getValue());
```

```
                }

                else
                {
                        sigma.setQuick(i, j, 0);

                }
            }
            Algebra A = new Algebra();
        }

}
public MonteCarlo(Variable m, double acc)
{
        this(m);
        relAccuracy = acc;

}
public void MCp()
{
        method = "MCp";
        //Creation of the object that will generate the points for
        each step
        RndStream = new RandomGen();
        ND = new NormalDist();
        NDG = new NormalGen(RndStream, ND);
        MNDG = new MultinormalCholeskyGen(NDG, mu, sigma);
        //-----------------------------------------------------------
        //The stop critereas and results-------------------------
        double sy = 0;
        double su = 0;
        double ylow = 0;
        double yhigh = 0;
        //-----------------------------------------------------------
        point = new double[mu.length];//where each point value
        will be saved
        res = 0; //uppskattnigen av variansen
        res1 = 0; //uppskattningen av y
        old = 1;
        old1 = 1;
        stdOld = 0;
        nr = 1;
        Meter m1;
        Vector all = new Vector();
        double thirdMoment = 0;
        double start = System.currentTimeMillis();
        while ((Math.abs((res1 / (it * (nr - 1)) - m.getValue()) /
        m.getValue()) > relAccuracy || Math.abs((std - stdOld) /
        std) > relAccuracy || nr < 2) && nr * it < maxit * 100)
        {

                old = res;
                old1 = res1;
                stdOld = std;
                for (int k = 0; k < it; k++)
                {
                        MNDG.nextPoint(point);
                        for (int i = 0; i < v.size(); i++)
                        {
                                m1 = (Meter)v.elementAt(i);
                                m1.setMCvalue(point[i]);
```

```
                    }
                    double b1 = m.evalMC();
                    double b2 = m.getValue();
                    res += Math.pow(b1 - b2, 2);
                    res1 += b1;
                    all.add(b1);
                    thirdMoment += Math.pow(b1 - b2, 3);
              }
              nr++;
              std = (Math.sqrt(res / (it * (nr - 1))));

        }
        double q;
        itTime = System.currentTimeMillis() - start;

        error = m.getValue() - res1 / (it * (nr - 1));
        if (isInt(0.95 * it * (nr - 1)))
        {
              q = (int)0.95 * it * (nr - 1);
        }
        else
        {
              q = (int)0.95 * it * (nr - 1) + 1 / 2;
        }
        start = System.currentTimeMillis();
        Collections.sort(all);
        sortTime = System.currentTimeMillis() - start;
        double n = (it * (nr - 1));
        nrOfIt = it * (nr - 1);
        ymin = ((Double)all.elementAt((int)(it * (nr - 1)) /
        40)).doubleValue();
        ymax = ((Double)all.elementAt(all.size() - ((int)it * (nr
        - 1) / 40))).doubleValue();
        skewness = (Math.sqrt(n * (n - 1)) / (n - 2)) *
        (thirdMoment / n) / (Math.pow((1 / n) * res, 3 / 2));

}

public void qMCp()
{

/*************************************************************
***
* int log2nrPoints -Decides the nr of point to be generated
* the number of point will be 2^log2nrPoints where 0 <=
log2nrPoints <= 30
* int w - w is the number of output digits and w<=log2nrPoints
      * int dim - is the dimension of the sequence and is
      restricted to 318
      */
      method = "qMCp";
      Vector all = new Vector();
      Meter m1;// Variable used to save references to meters
      tepomorary
      //RndStream1.increasedPrecision(true);
      int log2nrPoints = 13;
      int w = 31;
      int nrOfPoints = (int)Math.pow(2, log2nrPoints);
      int dim = mu.length;//we want points for all inputs
      //System.out.println("FELET : "+dim);
```

```
NiedSequenceBase2 nxsb2 = new
NiedSequenceBase2(log2nrPoints, w, dim);
RndStream = new RandomGen();
//RndStream1.resetNextSubstream();
//nxsb2.addRandomShift();
//RandShiftedPoint
nxsb2.addRandomShift(0, dim, RndStream);
nxsb2.rightMatrixScramble(RndStream);
PointSetIterator a = nxsb2.iterator();
NormalGen[] NDGa = new NormalGen[mu.length];
//Creation of the object that will generate the points for
each step

ND = new NormalDist();
NDG = new NormalGen(a, ND);
try
{
      MNDG = new MultinormalCholeskyGen(NDG, mu, sigma);
}
catch (Exception ex)
{
      System.out.println(ex);
}
//-----------------------------------------------------------
-----------
// for(int q = 0; q < mu.length ;q++)NDGa[q] = new
NormalGen( a, new NormalDist(mu[q], sigma[q][q]));

//MNDG = new MultinormalCholeskyGen(NDG,mu,sigma);//DET ÄR
VAR TIDEN FÖRSVINNER
double thirdMoment = 0;
double stdOld = 0;
res = 0;
res1 = 0;
old = 1;
old1 = 1;
nr = 1;
double start = System.currentTimeMillis();
while ((Math.abs(((old1 - m.getValue() * ((nr - 1) *
nrOfPoints)) / ((nr - 1) * nrOfPoints * m.getValue()))) >
relAccuracy || Math.abs((std - stdOld) / std) >
relAccuracy) && nr * nrOfPoints < maxit)
{
      nxsb2.addRandomShift(0, dim, RndStream);
      nxsb2.leftMatrixScramble(RndStream);
      a.resetStartStream();
      old = res;
      stdOld = std;

      for (int k = 0; k < nrOfPoints; k++)
      {

            MNDG.nextPoint(point);
            for (int i = 0; i < v.size(); i++)
            {
                  m1 = (Meter)v.elementAt(i);
                  m1.setMCvalue(point[i]);


            }
            double b1 = m.evalMC();
```

```
                        double b2 = m.getValue();
                        all.add(b1);
                        res += Math.pow(b1 - b2, 2);
                        res1 += b1;
                        thirdMoment += Math.pow(b1 - b2, 3);

                        a.resetToNextPoint();
                  }
                  if (nr > 1)
                  {
                        old1 += res1;
                  }
                  else { old1 = res1; }
                  nr++;
                  std = (Math.sqrt(res / (nrOfPoints * (nr - 1))));
            }
            int q;
            itTime = System.currentTimeMillis() - start;
            std = (Math.sqrt(res / (nrOfPoints * (nr - 1))));
            error = m.getValue() - res1 / ((nrOfPoints * (nr - 1)));
            if (isInt(0.95 * nrOfPoints * (nr - 1)))
            {
                  q = (int)0.95 * nrOfPoints * (nr - 1);
            }
            else
            {
                  q = (int)0.95 * nrOfPoints * (nr - 1) + 1 / 2;
            }
            start = System.currentTimeMillis();
            Collections.sort(all);
            sortTime = System.currentTimeMillis() - start;
            double n = (nrOfPoints * (nr - 1));
            ymin = ((Double)all.elementAt((int)(nrOfPoints * (nr - 1))
            / 40)).doubleValue();
            ymax = ((Double)all.elementAt(all.size() -
            ((int)nrOfPoints * (nr - 1) / 40))).doubleValue();
            skewness = (Math.sqrt(n * (n - 1)) / (n - 2)) *
            (thirdMoment / n) / (Math.pow((1 / n) * res, 3 / 2));
            nrOfIt = (nrOfPoints * (nr - 1));

      }
      private boolean isInt(double d)
      {
            return true;
      }
      private void getList(Variable m)
      {
            if (m.getId() > 0 && !v.contains(m))
            {
                  // if (v.size()==0){
                  v.add(m);
                  return;
            }
            else
            {
                  for (int i = 0; i < m.getElements().size(); i++)
                  {

      getList(((Variable)m.getElements().elementAt(i)));
                  }
            }
```

```
        }
    }
```