Surface topography analysis software validation

University of Nottingham

UK | CHINA | MALAYSIA

Luke D Todhunter

Faculty of Engineering

University of Nottingham

Thesis submitted to the University of Nottingham for the degree of

Doctor of Engineering

April 2020

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This thesis contains fewer than 100,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

This work has been carried out with the support of the Engineering and Physical Sciences Research Council (EPSRC) and the Manufacturing Technology Engineering Doctoral Centre.

This work has been funded by Digital Surf, the National Physical Laboratory and EPSRC.

Luke D Todhunter

April 2020

Acknowledgements

I would like to acknowledge Richard Leach, François Blateyron, Peter Harris and Simon Lawes for their consistently valuable supervision and guidance throughout the EngD, and thank Digital Surf, the National Physical Laboratory and EPSRC for funding the project.

I would also like to thank my friends who have helped me endure the past four years. This includes, but is not limited to, Peter Connor, Adam Thompson, Lewis Newton, Lars Korner, Alex Gasper, Duncan Hickman, Hatim Cader, Michael Hillier, Myles Selvey and Owen Kelly. And special thanks to Patrick Bointon, without whom I would surely have completed twice as much work in half the time.

Further thanks go to my parents, who have given me their unwavering support for far longer than anyone else on this list, and my sister, who was also there.

Abstract

This thesis presents work in the field of surface texture analysis, focussing on improving the current state-of-the-art regarding the calculation of areal surface texture parameters by software. The work given in this thesis utilises the mathematical definitions of surface texture parameters to develop mathematically-defined reference values against which third-parties can compare, to assess the performance of their software.

An in-depth analysis of ISO 5436-2 type F2 reference software for the calculation of profile surface texture parameters is performed on the input, implementation and output results of the reference software developed by the National Physical Laboratory (NPL), the National Institute of Standards and Technology (NIST) and Physikalisch-Technische Bundesanstalt (PTB). Algorithm flowcharts are developed for reference software on offer from the three national metrology institutes to understand the structure of each software package and highlight differences between them. Surface texture parameters are calculated for a selection of seventeen test data files obtained from the type F1 reference data sets on offer from NPL and NIST. The results show differing results across the three reference software packages for a variety of test datasets, and attempts are made to explain these differences from the standpoint of software implementation and the interpretation of the definitions given in ISO 25178-2. The investigation concludes that the use of different software implementations contributed to the differences in obtained parameter values.

Results of an international survey are presented, detailing the use of surface texture parameters in industry. The survey received 179 responses from a total of 34 countries,

revealing the use of a variety of parameters from ISO 4287, ISO 12085, ISO 13565-2/3 and ISO 25178-2. The survey responses show an increase in the number of users of profile parameters and an increase in the range of surface texture parameters used, compared to the results from a similar survey in 1999, as well as a significant uptake of the newer areal surface texture parameters. Individual sector usage is also discussed.

A programmable software framework is developed for the user-defined creation of test surfaces. The framework supports the bespoke design of surfaces using a variety of user-configurable functions and utilises a Fourier series approach to produce mathematically-defined surface functions alongside discrete height-map datasets in accordance with ISO 25178-71. In addition, the software introduces the use of multi-scale Fourier space Gaussian blur to incorporate optional pseudo-realistic random elements into the mathematical surface function.

A new method for performance validation of surface texture parameter calculation software is introduced, focussing on functional surface texture parameters. Material ratio curves are defined algebraically and used to calculate functional surface texture parameters mathematically. Discrete datasets are created from the material ratio curves and input into three third-party parameter calculation software packages. Comparisons are made between the software-obtained parameter values and the mathematical values. Work is carried out to highlight inaccuracies introduced by sampling discrete datasets from mathematical representations.

A new method for performance validation of surface texture parameter calculation software is introduced, focussing on field surface texture parameters. Surface height functions are defined mathematically and are then input into the surface texture parameter definitions to obtain mathematical parameter values. A series of user-adjustable parametric surface functions are defined that correspond to each surface texture parameter. Chebyshev polynomials are used to perform high accuracy numerical calculations of surface texture parameters for a

selection of complex polynomial surface functions. The mathematical reference parameter values are calculated for a series of fifteen predefined surfaces and ten parametric surfaces to assess the performance of the software under test for a given dataset resolution.

Two methods of surface texture parameter software performance assessment are presented that utilise the mathematically-traceable surface-parameter reference pairs. Extrapolation of parameter values calculated from a series of increasing resolution datasets is performed to account for the effect of discretisation error on software-obtained results. In addition, an assessment of the number of significant figures of the software-obtained values that agree with the reference values is used as a simple performance metric that enables easy comparison between different third-party software applications, for a given dataset resolution. An assessment of the sampling methods used to create discrete datasets of a mathematical surface function for use with numerical third-party software is performed.

Table of contents

List of figures

List of tables

List of abbreviations

NPL	National Physical Laboratory
NIST	National Institute of Standards and Technology
PTB	Physikalisch-Technische Bundesanstalt
NMI	National Metrology Institute
ISO	International Organisation for Standardisation
SI	International System of units
VIM	International Vocabulary of Metrology
CAS	Computer Algebra System

Chapter 1

Introduction

1.1 Background

High precision manufacturing is dependent on metrology to ensure parts are being manufactured within specified tolerances. The ability to precisely measure a manufactured part ensures repeatability and doing so at the micro-scale allows a whole new level of functional performance characterisation to be designed with confidence [2, 3]. Surface texture can have a profound effect on the function of a part, as it is the interactions between a surface and its environment that determine attributes such as friction and wear, which directly contribute to the overall lifetime of a component [4, 5]. These factors can strongly influence the efficiency of a part, affecting total energy consumption and resource usage, both of which are becoming increasingly important in today's climate [6, 7].

The need to measure surfaces comes down to an increased need for tight tolerances on component designs. As manufactured parts get more complex, the effects of their micro-scale attributes play a more important role in their functional properties. The surface of a component serves as the point of interaction between itself, other components and the environment [8]. The topography of a surface is defined as the entire surface structure of a part, including all surface features as a continuum of spatial wavelengths. Surface texture

describes the features on the surface that remain once the shape of the part has been removed from the surface topography, focussing on a window of spatial frequencies of interest. The texture of a surface can influence material properties such as friction and durability [5], and consequently there is an increased need for greater surface control. Perhaps unsurprisingly, this control is of great interest to the automotive industry and other heavy machinery parts manufacturers [9, 10], but also it poses great potential application in the pharmaceutical industry, as different molecule surface textures can cause different absorption rates in the body [11, 12].

The characterisation of surface texture is a necessary step in successfully understanding how the topography of a surface affects tribological properties, and is broadly split into two processes: filtration, to separate and extract spatial frequency components of interest; and numerical parameters, to describe the surface. Surface texture parameters are obtained through a series of mathematical operations that are performed on the surface topography measurement result and serve as numerical descriptors of the surface that impart information about the surface height variation [13, 1]. In addition to enabling functional part performance to be related to surface topography using quantitative measures, the use of simple numerical values to describe a surface's properties facilitates easy collaboration throughout industry and ensures meaningful surface texture tolerances can be made on part specifications, further promoting precision manufacturing on an international scale.

As with any aspect of metrology, it is important that surface texture parameters are internationally standardised to guarantee all collaborators are working to the same values. Software measurement standards fulfil this need, which revolve around using discrete numerical software developed by national metrology institutes (NMIs) to calculate surface texture parameter values for a given input surface [14, 15]. This software, in combination with reference surface height datasets, enables developers of parameter calculation software to assess their performance and relate back to the internationally accepted standards.

Previous work (discussed in section 2.4.2), and work presented in chapter 3 has highlighted differences in the surface texture parameter values obtained by different software measurement standards when given the same input surface measurement dataset. The causes of these differences are discussed further in chapter 3, with a primary reason for the differences between software measurement standards being due to different discrete-based algorithms used to implement the analytical definitions given in the ISO specification standards. With multiple software measurement standards giving different parameter values, differing surface texture parameter calculation software packages are at risk of being validated against different reference values. This can lead to inconsistencies when collaborating institutions or companies attempt to produce high precision parts to surface texture parameter tolerances set according to differing software.

1.2 Proposed work

The aim of this body of work is to develop a new method for the validation of surface texture parameter calculation software. Current practices rely on discrete numerical software algorithms, which can be subject to different methods of standard interpretation and software implementation during development. These developmental variations may lead to differences in final output surface texture parameter values between NMIs, which can cause errors across collaborations.

A new approach to the validation of surface texture parameter calculation is proposed that relies on mathematical definitions instead of software algorithms. By defining a fully mathematical framework for the validation of surface texture parameters, a traceable solution can be realised. Using this approach, third-parties can compare their software to mathematical outputs and perform performance assessments of increased accuracy, giving users increased confidence in the software results. Such assessments can be performed without the need for

multiple instances of NMI reference software, and will be fully comparable as all partners work to the same mathematical reference.

The proposed work is split into the following objectives:

1. Identify the reasons for differing results between NMI reference software by performing a comparison of surface texture parameter value outputs, given identical surface dataset inputs. This motivates the need for a new method for the validation of surface texture parameter calculation software.

2. Develop a method for creating mathematically defined surfaces that can be used as reference input surfaces for obtaining mathematical surface texture parameter values.

3. Derive mathematical solutions for areal functional surface texture parameters based on a mathematically defined material ratio curve.

4. Derive mathematical solutions for areal field surface texture parameters.

5. Develop methods to assess the performance of surface texture parameter calculation software by addressing the relationship between continuous, mathematically-defined reference values and discrete dataset-based software.

The majority of the work performed as part of this thesis will focus on developing mathematical references for areal surface texture parameters, and not profile surface texture parameters. Focussing on areal surface texture parameters ensures the work in this thesis is at the cutting-edge of research and remains of value to the field for an extended period of time. In addition, developing this framework for the areal case will allow for easier adaptation to the profile case due to its reduced complexity, rather than vice versa. This approach encourages a wider impact of the resulting work.

1.3 Relevance to industry

To motivate this work and highlight its relevance to industry, an international on-line survey was conducted to identify the use of surface texture parameters. In 1999, a survey was conducted by De Chiffre to identify the surface texture parameters used in industry [16]. The survey obtained a total of 284 responses, shedding light on the use of profile parameters from current ISO specification standards of the time: ISO 4287 [13], ISO 12085 [17], and ISO 13565-2 [18].

In the twenty years since the publication of the 1999 survey, much has changed, with the most important advancement being the introduction of areal surface texture parameters [19, 20], as described in ISO 25178-2 [1]. In addition, commercial software packages have been released for the calculation of surface texture parameters and it is, therefore, expected that industry is starting to embrace areal surface texture characterisation. Optical instruments have also become increasingly popular in industry, which are often areal in nature [21, 22].

These factors present the need for a new surface texture parameter survey, to identify the new usage patterns of surface texture parameters that are born from the new instruments and software that are becoming more popular in industry. Additionally, a new surface texture parameter survey gives insight into the uptake of the areal surface texture parameters of ISO 25178-2, since its publication in 2012. This insight helps to identify the proliferation of areal surface texture parameter usage in industry and highlight the importance of robust validation methods for surface texture parameter calculation software.

The survey obtained a total of 179 responses from industrial companies spread internationally across thirty-four countries and a variety of sectors. Full details of the survey are provided in appendix B. The results showed a relative increase in the use of virtually all parameters in comparison to 1999, for example the popular *Ra* parameter increased from 66% in 1999 to over 90%, showing a marked improvement in the uptake and importance of surface texture parameters in industry. It should be considered that this could be a consequence of

the exponential increase in computational power available to users since 1999, along with greater availability of third party surface texture parameter calculation software, enabling the calculation of many more parameters with relative ease; this scenario does not necessarily mean a greater understanding of the parameters used.

In addition, this survey gave insight into the adoption of areal surface texture parameters in industry, and resulted in a significant proportion, ~30% for areal field parameters, of participants indicating regular use of areal surface texture parameters. This relatively fast adoption rate indicates the value that areal parameters have in industry and the additional layer of information that they can deliver to users that the older profile parameters cannot. A deeper look into these adoption rates, however, showed that it was mainly the 'research institutions' and 'metrology and calibration' sectors that have shown adoption of the areal parameters, and that these new parameters are yet to be widely used by the other sectors in industry. Further education is required for the areal parameters to allow all sectors to engage with them.

The adoption of areal surface texture parameters showcases the need for accurate surface texture parameter calculation. Surface texture parameters are an important aspect of modern manufacturing processes, and as such it is crucial that the software packages used to calculate surface texture parameter values are well validated. Early adoption was expected for the academic and metrological sectors, and it is expected that this adoption will transfer to other sectors as collaborations and discussions take place over time. It is important, therefore, that reliable validation methods for the calculation of areal surface texture parameters are in place for when this happens.

1.4 Thesis outline

Chapter 2 will begin by giving an introduction into the field of surface metrology and the current state of the art, focussing on surface characterisation and its standardisation.

Chapter 3 expands upon this review by performing a comparison of the current reference software on offer from three major NMIs, highlighting their differences to provide motivation for the work performed.

Chapter 4 showcases the derivation of mathematical values for functional surface texture parameters using a material ratio curve as a foundation. Here, a material ratio curve is mathematically defined, and is used to derive functional surface texture parameters. Corresponding surface height datasets are sampled from the material ratio curve for use with third-party software, and comparisons are made between software-obtained and mathematically-obtained values.

Chapter 5 derives mathematical values for field surface texture parameters utilising mathematical surface function methods. Purely mathematical values are obtained for a variety of simple and parametric surfaces, and their results are compared to software-obtained values. Values are also obtained using numerical methods for a selection of complex surface functions to showcase the extension of this approach into numerical evaluations for scenarios that are too computationally intensive for current desktop computers to perform analytically.

Chapter 6 addresses the steps required to meaningfully compare the mathematically-obtained parameter values to software-obtained values in order to assess the performance of the software under test and deliver a quantitative performance metric to describe the quality of the software. The effect of discretisation is also discussed, and extrapolation methods are presented to showcase methods to account for discretisation error.

Finally, chapter 7 concludes the thesis by critically assessing the strengths and weaknesses of the work presented as a whole, and highlights opportunities for future work that can expand upon that which is introduced here to increase its scope and impact.

Chapter 2

The state of the art of surface texture characterisation

2.1 Measurements

The need for measurements has been a staple requirement of the manufacturing of parts for millennia - from the times of the pyramids to today's gravitational wave detectors. A measurement allows a property of an object to be referenced against a known 'unit' value of said property. This ensures collaborators are in agreement, and two people can reproduce an object with the same value of that property [23]. That object's measurement is then recorded as a certain number of that reference unit, for example, 'This pencil is 12 cm long'.

2.1.1 SI units

For measurements to be meaningful, there needs to exist an agreed and reproducible unit of measure. Enter: SI units. These were first introduced in France in 1960 and comprise a system of fully standardised, well defined units that ensure coherence between collaborating peers, institutions and countries. Fully named the International System of Units (or 'Système

International d'Unités'), the system features the definitions of seven base units, including the metre, kilogram and second [24]. For example, the second is defined by fixing the numerical value of 'the unperturbed ground-state hyperfine transition frequency of the caesium-133 atom' to 9192631770 in Hz, or s^{-1}. This definition can be read anywhere in the universe and, given access to the appropriate equipment, the necessary experiment can be performed to realise the value of the second. As of May 2019, the SI definitions have been updated to be defined in terms of fundamental constants, removing the need for physical reference artefacts. The final physical artefact to be made obsolete was the kilogram, which was a cylinder of platinum-iridium kept in a tightly controlled, sealed environment. Table 2.1 gives the current definitions for each SI unit. The redefinition of the SI units was done in such a way so as to fix the values of the fundamental constants at values that agree with their pre-existing values, obtained either via extrapolation from a physical artefact or from experimentation. In practice, this leads to no perceivable effect for the vast majority of people, and only those involved with experimentally realising these new definitions will experience a meaningful change.

Table 2.1 Full definitions of the SI units, as of 20th May 2019.

Name	Symbol	Measure	Definition
second	s	time	It is defined by taking the fixed numerical value of the Caesium frequency $\Delta\nu_{Cs}$, the unperturbed ground-state hyperfine transition frequency of the caesium-133 atom, to be 9 192 631 770 when expressed in the unit Hz, which is equal to s^{-1}.
metre	m	length	It is defined by taking the fixed numerical value of the speed of light in vacuum c to be 299 792 458 when expressed in the unit $m\,s^{-1}$, where the second is defined in terms of $\Delta\nu_{Cs}$.
kilogram	kg	mass	It is defined by taking the fixed numerical value of the Planck constant h to be $6.626\,070\,15 \times 10^{-34}$ when expressed in the unit J s, which is equal to $kg\,m^2\,s^{-1}$, where the meter and the second are defined in terms of c and $\Delta\nu_{Cs}$.
ampere	A	electric current	It is defined by taking the fixed numerical value of the elementary charge e to be $1.602\,176\,634 \times 10^{-19}$ when expressed in the unit coulomb, which is equal to A s, where the second is defined in terms of $\Delta\nu_{Cs}$.
kelvin	K	thermodynamic temperature	It is defined by taking the fixed numerical value of the Boltzmann constant k to be $1.380\,649 \times 10^{-23}$ when expressed in the unit $J\,K^{-1}$, which is equal to $kg\,m^2\,s^{-2}\,K^{-1}$, where the kilogram, meter and second are defined in terms of h, c and $\Delta\nu_{Cs}$.
mole	mol	amount of substance	One mole contains exactly $6.022\,140\,76 \times 10^{23}$ elementary entities. This number is the fixed numerical value of the Avogadro constant, NA, when expressed in the unit mol^{-1} and is called the Avogadro number. The amount of substance, symbol n, of a system is a measure of the number of specified elementary entities. An elementary entity may be an atom, a molecule, an ion, an electron, any other particle or specified group of particles.
candela	cd	luminous intensity	It is defined by taking the fixed numerical value of the luminous efficacy of monochromatic radiation of frequency 540×10^{12} Hz, K_{cd}, to be 683 when expressed in the unit $lm\,W^{-1}$, which is equal to $cd\,sr\,W^{-1}$, or $cd\,sr\,kg^{-1}\,m^{-2}\,s^3$, where the kilogram, meter and second are defined in terms of h, c and $\Delta\nu_{Cs}$.

2.1.2 Traceability

In reality, it is not feasible for every person who wishes to perform a measurement to recreate the experiments defined by the 'Système International d'Unités' to obtain the required unit value. For example, it is not always appropriate to obtain the 'distance travelled by light in vacuum in $1/299792458$ seconds'. Instead, it makes more sense to have a well-equipped laboratory perform the experiment and provide the units in a more accessible form.

However, how can one be sure that any piece of measuring equipment provided by someone else truly is accurate? Any number of approximations, rounding errors and uncertainties may have been introduced on the unknown number of steps taken from the point of experiment through to the end-user, reducing the closeness of agreement between a measured value and a true value, and hence reducing the measured value's accuracy [25]. With this in mind, what worth would results obtained from such a piece of equipment have if there is no certainty about its results?

Such a question reveals the importance of traceability in measurement. Traceability is defined in the international vocabulary of metrology (VIM) as 'the property of a measurement result relating the result to a stated metrological reference through an unbroken chain of documented calibrations of a measuring system or comparisons, each contributing to the stated measurement uncertainty' [25]. This unbroken chain of calibrations effectively creates a link between the measuring device in question and the SI definition, with each step along the way documenting any added uncertainty, thereby ensuring the end result gives a true representation of that measurement's value. A schematic representation of the traceability chain is given in figure 2.1 that showcases this chain from SI definition to end-use workpiece. For the case of surface texture software, the endpoint in the traceability chain is the surface texture parameter value. For this to be traceable, two routes of traceability are required. First, the measurement result must be obtained using a calibrated measurement instrument, traceable to the metre, providing information about the uncertainty of that

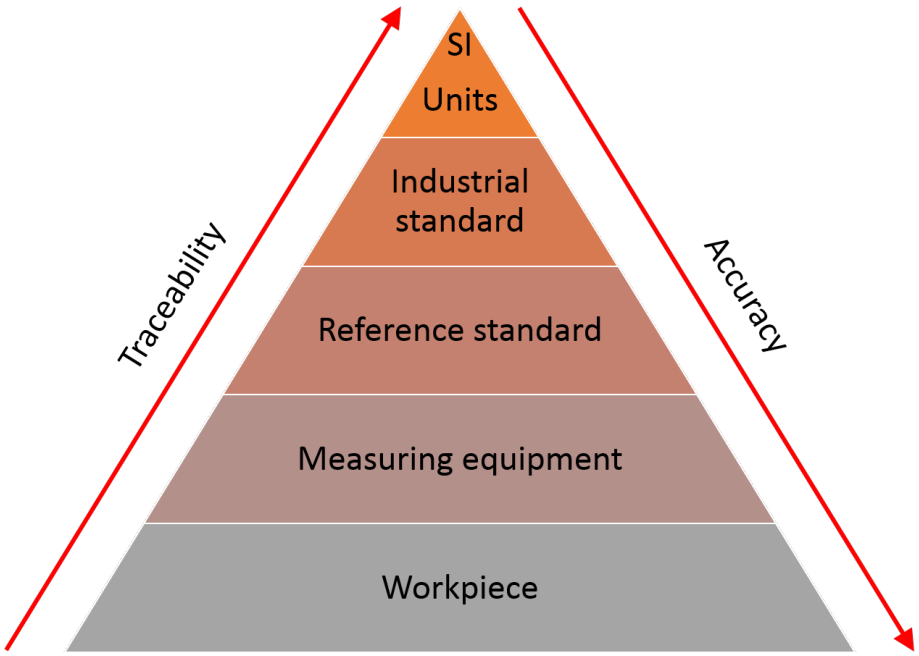


Fig. 2.1 Schematic representation of a traceability chain. A small number of international standards disseminate the SI definitions to larger numbers of increasingly inaccurate measurement artefacts and workpieces.

measurement. Second, the surface texture parameter calculation software must acknowledge its contribution to uncertainty in relation to the ISO specification standard definitions of the surface texture parameters, and also to the definition of the metre (for example, the number of significant figures to which length values are stored will impact the uncertainty of any resulting length values). Without a known definition for a parameter value (which would require a mathematical parameter definition and a mathematical surface definition), the contribution to uncertainty of the software cannot be properly understood. In addition to these factors, measurement dataset uncertainties are required to be propagated through the parameter calculation algorithms utilised within the software. This is a complex task is yet to be fully performed for all parameter definitions, however it is possible for some simpler parameters [26, 27].

2.2 Surface measurements

There are two different approaches to surface measurement - profile and areal [28, 29]. Profile measurements are the older and simpler of the two, comprising of a series of height measurements taken in a line across a surface. Areal measurements extend the profile case to two dimensions to cover an area. Areal measurements obtain many more data points than profile measurements and provide much more information about the surface being measured, however, this comes at the cost of much larger data files requiring more computational power to process. With modern, powerful computers, the computational costs of areal measurements are more easily addressed, and the benefits of the information density provided by areal measurements outweigh the negatives.

2.2.1 Types of profile measurement instrument

It comes as no surprise to find there are several different profile measurement instruments available, each with their own advantages and disadvantages. Put simply, they can be split up into two types - contact and non-contact.

Contact instruments, or stylus instruments, rely on a mechanical interaction with the surface to be measured to obtain height information. The instrument comprises of a small-tipped stylus that is 'dragged' across the surface and connected to a transducer which converts its vertical displacement into an electrical signal. The interaction of a contact stylus with the surface is well understood, allowing for simple, robust models. Further information about the workings of a stylus instrument can be found in [28], and a schematic is shown in Fig. 2.2.

Although simple, the stylus instrument is not without its disadvantages. In order to obtain a measurement, the stylus tip must be in contact with the surface. If the force of the tip is too light, the tip can skip over the surface as it is moved across and miss measurements. If the force is too heavy, the tip can dig into the surface and damage it. Because of this, care must be given for each surface to determine an appropriate tip force. The resolution of the

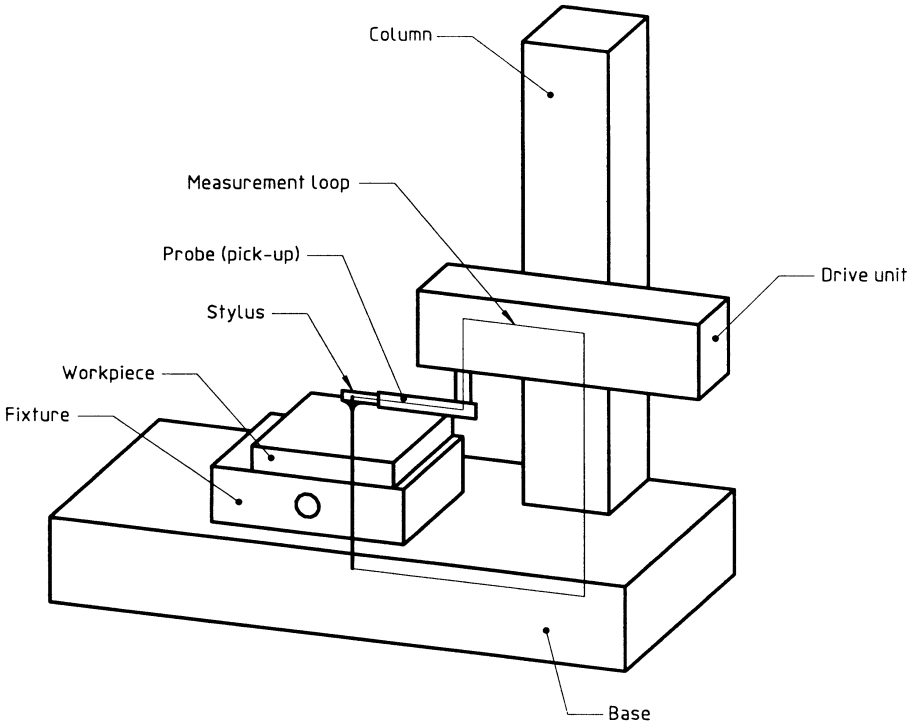


Fig. 2.2 Schematic of a stylus instrument [28]

detectable surface features is also limited by the size of the stylus tip, as anything within two peaks spaced closer together than the width of the tip is at risk of not being detected by the transducer.

Non-contact instruments have the advantage of avoiding mechanical interaction with the surface throughout the measurement process, instead relying on optical processes to image the surface. The performance of optical instruments is often determined by their optical properties, such as numerical aperture, optical resolution and spot size [8]. There is an ever-growing list of optical measurement techniques, each with its own strengths and weaknesses. A list of standardised optical instrument techniques is given by ISO, the International Standards Organisation [30]. Listed below are a selection of optical scanning techniques, which scan a light spot across a surface to take measurement points, similar to the process of the stylus instrument, making them more suitable to profile measurements than areal. These techniques are:

- *Triangulation* - Laser light is projected onto the surface. The scattered light is picked up by a position-sensitive detector and the relative distance to the surface is measured. A change in surface topography leads to a change in the spot location on the detector. Triangulation is much faster and less expensive than the other two examples but has a lower resolution [8].

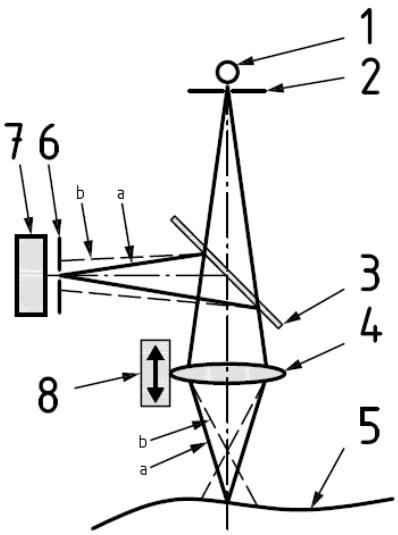
- *Confocal chromatic probe* - A microscope set-up with two pinhole apertures: at the detector and the light source. With this set-up, points in focus are much brighter than those not in focus. An integrated chromatic objective is included and the surface height at a single point is obtained by decoding the chromatic dispersion of the light [31]. A schematic is given in Fig. 2.3.

- *Point autofocus profiling* - A laser beam is reflected from the surface onto an autofocus sensor. This then feeds back to a z-displacement mechanism which adjusts accordingly until the reflected beam is in focus. This z-displacement is used to obtain the height information of the surface point [32]. A schematic is given in Fig. 2.4.

2.2.2 Types of areal measurement instrument

When progressing to an areal measurement, the surface needs to be measured across two dimensions to cover an area. A simple way to do this is to perform a series of profile measurements, each one shifted across perpendicular to the direction of travel. By this logic, all of the previously mentioned profile measurement instruments can be used to obtain an areal surface measurement. This approach, however, would take hours. A much faster approach would be to use optical techniques to image the entire area at once, reducing the measurement time to minutes or even seconds; this is the idea behind most areal measurement instruments [30].

Examples of areal measurement instruments include the following:



Key

1	light source	5	workpiece
2	light source pinhole	6	discrimination pinhole
3	semi-transparent mirror	7	photo-detector
4	achromatic objective lens	8	vertical displacement device

a Beam focused on workpiece.

b Defocused beam.

Fig. 2.3 Schematic of a focus-sensing confocal instrument [31].

Key

1 autofocus sensor

2 image lens

3 light source

4 vertical positioning sensor

5 workpiece

6 laser beam

7 half mirror

8 autofocus mechanism

9 objective

10 XY stage

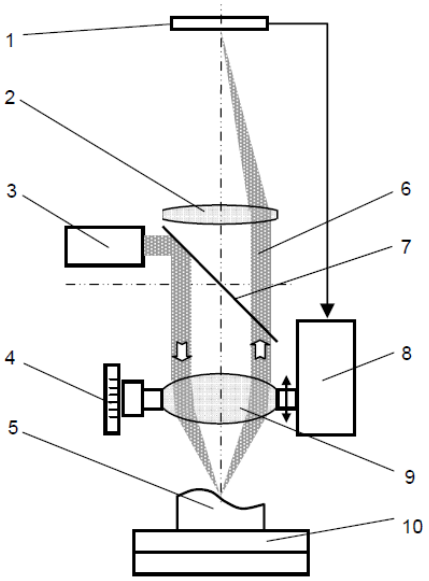


Fig. 2.4 Schematic of a point autofocus instrument [32].

- *Phase shifting interferometry* - Deviations in interference patterns from a reference are used to reconstruct the surface being imaged. Interference patterns related by known phase shifts, commonly $2\pi/3$, are used to aid in the reconstruction of the surface [33].

- *Coherence scanning interferometry* - Utilises the short coherence length of white light by moving through the z direction and recording the areas of the image that have zero optical path difference, as these will show the highest amplitude interference patterns [34].

- *Focus variation* - Microscope with a fixed focal length in scanned through the z direction. Areas of the image that are the sharpest are at the focal length of the imaging system and their height values can be determined. Focus variation has the benefit of being able to obtain true colour images of the surface [35].

- *Confocal* - Similar to the chromatic probe profile scanning instrument, but without any chromatic elements. An areal image of the surface is obtained, with the areas in focus appearing brightest. Like the focus variation instrument, focal length and z-scanning position is used to determine surface height [36].

Each of these measurement instruments, both profile and areal, obtain height information from the surface using a different technique. Work by Thompson et al. has demonstrated that different instruments measuring the same surface can obtain different resulting height values, which leads to different surface texture parameter values [37]. These differences can be seen in the amount of measurement noise, or high frequency components, obtained by the instrument, as well as around individual surface features, such as steep slopes. The obtainable frequency bandwidths of the measurement instruments can play a significant role in the measured height values in the dataset, and therefore in the resulting surface texture parameter values. Some aspects of this, such as the presence of small-scale high frequency noise components in the measured surface, can be accounted for and removed from the

surface texture analysis by utilising appropriate filtration techniques (discussed further in sections 2.3.3 and 2.3.4).

2.2.3 Performing a profile surface measurement

Without a well-defined procedure, repeat measurements may give substantially different results, and the maximum effectiveness of the measurements may not be fully realised. An ISO document is available that defines a selection of default procedures in [38], and a guide detailing good practice for performing profile measurements is given in [39]. The standards and guides for profile measurements focus on stylus instrument measurements primarily, however it is possible to adapt the relevant sections to apply them to optical techniques.

One of the main choices to make when performing a surface profile measurement is the location and orientation of said measurement. Different manufacturing processes produce different surface finishes, and these may have anisotropies. An example of this can be seen in Fig. 2.5, which shows the surface texture of a polished surface. Here, there is a clear uniformity of the surface running from top-left to bottom-right. If a measurement were to be take in this same orientation, very few undulations would be recorded, and little would be learned about the surface texture. This direction of dominant surface finish is known as the lay of the surface, and all measurements should be taken perpendicular to it in order to maximise the amount of information obtainable from a measurement [38].

Some surfaces may show other, more localised defects, such as weld-spots, scratches and divots. These features are poorly accounted for in the characterisation of overall surface texture and should be avoided by choosing a measurement location free from such defects.

2.2.4 Performing an areal surface measurement

Areal surface measurements obtain surface height information over two dimensions. Consequently, some of the procedures required by a profile measurement are no longer necessary.

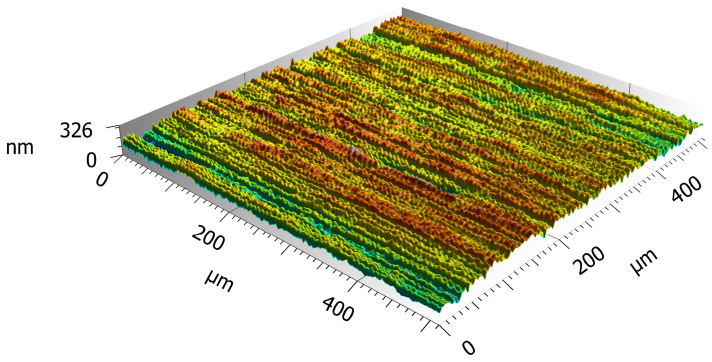


Fig. 2.5 Computer generated representation of the areal surface texture of a polished surface.

For example, the measurement no longer needs to be carried out perpendicular to the lay of the surface, as the lay will be captured within the measured area regardless. That said, to minimise the differences in analysis results obtained by areal measurement methods to those obtained by profile methods, it is recommended that the profile procedures are followed as best as possible [40]. For example, the orientation of a rectangular areal measurement should be aligned with the lay of the surface. Following such practices allows for profiles to be extracted from the areal measurement at later points in time and enables profile surface texture characterisation to be performed on measurement data that conforms to profile measurement good practice.

2.3 Characterisation of surface texture

A profile or areal measurement of a surface will yield an output file of point values, each point defining the height value relative to some predefined nominal surface. This output itself

is not of much use, and work must be done to extract useful information from it. This is done with surface texture parameters.

Surface texture parameters give the measured data of a surface a single, quantitative value. This simplifies the description of a surface, using a selection of numerical values instead of a long, worded description of its features, and lends itself to easier comparisons with other surfaces and easier part design and tolerancing. There are a great number of parameters, each defined differently to convey a different surface texture property. However, this leads to a phenomenon known as 'parameter rash' [41], where there are so many that it can be unclear as to which parameter is most suitable for any particular case, and can lead to users of surface texture parameters simply stating as many as they can in the hope to cover all bases. Nonetheless, parameters are still a hugely useful tool in the surface texture industry, and when used properly can deliver a good deal of useful information about a surface in a very succinct, standardised manner.

2.3.1 Profile surface texture parameters

Before delving any further into surface parameters, it will first be useful to define some key terms. Each of these is taken from the ISO document dedicated to definitions related to profile surface texture [13]:

- *Evaluation length* - Length in the direction of the x-axis used for assessing the profile under evaluation. The evaluation length is typically shorter than the total traverse length of the measuring instrument to account for over-travel at the start and end of the measurement.

- *Sampling length* - Length in the direction of the x-axis used for identifying the irregularities characterising the profile under evaluation. One or more sampling lengths may make up an evaluation length.

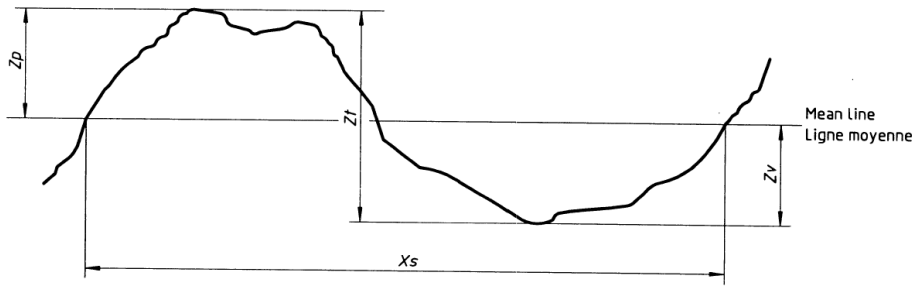


Fig. 2.6 An example profile element, comprised of a profile peak and the adjacent profile valley [13]. Here, *Zt* is the sum of the height of the peak *Zp* and the depth of the valley *Zp* within the profile element, and *Xs* is the width of the profile element.

- *Profile peak* - An outwardly directed (from material to surrounding medium) portion of the assessed profile connecting two adjacent points of the intersection of the profile with the *x*-axis.

- *Profile valley* - And inwardly directed (from material to surrounding medium) portion of the assessed profile connecting two adjacent points of the intersection of the assessed profile with the *x*-axis.

- *Profile element* - Profile peak and adjacent profile valley (see Fig. 2.6).

There are 14 different profile parameters listed in ISO 4287, each with their own definition [13]. Most profile surface texture parameters are calculated for each sampling length, unless otherwise stated in their definition, and then averaged over the evaluation length. The relationships between the sampling, evaluation and traverse lengths are shown in figure 2.7. Broadly speaking, they fall into one of the following categories:

- *Amplitude parameters (peak and valley)* - These are parameters that give an absolute definition of the heights of peaks and valleys on the surface. For example, the largest profile valley depth, or the sum of the largest peak and largest valley within a sampling length (see Fig. 2.8).

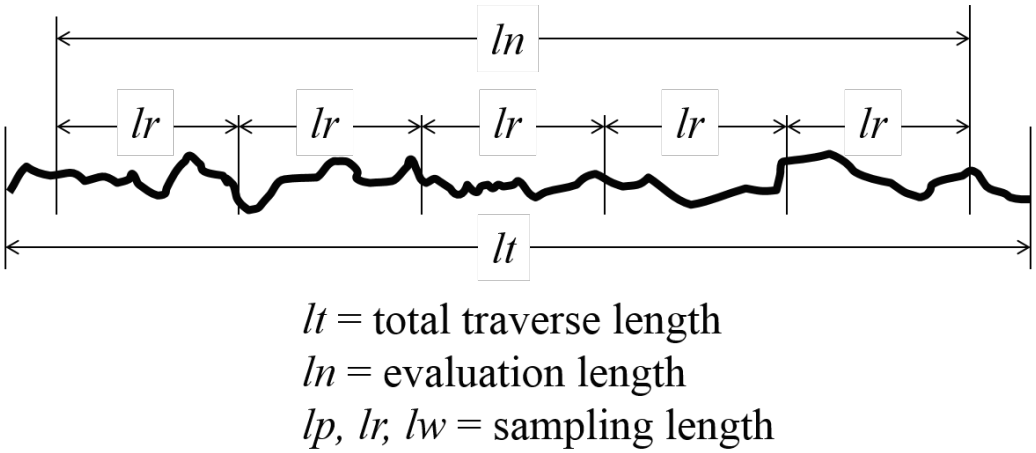


Fig. 2.7 Relationship between total traverse length, evaluation length, and sampling length.

- *Amplitude parameters (average of ordinates)* - These parameters give values based on different ways to average the values of each measurement point. For example, the root mean square (RMS) value of the ordinate values within a sampling length.
- *Spacing parameters* - These give a numerical description of how spaced apart features are. For example, the mean value of profile element widths within a sampling length (see Fig. 2.9).
- *Hybrid parameters* - These are formed from a combination of other parameters. For example, the RMS value of ordinate slopes within a sampling length.
- *Curves and related parameters* - These parameters are related to the use of graphical line representations of the cumulative data, such as the material ratio curve, which displays the relative amount of points that are above a certain height. These are assessed across the whole evaluation length.

2.3.2 Areal surface texture parameters

In 2012, areal surface texture parameters were published by ISO [1], expanding surface texture analysis to two-dimensional measurements. Created as an evolution from the profile

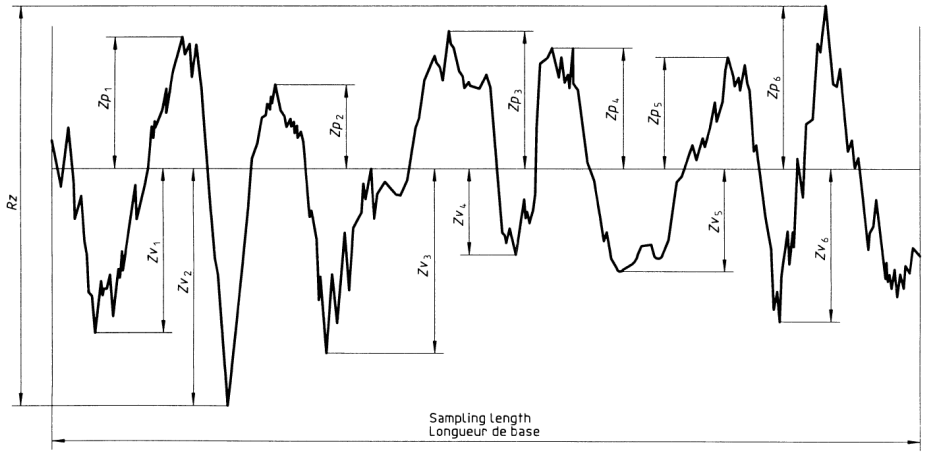


Fig. 2.8 An example roughness profile displaying the *Rp*, *Rv* and *Rz* parameter definitions [13]. Here, *Rp* would be the value of Zp_6, *Rv* would be the value of Zv_2, and *Rz* would be the sum of Rp_6 and Rv_2 together

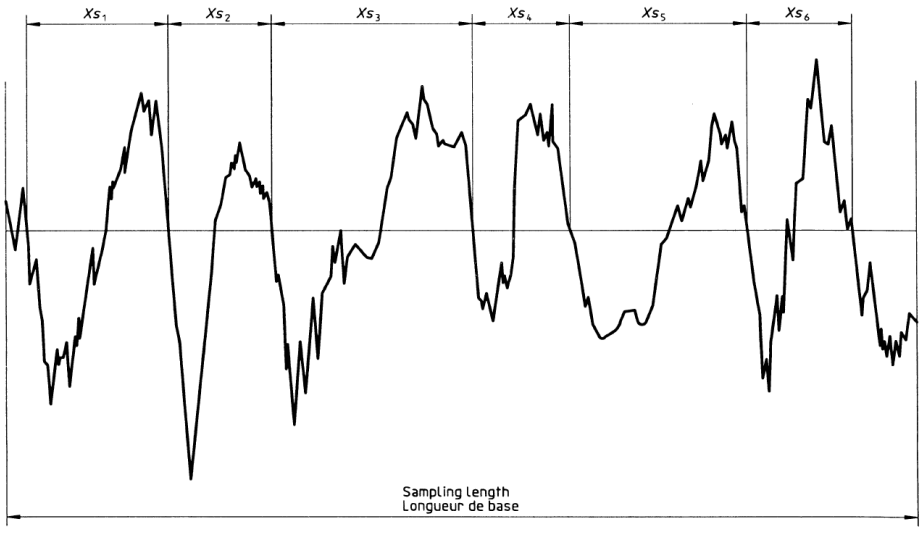


Fig. 2.9 An example profile displaying the calculation of the *XSm* spacing parameter [13]. The mean value of the widths of each of the profile elements *Xs* within the sampling line produces the parameter value.

case, many of the areal surface texture parameters have profile measurement analogues, and are essentially the same parameter, only operating in two dimensions instead of one. However, due to the added information available when using two dimensions, new categories of surface parameter have been developed. The areal surface texture parameters described in ISO 25178-2 fall into these categories:

- *Height parameters* - Share close similarity with the profile amplitude parameters. These are calculated based on the height of all points within the definition area.
- *Spatial parameters* - Analysis of the spatial information of the measured surface utilising the autocorrelation function for the surface.
- *Hybrid parameters* - Similar to the profile case, these are formed using a combination of other categories, for example, RMS of the gradient of the surface.
- *Functional parameters* - These are similar to the curves and related parameters from the profile case, assessing ratios of the surface that fall above/below a specified height criterion. This category also includes volumetric analysis parameters, which calculate the volume of void regions on the surface.
- *Feature parameters* - A toolbox of pattern recognition techniques is used to identify specific surface features, such as saddle points, hills and course lines. Feature characterisation utilises advanced feature identification and segmentation techniques such as Wolf pruning [42], and as such have no profile equivalent.

With the exception of the height parameters, all of these categories of surface texture parameters require additional interpretation of the surface topography beyond just the height values on the surface in order for surface texture parameter software to obtain the parameter value. For example, spatial parameters require the calculation of the autocorrelation function of the surface, and functional parameters require a continuous interpretation of

the height distributions across the surface. These instances of additional processing often rely on some form of interpolation of the discrete heights in the dataset, in order to obtain a continuous representation that is used to conform to the continuous definitions of the parameters given in the ISO specification standards. By validating surface texture parameter software against continuously-defined reference surfaces and values, their specific interpolation implementations can be better assessed than if compared to an alternative interpolation technique.

2.3.3 Profile filtering

Surfaces have multiple scales of structure, each produced by different processes and occupying different bands of spatial frequencies, and are often called multi-scale surfaces [43].

The largest of these is form, which is the overall shape of the object being measured. For example, if the measured surface was a curved cylinder face, the form would be the shape of that curved edge. The form component of a measurement is not related to surface texture, and should be removed [28], for example using a best fit least-squares method.

What remains are two main orders of structure, termed 'waviness' and 'roughness' profiles. The waviness profile corresponds to longer wavelength features, typically showing as undulations or 'waves' within the measured profile. The roughness profile features have a shorter wavelength, and typically show much more variation [44]. As these different orders of structure relay different information about the surface texture, the next logical step is to separate them from each other so that they can be analysed independently.

Separating the roughness and waviness profiles is achieved by performing a filtering operation on the measured profile, blocking out unwanted spatial frequencies and letting desired spatial frequencies pass. For this to be possible, cut-off frequencies, values at which high-pass and low-pass filters attenuate the amplitude of a signal, must be defined for the

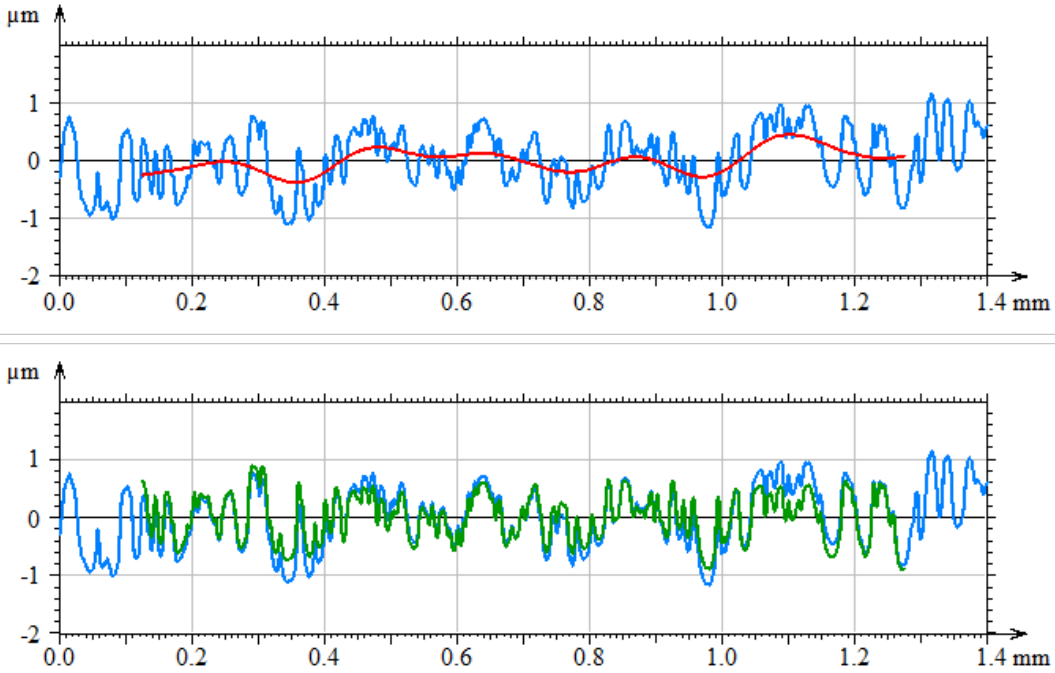


Fig. 2.10 Example profile (*blue*) with the waviness (*top, red*) and roughness (*bottom, green*) profile components separated. Separation of the waviness and roughness profiles was performed using a linear Gaussian filter with $\lambda c = 0.25$ mm.

filter. The first of these is the λs cut-off, which filters any very high wavelength features which do not count as roughness and are considered noise. This would be performed with a low-pass filter. The next is the λf cut-off; used to remove any long wavelength features which do not constitute as waviness and that have not already been removed by the form removal operation. This would be achieved with a high-pass filter. Finally, the λc cut-off, termed the cut-off length, is used to separate roughness from waviness. An example of a profile measurement with the roughness and waviness profiles separated is shown in figure 2.10

ISO 4287 states that the filtration operation should be performed by a linear Gaussian filter, defined in ISO 16610-21 [45]. This leads to more definitions for the resulting profiles [13]:

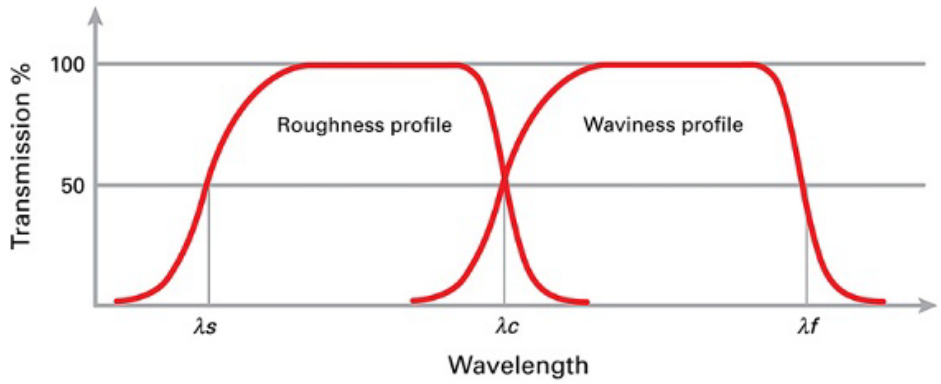


Fig. 2.11 Transmission characteristics profile filters, showing the separation into roughness and waviness profiles [8].

- *Primary profile* - Total profile after application of the short wavelength filter cut-off length, λs, and removal of nominal form [38].
- *Roughness profile* - Profile derived from the primary profile by suppressing the long-wave component using the profile filter cut-off length λc.
- *Waviness profile* - Profile derived from the primary profile by suppressing the long-wave component using the profile filter cut-off length λf, and suppressing the short-wave component using the profile filter cut-off length λc.

These three profiles form the basis for the evaluation of the primary, roughness and waviness profile parameters. A visual representation of the transmission characteristics for these filters is given in Fig. 2.11. These three sets of parameters all share the same parameter definitions. They are differentiated by their labelling, where the first letter of the parameter label is P, R or W, denoting a primary, roughness or waviness parameter, respectively. Following this, the remaining letters of the parameter label dictate the parameter type. For example, '*Ra*' is the arithmetical mean deviation of the roughness profile, *Wv* is the maximum waviness profile valley depth, and *Psk* is the skewness of the primary profile. Full definitions are given in ISO 4287.

The choice of value for λc is no trivial task, as a different value of λc can lead to very different values of surface texture parameters for roughness and waviness profiles. ISO 4288 states that the default cut-off wavelength, λc, should be chosen such that it is equal to the roughness profile sampling length, which is given a default value of one fifth of the evaluation length.

2.3.4 Areal filtering

For the areal case, the ISO documents no longer specify different parameters depending on the scale of focus, as is the case for profile analysis with *P*-, *W*- and *R*- parameters. Instead, only one set of parameters is defined, usually prefixed with an *S* [1], and it is up to the user to accompany it with the necessary scale-limiting details [29].

As with the profile case, there are three ways to scale limit the measured surface. These include the S-filter, which removes small scale components from the surface; the L-filter, which removes large scale components from the surface; and the F-operator, which removes form from the surface. These operations are performed in a similar manner to the profile case, albeit in two dimensions. For example, filtration can be performed using a linear areal Gaussian filter [46, 47].

The action of using these operations on the surface leads to the production of one of three defined surfaces [1]:

- *Primary surface* - Surface obtained after application of an S-filter to the measured surface. This is different to the definition of the primary profile, which requires both small-scale component removal and form removal.
- *S-F surface* - Surface obtained from the primary surface after removing form using an F-operator.

- *S-L surface* - Surface obtained from the S-F surface after application of the L-filter to remove large scale components.

A schematic detailing the areal filtration process is given in figure 2.12.

For both areal and profile filtering, recommended filter cut-off and nesting index values are given in the ISO specification standards [38, 40], and should be chosen to best match each individual surface measurement in order to obtain scale-limited surfaces of interest. The choice of filter can have a significant effect on the resulting dataset, which will in turn lead to a significant effect on calculated parameter values. An awareness of the frequency bandwidth of the resulting surface dataset is required in order to provide a full description of each operation performed on a surface measurement, in order to obtain parameter values from software that are traceable.

2.4 Software measurement standards

As previously mentioned, software exists that receives surface measurement data as input and performs the filtration and calculation steps required to obtain surface texture parameter values. However, the use of computer algorithms comes coupled with accuracy sacrifices due to adapting mathematics for a discrete system. This is especially prominent for commercial software, which often sacrifice accuracy, following customer demands for speed, in the fight to be best on the market. This can introduce uncertainty into the output parameter values that may be significant for high precision applications where tight tolerancing of the surface texture is required. The question must be asked: How can one be sure the results obtained by software used to calculate the surface texture parameters is accurate?

A solution to this problem comes in the form of software measurement standards. These are ISO defined guidelines that specify a collection references against which surface texture

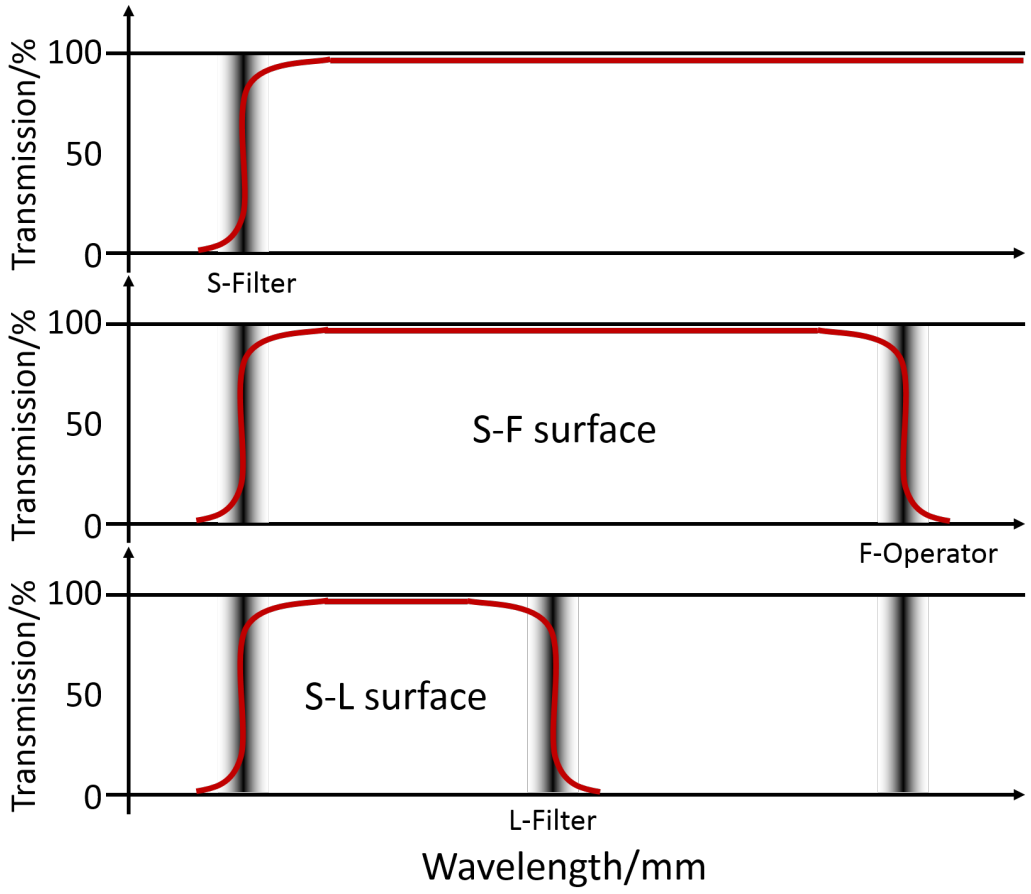


Fig. 2.12 Transmission characteristics for areal filters, showing the separation into S-F and S-L surfaces. Different choices of filtration and form removal operations can lead to different transmission characteristics.

parameter calculation software can be compared. The software measurement standards come in two types; F1 and F2 [14].

Type F1 standards are reference datasets that depict a digital representation of a primary profile. These can come in a variety of profile forms, from measurements of real surfaces such as a ground or milled surface, to simulated profiles such a saw-teeth and sinusoids. F1 datasets can be fed into third-party surface texture parameter calculation software and the obtained parameter values compared to certified results. For mathematically designed data, such as sinusoids, the parameter values can be calculated mathematically, without the need for software calculation. This makes the certified results against which users will compare as accurate as possible. ISO 5436-2 defines a type F1 reference data set to be a primary profile, not a total profile. This is due to the many operations taken between the total and primary profiles which are instrument specific. Such steps are difficult to standardise, and the primary profile is the first stage at which all subsequent operations are defined. These profile datasets are saved in the '.smd' format, defined in the standard [14].

Conversely, type F2 standards are reference software. These are traceable software applications that calculate surface texture parameters with a focus on accuracy over speed or additional features. These can then be used to compare against test software by inputting the same data set into both and comparing the results. For F1 datasets whose parameter values cannot be mathematically calculated, type F2 software can be used to produce certified results. This makes the accuracy of the reference software doubly important, as any inaccuracies in the F2 standard can transfer to inaccuracies in the F1 standard.

The areal analogues to these F1/2 standards are type S1 and type S2 standards. Here, S1 corresponds to the reference datasets, saved in either '.SDF' or '.X3P' formats [15, 48], and S2 corresponds to the reference software. The purpose of each of these areal software measurement standards is identical to the profile equivalents, so no further explanation is needed here.

2.4.1 NMI reference software

Such software standards are created by National Measurement Institutes (NMIs). These are nationally recognised bodies tasked with performing cutting-edge measurements and upholding measurement standards. NMIs often perform the experimental realisations of the SI definitions (such as the definition of the metre, mentioned earlier), and form the first link in the chain of traceability for all measurements performed in the country.

Some of the most well regarded software measurement standards for profile surface texture come from the three leading NMIs in the world: 'softgauges' produced by the National Physical Laboratory (NPL) in London, UK [49, 50]; 'SMATS' produced by the National Institute of Standards and Technology (NIST) in Maryland, US [51–53]; and 'RPTB' produced by Physikalisch-Technische Bundesanstalt (PTB) in Braunschweig, Germany [54, 55].

All three NMIs offer type F2 reference software which can be used compare parameter results for a common data set with a test software. In addition, NIST an NPL also offer type S2 areal reference software. PTB and NIST offer web-based software, whereas NPL provides downloadable java programs. Each have their benefits; a web-based program is accessible from any computer with an internet connection and ensures the user is always using the latest version of the software, whereas a downloadable program can be used at any time, regardless of web connectivity or server issues. NPL and NIST also offer type F/S1 reference data sets, which can be downloaded for use with test software.

2.4.2 Reference software comparisons

With multiple NMIs producing reference software, it becomes necessary to compare them with each other. The definitions laid out in the ISO standards are subject to interpretation, and there is more than one way of implementing the same thing through computational algorithms. Each of these different implementations come with their own approximations,

and hence can give conflicting results compared to other implementations. Not only that, but some ISO definitions, when analysed thoroughly, may present ambiguity. This can lead to further variations in the end result. A good example of this is shown in [56], where Leach and Harris reveal an ambiguity in the definition of profile element discrimination, which leads to the number of profile elements counted in a certain length to be unclear, producing different values for the *RSm* spacing parameter. This is just one, fairly severe, example of ambiguity in the parameter definitions. It is expected that there are more ambiguities elsewhere in the ISO steps required to go from a primary profile to a surface texture parameter value.

Comparison allows differences in the obtained parameter values to be discovered. This has the potential to highlight any further ambiguities in the ISO standards in the hope that they may be better defined. Baker et al. organised a comparison across sixteen laboratories internationally wherein two simulated surface profile datasets were distributed and each laboratory's surface texture software was used to calculate parameter values according to ISO 4287 [57]. The results of these comparisons show variations between software for both datasets for each of the seven parameters tested, with increased variations shown for the profile with lower frequency components. Little analysis into the causes of these variations was performed. This work focussed on only two profile datasets, both of which were comprised of a combination of sinusoids, indicating a limited variety in the types of profile surfaces assessed. Furthermore, while a λ_c cut off length was suggested for the laboratories to use, the specific type of filter was not. This could lead to laboratories using different filtration techniques and obtaining different roughness profiles, which would affect the parameter values.

Koenders et al. performed a similar experiment with seventeen laboratories and three profile surfaces of unknown composition, however, only nine of the participants provided results [58]. Roughness parameters were calculated from standards ISO 4287, DIN 4768 and ISO 13565-2. Different calculation methods for the *Rp* and *Rv* parameters were identified

from the results, highlighting a misinterpretation of the ISO specification standards and leading to average variations from the designated PTB reference value of approximately 7% for one of the profile datasets. Similar magnitude variation were seen for other parameters, for example an average variation of approximately 8% for the *Rvk* material ratio parameter, however, little explanation was given for these variations. Again, as only three profile datasets were used in this comparison, and specific details of these surfaces are unknown, the scope of the tests on the software is limited.

In 2009, an NPL report authored by Li et al. focussed more on the in-house software offerings of NPL, PTB and NIST, along with three commercial software packages [59]. This report went into further detail about the software packages, identifying some differences in software implementation for ISO 4287 parameters. For example, they highlighted that NPL uses interpolation of the input data points to create a continuous profile, whereas PTB and NIST work with a discrete profile. They performed their comparison with six type F1 reference data sets, one simulated simple sinusoid and five measured profiles of real surfaces, finding generally good agreement between the NMIs with the relative differences for the three NMI software packages being less than 0.5%, with the exception of height amplitude parameters for the NPL software. This work also highlighted the increased disagreement for obtained *RSm* values and the need for better definitions for this parameter, along with the *Rc* parameter. Again, this work focussed on a small selection of five surfaces, and while the simulated simple cosine profile was well defined, there was little information on the structure or frequency components of the five measured profiles. More recently, Paricio et al. performed a similar experiment, comparing the software on offer from NPL, PTB and NIST [60]. Their work took a greater focus on the user side of the software, commenting on the usability for the end-user. The comparison was performed using two data files created in-house, from rectilinear sections. The results showed good agreement for a small selection

of P and R parameters, with PTB's results differing slightly from the other two, showing a tendency to provide smaller values.

The comparisons presented in this section showcase a long-standing desire within the field to adequately assess to effectiveness of type F2 reference software and identify discrepancies between different implementations of reference software. Several comparisons have been made, with Li et al. performing the most detailed analysis to date, however there is a lack of investigation into the causes for the different obtained parameter values. Each of the previous comparisons have been performed using a small number of test profile datasets. A new comparison has been performed in chapter 3 that identifies common discrepancy trends in the obtained parameter values for three NMI software packages for seventeen test profile datasets, and identifies the potential causes.

While each of these comparisons has identified variations across type F2 reference software, no solutions have been proposed to develop methods that address these issues. The work in this thesis introduces the concept of developing mathematically defined references that work with the continuous parameter definitions given in the ISO specification standards to deliver reference surface/parameter value pairs that are free from the differences in implementation when dealing with reference software.

Chapter 3

An in-depth analysis of type F2 software measurement standards

The work presented in this chapter has been published in [61].

3.1 Introduction

The literature review performed in section 2.4.2 discussed previous work comparing software measurement standards. The review highlighted a gap in literature wherein comparisons were performed with a limited number of test profiles and with minimal analysis into the causes of the variations across software results through analysis of how each software package is implemented. This chapter attempts to address this gap, wherein a comparison is made with equal focus on parameter results and software implementation, enabling the opportunity for links to be made between the two. A larger number of test files are used for the comparison compared to previous work in order to provide a wider range of test cases to the software and reveal different software behaviours. Such work has the potential to highlight how subtle differences in implementation of type F2 software measurement standards affect the

calculated parameter values. This work also identifies parts of the ISO standards which require clearer definitions to avoid ambiguity.

This chapter focusses on type F2 profile reference software, instead of type S2 areal reference software. This is because profile reference software is simpler due to the reduced dimensional complexity and is more well-established in the field due to its maturity. Many areal parameters are very similar to profile parameters by definition, only given in two dimensions rather than one. Both are defined continuously, requiring software to implement discrete-based versions of the definitions, and rely on appropriate filtering to be performed in order to extract the scale limited surface of interest. Therefore, it is expected that the discrete implementation methods used for profile software will be similar to those for areal software, and so any processing steps that may lead to differences in profile parameter values are likely to also be seen in areal parameter values.

3.1.1 Objectives

The aim of this chapter is to perform an analysis of the F2 software measurement standards on offer from NPL, PTB and NIST. This analysis will comprise of both a comparison of the parameter values produced by each NMI software for a selection of test files, as well as an investigation into the differences in implementation of the standards by each of the NMI software. The combination of the two aspects has the potential to allow any observed implementation differences to explain differences in the parameter results. The work is split into the following objectives:

1. Produce flowcharts detailing the algorithm flow for each NMI reference software to identify differences in implementation.

2. Collate a range of test profile datasets, taken from the NPL and NIST type F1 reference dataset databases [50, 53], suitable for testing the reference software under a variety of conditions, including a range of profile lengths, amplitudes and frequency components.

3. Perform a comparison of the results of each NMI software package to identify differences in the output parameter values and identify the reasons for these differences in relation to both implementation of software and interpretation of the ISO 4287 specification standards.

3.2 Methodology

3.2.1 Implementation of software

The first task was to identify differences in the implementation of the standards by each NMI software package. This was done by reading the published work ([49, 54, 51, 52]), going through the help documentation available, and becoming familiar with the software interface.

3.2.1.1 Input file requirements

The first things to identify were the input requirements for the software. As expected, all three NMIs accept the .SMD file format, as defined in ISO 5436-2. On top of this, both NIST and PTB also accept .SDF files, which is the ISO defined file-type for areal software measurement standards [15]. This is a useful addition, as it allows profile and areal measurements to be processed for parameter calculation in the same way, saving time for users of the software that use both measurement types. Additionally, the NIST software accepts the .TXT format, a standard text file type, and PTB accepts the .PR filetype, a simplified version of the .SMD of their own creation.

All three NMIs assume the height value points within the input file are uniformly spaced along the X-axis. This is part of the .SMD definition in the standard, and allows for much simpler analysis of the file. On top of this, NPL assumes some initial processing has already been carried out. In line with the definitions in ISO 5436-2, the NPL software requires an input of a primary profile. This means the input file must already have form removed, and

must be λs filtered, so that any wavelengths shorter than those associated with roughness are removed. NIST and PTB, however, have no such restrictions, and will accept files with no prior processing undertaken.

3.2.1.2 Algorithm flow

The next step is to determine what options each software package gives. This includes identifying all choices available to the user and highlighting the different paths that a user could take through the software.

It was decided that a good way to visualise the routes taken through the software was to create flow diagrams. This allows each user option to be shown, along with flow lines describing the possible routes that can be taken. The flow diagrams for NPL, PTB and NIST are given in appendix A. What is immediately clear is the large difference in complexity between the three software packages.

The NPL software is by far the simplest of the three. Once the file is input, there are only two sections to the software: 'λc Filter' and 'Parameter Calculations', all that is required according to the ISO standards. The software has no user options; once the file is chosen, the process is autonomous, and the parameter values are output in the form of a .SMD file. Because of this, there is only one route through the software. Whilst this ensures all parameter values are obtained through the same method, it is not very flexible, should another filtration method be more suitable for a particular file. The filter used by the software is a linear Gaussian convolution, as defined in ISO 11562 [62]. Following this, the data is interpolated to form a continuous representation. This is achieved using a cubic spline. The software uses this continuous representation to evaluate the parameters using the exact integral forms given in the ISO standard. For the waviness and roughness profiles, a length of λc is removed from each end of the profile to avoid any end effects that may be caused by the filter application.

The PTB software is more complex than the NPL software. This is in part because of the addition of form removal and filtering with a λs cut-off length, adding two steps that precede those included in the NPL software. The order of these steps is also up the user, as is whether they are used at all; both are optional. The most dominant feature of the PTB flow diagram is the amount of filtration choice given to the user. The software gives the user three λs choices, two form removal choices, and ten combinations of filters with a λc cut-off; six to perform the main filtration, and four secondary options to obtain the curve parameters. These filters include Gaussian convolution, Gaussian regression, robust Gaussian and spline filters. This amount choice makes for a very flexible system but could add confusion for inexperienced users. Following the filtration, the choices become more limited. Users are able to select which parameter values are exported, along with certain inputs such as height/spacing discrimination values. The user can also choose which algorithm to calculate the curve parameters with, which are not included in the NPL software. Following these parameter choices, the results can be exported in .CSV file format and a .PDF report. During the filtration step of the software, the data is shown to the user visually, along with all obtained profiles. This is a useful addition, as seeing the data allows for a much better understanding of the data compared to a long dataset of raw values.

The NIST software at first looks complex, however, closer inspection reveals that the choices available to the user are relatively simple; there are instead more optional steps following the parameter calculations. First, the user has the option of not only choosing their own data file to run through the software, but also a reference data file from NIST's database. This makes it easier for users who are using a NIST type F1 data file with their own software, as they can run the same file through the NIST software without having to download the file first. The user interface for the NIST software is tabbed, allowing user control over which operations are performed. Additional sections include power spectral density calculation, autocorrelation/cross-correlation and bearing-area-curve production. Each of these come

with simple choices to the user, such as the profile to operate on, and which method to use. As with PTB, NIST includes optional form removal and filtering with a λs cut-off length. For filtering with a λc cut-off length, the choices are more limited than those given by PTB. The user can choose between Gaussian or 2RC filter methods. 2RC choices are between recursive and convolution methods, whereas Gaussian methods are between convolution, 'Fast Gaussian' and Fast Fourier Transform. For the Gaussian methods, the user can also choose whether to use a $\pm 0.5\lambda c$ or $\pm\lambda c$ Gaussian window, which dictates how large the cut-off regions are for the roughness and waviness profiles. Similar to PTB, the NIST software also shows the input data visually, albeit at a lower resolution.

3.2.1.3 Software choices for greatest comparability

With so many different routes through the software, it is important to establish a particular route for the comparison. This should be chosen such that each piece of software is performing in a similar manner. This will minimise the number of variables in the experiment and make it easier to investigate any differences that do arise.

As the NPL software has only one algorithm route available, it makes sense to choose options in the other two software packages to best match the steps taken by NPL. This includes the following:

1. The input data given is a primary profile. This means it must have form removed and be filtered with a λs cut-off length prior to input into the software.

2. The filtration with a λc cut-off length shall be performed by a Gaussian convolution, as described in ISO 11562.

3. The parameters shall be calculated with the definitions given in ISO 4287.

4. For parameters that require height/spacing discrimination, a height discrimination of 10% of Xz and spacing discrimination of 1% of the sampling length will be used.

5. If possible, no additional steps shall occur.

The closest routes through the NIST and PTB software are shown in red on the diagrams in appendix A. For both, it was possible to avoid extra steps such as form removal and λs filtering. The NIST software, being tabular in nature, allowed the selection of only the λc filtering and parameter calculation steps. The PTB software is not tabular, forcing users to go through material ratio calculations. However, while these added time to the processing of the data files, it did not affect the filtration or values of the other parameters.

Both software packages allowed the selection of a Gaussian convolution as the λc filter, matching that used by the NPL software. The NIST software also allowed the use of a $\pm\lambda c$ Gaussian window, leading to a cut-off length of λc from each end of the filtered profiles, matching that of NPL. Both NIST and PTB also allowed the selection of the same ISO defined parameters that NPL calculates. Combined, these options allow for pathways through the software packages that are sufficiently close to allow a meaningful comparison.

3.2.2 Choice of test files

With the software routes understood, a selection of test data files were chosen to run through the software to compare the values. To make the comparison meaningful, the test files were chosen to ensure a wide variety of profile types. The aim of this was to test various aspects of parameter calculation in order to identify situations where differences may occur. This included variations such as:

- *Profile frequency* - The density of the peaks and valleys of the profiles.
- *Periodicity* - The repeatability of the profile elements. Profiles that look very similar throughout the whole length of the profile have high periodicity; profiles that appear to have no repeatable pattern are likely to have low periodicity.

- *Master-piece profiles* - Profiles that are taken from real surfaces and represent realistic surface finishes, for example, polished and ground surfaces.
- *Simulated profiles* - Profiles that are numerically generated and present as simple shapes, for example, sinusoids, steps and squares.
- *Profile amplitude* - The height values of the peaks and valleys.
- *Profile source* - The location from which the profiles were obtained.

The test files used were type F1 reference data files taken from both NPL and NIST. This ensures a good standard of quality of the data and allows the test to be easily repeatable by anyone. In order to comply with the input requirements of NPL, primary profiles must be used. The NPL datasets are assumed to already comply with their own software requirements, and the NIST datasets are by default given as total profiles, however, primary profiles are included in the downloadable .ZIP folder. All files available from each NMI were downloaded and compared. From these, a short-list of seventeen test files was selected: ten from NPL and seven from NIST, given in table 3.1. These profiles were chosen to cover the structure variations discussed earlier and include a combination of simulated simple surfaces and real measurements. While the majority of the profiles from the NPL database were comprised of the same profile length and number of data points (5.6 mm and 22401, respectively), the NIST profile datasets were much more varied. This will provide variation in the resolution of the datasets given, which may affect the ability of some software to obtain parameter values in some scenarios, and will also test the effect of profile filtration when using the same λc cut off value on different profile lengths. In addition, investigation into the multiscale composition of the profiles was performed, selecting profiles that include a wide spread of spatial frequencies as well as profile with only a small number of prominent spatial frequencies. This influences the periodicity of the profiles, which can impact spacing parameters such as PSm.

Table 3.1 Details of selected test profiles for NMI reference software comparison.

Filename	Description	No. of data points	Profile length	Approx. amplitude range	Prominent spatial wavelengths
501E101_4	High frequency roughness specimen	13197	3.3 mm	200 nm	367 µm
D_course	Low frequency roughness specimen	7998	800 µm	8 µm	800 µm, 571 µm
EDM16ms	EDM master piece (NPL)	22401	5.6 mm	8 µm	100 µm (wide spread)
EDM	EDM master piece (NIST)	22395	5.6 mm	2 µm	267 µm (wide spread)
Mill	Milled master piece	22395	5.6 mm	1.4 µm	200 µm
SRM3filtered	Aperiodic Bullet measurement	5714	1.43 mm	1 µm	130 µm, 179 µm
Cor10gau	Simulated coarse Gaussian distribution	5602	5.6 mm	10 µm	N/A
Impulse	Simulated Dirac delta	7998	4 mm	0.4 µm	N/A
Lapping01ms	Lapped master piece	22401	5.6 mm	0.7 mm	2.8 mm
Millsim	Simulated periodic milled surface	2241	5.6 mm	2 µm	193 µm
Normrand	Simulated random noise	22401	5.6 mm	2 µm	N/A
Saw123	Simulated three sinusoids	22401	5.6 mm	4 µm	80 µm, 160 µm, 320 µm
Saw1pad	Simulated saw wave	22401	5.6 mm	2 µm	160 µm, 320 µm
Sin1	Simulated sine wave	22401	5.6 mm	2 µm	160 µm
Square	Simulated square wave	7998	4 mm	2 µm	200 µm, 67 µm, 40 µm
Steprf	Simulated single step	22401	5.6 mm	1 µm	5.6 mm
Stepsrf	Simulated multiple steps	22401	5.6 mm	2 µm	5.6 mm (wide spread)

3.2.3 Running files through the software

With the test files downloaded in .SMD format, and the closest options decided for each software package, it only remained to run each test file through the software and obtain the parameters. When attempting to input the files into the different software packages, some errors were found.

3.2.3.1 NIST input file issues

ISO 5436-2 defines the standard file-type for profile datasets to be .SMD [14]. The document also gives a description of how the file should be structured, including the different records in the file and what they should contain. The ISO standard also defines that each line, record and file should be ended with specific control characters, which are ASCII characters used to give information to the software reading the file and are not represented by any written symbol. As a result, most common text editors do not show these characters, and a more advanced program is needed to read/write them.

It was found that in order for files to be accepted by the NIST software, the .SMD file had to be terminated with the <SUB>, <CR> and <LF> control characters. The ISO document states 'The last record is further terminated by an end of file (<ASCII 26>)' (<26> is the SUB control character). While somewhat ambiguous as to what else should go on the final line, the examples given in the document show the <SUB> character should be on its own. Because of this, the NPL files, which terminated with just <SUB> in line with the ISO examples, could not be opened by NIST.

In addition to this, it was found that the NIST software would only open the NPL files if the date in the revision number inside record 1 was changed from '2000' to '1999'. This change has a knock-on effect of invalidating the checksum given at the bottom of the file, which is calculated using the ASCII values from the first three records. For the file to be accepted, the checksum must be updated.

Fortunately, both of these changes were still accepted by the NPL and PTB software. To remedy these issues, a MATLAB script was created to read in an NPL .SMD file, change the date to 1999, add the <CF> and <LR> characters to the end, recalculate and update the checksum, and finally output in a separate folder as a new .SMD file.

In addition to this, a further error was found in some of the NIST type F1 files. The files used were primary profile versions of the total profiles listed in the NIST database. It appears that, in recreating the .SMD files with the primary profile height values, the number of points in the profile were not recounted. For some files, a small number of points had been removed from the profile, perhaps due to an effect of the form removal or λs filter. However, the first record in the .SMD file defines the number of points in the file, and this number had not been updated to the true number for the primary profile, and instead matched the number for the total profile. In order for these files to be successfully input into the software, the number of points had to be recounted, and the first record in the .SMD updated.

3.2.3.2 NPL input requirements

Because the NPL software has no user input once the file has been selected, all settings and required information must be included in the .SMD file. This includes which parameters to calculate, and the λc cut-off length for the Gaussian filter. While versions of the NPL datasets could be downloaded with this information already in the .SMD file, the NIST files did not. It was therefore required to edit each file to include these instructions. A MATLAB script was created to perform this task on the NIST files, as well as to recount the number of data points and update the record, as mentioned in the previous section. As both tasks edit the characters in the .SMD file, the checksum also needed to be recalculated and updated for these files.

3.2.3.3 Software outputs

With all the pre-processing complete, the same set of files could then be run through all three software packages. Each file was processed through each software package using the user options shown in appendix A. It was decided to run all files with $\lambda c = 0.8$ mm. This kept another variable stable, was the default setting for the NPL downloaded files, and allowed for potential effects to be seen when the cut-off value was too large or small. Once complete, the next step was to extract the parameter values out from the software in order to analyse them. Again, this was more effort than anticipated.

Each software package offers a way to save the parameter values calculated by the software, however, each one does so in a different format. The NPL software allows either a .SMD or .HTML file to be saved. As work had already been done with .SMD files for the pre-processing, this was the chosen file to download. The structure of the output files was similar to the input .SMD files, but with an additional section with the parameter values present. The PTB software gave the parameter values in the form of .CSV, comma separated value, files, and also gave a .PDF report. The .CSV files were chosen to be downloaded as they would be easier to work with. The NIST software only offered a .PDF report for download, which was not an ideal format for extracting information. The NIST software also presented the parameter values on screen in a table. It was decided that these tables would simply be highlighted and pasted into a .TXT file, as this would lead to a much simpler file than a .PDF.

As all three software packages produced different files, it was required to extract the relevant information from each and reformat them, so they are all the same. This would allow for much easier analysis when it comes to comparison. A set of MATLAB scripts were created, one for each software package, to read in each file, extract the parameter name, value and unit, and write them to a very simple .SMD file, all of the same format and with the parameter values given in micrometres.

3.2.3.4 Parameter value comparison

After successfully obtaining parameter values for all seventeen files from all three software measurement standards, and having them converted into the same format, the next step was to perform comparisons of the values to identify any differences between the software. To do this, graphs were created to allow for easy, visual comparison. A MATLAB script was produced to read in the parameter names and values from the unified .SMD file and produce the graphs from the collected data. Three types of graphs were produced by the script:

1. *Relative difference bar graph* - For a data file, each parameter value is divided by the parameter value obtained by an NMI of choice. The relative values for the two NMIs not selected are displayed on a bar graph, with all parameters shown on the same graph. An example is shown in Fig. 3.1. A value of 1 shows a match with the value of the comparison NMI. This graph is useful for showing large relative differences in the data, however for very small parameter values, can make a very slight difference appear deceptively large.

2. *Parameter value line graph* - This is a simple graph, showing the obtained parameter values for each NMI for a single file. This allows for a quick comparison to highlight any obvious discrepancies, and makes it easy to view the true values of the parameters, in comparison to the relative difference graph. For very similar values, however, it becomes difficult to distinguish between the lines.

3. *Individual parameter bar graph* - In contrast to the other two graphs, this displays the values of just one parameter, but for a chosen selection of test files, for all three NMIs. An example is shown in Fig. 3.2. This allows the value of the same parameter for multiple different test files to be compared. This alternate look at the data is useful as it allows a new set of trends to be found that may not be as easily noticed using one of the other graphs.

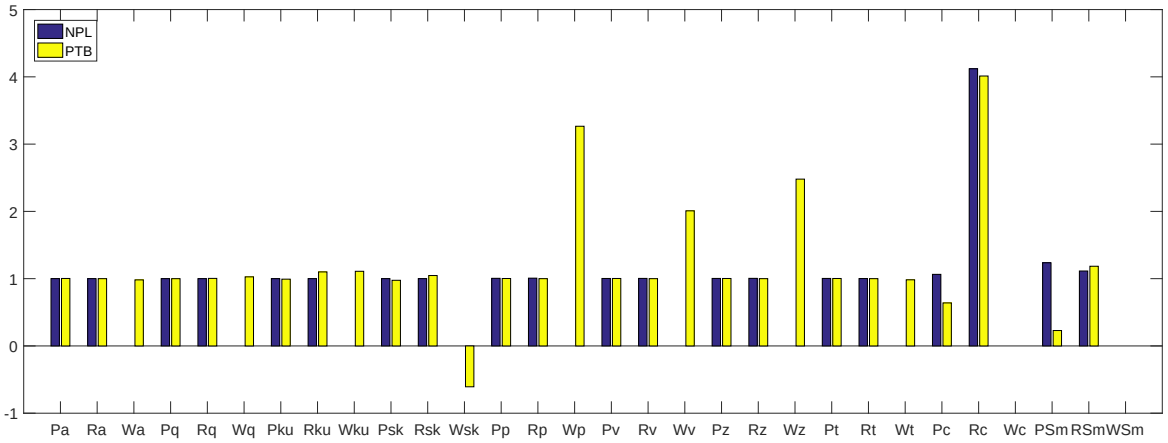


Fig. 3.1 Relative difference bar graph for the test file *lapping01ms*. For this example, NIST has been chosen to be the comparison NMI, so the NPL and PTB results are shown as relative values of those obtained by NIST

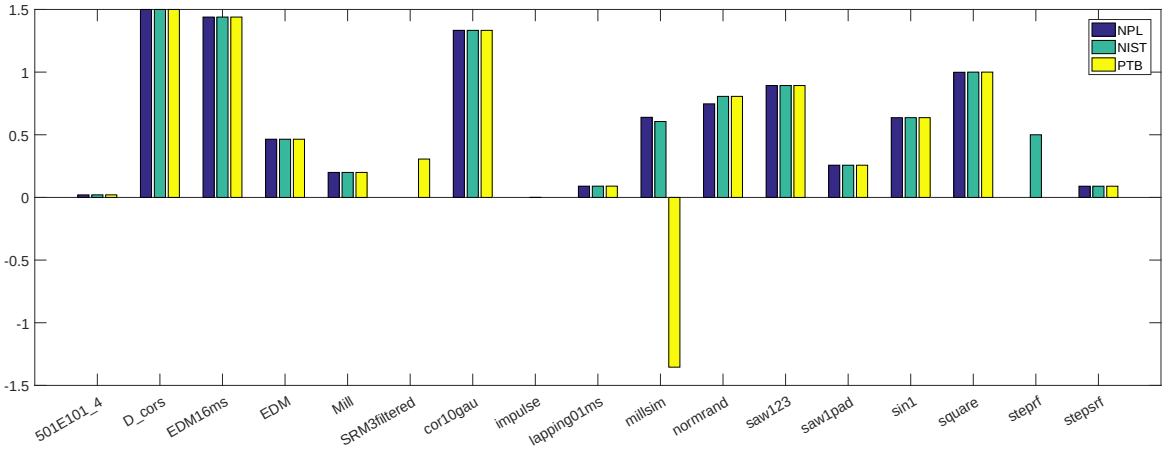


Fig. 3.2 Parameter value graph for the *Pa* parameter, for all test files.

At this stage, it was found that PTB displayed its *Xv* parameter values as negative. The ISO definition of the parameters define the value as the magnitude of the valley depth, thus making the negative result incorrect. This is an error in the code and stems from the calculation of the parameter, as the height values are given relative to a mean line. For the following analysis, the magnitude of the *Xv* values was used and displayed.

3.3 Results and analysis

The analysis of the results revolved around looking at each graph produced and identifying any disagreements in the results. These were noted down, along with details about the size of the disagreement in parameter value and which files and parameters were involved. Once complete, the results were then sifted through again to find any recurring themes. These were then used to form a short-list, where each discrepancy was analysed further in an attempt to explain its cause.

3.3.1 NPL sharp step overestimation

During the analysis it was found that, compared to the results obtained by the other two, the NPL software showed an overestimation for some parameter values. This was found mainly on peak/valley parameters, where no averaging of the ordinate values occurs, for test files that had very sharp steps in them. Some examples are shown in Fig. 3.3, which shows the relative parameter values for the *square* and *stepsrf* files, with NIST as the comparison NMI. This shows that for both files, the primary and roughness peak/valley parameters obtained by NPL are around 20% larger than the values obtained by the other two.

These overestimations for peak/valley parameters suggest that the interpreted profiles are slightly larger than those of NIST/PTB. Both NIST and PTB assess the discrete data point directly, however, NPL uses a cubic spline function to interpolate the data and produce

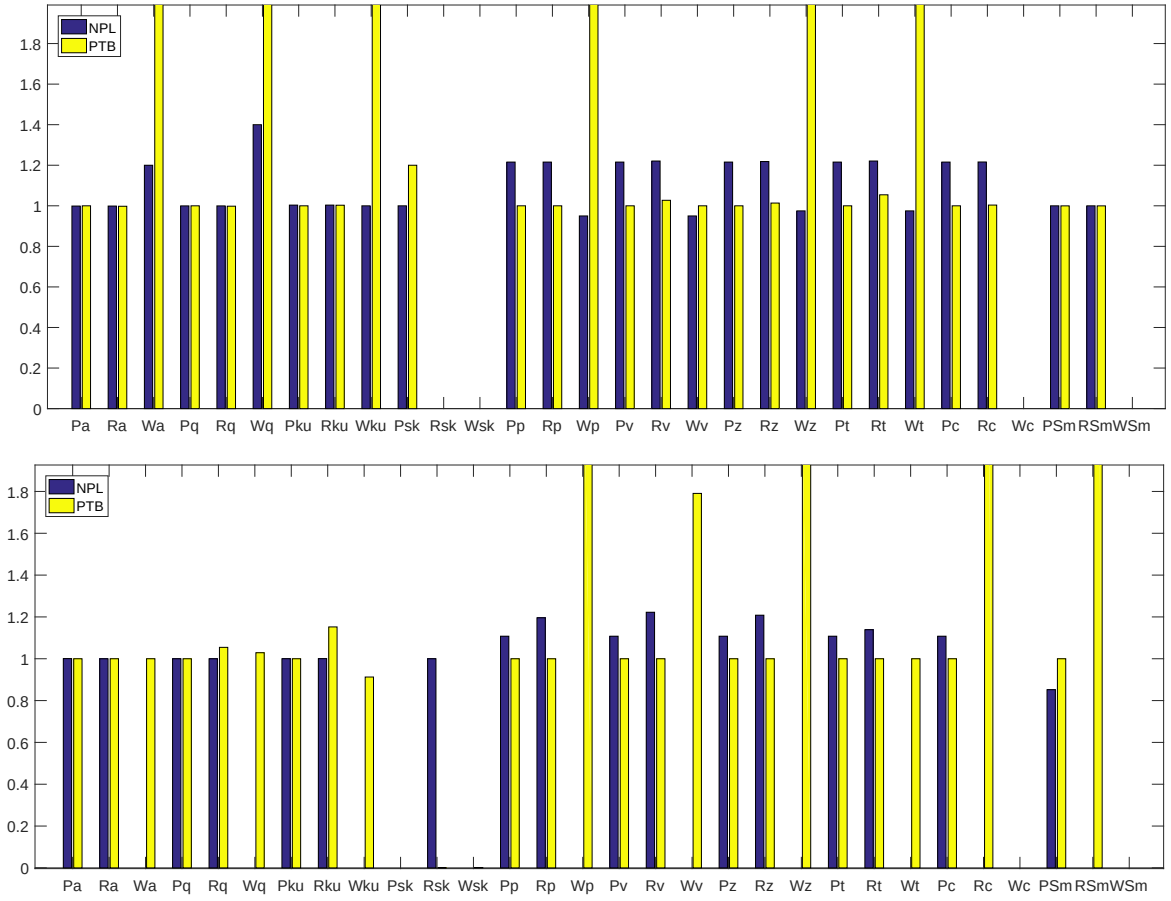


Fig. 3.3 Relative difference graphs for the *square* (top) and *stepsrf* (bottom) test files. NIST has been used as the comparison NMI. Overestimation is observed for P and R peak/valley parameters.

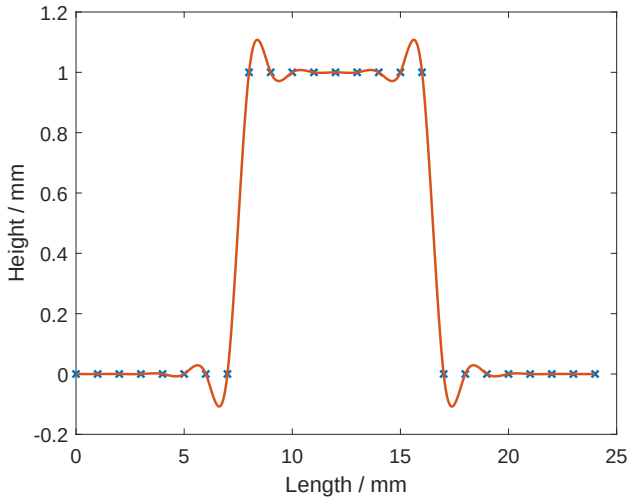


Fig. 3.4 A cubic spline used to fit a top-hat function. Oscillations around the step points are clearly visible.

a continuous representation. For very steep steps, as found in both *square* and *stepsrf*, the cubic spline can oscillate, causing small overestimations of peak and valley values. A simple example of this is shown in Fig. 3.4, where a cubic spline function has been used to fit a simple top-hat. At the step points, overestimation is shown as the spline oscillates to meet the required points. It is expected that a similar effect is happening with the NPL software.

Both methods of calculating parameters, either directly from the discrete data or by evaluating an interpolation of the data, are valid approaches of software implementation that comply with the specifications in ISO 4287 and ISO 5436-2. This, however, highlights and issue with using software implementations for assessing parameter calculation software. As both methods are acceptable, yet produce different results, end users can validate the performance of their software against different references and obtain results which are in disagreement, which can affect part tolerance assessment in high precision applications. Instead, an alternative software validation method is required which incorporates the mathematical definitions in the ISO 4287 standard directly, and is free from the variations caused by software implementation.

3.3.2 PTB large waviness values

For a variety of files, it was found that PTB would obtain larger waviness parameter values. This was found to be most prominent for some of the peak/valley parameters, occurring for almost all files, however, it was also present on some files for the average ordinate parameters (*Wa*, *Wsk*, etc.). Fig. 3.5 gives examples of some of these, showing PTB in yellow giving larger waviness values than the other two. The top graph shows the *Wp* parameter in more detail, showing several files such as *EDM16ms, Mill* and *cor10gau* with much higher *Wp* values. The bottom graph looks at the *EDM16ms* file specifically, and reveals that PTB obtained larger values than NIST and NPL for several waviness parameters, including *Wv, Wz, Wku* and *Wq*.

After discussions with PTB directly about this result and differences in their software, it was discovered that PTB treat the ends of the waviness profile differently to NPL and NIST. In order to account for end effects, both NPL and NIST cut off the ends of the profile of length λc and assess the remaining central portion. PTB, however, does not do this for the waviness profile. This difference in filtration technique is not an option that users can change in the software, resulting in PTB performing a different operation that NPL and NIST and consequently obtaining a different waviness profile. The use of a Gaussian convolution filter, as was used for all filtration operations as part of this comparison, can produce end effects on the profile that cause the height values to skew significantly positive or negative. This effect is likely to result in height values that exceed those across the rest of the profile, and so would contribute to height amplitude parameters such as *Wp*.

The reason for this difference in software implementation stems from an ambiguity in the ISO definition. The evaluation lengths for the primary and roughness profiles are defined in ISO 4288 section 4.4 [38]. The primary profile is equal to the length of the feature being measured, and the roughness profile is equal to $5 \times \lambda c$. The waviness profile, however, has no such definition. It is, therefore, unclear as to whether its length should be made similar

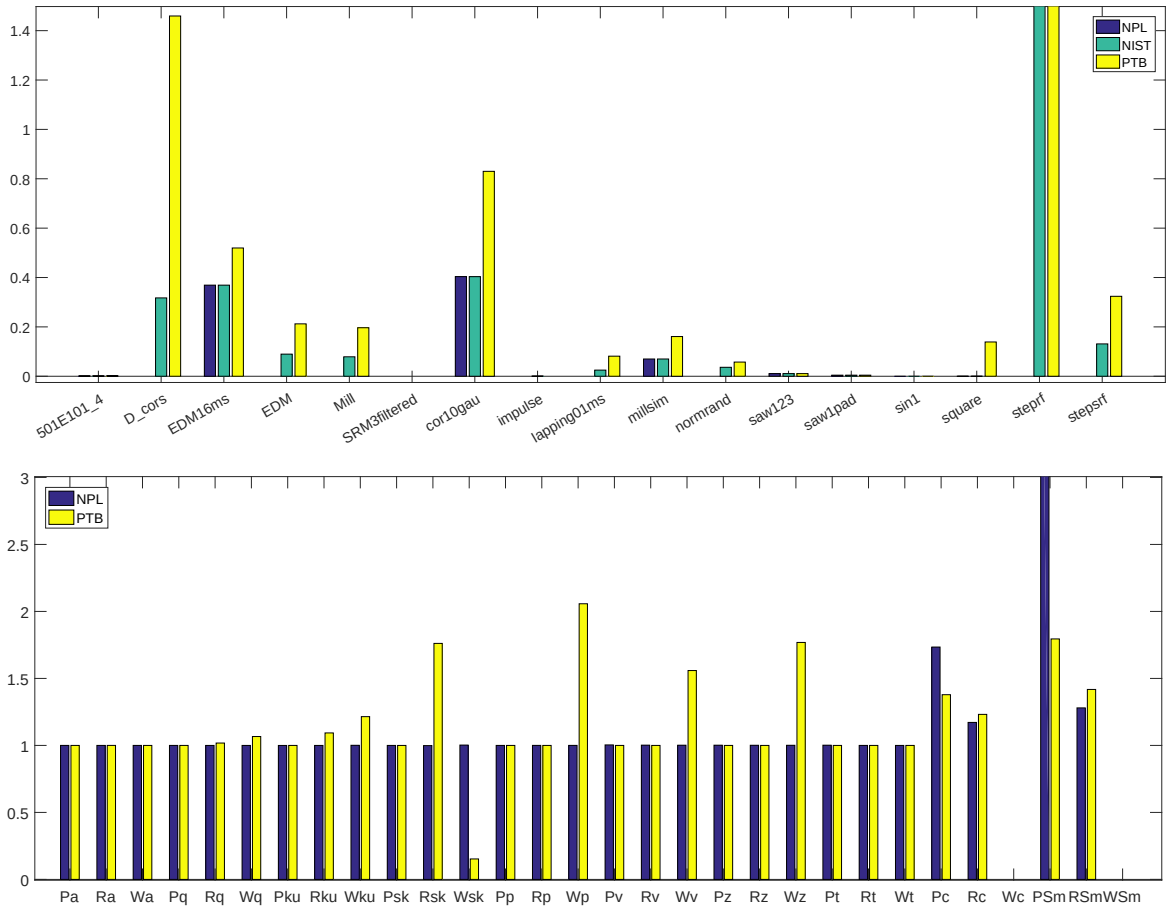


Fig. 3.5 *Top*: Values for the *Wp* parameter for all files, showing a clear tendency for the *Wp* value obtained by PTB (yellow) to be larger than the others for many test files. *Bottom*: Relative difference graph for the *EDM16ms* test file, showing multiple waviness parameters with larger values obtained by PTB (yellow).

to the roughness profile, or the primary profile. It appears that NIST and NPL have chosen to treat it the same as the roughness profile, accounting for any end effects that may occur after filtration, and PTB have decided to keep it the same length as the primary profile, perhaps under the assumption that it should not be altered unless specifically stated in the ISO standard. Another possible interpretation would be that if the length of the roughness profile is defined to be $5 \times \lambda_c$, then the length of the waviness profile should be $5 \times \lambda_f$. This approach mirrors the roughness profile length in that the relevant high-pass cut-off filter length is used to determine the length of the profile. This ensures an appropriate length of surface measurement is used to include the same amount of spatial frequency component periods to enable an analysis that is more equivalent to the roughness profile case. Without a suitable profile length, any analysis performed of the waviness profile is limited, and the information obtained from the spatial frequencies of interest cannot be properly compared to higher spatial frequency components found in the roughness profile.

3.3.3 NIST *Wc* and *WSm* failures

It was discovered that that NIST software had some issues with the *Wc* and *WSm* parameters. In fact, the NIST software was unable to obtain a value for any test file at all. Due to the nature of the calculation of these parameters, and the low amplitude of the waviness profile for many of the test files, it was anticipated that the software packages would struggle to obtain the values for all test files, however, not being able to obtain any at all is a notable issue. Evidence of this is shown in Fig. 3.6, where PTB and NPL are shown to obtain values for some files, but NIST obtains none.

Upon further research into the implementation of these parameters by NIST, it was found in their 'help' documentation that the height discrimination for the *Xc* and *XSm* parameters is defined as a user set percentage of *Rz*. The ISO standards define the height discrimination to

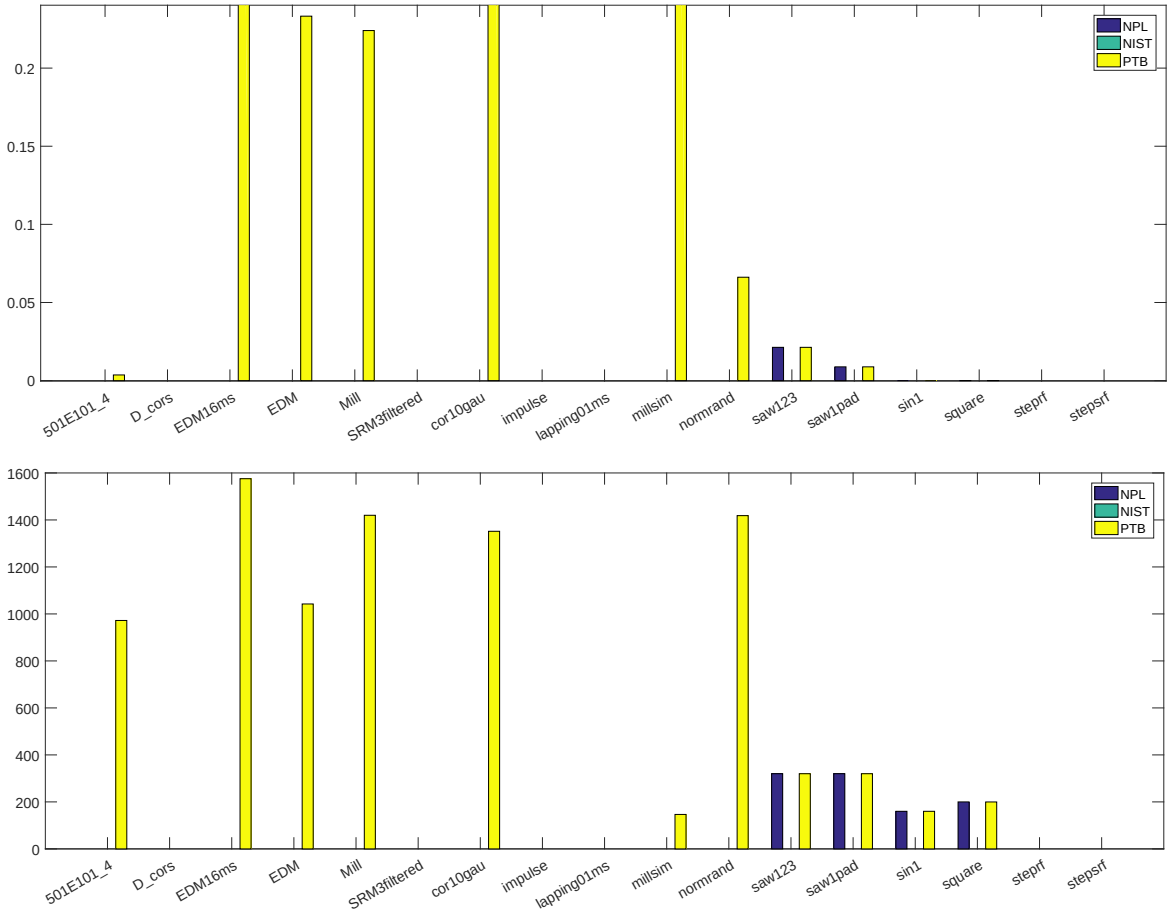


Fig. 3.6 *Top*: Graph showing the values obtained for the *Wc* parameter. *Bottom*: Graph showing the values obtained for the *WSm* parameter.

use *Xz*, e.g. use *Wz* for *Wc*/*WSm*. This could simply be a poorly worded statement, however, if it truly describes the implementation of the software, it could explain the lack of results.

As the waviness profiles have a lower amplitude than the other profiles, if the software were to be using 10% of *Rz* for the height discrimination instead of *Wz*, the cut-off height would be higher than the majority of the waviness profile, leading to a greatly reduced number of profile elements and, therefore, no data from which to calculate the parameters.

3.3.4 NPL waviness parameter failures

Another recurring issue discovered in the NPL results was an inability to obtain waviness parameters for some test files. These files include *steps, normrand, Mill, lapping01ms, D_coarse* and *EDM*. An example of some of these files, in particular *Mill* and *normrand*, is given in Fig. 3.7. It should be mentioned here that for the files for which NPL does obtain waviness parameters, the values are in good agreement with the other NMIs, suggesting the parameter calculation algorithm can work well.

When investigating the outputted .SMD file, the error message 'Missing crossing point in sample lengths' is found next to the failed waviness parameters. Through further investigation with NPL it was discovered that for the waviness profiles that were unable to produce parameter values, a small number of mean line crossing points were present within each sample length. A good example of this is shown in Figure 3.8, where the waviness profiles for the *millsim* and *normrand* files are shown. For the *millsim* dataset, from which the NPL software was able to obtain waviness parameters, at least three crossing points are present for each sampling length. For the *normrand* dataset, from which the NPL software could not obtain waviness parameters, there are at most two crossing points per sampling length, with one sampling length containing no crossing points.

The information obtained from Figure 3.8 suggests that the NPL software requires a minimum number of mean line crossing points within each sample length to calculate the

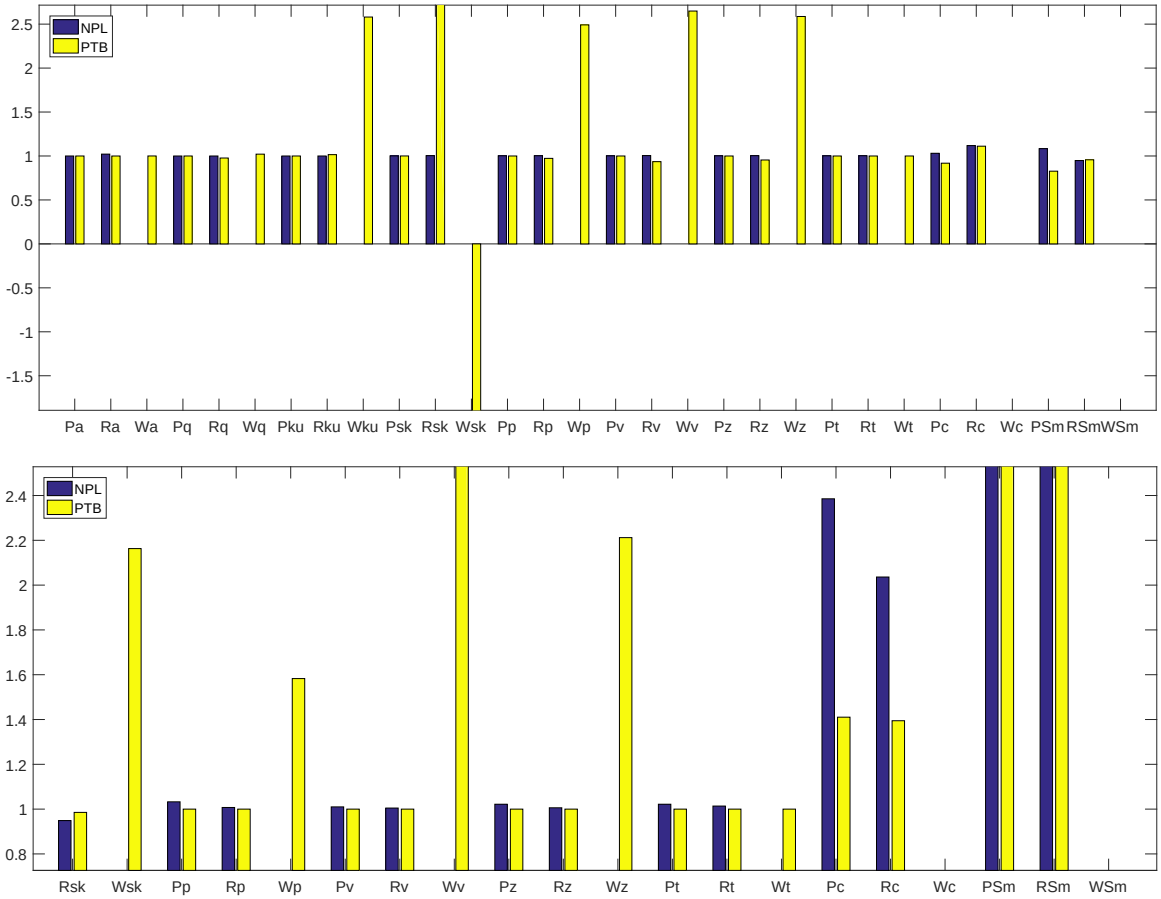


Fig. 3.7 *Top*: Relative difference graph for the *Mill* test file, with NIST as the comparison NMI. *Bottom*: Relative difference graph for the *normrand* test file, with NIST as the comparison NMI. For both graphs, the lack of waviness parameter values for the NPL software is clear.

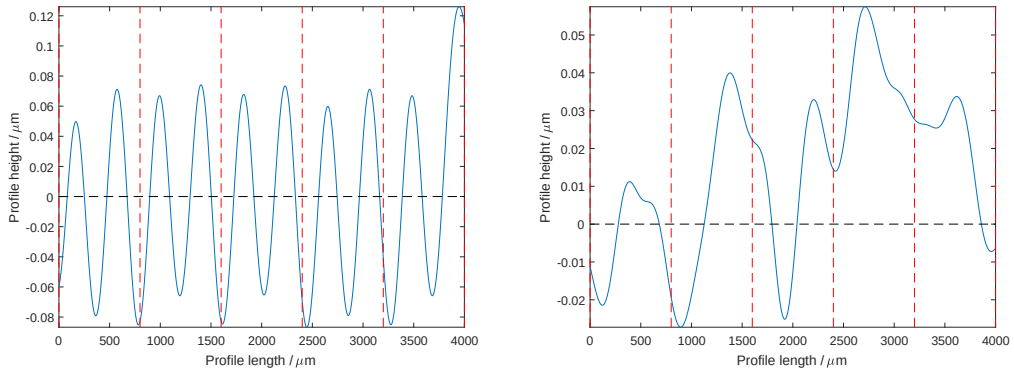


Fig. 3.8 *Left*: Waviness profile for the *millsim* file. *Right*: Waviness profile for the *normrand* file. For each graph, the vertical dashed lines separate the sampling lengths, and the horizontal dashed line highlights the mean line.

parameter values. Whether this is the correct approach is up for debate, as an argument can be made that a surface with so few mean line crossing points may not be suitable to calculate parameter values that are meaningful and descriptive of the surface, although it is not strictly in accordance with the specifications laid out in ISO 4287. However, both NIST and PTB were able to calculate waviness parameter values for many of these surfaces, and from their perspective it may be up to the user to decide whether the results obtained are useful.

Regardless of the reasons, the fact that two approaches to handling such surfaces with minimal mean line crossing points exist across different software measurement standards is evidence of ambiguity in the ISO specification standards.

3.3.4.1 NPL reference file form removal

As discussed above, part of the investigation into the NPL waviness issue was a test to see if form removal was not performed on some of the files. Interestingly, for two of the seventeen files, form removal was missing. These files were *millsim* and *step*, both taken from the NPL database. Originally, *millsim* was one of the files that failed to obtain waviness parameters, however, once form had been removed, it was successful and obtained results that showed good agreement with the NIST software.

Upon bringing this to the attention of NPL, it was discovered that, although the NPL reference software requires a primary profile as input, it is not necessarily the case that the reference datasets match the input requirements of the reference software. This presents the potential for confusion to users, as it would be assumed by many that the datasets would be ready to work with the associated software. Furthermore, of the ten NPL files tested, only two needed any form removal, suggesting the files are not all given in the same condition.

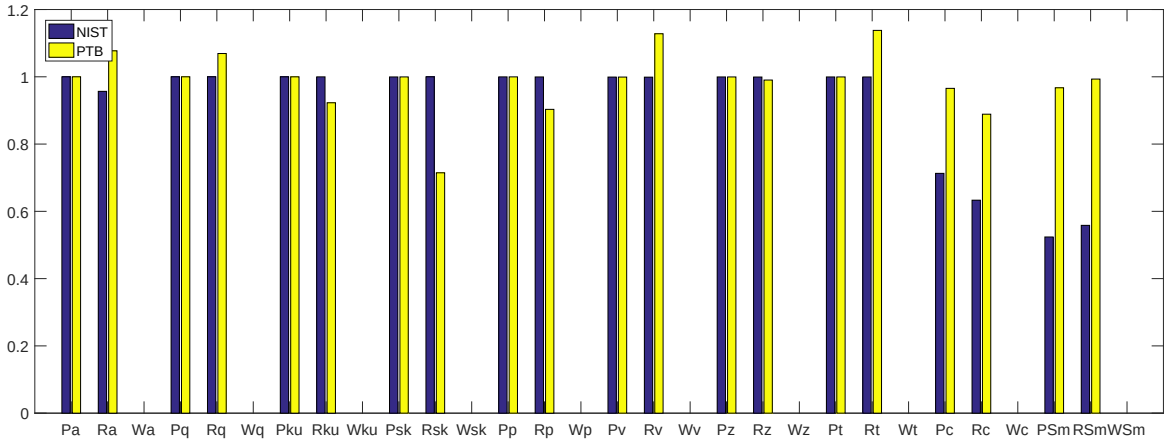


Fig. 3.9 Relative difference graph for the *D_cors* test file, with NPL as the comparison NMI. NIST shows many parameter values at 1, indicating a good agreement with the NPL value. PTB shows more variation on the roughness parameters.

3.3.5 PTB D_coarse disagreement

One file that showed disagreements between NMIs was the *D_coarse* file. The results for this file, with NPL as the comparison NMI, are shown in Fig. 3.9. PTB shows some disagreement, in particular on the roughness parameters. This comes in stark contrast to the relationship between NPL and NIST, which show good agreement throughout, ignoring the height/spacing parameters. The PTB roughness values do not vary by a large amount, with the variations being less than 30% of the NPL value, however, the result is interesting when compared to the close agreement in the values given by the NPL and NIST software.

The reason for these disagreements is not certain, however the fact that the primary values do not exhibit such disagreement suggest the filtration of the profile has some effect. Table 3.1 shows that the *D_coarse* profile has the lowest mean spatial frequencies of all profiles included in the comparison, with its most prominent spatial wavelength at a value of 800 µm, equivalent to the value of the λc cut off filter used. The Gaussian convolution filter used in this comparison does not have a perfect transmission characteristic, and instead exhibits a monotonically increasing curve, with 50% transmission centred on the cut off value [62]. Therefore, spatial frequency components close to the λc cut off value are subject to

significant amplitude modulation. For the case of the *D_coarse* profile, the most significant spatial frequency components are at and around this cut off value, and so the specific shape of the filter transmission curve will have a significant effect on the resulting surface. It is expected, here, that the PTB software implements a Gaussian convolution filtration algorithm that differs from that implemented by NPL and NIST, giving a different transmission curve and thus a different resulting roughness profile.

3.3.6 Height/spacing parameter disagreements

As was expected, notable amounts of disagreement between the three NMIs were found for the height/spacing parameters, *Xc* and *XSm*. As mentioned previously, previous work uncovered some ambiguities in the definition of the *RSm* parameter that led to the potential for different interpretations of said definition to obtain different results [56, 63]. As *Xc* uses the same discrimination methods, the same issues are expected to apply. Although this work was carried out in 2002, it appears these definitions are still leading to different results. Examples of these results are shown in Fig. 3.10.

These results show a tendency for NIST to produce slightly lower values than the other two NMIs. Such an effect is not present on the very periodical, simulated test files such as *saw1pad* and *sin*, suggesting it is related to the methods employed to determine the profile elements. It appears that the NIST software uses a different interpretation of the discrimination definitions, causing it to obtain different profile elements to the other two, and hence different values.

In addition to this, the *PSm* results, shown in figure 3.11, show the NPL software obtaining significantly larger values than NIST and PTB for a good number of data files. Such an effect is not observed for the roughness parameters, suggesting the application of the filter removes this issue. This could, therefore, be related to the extended length of the profile, as the ends are not cut off to manage any affects. The details of the effect are unknown, however, it is

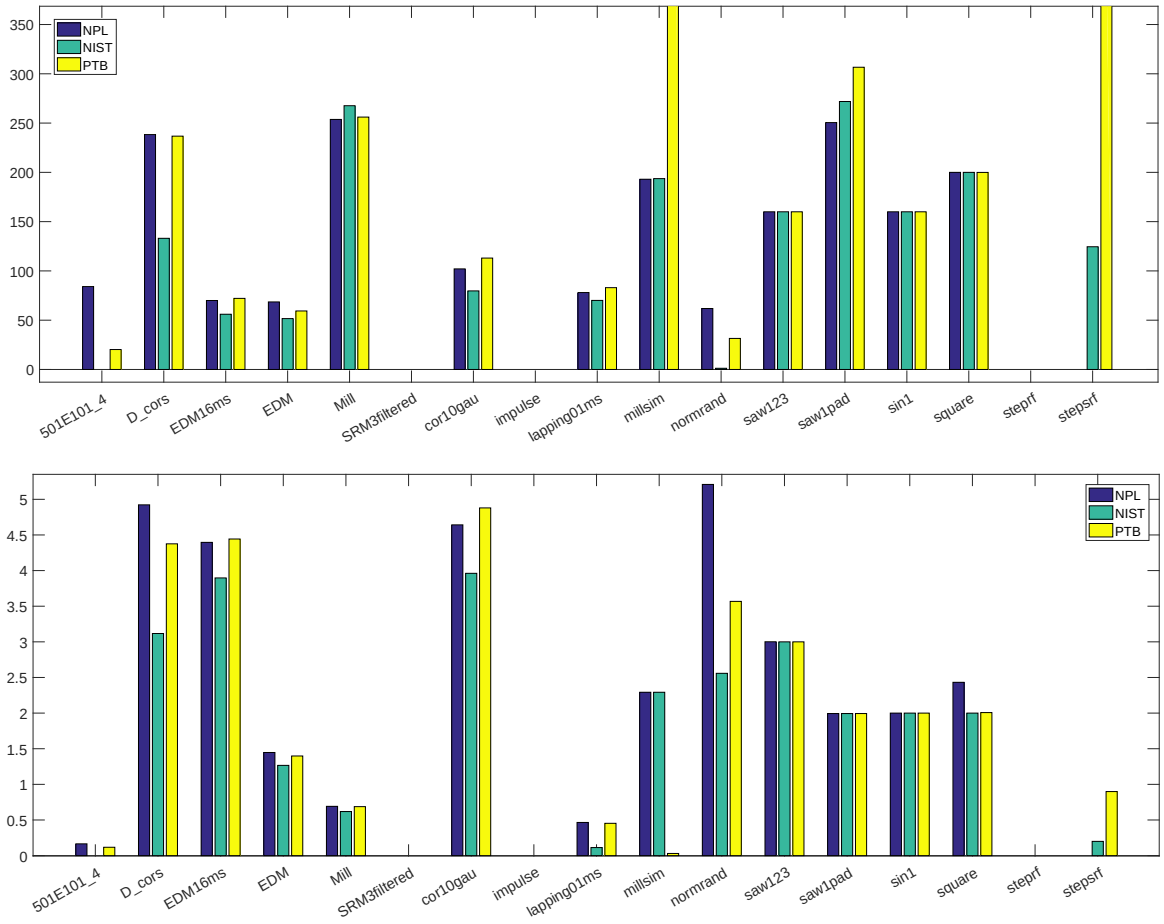


Fig. 3.10 *Top*: Graph displaying each file result for the *RSm* parameter. *Bottom*: Graph displaying each file result for the *Rc* parameter.

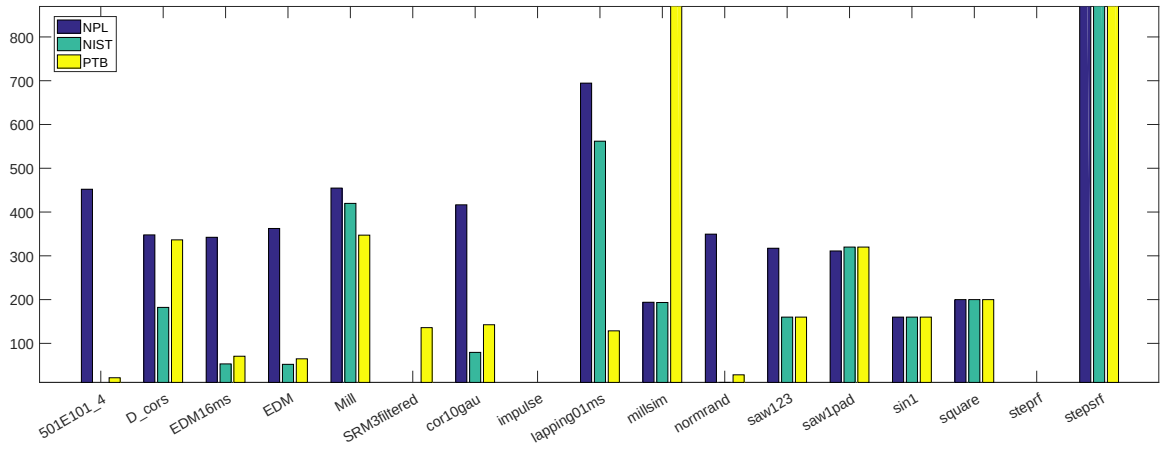


Fig. 3.11 Graph displaying each file result for the *PSm* parameter.

likely to be related to the discrimination of the profile elements, as this effect is not seen for other parameters.

3.4 Significance of work

The work in this chapter has sought to identify any potential areas of concern in the current state of profile surface texture parameter calculation. The work has aimed to be inclusive of all stages of the parameter calculation software, giving focus to the inputs and implementation as well as the results, in the hope to develop a more rounded image of the software packages and facilitate better analysis. It has been found that some differences between the reference software packages are present. These are varied, and stem from both differing implementations and differing interpretations of the same standard.

Several of the results presented in this chapter highlight variations found in waviness profiles. Section 3.3.2 discussed the limitations of the results obtained from waviness profiles due to the length of the profile relative to the spatial frequencies present. This is not an issue for areal surface texture parameters, which the rest of the work presented in this thesis will focus on. In the areal case, a scale limited surface of interest is exacted from the measured surface area, utilising the application of user-defined high-pass and low-pass filters to obtain what is known as an S-F surface. No distinction is made between roughness and waviness profiles; the filters are used to define the spatial frequency bandwidth of the surface. Filtration aside, results were still presented here that showed variation between software packages due to the direct calculation of surface texture parameters, most notably in sections 3.3.1 and 3.3.5. This area will be the focus of the rest of the thesis as it affects all areas of surface texture parameter calculation, regardless of the filter used. Surface texture filtration will instead by the subject of focus for future work.

The results obtained from performing this comparison further justifies the need for a validation method for surface texture parameter calculation software that is free from specific

algorithmic implementations. Various results throughout the chapters showed slight variations in results between the three NMI reference software packages, with section 3.3.1 highlighting a prominent example. As discussed, using software implementations as the primary method of assessing the performance of surface texture parameter calculation software introduces the potential for disagreements between reference values, which can lead to errors during collaborative projects. As discussed in section 2.1.2, the traceability of a surface texture parameter result is dependent on both the measurement of the surface and the software used to calculate the parameters. For the traceability of reference software to be properly established, an understanding of the uncertainties added to a parameter result due to the software implementation is required. With each reference software providing different results for a given type F1 reference dataset, it is not clear to what extent the software is causing a deviation from the 'true' parameter value. Moving toward a mathematical basis for reference values serves as a promising method to achieve this goal, and is investigated in later chapters.

Another benefit of this work is a demonstration of the need for clearer, less ambiguous definitions in the standards documentation. At present, two reference software packages can give different parameter values for the same dataset, and thus two different results against which third-party software is expected to be validated. This leads to differences in values in collaboration projects, causing disagreements in potentially crucial measurements.

In addition, several issues were highlighted during the stages preceding the actual parameter value calculation. Problems were encountered when dealing with the .SMD files, in which slightly different formats were leading to input failures. Some of this was due to poor formatting of the file or limitations of the software on NIST's side, however, it should be considered that is in part due to the convoluted nature of the .SMD file structure, including outdated control characters which are invisible to the user of any mainstream text editor. It comes as no surprise that both NPL and NIST offer the reference data files in a selection of other data formats, as the .SMD format adds an unnecessary additional layer of complexity

to the process. It is suggested that this file format be replaced by a more robust format which does not require such specific use of control characters. One potential candidate is the .SDF format. This is already used for areal surface texture so is familiar to many in the industry, and is more robust, accepting any combination of <CR> and <LF> to end lines, and ends records simply with a '*' [15].

Chapter 4

The development of mathematical references for functional surface texture parameters

The work presented in this chapter has been published in [64].

4.1 Introduction - A mathematical approach

Chapter 3 and appendix B motivated the need for a novel method of surface texture parameter software validation that is traceable. Introducing mathematical traceability into software validation makes performance assessment more meaningful and reduces the possibility of variations between parameter reference values calculated by different organisations. Chapter 3 performed an analysis of the current state of the art of type F2 software measurement standards, and found multiple examples of parameter value differences from software developed by different NMIs, due to differences in the discrete implementations of surface texture parameter calculations. Appendix B further motivates the need for a traceable software

validation method, by presenting the importance and adoption of surface texture parameters in a wide variety of fields in the manufacturing industry.

An approach is introduced in this chapter that seeks to develop a method for validating surface texture parameter calculation software in a way that removes reliance on type F2 reference software and their inherent errors. Such an approach could be used alongside the current practice of F2 reference software, and serve to create a more complete, traceable assessment of software for surface metrology. The new method involves creating surfaces that are mathematically defined using algebraic expressions that express their surface height as continuous functions of spatial location. Surface texture parameters for these surfaces are then evaluated accurately using the analytical definitions of the parameters given in ISO 25178-2 [1]. This approach removes any approximations or inaccuracies that can be introduced when dealing with numerical methods. The resulting parameter values can then be used to validate and assess the performance of surface texture parameter calculation software. Previous work has calculated parameter values from mathematical functions [59, 65], but these have often only calculated a small subset of height amplitude parameters such as *Rq*, *Rsk* and *Rku* and have not attempted to calculate a wider range of parameters such as functional parameters. In addition, these attempts have been limited to profiles, and have only generated single term sinusoid functions, not expanding to more complex surfaces and alternative surface functions that can test software across a wider range of topographies.

This chapter focusses on the mathematical evaluation of 'functional' surface texture parameters, a series of parameters that are calculated from a material ratio curve and defined in ISO 25178-2 [1]. Mathematical evaluation of field ISO 25178-2 surface texture parameters will be covered in chapter 5.

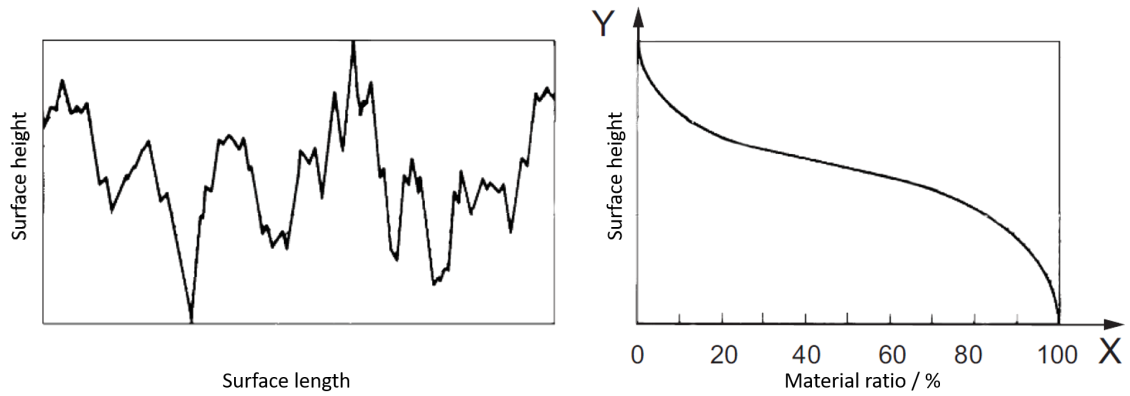


Fig. 4.1 Example material ratio curve (*right*) for a profile surface measurement (*left*) [1]. For the material ratio curve, the Y-axis defines the surface height from the lowest measured value and the X-axis defines the material ratio as a percentage. For the profile measurement, the X-axis is a distance in one direction.

4.2 Mathematically defined surfaces

The new approach to the validation of surface texture parameter calculation software relies on mathematical definitions of surfaces. These definitions comprise analytical functions that describe the height of the surface at any given (x,y) point, and are used to calculate surface texture parameter values.

The calculation of functional surface texture parameters first relies on obtaining a material ratio curve for the surface under evaluation, $S_{\mathrm{mr}}(c)$. The material ratio curve, defined in [1], describes the ratio of the area of the material at a specified height, c, to the evaluation area, which is the portion of the surface for specifying the area under evaluation. Another, more statistical interpretation of the material ratio curve is that it is the cumulative probability function of the height values within the evaluation area. An example material ratio curve for a profile surface measurement is shown in figure 4.1.

As the material ratio curve forms the basis of the functional surface texture parameters, it is not necessary to explicitly define a surface using an analytical function. Instead, a material ratio curve can be analytically defined, and functional surface texture parameters can be calculated directly from that. This approach simplifies the process and removes the

complexities of requiring a mathematical operation to relate a surface height function to its cumulative probability function.

As the material ratio curve only relates surface height to material ratio, the curve can be defined using a relation between just two variables. As material ratio curves are often displayed with the material ratio on the horizontal axis (keeping the surface height values on the vertical axis as is the case for a traditional profile surface measurement), it has been decided to define the material ratio curve using the inverse material ratio, $S_{\mathrm{mc}}(m_r)$, with the material ratio, $\mathrm{m_r}$, as the independent variable expressed as a percentage, and the surface height as the dependent variable.

It is worth noting here that this work defines vertical range with the lowest surface height as $0\,\mu$m. This choice differs from the convention used in [1], in which height values are taken from the reference plane, and has been made solely to simplify the graphs and equations for the purpose of this work. The shape of the material ratio curve and final mathematical values are not affected by this choice. For a function to represent a physically realistic material ratio curve of a measurement, it must adhere to the following conditions:

- The value of $S_{\mathrm{mc}}(m_r)$ corresponding to a material ratio value of $100\,\%$ is zero, equivalent to the lowest surface height.
- The value of $S_{\mathrm{mc}}(m_r)$ corresponding to a material ratio value of $0\,\%$ is the highest surface height value, equivalent to the distance from the lowest surface height.
- The material ratio curve is monotonically non-increasing. That is, the gradient of the curve must never be positive.

These conditions ensure that the mathematically defined curve represents a physically realistic material ratio curve by ensuring that it encompasses the full range of material ratios between $0\,\%$ and $100\,\%$, and that a decrease in specified surface height guarantees that no decrease in the value of material ratio is obtained.

For this work, material ratio curves are represented using third degree polynomial functions. Such functions are easily able to satisfy the conditions listed above. It should be stated here that third degree polynomials are not the only option available to produce mathematical material ratio curves, and any function that satisfies the requirements of reproducing a physically realistic curve can be used. A material ratio curve is defined mathematically using the following polynomial:

$$S_{\mathrm{mc}}(m_r) = Z_{\mathrm{Range}}\left[1 + bm_r + c{m_r}^2 + d{m_r}^3\right], \tag{4.1}$$

where Z_{Range} is the vertical height range of the material ratio curve, and the constraint

$$1 + b + c + d = 0 \tag{4.2}$$

applies. An example curve is shown in figure 4.2, which uses the values

$$\begin{aligned} b &= -\frac{27}{10} \\ c &= \frac{51}{10} \\ d &= -\frac{34}{10} \\ Z_{\mathrm{Range}} &= 1\,\mu\mathrm{m}. \end{aligned} \tag{4.3}$$

For the derivations in the following section, $Z_{\mathrm{Range}} = 1$.

4.2.1 Material ratio curve evaluation

Defining a surface mathematically using a material ratio curve allows for functional surface texture parameters to be calculated without the need for an analytical representation of the height values of the surface. Whilst this approach makes calculating the functional parameters

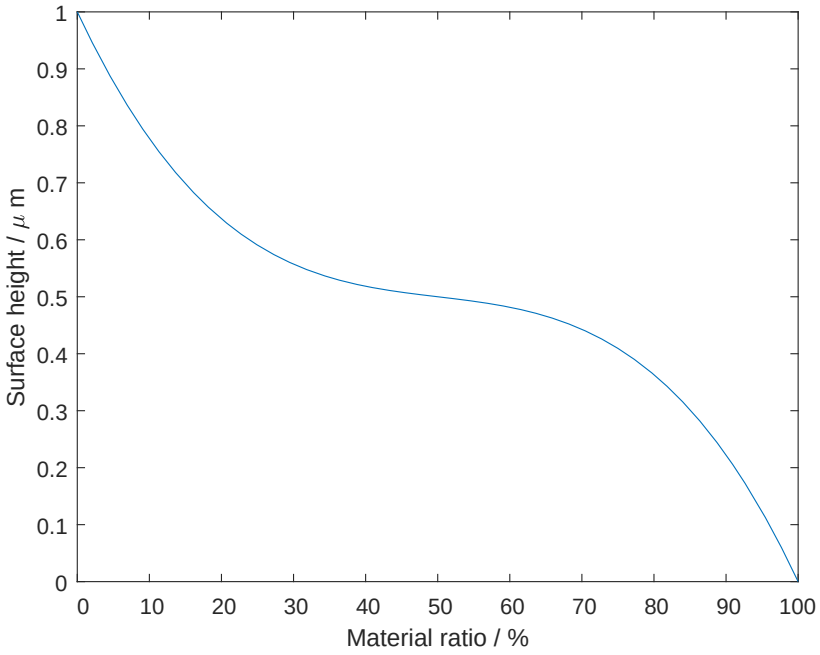


Fig. 4.2 Material ratio curve defined from the general case in equation 4.1.

easier, a discrete dataset describing the height values of the surface is still required for use with surface texture parameter calculation software.

A discrete dataset representation of the defined surface can be obtained by sampling the inverse material ratio curve to obtain height values. Material ratio values, in the closed interval $[0\,\%, 100\,\%]$, can be input into the material ratio curve equation, such as that given in equation 4.1, and corresponding height values will be returned.

It is important that a uniform distribution of material ratio values is used to ensure a representative distribution of returned height values. An example dataset created by sampling the material ratio curve defined in equation 4.1 is shown in figure 4.3. An array of 250 000 material ratio values uniformly-spaced between $0\,\%$ and $100\,\%$ was input into equation 4.1 to obtain an array of height values. The height values are then structured into a 500×500 2D array to produce a surface height dataset, arranged in such a way as to ensure there are no discontinuous steps in data height across the surface.

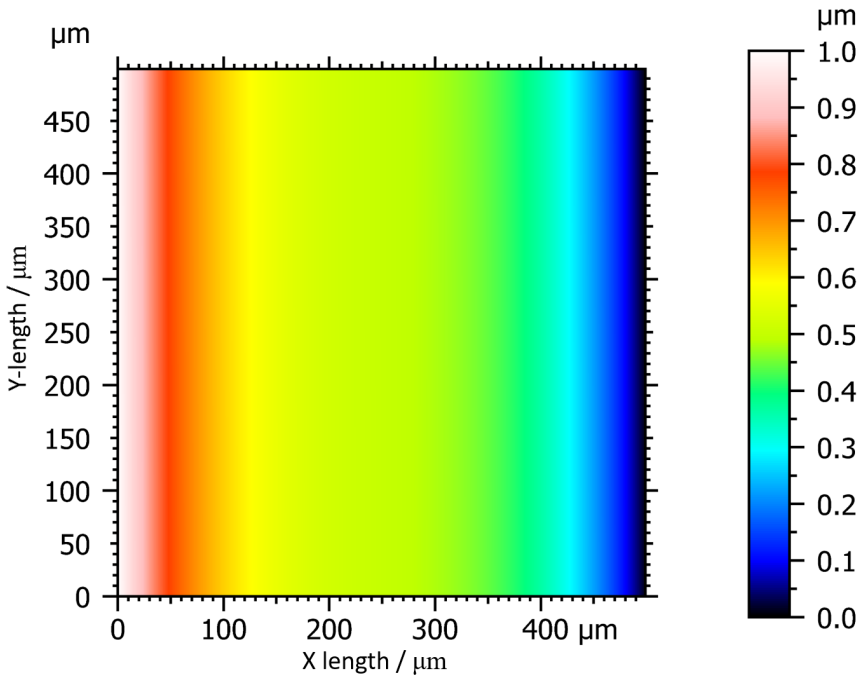


Fig. 4.3 Discrete dataset generated by sampling the material ratio curve defined in equation 4.1.

4.3 Parameter evaluation

With the surface defined and sampled, the surface is represented as both a mathematical material ratio function, and as a discrete dataset. To evaluate the performance of surface texture parameter calculation software, the mathematical function is used to obtain parameter values. These parameter values can then be used as a reference against which the parameter calculation software can be assessed.

4.3.1 Equivalent straight line

The evaluation of functional surface texture parameters first requires the calculation of the equivalent straight line. As detailed in clause 5.2 of [1], the equivalent straight line is calculated for the central region, which is defined by the smallest gradient secant of the material ratio curve that intersects at two points and encapsulates 40 % of the material ratio.

As the material ratio curve must be monotonically decreasing, calculating the central region can be achieved by minimising the height difference between two evaluation points on the material ratio curve that are 40 % apart. Minimising the height difference can be achieved by finding the value of the material ratio at which the gradient of the function describing the difference between two heights 40 % apart is zero, given by

$$\frac{\mathrm{d}}{\mathrm{d}m_r}\left[S_{\mathrm{mc}}(m_r) - S_{mc}\left(m_r + 40\,\%\right)\right] = 0. \tag{4.4}$$

For the example material ratio curve given in equation 4.3, the difference is given by the equation

$$\begin{aligned} S_{\mathrm{mc}}(m_r) - S_{\mathrm{mc}}(m_r + 40\,\%) = {} & \frac{Z_{\mathrm{Range}}}{10}\left[10 - 27\left(\frac{m_r}{100}\right) + 51\left(\frac{m_r}{100}\right)^2 - 34\left(\frac{m_r}{100}\right)^3\right] \\ & - \frac{Z_{\mathrm{Range}}}{10}\left[10 - 27\left(\frac{m_r+40}{100}\right) + 51\left(\frac{m_r+40}{100}\right)^2 - 34\left(\frac{m_r+40}{100}\right)^3\right] \\ = {} & \frac{Z_{\mathrm{Range}}}{10}\left[\frac{204}{5}\left(\frac{m_r}{100}\right)^2 - \frac{612}{25}\left(\frac{m_r}{100}\right) + \frac{602}{125}\right]. \end{aligned} \tag{4.5}$$

Taking the derivative gives

$$\frac{\mathrm{d}}{\mathrm{d}m_r}\left\{\frac{Z_{\mathrm{Range}}}{10}\left[\frac{204}{5}\left(\frac{m_r}{100}\right)^2 - \frac{612}{25}\left(\frac{m_r}{100}\right) + \frac{602}{125}\right]\right\} = \frac{Z_{\mathrm{Range}}}{10}\left[\frac{102}{125}\left(\frac{m_r}{100}\right) - \frac{153}{625}\right], \tag{4.6}$$

and equating to zero yields the value for the lower central region material ratio, $m_{r,l}$,

$$\begin{aligned} \frac{Z_{Range}}{10}\left[\frac{102}{125}\left(\frac{m_{r,l}}{100}\right) - \frac{153}{625}\right] &= 0 \\ m_{r,l} = 100\frac{153\cdot 125}{102\cdot 625} &= 30\,\%, \end{aligned} \tag{4.7}$$

and a value for the upper central region material ratio, $m_{r,u}$, of 70 %.

The equivalent straight line is defined as the straight line within the central region that gives the least square deviation in the direction of the surface ordinates. To calculate the equivalent straight line, a linear least-squares fit must be applied to the material ratio curve within the central region. This fitting is implemented by using a continuous version of the regular discrete linear least-squares operation.

A linear approximation to the material ratio curve within the central region is required of the form

$$S_{\mathrm{mc,linear}}(m_r) = \alpha + \beta m_r, \tag{4.8}$$

which requires a minimisation of the equation

$$f(\alpha,\beta) = \int_{m_{r,l}}^{m_{r,u}} (\alpha + \beta m_r - S_{\mathrm{mc}}(m_r))^2 \, dm_r, \tag{4.9}$$

where $m_{r,l}$ and $m_{r,u}$ are the boundaries of the central region. The minimisation can be found by satisfying

$$\frac{\partial f}{\partial \alpha} = \frac{\partial f}{\partial \beta} = 0. \tag{4.10}$$

Evaluating these derivatives, the solutions can be found by solving the simultaneous equations

$$(m_{r,u} - m_{r,l})\alpha + \frac{({m_{r,u}}^2 - {m_{r,l}}^2)}{2}\beta = \int_{m_{r,l}}^{m_{r,u}} S_{\mathrm{mc}}(m_r) dm_r \tag{4.11}$$

and

$$\frac{({m_{r,u}}^2 - {m_{r,l}}^2)}{2}\alpha + \frac{({m_{r,u}}^3 - {m_{r,l}}^3)}{3}\beta = \int_{m_{r,l}}^{m_{r,u}} m_r S_{\mathrm{mc}}(m_r) dm_r. \tag{4.12}$$

For the example given in equation 4.3, equations 4.11 and 4.12 become

$$\begin{aligned} (70-30)\,\alpha_1 + \frac{1}{2}\left(70^2 - 30^2\right)\beta_1 = \\ \int_{30}^{70} \left\{ \frac{Z_{\mathrm{Range}}}{10} \left[10 - 27\left(\frac{m_r}{100}\right) + 51\left(\frac{m_r}{100}\right)^2 - 34\left(\frac{m_r}{100}\right)^3 \right] \right\} dm_r \end{aligned} \tag{4.13}$$

$$2\alpha_1 + 100\beta_1 = Z_{\text{Range}} \tag{4.14}$$

and

$$\begin{aligned} &\frac{1}{2}\left(70^2 - 30^2\right)\alpha_1 + \frac{1}{3}\left(70^3 - 30^3\right)\beta_1 \\ &= \int_{30}^{70} \left\{ \frac{Z_{\text{Range}}}{10} \left[10m_r - 27\left(\frac{{m_r}^2}{100}\right) + 51\left(\frac{{m_r}^3}{100^2}\right) - 34\left(\frac{{m_r}^4}{100^3}\right) \right] \right\} dm_r \end{aligned} \tag{4.15}$$

$$2000\alpha_{\text{example}} + \frac{316\,000}{3}\beta_{\text{example}} = \frac{123456}{125}Z_{\text{Range}}. \tag{4.16}$$

Finally, substituting equation 4.14 into equation 4.16 gives the values

$$\alpha_1 = \frac{3079}{5000}Z_{\text{Range}} \tag{4.17}$$

$$\beta_1 = -\frac{579}{250\,000}Z_{\text{Range}}, \tag{4.18}$$

which result in the equivalent straight line, S_{eq},

$$S_{\text{eq}} = S_{\text{mc,linear}}(m_r) = Z_{\text{Range}}\left(\frac{3079}{5000} - \frac{579}{250\,000}m_r\right). \tag{4.19}$$

Figure 4.4 shows the equivalent straight line for the example given in equation 4.3, overlaid onto the material ratio curve shown in figure 4.2.

4.3.2 Functional surface texture parameters

With the equivalent straight line calculated, the functional surface texture parameters can be obtained. The simplest parameter is the S_{k} parameter, which is defined as the distance

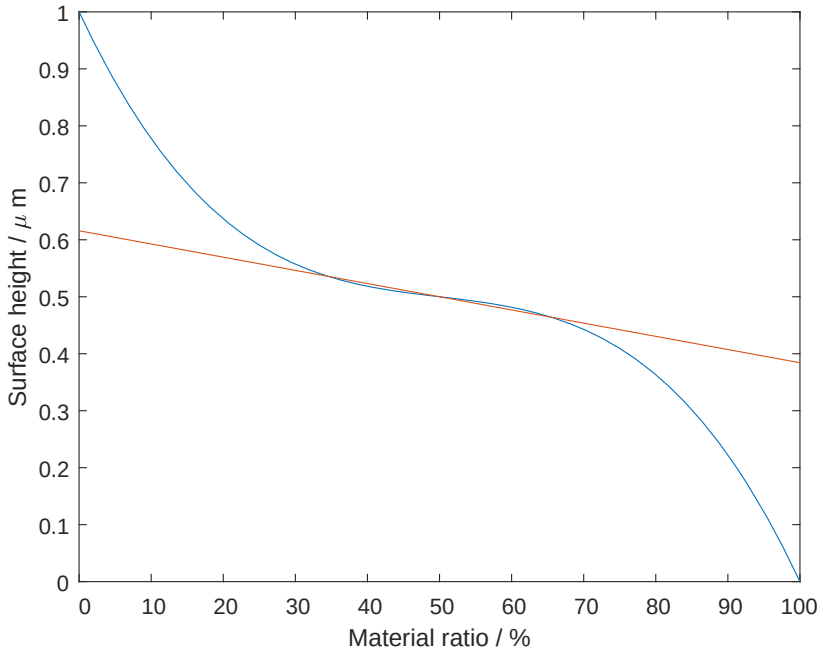


Fig. 4.4 Equivalent straight line (red) and corresponding material ratio curve (blue) for the example defined in equation 4.3.

between the highest and lowest levels of the core surface, and can be evaluated as

$$\begin{aligned} S_{\mathrm{k}} &= S_{\mathrm{eq}}(0\,\%) - S_{\mathrm{eq}}(100\,\%) \\ &= \frac{Z_{\mathrm{Range}}}{10}\left[\frac{3079}{500} - \left(\frac{3079}{500} - \frac{579}{250}\right)\right] \\ &= 231.6\,\mathrm{nm} \end{aligned} \tag{4.20}$$

for the example from equation 4.3.

S_{mr1} and S_{mr2} are obtained by finding the material ratio values that correspond to the points on the material ratio curve that have the same surface height values as those obtained by evaluating $S_{\mathrm{eq}}(0\,\%)$ and $S_{\mathrm{eq}}(100\,\%)$, respectively; that is, for S_{mr1} for example, by finding the solution for m_r of the equation

$$S_{\mathrm{mc}}(m_r) = S_{\mathrm{eq}}(0\,\%). \tag{4.21}$$

Continuing with the equation 4.3 example, this gives

$$\frac{Z_{\text{Range}}}{10}\left[10-27\left(\frac{m_r}{100}\right)+51\left(\frac{m_r}{100}\right)^2-34\left(\frac{m_r}{100}\right)^3\right]=\frac{Z_{\text{Range}}}{10}\left(\frac{3079}{500}\right) \tag{4.22}$$

which can be rearranged in the form

$$\left[-34\left(\frac{m_r}{100}\right)^3+51\left(\frac{m_r}{100}\right)^2-27\left(\frac{m_r}{100}\right)+\frac{1921}{500}\right]=0, \tag{4.23}$$

which is a third-degree polynomial equation that is solvable using a general formula [66]. Evaluating this formula gives the result

$$\begin{aligned} m_r=\frac{50}{51}\Bigg\{51+10\,000\left[\frac{1}{2}\left(\frac{-4\,517\,937}{125\times10^{12}}\pm\frac{459\sqrt{97\,947\,149}}{125\times10^{12}}\right)\right]^{1/3} \\ -\frac{153}{5000}\left[4\left(\frac{-4\,517\,937}{125\times10^{12}}\pm\frac{459\sqrt{97\,947\,149}}{125\times10^{12}}\right)\right]^{-1/3}\Bigg\} \end{aligned} \tag{4.24}$$

Taking the positive root yields the real solution, which is given as

$$S_{\text{mr1}}=m_r=22.0883\,\% \tag{4.25}$$

when evaluated numerically and reported to six significant figures. By calculating the number to high precision, it would be possible to report the value to 16 significant figures and give a bound on the numerical error in that rounded value. For readability, five significant figures are presented here. The same method can be used to find S_{mr2}, giving

$$S_{\text{mr2}}=77.9117\,\%. \tag{4.26}$$

The next parameters of interest are S_{pk} and S_{vk}. These parameters are defined such that they represent the height of right-angled triangles that have S_{mr1} and S_{mr2} as the widths of their

bases, respectively, and that are of the same areas as the 'hill area' and 'dale area' respectively. The hill and dale areas are defined as the areas above and below the material ratio curve which delimit the core height S_{k} and are shown graphically in figure 4.5. Mathematically, these areas, denoted by S_{a1} and S_{a2}, can be obtained by integrating the material ratio curve to obtain the area enclosed between the material ratio limits of interest. For S_{a1}, the material ratio range is $0 \leq m_r \leq S_{\mathrm{mr1}}$, and the area under the highest level of the core surface should be subtracted from the total area under the curve, given by

$$S_{\mathrm{a1}} = \int_0^{S_{\mathrm{mr1}}} S_{\mathrm{mc}}(m_r)dm_r - S_{\mathrm{mr1}} \cdot S_{mc}(S_{\mathrm{mr1}}). \tag{4.27}$$

For S_{a2}, the material ratio range is $S_{\mathrm{mr2}} \leq m_r \leq 1$, and the area under the curve should be subtracted from the total area under the lowest level of the core surface, given by

$$S_{\mathrm{a2}} = (1 - S_{\mathrm{mr2}}) \cdot S_{\mathrm{mc}}(S_{\mathrm{mr2}}) - \int_{S_{\mathrm{mr2}}}^{1} S_{\mathrm{mc}}(m_r)dm_r. \tag{4.28}$$

Substituting the required expressions into these equations for the example defined in equation 4.3, whether performed manually or using a symbolic mathematics software package, is a straightforward task that results in long equations that do not need to be written in their entirety here. Evaluated numerically, these equations give

$$S_{\mathrm{a1}} = 352948\,\mu\mathrm{m}^3/\mathrm{mm}^2 \tag{4.29}$$

and

$$S_{\mathrm{a2}} = 352948\,\mu\mathrm{m}^3/\mathrm{mm}^2. \tag{4.30}$$

These areas, along with S_{mr1} and S_{mr2}, can be used to find S_{pk} and S_{vk} using the simple formulae

$$S_{\mathrm{pk}} = \frac{2S_{\mathrm{a1}}}{S_{\mathrm{mr1}}} \tag{4.31}$$

and

$$S_{\mathrm{vk}} = \frac{2S_{\mathrm{a2}}}{1 - S_{\mathrm{mr2}}}, \tag{4.32}$$

giving values for the example given in equation 4.3 of

$$S_{\mathrm{pk}} = 0.319584\,\mu\mathrm{m} \tag{4.33}$$

and

$$S_{\mathrm{vk}} = 0.319584\,\mu\mathrm{m}. \tag{4.34}$$

The final set of parameters included in this work are the volume surface texture parameters. These relatively simple parameters stem from two integral functions, the void volume

$$V_{\mathrm{v}}(p) = K \int_{p}^{100\,\%} [S_{\mathrm{mc}}(p) - S_{\mathrm{mc}}(q)]\,dq \tag{4.35}$$

and the material volume

$$V_{\mathrm{m}}(p) = K \int_{0\,\%}^{p} [S_{\mathrm{mc}}(q) - S_{\mathrm{mc}}(p)]\,dq, \tag{4.36}$$

where K is a scaling factor to convert to millilitres per metre squared. For this work, which has been operating in units of micrometers, $K = 1$. With these functions defined, the volume surface texture parameters can be obtained by evaluating the two equations for different

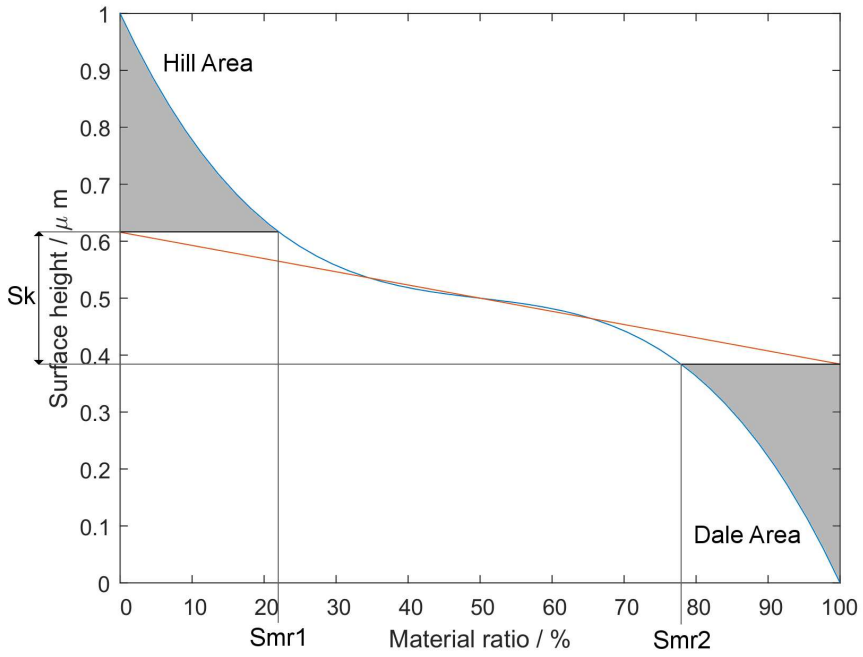


Fig. 4.5 Material ratio curve and associated hill and dale areas (shaded) [1].

values of p and q. Evaluating $V_{\mathrm{v}}(p)$ for the example given in equation 4.3 gives

$$V_{\mathrm{v}}(p) = \frac{27}{2}\left[1-\left(\frac{p}{100}\right)^2\right] - \frac{51}{3}\left[1-\left(\frac{p}{100}\right)^3\right] + \frac{34}{4}\left[1-\left(\frac{p}{100}\right)^4\right] + \left(1-\frac{p}{100}\right)\left[-27\left(\frac{p}{100}\right) + 51\left(\frac{p}{100}\right)^2 - 34\left(\frac{p}{100}\right)^3\right]. \tag{4.37}$$

A similar expression can be obtained for $V_{\mathrm{m}}(p)$. Evaluating both allows the following volume parameters to be obtained:

$$V_{\mathrm{vv}} = V_{\mathrm{v}}(80\,\%) = \frac{193}{625} = 0.03088\,\mathrm{ml/m^2} \tag{4.38}$$

$$V_{\mathrm{vc}} = V_{\mathrm{v}}(10\,\%) - V_{v}(80\,\%) = \frac{10283}{4000} = 0.257075\,\mathrm{ml/m^2} \tag{4.39}$$

$$V_{\mathrm{mp}} = V_{\mathrm{m}}(10\,\%) = \frac{2071}{20000} = 0.010355\,\mathrm{ml/m^2} \tag{4.40}$$

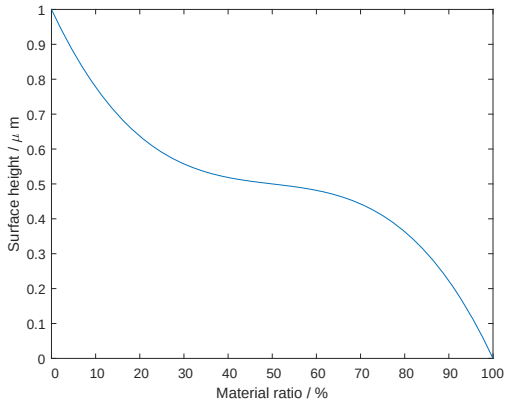
$$V_{\mathrm{mc}} = V_{\mathrm{m}}(80\,\%) - V_{m}(10\,\%) = \frac{6293}{4000} = 0.157325\,\mathrm{ml/m^2}. \tag{4.41}$$

4.4 Comparison with existing software

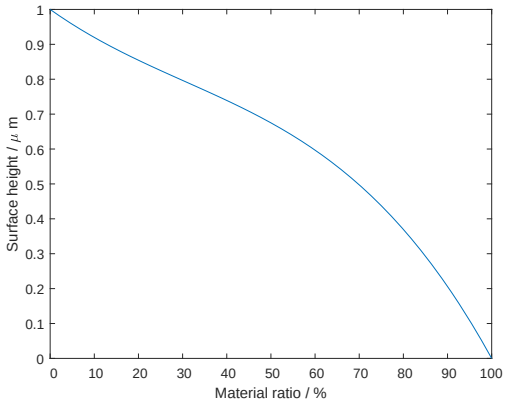
A mathematically defined material ratio curve has been used to obtain accurate parameter values that are calculated according to the mathematical definitions given in ISO 25178-2 [1]. These parameter values can now be used as a reference against which the results returned by surface texture parameter software can be compared. To perform the comparison, the material ratio curve must be sampled to create a discrete dataset, as explained in section 4.2.1.

To test the software, a range of material ratio curves are needed to recreate difference surface types. A total of five material ratio curves were defined from third degree polynomials for use in this comparison, shown in figure 4.6. The material ratio curve shown in figure 4.2 represents a normally distributed surface in which the amounts of peaks and valleys on a surface are approximately equal, but with the majority of surface heights occurring around the mean [19], and is shown in figure 4.6 (a). Such curves can be seen in measurements from polished surfaces [67]. Material ratio curves also exist that are almost linear, corresponding to surfaces with an approximately even distribution of surface heights, such as unworn turned steel [68, 69]. Such surfaces are represented in figures 4.6 (b) and 4.6 (c). In addition, surface topographies can exist with large positive or negative skewness, that is, a large number of high peaks but few large valleys, or vice versa. This can be seen in certain ground or milled surface finishes [70], and are represented in figures 4.6 (d) and 4.6 (e). Mathematical parameter values were calculated for each of these five surfaces using the methods described in section 4.3.

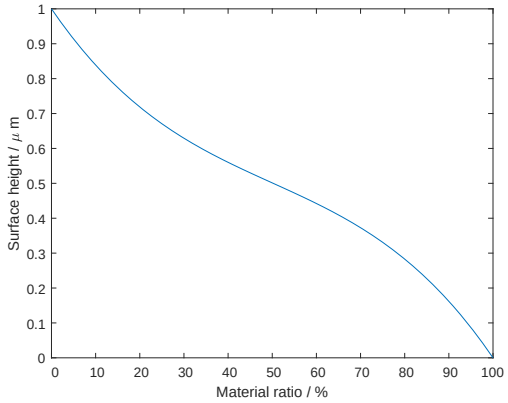
Three surface texture parameter calculation software packages were used in the comparison, however, not all of them gave the option to calculate every required functional surface texture parameter. For each software package, all additional surface dataset processing was disabled and only the parameter calculation was performed.



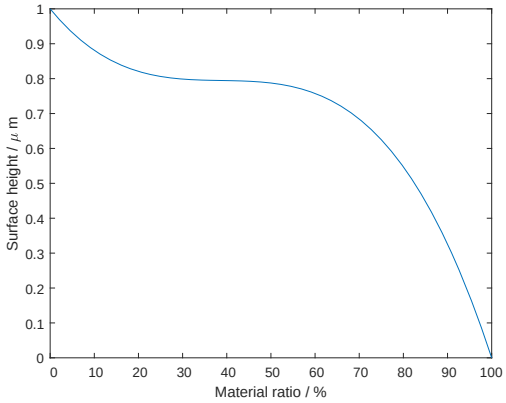
(**a**) Regular curve.



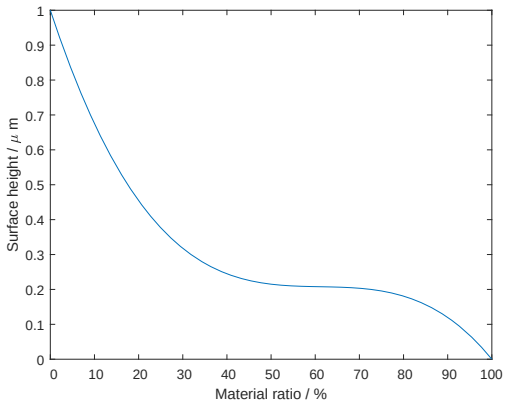
(**b**) Edge case curve.



(**c**) Slight curve.

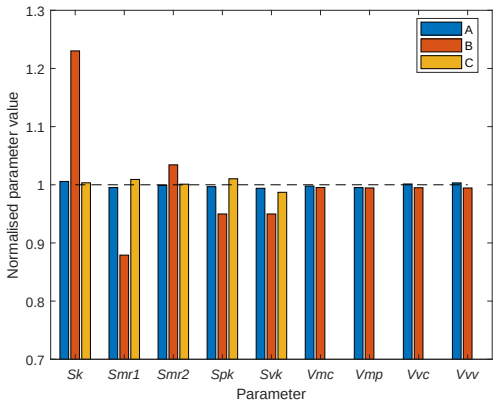


(**d**) High curve.

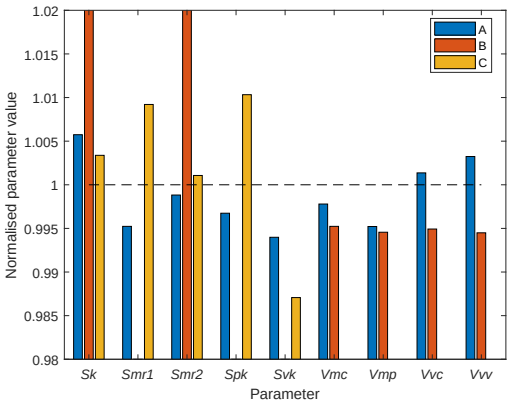


(**e**) Low curve.

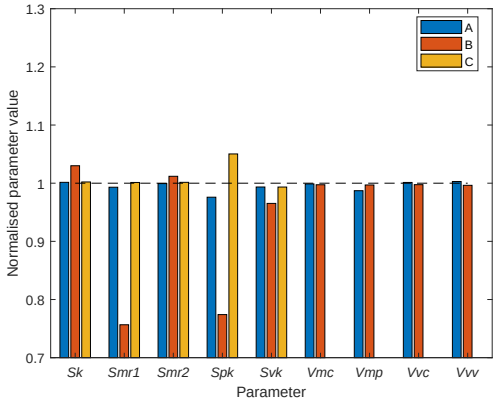
Fig. 4.6 Material ratio curves used in the comparison with existing software.



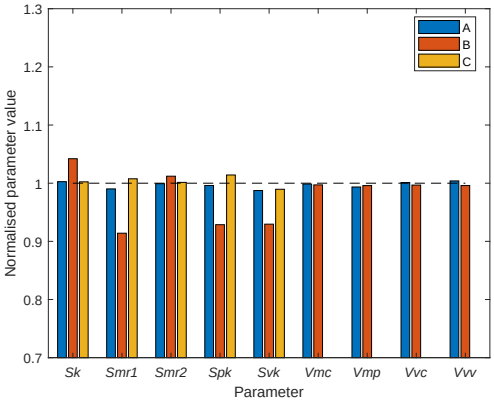
(**a**) Regular curve.



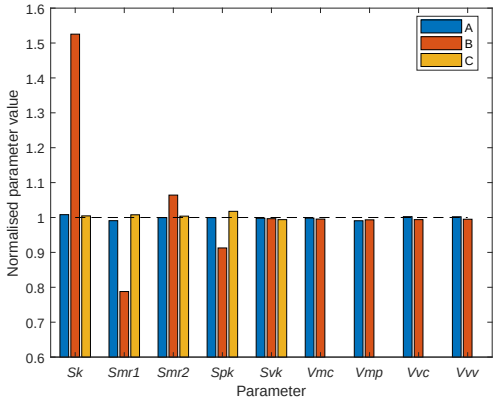
(**b**) Regular curve zoomed in.



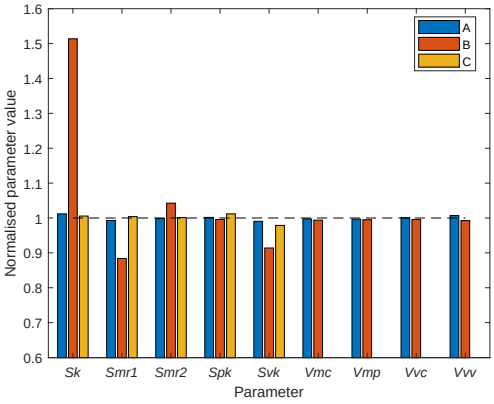
(**c**) Edge case curve.



(**d**) Slight curve.



(**e**) High curve.



(**f**) Low curve.

Fig. 4.7 Software calculation results for functional surface texture parameters, for five different material ratio curves. Software calculated values have been normalised to the corresponding mathematically obtained parameter value, shown by the horizontal line.

Figure 4.7 shows the results obtained by parameter calculation software for the five material ratio curves, normalised to the mathematical value. Note that some parameters were not available on all parameter calculation software packages, and so some bars are not present.

From the results, it is clear that there is some discrepancy between the values obtained by each software package. Software A obtains values more consistently close to the mathematical value, with some compared to the mathematical value seen for S_{pk} and S_{vk} for the 'Edge case' curve. Software C performs similarly to A for the parameters it is able to calculate, however, volume parameters were not available. Software C also showed a tendency to overestimate the S_{pk} and S_{mr1} parameters. Software B shows the largest consistent deviation from the mathematical value for the first five material ratio parameters, however, volume parameters showed good agreement. In particular, software B showed consistent overestimation of the S_{k} parameter, with values differing by over 50 % in some cases when compared to the mathematical value. Software B also showed consistent underestimation of S_{mr1}, S_{pk} and S_{vk}, differing by as much as 20 % for the 'Edge case' curve. The 'Edge case' curve was also the surface for which software A and C performed worse, particularly for the S_{pk} parameter, though the effect is only slight in comparison to the results for software B. The 'Edge case' curve has an almost linear surface height distribution and the least prominent characteristic 'S' curve shape. This shape makes the calculation of the region of the curve corresponding to the peaks for difficult, as the gradient deviation between the peak and central regions is very small. Because of this, it is unsurprising that the software packages tested showed an increase in deviation from the mathematical reference in this instance. Figure 4.7 (b) shows a more detailed zoom of figure 4.7(a) to better highlight the small-scale deviations. Here, software A and C deviate around 1 % from the mathematical value. The 'High' curve and 'Low' curve in figures 4.7 (e) and (f) show similar results to the 'Regular' curve. This is due to the similar curve shape and gradient values across the curves, with only the height of the

central region differing significantly, which is shown to have little effect on the calculation of the parameters.

These results highlight the need for a new approach to providing reference values and assessing the performance of parameter calculation software, as each software package is giving different results when compared to the mathematically-defined reference. Without a traceable, mathematically obtained value, the performance of parameter calculation software cannot be adequately assessed [71].

4.5 Verification of the dataset

As the parameter calculation software operates on a discrete dataset instead of the mathematical function, it is important that the dataset is an appropriately accurate representation, and that any errors that may affect the results are identified.

4.5.1 Dataset height order assessment

As the only aspect of the discrete dataset of interest is the derived material ratio curve, in principle how the discrete height values are organised across the surface should not be significant. In reality, the parameter calculation software may process the height values in a way that is dependent on the height arrangement.

Four different datasets were created that contain the same height values arranged in the following orders:

1. Linearly increasing along each row, where each subsequent row increases carrying on from the row above. This method is used in the rest of the chapter;
2. Linearly increasing, with alternate rows flipped;
3. Linearly increasing, with alternate columns flipped;

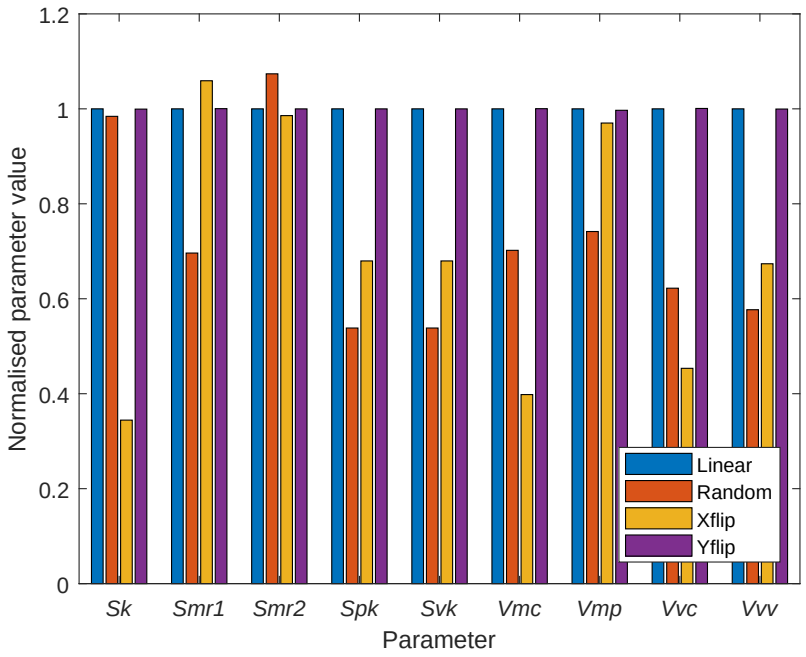


Fig. 4.8 Parameter values obtained by software B for four datasets with re-ordered height values. The values are normalised to the linearly ordered dataset values.

4. Random order.

Each of these datasets were processed by the three parameter calculation software packages. Software A and C showed no change in parameter values with different ordering, suggesting that their implementations calculating the material ratio curve and its associated parameters are independent of the arrangement of the dataset height values. As software A and C showed no change due to dataset height arrangement, they are not included in figure 4.8. Software B, however, showed significant variation in the obtained parameter values, as shown in figure 4.8. Such variation in the obtained parameter values shows a significant dependence on the structure of the surface dataset under test. Because of this, the results shown in figure 4.7 for software B are not completely representative of the software performance. A different structure will likely result in different parameter values for software B, which may or may not be closer to the mathematical values.

4.5.2 Discretisation error

When sampling the mathematical material ratio curve to create a dataset, only a finite number of evaluations are performed. Because of this sampling, information is inevitably lost when moving from a continuous to a discrete representation, resulting in potential inaccuracies in the calculation of parameter values compared to the mathematical value. In principle, higher density datasets should contain more information about the surface, and so should allow parameter values to be obtained that are closer to the mathematical definition. This comes at a cost of dataset file size and computation time.

Table 4.1 shows the effects of varying the dataset density on the calculated functional parameter values for the material ratio curve defined in equation 1. This is performed by averaging adjacent height values from a high density 2000×2000 dataset to create smaller datasets. Only one dataset height arrangement has been used here to focus the analysis solely on discretisation error and reduce the number of presented results. The volume parameters obtained by software A and B show convergence towards the mathematical value with increasing dataset density, suggesting the mathematical value could be reached given a sufficiently dense dataset. Such a result implies that the effect of discretisation is the primary contributor to the parameter value error.

The results for the remaining material ratio parameters also show convergence, however, the majority of these do not appear to be converging on the mathematical value. This is shown in figure 4.9, in which the values for the S_k, S_{pk} and S_{mr2} parameters have been extracted from table 4.1 to highlight the results more clearly. Software B shows the most obvious signs of convergence, while software A and C show more variation, particularly for S_k and S_{pk}. The lack of clear convergence on the mathematical value suggests that without the effect of discretisation, there would still remain an error in the calculated parameter values. These results would imply that there is an additional source(s) of error causing the deviation from the mathematical value. Considering computation time, which for some software packages

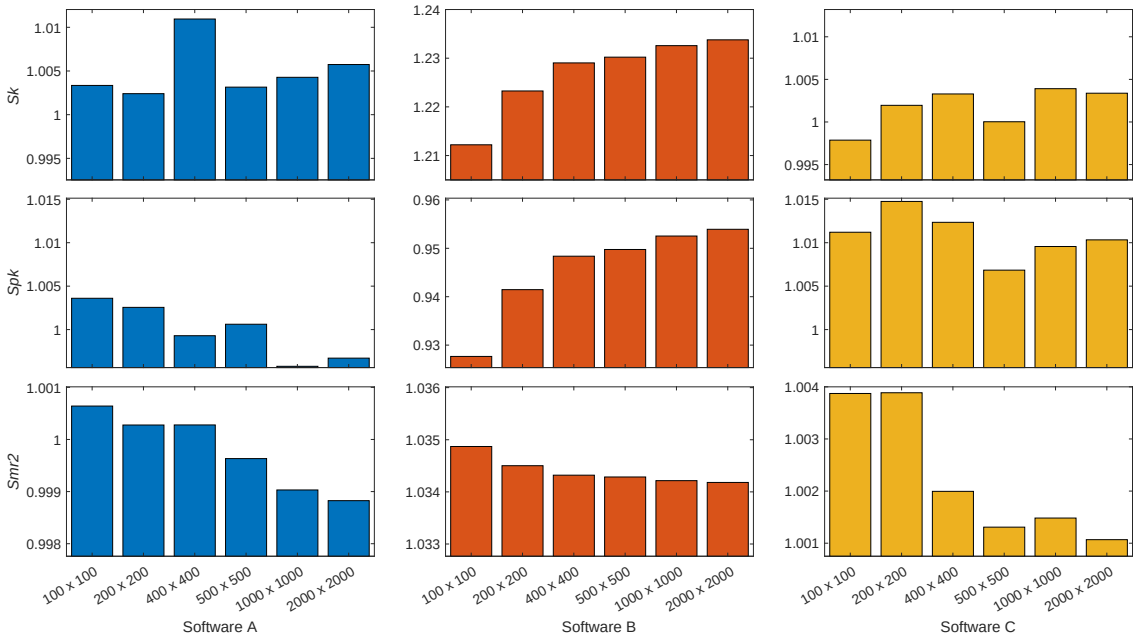


Fig. 4.9 S_k, S_{pk} and S_{mr2} parameter values obtained by each software package for a range of increasing dataset sizes given in table 4.1, normalised to the mathematical values. Note that the Y-axis scales are not consistent across all sub-figures due to the variation in ranges for different parameters and software.

can be substantially longer for higher density datasets, increasing the dataset density in order to obtain more accurate values delivers diminishing returns for these parameters.

4.6 Conclusions

The work presented in this chapter introduces a new method for validation and performance assessment of surface texture parameter calculation software, focussing on functional parameters. By using mathematical functions to define material ratio curves, surface texture parameters can be calculated accurately, respecting the definitions of the parameters given in standards. The resulting parameter values can be used as reference values against which third-party software can be compared, confident in the quality of the given reference values. In addition, the new validation method provides the first steps in developing a new way to test software whilst still leaving freedom for software developers to design their own

Table 4.1 Software parameter values for six dataset densities of the material ratio curve defined in equation 4.3.

	Dataset Size	S_k / nm	S_{pk} / nm	S_{vk} / nm	S_{mr1} / %	S_{mr2} / %	V_{vv} / ml m^{-2}	V_{vc} / ml m^{-2}	V_{mp} / ml m^{-2}	V_{mc} / ml m^{-2}
A	100 x 100	232.3751388	320.7300021	317.951784	22.02479616	77.96159525	0.0319250338	0.2640832838	0.0094609876	0.1540419062
	200 x 200	232.1569364	320.3955286	318.1639427	22.02262674	77.93326682	0.031368728	0.2601468772	0.099594052	0.1557659628
	400 x 400	234.1358484	319.3525952	317.9268969	21.84903612	77.93335822	0.0311870947	0.2584654698	0.0101664246	0.1562879802
	500 x 500	232.3300215	319.775032	318.188994	22.01160806	77.88305343	0.0309508177	0.2581453445	0.0102282322	0.1571686855
	1000 x 1000	232.5905018	318.2273088	318.6136068	22.00152831	77.83617201	0.030983133	0.2572414415	0.0103250752	0.1569428685
	2000 x 2000	232.9294072	318.5279578	317.6373201	21.98307829	77.82018191	0.0309800105	0.2574254089	0.0103050594	0.1569756711
B	100 x 100	280.7476984	296.460337	296.4603365	19.3715229	80.6284771	0.0300230981	0.2506882814	0.0100638137	0.1536847091
	200 x 200	283.3089332	300.8698233	300.8698233	19.40022442	80.59977558	0.0304516085	0.2538414938	0.0102088022	0.1554657767
	400 x 400	284.6437763	303.0784251	303.0784251	19.41438531	80.58561469	0.0306646508	0.2554477018	0.0102819812	0.1563903941
	500 x 500	284.9176365	303.5216567	303.5216567	19.41708191	80.58291809	0.0307080601	0.2557692079	0.0102970354	0.156574931
	1000 x 1000	285.4650411	304.4101723	304.4101723	19.42257231	80.57742769	0.030794243	0.2564213393	0.010326314	0.1569492127
	2000 x 2000	285.7416866	304.8551846	304.8551845	19.42525221	80.5747478	0.0308374743	0.2567479103	0.0103410824	0.1571370742
C	100 x 100	231.1062317	323.1591615	316.944183	22.1684247	78.21336898	-	-	-	-
	200 x 200	232.0525818	324.2924556	317.7125148	22.15496454	78.21445421	-	-	-	-
	400 x 400	232.3618317	323.5240095	315.963993	22.24010122	78.06707224	-	-	-	-
	500 x 500	231.6060944	321.7649768	314.2113545	22.29401772	78.01358045	-	-	-	-
	1000 x 1000	232.5050507	322.6344143	315.0575663	22.28031042	78.02717009	-	-	-	-
	2000 x 2000	232.3820801	322.8787867	315.4467954	22.2917175	77.9948555	-	-	-	-
	Mathematical value	231.6	319.5783833	319.5783833	22.08834838	77.91165162	0.03088	0.257075	0.010355	0.157325

implementations depending on their specific constraints, such as accuracy, speed or resource usage. This approach is an improvement over the current practice of using reference software, which may have the effect of forcing particular implementations upon software developers.

By using the material ratio curve as the basis of the definition of the surface, discrete dataset representations of the surface can be created by sampling the curve. These datasets can be input into third-party software and used to test the performance of the surface texture parameter calculation methods and algorithms used. Additional work was carried out to analyse the created datasets and assess whether the methods used to produce a discrete representation of the surface significantly impact the functional parameter values obtained by the software. This work revealed a dependence on dataset height order for software B.

The results of the comparison performed in this chapter reveal differences between the parameter values obtained by software and the values calculated mathematically. The reasons for these differences are unknown, as they are subject to the specific algorithms used within the software. Irrespective of the reasons, the differences highlight a need to move toward utilising mathematically defined surfaces to provide traceable methods of verifying the calculation of surface texture parameters. This approach can be combined with existing

methods, which allow for the assessment of more realistic surface datasets, to provide a more complete and traceable method of surface texture parameter calculation validation.

Chapter 5

The development of mathematical references for field surface texture parameters

5.1 Introduction

The development of mathematical references for the validation of surface texture parameter calculation software is a valuable step towards providing users with a traceable assessment of the parameter values they obtain. By moving away from reference software, which is subject to differing interpretations of algorithms, users can have more assurance in the performance of the software under test and deliver more accurate specification and analysis of high-precision parts.

Chapter 4 introduced the use of mathematical references for functional areal surface texture parameters, showcasing this new mathematical approach. As all functional surface texture parameters are derived from the material ratio curve and not the surface heights directly, it was logical to define the material ratio curve analytically for direct calculation of the parameters and obtain surface datasets by sampling the curve. For field surface texture

parameters, the direct surface height information is needed, and so analytical definitions of the surface height, $z(x,y)$, are required.

The work presented in this chapter develops a framework for creating simulated surfaces for use in obtaining reference surface texture parameter values. The framework is incorporated into a graphical user interface (GUI), presented in detail in appendix D. A series of simple test surfaces, created using a variety of analytical functions, are used to showcase the evaluation of field surface texture parameters. In addition, an alternative numerical method is presented to overcome the current limitations of the analytical methods and obtain mathematical parameter values for a series of complex test surfaces. Parameter values obtained by third-party metrology software are compared to the reference values obtained in both cases to showcase the use of the approach.

5.2 Analytical surface representations

There are variety of established methods of generated surfaces. One example, which produces a wide range of spatial frequency components and thus produces realistic looking surfaces, is fractal surfaces [72]. Methods include fractional Brownian motion and random midpoint displacement [73, 74], which offer reliable methods of multiscale surface generation, however, each of these are stochastic by nature, and cannot generate continuous analytical surface height definitions as a function of position x,y, suitable for input into the surface texture parameter definitions.

The field of areal surface texture operates using measurement datasets that are commonly referred to as '2.5D', that is, an array of height values spread across a uniformly-spaced grid on an x,y plane. Algebraically, this dataset can be thought of as samples of a function $z(x,y)$ in which the surface height, z, is defined in terms of the position in x and y. As the surface texture parameters given in ISO 25178-2 are defined in terms of surface heights for all values

of x and y, it follows that they are given as operations on the function $z(x,y)$. Therefore, it is required that analytical surface representations are given in a similar form.

Surfaces of the form $z(x,y)$ can be defined in a number of ways. Deterministic surface height expressions can be achieved using any combination of elementary functions. One such example are polynomials, which combine scaled power orders of x and y to define a surface wherein the number of critical points is determined by the highest order term. A positive of using polynomial surfaces is that the individual terms within any given surface expression are relatively simple, which can lead to simple mathematical calculations when obtaining surface texture parameter values. However, it can be difficult to design particular surface structures by defining polynomial terms. In order to develop a re-usable framework, a structure must be chosen that can be built upon to produce a wide variety of surface textures. Any real-valued, continuous function can be approximated by a Fourier series: a periodic function comprised of a weighted summation of harmonic sinusoids. By utilising Fourier series, it is possible to create any desired continuous function by selecting the right combination of sinusoid functions. However, not all surfaces can be described using Fourier series. Discontinuous surfaces features such as sharp steps, which can be found on real surface measurements, are not well-approximated by a finite Fourier series. Despite this drawback, a Fourier series framework delivers the best combination of surface variability and mathematical simplicity, whilst still conforming to the $z(x,y)$ requirements for use with surface texture parameters.

The analytical surface creation introduced in this work uses the Fourier series as the foundation for creating surface functions of the form

$$z(x,y) = \sum_{n=1}^{N} \sum_{m=1}^{M} A_{n,m} \cos\left(2\pi f_s \left(\frac{m\,(x+\phi_x)}{N_x} + \frac{n\,(y+\phi_y)}{N_y}\right)\right), \tag{5.1}$$

where $N \times M$ is the total number of terms; m and n denote the x and y spatial frequency components of each term, respectively; $A_{n,m}$ defines the amplitude of each term; N_x and N_y represent the size of the surface in the x and y directions as multiples of the scaling

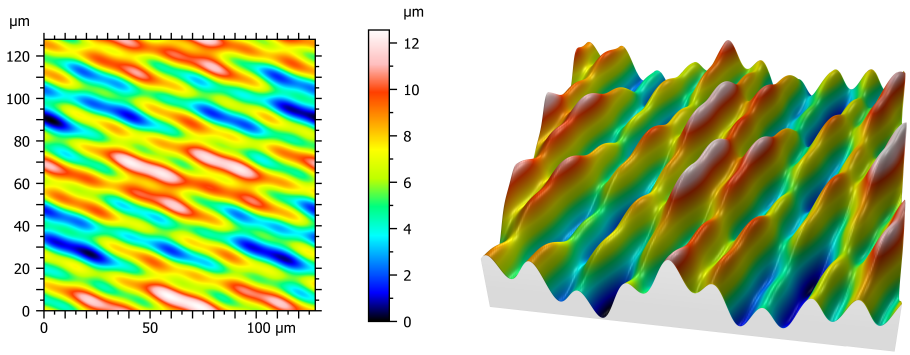


Fig. 5.1 Simulated surface generated using five cosine terms in the Fourier series structure.

length $1/f_s$; and ϕ_x and ϕ_y are the phase shifts of the cosine term in the x and y directions, respectively. For ease of surface design, the area over which the surface function is defined, $N_x/f_s \times N_y/f_s$, has been incorporated into the surface equation to enable integer values of m and n to correspond to harmonic wavelengths within the surface area. By combining a number of these terms together, a variety of surface shapes can be made, each well-defined and with a simple structure. An example surface is shown in figure 5.1, utilising the software presented in detail in appendix D, which has an analytical definition comprised of five cosine terms.

5.3 Mathematical parameter evaluation

5.3.1 Parameter evaluation

To ensure mathematical traceability to analytical surface representations, the surface texture parameter calculations must also be performed analytically. To perform analytical, or symbolic, calculations computationally, a computer algebra system (CAS) is required. A CAS is a mathematical software package or module that can manipulate symbolic mathematical expressions directly, typically using a series of rules or look-up tables dictating mathematical procedures, without relying on numerical approximation. Such an approach

delivers seemingly infinite precision results, providing the system is successfully able to obtain a mathematical expression, and ensures mathematical traceability throughout the operation. There are a variety of CASs available, and while some are open-source such as SageMath [75], the systems that are most popular are closed-source and commercial products. For the majority of the work in this chapter, Wolfram Mathematica 11.1 was used as the primary CAS, with MAPLE 2017 also used to verify the results of Mathematica for first time calculations (subsequent calculations using a similar mathematical form did not require MAPLE verification) [76, 77]. Both of these are among the most capable CASs available at the time of writing.

The calculation of each parameter requires adhering to the definitions given in ISO 25178-2 [1, 78]. In an ideal situation, this means substituting the surface equation for $z(x,y)$ in the parameter definition and evaluating the result. The following sections give the mathematical operations used to calculate each parameter, and where necessary, any alternative steps taken to circumvent limitations of the CAS used.

Before any parameter calculation can be performed, the surface function must be adjusted so that it has a mean-plane of zero. This is achieved by simply finding the mean height across the entire evaluation area of the surface and subtracting it from the surface function, given by

$$z(x,y) = z_0(x,y) - \frac{1}{A}\iint_{A_R} z_0(x,y)\mathrm{d}x\mathrm{d}y, \tag{5.2}$$

where $z_0(x,y)$ is used to represent the unadjusted surface function, the integration region, A_R, is the defined evaluation area of the surface, A is the total evaluation area, and $z(x,y)$ will represent the adjusted surface function in all future instances.

5.3.1.1 *Sq*, *Ssk* and *Sku*

The first three parameters, *Sq*, *Ssk* and *Sku*, represent the root mean square, skewness and kurtosis of the surface, respectively [1]. The definitions for each of these have similar forms,

relying on an integral of some power function of $z(x,y)$. Such an operation is well supported by most modern CASs, and so these parameters can be evaluated directly using the original definitions,

$$S_{\mathrm{q}} = \sqrt{\frac{1}{A}\iint_{A_R} z^2(x,y)\mathrm{d}x\mathrm{d}y}, \tag{5.3}$$

$$S_{\mathrm{sk}} = \frac{1}{S_{\mathrm{sq}}^3}\left(\frac{1}{A}\iint_{A_R} z^3(x,y)\mathrm{d}x\mathrm{d}y\right) \tag{5.4}$$

and

$$S_{\mathrm{ku}} = \frac{1}{S_{\mathrm{sq}}^4}\left(\frac{1}{A}\iint_{A_R} z^4(x,y)\mathrm{d}x\mathrm{d}y\right). \tag{5.5}$$

As both *Ssk* and *Sku* rely on *Sq* in their definitions, *Sq* is calculated first, and the obtained expression substituted into the expression for *Ssk* and *Sku*. In the definitions given above, the double integral is performed over the defined area, A_R, which is the chosen domain of the continuous analytical expression $z(x,y)$ over which the surface area is defined. A_R is assumed to be a rectangle defined by the interval $[xl,xh]$ in the x direction and $[yl,yh]$ in the y direction.

5.3.1.2 *Sp*, *Sv* and *Sz*

The next three parameters, *Sp*, *Sv* and *Sz*, are conceptually simple in that they relate to the largest peak/valley height values on the surface [1]. *Sp* corresponds to the largest peak height within the defined area, *Sv* corresponds to the magnitude of largest valley depth, and *Sz* is the total distance between the largest peak height and largest valley depth. For a discrete dataset, maximum peak/valley heights are obtained trivially by sorting the discrete heights and obtaining the highest and lowest values.

For a continuous analytical surface, obtaining maximum peak/valley heights symbolically becomes more complex. The parameters require calculating the global maxima and minima of the function within the specified region. Several optimisation techniques can potentially obtain values corresponding to only local minima, not global minima, and so care must

be taken to avoid this. Modern CASs have many global optimisation functions available, each with strengths and weaknesses, which can be adopted to find these global maxima and minima [79].

A more manual approach to obtaining the largest peak and valley can be performed by finding all local minima and maxima points within the defined area using first and second derivatives and sorting the results. Local minima/maxima are found at values of $z(x,y)$ with zero gradient values in both x and y,

$$\frac{\partial z(x,y)}{\partial x} = 0 \tag{5.6}$$

$$\frac{\partial z(x,y)}{\partial y} = 0. \tag{5.7}$$

Furthermore, local maxima are found at locations where the second derivative of the function $z(x,y)$ is less than zero, and local minima are found where the second derivative is greater than zero, given by

$$\frac{\partial^2 z(x,y)}{\partial x^2} < 0 \tag{5.8}$$

$$\frac{\partial^2 z(x,y)}{\partial y^2} < 0 \tag{5.9}$$

and

$$\frac{\partial^2 z(x,y)}{\partial x^2} > 0 \tag{5.10}$$

$$\frac{\partial^2 z(x,y)}{\partial y^2} > 0, \tag{5.11}$$

respectively. By solving these equations simultaneously, values for x and y can be found that correspond to each local maxima and minima. By evaluating each of these points in $z(x,y)$, expressions for surface height can be found, which can then be sorted to obtain the largest peak and valley.

Using the method of minimising the derivatives of a surface, any values that are not located at a zero-gradient in both x and y will not be found. This can miss the true *Sp* or *Sv* values if they happen to be located on the boundary of the defined area, where the zero-gradient for both x and y is not necessarily satisfied. Visual inspection of the surface can help to identify if this is the case for any individual scenario. If so, simple one-dimensional gradient analysis along the boundary line, at $z(x, yl)$ for example, can be performed for each of the four boundary edges to find any zero-gradient locations, which can then be evaluated to find the expressions for the heights. Finally, direct evaluation of $z(x,y)$ at each for the four boundary corners can be performed to ensure any potential maxima/minima are identified there, as these points may not satisfy the boundary gradient analysis conditions, for similar reasons as before.

5.3.1.3 *Sal* and *Str*

Sal and *Str* are part of a subset of field parameters called spatial parameters [1]. Spatial parameters address the similarities of a surface with itself if spatially translated by a certain amount in the x and y directions. Such an analysis is valuable in identifying the uniformity of the surface. *Sal* assesses how abruptly a surfaces height changes, and *Str* gives the ratio of fast-changing directions to slow-changing directions, indicating the presence of any directionality of the surface texture, such as grinding marks.

These parameters are calculated on the autocorrelation function of the surface, $f_{ACF}(t_x, t_y)$. The autocorrelation function describes the degree of agreement between the surface function, $z(x,y)$, and the same surface translated by t_x and t_y in the x and y directions, respectively. The definition for the autocorrelation function is given in ISO 25178-2 as

$$f_{ACF}(t_x, t_y) = \frac{\iint_{A_R} z(x,y) z(x-t_x, y-t_y) \mathrm{d}x\mathrm{d}y}{\iint_{A_R} z(x,y) z(x,y) \mathrm{d}x\mathrm{d}y}. \tag{5.12}$$

The denominator of equation 5.12 is used to normalise the autocorrelation function to give a value of 1 at $t_x = 0$ and $t_y = 0$. Calculation of the autocorrelation function requires evaluation of integrals similar to *Sq*, *Ssk* and *Sku*, and thus can be performed using a CAS.

Sal and *Str* are obtained by finding the distance it takes for f_{ACF} to decay from 1 to a specified value, s, defaulting at $s = 1/5$ in ISO 25178-3 [40]. This is achieved by finding the expression for the contour line of

$$f_{ACF}(t_x, t_y) = 1/5. \tag{5.13}$$

Sal is defined by the shortest distance from the origin, $(0,0)$, to the contour line. The distance from the origin to a point on the contour is given by

$$D = \sqrt{t_x^2 + t_y^2}. \tag{5.14}$$

By rearranging equation 5.13 in terms of t_y, and substituting into equation 5.14, it is possible to obtain an expression in one dimension that gives the distance from the contour to the origin. This distance can then be minimised within the defined surface area to find the shortest distance to the origin, corresponding to *Sal*. Maximum points can be found in a similar fashion, and the ratio of shortest distance to longest distance gives the value for *Str*.

5.3.1.4 *Sdq* and *Sdr*

Sdq and *Sdr* are hybrid parameters, and related to the slope of the surface [1]. These parameters assess the steepness of features on the surface and can give valuable information for surfaces that could otherwise present similar *Sa* or *Sq* values. Hybrid parameters can be useful for the assessment of sealing performance and wettability, or any application that requires understanding of average surface gradient values.

These parameters require calculation of the surface gradients in the x and y directions, which can be obtained using partial derivatives as shown in section 5.3.1.2 for calculating *Sp* and *Sv*. Once the partial derivatives are calculated, *Sdq* can be found by evaluating the definition

$$S_{dq} = \sqrt{\frac{1}{A}\iint_{A_R}\left[\left(\frac{\partial z(x,y)}{\partial x}\right)^2 + \left(\frac{\partial z(x,y)}{\partial y}\right)^2\right]\mathrm{d}x\mathrm{d}y}, \tag{5.15}$$

which can be performed similarly to that discussed in section 5.3.1.1 for *Sq*, *Ssk* and *Sku*. *Sdr* has the definition

$$S_{dr} = \frac{1}{A}\left[\iint_{A_R}\left(\sqrt{\left[1+\left(\frac{\partial z(x,y)}{\partial x}\right)^2 + \left(\frac{\partial z(x,y)}{\partial y}\right)^2\right]} - 1\right)\mathrm{d}x\mathrm{d}y\right], \tag{5.16}$$

which, while appearing similar in composition to *Sdq* at first glance, poses a problem to CASs due to the square root of disparate terms within the integral being inseparable. Calculating *Sdr* using this definition, for the general case of an analytical surface expression of any form, requires the object within the integral to be evaluated as a single object as it is inseparable, leading to incalculable symbolic expressions and hanging run-times due to seemingly endless cycles of techniques such as integration by parts, wherein an integral can be simplified by performing an integral on the derivative of some part of the original function (in this instance, the derived function would still be as complex as the original function, requiring further integration by parts). Equation 5.16 takes the form of a nonelementary integral, which is an integral of an elementary function whose solution is not elementary (an elementary function is a function comprised of a finite number of arithmetic operations on traditional functions such as powers, roots, trigonometric functions, exponential functions and logarithmic functions). The concept of a nonelementary integral is proven in Louiville's theorem [80, 81], and evaluations of nonelementary integrals require the use of either Taylor series expansion (to a required order), numerical integration or nonelementary 'special' functions (such as incomplete elliptical integrals or error functions). Both Taylor expansion and numerical

integration are approximations, giving the result to a finite precision. Nonelementary special functions are advanced mathematical concepts that cannot be expressed using regular elementary functions, making it challenging to promote widespread use and adoption, and so will not be used here.

Practically, *Sdr* is the ratio between the area of evaluation of the surface, A, and the total topographical area of the surface itself. The topographical area can be approximated by splitting the surface into a series of triangles, ABC, and summing the area of each of the triangles using the formula

$$\text{Area} = \frac{1}{2}|\hat{u} \times \hat{v}|, \tag{5.17}$$

where $\hat{u}$ is the vector that joins point A to point B, and $\hat{v}$ is the vector that joins point A to point C. By sampling a grid of height values from the surface expression, splitting each square defined by four height values into two triangles where the height values describe the vertices, and evaluating the cross product defined in equation 5.17, an expression can be derived for the approximation of the total topographical area of the surface,

$$\text{Area} = \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{\delta x \delta y}{2}\left(\sqrt{1+\frac{(Z_{n+1,m}-Z_{n,m})^2}{\delta x^2}+\frac{(Z_{n,m+1}-Z_{n,m})^2}{\delta y^2}}+ \sqrt{1+\frac{(Z_{n+1,m+1}-Z_{n,m+1})^2}{\delta x^2}+\frac{(Z_{n+1,m+1}-Z_{n+1,m})^2}{\delta y^2}}\right), \tag{5.18}$$

where N and M are the total number of vertices in the x and y directions, respectively, and δx and δy are the widths between adjacent vertices in the x and y directions. $Z_{n,m}$ describes the surface height values at each vertex,

$$Z_{n,m} = Z(x_n, y_m), \tag{5.19}$$

where

$$x_n = x_0 + (n-1)\delta x \tag{5.20}$$

$$y_m = y_0 + (m-1)\delta y. \tag{5.21}$$

Here, x_0 and y_0 are the coordinates of the starting corner of the surface area of interest. By using this method with an analytical surface expression, it is possible to define δx and δy to be any desired value. δx and δy can be reduced to the infinitesimal case, allowing a value for the total topographical area to be found that is equivalent to a symbolic evaluation. In practice, this would require seemingly infinite computational resources, and so compromises must be made, instead calculating to a precision that is appropriate for the required application. This approach is an alternative workaround to the limitations of calculating *Sdr* in the general case for any surface, however, it sacrifices the mathematical traceability by resorting to discrete, numerical methods. An alternative method, that calculates *Sdr* analytically for a specific case, in presented in section 5.5.4.

5.3.1.5 *Sa*

Sa is one of the most popular areal surface texture parameters in industry [82]. *Sa* is simply the mean deviation of height values on the surface from the mean plane and is broadly used in industry to define the 'roughness' of a surface, as a high *Sa* value would correspond to a larger spread between high and low points on a surface. However, due to the simplicity of the definition, the practical application of this metric, for example to understand lubricant retention, is limited, as surfaces with deterministic elements can skew the *Sa* value.

Mathematically, *Sa* is defined as the arithmetic mean of the absolute of the surface heights within the evaluation area,

$$S_a = \frac{1}{A}\int_{A_R} |z(x,y)| \mathrm{d}x\mathrm{d}y. \tag{5.22}$$

Unfortunately, due to the absolute function, direct integration of the analytical surface expression is not possible.

All areas of the surface that lie below the mean plane are effectively mirrored to the positive z-axis, causing discontinuous edges at the points where the surface meets the mean plane. These discontinuities cannot be integrated as a single function. For simple cases, it is possible to separate the full discontinuous expression into a series of region-bound expressions and evaluate them all individually, however, a new expression is required for each region, along with information about the boundaries for each region (which are not necessarily linear, thus requiring line integrals). For more complex surfaces with larger numbers of mean plane crossing points, the formulation of each discrete term becomes prohibitively complex, particularly as the process would be bespoke for reach individual surface expression, and not something easily implementable within a CAS. An alternative solution for simple parametric surfaces has been implemented in section 5.5.

5.3.2 Simple surfaces

A selection of ten simple analytical surfaces were defined for use in showcasing the calculation of field surface texture parameters. In order to highlight the variety of surface functions that can be used, three different types of function were selected, each defined using elementary functions that can be given in the form $z(x,y)$, suitable for use with field surface texture parameter definitions:

- Three cosine functions, using the technique introduced in section 5.2. Cosine functions were chosen as they are an effective method of creating continuous functions that are periodic and can be easily combined to increase complexity, as discussed in section 5.2. The three functions chosen are comprised are two, three and four terms, each with increasing complexity in order to test the software with increasing amounts of surface height variation.

- Five polynomial functions, comprising of a cubic polynomial and four Bernoulli polynomials (two fourth order and two fifth order). Polynomial functions, as discussed in section 5.2, are functions with a simple composition of multiple order power terms. One negative of polynomials is their aperiodicity, leading to rapidly increasing magnitudes across a range of x, y values, which is not a good simulation of surface height measurements. A solution to this is the use of Bernoulli polynomials, which deliver a much more periodic structure within a defined positive range, approaching sinusoids at high orders [83, 84]. Fourth and fifth order Bernoulli polynomials were selected here for the simplicity of the functions whilst still providing an increased level of topographical complexity to simple cosine functions. Both separable and inseparable versions of the Bernoulli polynomials were included. An additional regular cubic polynomial was included to test the software on a non-periodic surface.

- Two Gaussian functions, using the natural exponential function, e. Gaussian functions were chosen specifically for the calculation of autocorrelation parameters *Sal* and *Str*, using the methods given in section 5.5.1. This method allows for a range of ratio between the longest and shortest decay length to a specified height (i.e an elliptical contour) to be created [1]. For these test surfaces, one round contour and one contour with high ellipticity was created to test the software at the upper and lower range of decay length ratios.

Details of a selection of example surfaces, including their definition, range of evaluation, and a representative image can be found in appendix E.

5.3.3 Assessment of third-party software

To showcase the validity of this new approach, comparisons were performed with a selection of third-party parameter calculation software packages. A total of four different software packages were used, labelled A to D, which included a combination of commercial software

and software developed by NMIs. Each software package was given a 700×700 high resolution SF surface .SDF datasets sampled from the surface expressions defined in section 5.3.2, where resolution is defined as the number of individual height values included in the surface dataset. This resolution was chosen because software D suffered from an upper limit on the file size of datasets of 10 MB, and a resolution of 700×700 was the highest resolution that reliably came under this limit, ensuring comparisons to be possible across all four software packages. No additional form removal or filtration operations were performed to enable sole assessment of the parameter value calculation methods.

The results of each of the third-party software tests are normalised relative to the mathematically-obtained parameter values to allow for more direct comparisons. Figure 5.2 shows the normalised values obtained for the *Sq* parameter, relative to the mathematical value, for a selection of eight analytical surfaces. Upon initial investigation, all software packages perform well in calculating *Sq*, however, closer inspection shows deviation from the mathematical value of the order of tenths of a percent, particularly for software B and D. This level of deviation could prove important when dealing with high precision applications and helps metrology software developers to identify the aspects of their software that perform worse than competitors, highlighting key areas where their software requires improvement.

Figures 5.3 and 5.4 show the results for the *Sp* and *Sdq* parameters, respectively. All software packages under test show good agreement with the mathematically-obtained values for *Sp*, which is expected due its simple definition of maximum peak height within the evaluation area. Discrete-based software can obtain the largest height value in a surface dataset easily with simple sorting algorithms for the height values (although this may not correspond to the true peak, depending on where the continuous surface was sampled). In contrast, the *Sdq* parameter values show more significant differences from the mathematically-defined values, although these variations are still within a tenth of a percent of the mathemtically-obtained value. Despite these deviations, three of the four software packages perform similarly. The

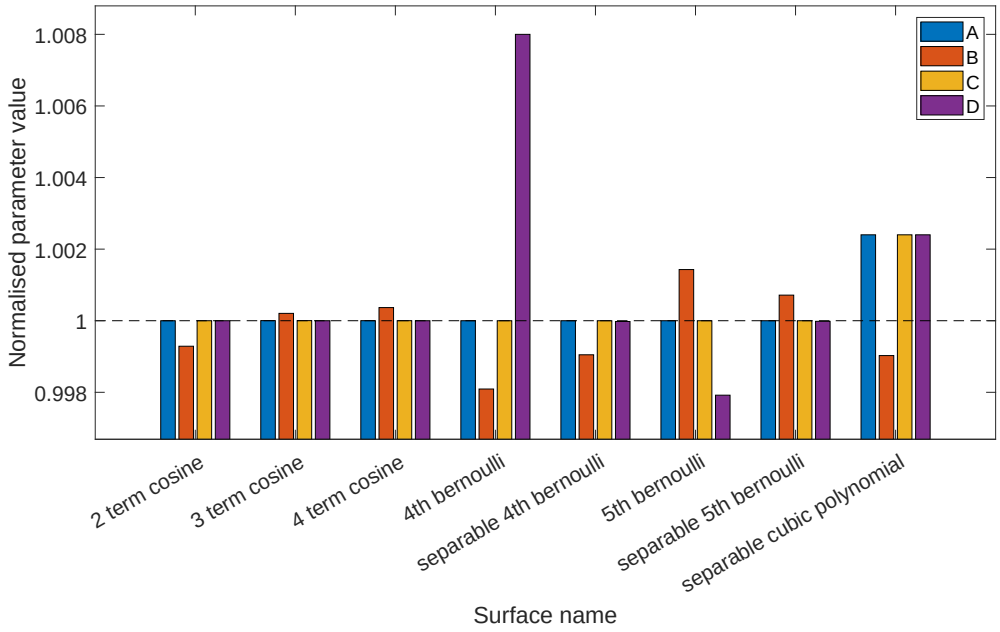


Fig. 5.2 Software obtained values for the *Sq* parameter, normalised to the mathematically-obtained value, for a series of analytically-defined surfaces.

definition of the *Sdq* parameter is based on the average gradient of the surface, which can be highly dependent on the scale at which the surface is measured [43]. This effect can also be affected by the resolution of the surface, and so it should be considered that the 700×700 dataset sampling resolution of the analytical surfaces is a primary reason for the deviations.

The results for autocorrelation parameter *Sal* are given in figure 5.5, for a selection of six analytical surfaces for which the autocorrelation parameters were calculable. In comparison with the previous results, the scale of deviations from the mathematically-obtained value is immediately apparent, with differences of approximately 10% common. This suggests the increased complexity and higher number of algorithmic steps required to calculate the parameter, relative to the simpler parameters, is contributing to the deviations in the values obtained by the software under test. It should be noted here that only software packages A and B provided the option to calculate autocorrelation parameters, and so results for only two of the four software packages under test are presented here.

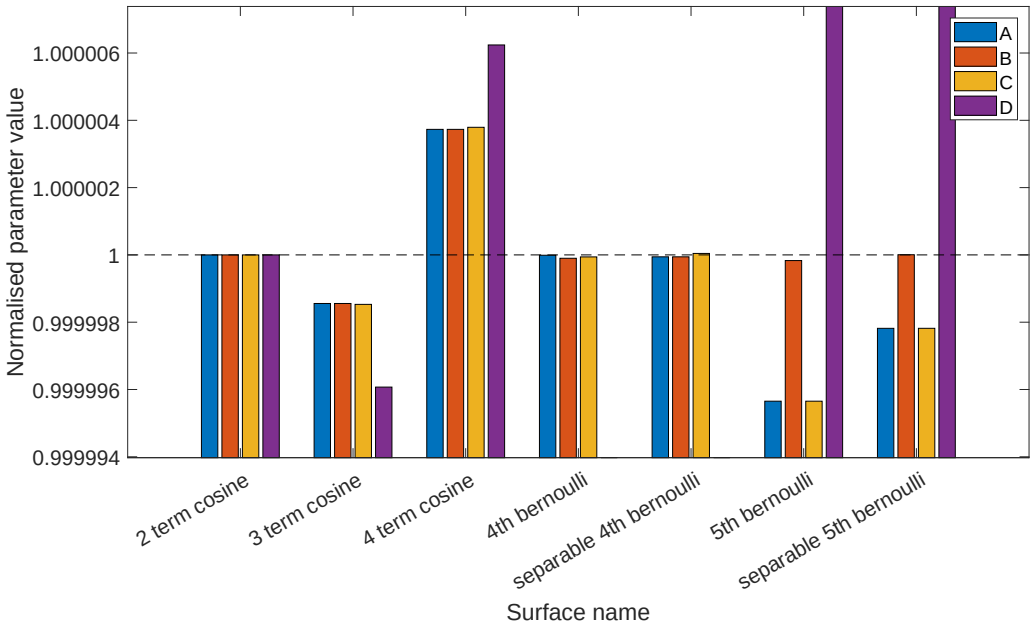


Fig. 5.3 Software obtained values for the *Sp* parameter, normalised to the mathematically-obtained value, for a series of analytically-defined surfaces.

Several of the results presented here reveal situations where several of the software obtained values show good agreement with each other, despite differing from the mathematical value. A good example of this can be seen in the bottom graph in figure 5.4. This can be caused by a variety of reasons, including choice of algorithms or discretisation error in the dataset. This agreement between software can be used to argue that a mathematical reference standard is unnecessary; if all software is in agreement then multiple measurements can still be compared to each other, using the commonly obtained values as an agreed 'standard', and collaborations across users with different software can be performed without issue. In this scenario, the amount that all of the software packages deviate from a mathematical reference is not important.

The problem with this approach, however, is that it risks the industry being forced to use one particular algorithmic implementation in order to conform, regardless of whether alternative methods are technically more accurate. By working to a mathematically traceable

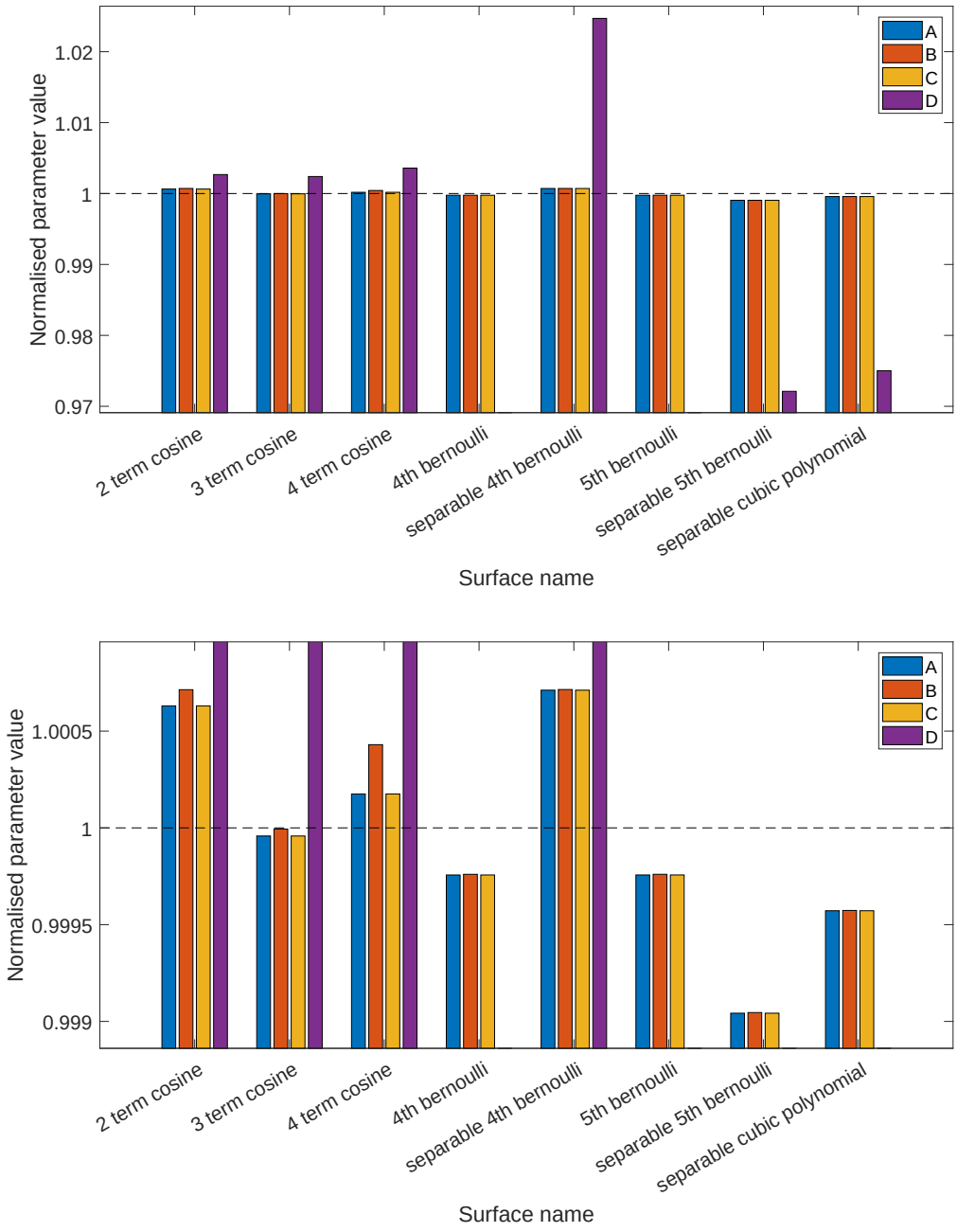


Fig. 5.4 Software obtained values for the *Sdq* parameter, normalised to the mathematically-obtained value, for a series of analytically-defined surfaces. *Top*: Full view of the results. *Bottom*: Zoomed-in view of the results for software A, B and C.

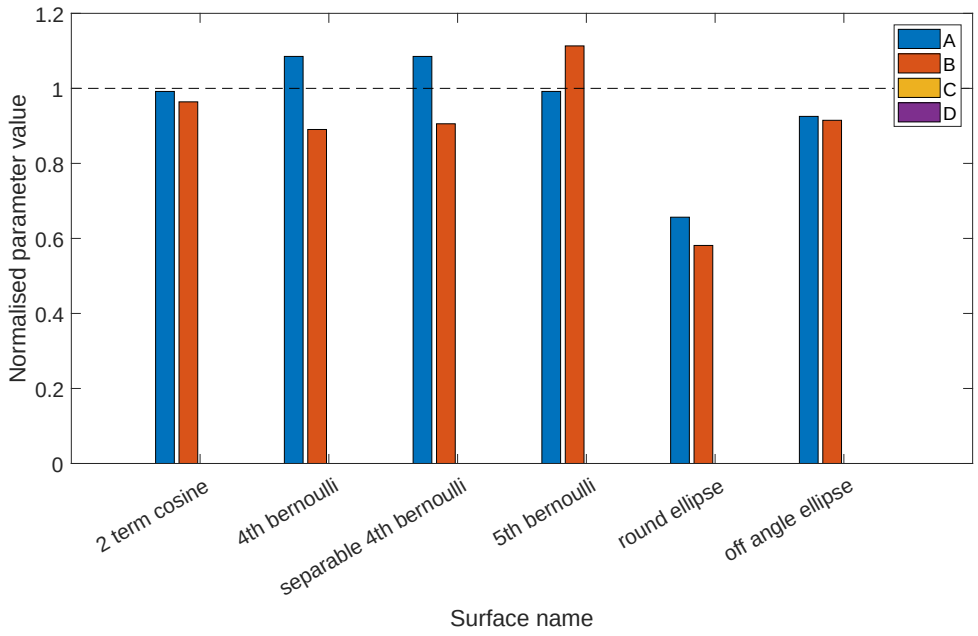


Fig. 5.5 Software obtained values for the *Sal* parameter, normalised to the mathematically-obtained value, for a series of analytically-defined surfaces.

standard, software developers are encouraged to find methods that improve upon the current implementations, deviating from the common 'standard' and obtaining results that are objectively more accurate. Furthermore, traceability can be established to a mathematical reference, removing dependency on comparing one's algorithms to others.

This novel method of software assessment by using mathematically-obtained reference values has demonstrated an ability to identify areas of the software under test that would benefit the most from improvement. By identifying which parameters deviate the most from a mathematical reference, in comparison to other software packages, resources can be allocated more efficiently, ensuring the weakest areas of the software are addressed first. The amount of information given to the software via a discrete dataset input can put a limit on the software's ability to obtain a parameter value close to the mathematical reference value. However, by performing an assessment at several resolutions of input surface dataset, software developers can identify to what extent the deviation from the mathematical parameter value is due to

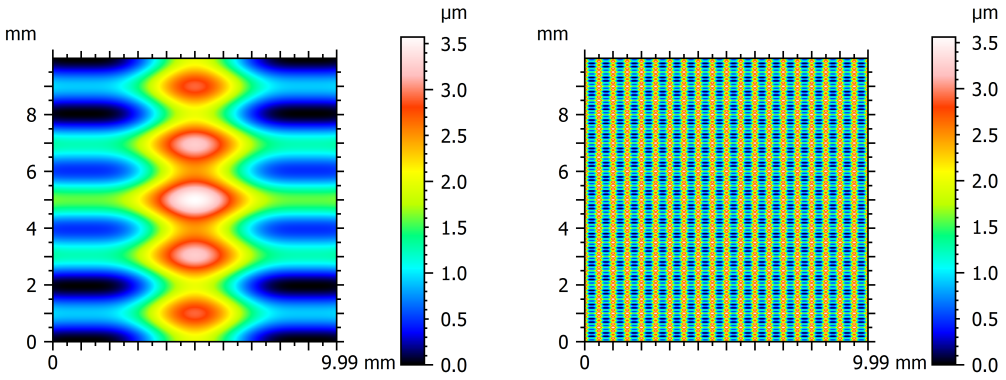


Fig. 5.6 Representations of four term cosine surfaces with varying spatial frequency component multiplication factors. *Left*: 0.5×. *Right*: 8×.

discretisation, and what is due to shortcomings in the software. This idea is addressed further in chapter 6.

5.3.3.1 Effect of spatial frequency on software parameter calculation

An additional benefit of creating surface texture parameter reference values using a mathematically traceable approach is the ability to design surfaces that have particular properties and analyse the effect that those properties may have on the algorithms used by surface texture analysis software.

In this section, a series of five surfaces have been defined that have different spatial frequency components. Starting with a four-term cosine surface, created using the method as described in section 5.2, each subsequent surface has had each spatial frequency component multiplied by a constant factor, ranging from 0.5× to 8×, as shown in figure 5.6. As the only difference between the surfaces is the multiplication factor of the spatial frequency components present, analysis into the performance of software is able to understand the effect of spatial frequency on software parameter calculation.

Figure 5.7 shows the results for the *Ssk* parameter for software A, B and D (at time of writing, software C was no longer available). The primary result is the significant variation

presented by software B, which differs from the mathematical reference by almost 3% for the high spatial frequency surfaces. Variations are still significant, however, even for the lower spatial frequency surfaces, suggesting that the frequency components are not the sole contributor for the variation. Zooming in to focus on the results for software A and D, good agreement with the mathematical reference values is presented, again with no perceivable association with the spatial frequency of the surfaces. This suggests that the algorithms implemented for the calculation of *Ssk* are mostly unaffected by spatial frequency. Similar trends are also found for the *Sq* and *Sku* parameters. Each of these parameters are defined using mathematically similar definitions, integrating some power expression of the surface over the evaluation area. It is therefore unsurprising that the results for these parameters show similar trends for mathematically similar surface expressions.

Figure 5.8 shows the spatial frequency variation results for the *Sv* parameter. In contrast to the results for *Ssk*, a spatial frequency dependency can be seen here. For software packages A and D, the deviation from the mathematical reference value increases as the spatial frequency components present in the surface increase. One possible contributing factor to this could be the effect of discretising the mathematical surface expression to produce a discrete dataset. As the spatial frequencies of the surface increase, the relative sizes of the peaks and valleys present on the surface decrease and, consequently, are represented on the discrete surface by a fewer number of pixels. This increases the likelihood of not representing the true peak or valley of the surface. Software B, on the other hand, shows no dependence on the spatial frequencies of the surface, and instead presents *Sv* values that agree well with the mathematical references. This is likely due to the expected interpolation techniques used in software B, which would be able to approximate peak/valley magnitudes in regions with resolution limitations.

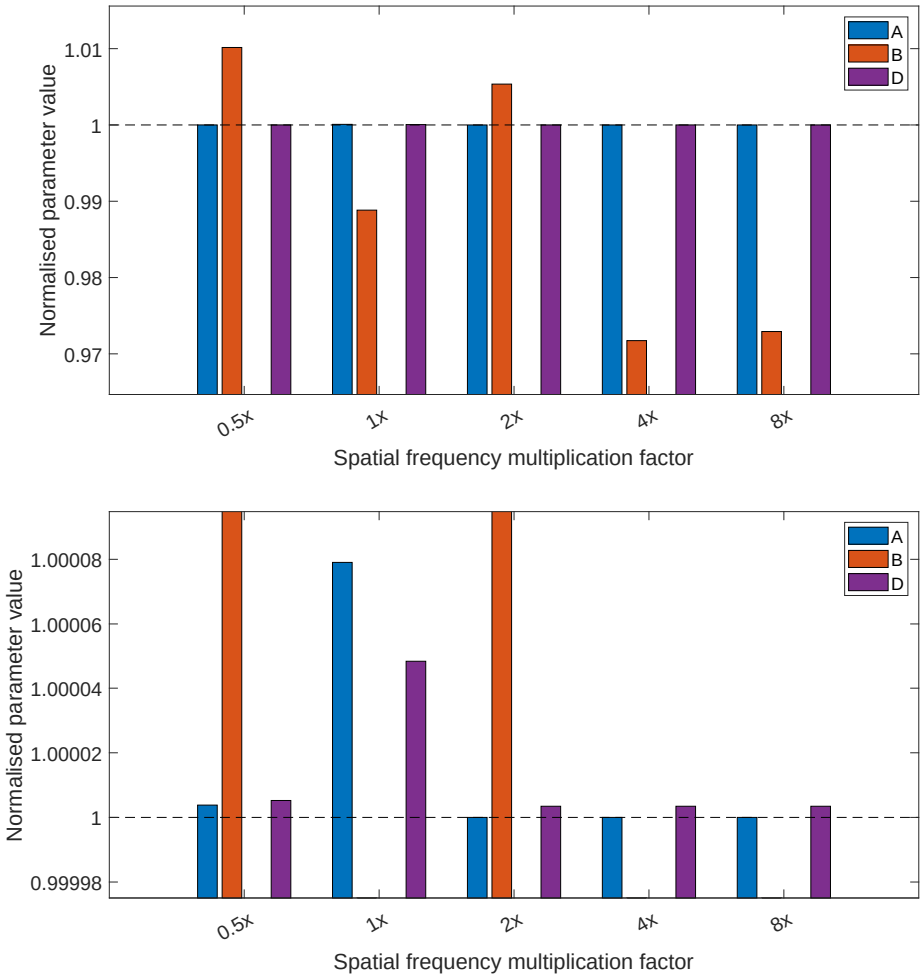


Fig. 5.7 Software obtained values for the *Ssk* parameter, normalised to the mathematically-obtained value, for a series of four term cosine surfaces with increasing spatial frequency components, for software A, B and D. *Top*: Full view of the results. *Bottom*: Zoomed-in view of the results for software A and D.

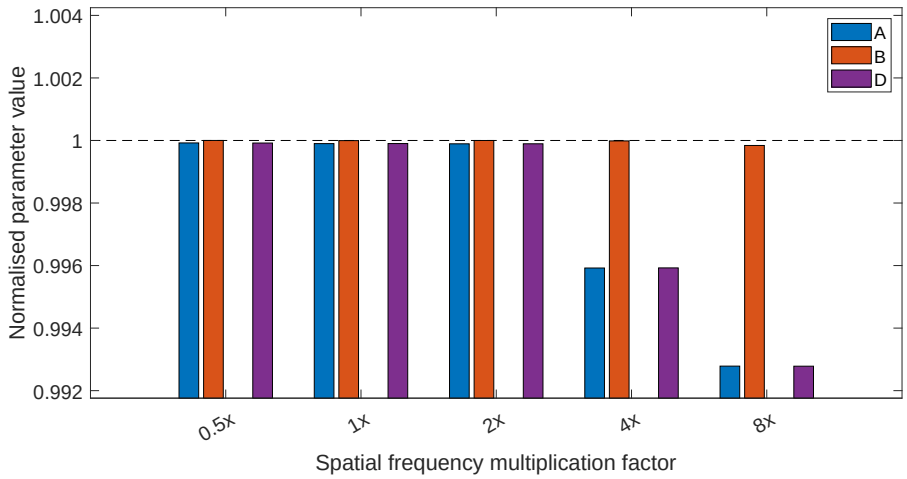


Fig. 5.8 Software obtained values for the *Sv* parameter, normalised to the mathematically-obtained value, for a series of four term cosine surfaces with increasing spatial frequency components, for software A, B and D.

5.4 Numerical parameter evaluation

As discussed in section 5.3, there are certain mathematical operations, such as those required for *Sa* and *Sdr*, that are too complex to be reliably calculated in current CASs (results may be calculable for individual cases with some manipulation, but not in a general sense). In addition, very complex analytical surfaces with a high number of mathematical terms ($>$1000) can prove to be too computationally intensive for modern high-performance computers. As a workaround to both of these options, numerical methods can be implemented that perform discrete-based calculations using numbers stored to a finite precision far less than those used in CASs. This approach can lead to approximations and inaccuracies in the results on a potentially significant scale, at the very least due to rounding errors [71].

For the purpose of assessing the performance of software, these issues can be minimised by performing reference calculations using methods with an operating precision better than that for the software under test. In this way, numerically-obtained reference values can retain their value and be reliably compared against to assess software performance, up to a certain precision. By unlocking the use of extended precision numerical methods,

complex calculations can be performed, and the scope of the mathematical references can be broadened.

5.4.1 Chebfun

To perform the extended precision calculations, the MATLAB plug-in Chebfun was used [85, 86]. Chebfun is an open source software package that utilises the idea that smooth functions can be effectively represented by expansions of Chebyshev polynomials (which are a series of recursively defined polynomials whose roots are used in polynomial interpolation techniques). Analytical functions are stored using the roots of Chebyshev polynomials, and are able to be represented accurately to machine precision, that is, fifteen to sixteen significant decimal digits. The use of Chebyshev polynomials to represent smooth non-periodic functions has been seen as an equivalent to Fourier series and its ability to represent smooth periodic functions and is a major component of the branch of mathematics known as approximation theory [87, 88]. The nature of the process is stable for even very large numbers of Chebyshev points, corresponding to complex analytical surfaces.

The aim of the Chebfun software is to achieve for functions what floating-point arithmetic achieves for numbers: rapid computation in which each successive operation is carried out exactly apart from a rounding error that is very small in relative terms [89]. This precise, numerical approach to computation with functions is an effective solution for overcoming the current computational limitations of purely symbolic calculations using CASs.

5.4.2 Comparison with mathematical evaluation

Before applying the Chebfun methods to new applications, such as complex surfaces and analytically challenging parameters, it is first worthwhile to perform a test to gain confidence in Chebfun's performance. This can be achieved by using the Chebfun method to calculate the same parameters as can be obtained symbolically using a CAS.

A selection of seven parameters were calculated for eight analytical surfaces mathematically, using the methods described in section 5.3.1. Each of the analytical results were then evaluated to fifteen significant decimal figures and stored numerically. Each analytical surface expression was then input into MATLAB and converted into a two-dimensional chebfun object over the required evaluation area. Algorithms for the calculation of each surface texture parameter were performed, utilising the relevant chebfun-specific functions. The results for each are given in tables 5.1 and 5.2 and show good agreement between the two methods. The Chebfun method is consistently able to match the mathematical result to within fourteen significant decimal figures, with the final fifteenth significant figure often only deviating by at most one digit.

Table 5.1 Obtained parameter values for analytical surface expressions from both symbolic methods and Chebfun methods.

Surface name	Method	*Sq*	*Ssk*	*Sku*	*Sp*
2 term cosine	Symbolic	1.00000000000000E-06	0.00000000000000E+00	2.25000000000000E+00	2.00000000000000E-06
	Chebfun	1.00000000000000E-06	-8.81620763116716E-17	2.25000000000000E+00	2.00000000000000E-06
3 term cosine	Symbolic	7.08960096303089E-07	1.69670632288577E-01	2.55374822035267E+00	1.69038664043398E-06
	Chebfun	7.08960096303090E-07	1.69670632288577E-01	2.55374822035268E+00	1.69038664043398E-06
4 term cosine	symbolic	8.43910029893118E-07	1.21384118148350E-01	2.10108580181232E+00	1.91356806070197E-06
	Chebfun	8.43910029893118E-07	1.21384118148350E-01	2.10108580181233E+00	1.91356806070198E-06
Cubic polynomial	Symbolic	5.34878713443747E-05	-8.69318622977992E-01	3.75227221178735E+00	1.11211299937499E-04
	Chebfun	5.34878713443747E-05	-8.69318622977993E-01	3.75227221178735E+00	1.11211299937499E-04
Separable 4th Bernoulli	Symbolic	3.08606699924183E-08	-9.56752706058658E-02	2.26929932812285E+00	5.83333333333333E-08
	Chebfun	3.08606699924183E-08	-9.56752706058660E-02	2.26929932812286E+00	5.83333333333333E-08
4th Bernoulli	Symbolic	4.76190476190476E-10	1.83075148110113E-02	2.36728582500112E+00	1.11111111111111E-09
	Chebfun	4.76190476190476E-10	1.83075148110110E-02	2.36728582500112E+00	1.11111111111111E-09
Separable 5th Bernoulli	Symbolic	2.45204119306874E-08	0.00000000000000E+00	2.24729792807811E+00	4.89163817393952E-08
	Chebfun	2.45204119306874E-08	7.18025354369594E-16	2.24729792807811E+00	4.89163817393952E-08
5th Bernoulli	Symbolic	3.00625300625300E-10	0.00000000000000E+00	2.23381677323936E+00	5.98203100618559E-10
	Chebfun	3.00625300625300E-10	4.48472620571130E-30	2.23381677323936E+00	5.98203100618560E-10

Table 5.2 Obtained parameter values for analytical surface expressions from both symbolic methods and Chebfun methods.

Surface name	Method	*Sv*	*Sz*	*Sdq*
2 term cosine	Symbolic	2.00000000000000E-06	4.00000000000000E-06	3.14159265358979E-03
	Chebfun	-2.00000000000000E-06	4.00000000000000E-06	3.14159265358979E-03
3 term cosine	Symbolic	-1.52250300534855E-06	3.21288964578253E-06	9.07822248471597E-04
	Chebfun	-1.52250300534855E-06	3.21288964578253E-06	9.07822248471598E-04
4 term cosine	symbolic	-1.65884305398862E-06	3.57241111469060E-06	2.59070341128246E-03
	Chebfun	-1.65884305398862E-06	3.57241111469060E-06	2.59070341128246E-03
Cubic polynomial	Symbolic	2.20000000000000E-04	3.31211299937499E-04	1.74355957741626E-04
	Chebfun	-2.20000000000000E-04	3.31211299937499E-04	1.74355957741627E-04
Separable 4th Bernoulli	Symbolic	6.66666666666666E-08	1.25000000000000E-07	1.95180014589706E-04
	Chebfun	-6.66666666666668E-08	1.25000000000000E-07	1.95180014589706E-04
4th Bernoulli	Symbolic	9.72222222222222E-10	2.08333333333333E-09	4.25917709999959E-06
	Chebfun	-9.72222222222225E-10	2.08333333333334E-09	4.25917709999960E-06
Separable 5th Bernoulli	Symbolic	-4.89163817393952E-08	9.78327634787904E-08	1.54303349962091E-04
	Chebfun	-4.89163817393951E-08	9.78327634787904E-08	1.54303349962091E-04
5th Bernoulli	Symbolic	-5.98203100618559E-10	1.19640620123711E-09	2.67539627961604E-06
	Chebfun	-5.98203100618559E-10	1.19640620123712E-09	2.67539627961603E-06

5.4.3 Complex surfaces

With the Chebfun approach successfully able to obtain parameter values that agree with the mathematically obtained values to machine precision (minus rounding error in some cases) for the analytical surfaces considered, the next step is to extend this approach to more complex surfaces that better simulate realistic surfaces.

A selection of five surfaces were created with increased complexity, three pseudo-random surfaces with increasing high spatial frequency components, and two modified versions of simple surfaces from section 5.3.2 that have had pseudo-random elements added. Each of these surfaces were created utilising the Chebfun framework. For each, an analytical expression is converted into a chebfun object and represented as a matrix of Chebyshev coefficients. This matrix is then manipulated, adding pseudo-random components to each element to increase the complexity of the surface and add realistic height variations. The Chebyshev coefficients in each matrix are then combined to form polynomial expressions using the recursive Chebyshev polynomial definitions

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x), \end{aligned} \tag{5.23}$$

and substituting them into the following

$$z(x,y) = \sum_{m=0}^{M} \sum_{n=0}^{N} a_{n,m} T_n(x) T_m(y) \tag{5.24}$$

to create the final polynomial expression, where $a_{n,m}$ denotes the Chebyshev coefficient matrix elements, and N and M are the size of the Chebyshev coefficient matrix in the x and y directions, respectively. Details of a selection of example surfaces are given in appendix E alongside the simple surfaces.

5.4.4 Assessment of third-party software

The five complex surfaces were converted into chebfun objects and then used to obtain surface texture parameters.

As the limitations of some symbolic calculations have been alleviated due to the numerical based approach, parameters *Sa* and *Sdr* were also calculated in addition the original set shown in section 5.3.3. Whilst the Chebfun method's calculation of the original set of parameters was verified against symbolic mathematical calculation in section 5.4.2, no such verification could be performed for *Sa* and *Sdr*. Therefore, although Chebfun may be able to calculate these parameters as accurately as the original set, caution must be taken when using the Chebfun parameter values as a reference, as errors may be present. Therefore, the calculation methods used should be explicitly stated alongside the obtained value to inform users of the potential sources of error. This is good practice that promotes transparency and should be carried out for any method of calculating reference values. In addition, alternative numerical calculation methods can be used alongside the Chebfun methods, and the number of significant figures that agree between both methods can be used as the reference value, increasing the confidence in the presented value.

With these reference parameter values obtained, it is now possible to use these to assess the performance of third-party metrology software packages. The process was performed in a similar method to section 5.3.3; the five complex surfaces were sampled to create high resolution 700×700 .SDF datasets which were then input into each third-party software package to calculate parameters. The results were then normalised to the mathematically-obtained values to enable easier comparisons.

Figure 5.9 shows the results for the *Ssk* parameter obtained by each of the software packages under test for all five complex surfaces. An interesting result to note here is the apparent similarities and differences between the underlying calculation methods employed by each of the software packages. Software A and C obtain similar results for each surface;

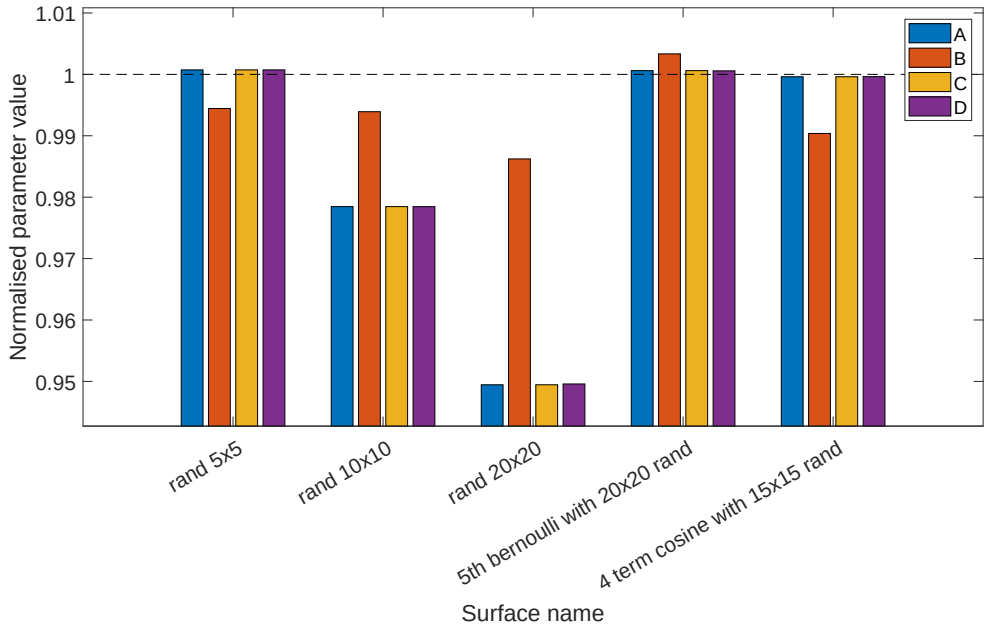


Fig. 5.9 Software obtained values for the *Ssk* parameter, normalised to the mathematically-obtained value, using Chebfun, for a series of analytically-defined surfaces.

even those for which they deviate from the reference value by as much as 3%. This suggests software A and C both calculate *Ssk* using a similar approach. Software B, however, clearly uses a different calculation method, allowing it to remain more consistently close to the reference value, but does not perform as well as the best cases of the other software packages.

Figure 5.10 shows the results for the *Sa* parameter, newly calculable using the Chebfun method. All software packages under test show good agreement with the reference value, deviating by no more than one tenth of a percent. Here, software B shows the greatest amount of variation, suggesting the methods employed to calculate the *Sa* parameter are not as accurate as those employed by the other three software packages. Again, there is very close agreement between software A and C.

Figure 5.11 shows the results for the *Sdr* parameter. Here, there are significant differences in the values obtained by software A and C, suggesting *Sdr* is one parameter that the two software packages calculate differently. Similar to the results for the *Sdq* parameter shown in

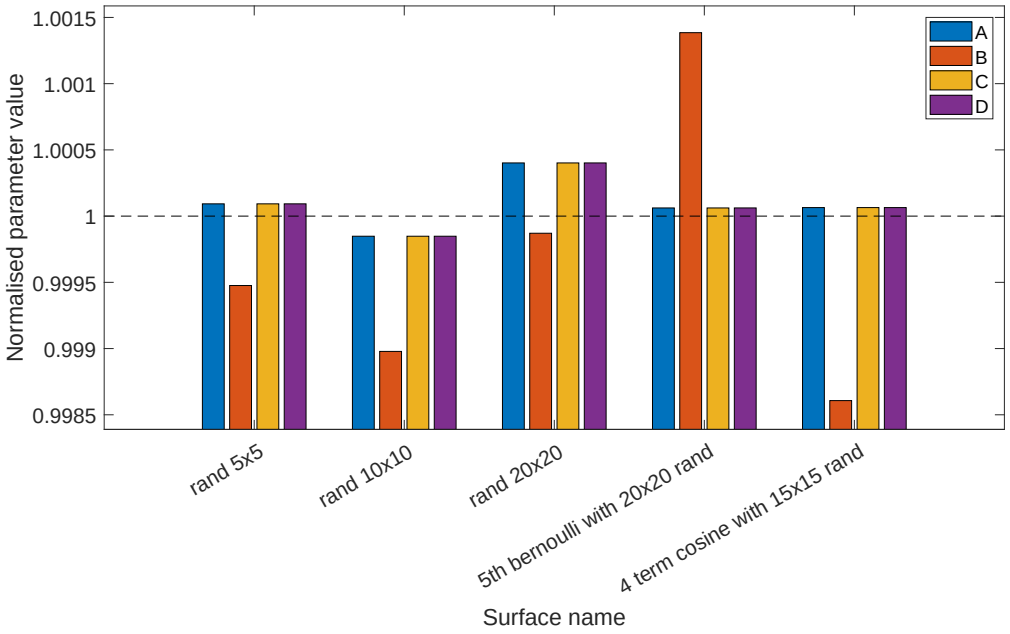


Fig. 5.10 Software obtained values for the *Sa* parameter, normalised to the mathematically-obtained value, using Chebfun, for a series of analytically-defined surfaces.

figure 5.4, the *Sdr* parameter is affected by the resolution of the dataset, and the results here show that surfaces with more high spatial frequency components are the more significantly affected, with each of the software packages performing worse for those cases. It should be noted here that software D only gave *Sdr* results to two decimal places, making meaningful comparisons with the other software package results difficult.

Through the use of the numerical methods made available using Chebfun, high precision reference parameter values can be obtained not only for parameters whose calculations are prohibitive for symbolic calculations using CASs, but also for surfaces with an increased degree of complexity that would otherwise be too computationally intensive. Using these new methods, a greater understanding of the strengths and weaknesses of the software under test can be performed using a wider range of surface texture parameters. In addition, the use of high complexity surfaces with high spatial frequency components delivers more information about the software under test, and how the calculated parameter values can vary.

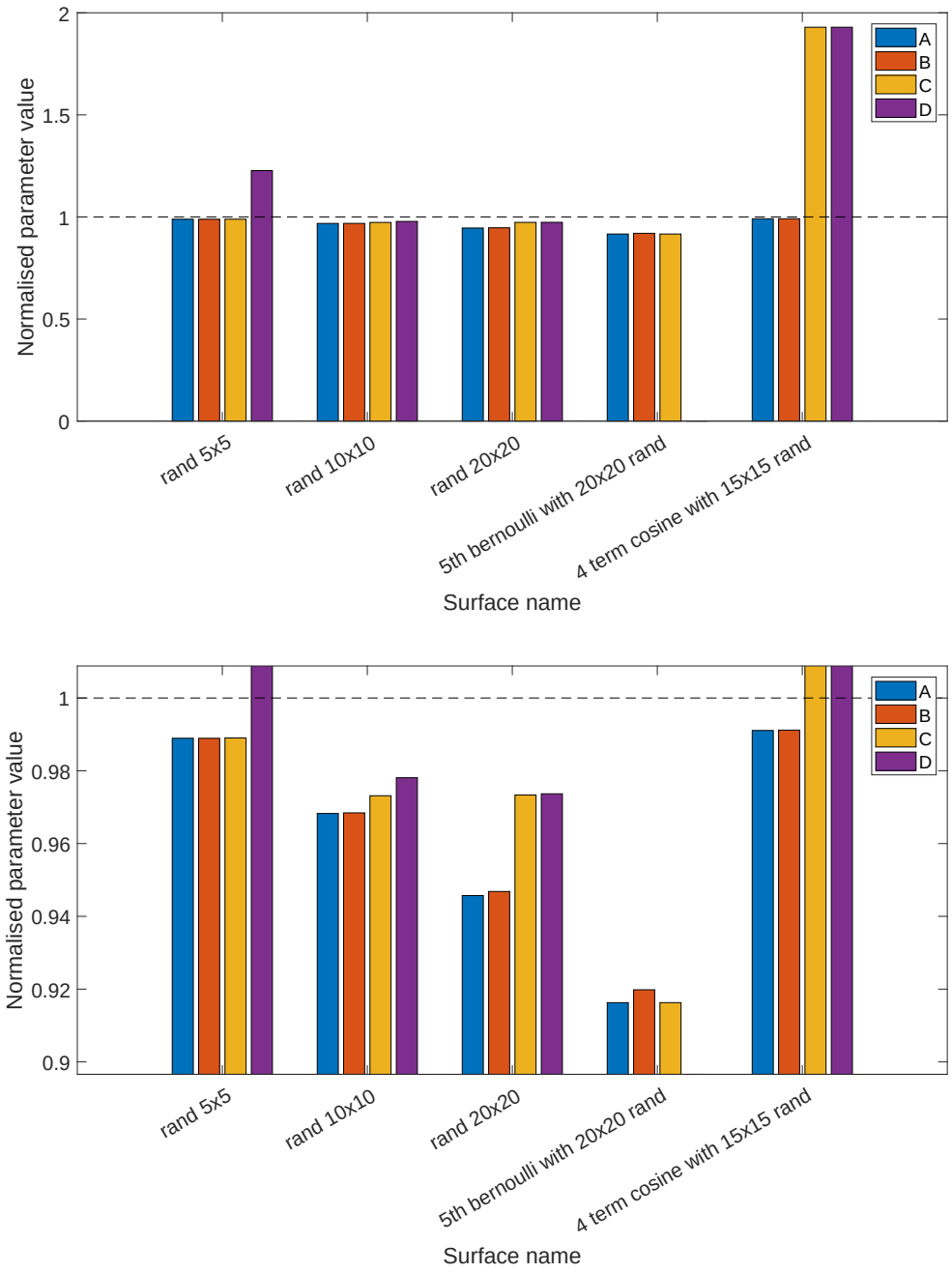


Fig. 5.11 Software obtained values for the *Sdr* parameter, normalised to the mathematically-obtained value, using Chebfun, for a series of analytically-defined surfaces. *Top*: Full zoom showing all results. *Bottom*: Zoomed in to highlight differences between the software under test.

5.5 Parametric surfaces

In section 5.3, methods for calculating surface texture parameters were presented that are applicable to any general surface. That is, for any given analytical surface function describing surface height given x, y coordinates, those general symbolic methods could be performed to obtain parameter values. For computationally prohibitive cases, section 5.4 introduced the use of numerical methods to extend the application to more complex computations, still using the general case.

Alternatively, it is possible to define a limited set of analytical reference surfaces which have been designed by using a specific surface texture parameter as a starting point. Using this approach, reference pairs [90] of surfaces and parameter values can be achieved symbolically for parameters that are incalculable for the general case. In addition, these surfaces can be defined parametrically, in terms of variables instead of fixed numbers, allowing for any number of surfaces to be created with known parameter values, given they are defined in accordance with specific rules. As a downside to the general case, not all parameters will be calculated for each surface.

In this work, parametric surfaces have been created as an alternative to using numerical methods, such as Chebfun, that allow for the complete symbolic calculation of the full range of field surface texture parameters. As with previous methods, the details of a selection of parametrically defined surfaces, along with their parametrically-defined parameter values, are given in appendix E. Further details of how to obtain symbolic values for surfaces that were not possible in the general case are given in the following sections.

5.5.1 *Sal* and *Str*

As mentioned previously, *Sal* and *Str* are based on the autocorrelation function, and are calculated using the longest and shortest distances from the origin to a specified autocorrelation value. For the general case, difficulty can occur when applying the autocorrelation

definition to the surface function, leading to an autocorrelation function of unknown shape which must be analysed to find the required decay lengths. This can be avoided by using a two-dimensional Gaussian function, whose autocorrelation function is also a Gaussian function. The Gaussian function about the origin is given by

$$f_{ACF}(t_x, t_y) = e^{-\left(at_x^2 + bt_x t_y + ct_y^2\right)} \tag{5.25}$$

where

$$a = \frac{\cos^2(\theta)}{\sigma_x^2} + \frac{\sin^2(\theta)}{\sigma_y^2}, \tag{5.26}$$

$$b = \frac{-\sin(2\theta)}{\sigma_x^2} + \frac{\sin(2\theta)}{\sigma_y^2}, \tag{5.27}$$

$$c = \frac{\sin^2(\theta)}{\sigma_x^2} + \frac{\cos^2(\theta)}{\sigma_y^2}, \tag{5.28}$$

where σ_x and σ_y are the standard deviations, or widths, of the two-dimensional bell, and θ is the rotation of the Gaussian function about its centre. Using the default specified height of 0.2, the equation for the contour of interest becomes

$$e^{-\left(at_x^2 + bt_x t_y + ct_y^2\right)} = \frac{1}{5}, \tag{5.29}$$

so

$$-\left(at_x^2 + bt_x t_y + ct_y^2\right) = \ln\frac{1}{5} \tag{5.30}$$

and

$$\left(at_x^2 + bt_x t_y + ct_y^2\right) = \ln 5. \tag{5.31}$$

Dividing each side by $\ln 5$ gives

$$\left(\frac{at_x^2}{\ln 5} + \frac{bt_x t_y}{\ln 5} + \frac{ct_y^2}{\ln 5}\right) = 1, \tag{5.32}$$

which is the form of an equation for an ellipse, where the semi-major and semi-minor axes are given by $\sigma_x\sqrt{\ln 5}$ and $\sigma_y\sqrt{\ln 5}$ respectively. Therefore, the values of *Sal* and *Str* can be calculated directly using the semi-major and semi-minor axes from a surface defined in terms of σ_x, σ_y and θ.

5.5.2 *Sa*

Section 5.3.1.5 explains how the definition of *Sa* leads to a discontinuous surface function, causing problems when trying to evaluate a surface in the general case. Discontinuities occur after the application of the absolute function at the points where the surface crosses the zero-plane with a non-zero gradient. The absolute function causes the sign of the gradient to switch, leading to an abrupt change in surface slope. This can be avoided by defining a surface in such a way that the gradient at the zero-line is always zero.

One example of this is a modified version of a simple cosine wave, such that the sign of the wave is inverted for every second period. This example is given by an equation of the form

$$z(x,y) = \begin{cases} A\left(1-\cos\left(f_1\pi x\right)\right), & 0 \leq x \bmod \frac{4}{f_1} < \frac{2}{f_1}, \\ -A\left(1-\cos\left(f_1\pi x\right)\right), & \frac{2}{f_1} \leq x \bmod \frac{4}{f_1} < \frac{4}{f_1}, \end{cases} \tag{5.33}$$

where A is the amplitude of the cosine wave and f_1 is the frequency. A profile example is given in figure 5.12. When the absolute function is applied to this surface, the parts of the surface below the zero-plane, which are the inverted sections of the surface, return above the zero-plane and recreate the unmodified cosine wave, for which *Sa* can easily be found.

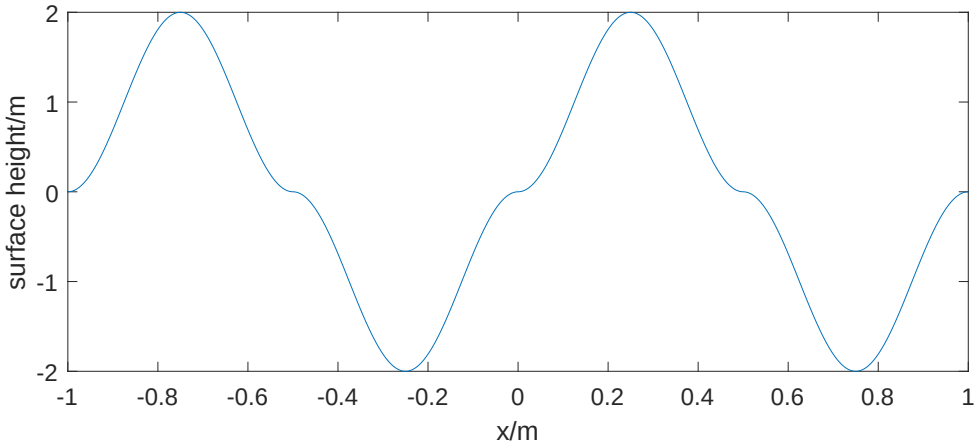


Fig. 5.12 Modified cosine wave to allow for a continuous surface after the application of the absolute function.

5.5.3 *Std*

The *Std* parameter is defined as the texture direction of the scale-limited surface and gives the angle (with respect to a specified direction) of the most prominent texture on a surface. Mathematically, this is given as the absolute maximum value of the angular power spectrum of the surface for a given direction, s, given by

$$f_{APS}(s) = \int_{R_2}^{R_1} r|F\left[r\sin\left(s-\theta\right), r\cos\left(s-\theta\right)\right]|^2 \mathrm{d}r, \tag{5.34}$$

where R_1 and R_2 denote the range of integration in the radial direction, and also requires the Fourier transformation

$$F(u,v) = \iint_{A_R} z(x,y)e^{-(iux+ivy)}\mathrm{d}x\mathrm{d}y. \tag{5.35}$$

where u and v are spatial frequencies in the x and y directions, respectively. This calculation, involving two transformations and an absolute function (which has already been explained to be symbolically challenging for the case of *Sa*, see section 5.3.1.5), poses a complex symbolic challenge for the general case. Instead, it is possible to create reference analytical

expressions with a known *Std* value that operate based on the parameter's more practical definition: the angle of the most prominent directional texture on a surface.

Creating a surface with a known, specific direction of texture can be achieved with a sinusoid with a known direction of propagation. By defining a surface of the form

$$z(x,y) = \cos\left([A\pi x] + [B\pi y]\right), \tag{5.36}$$

the angle perpendicular to the direction of wave propagation, which gives the texture direction, is given by

$$S_{td} = \tan^{-1}(B/A) + 90^\circ \tag{5.37}$$

in degrees.

5.5.4 *Sdr*

As mentioned previously, *Sdr* is calculated by finding the ratio of the evaluation area of the surface to the topographical area of the surface and subtracting 1. For the general case, the calculation of topographical area takes the form of a nonelementary integral. However, *Sdr* is readily calculable for more specific surfaces, a simple example of which is a linear slope, defined by

$$z(x,y) = Ax + By. \tag{5.38}$$

A schematic of this is shown in figure 5.13. The topographical area of a linear gradient surface can be found using the Pythagorean theorem to find the lengths of the two perpendicular sides of the rectangle of the form

$$\sqrt{x^2 + A^2x^2} = x\sqrt{1 + A^2}, \tag{5.39}$$

$$\sqrt{y^2 + B^2y^2} = y\sqrt{1 + B^2}, \tag{5.40}$$

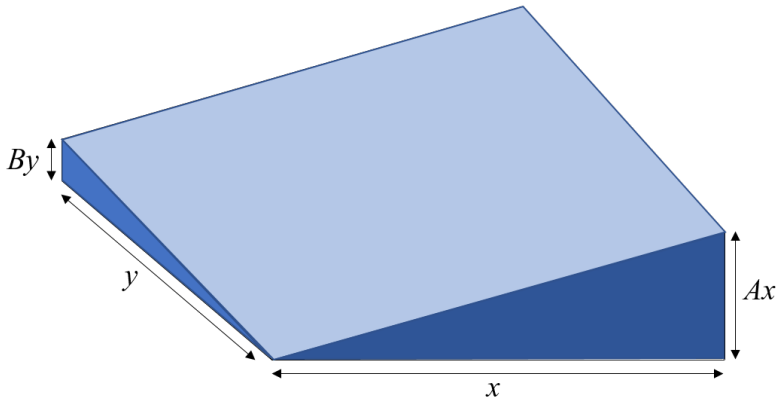


Fig. 5.13 Schematic of the linear gradient surface given in equation 5.38 for the calculation of topographical area.

and multiplying them together. The ratio of the topographical area to the evaluation area is then easily obtained, giving an expression for *Sdr* of

$$S_{dr} = \frac{xy\sqrt{1+A^2}\sqrt{1+B^2}}{xy} - 1, \tag{5.41}$$

or

$$S_{dr} = \sqrt{1+A^2}\sqrt{1+B^2} - 1. \tag{5.42}$$

5.5.5 Assessment of third-party software

Similar to the assessments performed in sections 5.3.3 and 5.4.4, four surface texture parameter calculation software packages were used to showcase the validity of the parametric reference surfaces as a means to assess the performance of software. A total of five surfaces were created for each parameter, substituting different variable values into the parametric surface definitions. This approach demonstrates the flexibility of parametrically-defined surface/parameter reference pairs, as it enables a theoretically infinite variety of surfaces to be created that have mathematically traceable associated parameter values by adjusting the variables in the surface definition.

Figure 5.14 shows the results obtained by the software packages of the *Sdr* parameter for five surfaces obtained using a parametric surface function. Software C and D both show good agreement with the mathematical reference value for all surfaces, with software C obtaining normalised parameter values on the order of 1.00000005 for each surface. Software D shows more deviation than the other software packages for the fourth and fifth surface, which correspond to the surfaces with the lowest gradients. Conversely, software A and B performed poorly for the three surfaces that have the highest gradients and improve for the low gradient surfaces. In addition, software A and B obtain similar results for all five surfaces, suggesting a similar implementation of the *Sdr* calculation is performed for both software packages.

Figure 5.15 shows the results obtained for the *Std* parameter, for five surfaces obtained using a parametric surface function. It should be noted here that only software packages A and C offered the ability to calculate the *Std* parameter. Initial results, given in the upper graph, show a significant difference between the performance of software A and software C in their ability to obtain parameter values comparable to the mathematical reference value. Further investigation revealed the values obtained by software C were all less than those obtained by software A by around 90°, suggesting a difference in interpretation of the definition of the *Std* parameter given in ISO 25178-2 [1], specifically, relative to what axis should the most prominent angle of texture should be measured. Manually rotating the values obtained by software C gives the results shown in the lower graph, which show significantly better agreement with the mathematical reference values. With the corrections applied, software A is able to obtain a value for *Std* closer to the mathematical reference than software C for four of the five surfaces tested.

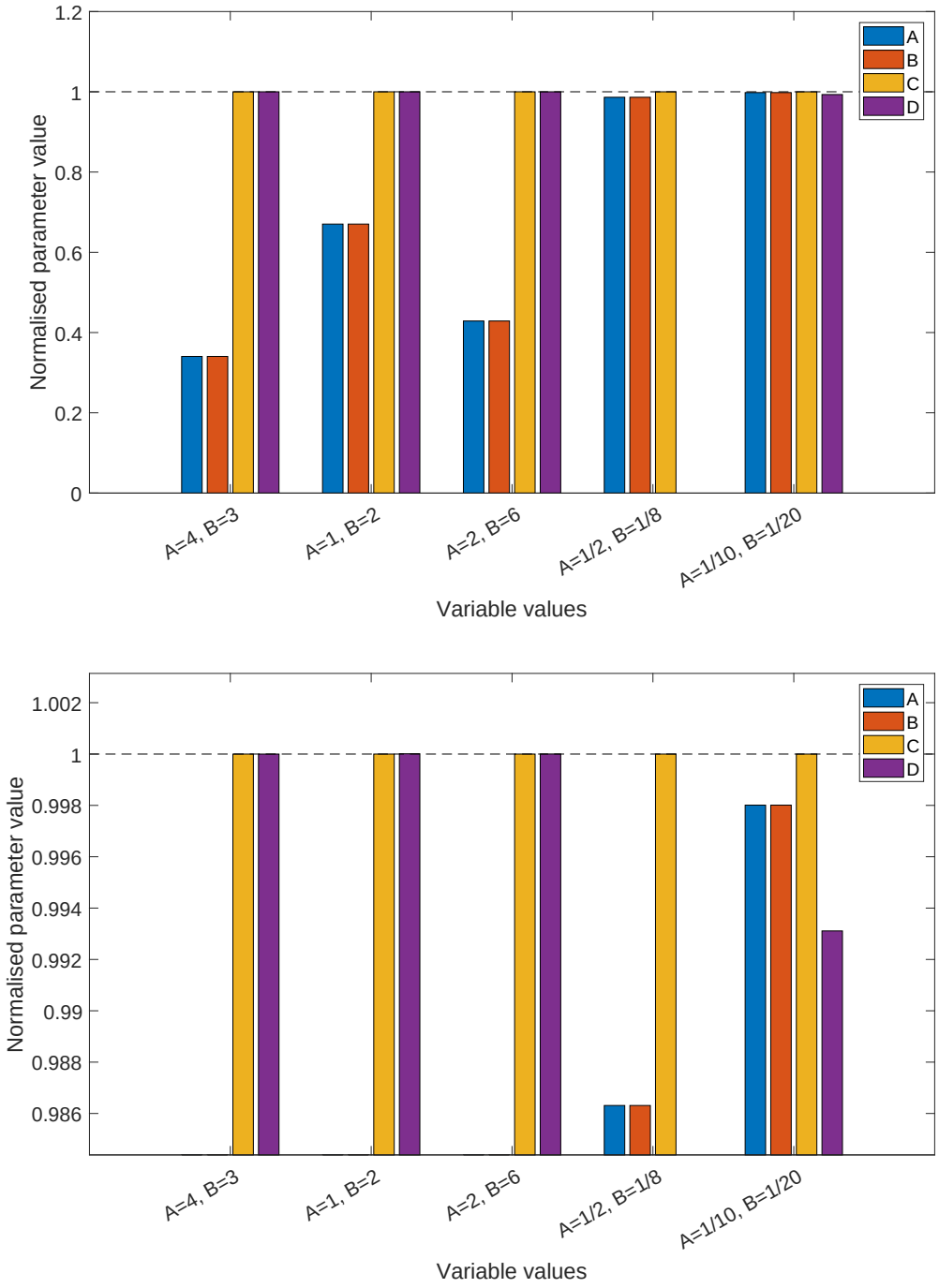


Fig. 5.14 Software obtained values for the *Sdr* parameter, normalised to the mathematically-obtained value, for a series of parametrically-defined surfaces. *Top*: Full zoom showing all results. *Bottom*: Zoomed in to highlight differences between the software under test.

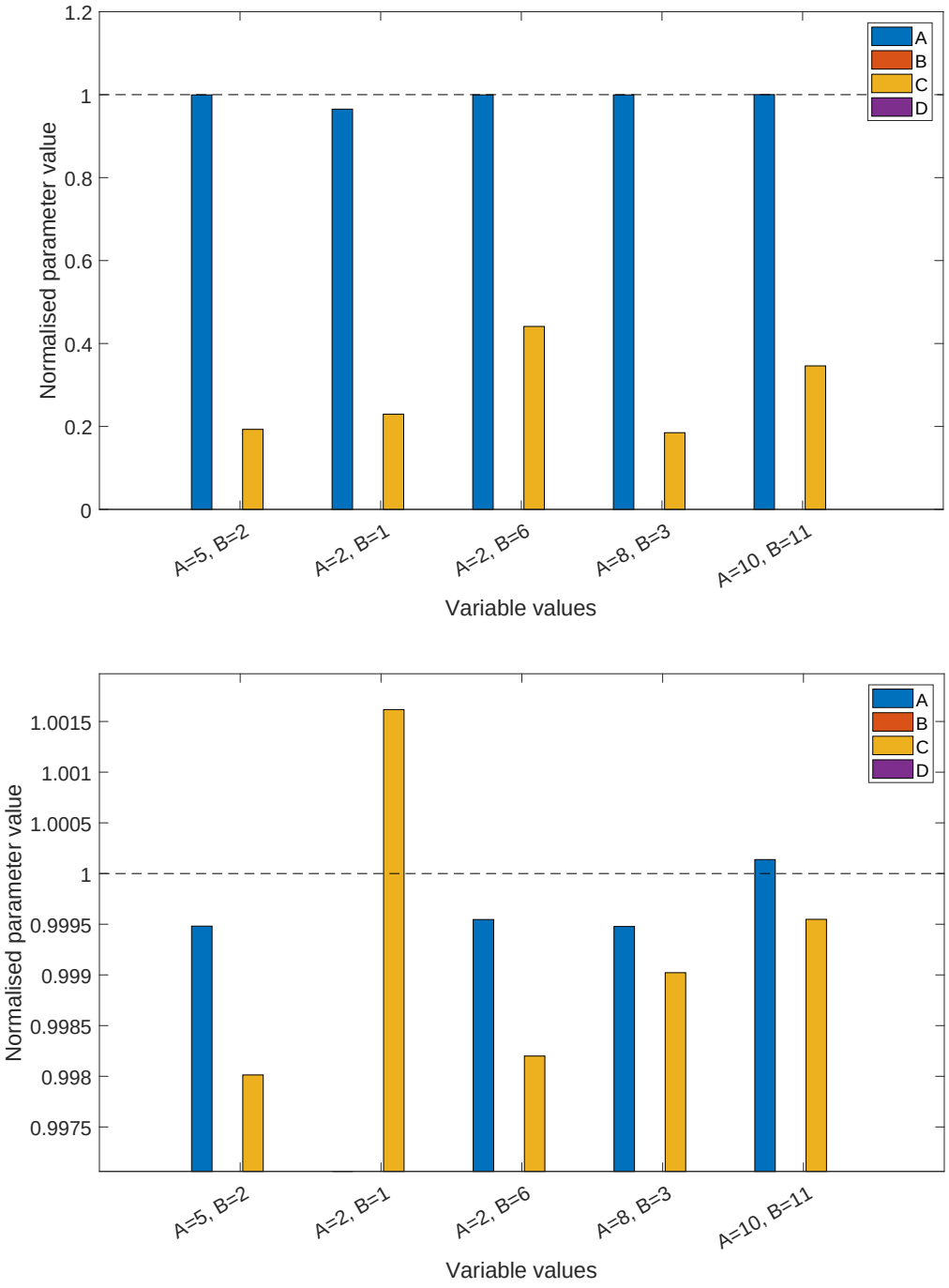


Fig. 5.15 Software obtained values for the *Std* parameter, normalised to the mathematically-obtained value, for a series of parametrically-defined surfaces. *Top*: Original results. *Bottom*: Software C results rotated by 90°.

5.6 Conclusion

The work presented in this chapter applied the mathematical reference concept that was introduced in chapter 4 to field surface texture parameters. Chapter 4 focussed on functional surface texture parameters, which are calculated from the material ratio curve, a relatively simple one-dimensional function. For field surface texture parameters, calculations are based, often directly, on the surface itself, which can be a complex two-dimensional function.

For several height based field parameters, namely *Sq*, *Ssk*, *Sku*, and *Sdq*, direct substitution of the surface expression into the parameter definition is readily calculable for the general case, which enables the parameter values to be found for theoretically any surface expression (providing an adequate amount of computing time and power). For others, such as *Sp* and *Sv*, additional operations are required. Mathematical parameter values were calculated for ten analytical surfaces and were used as a comparison tool for four third-party metrology software packages. For the field surface texture parameters that remain, symbolic calculation for a general surface expression is infeasible. In addition, highly complex surfaces can pose a problem for modern symbolic calculation tools, such as CASs, due to the high computational power required to calculate parameters for them.

One method that was presented for obtaining reference parameter values for a wider range of parameter values was to utilise numerical methods to perform 'approximate' calculations of the parameters. By calculating accurately to a number of significant figures higher than that obtained by the software under test, it is still possible to perform meaningful assessment of the software. The open source Chebfun software was used and was demonstrated to agree with symbolically obtained parameter values to fourteen significant figures. Following this, a selection of five high-complexity analytical surfaces were created, and Chebfun was used to obtain parameter values. Similar to the symbolic case, these values were used as references against which third-party software was compared.

An alternative method to extend the selection of calculable parameters was introduced that focussed on tailoring specific surfaces for certain parameters. These surfaces were simplistic in design but allowed for the calculation of the full range of parameters, and can be used as simple reference surfaces to assess the basic calculation of each parameter by a software package under test. Additionally, these specific surfaces, and the associated parameter values, were defined parametrically in terms of user-defined variables, allowing for a wider variety of surfaces to be created that can better test the software.

In conclusion, the combination of the methods developed in this chapter delivers a comprehensive toolkit for the calculation of high-accuracy reference parameter values for analytical surface expressions. These reference pairs can be used to assess surface texture parameter calculation software and give both developers and end-users better insight into the performance of the software.

Chapter 6

Performance assessment of surface texture parameter software

6.1 Introduction

Chapters 4 and 5 presented mathematically-defined reference pairs [90] for the performance assessment of surface texture parameter software. An important aspect of this process, however, is the delivery of the analytical surface to the discrete software under test.

All surface texture parameter software operates with a discrete surface height dataset as the primary input. As the reference surfaces are defined analytically, it is therefore necessary to perform a discretisation operation in order to create a discrete dataset that can be used by the software. This is achieved by some method of sampling; extracting a finite number of height values by evaluating the surface expression at different points in x and y. By virtue of this finite sampling, however, information about the surface is inevitably lost, as the areas in between the sampling locations contain surface information that is not transferred to the discrete representation.

Due to the loss of information when moving from an analytical representation to a discrete representation, surface texture parameter calculation software will not be able to

obtain the exact same results as the mathematical reference. The software is working with less information, and so will deviate from the mathematical reference values in scenarios where the information required has not been sampled and incorporated into the dataset. It is important, therefore, to attempt to account for this discretisation error and assess the software under test fairly, based on the information it is given.

This chapter introduces different performance metrics that can be used to numerically assess the performance of surface texture parameter calculation software under test against the mathematical reference values introduced in chapter 4 and 5, while taking into account the discretisation error introduced from using discrete datasets for testing.

6.2 Parameter value extrapolation

The amount of information lost to discretisation, and thus the amount of discretisation error introduced, is related to the resolution of the dataset, where, here, resolution is the number of height values included in the dataset. Higher resolution means a larger number of sampling points, which means a greater amount of information is transferred from the analytical surface to the dataset. Taking this to the extreme, a theoretical 'infinite resolution' dataset would be created from an infinite number of discrete sampling points of the analytical surface, and the discretisation error would reduce to zero, giving an accurate discrete representation of the surface, as shown in figure 6.1. An infinite resolution dataset is purely theoretical and cannot be created in practice. However, by obtaining information about the relationship between surface resolution and discretisation error, the infinite case can be approximated to a desired tolerance.

By approximating the infinite resolution dataset, or rather the surface texture parameter values obtained by software from an infinite resolution dataset, comparisons can be made against the mathematical reference values that are free from discretisation error, allowing a greater focus on deviations that are due to implementations of the software. This can be

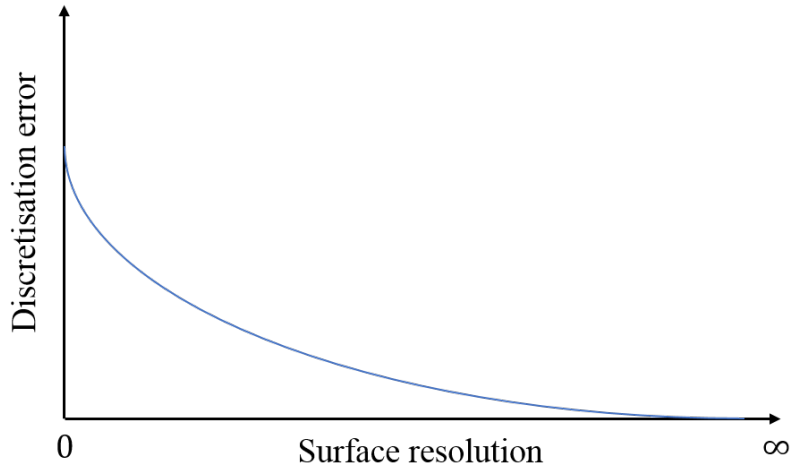


Fig. 6.1 Exponential relationship between the resolution of a discrete surface dataset and the discretisation error introduced as part of the sampling process. Different discretisation methods may result in relationships that are not exponential.

achieved by creating a series of datasets of increasing resolution. These datasets can each be processed by the software under test, obtaining parameter values that get increasingly close to the scenario without discretisation error. With enough values, a relationship can be found, and the results can be extrapolated to find the parameter value against which the relationship asymptotes. This asymptote denotes the parameter value for the infinite resolution dataset, and hence the parameter value that would be obtained by the software if it were free from discretisation error. This technique is shown in figure 6.2.

The parameter-resolution relationship is obtained by performing a curve fitting operation. This operation obtains a curve equation describing the relationship which can then be evaluated to find the asymptote. As the curve will be extrapolated significantly beyond the range of dataset resolutions used to fit the curve, the form of the curve and the quality of the fit will be important.

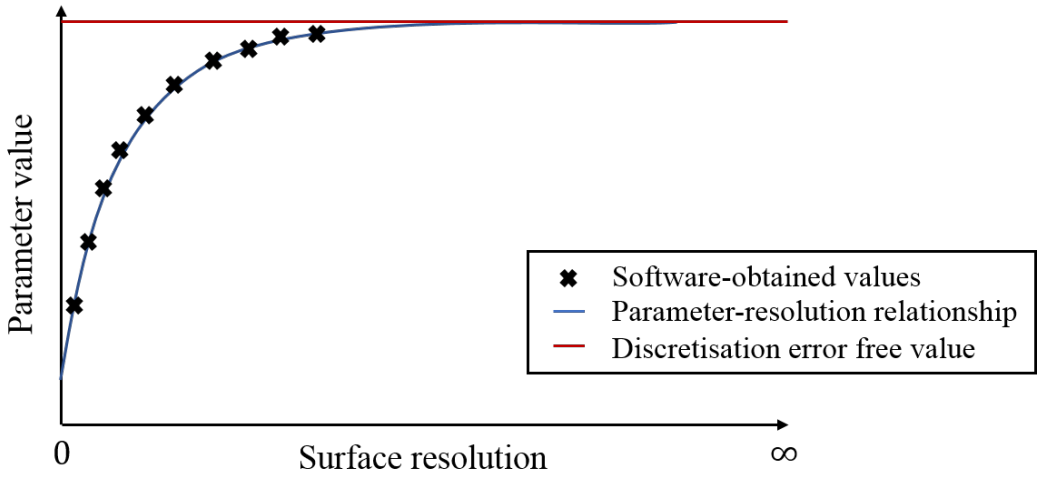


Fig. 6.2 Relationship between the resolution of a discrete surface dataset and the parameter values obtained by software under test. The software-obtained values are used to obtain a curve equation that describes the parameter-resolution relationship. The relationship is extrapolated to obtain the software-obtained parameter value free from discretisation error.

6.2.1 Implementation

The parameter value extrapolation technique is implemented in MATLAB utilising the built-in *fit* function, and the quality of fit is assessed using the coefficient of determination, or R-squared, that measures the amount of variance in the data from the fitted curve. The script runs through an iterative process outlined in the following:

1. An array of software-obtained parameter values are input with information of the resolutions of the datasets used.

2. These parameter values are manipulated into four array formats, each tested to find the best fit:

   (a) Raw parameter values;

   (b) $\log_{10}$ of the raw parameter values;

   (c) Differences between adjacent parameter values, $val_{j+1} - val_j$;

   (d) $\log_{10}$ of the differences between parameter values.

Here, the array format for taking the differences between adjacent parameter values is used to bring the asymptote of the fitted curve equation to zero (in the ideal case), changing the equation that needs to be found by the MATLAB fitting algorithm and giving an additional opportunity to find a higher quality fit. The inclusion of $\log_{10}$ formatted data is to produce a linear relationship instead of an exponential relationship, again giving the potential for an easier task for the MATLAB fitting algorithm.

3. Each array format is then fitted with three different equations:

   (a) A power function of the form

   $$ax^b + c \tag{6.1}$$

   (b) An exponential function of the form

   $$ae^{bx} + ce^{dx} \tag{6.2}$$

   (c) A linear polynomial function of the form

   $$ax + b \tag{6.3}$$

   where x denotes the resolution of the surface. The fit with the highest R-squared value is stored.

4. The lowest resolution parameter value (or equivalent in other array formats) is removed from the array, in order to account for any outlying points that skew the fit due to their low resolution, and the fit is repeated. If the R-squared value is improved, the current best-fit is updated.

5. The best-fit is extrapolated to a very high resolution by evaluating the curve equation at much larger resolution values.

This iterative algorithm obtains the best quality fit by testing a combination of array formats, fit types and number of data points, and selecting the combination that delivers the highest R-squared value. The numerical precision of the parameter values output by the software under test will have an effect on the results obtained. The MATLAB fitting software operates at machine precision, and so entered values that are below this level will have zeros appended to meet the machine precision required, potentially affecting the obtained results.

6.2.2 Application of the method

Figure 6.3 shows the results of the parameter extrapolation method applied to the *Sa* parameter values for a simple cosine analytical surface, obtained by a third-party software package (software A from chapter 5). The software was given 50 datasets, with resolutions ranging from 50×50 to 2500×2500. For simplicity, the surface datasets were square to ensure an equal resolution in both x and y directions. This method is easily adapted for uneven resolution datasets by relating the parameter value to the total number of height values within the datasets. After running through the parameter extrapolation algorithm, a fit was found with an R-squared value of 0.999999990661360, using an exponential equation with an array format of the logarithm of the differences between adjacent parameter values, as addressed previously. The marked blue line in figure 6.3 shows the original parameter values obtained by software A. The data points show a smooth trend, allowing the fitting algorithm to find a close agreement, as given by the R-squared value. The first few data points, corresponding to very low dataset resolutions, do not fit the trend displayed by the rest of the data, and were not included in the fitting algorithm. The obtained curve equation has been evaluated to a resolution of 7500×7500 and is presented as the dashed blue line. The curve equation has not been evaluated to very high-resolution values here as the nature of the asymptotic curve delivers diminishing returns, and further extrapolation would yield minimal changes to the final value. In addition to the 50×50 to 2500×2500 resolution datasets, an additional high

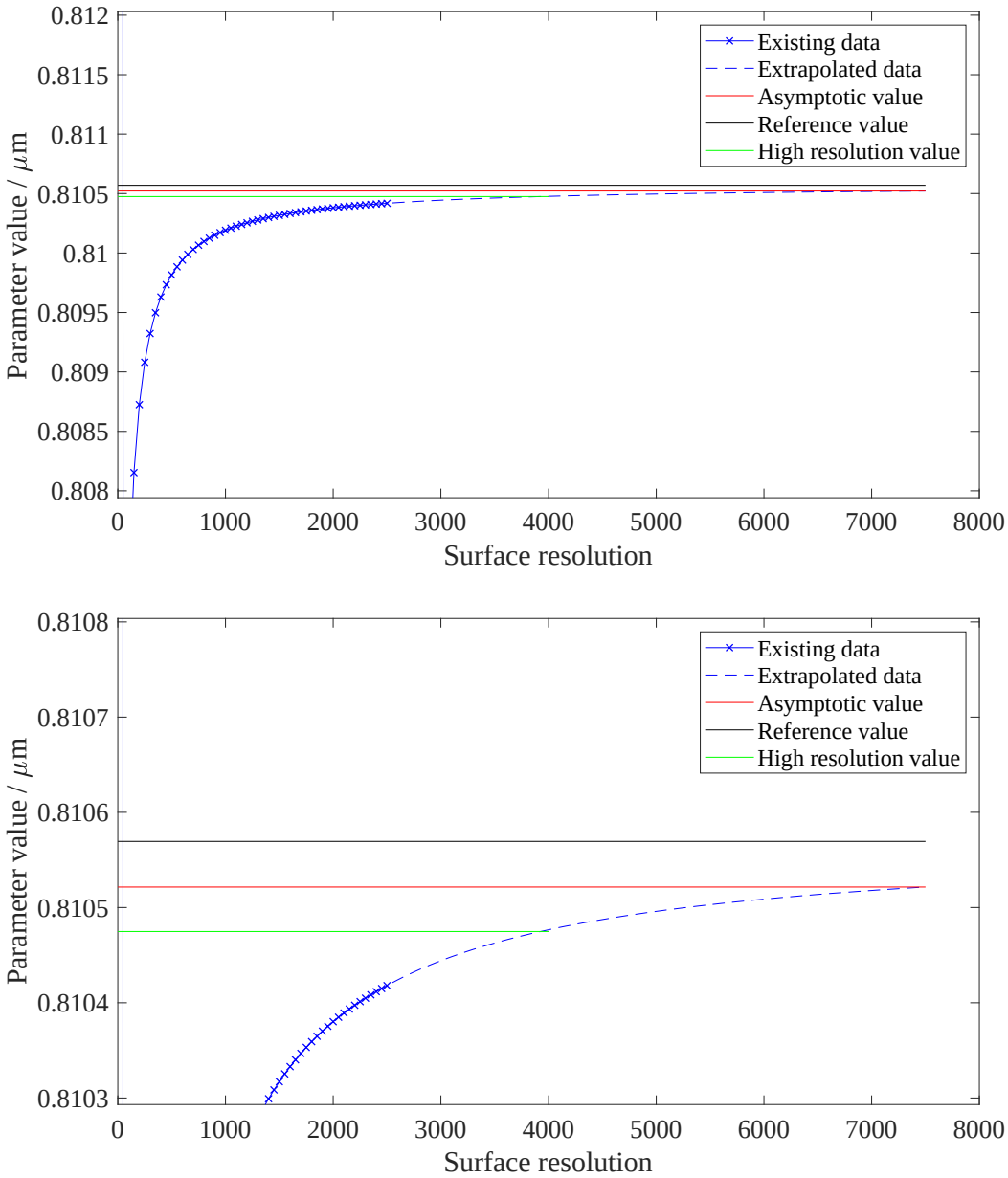


Fig. 6.3 *Sa* parameter value extrapolation for software A. *Top*: Full view. *Bottom*: Zoomed view.

resolution 4000×4000 resolution dataset was also input into software A, but not used in the fitting algorithm. This extra data point is used as a checking tool to assess the quality of the extrapolation. Figure 6.3 shows close agreement between the high-resolution data point and the extrapolated curve.

The reference parameter value obtained mathematically is also included in figure 6.3 as a benchmark, to show the deviation of the software-obtained value. The asymptotic value can be compared directly with the reference value to obtain a value for the software error, which in this case is

$$\text{Error}_{7500} = -4.790306426749602 \times 10^{-5} \mu\text{m}. \tag{6.4}$$

As this is not a true asymptote, and is based on an approximate fitting algorithm, it is not accurate to say that discretisation error has been completely eliminated. However, discretisation error has been substantially reduced, and this approach can serve as a valuable method of software performance assessment, providing the details of extrapolation are provided alongside the assessment. Dividing the magnitude of this error by the reference value gives a performance metric for the software useful for comparisons with other software and performance specification, of the form

$$\text{Error}_{rel,7500} = 5.9098 \times 10^{-3}\% \tag{6.5}$$

to five significant figures.

For comparison, a fit of similar quality with an R-squared value of 0.999999914523633 was found using an exponential equation with an array format of the raw parameter values that gave a value for the absolute error of

$$\text{Error}_{7500} = 1.667711284858431 \times 10^{-5} \mu\text{m}. \tag{6.6}$$

This result appears better than the previous result, however inspection of the resulting equation shows a divergence to infinity rather than any asymptotic value, and so is not a reliable fit. Evaluating this equation at a higher resolution, for example, gives an error value of

$$\text{Error}_{15000} = 2.350367534937892 \times 10^{-4} \mu\text{m}, \tag{6.7}$$

which is significantly worse. The original fit, however, gives an improved value of

$$\text{Error}_{15000} = -2.271270655940238 \times 10^{-5} \mu\text{m}. \tag{6.8}$$

These results show the sensitivity of the result to the choice of curve and the resolution to which the data has been extrapolated, and so care must be taken to ensure a reliable fit has been chosen and the extrapolated resolution is clearly defined.

In addition, it is possible to evaluate the equations obtained via the curve fitting process by extrapolating to infinite resolution. This allows the asymptotic value for the fit equation to be found and approximates the parameter value obtained by the software under test, free from discretisation error. For the fit described above, this gives an absolute error of

$$\text{Error}_{\infty} = -1.7516966861254 \times 10^{-6} \mu\text{m}, \tag{6.9}$$

and a relative error of

$$\text{Error}_{rel,\infty} = 2.1611 \times 10^{-6}\% \tag{6.10}$$

to five significant figures. It should be noted here that this approach is very sensitive to the fit equation obtained by the algorithm, and small deviations in the curve equation when evaluated for relatively low surface resolutions can lead to large deviations when evaluated at very high values.

6.3 Agreement of significant figures

The previous section introduced a method of parameter value extrapolation as a means to account for discretisation error when comparing analytical reference surface solutions to discrete datasets. This approach successfully reduced the amount of discretisation error present in the software-obtained parameter value, but unfortunately, has drawbacks. Firstly, the extrapolation method requires a large number of datasets to be input into the software, each at a different resolution, to allow the parameter-resolution relationship to be found. This is time consuming and would need to be done for each performance assessment. Second, the extrapolation technique is highly dependent on the quality of curve fitting. Even a small deviation in the fit of the curve to the data points can lead to a large deviation from the real value when extrapolated to very high resolutions. This adds inevitable inaccuracies to the method which must be factored into the performance assessment.

In this section, an alternative, simpler method for the assessment of surface texture parameter software performance is introduced which incorporates the presence of discretisation error into the assessment instead of attempting to remove it. This alternative approach quantifies the number of significant figures given by the software-obtained value that agree with the mathematical reference value. This number is then given alongside the resolution of the dataset used to obtain the software value, giving a performance metric describing the quality of the software's performance relative to the amount of discretisation error present. In addition, this performance metric allows for direct comparisons between third-party software, as the number of agreed significant figures for a given resolution dataset are directly comparable, whereas different parameter values obtained using the extrapolation method are dependent on the quality of the fit, and so cannot be directly compared when different fits are used. One limitation of this approach is in the case that the mathematical reference value is zero. In this case, when dealing with floating point arithmetic, the relative measure of agreed

Table 6.1 Number of significant figures from the software-obtained parameter value results that agree with reference values obtained using the methods in chapter 5 for 700×700 resolution datasets.

	4 term cosine				4th Bernoulli			
	A	B	C	D	A	B	C	D
Sq	5	3	5	5	10	2	10	2
Ssk	4	2	4	4	8	2	10	5
Sku	5	2	5	5	8	3	8	6
Sp	5	5	5	5	8	7	7	3
Sv	4	6	4	4	7	6	7	2
Sz	4	8	4	4	8	7	8	3
Sa	4	2	4	4	6	2	6	2
Sdq	6	5	6	4	3	3	3	0
Sdr	6	6	6	0	0	0	0	0
Total	**43**	**39**	**43**	**35**	**58**	**32**	**59**	**23**

significant figures may not be useful, as the exponent is much more valuable than other digits in the number. Instead the absolute difference is a more valuable value for comparison.

Table 6.1 showcases the assessment of significant figure agreement for two analytically defined surfaces introduced in chapter 5. Here, the level of numerical precision that the software-obtained values agree with the references is presented for each parameter, allowing direct comparisons between software in a meaningful way that is directly applicable to the use of the software under test: to understand the extent to which the software gives an accurate result in comparison to a traceable reference. In addition, the total number of agreed significant figures is also calculated for each parameter set, providing a simple performance metric for the software under test for the reference surface as a whole. The results of the totals show software C gives the overall best agreement with the reference values for a dataset resolution of 700×700, narrowly improving upon software A for the '4^{th} Bernoulli' analytical surface.

It should be noted that software D, as discussed in chapter 5, often gave parameter values to a reduced precision in comparison to the other test software, restricting its ability to perform well in this assessment.

6.3.1 Statement of software uncertainty

With the 'agreement of significant figures' method established as a reliable way to relate the performance of surface texture parameter calculation software, it is also possible to utilise the traceability to provide a statement of uncertainty for the software. The software can be assessed to obtain a number of significant figures to which its obtained parameter result agrees with numerical evaluations of the mathematical reference parameter values. The true value therefore lies somewhere within the upper and lower rounding bounds of that recorded digit, with a probability distribution that can be assumed to be uniform. For example, for the *Sq* result from software A for the 4 term cosine surface, a significant figure agreement of 5 is found, as seen in table 6.1. Normalised to 1 m, this corresponds to a software value of 1.0000 m, with the true value lying somewhere between $a_- = 0.999\,95$ m and $a_+ = 1.000\,05$ m, with uniform probability.

These bounds, coupled with the uniform probability distribution, give a calculation for the standard uncertainty in the result of

$$u(x) = \frac{(a_+ - a_-)}{2\sqrt{3}}, \tag{6.11}$$

where x is the software result [91]. Applying this to the normalised example gives a result with associated standard uncertainty of

$$S_q = (1.0000 \pm 2.89 \times 10^{-5})\,\text{m}. \tag{6.12}$$

Alternatively, if applied directly to the obtained software value,

$$S_q = (843.91 \pm 0.00289)\,\text{mm}. \tag{6.13}$$

6.4 Variation due to sampling

The previous sections have presented methods to assess the performance of surface texture parameter calculation software that address the discretisation error that is introduced when sampling an analytical surface to create a dataset. This is performed in order to ensure the assessment of software focusses on the in-built algorithms and implementation choices used in the software, and not aspects of the process that are out of the software's control.

Another stage of the discretisation process that can affect the dataset and, therefore, the resulting parameter values is the sampling method used to obtain discrete height values from the analytical surface. The continuous surface can have different height values at very close x,y locations on the surface, and so small variations in the sampling locations can obtain different height values for use in the dataset. Additionally, averaging methods can be used that obtain single height values that describe areas of the analytical surface instead of specific sampling locations, which could also produce a different dataset for the same evaluation area.

There is no 'correct' sampling method, as different choices can be used to represent the height capture methods of different measurement processes. For example, point autofocus and contact stylus measurement instruments sample at discrete lateral intervals, whereas areal techniques, such as focus variation, obtain a full area image wherein each pixel represents a small area of the surface. In any case, sampling method variations can be addressed by detailing the method used when performing the software assessment. By explicitly defining the sampling method used, any further tests can be made using the same sampling methods and inter-software comparisons can still be made.

To highlight this variation, an alternate sampling implementation was used to compare to that used in all previous sections. For the previous sections, each height value data point was treated as a pixel with a width equal to the spacing between x,y sampling points. Therefore, to span the appropriate area, one sampling point was removed to account for the extra half spaces either side of the first and last data points in each direction. For example, for a 1000×1000 resolution surface spanning 10 mm by 10 mm, height values were sampled between 0 mm and 9.99 mm in each direction, accounting for the pixel widths which extend the surface size to between -0.005 mm and 9.995 mm. The alternative sampling implementation uses the same spacing between points, but includes an extra point on the end, giving a 1001×1001 resolution dataset with height values sampled between 0 mm and 10 mm. This method treats the dataset as a collection of discretely sampled height values with zero pixel-width, and so still spans the same evaluation area as the previous implementation.

Table 6.2 gives the same significant figure analysis as performed in table 6.1, but for the alternate sampling method. Software B shows a significant improvement in its agreement with the reference values, while software A, C and D all show worse results. These performance metrics suggest this style of sampling is well interpreted by software B, whereas the other software packages are more suitable for the previous implementation. Again, neither implementation is 'correct', and clear specification of the sampling choices used when creating a dataset are necessary for proper assessment of metrology software.

6.5 Conclusion

The work described in this chapter presented methods to assess the performance of surface texture parameter software whilst accounting for discretisation error introduced when creating a discrete dataset from an analytical surface. Performance assessment tools are valuable in enabling software users to understand the quality of the software being used and to what numerical precision it can be trusted. In addition, performance assessment is valuable to

Table 6.2 Number of significant figures from the software-obtained parameter value results that agree with reference values obtained using the methods in chapter 5 for 701×701 resolution datasets calculated using an alternate sampling method.

	4 term cosine				4th Bernoulli			
	A	B	C	D	A	B	C	D
Sq	3	5	3	3	2	5	2	2
Ssk	2	3	2	2	2	4	2	2
Sku	2	4	2	2	3	6	3	3
Sp	3	3	3	3	6	6	6	3
Sv	5	4	5	5	5	4	5	2
Sz	4	8	4	4	7	7	7	3
Sa	3	4	3	3	2	5	2	2
Sdq	7	9	7	4	5	9	5	0
Sdr	5	10	5	0	0	4	0	0
Total	**34**	**50**	**34**	**26**	**32**	**50**	**32**	**17**

software developers as an established method to compare their software to the reference values and identify software strengths and weaknesses.

The technique of extrapolating the software-obtained parameter values to very high resolutions was a novel approach to minimising the discretisation error in the parameter value. This technique relied heavily on the quality of the fit performed on the data and required a large number of software-obtained parameter values, each from a different resolution dataset in order to approximate the parameter-resolution relationship. These challenges limit the accuracy of the extrapolation method; however, tests successfully showed an ability to reduce the discretisation error and approximate parameter values for higher resolution datasets than those calculated directly.

As an alternative, simpler approach to performance assessment, a technique was presented that calculated the number of significant figures in the software-obtained parameter values that agree with the reference value. This method gave a succinct performance metric for each parameter that enabled easy inter-software comparisons and allowed for quick assessment of a software's ability to agree with the reference value. The performance metrics for each

parameter were also combined to assess the performance of the software as a whole, taking each parameter into account.

In addition, the implementation of sampling on the created dataset, and hence the resulting parameter values, was addressed. The importance of explicitly defining the sampling methods used when producing a dataset were emphasised, and an example comparison of two different sampling implementations were compared, showing a significant effect on different software packages.

The combination of the performance metrics introduced in this chapter, and the reference surface texture parameter values introduced in chapters 4 and 5, produce a novel framework for the validation of surface texture parameter software that improves upon the current state of the art by introducing mathematical traceability and increased accuracy. By incorporating performance metrics, both developers and end-users of metrology software are able to assess and compare software results easily and accurately.

Chapter 7

Conclusion

7.1 Assessment of the current state of the art

Surface texture parameters are an important part of manufacturing. By giving quantitative numerical descriptors, the industry is able to characterise the texture of parts using internationally agreed upon method, facilitating inter-disciplinary comparisons and collaborations. In working to well-defined specification standards, this surface characterisation enables fine tolerances on part properties, reducing waste and improving part performance.

For the true value of surface texture parameters to be realised, it is crucial that all users are working to the same definitions. ISO 4287 [13] and 25178-2 [1] lay out the mathematical definitions for profile and areal surface texture parameters, respectively, however, implementation of these definitions is often performed using numerical software. Therefore, care must be taken to ensure the parameter calculation software is conforming to the definitions given in the standards to a level of accuracy appropriate for the required task. The current state of the art approaches this validation process using reference software: parameter calculation software, often developed by NMIs, with a focus on accuracy over speed in order to obtain the highest accuracy parameter values possible, for a given input surface.

An international survey was carried out to investigate the use of surface texture parameters in industry, as presented in appendix B, and assess the adoption of the areal surface texture parameters [1]. This work was a follow-on to a survey that took place twenty years ago [16], and sought to identify the evolution of surface texture parameter usage in industry. The results of the survey showed both a significant increase in the use of profile surface texture parameters, but also a marked uptake of areal surface texture parameters, with approximately 30% of participants using at least one areal parameter. The increase in usage of surface texture parameters in industry highlights their importance in the manufacturing field, and thus the necessity for high-accuracy validation methods for surface texture parameter calculation software.

Chapter 3 performed an in-depth comparison of profile surface texture parameter reference software developed by three independent NMIs. This work used a selection of NMI-developed reference surface datasets and investigated the differences in parameter values obtained. In addition to finding differences in the desired input dataset format between reference software (despite each supporting the internationally standardised .SMD file format as defined in ISO 5436-2 [14]), differences in parameter values were found between each of the three reference software packages, with possible reasons for each of these variations suggested. In spite of these differences, each of the three software packages are regarded as NMI-endorsed references and can be used to validate third-party surface texture parameter calculation software. This can lead to variations and disagreements between validated software, which can cause manufacturing errors and increased material waste.

7.2 Contribution to the field

The main goal of the work presented in this thesis was to develop a novel method for the validation of areal surface texture parameter calculation software that improves upon the current state of the art. By introducing mathematically traceable, analytically defined surface-

parameter reference pairs, third-party software could be compared against a high accuracy reference that can be evaluated by anyone. This novel method required the development of analytical, two-dimensional equations that represented heights on a surface, which could then be evaluated using a series of analytical algorithms can conform to the mathematical parameter definitions given in ISO 25178-2.

The first development of mathematical surface-parameter reference pairs was introduced in chapter 4 for functional surface texture parameters based on the material ratio curve. By using the simple material ratio curve as the analytical definition instead of a surface height function, all functional surface texture parameters can be obtained by evaluating the material ratio curve directly. Therefore, traceable, analytically defined reference pairs were created that can be used as infinite precision references against which third-party software can be compared. By linearly sampling the analytical definition for the material ratio curve, height values can be obtained that can be converted into a discrete dataset for input into third-party software for assessment. Comparisons between three third-party surface texture parameter calculation packages were performed, with the results presented relative to the mathematically-obtained reference values. These comparisons showed how each software deviated from the reference, identifying strengths and weaknesses that can be addressed by the developer.

Chapter 5 presented a method for the creation of mathematical reference surfaces and applied it to the calculation areal field surface texture parameters. A variety of mathematical operations were presented (in some ideal cases identical to the ISO 25178-2 definitions) that utilise CASs to obtain parameter values for general case surface representations. Similar to chapter 4, these produce mathematically traceable reference pairs for the assessment of software, and comparisons were performed for four third-party software packages, using a variety of surface function types. To address the computational limitations of modern CASs, the use of extended precision numerical methods was introduced to enable to high-accuracy

calculation of a wider range of parameters, and higher-complexity surfaces. The open-source Chebyshev polynomial approximation software Chebfun was used, and comparisons were performed against the mathematically-traceable values, showing agreement to fourteen significant figures. The chebfun method was then used to obtained reference parameter values for five complex surfaces, and again, comparisons were made for four third-party software packages to assess their individual strengths and weaknesses. Finally, chapter 5 also introduced parametric surface definitions, which give user-adjustable simple surface definitions that each correspond to a limited parametric parameter set. These are not general case solutions, but instead aim to offer fully mathematically-traceable surface-parameter reference pairs, and offer a degree of variability to the end user in terms of the parametric components of the surface definitions.

Chapters 4 and 5 successfully introduced surface-parameter reference pairs, however, proper assessment of third-party surface texture parameter software was still required. Chapter 6 addressed the performance assessment of parameter software, taking the discretisation error introduced when sampling a discrete dataset from an analytical surface representation into account. A novel parameter extrapolation method was introduced, which reduced the effective size of the discretisation error present in a parameter value calculation by approximating the value obtained from a much higher resolution dataset. This method required a large number of parameter value calculations in order to obtain a parameter-resolution relationship and was found to be very dependent on the quality of the fit applied. As an alternative, a simple performance metric was developed that identified to number of significant figures in the software-obtained parameter values that agree with the reference value, at a given dataset resolution. This method was applied to two analytical reference surfaces for a set of nine parameters and allowed for direct comparison between third-party software. In addition, assessing the total number of agreed significant figures for a full parameter set allowed for a single performance metric that described the performance of the software

as a whole for any particular surface. Finally, chapter 6 described the effect of sampling implementation on resulting software-obtained parameter values, and emphasised the need for detailed descriptions of any sampling methods used, alongside dataset resolution, in order to properly and repeatably assess the performance of surface texture parameter software. A comparison was presented of two sampling methods which showed marked differences in the performance metric values, further highlighting the importance of explicitly defining the sampling methods used during performance assessment.

7.3 Areas for future work

The current scope of the work presented in this thesis covers the calculation of areal surface texture parameters. This is only one step in the surface texture characterisation chain, as discussed in chapter 2, which also includes form removal and surface filtration to extract the spatial frequency regions of interest. Each of these steps are also often performed using metrology software and can significantly affect the final parameter values that are obtained by the software. It is, therefore, equally important that these aspects of the software are validated, and their performance is assessed. A strong area for potential future work in this field is to extend the idea of mathematically traceable reference pairs for both form removal and filtration. This can be achieved with a series of simple analytical surfaces, pre-form removal and pre-filtration, upon which mathematical operations are performed to produce another set of analytical surfaces with their form removal or filtration successfully applied. These pre/post surfaces will act as reference pairs: the pre-process surface can be input into software under test and the resulting surfaces can be quantitatively compared to the post-process references. The development of traceable mathematical references for both form removal and filtration would complete the surface texture characterisation chain and allow for accurate software performance assessment for each stage of the process.

The work performed in this thesis, namely the reference pair development given in chapters D, 4 and 5, focusses only on areal surface texture parameters. This decision was made in order to focus on the more modern set of parameters, and also to increase the complexity and novelty of the work. Future work could adapt the work presented in this thesis to profile surface texture parameters. The reduction in dimensional complexity should streamline the adaptation, and potentially allow for an extended set of parameters that are calculable for the general case. Adapting to profile surface texture parameters is a valuable future task, as appendix B presented a significant increase in the use of profile surface texture parameters since 1999, and showed that profile parameters are still the most commonly used parameters in industry. The ISO technical committee assigned to developing and maintaining profile and areal surface texture standards are currently working on updating the ISO 4287 profile surface texture parameter standards to make them profile equivalents of the areal standards, ensuring comparable terms and definitions are used for both parameter sets. This work will further streamline this potential future work of creating mathematical references for profile surface texture parameters, as the work for areal parameters can be adapted more directly.

While chapter 4 focussed on functional areal surface texture parameters, and chapter 5 focussed on field areal surface texture parameters, feature areal surface texture parameter were not covered. Feature parameters are more complex than the other parameter groups, and often depend on particular algorithms, for example Wolf pruning [42], that are not mathematically defined. Adapting these algorithmic definitions for an analytical surface representation is a substantial task that was not included in the scope of this thesis. Future work could address these challenges, and develop mathematical reference for areal feature parameters, in a similar way as has been performed here for field and functional parameters.

In conclusion, the work performed as part of this thesis was successful in achieving its primary goal of developing a novel, high accuracy method for the validation of areal

surface texture parameter calculation software. The comparisons against third-party software gave examples demonstrating the value of using the method to assess the performance of software to a traceable standard, and performance assessment methods were successfully able to identify strengths and weakness in individual software packages, delivering valuable information to both software developers and industrial end-users. The use of high-accuracy software validation techniques, such as those presented in this work, will help to improve the accuracy of third-party software used internationally throughout the manufacturing industry, and will enable better inter-company collaborations, ultimately reducing waste and improving part performance. Chapter 3 performed an in-depth comparison of profile surface texture parameter reference software developed by three independent NMIs. This work used a selection of NMI-developed reference surface datasets and investigated the differences in parameter values obtained. In addition to finding differences in the desired input dataset format between reference software (despite each supporting the internationally standardised .SMD file format as defined in ISO 5436-2 [14]), differences in parameter values were found between each of the three reference software packages, with possible reasons for each of these variations suggested. Despite these differences, each of the three software packages are regarded as NMI-endorsed references and can be used to validate third-party surface texture parameter calculation software. This can lead to variations and disagreements between validated software, which can cause manufacturing errors and increased material waste.

7.4 Contribution to the field

The main goal of the work presented in this thesis was to develop a novel method for the validation of areal surface texture parameter calculation software that improves upon the current state of the art. By introducing mathematically traceable, analytically defined surface-parameter reference pairs, third-party software could be compared against a high accuracy reference that can be evaluated by anyone. This novel method required the development of

analytical, two-dimensional equations that represented heights on a surface, which could then be evaluated using a series of analytical algorithms can conform to the mathematical parameter definitions given in ISO 25178-2.

The first development of mathematical surface-parameter reference pairs was introduced in chapter 4 for functional surface texture parameters based on the material ratio curve. By using the simple material ratio curve as the analytical definition instead of a surface height function, all functional surface texture parameters can be obtained by evaluating the material ratio curve directly. Therefore, traceable, analytically defined reference pairs were created that can be used as infinite precision references against which third-party software can be compared. By linearly sampling the analytical definition for the material ratio curve, height values can be obtained that can be converted into a discrete dataset for input into third-party software for assessment. Comparisons between three third-party surface texture parameter calculation packages were performed, with the results presented relative to the mathematically-obtained reference values. These comparisons showed how each software deviated from the reference, identifying strengths and weaknesses that can be addressed by the developer.

Chapter 5 presented a method for the creation of mathematical reference surfaces and applied it to the calculation areal field surface texture parameters. A variety of mathematical operations were presented (in some ideal cases identical to the ISO 25178-2 definitions) that utilise CASs to obtain parameter values for general case surface representations. Similar to chapter 4, these produce mathematically traceable reference pairs for the assessment of software, and comparisons were performed for four third-party software packages, using a variety of surface function types. To address the computational limitations of modern CASs, the use of extended precision numerical methods was introduced to enable to high-accuracy calculation of a wider range of parameters, and higher-complexity surfaces. The open-source Chebyshev polynomial approximation software Chebfun was used, and comparisons

were performed against the mathematically-traceable values, showing agreement to fourteen significant figures. The chebfun method was then used to obtained reference parameter values for five complex surfaces, and again, comparisons were made for four third-party software packages to assess their individual strengths and weaknesses. Finally, chapter 5 also introduced parametric surface definitions, which give user-adjustable simple surface definitions that each correspond to a limited parametric parameter set. These are not general case solutions, but instead aim to offer fully mathematically-traceable surface-parameter reference pairs, and offer a degree of variability to the end user in terms of the parametric components of the surface definitions.

Chapters 4 and 5 successfully introduced surface-parameter reference pairs, however, proper assessment of third-party surface texture parameter software was still required. Chapter 6 addressed the performance assessment of parameter software, taking the discretisation error introduced when sampling a discrete dataset from an analytical surface representation into account. A novel parameter extrapolation method was introduced, which reduced the effective size of the discretisation error present in a parameter value calculation by approximating the value obtained from a much higher resolution dataset. This method required a large number of parameter value calculations in order to obtain a parameter-resolution relationship and was found to be very dependent on the quality of the fit applied. As an alternative, a simple performance metric was developed that identified to number of significant figures in the software-obtained parameter values that agree with the reference value, at a given dataset resolution. This method was applied to two analytical reference surfaces for a set of nine parameters and allowed for direct comparison between third-party software. In addition, assessing the total number of agreed significant figures for a full parameter set allowed for a single performance metric that described the performance of the software as a whole for any particular surface. Finally, chapter 6 described the effect of sampling implementation on resulting software-obtained parameter values, and emphasised the need

for detailed descriptions of any sampling methods used, alongside dataset resolution, in order to properly and repeatably assess the performance of surface texture parameter software. A comparison was presented of two sampling methods which showed marked differences in the performance metric values, further highlighting the importance of explicitly defining the sampling methods used during performance assessment.

7.5 Areas for future work

The current scope of the work presented in this thesis covers the calculation of areal surface texture parameters. This is only one step in the surface texture characterisation chain, as discussed in chapter 2, which also includes form removal and surface filtration to extract the spatial frequency regions of interest. Each of these steps are also often performed using metrology software and can significantly affect the final parameter values that are obtained by the software. It is, therefore, equally important that these aspects of the software are validated, and their performance is assessed. A strong area for potential future work in this field is to extend the idea of mathematically traceable reference pairs for both form removal and filtration. This can be achieved with a series of simple analytical surfaces, pre-form removal and pre-filtration, upon which mathematical operations are performed to produce another set of analytical surfaces with their form removal or filtration successfully applied. These pre/post surfaces will act as reference pairs: the pre-process surface can be input into software under test and the resulting surfaces can be quantitatively compared to the post-process references. The development of traceable mathematical references for both form removal and filtration would complete the surface texture characterisation chain and allow for accurate software performance assessment for each stage of the process.

The work performed in this thesis, namely the reference pair development given in chapters D, 4 and 5, focusses only on areal surface texture parameters. This decision was made in order to focus on the more modern set of parameters, and also to increase the

complexity and novelty of the work. Future work could adapt the work presented in this thesis to profile surface texture parameters. The reduction in dimensional complexity should streamline the adaptation, and potentially allow for an extended set of parameters that are calculable for the general case. Adapting to profile surface texture parameters is a valuable future task, as appendix B presented a significant increase in the use of profile surface texture parameters since 1999, and showed that profile parameters are still the most commonly used parameters in industry. The ISO technical committee assigned to developing and maintaining profile and areal surface texture standards are currently working on updating the ISO 4287 profile surface texture parameter standards to make them profile equivalents of the areal standards, ensuring comparable terms and definitions are used for both parameter sets. This work will further streamline this potential future work of creating mathematical references for profile surface texture parameters, as the work for areal parameters can be adapted more directly.

While chapter 4 focussed on functional areal surface texture parameters, and chapter 5 focussed on field areal surface texture parameters, feature areal surface texture parameter were not covered. Feature parameters are more complex than the other parameter groups, and often depend on particular algorithms, for example Wolf pruning [42], that are not mathematically defined. Adapting these algorithmic definitions for an analytical surface representation is a substantial task that was not included in the scope of this thesis. Future work could address these challenges, and develop mathematical reference for areal feature parameters, in a similar way as has been performed here for field and functional parameters.

In conclusion, the work performed as part of this thesis was successful in achieving its primary goal of developing a novel, high accuracy method for the validation of areal surface texture parameter calculation software. The comparisons against third-party software gave examples demonstrating the value of using the method to assess the performance of software to a traceable standard, and performance assessment methods were successfully

able to identify strengths and weakness in individual software packages, delivering valuable information to both software developers and industrial end-users. The use of high-accuracy software validation techniques, such as those presented in this work, will help to improve the accuracy of third-party software used internationally throughout the manufacturing industry, and will enable better inter-company collaborations, ultimately reducing waste and improving part performance.

References

[1] ISO 25178-2 2012 *Geometrical product specifications (GPS) - Surface texture: Areal Part 2: Terms, definitions and surface texture parameters* (Geneva: International Organisation for Standardisation)

[2] Bruzzone A A G, Costa H L, Lonardo P M and Lucca D A 2008 Advances in engineered surfaces for functional performance *Ann. CIRP* **57** 750–769

[3] Blunt L, Jiang X and Scott P J 2005 Advances in micro and nano-scale surface metrology *Key Engineering Materials* vol 295 (Trans Tech Publ) pp 431–436

[4] Menezes P L, Kailas S V, Lovell M R *et al.* 2011 Role of surface texture, roughness, and hardness on friction during unidirectional sliding *Tribology letters* **41** 1–15

[5] Pettersson U and Jacobson S 2003 Influence of surface texture on boundary lubricated sliding contacts *Tribology International* **36** 857–864

[6] Holmberg K, Andersson P and Erdemir A 2012 Global energy consumption due to friction in passenger cars *Tribology International* **47** 221–234

[7] Holmberg K and Erdemir A 2017 Influence of tribology on global energy consumption, costs and emissions *Friction* **5** 263–284

[8] Leach R K 2014 *Fundamental Principles of Engineering Nanometrology* 2nd ed (Binghamton: Elsevier Science)

[9] Merlo A M 2003 The contribution of surface engineering to the product performance in the automotive industry *Surface and Coatings Technology* **174 - 175** 21–26

[10] Ronen A, Etsion I and Kligerman Y 2001 Friction-reducing surface-texturing in reciprocating automotive components *Tribology Transactions* **44** 359–366

[11] Roach P, Farrar D and Perry C C 2006 Surface tailoring for controlled protein adsorption: effect of topography at the nanometer scale and chemistry *Journal of the American Chemical Society* **128** 3939–3945

[12] Sutherland D S, Broberg M, Nygren H and Kasemo B 2001 Influence of nanoscale surface topography and chemistry on the functional behaviour of an adsorbed model macromolecule *Macromolecular Bioscience* **1** 270–273

[13] ISO 4287 1998 *Geometrical Product Specifications (GPS) - Surface texture: Profile method - Terms, definitions and surface texture parameters* (Geneva: International Organisation for Standardisation)

[14] ISO 5436-2 2012 *Geometrical product specifications (GPS) - Surface texture: Profile method; Measurement standards - Part 2: Software measurement standards* (Geneva: International Organisation for Standardisation)

[15] ISO 25178-71 2012 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 71: Software measurement standards* (Geneva: International Organisation for Standardisation)

[16] De Chiffre L 1999 Industrial survey on ISO surface texture parameters *CIRP Annals* **48** 74–77

[17] ISO 12085 1997 *Geometric Product Specifications (GPS) - Surface texture: Profile method - Motif Parameters* (Geneva: International Organisation for Standardization)

[18] ISO 13565-2 1997 *Geometric Product Specifications (GPS) - Surface texture: Profile method - Surfaces having stratified functional properties* (Geneva: International Organisation for Standardization)

[19] Leach R K 2013 *Characterisation of Areal Surface Texture* 1st ed (Berlin: Springer Berlin Heidelberg)

[20] Blunt L and Jiang X 2003 *Advanced Techniques for Assessment Surface Topography: Development of a Basis for 3D Surface Texture Standards "SURFSTAND"* (London: Kogan Page Science)

[21] Whitehouse D J 2010 *Handbook of Surface and Nanometrology* 2nd ed (Boca Raton: CRC Press)

[22] Leach R K 2011 *Optical Measurement of Surface Topography* (Berlin: Springer)

[23] Flack D and Hannaford J 2005 Fundamental good practice in dimensional metrology *Measurement Good Practice Guide, NPL*

[24] 2006 *The International System of Units (SI)* 8th ed (Paris: International Bureau of Weights and Measures)

[25] VIM I 2004 International vocabulary of basic and general terms in metrology (vim) *International Standards Organization*

[26] Leach R K 1999 Calibration, traceability and uncertainty issues in surface texture metrology. *NPL Report CLM 7*

[27] Harris P, Leach R and Giusca C 2010 Uncertainty evaluation for the calculation of a surface texture parameter in the profile case *NPL Report MS* **8**

[28] ISO 3274 1996 *Geometrical product specifications (GPS) - Surface texture: Profile method - Nominal characteristics of contact (stylus) instruments* (Geneva: International Organisation for Standardisation)

[29] ISO 25178-1 2016 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 1: Indication of surface texture* April (Geneva: International Organisation for Standardisation)

[30] ISO 25178-6 2010 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 6: Classification of methods for measuring surface texture* (Geneva: International Organisation for Standardisation)

[31] ISO 25178-602 2010 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 602: Nominal characteristics of non-contact (confocal chromatic probe) instruments* (Geneva: International Organisation for Standardisation)

[32] ISO 25178-605 2014 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 605: Nominal characteristics of non-contact (point autofocus probe) instruments* (Geneva: International Organisation for Standardisation)

[33] ISO 25178-603 2013 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 603: Nominal characteristics of non-contact (phase-shifting interferometric microscopy) instruments* October (Geneva: International Organisation for Standardisation)

[34] ISO 25178-604 2013 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 604: Nominal characteristics of non-contact (coherence scanning interferometry) instruments* August (Geneva: International Organisation for Standardisation)

[35] ISO 25178-606 2015 *Geometrical product specification (GPS) - Surface texture: Areal - Part 606: Nominal characteristics of non-contact (focus variation) instruments* June (Geneva: International Organisation for Standardisation)

[36] ISO 25178-607 2017 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 607: Nominal characteristics of non-contact (confocal microscopy) instruments.* (Geneva: International Organisation for Standardisation)

[37] Thompson A, Senin N, Giusca C and Leach R 2017 Topography of selectively laser melted surfaces: a comparison of different measurement methods *CIRP Annals* **66** 543–546

[38] ISO 4288 1998 *Geometric Product Specification (GPS) - Surface texture - Profile method: Rules and procedures for the assessment of surface texture* (Geneva: International Organisation for Standardisation)

[39] Leach R K 2014 The measurement of surface texture using stylus instruments *Measurement Good Practice Guide No. 37, NPL*

[40] ISO 25178-3 2012 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 3: Specification operators* July (Geneva: International Organisation for Standardisation)

[41] Whitehouse D J 1994 *Handbook of surface metrology* (Bristol: Institute of Physics)

[42] Wolf G W 1991 A fortran subroutine for cartographic generalization *Computers & Geosciences* **17** 1359–1381

[43] Brown C A, Hansen H N, Jiang X J, Blateyron F, Berglund J, Senin N, Bartkowiak T, Dixon B, Le Goic G, Quinsat Y, Stemp W J, Thompson M K, Ungar P S and Zahouani E H 2018 Multiscale analyses and characterizations of surface topographies *CIRP annals* **67** 839–862

[44] Vorburger T 1992 Methods for characterizing surface topography *Tutorials in Optics* 137–151

[45] ISO 16610-21 2011 *Geometrical product specifications (GPS) - Filtration - Part 21: Linear profile filters: Gaussian filters* (Geneva: International Organisation for Standardisation)

[46] ISO 16610-60 2015 *Geometrical Product Specification (GPS) - Filtration - Part 60: Linear areal filters: Basic concepts* November (Geneva: International Organisation for Standardisation)

[47] ISO 16610-61 2015 *Geometrical product specification (GPS) - Filtration - Part 61: Linear areal filters - Gaussian filters* July (Geneva: International Organisation for Standardisation)

[48] ISO 25178-72 2015 *Geometrical product specifications (GPS) - Surface texture: Areal - Part 72: XML file format x3p* 0 (Geneva: International Organisation for Standardisation)

[49] Blunt L, Jiang X, Leach R K, Harris P and Scott P 2008 The development of user-friendly software measurement standards for surface topography software assessment *Wear* **264** 389–393

[50] Softgauges for surface topography http://www.npl.co.uk/science-technology/dimensional/services/softgauges-for-surface-topography/ (Date last accessed: 2018-08-09)

[51] Bui S H, Renegar T B, Vorburger T V, Raja J and Malburg M C 2004 Internet-based surface metrology algorithm testing system *Wear* **257** 1213–1218

[52] Bui S H and Vorburger T V 2007 Surface metrology algorithm testing system *Precision Engineering* **31** 218–225

[53] Internet based surface metrology algorithm testing system http://physics.nist.gov/VSC/jsp/index.jsp (Date last accessed: 2018-08-09)

[54] Jung L, Krüger-Sehm R, Spranger B and Koenders L 2004 Reference software for roughness analyses - features and results *XI International Colloquium on Surfaces* (Chemnitz, Germany) p 164

[55] Rptb - software to analyse roughness of profiles https://www.ptb.de/rptb (Date last accessed: 2018-08-09)

[56] Leach R K and Harris P M 2002 Ambiguities in the definition of spacing parameters for surface-texture characterization *Measurement Science and Technology* **13** 1924

[57] Baker A, Tan S L, Leach R K, Jung L, Wong S Y, Tonmueanwai A, Naoi K, Kim J, Renegar T B, Chaudhary K P, Kruger O, Amer M, Gao S, Tsai C L, Anh N and Drijarkara A 2013 Final report on apmp.l-k8: International comparison of surface roughness *Metrologia* **50** 04003

[58] Koenders L, Andreasen J L, De Chiffre L, Jung L and Krüger-Sehm R 2004 Euromet project 600 - comparison on surface roughness standards *Metrologia* **41** 04001

[59] Li T, Leach R K, Jung L, Jiang X and Blunt L 2009 Comparison of type f2 software measurement standards for surface texture *NPL Report ENG 16*

[60] Paricio I, Sanz-Lobera A and Lozano F 2015 Comparative analysis of software measurement standard according to iso 5436-2 *Procedia Engineering. MESIC Manufacturing Engineering Society International Conference 2015.* **132** 864–871

[61] Todhunter L D, Leach R K, Lawes S D A and Blateyron F 2017 An analysis of type F2 software measurement standards for profile surface texture parameters *Measurement Science and Technology* **28** 065017

[62] ISO 11562 1997 *Geometrical product specifications (GPS) - Surface texture: Profile method - Metrological characteristics of phase correct filters* (Geneva: International Organisation for Standardisation)

[63] Seewig J, Scott P J, Eifler M and Hueser D 2019 Crossing-the-line segmentation as a basis for roughness parameter evaluation *22nd International Conference on Metrology and Properties of Surfaces, Lyon* pp 77–78

[64] Todhunter L D, Leach R K, Lawes S, Harris P M and Blateyron F 2019 Mathematical approach to the validation of functional surface texture parameter software *Surface Topography: Metrology and Properties* **7** 015020

[65] Brennan J K, Crampton A, Jiang X, Leach R and Harris P 2005 Approximation of surface texture profiles *Journal of Physics: Conference Series* vol 13 (IOP Publishing) p 264

[66] Cardano G [1545] 1993 *Ars magna or the rules of algebra* (Dover Publications)

[67] Ţălu Ş, Stach S, Klaić B, Mišić T, Malina J and Čelebić A 2015 Morphology of co–cr–mo dental alloy surfaces polished by three different mechanical procedures *Microscopy research and technique* **78** 831–839

[68] Rîpă M, Tomescu L, Hapenciuc M and Crudu I 2003 Tribological characterisation of surface topography using abbott-firestone curve *Annals of University Dunărea de Jos of Galati, Fascicle VIII, Tribology* 208–212

[69] Cîrstoiu A C 2010 Surface roughness evaluation in turning based on abbott–firestone curve *The Romanian Review Precision Mechanics, Optics & Mechatronics* **38** 163–169

[70] Zhu S and Huang P 2017 Influence mechanism of morphological parameters on tribological behaviors based on bearing ratio curve *Tribology International* **109** 10–18

[71] Linares J, Goch G, Forbes A, Sprauel J, Clément A, Haertig F and Gao W 2018 Modelling and traceability for computationally-intensive precision engineering and metrology *CIRP Annals* **67** 815–838

[72] Zhou G and Lam N S N 2005 A comparison of fractal dimension estimators based on multiple surface generation algorithms *Computers & Geosciences* **31** 1260–1269

[73] McGaughey D R and Aitken G 2002 Generating two-dimensional fractional brownian motion using the fractional gaussian process (fgp) algorithm *Physica A: Statistical Mechanics and its Applications* **311** 369–380

[74] Fournier A, Fussell D and Carpenter L 1982 Computer rendering of stochastic models *Communications of the ACM* **25** 371–384

[75] Stein W and Joyner D 2005 Sage: System for algebra and geometry experimentation *Acm Sigsam Bulletin* **39** 61–64

[76] Wolfram Research, Inc 2017 *Mathematica, Version 11.1* (Champaign, Illinois: Wolfram Research, Inc)

[77] Maplesoft, a division of Waterloo Maple Inc 2017 *Maple 2017* (Waterloo, Ontario: Maplesoft)

[78] Blateyron F 2013 The areal field parameters *Characterisation of Areal Surface Texture* ed Leach R K (Berlin: Springer Berlin Heidelberg) chap 2, pp 15–41 1st ed

[79] Loehle C 2006 Global optimization using mathematica: A test of software tools *Mathematica in Education and Research* **11** 139–152

[80] Rosenlicht M 1968 Liouville's theorem on functions with elementary integrals *Pacific Journal of Mathematics* **24** 153–161

[81] Rosenlicht M 1972 Integration in finite terms *The American Mathematical Monthly* **79** 963–972

[82] Todhunter L D, Leach R K, Lawes S D A and Blateyron F 2017 Industrial survey of ISO surface texture parameters *CIRP Journal of Manufacturing Science and Technology* **19** 84–92

[83] Subbotin Y N 2011 Bernoulli polynomials *Encyclopedia of Mathematics*

[84] Kouba O 2013 Lecture notes, bernoulli polynomials and applications *arXiv preprint arXiv:1309.7560*

[85] Driscoll T A, Hale N and Trefethen L N 2014 *Chebfun Guide* (Oxford: Pafnuty Publications)

[86] Townsend A and Trefethen L N 2013 An extension of Chebfun to two dimensions *SIAM Journal on Scientific Computing* **35** C495–C518

[87] Powell M J D 1981 *Approximation theory and methods* (Cambridge university press)

[88] Mason J C and Handscomb D C 2002 *Chebyshev polynomials* (Chapman and Hall/CRC)

[89] Trefethen L N 2007 Computing numerically with functions instead of numbers *Mathematics in Computer Science* **1** 9–19

[90] Kok G J P, Harris P M, Smith I M and Forbes A B 2016 Reference data sets for testing metrology software *Metrologia* **53** 1091–1100

[91] Cox M and Harris P 2010 Software support for metrology best practice guide no. 6. uncertainty evaluation. *National Physical Laboratory report MS6*

[92] ISO 13565-3 2000 *Geometrical Product Specifications (GPS) — Surface texture: Profile method - Surfaces having stratified functional properties — Part 3: Height characterization using the material probability curve* (Geneva: International Organisation for Standardization)

[93] CNOMO E0014015N 1993 *Etats geometriques de surface calcul des parametres de profil*

[94] Townsend A, Senin N, Blunt L, Leach R and Taylor J 2016 Surface texture metrology for metal additive manufacturing: a review *Precision Engineering* **46** 34–47

[95] Todhunter L D, Senin N, Leach R K, Lawes S, Blateyron F and Harris P 2018 A programmable software framework for the generation of simulated surface topography *18th International Conference of the European Society for Precision Engineering and Nanotechnology, Venice* (EUSPEN)

[96] Lewis J P 1989 Algorithms for solid noise synthesis *ACM SIGGRAPH Computer Graphics* vol 23 (ACM) pp 263–270

[97] Perlin K 1985 An image synthesizer *ACM Siggraph Computer Graphics* **19** 287–296

[98] Perlin K 2002 Improving noise *ACM transactions on graphics (TOG)* vol 21 (ACM) pp 681–682

[99] Cook R L and DeRose T 2005 Wavelet noise *ACM Transactions on Graphics (TOG)* vol 24 (ACM) pp 803–811

Appendix A

NMI reference software algorithm flow diagrams

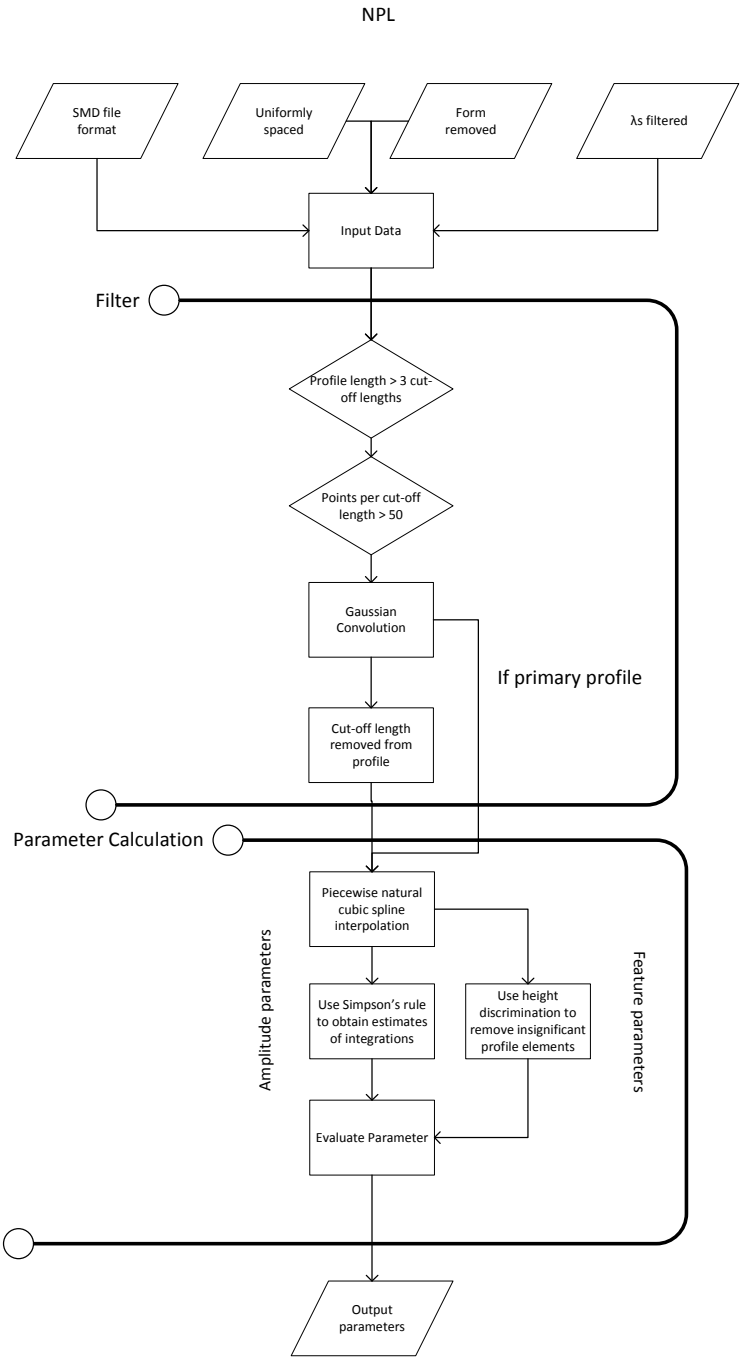
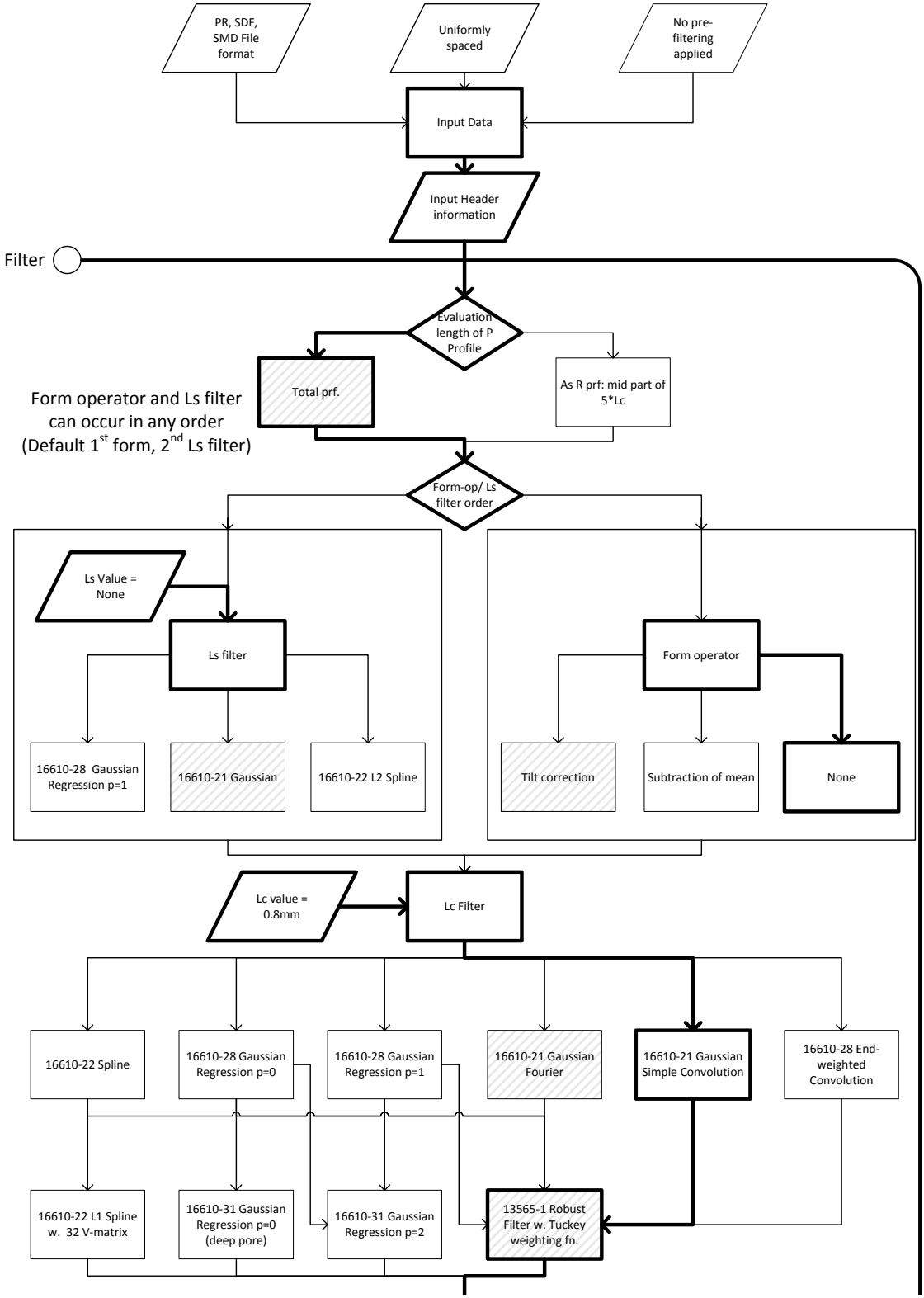


Fig. A.1 Algorithm flow diagram for the NPL softgauges software.


PR, SDF, SMD File format
Uniformly spaced
No pre-filtering applied
Input Data
Input Header information
Filter
Evaluation length of P Profile
Total prf.
As R prf: mid part of 5*Lc
Form operator and Ls filter can occur in any order (Default 1st form, 2nd Ls filter)
Form-op/ Ls filter order
Ls Value = None
Ls filter
16610-28 Gaussian Regression p=1
16610-21 Gaussian
16610-22 L2 Spline
Form operator
Tilt correction
Subtraction of mean
None
Lc value = 0.8mm
Lc Filter
16610-22 Spline
16610-28 Gaussian Regression p=0
16610-28 Gaussian Regression p=1
16610-21 Gaussian Fourier
16610-21 Gaussian Simple Convolution
16610-28 End-weighted Convolution
16610-22 L1 Spline w. 32 V-matrix
16610-31 Gaussian Regression p=0 (deep pore)
16610-31 Gaussian Regression p=2
13565-1 Robust Filter w. Tuckey weighting fn.

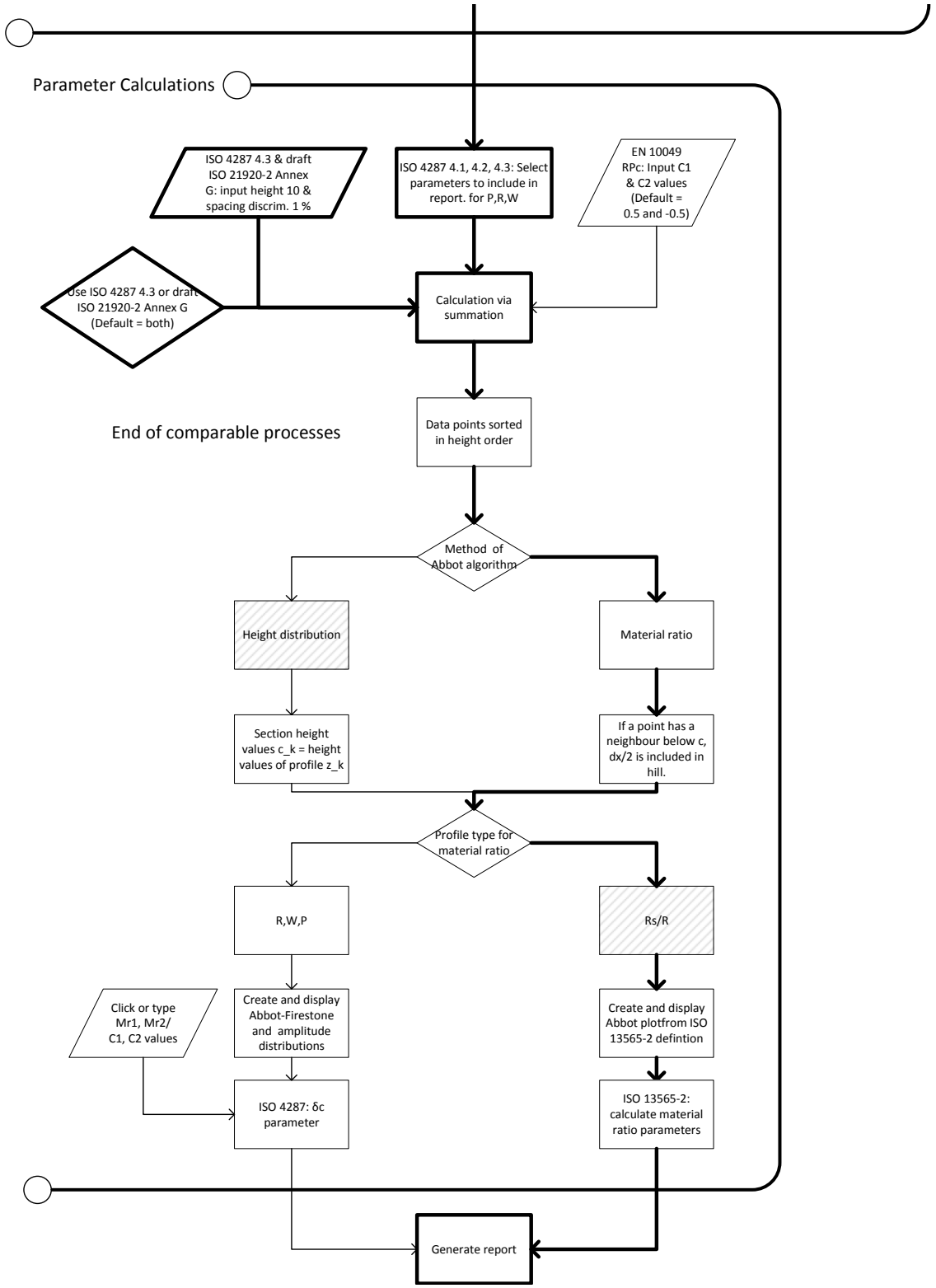
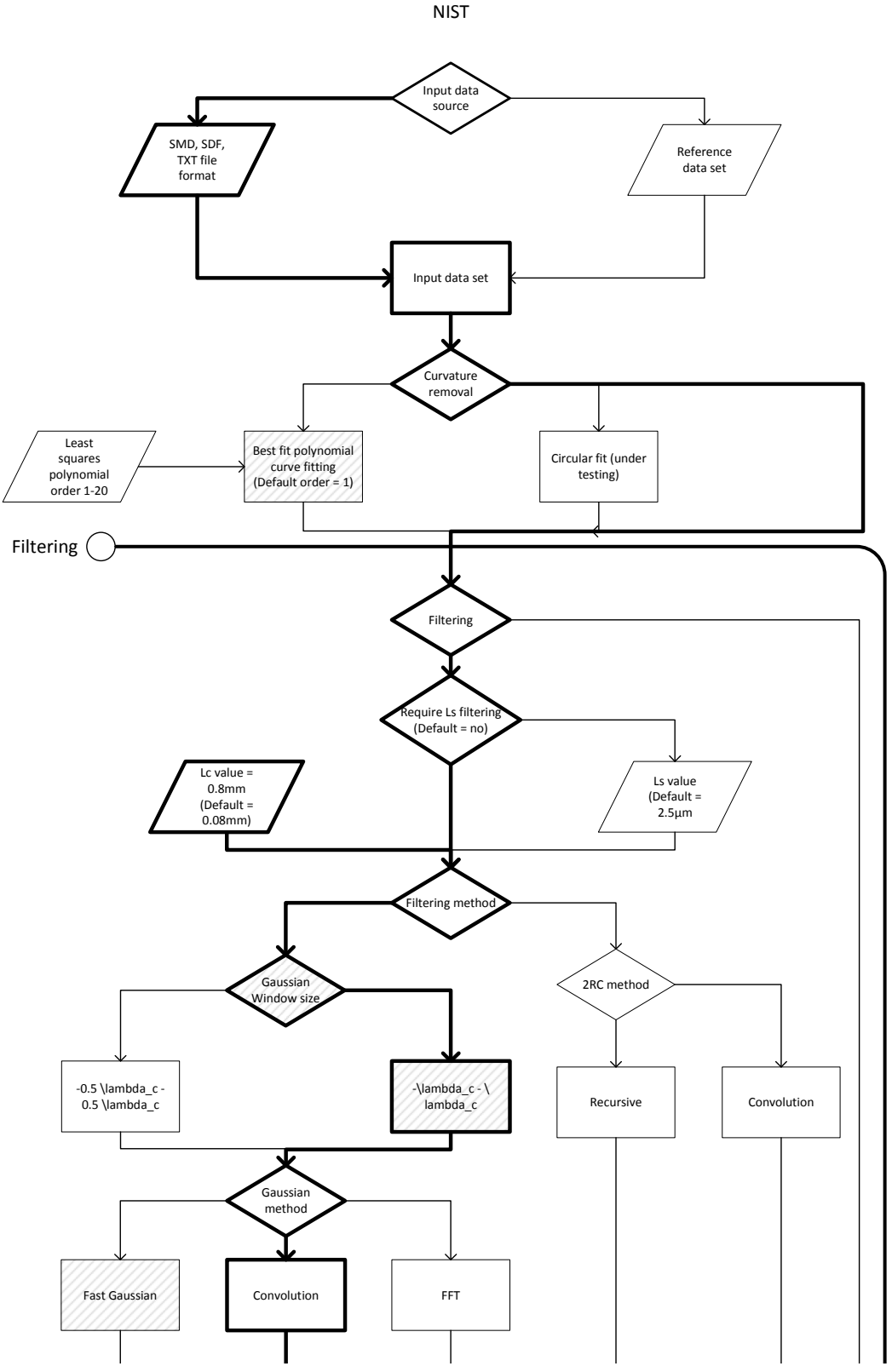
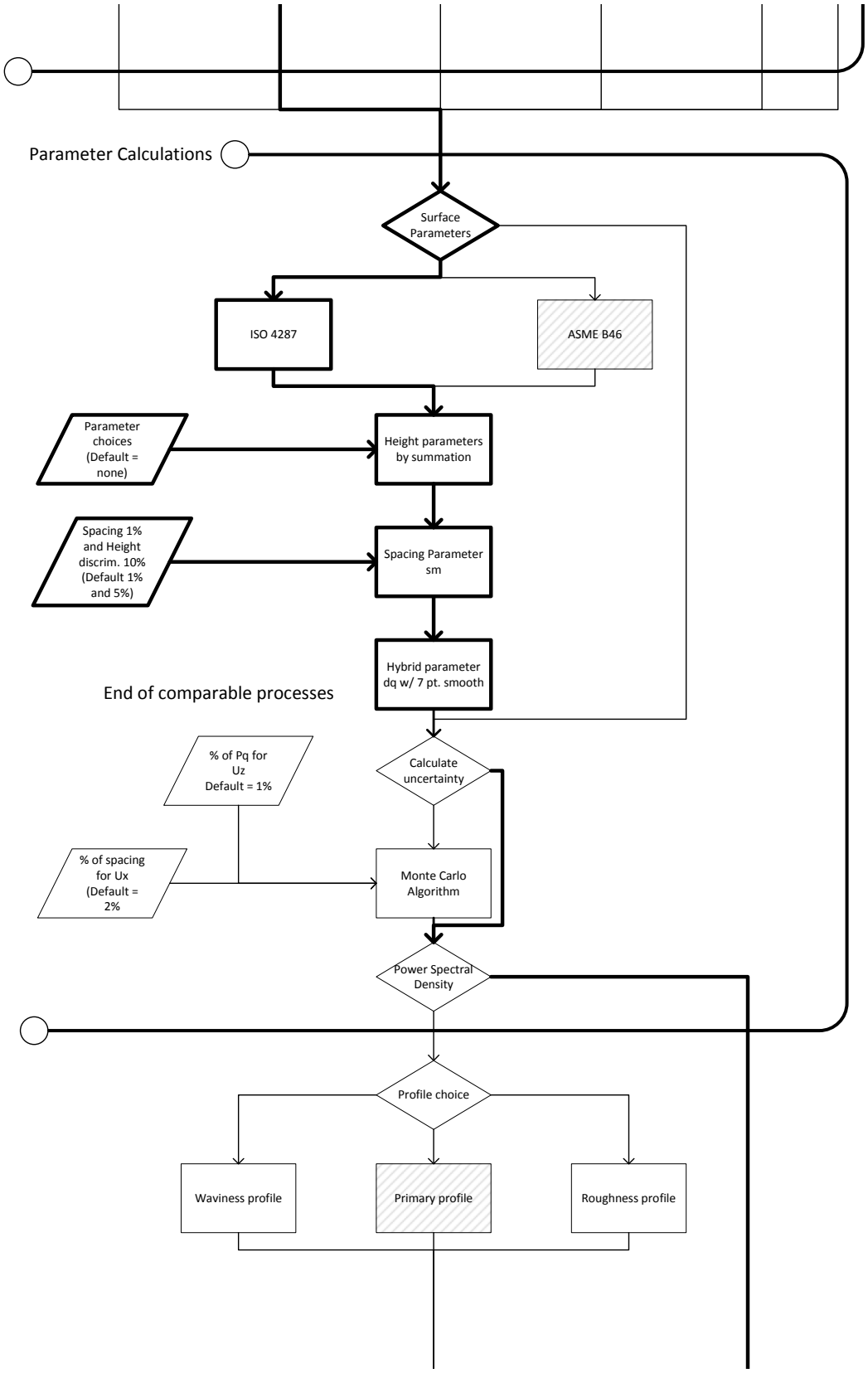
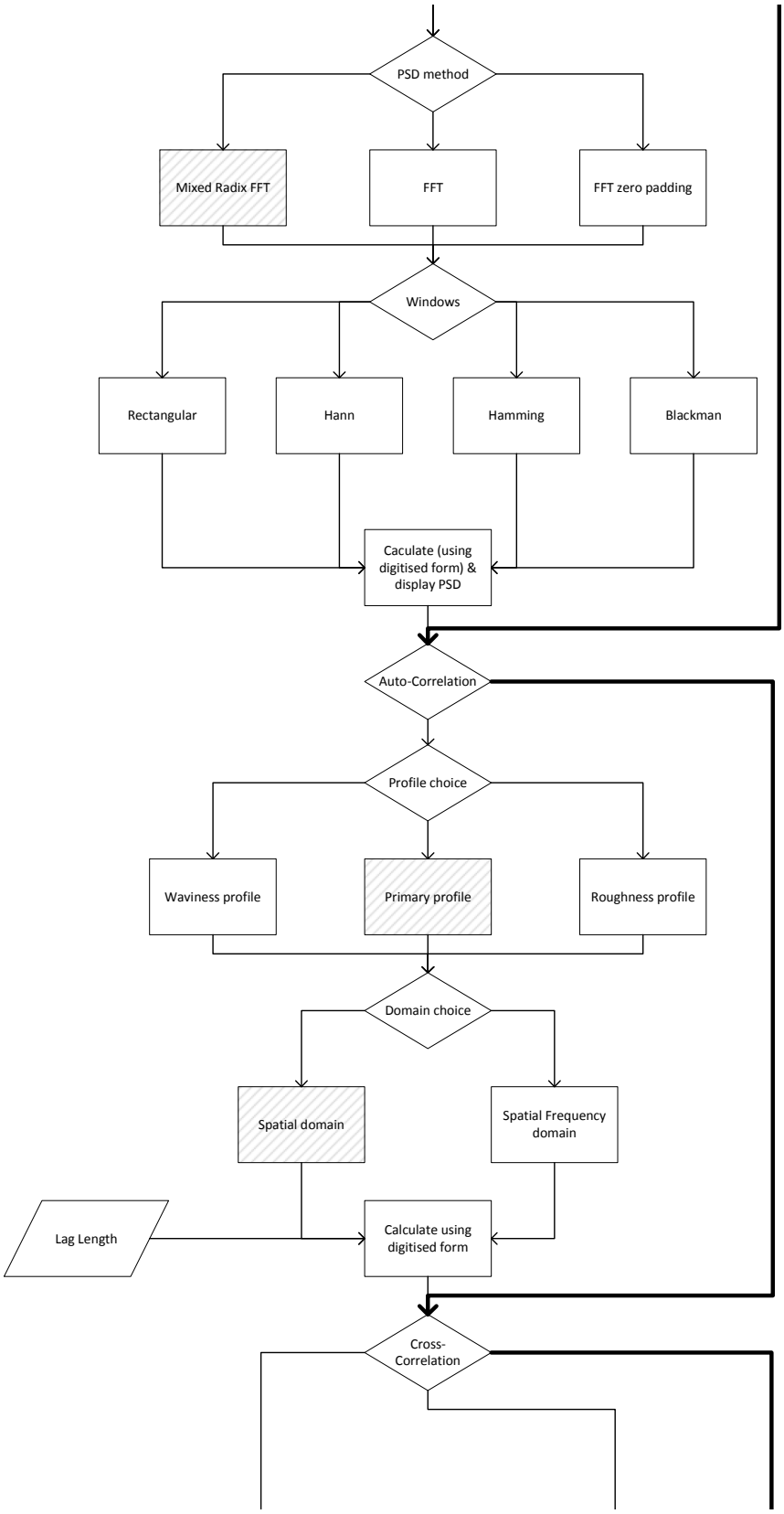


Fig. A.2 Algorithm flow diagram for the PTB RPTB software, showing the software route taken to best match the NPL software. Shaded areas show the default settings provided by the software.


NIST
Input data source
SMD, SDF, TXT file format
Reference data set
Input data set
Curvature removal
Least squares polynomial order 1-20
Best fit polynomial curve fitting (Default order = 1)
Circular fit (under testing)
Filtering
Filtering
Require Ls filtering (Default = no)
Lc value = 0.8mm (Default = 0.08mm)
Ls value (Default = 2.5µm
Filtering method
Gaussian Window size
2RC method
-0.5 \lambda_c - 0.5 \lambda_c
-\lambda_c - \ lambda_c
Recursive
Convolution
Gaussian method
Fast Gaussian
Convolution
FFT


Parameter Calculations
Surface Parameters
ISO 4287
ASME B46
Parameter choices (Default = none)
Height parameters by summation
Spacing 1% and Height discrim. 10% (Default 1% and 5%)
Spacing Parameter sm
Hybrid parameter dq w/ 7 pt. smooth
End of comparable processes
% of Pq for Uz Default = 1%
Calculate uncertainty
% of spacing for Ux (Default = 2%
Monte Carlo Algorithm
Power Spectral Density
Profile choice
Waviness profile
Primary profile
Roughness profile


PSD method
Mixed Radix FFT
FFT
FFT zero padding
Windows
Rectangular
Hann
Hamming
Blackman
Caculate (using digitised form) & display PSD
Auto-Correlation
Profile choice
Waviness profile
Primary profile
Roughness profile
Domain choice
Spatial domain
Spatial Frequency domain
Lag Length
Calculate using digitised form
Cross-Correlation

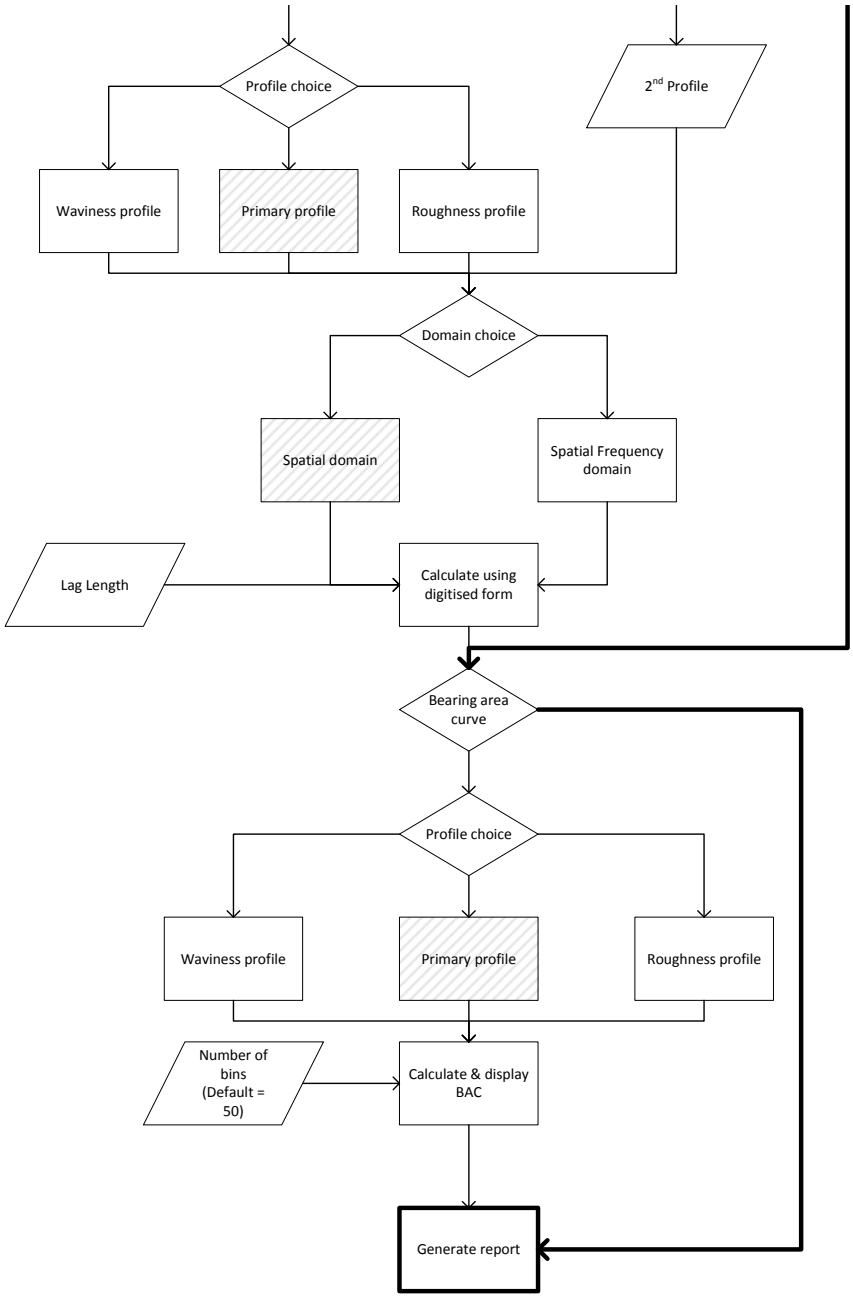


Fig. A.3 Algorithm flow diagram for the NIST SMATS software, showing the software route taken to best match the NPL software. Shaded areas show the default settings provided by the software.

Appendix B

International case study into the industrial adoption of ISO surface texture parameters

The work presented in this appendix has been published in reference [82].

B.1 Surface texture parameter survey

This work used an on-line based survey to obtain information from voluntary respondents about the surface texture parameters that they use. Invitations to complete the survey were sent out by email to suitable correspondents. The survey was mainly aimed at CIRP industry contacts, as this provided opportunity for responses from a wide range of relevant disciplines. The survey was later expanded to a much wider audience to increase the number of responses, for example by advertising the survey to conference attendees. The survey included parameters from ISO 4287, ISO 12085 and ISO 13562-2, similar to the 1999 survey, but also included ISO 13565-3 [92] and areal parameters from ISO 25178-2. In addition, the survey requested details about the respondents' company, and also gave the option for

respondents to share their opinions on the current selection of surface texture parameters available. A copy of the online form is included in appendix C.

The survey was open to responses for eight months, from March 2016 to November 2016, and obtained a total of 179 responses from a variety of industrial users spread internationally across thirty-four countries. The distribution of responses is shown in Table B.1. The responses come from a variety of disciplines and countries and serve as a viable sample from which to learn about current surface texture parameter use, delivering a useful update to the original 1999 survey.

B.2 Results

B.2.1 Participant details

Figure B.1 shows the company sizes of the participants, in terms of number of employees. The majority of responses were from large companies, with 55%. This is in contrast to the 1999 survey, in which medium sized companies gave the most responses with 45%.

Information about the participants' instrument usage is shown in Figure B.2. The results indicate that whilst contact instruments are still popular, optical instruments have seen significant adoption, with 66% of respondents using them either exclusively or in conjunction with contact instruments. As optical instruments are often areal in nature, it comes as no surprise that areal instrument modes have seen similar adoption, again with 66% using some form of areal instrument mode.

Figure B.3 shows how the survey responses were split in terms of industry sector. The most responses for this survey came from research institutions, such as universities. These operate in many of the other disciplines listed, but focus on none in particular. Other sectors that gave a notable number of responses were 'metrology & calibration', 'automotive & aerospace' and 'product manufacturing'. The 'product manufacturing' category is a broad

Table B.1 List of participating countries and the number of responses from each.

Country	**Responses**
Algeria	1
Austria	1
Belgium	3
Brazil	3
Canada	1
China	4
Czech Republic	1
Denmark	11
Finland	1
France	11
Germany	22
Honk Kong	1
Hungary	1
India	5
Ireland	1
Israel	1
Italy	7
Japan	1
Japan	1
South Korea	5
Latvia	1
Luxembourg	1
Mexico	1
Netherlands	4
Poland	1
Portugal	1
Russia	1
South Africa	1
Spain	7
Sweden	12
Switzerland	9
United Kingdom	17
United States	40
Not specified	1
Total	179

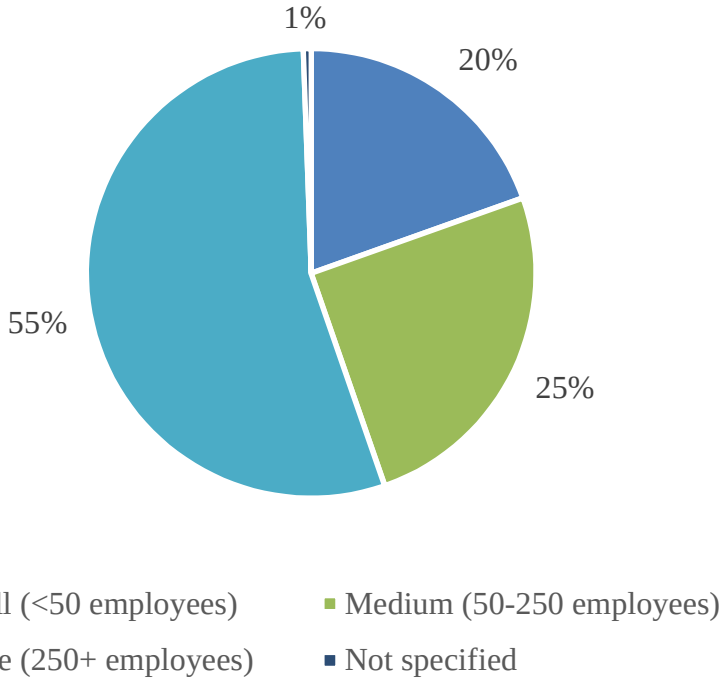


Fig. B.1 The percentage of responses to the survey with a given company size.

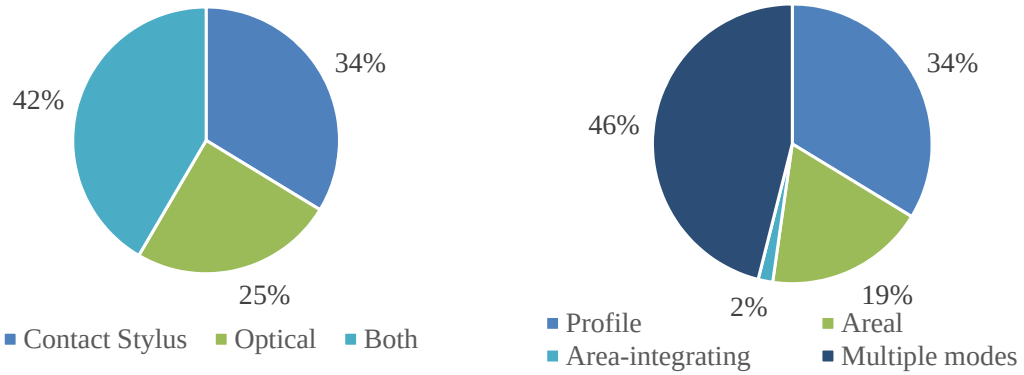


Fig. B.2 *Left*: Instrument type usage as a percentage of responses. *Right*: Instrument mode usage as a percentage of responses.

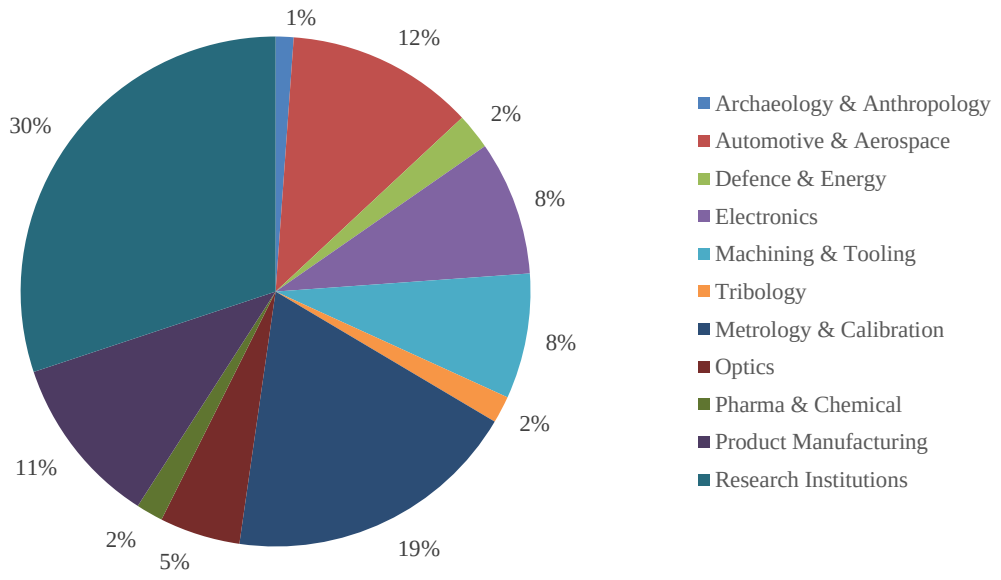


Fig. B.3 Industry sectors represented in the survey as a percentage of total number of responses.

sector encompassing participant companies who produce consumer or specialist goods or materials, but do not specifically operate in one of the other industrial sectors. Several of the sectors here are comprised of only a few participants, such as 'tribology' and 'archaeology & anthropology', and as such are poorly represented. Individual analysis of these sectors would not provide useful information, and consequently they have been omitted from any individual analyses.

B.2.2 Profile surface texture parameters

Before discussing the adoption of the recently introduced areal surface texture parameters, it is first useful to analyse the profile case. As many of the profile parameters were the subject of the 1999 survey, this analysis has the ability to compare the current profile parameter usage to that of 1999 and obtain insight into the evolution of parameter usage in industry.

B.2.2.1 ISO 4287 profile parameters

Figure B.4 gives the survey responses for the usage of the ISO 4287 profile surface texture parameters for both 2016 and 1999, as a percentage of the total number of responses. For example, 15% of the 179 participants in the 2016 survey used the *Pz* parameter, compared to <5% of the 284 participants in the 1999 survey. The most immediate conclusion to draw from the results is the unanimous increase in parameter usage across all parameters. Such a result shows the increased awareness of surface texture parameters in industry, and their more widespread use.

As expected, the *Ra* parameter and the primary/waviness profile equivalents remain the most popular parameter used, however, since 1999 a significant relative increase is seen for the less well-known parameters, such as the skewness and kurtosis parameters *Xsk* and *Xku*. This indicates a much greater understanding of the surface texture parameters as a whole, and their individual uses. This being said, it should be considered that this widespread adoption of all surface texture parameters could be due to the exponential increase in computational power available to measurement instrument users since 1999, allowing them to calculate all parameters with relative ease. This blanket approach to parameter selection would unfortunately, therefore, indicate no further understanding of individual surface texture parameters than the results obtained in 1999. This over-abundance of calculated parameter values has been termed 'parameter rash' [41], and serves to deliver no more insight into the required information from a surface measurement than the use of no parameters. Instead, parameter values should be displayed with an intended purpose.

An interesting observation from the results is the significant increase in the use of waviness parameters compared to 1999. With the exception of the *Wt* parameter, which was used by ~15% of participants, most waviness parameters were hardly used in 1999, with only ~1% of participants indicating parameter use. The new results show a much greater uptake, with waviness parameters on average used by 12% of participants.

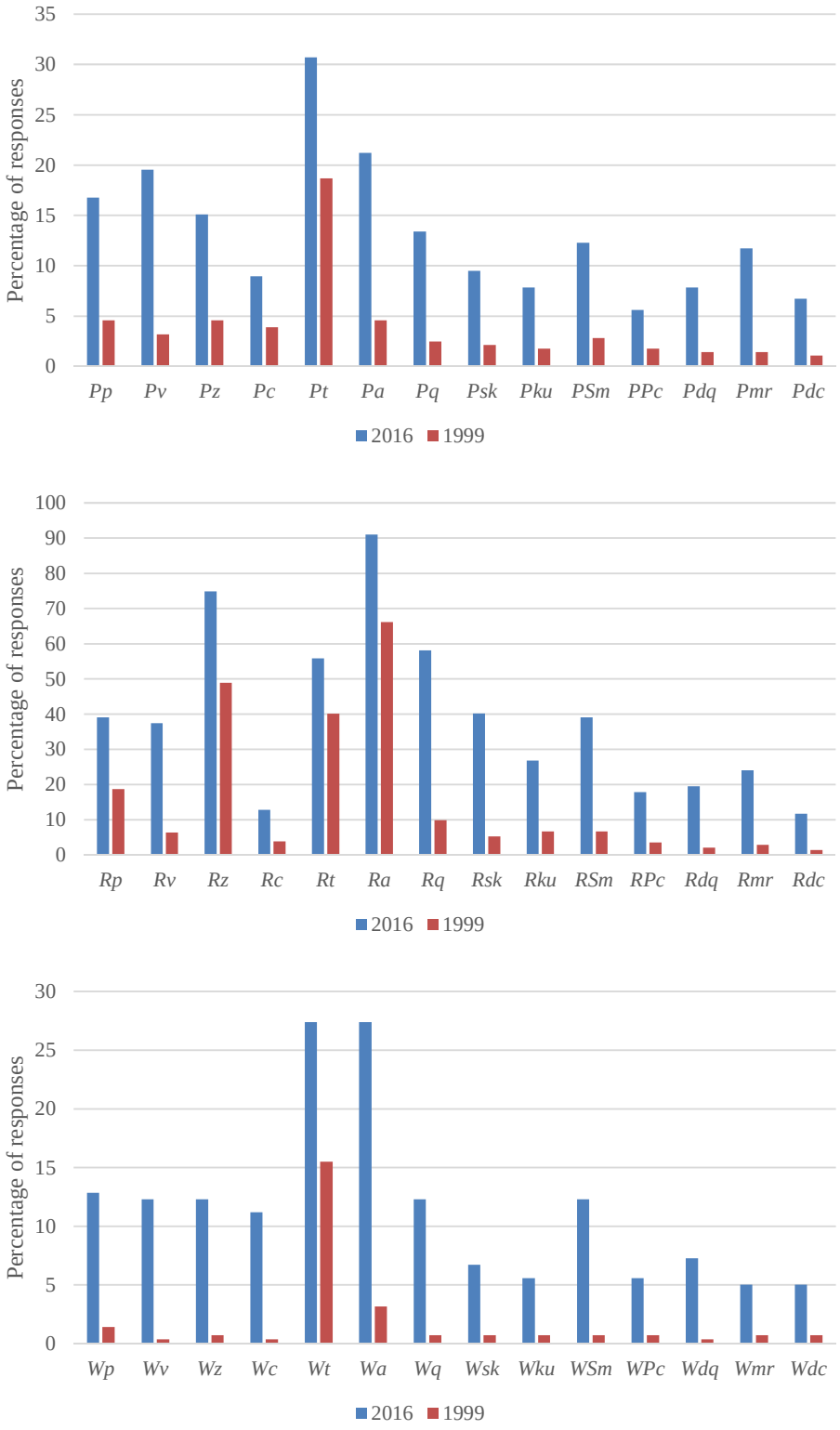


Fig. B.4 Responses that indicated use of ISO 4287 parameters for the 2016 (blue) and 1999 (red) surveys, given as the percentage of the total number of responses that use each parameter. *Top*: Primary parameters. *Middle*: Roughness parameters. *Bottom*: Waviness parameters.

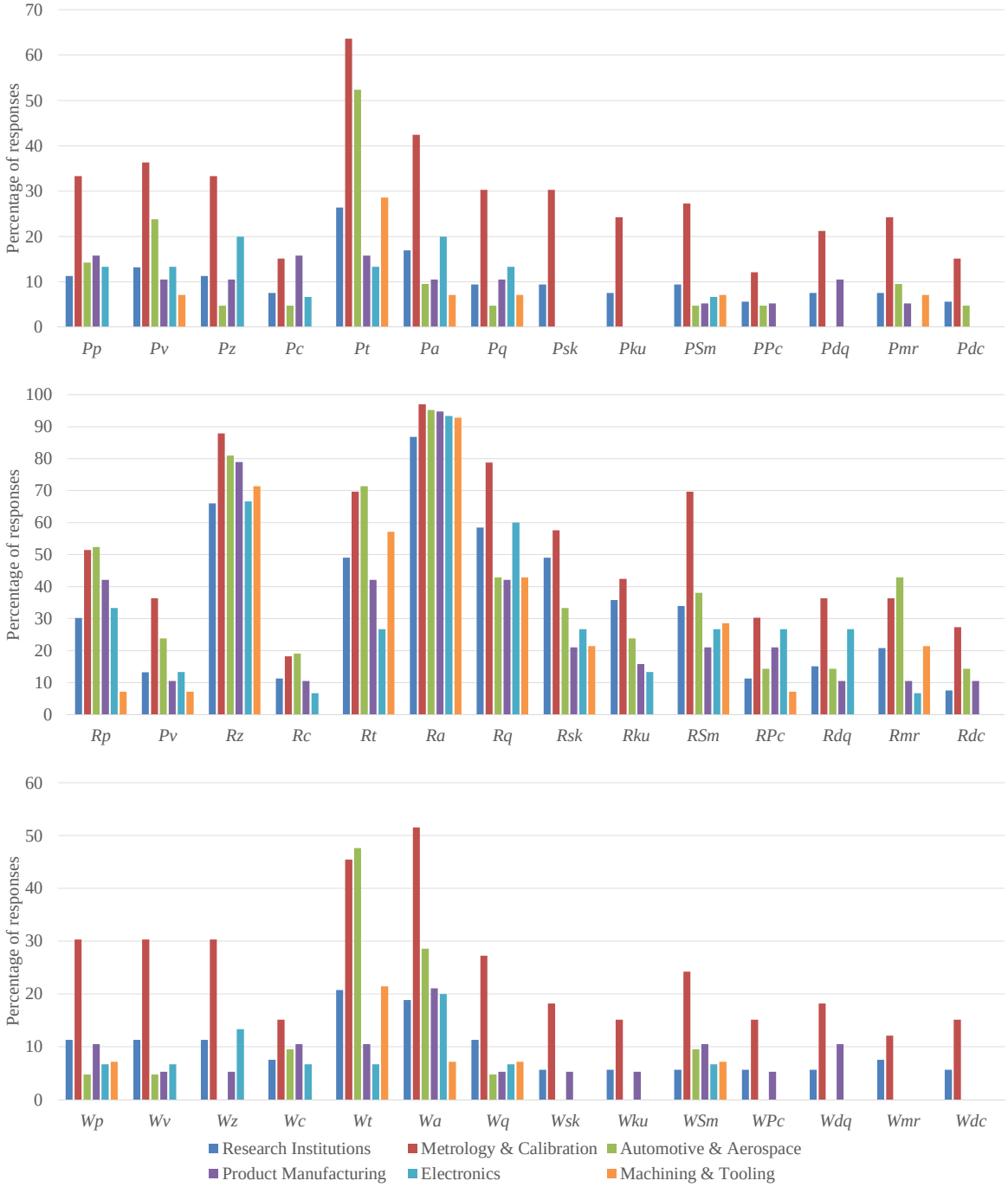


Fig. B.5 Percentage of responses that indicated the use of ISO 4287 parameters, displayed for individual sectors. *Top*: Primary parameters. *Middle*: Roughness parameters. *Bottom*: Waviness parameters.

Figure B.5 shows the 2016 results for the ISO 4287 profile parameters split into individual sectors. Whilst the same overall trends are visible for each sector as for all participants, some results of interest for individual sectors can be seen. The most obvious result is the high uptake of almost all parameters by the 'metrology and calibration' sector. This is to be expected; the 'metrology and calibration' sector has a specific focus on characterising a surface measurement as completely as possible, and so it comes as no surprise that the sector would use a wide variety of parameters. This trend continues throughout the results of this paper.

Ignoring the roughness parameters, another interesting result is that the 'automotive and aerospace' sector shows quite high usage for the *Pt*, *Wt* and *Wa* parameters, but very low usage for the rest. This small variety of parameters used suggests the sector has identified a small selection of parameters that are useful and avoided the so-called 'parameter rash' [41] by only using those that are necessary.

B.2.2.2 ISO 12085 motif parameters and ISO 13565-2/3 stratified surface parameters

Figure B.6 and Figure B.8 give the survey results of the usage of profile surface texture parameters given in ISO 12085 and ISO 13565-2/3 respectively.

As seen for the ISO 4287 parameters, ISO 12085 has seen an increase in use for all parameters. In particular, the *R*, *W* and *Pt* parameters have seen a substantial increase in use, although the number of users is still low compared to the ISO 4287 parameters. Parameters *R* and *W* indicate the mean depths of the roughness and waviness motifs, respectively, and *Pt* is the peak-to-valley height of the roughness motifs. These are all relatively simple to calculate, yet valuable parameter definitions, so it is understandable that these are the most adopted parameters from the standard.

It should be noted here that the 1999 survey only included the first seven parameters shown in figure B.6, as these are the only ones to be included in the ISO 12085 standard

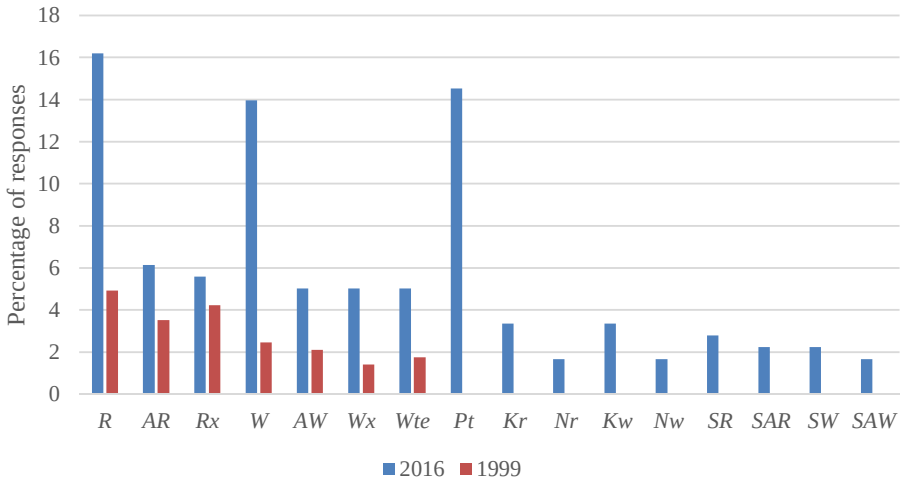


Fig. B.6 Percentage of responses that indicated use of ISO 12085 motif parameters for the 2016 (blue) and 1999 (red) surveys.

document. The 12085 parameters were originally defined in a French automotive standard, but not all parameters were transferred to the ISO document [93]. The rest are still featured in many third-party surface texture parameter calculation software packages, and so were included in this survey.

The individual sector results for the ISO 12085 motif parameters are given in figure B.7. Here, several of the motif parameters are shown to be used by over 22% of the 'electronics' sector, suggesting the 'electronics' sector finds characterisation of surface motifs more useful than other sectors.

The survey results for the ISO 13565 parameters show a significant increase in the use of all parameters. An interesting point is that in 1999, there is a fairly even uptake of the parameters, with each one used by ~10% of participants. This has changed somewhat in 2016, where the *MR1* and *MR2* parameters are used less, and the *Rpk* parameter has become the most popular. The four ISO 13565-3 parameters were not included in the 1999 survey and so are not able to be compared; however, it is clear that these are less popular than the slightly older parameters found in part 2.

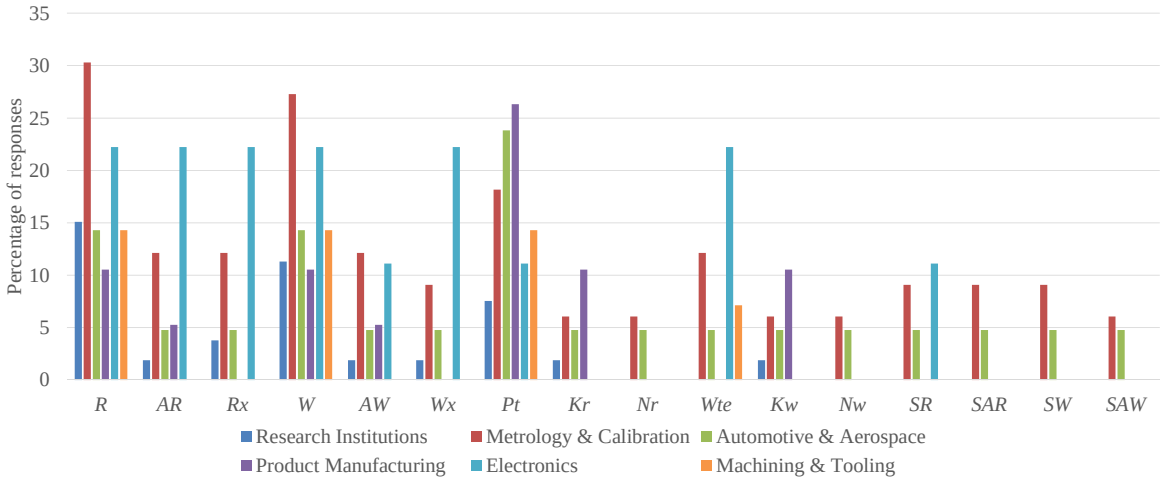


Fig. B.7 Percentage of responses that indicated use of ISO 12085 motif parameters, split into individual sectors

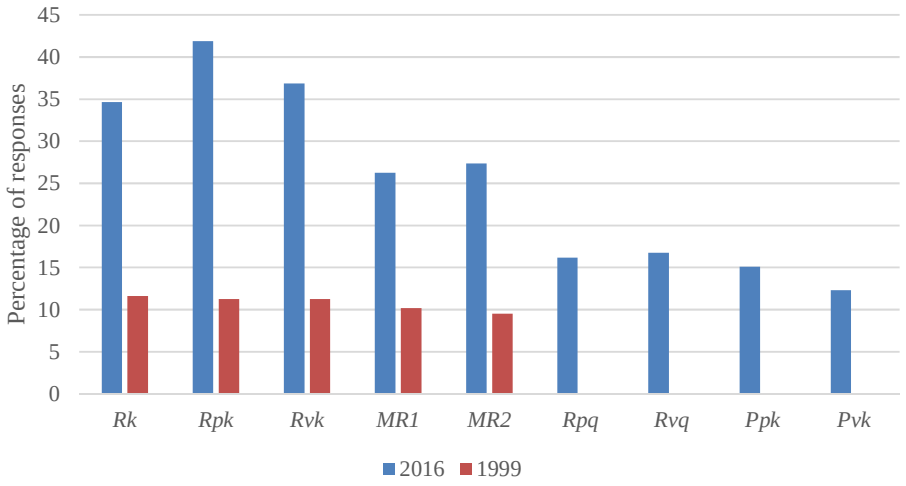


Fig. B.8 Percentage of responses that indicated use of ISO 13565-2/3 stratified surface parameters for the 2016 (blue) and 1999 (red) surveys.

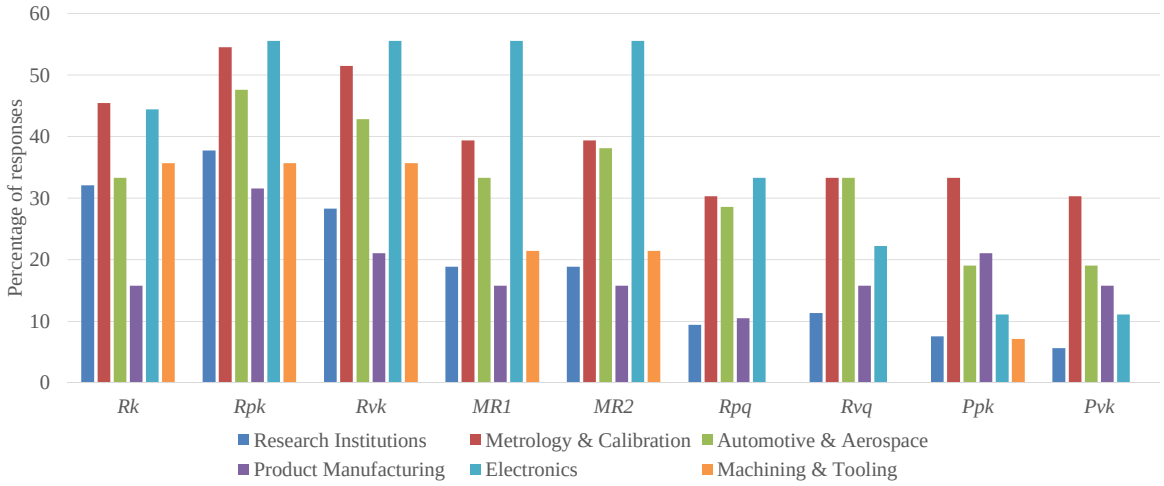


Fig. B.9 Percentage of responses that indicated use of ISO 13565-2/3 stratified surface parameters, split into individual sectors

Figure B.9 shows the individual sector results for the stratified surface parameters of ISO 13565-2/3. As mentioned above, the ISO 13565-2 parameters are the more popular of the two sets, especially with the 'electronics', 'automotive and aerospace' and 'metrology and calibration' sectors. The 'machining and tooling' also displays a notable uptake of these parameters, albeit not as strongly as those aforementioned. These standards focus on characterising stratified surfaces, created through two processes, such as machining and polishing, and so these parameters are of use for sectors that utilise such techniques.

B.2.3 ISO 25178-2 Areal surface texture parameters

Figure B.10 gives the survey results for the field, feature and functional areal surface texture parameters given in ISO 25178-2. These parameters were published in 2012, more recently than many of the other parameters featured in this survey, and as a result, no comparisons can be made with the 1999 survey. Areal surface texture parameters operate in an additional dimension to the profile parameters that precede it. This is a large step forward in the field of surface texture analysis, and thus the results of this survey deliver an interesting insight into the industry's adoption of a new era of surface analysis.

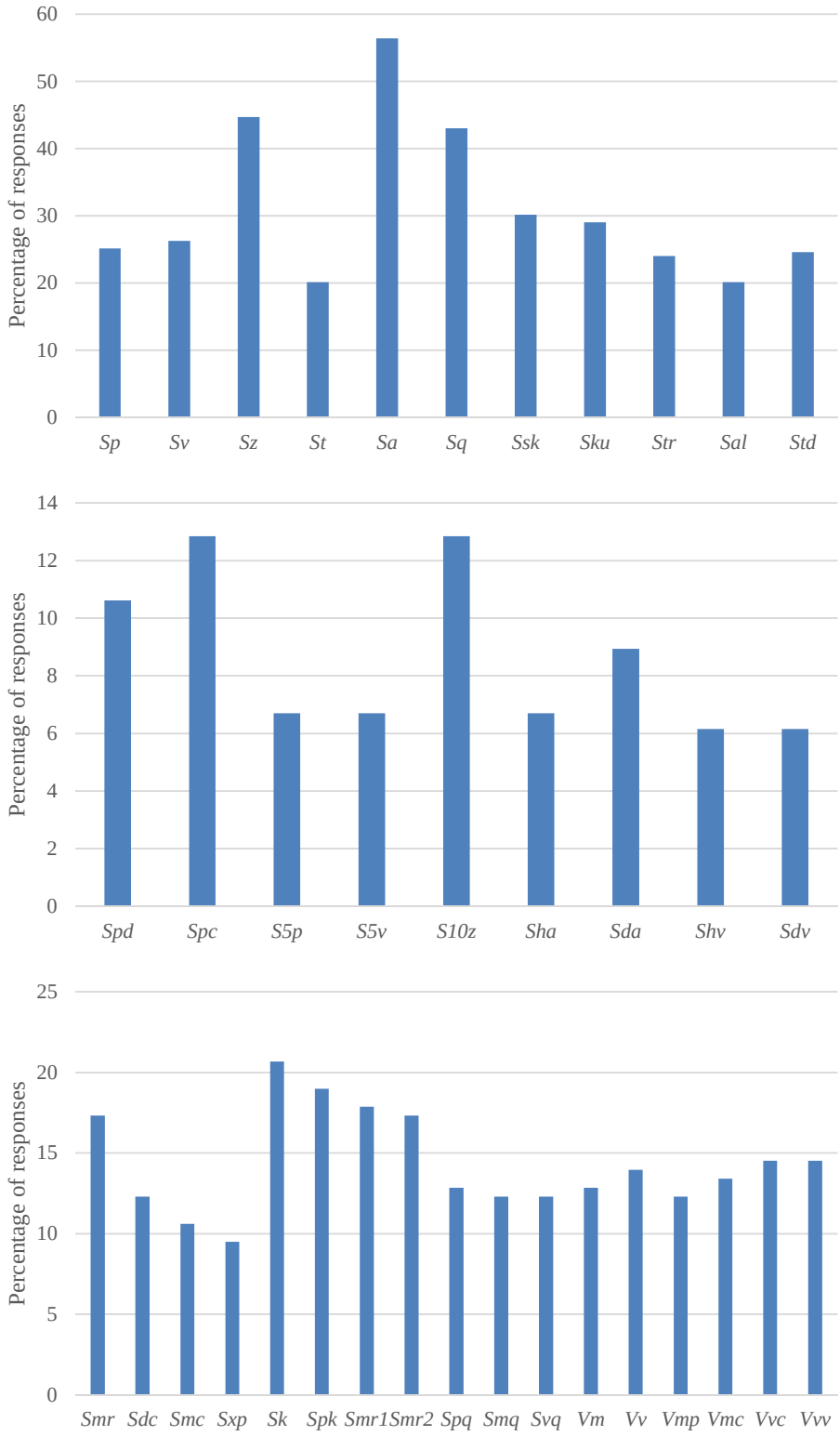


Fig. B.10 Percentage of responses that indicated use of ISO 25178-2 parameters for the field (top), feature (middle) and functional (bottom) parameters. Note that there are no 1999 results to display here as the parameters were not defined at that time.

The ISO 25178-2 parameters are split into three categories: field, feature and functional parameters. The field parameters share the greatest similarity with the ISO 4287 profile parameters, and so it comes as no surprise that these have seen the largest uptake by industry, with an average of 31% of participants claiming use. Unsurprisingly, the areal equivalents of *Ra*, *Rz* and *Rq*, some of the most popular profile parameters, are the most popular areal parameters, with *Sa* being used by 56% of participants. Interestingly, *St* shows one of the lower adoptions at 20%, despite its profile equivalent *R/P/Wt* being one of the most used profile parameters.

The ISO 25178-2 feature parameters focus on areal feature identification, a new type of surface analysis with no real parallel in the profile world. As a result, these are the least used of the three categories of areal surface parameters. That being said, they are still used by a notable amount, ranging from ~6% to 13%, which is more than the majority of motif parameters found in ISO 12085. Similar to the discussion in section B.2.2.1, part of the reason for these usage results could be due to the tendency to simply calculate all available parameters and identify useful parameter correlations afterwards, as computational power and third-party parameter calculation software are now readily available.

The functional parameters revolve around the concept of the material ratio of the surface, and variations thereof. This was introduced in ISO 4287, and so is a surface analysis method familiar to industry. As a result, these parameters have been used by an appreciable percentage of participants, with most ranging from 10% to 20% adoption.

Overall, the ISO 25178-2 areal parameters have been used by a large number of participants considering their age and their differences from the old profile parameters. The inclusion of several analogous parameters to those used for profile surfaces has probably enabled a smoother transition into the areal era and has allowed almost 60% of participants to perform some form of areal surface texture analysis.

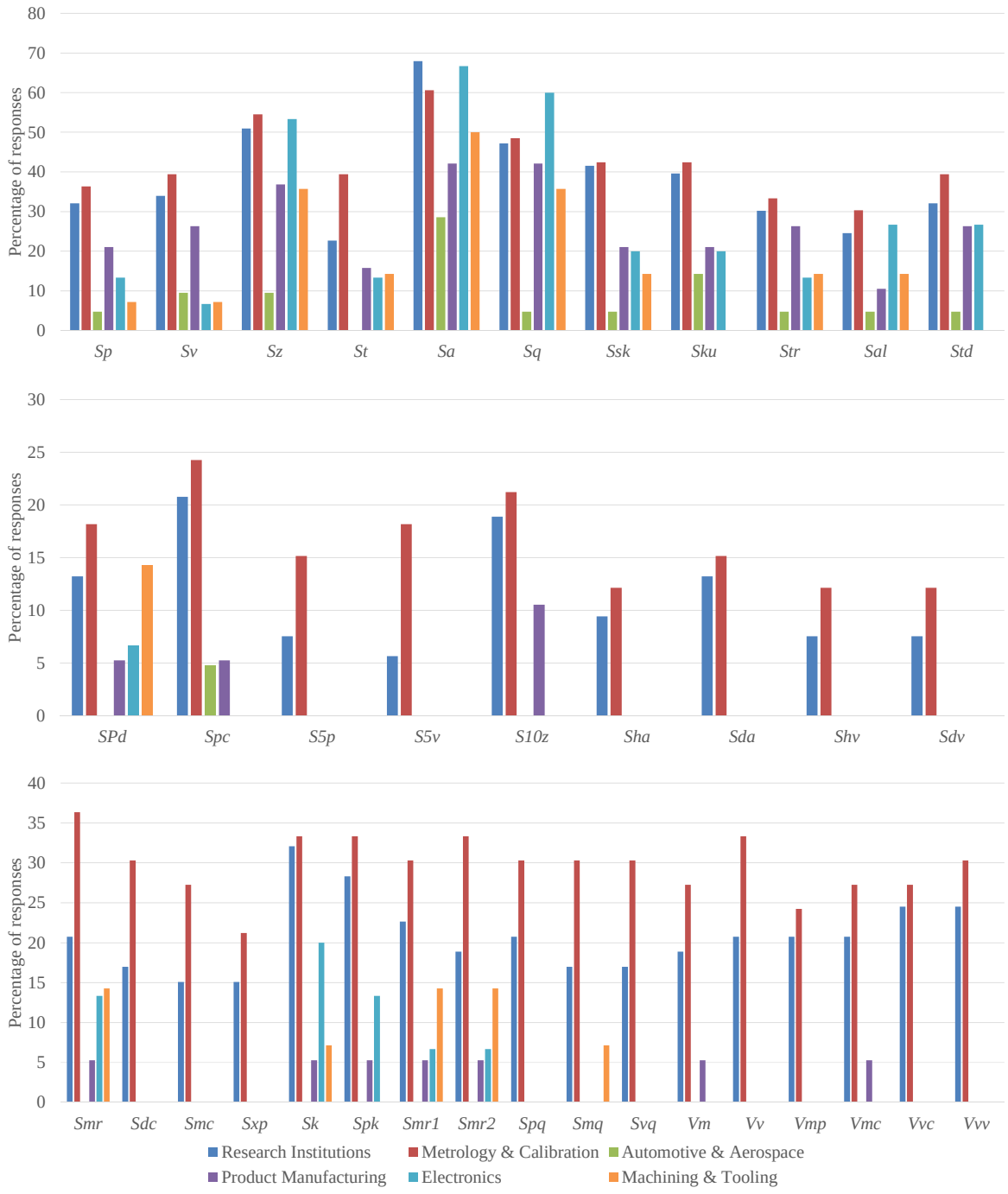


Fig. B.11 Percentage of responses that indicated use of ISO 25178-2 parameters for the field (top), feature (middle) and functional (bottom) parameters, split into individual sectors.

Figure B.11 gives the individual sector results for the ISO 25178-2 parameters. Aside from the previously mentioned widespread use by the 'metrology and calibration' sector, the results also show significant adoption of the areal parameters by the 'research institution' sector. This sector is comprised primarily of universities and other laboratories that conduct scientific research, so it is understandable that this sector would be among the first to adopt the latest parameters and methods available to conduct their research.

The areal field parameters also see adoption by the 'electronics' and 'product manufacturing' sectors. The 'automotive and aerospace' and, to a lesser extent, 'machining and tooling' sectors, however, show little use of these parameters. These is somewhat expected, as these are the two older industrial sectors of the group and rely on more traditional manufacturing and engineering methods. These results suggest it would be useful for these sectors to receive further education on the latest areal surface parameters to fully utilise the new surface information that can be obtained.

For the functional and feature parameters, the individual sector adoption is less impressive. The 'electronics' and 'machining and tooling' sectors show come adoption of a small range of parameters, in particular those which have a profile analogue, such as *Smr*. For the majority of the parameters, however, adoption is poor. Aside from the 'research institution' and 'metrology and calibration' sectors, the majority of participants do not see a use for the functional and feature parameters. Further education and guidance is required to give these sectors an understanding of what these new parameters can offer.

B.2.4 Additional information

In addition to selecting which parameters they used from a predefined list, participants were also encouraged to give their thoughts on the current ISO parameter selection. A total of sixty-nine comments were received detailing the thoughts of participants on the current state

of surface texture parameters. Though these responses were varied, several key points came up.

A common comment given by participants was a lack of in-depth understanding of what each parameter means, and what it tells the user about the surface that is different from any other parameter. Several comments called for further education and training on the current parameter selection, or a more in-depth explanation of the parameter in the ISO specification standard definition, with the hope to aid this understanding and tailor the parameter values calculated to the measurement purpose. These comments suggest that new parameters are published with little further explanation to users regarding their specific uses and differences, which is something that would be of use to parameter users in industry.

Another comment given comes from participants that only use profile surface texture parameters, mentioning that they know of no manufacturers that are using areal parameters. This lack of awareness of the usage of areal parameters suggests there are some fields in industry that are yet to adopt areal parameters and are still limited to older profile parameters. Further education is required on areal surface texture parameters in these fields to promote areal parameter use.

A final popular comment from participants is the need for more areal parameters, particularly for characterising specific features on the surface. Several of these comments make mention of additive manufacturing, and how the resulting surfaces from some of those processes contain features that do not conform well to the current selection of parameters (see Townsend for a recent review on this subject [94]). As areal surface measurements and new manufacturing techniques gain popularity, better education and guidance in good practice are required for surface texture parameters, allowing users to link surface texture parameters with process parameters and function, and better understand the parameter results and the information they give about the measured surface. Of course, in contrast to these comments, several comments were received explaining that there are already too many parameters, and

that these should be reduced down to a small number of key parameters, to avoid the so-called ‘parameter rash’ [41].

Appendix C

International parameter survey form

CIRP Parameter Survey

Page 1: Information

In 1999, CIRP conducted an industrial survey of the use of surface texture parameters. In the 17 years since, much has changed, with the most important advancement being the introduction of areal surface texture parameters as described in ISO 25178-2. There has also been the release of commercial software packages for the calculation of surface texture parameters and, therefore, it is expected that industry is starting to embrace areal surface texture characterisation. Industry is also increasingly using more optical instruments, which are often inherently areal in nature. These factors bring to light the need for a new parameter survey, to investigate whether industry really has been adopting areal surface texture parameters. The results of the survey will be published and presented at a forthcoming CIRP meeting.

Professor Richard Leach, Dr Simon Lawes and Mr Luke Todhunter
Manufacturing Metrology Team
Advanced Manufacturing Research Group
Faculty of Engineering
University of Nottingham

Questions about the survey should be addressed to richard.leach@nottingham.ac.uk.

The survey will take no more than 10 minutes to complete.

All responses will be treated as confidential.

Please take the time to fill in the survey below

If you wish to view the full survey before completing it, feel free to download a .PDF copy of the survey by clicking here.

1. Company ✱ *Required*

2. Address ✱ *Required*

3. Phone ✱ *Required*

4. Email ✱ *Required*

5. Contact Person

6. Company size

- Small (<50 employees)
- Medium (50-250 employees)
- Large (250+ employees)

7. Department

- Development
- Production
- Quality
- Other

8. Type of instrument used

- Contact Stylus
- Optical
- Both
- Other

8.a. If you selected Other, please specify:

9. Instrument mode used

- Profile
- Areal
- Area-integrating (e.g. scattering)
- Multiple modes

Page 2: Parameters used

In the sections below, please mark the parameters used in your company.

10. Parameters defined in ISO 4287: 1997

☐ Pp	☐ Rp	☐ Wp
☐ Pv	☐ Rv	☐ Wv
☐ Pz	☐ Rz	☐ Wz
☐ Pc	☐ Rc	☐ Wc
☐ Pt	☐ Rt	☐ Wt
☐ Pa	☐ Ra	☐ Wa
☐ Pq	☐ Rq	☐ Wq
☐ Psk	☐ Rsk	☐ Wsk
☐ Pku	☐ Rku	☐ Wku
☐ PSm	☐ RSm	☐ WSm
☐ PPc	☐ RPc	☐ WPc
☐ Pdq	☐ Rdq	☐ Wdq
☐ Pmr	☐ Rmr	☐ Wmr
☐ Pdc	☐ Rdc	☐ Wdc

11. Parameters defined in ISO 12085:1996

☐ R	☐ AR	☐ Rx
☐ W	☐ AW	☐ Wx
☐ Pt	☐ Kr	☐ Nr
☐ Wte	☐ Kw	☐ Nw
☐ SR	☐ SAR	☐ SW
☐ SAW		

12. Parameters defined in ISO 16565-2: 1996 and ISO 16565-3: 1998

- ☐ Rk
- ☐ Rpk
- ☐ Rvk
- ☐ MR1
- ☐ MR2
- ☐ Rpq
- ☐ Rvq
- ☐ Ppk
- ☐ Pvk

13. Parameters defined in ISO 25178-2: 2010 - Field parameters

- ☐ Sp
- ☐ Sv
- ☐ Sz
- ☐ St
- ☐ Sa
- ☐ Sq
- ☐ Ssk
- ☐ Sku
- ☐ Str
- ☐ Sal
- ☐ Std

14. Parameters defined in ISO 25178-2: 2010 - Functional parameters

- ☐ Smr
- ☐ Sdc
- ☐ Smc
- ☐ Sxp
- ☐ Sk
- ☐ Spk
- ☐ Smr1
- ☐ Smr2
- ☐ Spq
- ☐ Smq
- ☐ Svq
- ☐ Vm
- ☐ Vv
- ☐ Vmp
- ☐ Vmc
- ☐ Vvc
- ☐ Vvv

15. Parameters defined in ISO 25178-2: 2010 - Feature parameters

- ☐ SPd
- ☐ Spc
- ☐ S5p
- ☐ S5v
- ☐ S10z
- ☐ Sha
- ☐ Sda
- ☐ Shv
- ☐ Sdv

16. Other parameters not defined in ISO standards (please indicate standard):

17. Please share any thoughts/opinions you have on the current range of parameters in use.

Appendix D

A Graphical user interface for the simulation of surface topographies

The work presented in this appendix has been published in reference [95].

D.1 Introduction

The work presented in this appendix explains the details of a graphical user interface (GUI) incorporating the surface creation methods introduced in chapter 5 that enables users to easily control the creation process and design surfaces to a required specification within the controlled bounds of the software. Surfaces are created using a layered approach and are provided to the user as a complete analytical description. The GUI also allows the surfaces to be exported as a dataset in the Surface Data File format [15], suitable for use with a wide variety of third-party surface metrology software applications.

D.2 Graphical user interface

In order to facilitate easy design of surfaces for users, a graphical user interface (GUI) has been developed in MATLAB. This uses a step-by-step process to build surfaces using a layered approach. The GUI features three separate methods for the creation of surfaces:

1. Manual definition of individual cosine terms;
2. Using a visual combination of pre-set shapes;
3. Approximation of an existing .SDF surface dataset.

In addition, the GUI includes the option to introduce pseudo-random 'noise' elements into the analytical definition to create more realistic surfaces.

The start page for the GUI is shown in figure D.1. On the left-hand side of the screen, the user can select one of three methods of surface creation, as briefly described above. These will each be discussed in more detail in later sections. If a manual surface creation method is selected, the upper right quadrant of the screen displays the option to specify the size of the created surface. Here, sliders can be used to specify the size of the surface in x and y dimensions between 1 and 256. The common factor, or scale, of these values is also specified here, with options of 'mm', 'μm' and 'nm' available to tailor the created surface to the desired scale. A multiplicative factor can also be added here (default: 1) to extend the size of the surface beyond the default values permitted by the slider for more advanced use. The two sliders correspond to values N_x and N_y in equation 5.1, and the scaling factor corresponds to f_s.

D.2.1 Manual creation - Cosine terms

Upon selecting 'Manual creation - Cosine terms' on the GUI starting page and pressing 'next', the software presents the screen shown in figure D.2. On this screen, individual cosine terms can be defined which are combined to produce an analytical surface description.

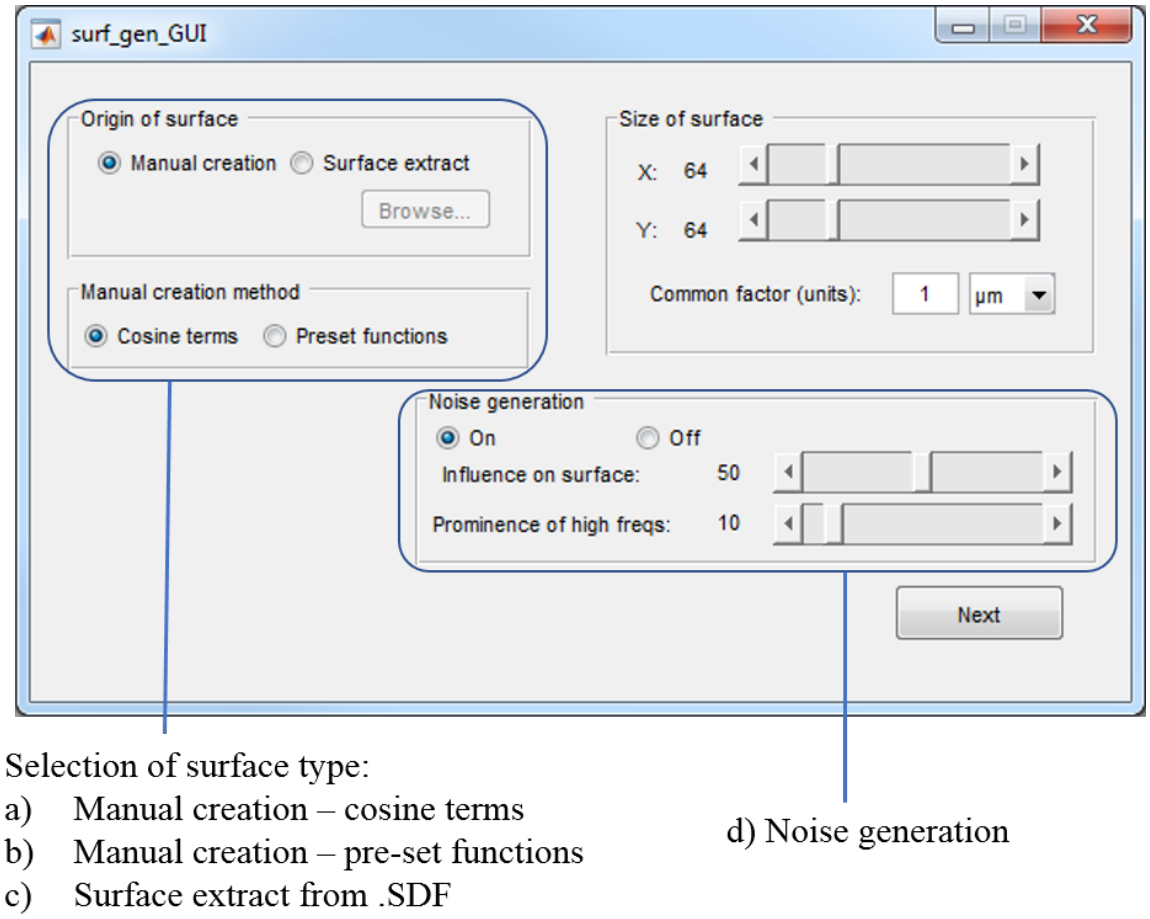


Fig. D.1 Starting page for the surface creation graphical user interface.

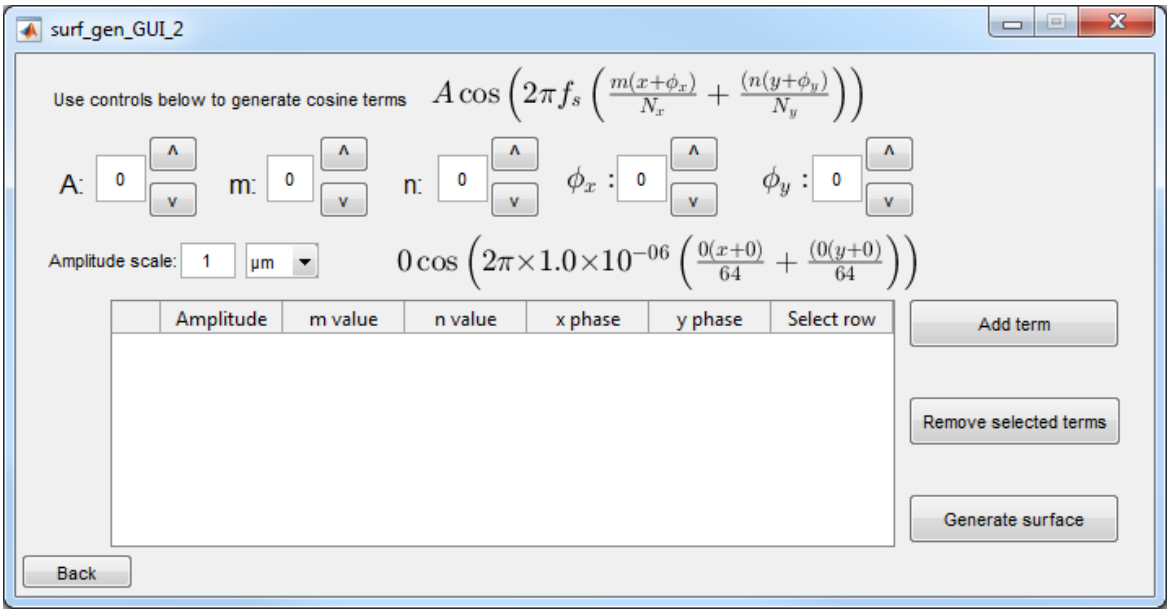


Fig. D.2 GUI page for manual surface creation using a combination of user-defined cosine terms.

At the top of the screen, a template equation of the form

$$A\cos\left(2\pi f_s\left(\frac{m(x+\phi_x)}{N_x}+\frac{n(y+\phi_y)}{N_y}\right)\right) \tag{D.1}$$

is displayed. Beneath the template equation, a series of values for each of the editable variables in the template are shown. Each of these values can be edited by either directly typing into the corresponding text boxes, or by using the adjacent 'up' and 'down' arrows. Once any value is changed, a live version of the template equation, shown below the editable variable boxes, is automatically updated to display the current form of the cosine term. Once the desired variable values are selected, pressing the 'add term' button saves the cosine term and displays the selected variable values in the table in the bottom half of the screen. From here, the variables can be edited again to define the second term, and the process can be repeated to add any number of desired terms. Figure D.3 shows the 'Manual creation - Cosine terms' page filled with a selection of terms. From the table, it is possible to select existing terms and remove them, if required.

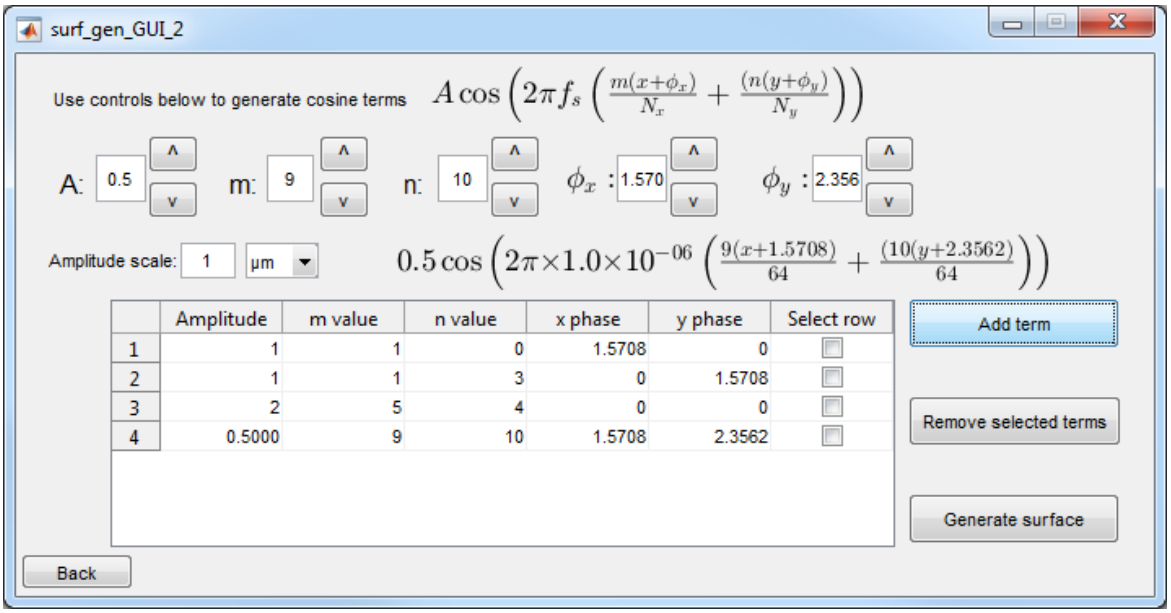


Fig. D.3 GUI page for manual surface creation with four terms saved.

Once all the desired cosine terms are complete, pressing the 'generate surface' button will begin the process of combining all the individual terms into a single analytical equation. The software performs this task by passing all the information stored in the table to a function that forms discrete terms from each row, and then combines the rows together. The back-end for the software uses exponential functions instead of cosine functions as it was found to process faster, and the conversion between the two is performed using Euler's formula,

$$\cos x = \frac{1}{2}\left(e^{ix}+e^{-ix}\right). \tag{D.2}$$

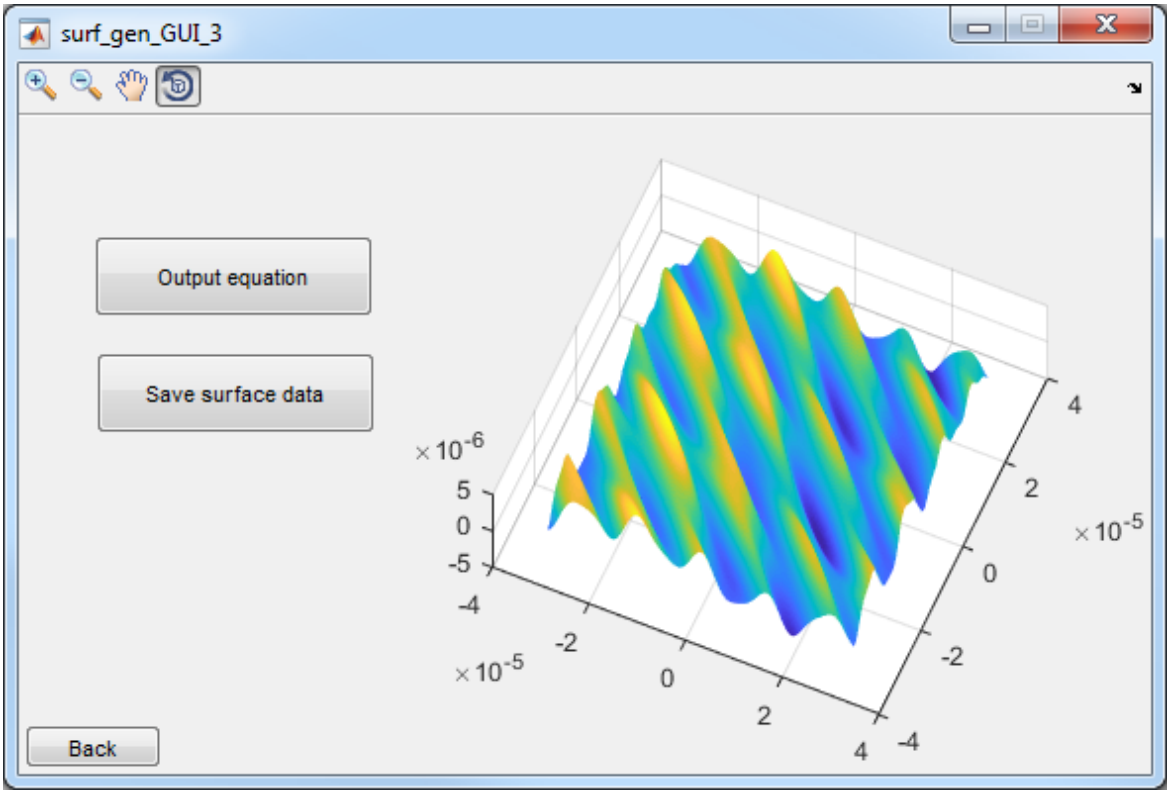
The complete equation is then sampled discretely to generate a low resolution 100×100 pixel preview of the surface suitable for display. This information is passed to a final GUI window, shown in figure D.5, which displays the preview to the user. Here, the x and y ranges are presented zero-centred, and are based on the size of the surface defined on the GUI starting page, as seen in figure D.1. This image can be rotated to give a better view of the surface. If the surface is not as required, the 'back' button can be used to return to

```
Z:\Mathematical reference standard\Surface generator\surface_gen_outputs\4_term_example_surface.txt  06 September 2019 15:27
Surface generator-1.0 -- Luke Todhunter
CreateDate  = 120620191506
Generation method: Cosine terms
*
      Amplitude         m value         n value
              1               1               0
              1               1               3
              2               5               4
   5.000000e-01               9              10
*
(5*^-07)*Exp[(2*Pi*I*1000000)*(1/64*(x+1.570796326794897)+0/64*(y+0))] +
(5*^-07)*Exp[(2*Pi*I*1000000)*(1/64*(x+0)+3/64*(y+1.570796326794897))] +
(1*^-06)*Exp[(2*Pi*I*1000000)*(5/64*(x+0)+4/64*(y+0))] +
(2.5*^-07)*Exp[(2*Pi*I*1000000)*(9/64*(x+1.570796326794897)+10/64*(y+2.35619
4490192345))] +
(5*^-07)*Exp[(2*Pi*I*1000000)*(-1/64*(x+1.570796326794897)+0/64*(y+0))] +
(5*^-07)*Exp[(2*Pi*I*1000000)*(-1/64*(x+0)+-3/64*(y+1.570796326794897))] +
(1*^-06)*Exp[(2*Pi*I*1000000)*(-5/64*(x+0)+-4/64*(y+0))] +
(2.5*^-07)*Exp[(2*Pi*I*1000000)*(-9/64*(x+1.570796326794897)+-10/64*(y+2.356
194490192345))]
```

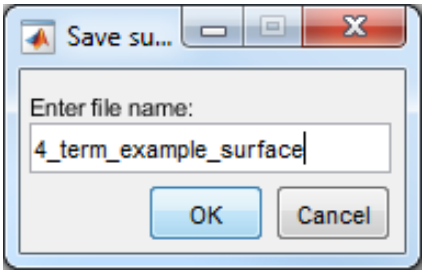
Fig. D.4 .TXT file output for the four-term cosine surface defined in figure D.3.

the previous screen and the cosine terms can be adjusted. From here, the formal expression of the surface equation can be output to a .TXT text file with a filename specified by the user, as shown in figure D.5 (b). For the example surface defined in figure D.3, the .TXT output equation is given in figure D.4. This allows the user to evaluate the analytical surface expression in any way they require, allowing them to generate mathematical reference values against which the corresponding dataset can be compared.

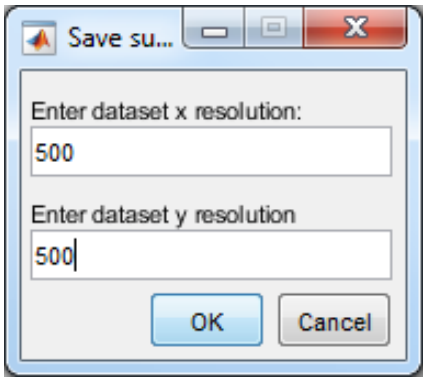
In addition, the software allows a discrete surface dataset of the surface to be exported in the standard ISO 25178-71 Surface Data File format [15], enabling transfer into a variety of third-party surface metrology software applications. Here, the resolution of the dataset can be specified, shown in figure D.5 (c), ensuring the file size and surface data quality are at the correct level for the required application. Both the .TXT equation file and the .SDF surface dataset file include a timestamp within the header of the respective document, so 'traceability' between equation and dataset can be ensured by the user.



(**a**) Output screen.



(**b**) Dataset/equation filename entry.



(**c**) Surface dataset resolution selection.

Fig. D.5 GUI output windows for the surface defined in figure D.3.

D.2.2 Manual creation - Pre-set functions

The second manual creation option is the use of pre-set functions that can be combined visually in a layered approach to design a surface. This method is more accessible to users of the software who do not wish to create a surface from a mathematical definition as it is more intuitive, giving a visual representation of the components of the surface that are being combined.

The main GUI for the creation of a surface using pre-set functions is comprised of two parts: the overview window and the surface preview window, shown in figure D.6. From here, the first function type can be selected, choosing from: a cosine function of the form

$$z(x,y) = A\cos\left(2\pi\left(\frac{f_x\,(x+\phi_x)}{N_x} + \frac{f_y\,(y+\phi_y)}{N_y}\right)\right), \tag{D.3}$$

where f_x and f_y are the x, y spatial frequencies, and the other symbols are as defined for equation 5.1; a Gaussian function of the form

$$z(x,y) = A\exp^{-}\left(\frac{(x-\phi_x)^2}{2\sigma_x} + \frac{(y-\phi_y)^2}{2\sigma_y}\right), \tag{D.4}$$

where σ_x and σ_y are the bell widths, or standard deviations, in the x and y directions, respectively; a box function of the form

$$z(x,y) = \begin{cases} A & \begin{array}{l} \Delta_y - \phi_y < y < \Delta_y - \phi_y \\ -\Delta_x - \phi_x < x < \Delta_x - \phi_x \end{array} \\ 0 & \text{Elsewhere} \end{cases}, \tag{D.5}$$

where $2\Delta_x$ and $2\Delta_y$ are the box widths in the x and y directions, respectively; a triangle function of the form

$$z(x,y) = A\left(1 - \left|\frac{x-\phi_x}{\Delta_x}\right| - \left|\frac{y-\phi_y}{\Delta_y}\right|\right), \tag{D.6}$$

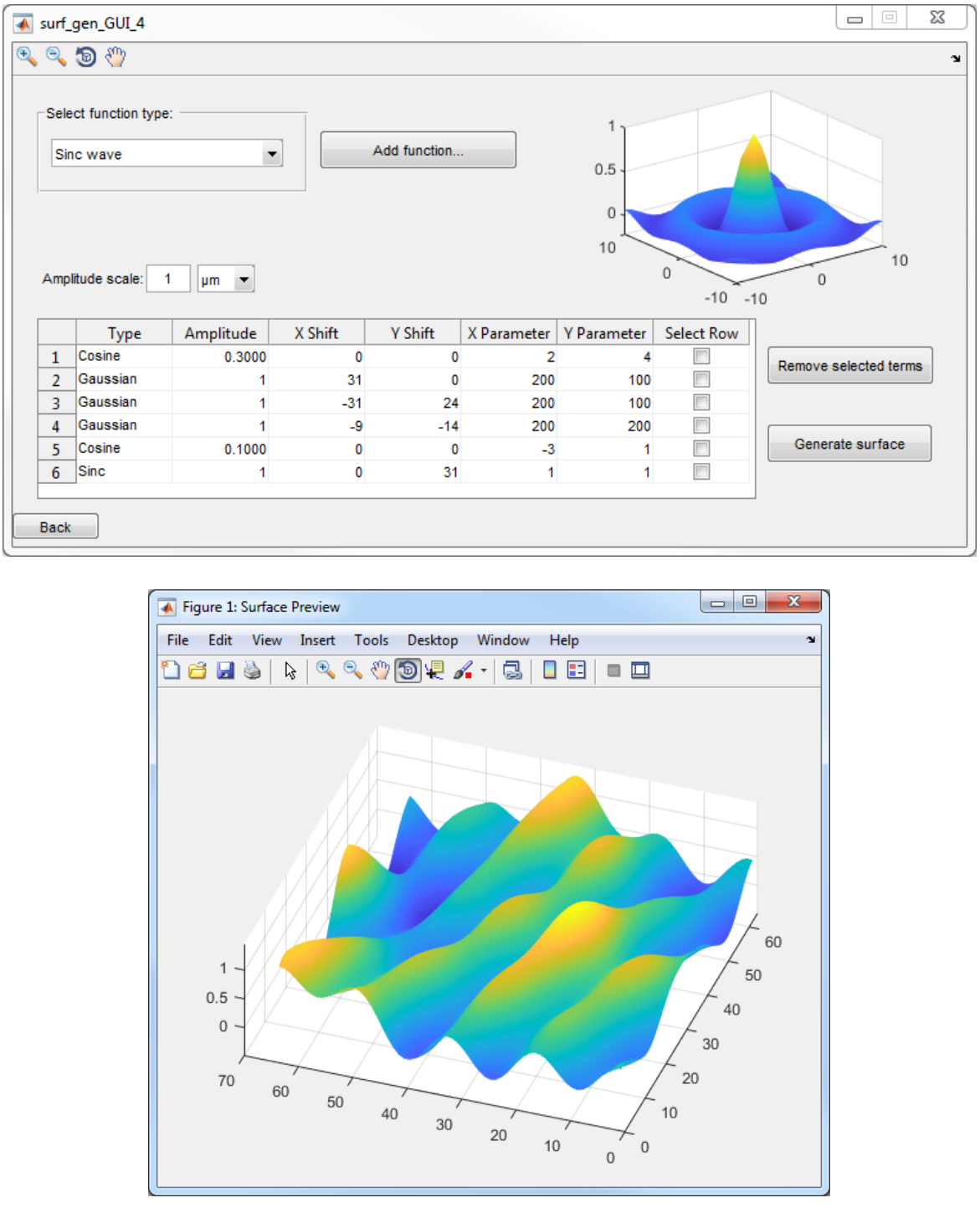


Fig. D.6 Starting GUI windows for the Manual creation - Pre-set functions section. *Top*: Main overview. *Bottom*: Surface preview.

where Δ_x and Δ_y define the gradient of the triangle in the x and y directions, respectively; and a sinc function of the form

$$z(x,y) = A\frac{\sin\left(2\pi\left(\frac{f_x(x+\phi_x)}{N_x}+\frac{f_y(y+\phi_y)}{N_y}\right)\right)}{2\pi\left(\frac{f_x(x+\phi_x)}{N_x}+\frac{f_y(y+\phi_y)}{N_y}\right)}. \tag{D.7}$$

Each of these equations have five adjustable parameters; amplitude, x and y displacement, and two additional parameters, one for each of the x and y directions, that are unique to each function. By combining a range of different function types, users can more intuitively design their required surface. After selecting a function type and pressing 'Add function', a parameter selection window is displayed, shown in figure D.7. Here, the five adjustable parameters for that particular function can be edited in a similar way to the variables in the 'Manual creation - Cosine terms' section. Alongside the parameter values, a live preview of the function is displayed that updates to correspond to the current parameter values selected. Once complete, pressing 'Add function' will save the selected values and function type to the table on the overview window. It will also update the surface preview window to include the newly added function. Pressing 'Cancel' will return to the overview window without saving the current function. Multiple functions can be created in the same way, each designed visually and added to the overview table. With each addition, the surface preview window updates to show the current surface shape, incorporating all saved functions together. As with the 'Cosine terms' method, previously saved functions can be removed from the overview table at any point. Doing so will also update the surface preview, reflecting the change. An example overview window and surface preview for a visually designed function is shown in figure D.6.

Once the design of the surface is complete, pressing the 'Generate surface' button will process the surface using the Fourier series framework. This is achieved by performing a Fourier transform on the designed surface dataset and using the resulting frequency-space

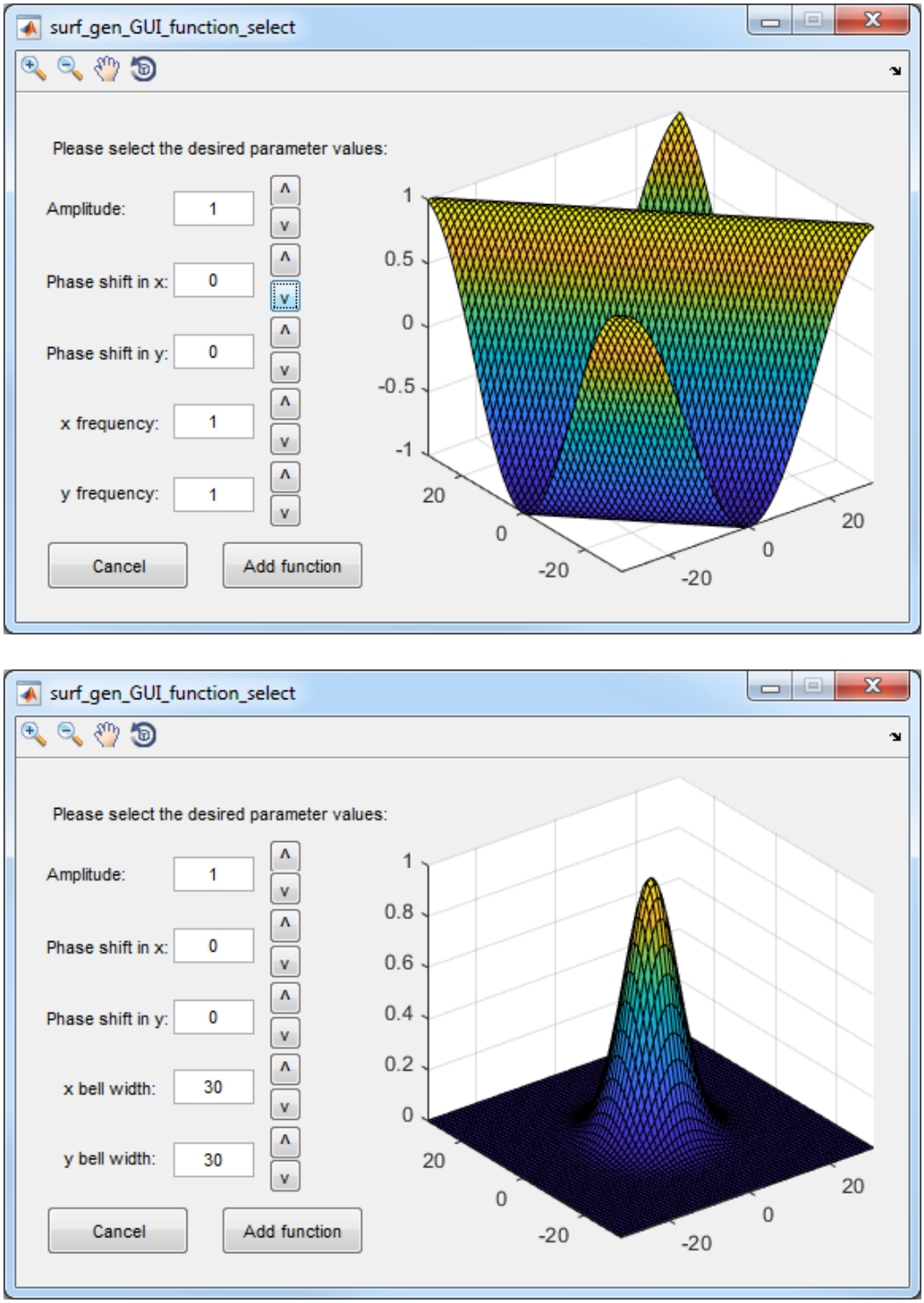


Fig. D.7 Parameter selection GUI window for the Manual creation - Pre-set functions section. *Top*: Cosine function. *Bottom*: Gaussian function.

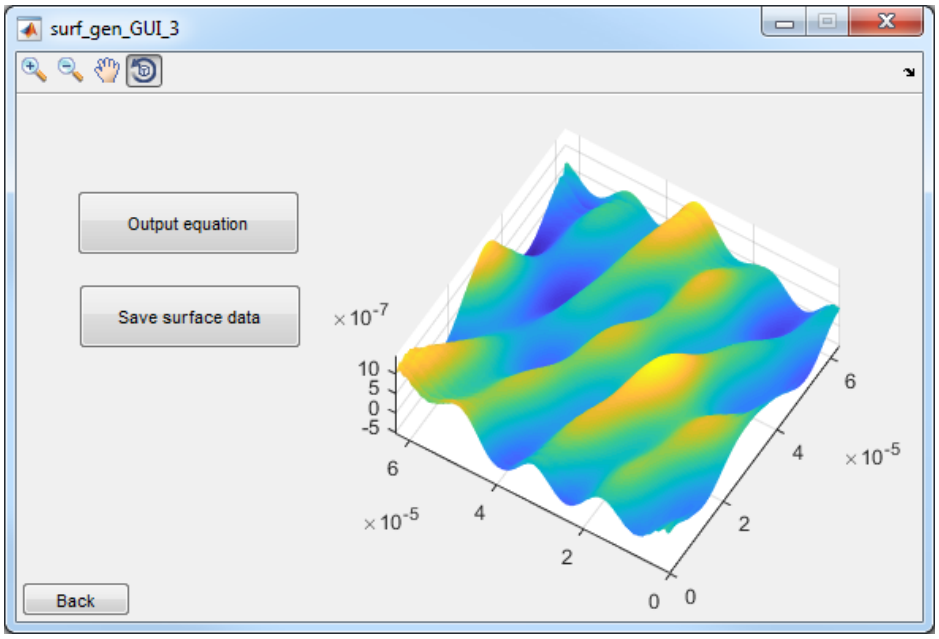


Fig. D.8 Output GUI window for the Manual creation - Pre-set functions section for the surface defined in figure D.6.

information to build a Fourier series that is a close approximation of the surface that has been designed. This information is then passed to the output window, where the surface equation and dataset can be exported, as shown in figure D.8. As this method relies on the layer-by-layer creation of a discrete, visual dataset, the resulting Fourier series output is based on a discrete Fourier transform (fast Fourier transform algorithm), and consequently is a close approximation of the designed function, but not an exact copy. This is due to to effects such as attempting to represent non-periodic functions as a Fourier series, for example box functions, and due to using a Fourier series approximation on a finite length surface. These effects will distort the resulting analytical surface in comparison to the original designed surface, however, as the exportable surface dataset will be directly sampled from the analytical definition and not the designed surface, all subsequent processes will remain valid.

D.2.3 Surface extract from a Surface Data File

In addition to building a surface from scratch, the software allows for the creation of analytical surfaces using a pre-existing .SDF dataset as a starting point. This enables traceable surface definitions to be produced that are based on realistic surfaces relevant to the users' specific interests.

Upon selecting the 'Surface extract' option on the GUI starting page as shown in figure D.1, a 'Browse' button is made available that allows for the selection of any .SDF Surface Data File using a file explorer window. Once a dataset is selected and the 'Next' button is pressed, the software automatically generates an analytical approximation of the chosen surface.

The software reads in the .SDF file and extracts the relevant height and scale information about the surface from the file structure. A discrete Fourier transform is then performed on the data to obtain amplitude and phase information in the frequency domain. Using this information, a Fourier series representation of the surface can be created using the form given in equation 5.1.

Once the process is complete, the output window of the software is displayed that once again enables the exportation of both the surface equation and an .SDF dataset of user specified resolution, equivalent to that shown in figures D.5 and D.8. An example is shown in figure D.9, in which a type S1 reference dataset taken from the NIST online database was input into the software [53], and a subsequent dataset was produced from the resulting analytical surface representation of the same 256×256 resolution as the input dataset. Figure D.9 shows good agreement between the original and recreated datasets, with the recreated dataset having the advantage of being sampled from a traceable, mathematically-defined surface definition. Full assessment of the difference between the original and recreated datasets shows deviations on the scale of 10^{-19}, which for a micrometre scale surface is on the order of rounding errors when performing operations using double precision arithmetic.

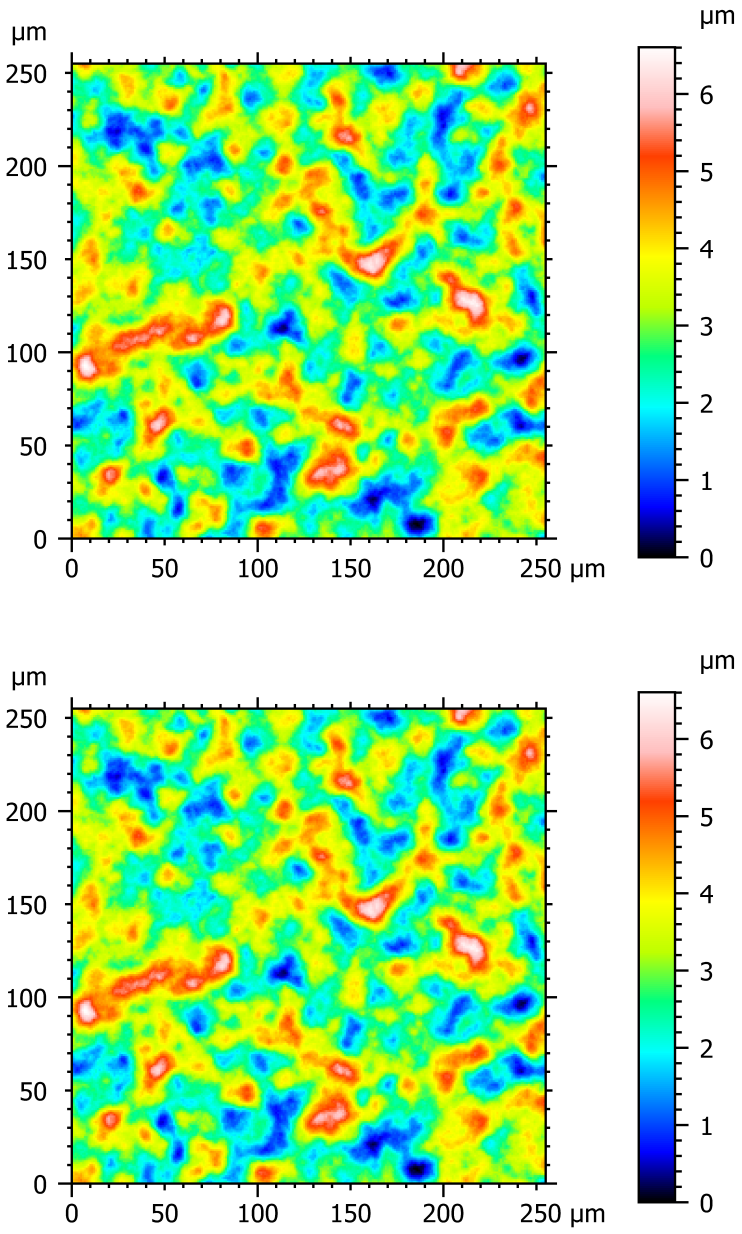


Fig. D.9 *Top*: Type S1 reference dataset from the NIST online database. *Bottom*: Recreated dataset using 'Surface extract' software method. Both datasets have a resolution of 256×256 pixels.

Because the recreated dataset is sampled from an analytical surface representation, it is possible to produce a dataset that is a higher resolution than the original. Figure D.10 shows a cropped section of both the original 256×256 NIST type S1 reference dataset, and a 1536×1536 resolution dataset taken from the analytical representation. The images show an increase in surface clarity and detail, consistent with what would be expected from an increased resolution measurement of the 'real' original surface.

D.2.4 Multiscale texture generation

In addition to defining surface structures and components with low spatial frequencies, it may also be desirable to include a large number of components with higher spatial frequencies in order to better simulate a realistic multiscale surface. As discussed in section 5.2, many real surface measurements are comprised of a wide range of spatial frequency components, with high spatial frequencies often at occurring with smaller amplitudes.

Whilst it is possible to create any desired surface using the manual creation methods introduced in this work, designing a surface with a high level of complexity would be a difficult task requiring a large number of individual terms, each defined by the user. Instead, it may be desirable to define the large-scale components of the surface manually and use an automated system to incorporate seemingly random, small-scale components to produce a final surface with a wider spatial frequency range.

It is important that the multiscale texture generation method chosen maintains compatibility with the existing Fourier series surface generation framework and can be fully incorporated into the resulting analytical surface definitions, maintaining mathematical traceability. Several methods of pseudo-random 'noise' generation exist in the literature that focus on multiscale components, including fractal surface generation methods as discussed in section 5.2 and other stochastic methods such as Weiner interpolation [96], however, these methods cannot be incorporated into a continuous mathematical surface definition.

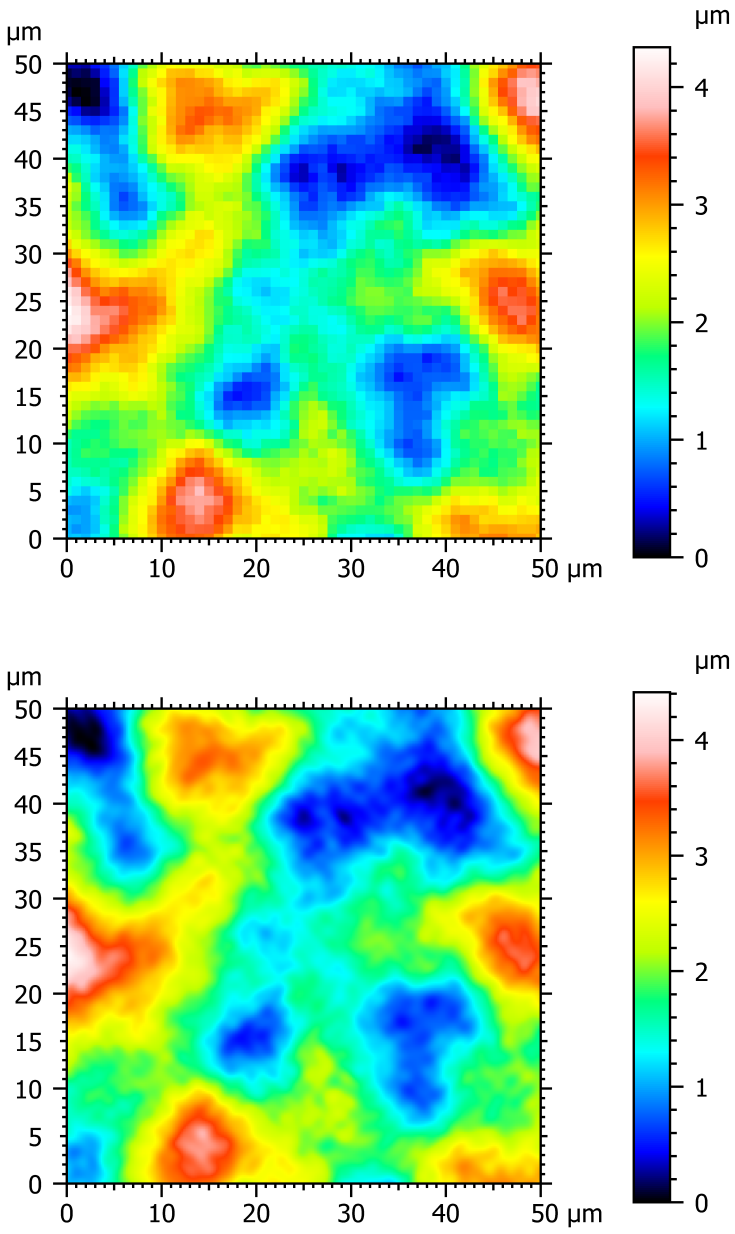


Fig. D.10 *Top*: Cropped section of the 256×256 type S1 reference dataset from the NIST online database. *Bottom*: Cropped section of a 1536×1536 recreated dataset using 'Surface extract' software method.

The software utilises a technique called multi-scale Fourier space Gaussian blur, similar in principle to Perlin noise [97, 98], a technique commonly used for generating textures in the computer graphics industry [99]. Multi-scale Fourier space Gaussian blur texture generation utilises an array of pseudo-random numbers generated using a seed, convolved with a Gaussian function to smooth the higher frequencies. This noise generation is performed on multiple scales; each iteration reduces the bell width of the Gaussian function, therefore sharpening the curve and reducing the effect on high frequencies. Additionally, each iteration reduces in amplitude by a power of two. This process means that the largest amplitude array is the smoothest, and each subsequent array of pseudo-random numbers has higher frequency components, but a smaller amplitude. Finally, each of these surfaces are summed together to create the multi-scale effect. An example of this process in profile form is given in figure D.11. By performing this entire process in Fourier space, the convolution operation is simplified to a multiplication operation, and the Fourier terms can be easily combined with the original surface terms to incorporate the multiscale spatial frequency components into the resulting surface. Because the multiscale texture is produced mathematically, it can be included in the analytical surface definition, maintaining its mathematically-defined, traceable nature.

This method results in enhanced realistic texture generation compared to manual methods with the high-frequency components contributing to small-variations, similar to most real-world measurements. The relative impact of high-frequency components on the total surface can be adjusted using the relative bell-widths of the Gaussian function; increasing the width of the bell increases the prominence of high-frequency components. Figure D.12 shows the texture generation algorithm acting upon a flat input surface, showcasing the noise generation alone. This is compared against real type S1 reference datasets from the NIST online database, similar to that shown in the section D.2.3. Figure D.12 shows very similar visual

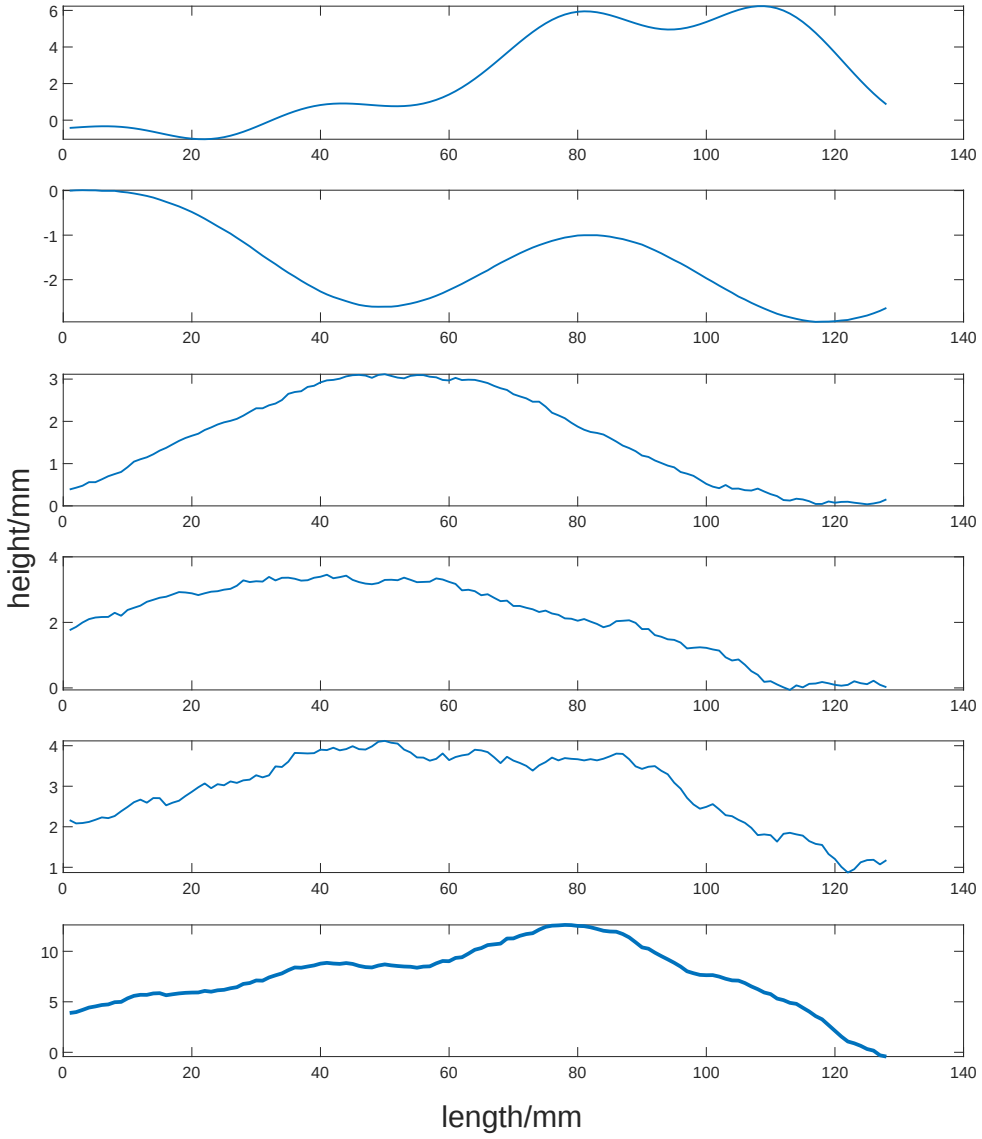


Fig. D.11 A profile example of multi-scale Fourier space Gaussian blur. The top five graphs show decreasing amplitudes with increasing high-frequency components. The bottom graph shows the summation of each multi-scale component.

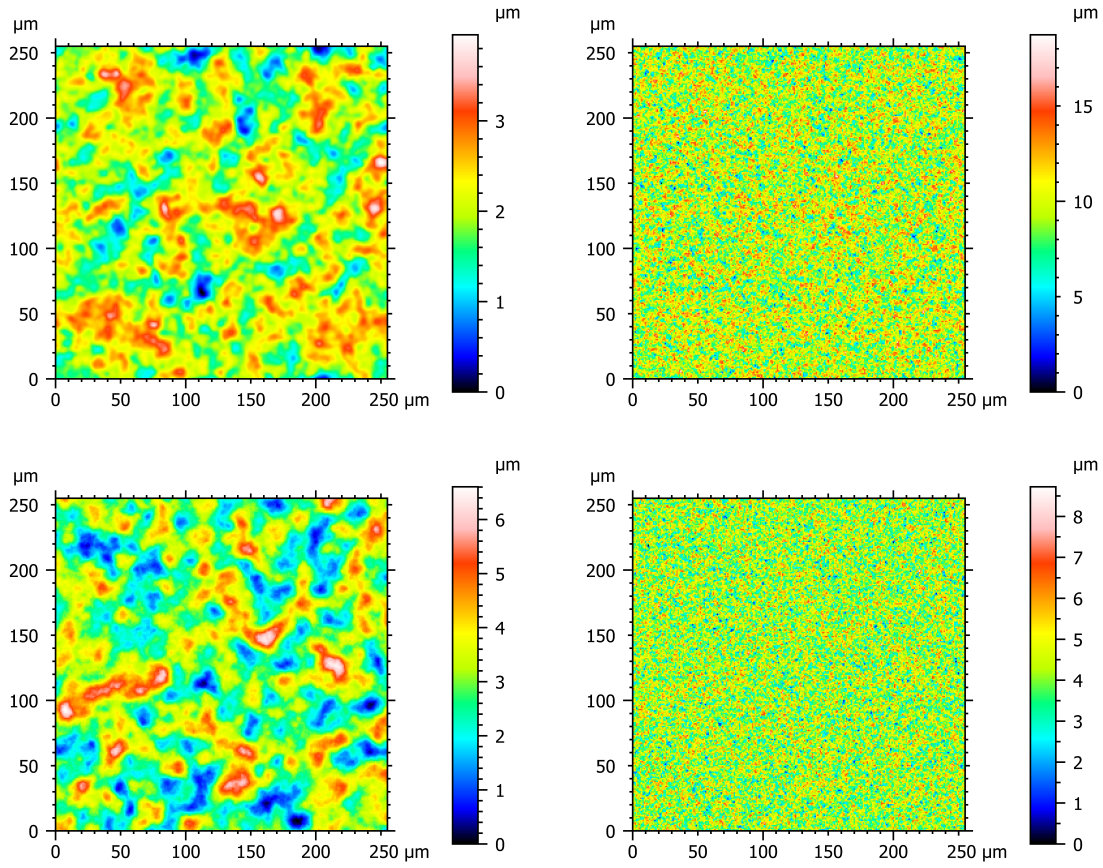


Fig. D.12 Surfaces generated using the texture generation function (*top*) compared against NIST type S1 reference datasets (*bottom*). The different simulated surfaces are created by adjusting the relative bell widths of the Gaussian functions from low (*left*) to high (*right*).

structures between the reference and texture-generated surfaces, suggesting a successful simulation of surface texture.

To enable the use of multi-scale Fourier space Gaussian blur, the ‘Noise generation’ section of the GUI starting page shown in figure D.1 is used. Upon turning the setting on using the appropriate radio button, two slider settings affecting the resulting texture generation are enabled. The first, ‘Influence on surface’, determines the overall amplitude of the created texture, relative to the amplitude of the existing surface, and determines whether the added texture has a large effect on the surface. The second setting, ‘Prominence of high frequencies’, adjusts the relative Gaussian bell widths to make high frequency components

more prominent in the final result, as shown in figure D.12. Both settings can take integer values between 1 and 100 and are adjusted using the associated sliders.

As mentioned previously, the realistic texture can also be incorporated into other surfaces created using the software, enabling user generated surfaces to have realistic high-frequency variations. Figure D.13 gives an example of a surface generated using the 'Manual creation - cosine terms' method that has had the multi-scale Fourier space Gaussian blur applied at various levels. The figure shows a clear distortion of the original surface, gradually increasing the amount of high frequency components in a pseudo-random, realistic manner. Figure D.13 shows the 'Noise generation' option applied to the 'Manual creation - Cosine terms' creation method, however, the option can also be applied to the 'Manual creation - Pre-set functions' and 'Surface extract' methods.

D.3 Significance of work

This work presents a complete software framework for the creation of simulated surfaces with a continuous analytical representation. The software is developed in MATLAB and is presented to the user via a graphical user interface that ensures ease of use and the intuitive creation of surfaces. Users can create surfaces in a variety of ways, including manual layer-by-layer approaches and approximations of existing datasets. In addition, multi-scale Fourier space Gaussian blur can be applied to the created surfaces to incorporate pseudo-random realistic texture onto the created surfaces, whilst still maintaining an analytical representation of the surface.

The purpose of this software is to provide an accessible method of creating mathematically-defined surface functions. Such definitions of surfaces are mathematically traceable, and serve as an appropriate starting point for the development of mathematical reference standards for the calculation of areal surface texture parameters. Utilising mathematical references in this manner encourages the traceable validation of surface metrology software applications,

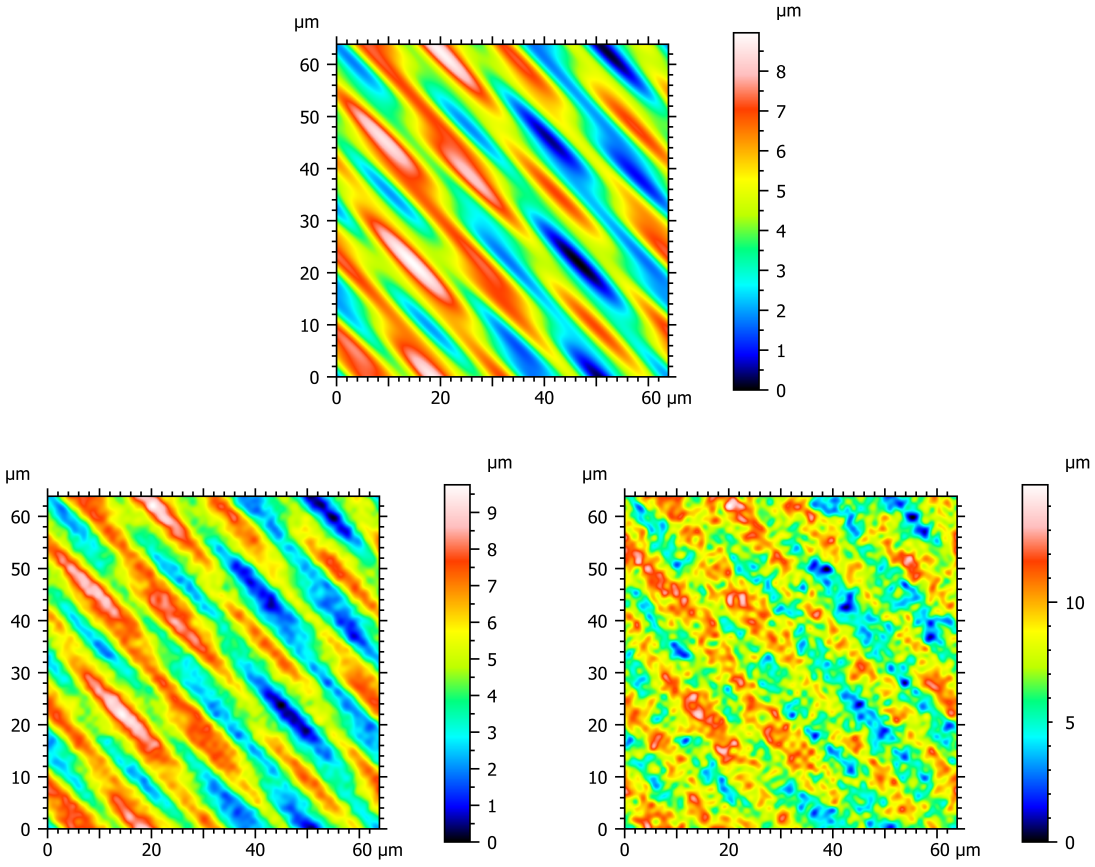


Fig. D.13 *Top*: Surface without noise, created in section D.2.1. *Bottom*: Same surface with added texture using noise generation, using 'Influence on surface' setting values of 20 (*Left*) and 50 (*Right*).

free from the inaccuracies and differing interpretations that are present when relying on the current state of the art of reference software, as shown in chapter 3.

Appendix E

Reference surface certificates

This appendix includes reference surface definitions, and associated parameter values, for the following analytical surfaces:

- Simple predefined surfaces

    1. 2 term cosine
    2. 4^{th} Bernoulli polynomial

    For the simple surfaces, parameter values that are listed with an asterisk (*) were calculated numerically, using both Chebfun and numerical methods included in the CAS, and presented to the number of significant figures to which both methods agree

- Complex predefined surfaces

    1. 10×10 random Chebyshev coefficients
    2. 5^{th} Bernoulli polynomial with added random Chebyshev coefficients

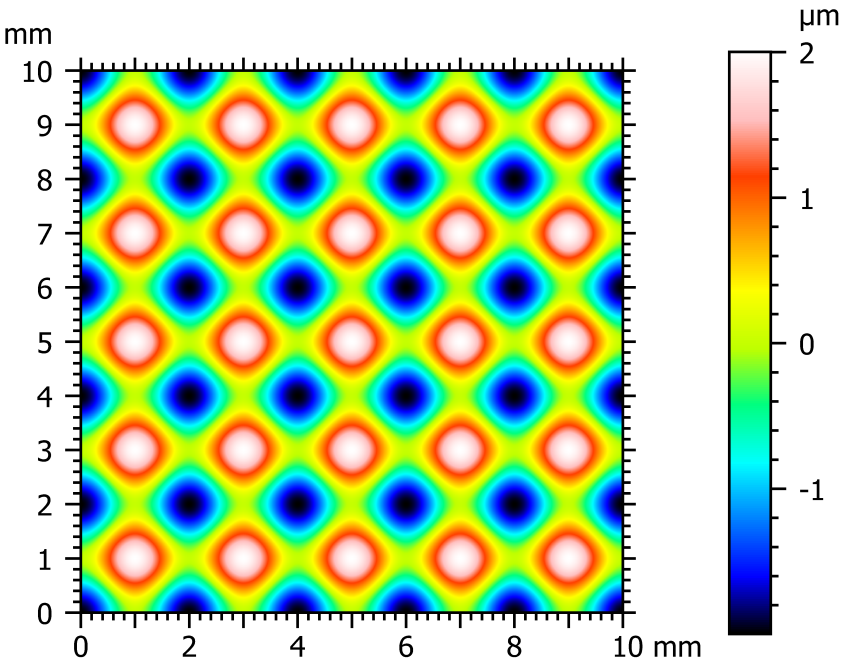
    Each of the parameter values given for the complex surfaces were calculated using Chebfun. For the full equations, see appendix F.

- Parametric simple surfaces

1. 1 term cosine
2. Post-absolute function cosine 1 term cosine

For each of the reference certificates, the use of a '$--$' symbol next to a parameter name indicates the inability to calculate that particular parameter using the methods available.

E.1 Simple predefined surfaces



Surface name 2 term cosine

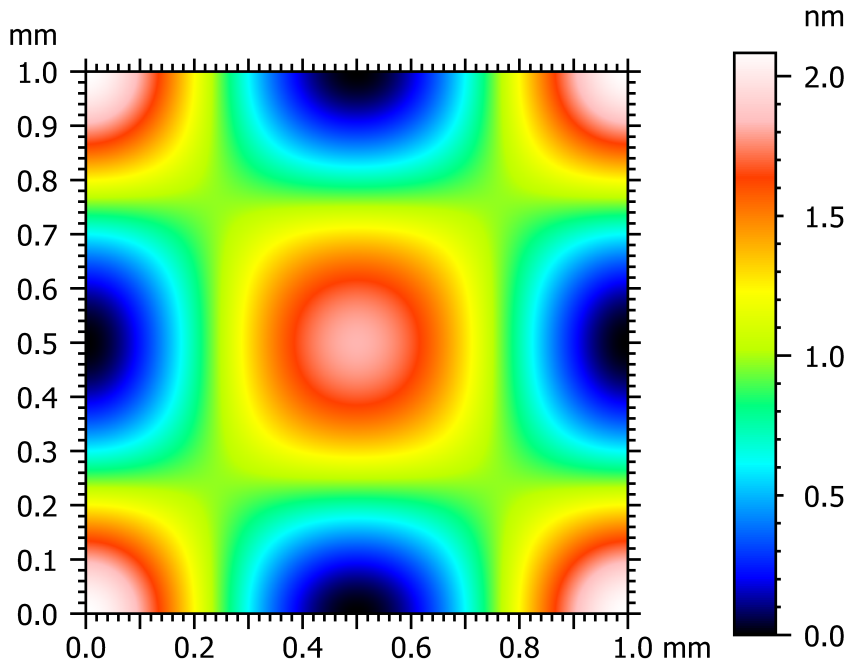
Surface range $-5\text{mm} \leq x, y \leq 5\text{mm}$

Surface equation

$$z(x,y) = 10^{-6}\left(\cos(1000\pi x) + \cos(1000\pi y)\right)$$

Parameter	Value/m
Sq	10^{-6}
Ssk	0
Sku	$9/4$
Sp	2×10^{-6}

Sv	2×10^{-6}
Sz	4×10^{-6}
Sa	$8.10569469138702\times10^{-7}$
Sdq	$3.14159265358979\times10^{-3}$
Sdr*	4.934786980×10^{-6}
Sal	$6.16463870449045\times10^{-4}$
Str	$8.74624309941366\times10^{-1}$
Std	$--$



Surface name 4th Bernoulli polynomial

Surface range $0\text{m} \leq x, y \leq 1\text{m}$

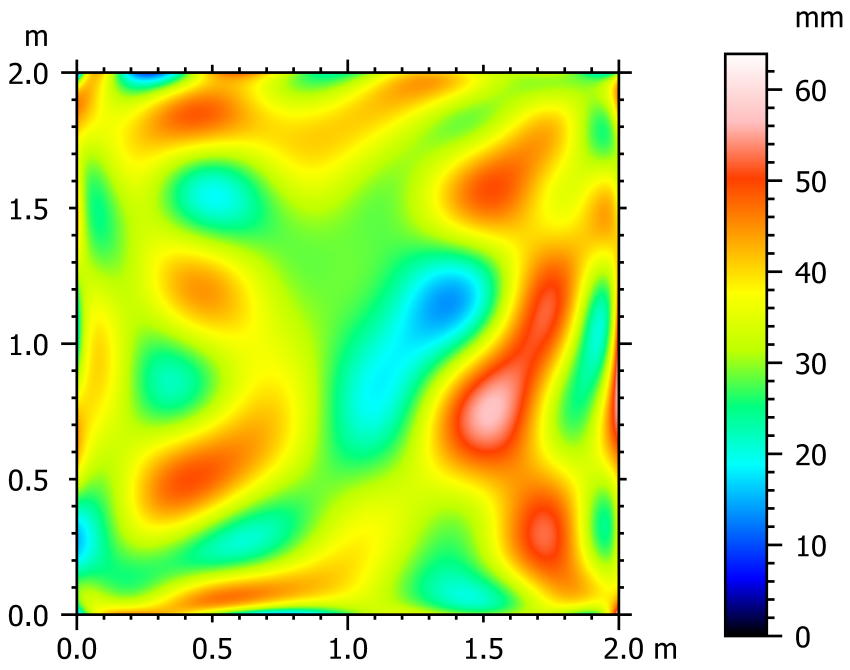
Surface equation

$$z(x,y) = 10^{-6}\left(-\frac{1}{30} + 10^{6}x^{2} - (2\times 10^{9})x^{3} + 10^{12}x^{4}\right)\left(-\frac{1}{30} + 10^{6}y^{2} - (2\times 10^{9})y^{3} + 10^{12}y^{4}\right)$$

Parameter	Value/m
Sq	$4.76190476190476\times 10^{-10}$
Ssk	$1.83075148110113\times 10^{-2}$
Sku	2.36728582500112
Sp	$1.1\dot{1}\times 10^{-9}$

Sv	$9.72\dot{2}\times10^{-10}$
Sz	$2.083\dot{3}\times10^{-9}$
Sa	$3.82849984395878\times10^{-10}$
Sdq	$4.25917709999959\times10^{-6}$
Sdr*	9.07×10^{-12}
Sal	2×10^{-4}
Str	$8.50650808352039\times10^{-1}$
Std	$--$

E.2 Complex predefined surfaces



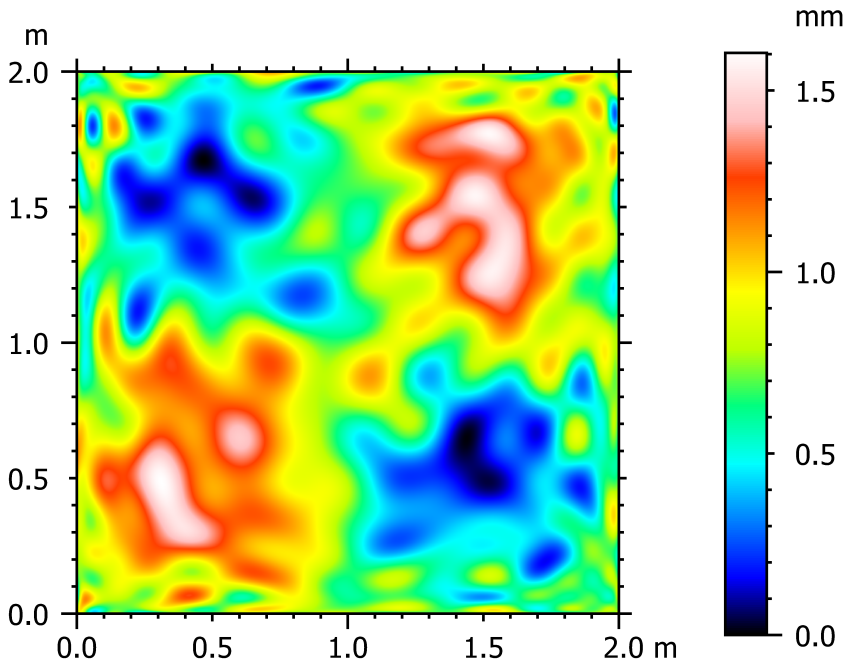
Surface name 10x10 random Chebyshev coefficients

Surface range $-1\text{m} \leq x,y \leq 1\text{m}$

Surface equation see rand_coeffs_10x10_scaled.txt

Parameter	Value/m
Sq	$7.85363058480348\times10^{-3}$
Ssk	$1.24011675529964\times10^{-1}$
Sku	2.73660101546426
Sp	$2.96865644414625\times10^{-2}$
Sv	$3.42239678532728\times10^{-2}$
Sz	$6.39105322947353\times10^{-2}$

Sa	$6.36770443191287\times10^{-3}$
Sdq	$1.13735817913419\times10^{-1}$
Sdr	$6.23665802565518\times10^{-3}$
Sal	−−
Str	−−
Std	−−



Surface name 5th Bernoulli polynomial with random Chebyshev coefficients

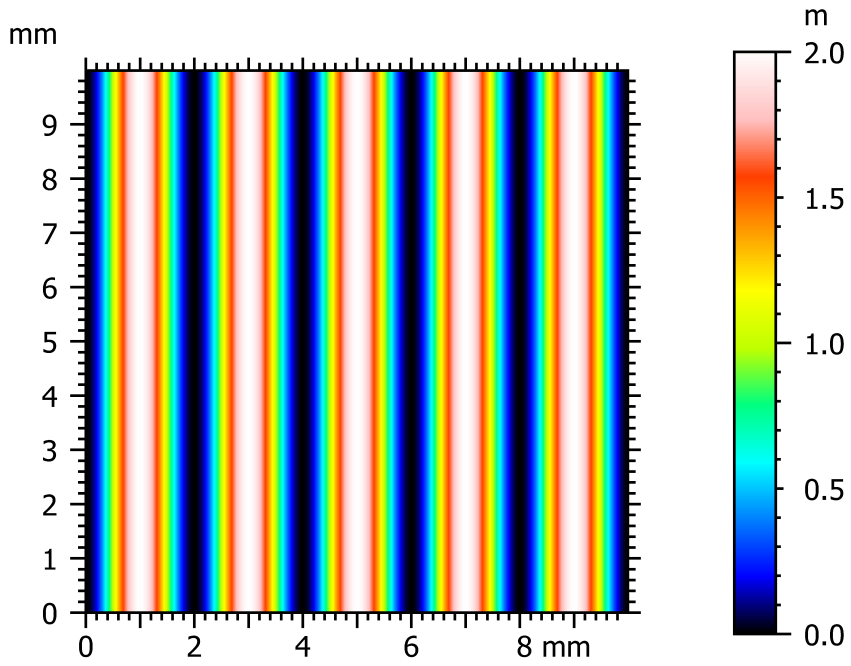
Surface range $-1\text{m} \leq x, y \leq 1\text{m}$

Surface equation see 5th_bernoulli_with_20x20_rand.txt

Parameter	**Value/m**
Sq	$3.36955685549571 \times 10^{-4}$
Ssk	$1.29755708040169 \times 10^{-1}$
Sku	2.39675241612746
Sp	$8.38656952524970 \times 10^{-4}$
Sv	$7.64655812746631 \times 10^{-4}$
Sz	$1.60331276527160 \times 10^{-3}$
Sa	$2.76180350718308 \times 10^{-4}$

Sdq	$5.03160619285385 \times 10^{-3}$
Sdr	$1.26563348904628 \times 10^{-5}$
Sal	– –
Str	– –
Std	– –

E.3 Parametric simple surfaces



Surface name 1 term cosine

Surface range $-5\text{mm} \leq x, y \leq 5\text{mm}$

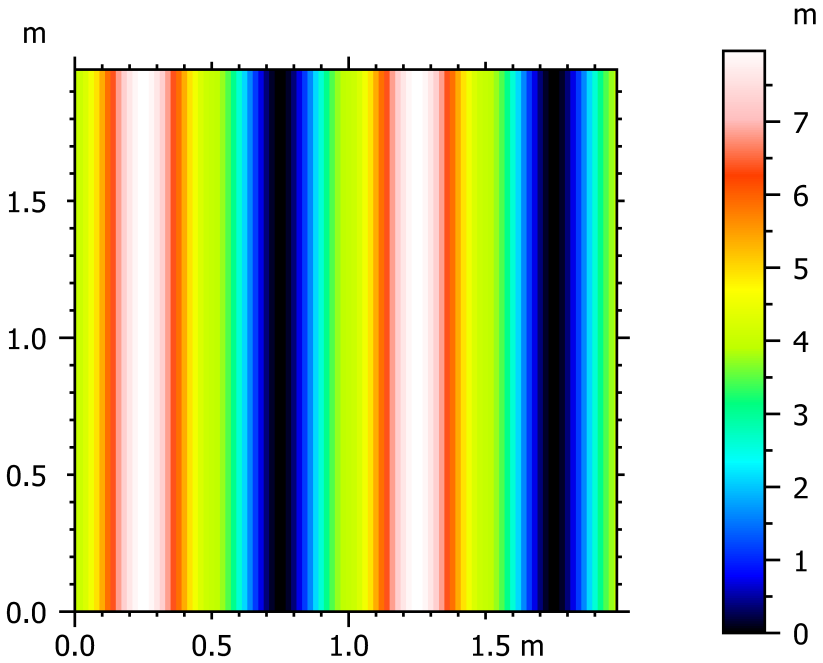
Surface equation

$$z(x,y) = A\cos(f\pi x)$$

Parameter	Value/m
Sq	$\frac{1}{\sqrt{2}\pi}\sqrt{\frac{A^2\left(\pi^2 f^2 + 100\pi f \sin\left(\frac{\pi f}{100}\right) + 40000\cos\left(\frac{\pi f}{100}\right) - 40000\right)}{f^2}}$

Ssk	$200\sqrt{2}A^3 \sin\left(\frac{\pi f}{200}\right)\left(-4\pi^2 f^2 + \left(\pi^2 f^2 - 240000\right)\cos\left(\frac{\pi f}{100}\right) - 900\pi f \sin\left(\frac{\pi f}{100}\right) + 240000\right)$ $\left(3f^3\left(\frac{A^2\left(\pi^2 f^2 + 100\pi f \sin\left(\frac{\pi f}{100}\right) + 40000\cos\left(\frac{\pi f}{100}\right) - 40000\right)}{f^2}\right)^{3/2}\right)^{-1}$
Sku	$\left(9\pi^4 f^4 + 150\pi^3 f^3 \sin\left(\frac{\pi f}{50}\right) + 1200\pi^3 f^3 \sin\left(\frac{\pi f}{100}\right) + 1440000\pi^2 f^2 - 2880000\pi^2 f^2 \sin^2\left(\frac{\pi f}{200}\right) - 320000\pi^2 f^2 \sin\left(\frac{\pi f}{200}\right)\sin\left(\frac{3\pi f}{200}\right) - 480000\left(3\pi^2 f^2 + 40000\right)\cos\left(\frac{\pi f}{100}\right) - 76800000000\sin^2\left(\frac{\pi f}{200}\right) - 72000000\pi f \sin\left(\frac{\pi f}{50}\right) + 144000000\pi f \sin\left(\frac{\pi f}{100}\right) + 4800000000\cos\left(\frac{\pi f}{50}\right) + 38400000000\sin\left(\frac{\pi f}{50}\right)\sin^2\left(\frac{\pi f}{200}\right)\csc\left(\frac{\pi f}{100}\right) + 14400000000\right)\left(6\left(\pi^2 f^2 + 100\pi f \sin\left(\frac{\pi f}{100}\right) + 40000\cos\left(\frac{\pi f}{100}\right) - 40000\right)^2\right)^{-1}$
Sp	$A - \frac{200A\sin\left(\frac{\pi f}{200}\right)}{\pi f}$ if $f > 400$ $f < -400$
Sv	$-A - \frac{200A\sin\left(\frac{\pi f}{200}\right)}{\pi f}$ if $f > 600$ $f < -600$
Sz	$2A$ if $f > 600$ $f < -600$
Sa	$--$
Sdq	$\sqrt{\frac{\pi}{2}}\sqrt{A^2 f\left(\pi f - 100\sin\left(\frac{\pi f}{100}\right)\right)}$
Sdr	$--$
Sal	$--$

Str	– –
Std	– –



Surface name Abs symmetric cosine wave

Surface range $0\text{m} \leq x, y \leq 4n/f_1\text{m}$

Surface equation

$$z(x,y) = \begin{cases} A\left(1-\cos\left(f_1 \pi x\right)\right) & 0 \leq x \bmod \frac{4}{f_1} < \frac{2}{f_1} \\ -A\left(1-\cos\left(f_1 \pi x\right)\right) & \frac{2}{f_1} \leq x \bmod \frac{4}{f_1} < \frac{4}{f_1} \end{cases}$$

Parameter	Value/m
Sq	– –
Ssk	– –
Sku	– –
Sp	– –

Sv	– –
Sz	– –
Sa	*A*
Sdq	– –
Sdr	– –
Sal	– –
Str	– –
Std	– –

Appendix F

Complex predefined surface equations

F.1 5×5 random Chebyshev coefficients

rand_coeffs_5x5_scaled.txt

```
Original function
*
1
*
Polynomial function
*
(5432121580940741.*x.*y)./2305843009213693952 -
(4594850171839545.*y)./1152921504606846976 -
y.*((8392891626566999.*x.^2)./2305843009213693952 -
8392891626566999./4611686018427387904) - ((4735373541752579.*x.^4)./288230376151711744
- (4735373541752579.*x.^2)./288230376151711744 +
4735373541752579./2305843009213693952).*(8.*y.^4 - 8.*y.^2 + 1) -
(429134766147299.*x.*(3.*y - 4.*y.^3))./1152921504606846976 -
((2680928755078007.*x.^2)./576460752303423488 -
2680928755078007./1152921504606846976).*(2.*y.^2 - 1) -
(11697473814333887.*x)./2305843009213693952 + (2.*y.^2 -
1).*((1924462611758271.*x.^4)./288230376151711744 -
(1924462611758271.*x.^2)./288230376151711744 + 1924462611758271./2305843009213693952) +
(214363214985763.*x.*(2.*y.^2 - 1))./576460752303423488 + (3.*y -
4.*y.^3).*((993393843895219.*x.^2)./1152921504606846976 -
993393843895219./2305843009213693952) + (889526272330871.*x.*(8.*y.^4 - 8.*y.^2 +
1))./2305843009213693952 - y.*((16795442009586909.*x)./2305843009213693952 -
(5598480669862303.*x.^3)./576460752303423488) + (2.*y.^2 -
1).*((8753600743565607.*x)./2305843009213693952 -
(2917866914521869.*x.^3)./576460752303423488) -
((5235432134005563.*x.^2)./2305843009213693952 -
5235432134005563./4611686018427387904).*(8.*y.^4 - 8.*y.^2 + 1) -
(127818053875009.*x.^2)./288230376151711744 +
(1093248741219849.*x.^3)./288230376151711744 +
(418700022691889.*x.^4)./288230376151711744 -
(4580700002820561.*y.^2)./1152921504606846976 +
(2347705869474293.*y.^3)./576460752303423488 +
(1107786967691889.*y.^4)./288230376151711744 +
y.*((4617734446500487.*x.^4)./288230376151711744 -
(4617734446500487.*x.^2)./288230376151711744 + 4617734446500487./2305843009213693952) +
((1151545280867019.*x)./288230376151711744 -
(383848426955673.*x.^3)./72057594037927936).*(3.*y - 4.*y.^3) -
((5764371220308369.*x)./2305843009213693952 -
(1921457073436123.*x.^3)./576460752303423488).*(8.*y.^4 - 8.*y.^2 + 1) - (3.*y -
4.*y.^3).*((500192439478327.*x.^4)./288230376151711744 -
(500192439478327.*x.^2)./288230376151711744 + 500192439478327./2305843009213693952) -
1756703322768625./1152921504606846976
```

F.2 10×10 random Chebyshev coefficients

rand_coeffs_10x10_scaled.txt

```
Original function
*
1
*
Polynomial function
*
(7934203717113.*x)./576460752303423488 - (7940954514287171.*y)./1152921504606846976 +
y.*((1592520702236937.*x)./576460752303423488 -
(2654201170394895.*x.^3)./72057594037927936 +
(4777562106710811.*x.^5)./36028797018963968 -
(1592520702236937.*x.^7)./9007199254740992 + (176946744692993.*x.^9)./2251799813685248)
+ (8.*y.^4 - 8.*y.^2 + 1).*((33362436966972453.*x.^2)./1152921504606846976 -
(11120812322324151.*x.^4)./144115188075855872 +
(3706937440774717.*x.^6)./72057594037927936 - 3706937440774717./2305843009213693952) +
(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1).*((19287933505536105.*x)./2305843009213693952 -
(32146555842560175.*x.^3)./288230376151711744 +
(57863800516608315.*x.^5)./144115188075855872 -
(19287933505536105.*x.^7)./36028797018963968 +
(2143103722837345.*x.^9)./9007199254740992) +
((2579937981611725.*x.^2)./2305843009213693952 -
2579937981611725./4611686018427387904).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7) +
(3.*y - 4.*y.^3).*((22915611983240895.*x.^4)./72057594037927936 -
(4583122396648179.*x.^2)./72057594037927936 -
(4583122396648179.*x.^6)./9007199254740992 +
(4583122396648179.*x.^8)./18014398509481984 + 4583122396648179./2305843009213693952) -
y.*((23055389227186395.*x.^4)./72057594037927936 -
(4611077845437279.*x.^2)./72057594037927936 -
(4611077845437279.*x.^6)./9007199254740992 +
(4611077845437279.*x.^8)./18014398509481984 + 4611077845437279./2305843009213693952) +
(5.*y - 20.*y.^3 + 16.*y.^5).*((2607670918787739.*x)./1152921504606846976 -
(2607670918787739.*x.^3)./144115188075855872 +
(2607670918787739.*x.^5)./72057594037927936 -
(372524416969677.*x.^7)./18014398509481984) + (2.*y.^2 -
1).*((48632667046340121.*x)./2305843009213693952 -
(81054445077233535.*x.^3)./288230376151711744 +
(145898001139020363.*x.^5)./144115188075855872 -
(48632667046340121.*x.^7)./36028797018963968 +
(5403629671815569.*x.^9)./9007199254740992) + (1604082575238815.*x.*(160.*y.^4 -
32.*y.^2 - 256.*y.^6 + 128.*y.^8 + 1))./2305843009213693952 +
y.*((15331615416708639.*x)./2305843009213693952 -
(5110538472236213.*x.^3)./576460752303423488) - (3.*y -
4.*y.^3).*((38041228704095361.*x.^2)./1152921504606846976 -
(12680409568031787.*x.^4)./144115188075855872 +
(4226803189343929.*x.^6)./72057594037927936 - 4226803189343929./2305843009213693952) +
(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7).*((31266602127631087.*x)./2305843009213693952
- (31266602127631087.*x.^3)./288230376151711744 +
(31266602127631087.*x.^5)./144115188075855872 -
(4466657446804441.*x.^7)./36028797018963968) -
y.*((6539879703512845.*x)./2305843009213693952 -
(6539879703512845.*x.^3)./576460752303423488 +
(1307975940702569.*x.^5)./144115188075855872) -
((3006441804964091.*x.^4)./576460752303423488 -
(3006441804964091.*x.^2)./576460752303423488 +
3006441804964091./4611686018427387904).*(5.*y - 20.*y.^3 + 16.*y.^5) -
((6847496636424795.*x)./576460752303423488 -
(6847496636424795.*x.^3)./144115188075855872 +
(1369499327284959.*x.^5)./36028797018963968).*(5.*y - 20.*y.^3 + 16.*y.^5) -
((5450405786602319.*x.^2)./2305843009213693952 -
5450405786602319./4611686018427387904).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1) + ((4636420302520511.*x.^2)./1152921504606846976 -
4636420302520511./2305843009213693952).*(8.*y.^4 - 8.*y.^2 + 1) -
((13400876082738945.*x)./1152921504606846976 -
(13400876082738945.*x.^3)./288230376151711744 +
(2680175216547789.*x.^5)./72057594037927936).*(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7
+ 256.*y.^9) - (336812834990991.*x.*(3.*y - 4.*y.^3))./2305843009213693952 + (18.*y.^2
- 48.*y.^4 + 32.*y.^6 - 1).*((42295790965631651.*x)./4611686018427387904 -
```

rand_coeffs_10x10_scaled.txt

```
(42295790965631651.*x.^3)./576460752303423488 +
(42295790965631651.*x.^5)./288230376151711744 -
(6042255852233093.*x.^7)./72057594037927936) + (8.*y.^4 - 8.*y.^2 +
1).*((46618917191594883.*x)./2305843009213693952 -
(77698195319324805.*x.^3)./288230376151711744 +
(13985675157478464 9.*x.^5)./144115188075855872 -
(46618917191594883.*x.^7)./36028797018963968 +
(5179879687954987.*x.^9)./9007199254740992) + (2.*y.^2 -
1).*((15736224327720715.*x)./1152921504606846976 -
(15736224327720715.*x.^3)./144115188075855872 +
(15736224327720715.*x.^5)./72057594037927936 -
(2248032046817245.*x.^7)./18014398509481984) -
((3375063311848749.*x.^2)./576460752303423488 -
(1125021103949583.*x.^4)./72057594037927936 +
(375007034649861.*x.^6)./36028797018963968 -
375007034649861./1152921504606846976).*(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9) + ((6423440051783367.*x)./2305843009213693952 -
(2141146683927789.*x.^3)./576460752303423488).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1) + (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((26868661172457935.*x.^4)./72057594037927936 -
(5373732234491587.*x.^2)./72057594037927936 -
(5373732234491587.*x.^6)./9007199254740992 +
(5373732234491587.*x.^8)./18014398509481984 + 5373732234491587./2305843009213693952) -
((3577791146193407.*x.^4)./288230376151711744 -
(3577791146193407.*x.^2)./288230376151711744 +
3577791146193407./2305843009213693952).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7) -
((4581441443886437.*x.^4)./288230376151711744 -
(4581441443886437.*x.^2)./288230376151711744 +
4581441443886437./2305843009213693952).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1) - (2.*y.^2 - 1).*((497445778490725.*x.^4)./2251799813685248 -
(99489155698145.*x.^2)./2251799813685248 - (99489155698145.*x.^6)./281474976710656 +
(99489155698145.*x.^8)./562949953421312 + 99489155698145./72057594037927936) -
(2365465567888361.*x.*y)./1152921504606846976 +
((17108848958021445.*x)./2305843009213693952 -
(17108848958021445.*x.^3)./576460752303423488 +
(3421769791604289.*x.^5)./144115188075855872).*(8.*y.^4 - 8.*y.^2 + 1) -
((11740348006757169.*x)./2305843009213693952 -
(3913449335585723.*x.^3)./576460752303423488).*(8.*y.^4 - 8.*y.^2 + 1) -
((30041290619473167.*x.^2)./1152921504606846976 -
(10013763539824389.*x.^4)./144115188075855872 +
(3337921179941463.*x.^6)./72057594037927936 -
3337921179941463./2305843009213693952).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1) - (8.*y.^4 - 8.*y.^2 + 1).*((38070917538831177.*x)./2305843009213693952 -
(38070917538831177.*x.^3)./288230376151711744 +
(38070917538831177.*x.^5)./144115188075855872 -
(5438702505547311.*x.^7)./36028797018963968) - (3.*y -
4.*y.^3).*((1275262208425.*x.^4)./1125899906842624 -
(1275262208425.*x.^2)./1125899906842624 + 1275262208425./9007199254740992) + (7.*y -
56.*y.^3 + 112.*y.^5 - 64.*y.^7).*((42803811823828599.*x)./2305843009213693952 -
(71339686373047665.*x.^3)./288230376151711744 +
(128411435471485797.*x.^5)./144115188075855872 -
(42803811823828599.*x.^7)./36028797018963968 +
(4755979091536511.*x.^9)./9007199254740992) +
((9218350239217509.*x)./2305843009213693952 -
(3072783413072503.*x.^3)./576460752303423488).*(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1) +
y.*((3447476315595067.*x)./2305843009213693952 -
(3447476315595067.*x.^3)./288230376151711744 +
(3447476315595067.*x.^5)./144115188075855872 -
(492496616513581.*x.^7)./36028797018963968) -
((5462168421318339.*x.^2)./2305843009213693952 -
5462168421318339./4611686018427387904).*(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9) - (8617093099443.*x.*(2.*y.^2 - 1))./9007199254740992 -
((19332051897496521.*x)./4611686018427387904 -
(6444017299165507.*x.^3)./1152921504606846976).*(7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7) - (8.*y.^4 - 8.*y.^2 + 1).*((862131289220355.*x.^4)./2251799813685248 -
(172426257844071.*x.^2)./2251799813685248 - (172426257844071.*x.^6)./281474976710656 +
(172426257844071.*x.^8)./562949953421312 + 172426257844071./72057594037927936) -
```

rand_coeffs_10x10_scaled.txt

```
1).*((7916185198695745.*x)./1152921504606846976 -
(7916185198695745.*x.^3)./288230376151711744 +
(1583237039739149.*x.^5)./72057594037927936) -
((17135665879806069.*x)./2305843009213693952 -
(5711888626602023.*x.^3)./576460752303423488).*(9.*y - 120.*y.^3 + 432.*y.^5 -
576.*y.^7 + 256.*y.^9) - (2742602330581925.*x.*(7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7))./2305843009213693952 - (2.*y.^2 -
1).*((1149745932721797.*x.^2)./288230376151711744 -
1149745932721797./576460752303423488) +
y.*((4040885745943959.*x.^4)./288230376151711744 -
(4040885745943959.*x.^2)./288230376151711744 + 4040885745943959./2305843009213693952) -
((3311618033688571.*x.^4)./288230376151711744 -
(3311618033688571.*x.^2)./288230376151711744 +
3311618033688571./2305843009213693952).*(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9) + (18.*y.^2 - 48.*y.^4 + 32.*y.^6 -
1).*((8314694360592273.*x.^2)./576460752303423488 -
(2771564786864091.*x.^4)./72057594037927936 +
(923854928954697.*x.^6)./36028797018963968 - 923854928954697./1152921504606846976) -
((58709552609155.*x.^2)./1152921504606846976 -
58709552609155./2305843009213693952).*(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1) +
(5823121746779915.*x.^2)./1152921504606846976 -
(56810192803047731.*x.^3)./576460752303423488 +
(31515521108210949.*x.^4)./288230376151711744 +
(41662309041434667.*x.^5)./72057594037927936 -
(20017771029591009.*x.^6)./72057594037927936 -
(17319328982683559.*x.^7)./18014398509481984 +
(1517982807185853.*x.^8)./9007199254740992 + (4403563265480011.*x.^9)./9007199254740992
+ (128022754750387413.*y.^2)./2305843009213693952 -
(661787716360325.*y.^3)./36028797018963968 -
(20127471677339281.*y.^4)./72057594037927936 +
(14647432484998257.*y.^5)./72057594037927936 +
(33030792990731139.*y.^6)./72057594037927936 -
(11668039514460173.*y.^7)./36028797018963968 -
(8541087396364273.*y.^8)./36028797018963968 +
(1302448489717073.*y.^9)./9007199254740992 + (318644920225287.*x.*(18.*y.^2 - 48.*y.^4
+ 32.*y.^6 - 1))./288230376151711744 - (5.*y - 20.*y.^3 +
16.*y.^5).*((12872005517173833.*x.^2)./288230376151711744 -
(4290668505724611.*x.^4)./36028797018963968 +
(1430222835241537.*x.^6)./18014398509481984 - 1430222835241537./576460752303423488) -
((5701526759776533.*x.^2)./2305843009213693952 -
5701526759776533./4611686018427387904).*(5.*y - 20.*y.^3 + 16.*y.^5) -
((14367543938167059.*x)./4611686018427387904 -
(4789181312722353.*x.^3)./1152921504606846976).*(3.*y - 4.*y.^3) +
((28773465510879995.*x)./2305843009213693952 -
(28773465510879995.*x.^3)./576460752303423488 +
(5754693102175999.*x.^5)./1441151880758558720).*(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1) -
(5357459118289087.*x.*(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9))./2305843009213693952 - ((13734573751036431.*x.^2)./1152921504606846976 -
(4578191250345477.*x.^4)./1441151880758558720 +
(1526063750115159.*x.^6)./72057594037927936 -
1526063750115159./2305843009213693952).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7) -
(2.*y.^2 - 1).*((20270450275686945.*x.^2)./576460752303423488 -
(6756816758562315.*x.^4)./72057594037927936 +
(2252272252854105.*x.^6)./36028797018963968 - 2252272252854105./1152921504606846976) +
(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9).*((1202274140839405.*x.^4)./36028797018963968 -
(240454828167881.*x.^2)./36028797018963968 - (240454828167881.*x.^6)./4503599627370496
+ (240454828167881.*x.^8)./9007199254740992 + 240454828167881./1152921504606846976) +
(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1).*((7196590254406735.*x.^4)./36028797018963968 -
(1439318050881347.*x.^2)./36028797018963968 -
(1439318050881347.*x.^6)./4503599627370496 + (1439318050881347.*x.^8)./9007199254740992
+ 1439318050881347./1152921504606846976) + ((7719527697170545.*x)./1152921504606846976
- (7719527697170545.*x.^3)./288230376151711744 +
(1543905539434109.*x.^5)./72057594037927936).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1) + (2.*y.^2 - 1).*((3964952359204465.*x.^4)./288230376151711744 -
(3964952359204465.*x.^2)./288230376151711744 + 3964952359204465./2305843009213693952) +
```

F.3 20×20 random Chebyshev coefficients

rand_coeffs_20x20_scaled.txt

```
Original function
*
1
*
Polynomial function
*
(111511480662973677.*y)./4611686018427387904 -
(156851986786776857.*x)./4611686018427387904 -
((33082047765271899.*x)./2305843009213693952 -
(55136746275453165.*x.^3)./288230376151711744 +
(99246143295815697.*x.^5)./144115188075855872 -
(33082047765271899.*x.^7)./36028797018963968 +
(3675783085030211.*x.^9)./9007199254740992).*(2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 + 1) -
(8.*y.^4 - 8.*y.^2 + 1).*((473550115309659465.*x.^4)./576460752303423488 -
(40590009883685097.*x.^2)./576460752303423488 -
(31570007687310631.*x.^6)./9007199254740992 +
(121770029651055291.*x.^8)./18014398509481984 -
(13530003294561699.*x.^10)./2251799813685248 +
(4510001098187233.*x.^12)./2251799813685248 + 4510001098187233./4611686018427387904) +
((13592535177151329.*x)./2305843009213693952 -
(13592535177151329.*x.^3)./288230376151711744 +
(13592535177151329.*x.^5)./144115188075855872 -
(1941790739593047.*x.^7)./36028797018963968).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1) - (5.*y - 20.*y.^3 +
16.*y.^5).*((434701677196474065.*x.^4)./288230376151711744 -
(37260143759697777.*x.^2)./288230376151711744 -
(28980111813098271.*x.^6)./4503599627370496 +
(111780431279093331.*x.^8)./9007199254740992 -
(12420047919899259.*x.^10)./1125899906842624 +
(4140015973299753.*x.^12)./1125899906842624 + 4140015973299753./2305843009213693952) -
((24832676862152425.*x)./4611686018427387904 -
(24832676862152425.*x.^3)./1152921504606846976 +
(4966535372430485.*x.^5)./288230376151711744).*(9.*y - 120.*y.^3 + 432.*y.^5 -
576.*y.^7 + 256.*y.^9) + ((15648090288245685.*x.^4)./72057594037927936 -
(3129618057649137.*x.^2)./72057594037927936 -
(3129618057649137.*x.^6)./9007199254740992 +
(3129618057649137.*x.^8)./18014398509481984 +
3129618057649137./2305843009213693952).*(13.*y - 364.*y.^3 + 2912.*y.^5 - 9984.*y.^7 +
16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) + (11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 - 1024.*y.^11).*((84664290140115435.*x)./2305843009213693952 -
(197550010326936015.*x.^3)./144115188075855872 +
(1066770055765454481.*x.^5)./72057594037927936 -
(1269964352101731525.*x.^7)./18014398509481984 +
(1552178652568782975.*x.^9)./9007199254740992 -
(253992870420346305.*x.^11)./1125899906842624 +
(84664290140115435.*x.^13)./562949953421312 -
(5644286009341029.*x.^15)./140737488355328) +
((10397190197427003.*x.^2)./1152921504606846976 -
(3465730065809001.*x.^4)./144115188075855872 +
(1155243355269667.*x.^6)./72057594037927936 -
1155243355269667./2305843009213693952).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7) +
((4966075061568049.*x)./2305843009213693952 -
(24830375307840245.*x.^3)./576460752303423488 +
(34762525430976343.*x.^5)./144115188075855872 -
(4966075061568049.*x.^7)./9007199254740992 + (4966075061568049.*x.^9)./9007199254740992
- (451461369233459.*x.^11)./2251799813685248).*(13.*y - 364.*y.^3 + 2912.*y.^5 -
9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) + (3.*y -
4.*y.^3).*((14034305320817845.*x)./2305843009213693952 -
(14034305320817845.*x.^3)./288230376151711744 +
(14034305320817845.*x.^5)./144115188075855872 -
(2004900760116835.*x.^7)./36028797018963968) + (2441279013657881.*x.*(19.*y -
1140.*y.^3 + 20064.*y.^5 - 160512.*y.^7 + 695552.*y.^9 - 1770496.*y.^11 +
2723840.*y.^13 - 2490368.*y.^15 + 1245184.*y.^17 - 262144.*y.^19))./1152921504606846976
+ ((4120631303640633.*x)./2305843009213693952 -
(6867718839401055.*x.^3)./288230376151711744 +
(12361893910921899.*x.^5)./144115188075855872 -
```

rand_coeffs_20x20_scaled.txt

```
(4120631303640633.*x.^7)./36028797018963968 +
(457847922626737.*x.^9)./9007199254740992).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) - ((4089514713079797 9.*x.^2)./1152921504606846976 -
(13631715710265993.*x.^4)./144115188075855872 +
(4543905236755331.*x.^6)./72057594037927936 -
4543905236755331./2305843009213693952).*(11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7 +
2816.*y.^9 - 1024.*y.^11) - ((34503304462479375.*x)./4611686018427387904 -
(34503304462479375.*x.^3)./1152921504606846976 +
(6900660892495875.*x.^5)./288230376151711744).*(13.*y - 364.*y.^3 + 2912.*y.^5 -
9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) - (160.*y.^4 - 32.*y.^2 -
256.*y.^6 + 128.*y.^8 + 1).*((336315002882861925.*x.^2)./2305843009213693952 -
(560525004804769875.*x.^4)./144115188075855872 +
(287736169133115 2025.*x.^6)./72057594037927936 -
(3699465031711481175.*x.^8)./18014398509481984 +
(5343671712472139475.*x.^10)./9007199254740992 -
(1133506120827423525.*x.^12)./1125899906842624 +
(560525004804769875.*x.^14)./562949953421312 -
(37368333653651325.*x.^16)./70368744177664 + (4152037072627925.*x.^18)./35184372088832
- 4152037072627925./4611686018427387904) + ((10523386678287873.*x)./2305843009213693952
- (73663706748015111.*x.^3)./576460752303423488 +
(73663706748015111.*x.^5)./72057594037927936 -
(31570160034863619.*x.^7)./9007199254740992 +
(52616933391439365.*x.^9)./9007199254740992 -
(10523386678287873.*x.^11)./2251799813685248 +
(809491282945221.*x.^13)./562949953421312).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) - (2.*y.^2 - 1).*((29458278849994183.*x)./2305843009213693952 -
(29458278849994183.*x.^3)./288230376151711744 +
(29458278849994183.*x.^5)./144115188075855872 -
(4208325549999169.*x.^7)./36028797018963968) - (9.*y - 120.*y.^3 + 432.*y.^5 -
576.*y.^7 + 256.*y.^9).*((52388822904705023.*x)./2305843009213693952 -
(261944114523525115.*x.^3)./576460752303423488 +
(366721760332935161.*x.^5)./144115188075855872 -
(52388822904705023.*x.^7)./9007199254740992 +
(52388822904705023.*x.^9)./9007199254740992 -
(4762620264064093.*x.^11)./2251799813685248) - (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((410429299358253735.*x.^4)./288230376151711744 -
(35179654230707463.*x.^2)./288230376151711744 -
(27361953290550249.*x.^6)./4503599627370496 +
(105538962692122389.*x.^8)./9007199254740992 -
(11726551410235821.*x.^10)./1125899906842624 +
(3908850470078607.*x.^12)./1125899906842624 + 3908850470078607./2305843009213693952) -
(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1).*((2068387393022067.*x)./72057594037927936 -
(14478711751154469.*x.^3)./18014398509481984 +
(14478711751154469.*x.^5)./2251799813685248 - (6205162179066201.*x.^7)./281474976710656
+ (10341936965110335.*x.^9)./281474976710656 -
(2068387393022067.*x.^11)./70368744177664 + (159106722540159.*x.^13)./17592186044416) +
(5.*y - 20.*y.^3 + 16.*y.^5).*((39109301366637685.*x)./1152921504606846976 -
(586639520499565275.*x.^3)./288230376151711744 +
(1290606945099043605.*x.^5)./36028797018963968 -
(1290606945099043605.*x.^7)./4503599627370496 +
(559263009542918 8955.*x.^9)./4503599627370496 -
(3558946424364029335.*x.^11)./1125899906842624 +
(1368825547832318975.*x.^13)./281474976710656 -
(39109301366637685.*x.^15)./8796093022208 + (39109301366637685.*x.^17)./17592186044416
- (2058384282454615.*x.^19)./4398046511104) - (11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 -
1024.*y.^11).*((169700625519398687.*x.^2)./1152921504606846976 -
(169700625519398687.*x.^4)./72057594037927936 +
(509101876558196061.*x.^6)./36028797018963968 -
(363644197541568615.*x.^8)./9007199254740992 +
(266672411530483651.*x.^10)./4503599627370496 -
(24242946502771241.*x.^12)./562949953421312 +
(3463278071824463.*x.^14)./281474976710656 - 3463278071824463./2305843009213693952) +
(8.*y.^4 - 8.*y.^2 + 1).*((3569608970385653.*x)./576460752303423488 -
(24987262792699571.*x.^3)./144115188075855872 +
```

rand_coeffs_20x20_scaled.txt

```
(1737958625275311.*x.^6)./9007199254740992 +
(1737958625275311.*x.^8)./18014398509481984 + 1737958625275311./2305843009213693952) +
(13.*y - 364.*y.^3 + 2912.*y.^5 - 9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 +
4096.*y.^13).*((87605394880281183.*x.^2)./1152921504606846976 -
(146008991467135305.*x.^4)./72057594037927936 +
(749512822864627899.*x.^6)./36028797018963968 -
(963659343683093013.*x.^8)./9007199254740992 +
(1391952385320023241.*x.^10)./4503599627370496 -
(295262627189095839.*x.^12)./562949953421312 +
(146008991467135305.*x.^14)./281474976710656 -
(9733932764475687.*x.^16)./35184372088832 + (1081548084941743.*x.^18)./17592186044416 -
1081548084941743./2305843009213693952) + (160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8
+ 1).*((236468421719904283.*x.^2)./1152921504606846976 -
(236468421719904283.*x.^4)./72057594037927936 +
(709405265159712849.*x.^6)./36028797018963968 -
(506718046542652035.*x.^8)./9007199254740992 +
(371593234131278159.*x.^10)./4503599627370496 -
(33781203102843469.*x.^12)./562949953421312 +
(4825886157549067.*x.^14)./281474976710656 - 4825886157549067./2305843009213693952) -
(5026927014848929.*x.*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1))./2305843009213693952 + ((6582110948553523.*x.^2)./2305843009213693952 -
6582110948553523./4611686018427387904).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7) -
((2857653106411397.*x)./1152921504606846976 -
(2857653106411397.*x.^3)./144115188075855872 +
(2857653106411397.*x.^5)./72057594037927936 -
(408236158058771.*x.^7)./18014398509481984).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) - ((81006827832473445.*x)./2305843009213693952 -
(189015931609104705.*x.^3)./144115188075855872 +
(1020686030689165407.*x.^5)./72057594037927936 -
(1215102417487101675.*x.^7)./18014398509481984 +
(1485125176928679825.*x.^9)./9007199254740992 -
(243020483497420335.*x.^11)./1125899906842624 +
(81006827832473445.*x.^13)./562949953421312 -
(5400455188831563.*x.^15)./140737488355328).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) -
((6335736626628925.*x.^2)./1152921504606846976 -
(6335736626628925.*x.^4)./144115188075855872 +
(8870031277280495.*x.^6)./72057594037927936 -
(1267147325325785.*x.^8)./9007199254740992 + (253429465065157.*x.^10)./4503599627370496
- 253429465065157./2305843009213693952).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) - (13.*y - 364.*y.^3 +
2912.*y.^5 - 9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 +
4096.*y.^13).*((17059316671452787.*x)./2305843009213693952 -
(51177950014358361.*x.^3)./144115188075855872 +
(358245650100508527.*x.^5)./72057594037927936 -
(562957450157941971.*x.^7)./18014398509481984 +
(938262416929903285.*x.^9)./9007199254740992 -
(221771116728886231.*x.^11)./1125899906842624 +
(119415216700169509.*x.^13)./562949953421312 -
(17059316671452787.*x.^15)./140737488355328 +
(1003489215967811.*x.^17)./35184372088832) -
((16734622896377541.*x)./2305843009213693952 -
(27891038160629235.*x.^3)./288230376151711744 +
(50203868689132623.*x.^5)./144115188075855872 -
(16734622896377541.*x.^7)./36028797018963968 +
(1859402544041949.*x.^9)./9007199254740992).*(11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 - 1024.*y.^11) - ((6881462422218939.*x.^4)./576460752303423488
- (6881462422218939.*x.^2)./576460752303423488 +
6881462422218939./4611686018427387904).*(17.*y - 816.*y.^3 + 11424.*y.^5 - 71808.*y.^7
+ 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 + 65536.*y.^17) +
((6335639404905735.*x)./2305843009213693952 -
(6335639404905735.*x.^3)./576460752303423488 +
(1267127880981147.*x.^5)./144115188075855872).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) + (15.*y - 560.*y.^3 +
6048.*y.^5 - 28800.*y.^7 + 70400.*y.^9 - 92160.*y.^11 + 61440.*y.^13 -
16384.*y.^15).*((2112673225738531.*x)./576460752303423488 -
```

rand_coeffs_20x20_scaled.txt

```
(56496454311753053.*x.^11)./2251799813685248 +
(4345881100904081.*x.^13)./562949953421312) -
((2601012596284809.*x.^2)./1152921504606846976 -
(867004198761603.*x.^4)./144115188075855872 +
(289001399587201.*x.^6)./72057594037927936 -
289001399587201./2305843009213693952).*(50.*y.^2 - 400.*y.^4 + 1120.*y.^6 - 1280.*y.^8
+ 512.*y.^10 - 1) + (160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1).*((64213478400701171.*x)./2305843009213693952 -
(192640435202103513.*x.^3)./144115188075855872 +
(1348483046414724591.*x.^5)./72057594037927936 -
(2119044787223138643.*x.^7)./18014398509481984 +
(3531741312038564405.*x.^9)./9007199254740992 -
(834775219209115223.*x.^11)./1125899906842624 +
(449494348804908197.*x.^13)./562949953421312 -
(64213478400701171.*x.^15)./140737488355328 +
(3777263435335363.*x.^17)./35184372088832) -
((5022476519015565.*x)./2305843009213693952 -
(5022476519015565.*x.^3)./576460752303423488 +
(1004495303803113.*x.^5)./144115188075855872).*(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1) +
(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1).*((4466141485841177.*x)./288230376151711744 -
(13398424457523531.*x.^3)./18014398509481984 +
(93788971202664717.*x.^5)./9007199254740992 -
(147382669032758841.*x.^7)./2251799813685248 +
(245637781721264735.*x.^9)./1125899906842624 -
(58059839315935301.*x.^11)./140737488355328 +
(312629904008882239.*x.^13)./70368744177664 - (4466141485841177.*x.^15)./17592186044416
+ (262714205049481.*x.^17)./4398046511104) - (9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7
+ 256.*y.^9).*((3524423290891369.*x)./576460752303423488 -
(10573269872674107.*x.^3)./36028797018963968 +
(74012889108718749.*x.^5)./18014398509481984 -
(116305968599415177.*x.^7)./4503599627370496 +
(193843280999025295.*x.^9)./2251799813685248 -
(45817502781587797.*x.^11)./281474976710656 +
(24670963036239583.*x.^13)./140737488355328 - (3524423290891369.*x.^15)./35184372088832
+ (207319017111257.*x.^17)./8796093022208) +
((7005948445698655.*x)./2305843009213693952 -
(7005948445698655.*x.^3)./576460752303423488 +
(1401189689139731.*x.^5)./144115188075855872).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) + ((15298710394641675.*x)./1152921504606846976 -
(45896131183925025.*x.^3)./72057594037927936 +
(321272918287475175.*x.^5)./36028797018963968 -
(504857443023175275.*x.^7)./9007199254740992 +
(841429071705292125.*x.^9)./4503599627370496 -
(1988832351303417775.*x.^11)./562949953421312 +
(1070909727624917 25.*x.^13)./281474976710656 -
(152987103946416 75.*x.^15)./70368744177664 +
(899924140861275.*x.^17)./17592186044416).*(19.*y - 1140.*y.^3 + 20064.*y.^5 -
160512.*y.^7 + 695552.*y.^9 - 1770496.*y.^11 + 2723840.*y.^13 - 2490368.*y.^15 +
1245184.*y.^17 - 262144.*y.^19) + ((1835762237009433.*x)./288230376151711744 -
(3059603728349055.*x.^3)./36028797018963968 +
(5507286711028299.*x.^5)./18014398509481984 -
(1835762237009433.*x.^7)./4503599627370496 +
(203973581889937.*x.^9)./1125899906842624).*(50.*y.^2 - 400.*y.^4 + 1120.*y.^6 -
1280.*y.^8 + 512.*y.^10 - 1) + ((95084743606533639.*x.^4)./18014398509481984 -
(4527844933644459.*x.^2)./18014398509481984 -
(95084743606533639.*x.^6)./2251799813685248 +
(747094414051335735.*x.^8)./4503599627370496 -
(498062942700890 49.*x.^10)./140737488355328 +
(588619841373779 67.*x.^12)./140737488355328 - (4527844933644459.*x.^14)./17592186044416
+ (4527844933644459.*x.^16)./70368744177664 +
4527844933644459./2305843009213693952).*(17.*y - 816.*y.^3 + 11424.*y.^5 - 71808.*y.^7
+ 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 + 65536.*y.^17) -
((84730829592056039.*x.^2)./576460752303423488 -
(84730829592056039.*x.^4)./36028797018963968 +
(254192488776168117.*x.^6)./18014398509481984 -
(181566063411548655.*x.^8)./4503599627370496 +
```

rand_coeffs_20x20_scaled.txt

```
131072.*y.^18 - 1) - ((16489262192325453.*x)./2305843009213693952 -
(5496420730775151.*x.^3)./576460752303423488).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) - (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((33136908118865631.*x)./2305843009213693952 -
(55228180198109385.*x.^3)./288230376151711744 +
(99410724356596893.*x.^5)./144115188075855872 -
(33136908118865631.*x.^7)./36028797018963968 +
(3681878679873959.*x.^9)./9007199254740992) - (160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1).*((13506577402004401.*x)./2305843009213693952 -
(67532887010022005.*x.^3)./576460752303423488 +
(94546041814030807.*x.^5)./144115188075855872 -
(13506577402004401.*x.^7)./9007199254740992 +
(13506577402004401.*x.^9)./9007199254740992 -
(1227870672909491.*x.^11)./2251799813685248) - (19.*y - 1140.*y.^3 + 20064.*y.^5 -
160512.*y.^7 + 695552.*y.^9 - 1770496.*y.^11 + 2723840.*y.^13 - 2490368.*y.^15 +
1245184.*y.^17 - 262144.*y.^19).*((9796108370146623.*x.^2)./1152921504606846976 -
(16326847283577705.*x.^4)./72057594037927936 +
(83811149389032219.*x.^6)./36028797018963968 -
(107757192071612853.*x.^8)./9007199254740992 +
(155649277436774121.*x.^10)./4503599627370496 -
(33016513395679359.*x.^12)./562949953421312 +
(16326847283577705.*x.^14)./281474976710656 - (1088456485571847.*x.^16)./35184372088832
+ (120939609507983.*x.^18)./17592186044416 - 120939609507983./2305843009213693952) -
((29451981161105625.*x.^2)./288230376151711744 -
(29451981161105625.*x.^4)./36028797018963968 +
(41232773625547875.*x.^6)./18014398509481984 -
(5890396232221125.*x.^8)./2251799813685248 +
(1178079246444225.*x.^10)./1125899906842624 -
1178079246444225./576460752303423488).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) + (18.*y.^2 - 48.*y.^4 + 32.*y.^6 -
1).*((46913030474440585.*x.^2)./576460752303423488 -
(46913030474440585.*x.^4)./36028797018963968 +
(140739091423321755.*x.^6)./18014398509481984 -
(100527922445229825.*x.^8)./4503599627370496 +
(73720476459835205.*x.^10)./2251799813685248 -
(6701861496348655.*x.^12)./281474976710656 + (957408785192665.*x.^14)./140737488355328
- 957408785192665./1152921504606846976) - (15.*y - 560.*y.^3 + 6048.*y.^5 - 28800.*y.^7
+ 70400.*y.^9 - 92160.*y.^11 + 61440.*y.^13 -
16384.*y.^15).*((258727512651275331.*x.^2)./1152921504606846976 -
(431212521085458885.*x.^4)./72057594037927936 +
(2213557608238688943.*x.^6)./36028797018963968 -
(2846002639164028641.*x.^8)./9007199254740992 +
(4110892701014708037.*x.^10)./4503599627370496 -
(872007542639483523.*x.^12)./562949953421312 +
(431212521085458885.*x.^14)./281474976710656 -
(28747501405697259.*x.^16)./35184372088832 + (3194166822855251.*x.^18)./17592186044416
- 3194166822855251./2305843009213693952) -
((1527920458278441.*x.^2)./1152921504606846976 -
1527920458278441./2305843009213693952).*(2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 + 1) -
(11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7 + 2816.*y.^9 -
1024.*y.^11).*((109181744354351403.*x.^4)./18014398509481984 -
(5199130683540543.*x.^2)./18014398509481984 -
(109181744354351403.*x.^6)./2251799813685248 +
(857856562784189595.*x.^8)./4503599627370496 -
(57190437518945973.*x.^10)./140737488355328 +
(67588698886027059.*x.^12)./140737488355328 - (5199130683540543.*x.^14)./17592186044416
+ (5199130683540543.*x.^16)./70368744177664 + 5199130683540543./2305843009213693952) -
(5562992804693687.*x.*(5.*y - 20.*y.^3 + 16.*y.^5))./2305843009213693952 +
((4043840535688885.*x.^4)./288230376151711744 -
(4043840535688885.*x.^2)./288230376151711744 +
4043840535688885./2305843009213693952).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) - (50.*y.^2 - 400.*y.^4 + 1120.*y.^6 - 1280.*y.^8 + 512.*y.^10 -
1).*((159491306644480209.*x.^4)./36028797018963968 -
```

rand_coeffs_20x20_scaled.txt

```
2161848735149277./2305843009213693952).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 + 6912.*y.^8
- 6144.*y.^10 + 2048.*y.^12 + 1) - (8.*y.^4 - 8.*y.^2 +
1).*((212076315043171371.*x.^2)./576460752303423488 -
(353460525071952285.*x.^4)./36028797018963968 +
(1814430695369355063.*x.^6)./18014398509481984 -
(2332839465474885081.*x.^8)./4503599627370496 +
(3369657005685945117.*x.^10)./2251799813685248 -
(714775728478836843.*x.^12)./281474976710656 +
(353460525071952285.*x.^14)./140737488355328 -
(23564035004796819.*x.^16)./17592186044416 + (2618226111644091.*x.^18)./8796093022208 -
2618226111644091./1152921504606846976) + ((13742382227030535.*x)./1152921504606846976 -
(96196675589213745.*x.^3)./288230376151711744 +
(96196675589213745.*x.^5)./36028797018963968 -
(41227146681091605.*x.^7)./4503599627370496 +
(68711911135152675.*x.^9)./4503599627370496 -
(13742382227030535.*x.^11)./1125899906842624 +
(1057106325156195.*x.^13)./281474976710656).*(13.*y - 364.*y.^3 + 2912.*y.^5 -
9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) - (2470864224384259.*x.*(9.*y -
120.*y.^3 + 432.*y.^5 - 576.*y.^7 + 256.*y.^9))./1152921504606846976 -
((30459527997053225.*x)./2305843009213693952 -
(30459527997053225.*x.^3)./288230376151711744 +
(30459527997053225.*x.^5)./144115188075855872 -
(4351361142436175.*x.^7)./36028797018963968).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) - y.*((75203217919668825.*x.^4)./72057594037927936 -
(6445990107400185.*x.^2)./72057594037927936 -
(5013547861311255.*x.^6)./1125899906842624 +
(19337970322200555.*x.^8)./2251799813685248 -
(2148663369133395.*x.^10)./281474976710656 + (716221123044465.*x.^12)./281474976710656
+ 716221123044465./576460752303423488) - (5.*y - 20.*y.^3 +
16.*y.^5).*((3183209017997235.*x)./1152921504606846976 -
(9549627053991705.*x.^3)./72057594037927936 +
(66847389377941935.*x.^5)./36028797018963968 -
(105045897593908755.*x.^7)./9007199254740992 +
(175076495989847925.*x.^9)./4503599627370496 -
(41381717233964055.*x.^11)./562949953421312 +
(22282463125980645.*x.^13)./281474976710656 - (3183209017997235.*x.^15)./70368744177664
+ (187247589293955.*x.^17)./17592186044416) - (9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7
+ 256.*y.^9).*((46034781799557525.*x)./2305843009213693952 -
(107414490865634225.*x.^3)./144115188075855872 +
(580038250674424815.*x.^5)./72057594037927936 -
(690521726993362875.*x.^7)./18014398509481984 +
(843970999658554625.*x.^9)./9007199254740992 -
(138104345398672575.*x.^11)./1125899906842624 +
(46034781799557525.*x.^13)./562949953421312 -
(3068985453303835.*x.^15)./140737488355328) + (160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1).*((20490921193521825.*x.^2)./576460752303423488 -
(20490921193521825.*x.^4)./72057594037927936 +
(28687289670930555.*x.^6)./36028797018963968 -
(4098184238704365.*x.^8)./4503599627370496 + (819636847740873.*x.^10)./2251799813685248
- 819636847740873./1152921504606846976) + ((7869025910829985.*x.^4)./36028797018963968
- (1573805182165997.*x.^2)./36028797018963968 -
(1573805182165997.*x.^6)./4503599627370496 + (1573805182165997.*x.^8)./9007199254740992
+ 1573805182165997./1152921504606846976).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) -
y.*((5237056599713425.*x.^2)./1152921504606846976 -
5237056599713425./2305843009213693952) + (3.*y -
4.*y.^3).*((975234375507829.*x.^4)./288230376151711744 -
(975234375507829.*x.^2)./288230376151711744 + 975234375507829./2305843009213693952) -
(11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7 + 2816.*y.^9 -
1024.*y.^11).*((95870345210561337.*x)./4611686018427387904 -
(1438055178158420055.*x.^3)./1152921504606846976 +
(3163721391948524121.*x.^5)./144115188075855872 -
(3163721391948524121.*x.^7)./18014398509481984 +
(1370945365110271191.*x.^9)./18014398509481984 -
(8724201414161081667.*x.^11)./4503599627370496 +
(3355462082369646795.*x.^13)./1125899906842624 -
```

F.4 5th Bernoulli polynomial with added random Chebyshev coefficients

5th_bernoulli_with_20x20_rand.txt

```
Original function
*
(1./1).*(-((1.*(x./2+1./2))./6)+(5.*(x./2+1./2).^3)./3-5./2.*(x./2+1./2).^4+(x./2+1./2).^
5).*(-((1.*(y./2+1./2))./6)+(5.*(y./2+1./2).^3)./3-5./2.*(y./2+1./2).^4+1.*(y./2+1./2).^5
)
*
Polynomial function
*
(2828674202986023.*x)./147573952589676412928 -
(58047456932418103.*y)./73786976294838206464 - (5.*y - 20.*y.^3 +
16.*y.^5).*((3463423376055907.*x)./73786976294838206464 -
(17317116880279535.*x.^3)./18446744073709551616 +
(24243963632391349.*x.^5)./4611686018427387904 -
(3463423376055907.*x.^7)./288230376151711744 +
(3463423376055907.*x.^9)./288230376151711744 -
(314856670550537.*x.^11)./72057594037927936) - (13.*y - 364.*y.^3 + 2912.*y.^5 -
9984.*y.^7 + 16640.*y.^9 - 13312.*y.^11 +
4096.*y.^13).*((34469355521338785.*x)./295147905179352825856 -
(8042849621645716 5.*x.^3)./18446744073709551616 +
(434313879568868691.*x.^5)./9223372036854775808 -
(517040332820081775.*x.^7)./2305843009213693952 +
(631938184557877725.*x.^9)./1152921504606846976 -
(103408066564016355.*x.^11)./144115188075855872 +
(34469355521338785.*x.^13)./72057594037927936 -
(2297957034755919.*x.^15)./18014398509481984) -
((21154763654733485.*x.^2)./147573952589676412928 -
(21154763654733485.*x.^4)./9223372036854775808 +
(63464290964200455.*x.^6)./4611686018427387904 -
(45331636403000325.*x.^8)./1152921504606846976 +
(33243200028866905.*x.^10)./576460752303423488 -
(3022109093533355.*x.^12)./72057594037927936 +
(431729870504765.*x.^14)./36028797018963968 -
431729870504765./295147905179352825856).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) + (8.*y.^4 - 8.*y.^2 +
1).*((33996843488282733.*x.^2)./147573952589676412928 -
(11332281162760911.*x.^4)./18446744073709551616 +
(3777427054253637.*x.^6)./9223372036854775808 -
3777427054253637./295147905179352825856) - (2.*y.^2 -
1).*((34715618781420717.*x)./295147905179352825856 -
(57859364635701195.*x.^3)./36893488147419103232 +
(104146856344262151.*x.^5)./18446744073709551616 -
(34715618781420717.*x.^7)./4611686018427387904 +
(3857290975713413.*x.^9)./1152921504606846976) +
((17160657817976619.*x)./295147905179352825856 -
(5720219272658873.*x.^3)./73786976294838206464).*(7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7) + (18.*y.^2 - 48.*y.^4 + 32.*y.^6 -
1).*((47877637264192935.*x.^2)./147573952589676412928 -
(15959212421397645.*x.^4)./18446744073709551616 +
(5319737473799215.*x.^6)./9223372036854775808 -
5319737473799215./295147905179352825856) - (840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 +
1).*((50558108545922715.*x)./295147905179352825856 -
(117968919940486335.*x.^3)./18446744073709551616 +
(637032167678626209.*x.^5)./9223372036854775808 -
(758371628188840725.*x.^7)./2305843009213693952 +
(926898656675249775.*x.^9)./1152921504606846976 -
(151674325637768145.*x.^11)./144115188075855872 +
(50558108545922715.*x.^13)./72057594037927936 -
(3370540569728181.*x.^15)./18014398509481984) + (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((2631682071817175.*x.^2)./36893488147419103232 -
(2631682071817175.*x.^4)./4611686018427387904 +
(3684354900544045.*x.^6)./2305843009213693952 -
(526336414363435.*x.^8)./288230376151711744 +
(105267282872687.*x.^10)./144115188075855872 - 105267282872687./73786976294838206464) -
(2381032750646535.*x.*(19.*y - 1140.*y.^3 + 20064.*y.^5 - 160512.*y.^7 + 695552.*y.^9 -
1770496.*y.^11 + 2723840.*y.^13 - 2490368.*y.^15 + 1245184.*y.^17 -
```

5th_bernoulli_with_20x20_rand.txt

```
262144.*y.^19))./295147905179352825856 +
((1540502713009019.*x.^2)./1475739525896764129 28 -
1540502713009019./295147905179352825856).*(2.*y.^2 - 1) + (5.*y - 20.*y.^3 +
16.*y.^5).*((181398656377529115.*x.^4)./36893488147419103232 -
(15548456260931067.*x.^2)./36893488147419103232 -
(12093243758501941.*x.^6)./576460752303423488 +
(46645368782793201.*x.^8)./1152921504606846976 -
(5182818753643689.*x.^10)./144115188075855872 +
(1727606251214563.*x.^12)./144115188075855872 +
1727606251214563./295147905179352825856) + (11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7
+ 2816.*y.^9 - 1024.*y.^11).*((9930708730190859.*x)./295147905179352825856 -
(1489606309528628 85.*x.^3)./73786976294838206464 +
(327713388096298347.*x.^5)./9223372036854775808 -
(327713388096298347.*x.^7)./1152921504606846976 +
(1420091348417292837.*x.^9)./1152921504606846976 -
(903694494447368169.*x.^11)./288230376151711744 +
(347574805556680065.*x.^13)./72057594037927936 -
(9930708730190859.*x.^15)./2251799813685248 +
(9930708730190859.*x.^17)./4503599627370496 -
(522668880536361.*x.^19)./1125899906842624) +
((29968688501426697.*x)./147573952589676412928 -
(1498434425071334 85.*x.^3)./36893488147419103232 +
(209780819509986879.*x.^5)./9223372036854775808 -
(29968688501426697.*x.^7)./576460752303423488 +
(29968688501426697.*x.^9)./576460752303423488 -
(2724426227402427.*x.^11)./144115188075855872).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) - (160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1).*((88236082882282101.*x)./295147905179352825856 -
(617652580175974707.*x.^3)./73786976294838206464 +
(617652580175974707.*x.^5)./9223372036854775808 -
(264708248646846303.*x.^7)./1152921504606846976 +
(441180414411410505.*x.^9)./1152921504606846976 -
(88236082882282101.*x.^11)./288230376151711744 +
(6787390990944777.*x.^13)./72057594037927936) - (18.*y.^2 - 48.*y.^4 + 32.*y.^6 -
1).*((31535270975299077.*x.^2)./36893488147419103232 -
(31535270975299077.*x.^4)./2305843009213693952 +
(94605812925897231.*x.^6)./1152921504606846976 -
(67575580661355165.*x.^8)./288230376151711744 +
(49555425818327121.*x.^10)./144115188075855872 -
(4505038710757011.*x.^12)./18014398509481984 +
(643576958679573.*x.^14)./9007199254740992 - 643576958679573./73786976294838206464) +
(3.*y - 4.*y.^3).*((2574411175906287.*x.^4)./18446744073709551616 -
(2574411175906287.*x.^2)./18446744073709551616 +
2574411175906287./147573952589676412928) -
((191452786513863475.*x.^2)./295147905179352825856 -
(191452786513863475.*x.^4)./36893488147419103232 +
(268033901119408865.*x.^6)./18446744073709551616 -
(38290557302772695.*x.^8)./2305843009213693952 +
(7658111460554539.*x.^10)./1152921504606846976 -
7658111460554539./590295810358705651712).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) + ((105452912800865.*x.^2)./9223372036854775808 -
105452912800865./18446744073709551616).*(15.*y - 560.*y.^3 + 6048.*y.^5 - 28800.*y.^7 +
70400.*y.^9 - 92160.*y.^11 + 61440.*y.^13 - 16384.*y.^15) +
((31607227710867845.*x.^4)./9223372036854775808 -
(6321445542173569.*x.^2)./9223372036854775808 -
(6321445542173569.*x.^6)./1152921504606846976 +
(6321445542173569.*x.^8)./2305843009213693952 +
6321445542173569./295147905179352825856).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) - ((7324350879992811.*x.^2)./147573952589676412928 -
7324350879992811./295147905179352825856).*(3.*y - 4.*y.^3) + (5.*y - 20.*y.^3 +
16.*y.^5).*((7590650588131365.*x)./73786976294838206464 -
(17711518038973185.*x.^3)./4611686018427387904 +
(95642197410455199.*x.^5)./2305843009213693952 -
(113859758821970475.*x.^7)./576460752303423488 +
```

5th_bernoulli_with_20x20_rand.txt

```
(11740490008743405.*x.^12)./72057594037927936 +
(1677212858391915.*x.^14)./36028797018963968 - 1677212858391915./295147905179352825856)
- (160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1).*((354500891665688115.*x.^2)./147573952589676412928 -
(590834819442813525.*x.^4)./9223372036854775808 +
(3032952073139776095.*x.^6)./4611686018427387904 -
(3899509808322569265.*x.^8)./1152921504606846976 +
(5632625278688155605.*x.^10)./576460752303423488 -
(1194799301539911795.*x.^12)./72057594037927936 +
(590834819442813525.*x.^14)./36028797018963968 -
(39388987962854235.*x.^16)./4503599627370496 +
(4376554218094915.*x.^18)./2251799813685248 - 4376554218094915./295147905179352825856)
- (3.*y - 4.*y.^3).*((50948045875154631.*x)./295147905179352825856 -
(50948045875154631.*x.^3)./36893488147419103232 +
(50948045875154631.*x.^5)./18446744073709551616 -
(7278292267879233.*x.^7)./4611686018427387904) +
((9430801764988415.*x.^4)./9223372036854775808 -
(1886160352997683.*x.^2)./9223372036854775808 -
(1886160352997683.*x.^6)./1152921504606846976 +
(1886160352997683.*x.^8)./2305843009213693952 +
1886160352997683./295147905179352825856).*(2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 + 1) +
(6712302504183917.*x.*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1))./295147905179352825856 - (8.*y.^4 - 8.*y.^2 +
1).*((8470441880392443.*x.^4)./2305843009213693952 -
(403354375256783.*x.^2)./2305843009213693952 -
(8470441880392443.*x.^6)./288230376151711744 +
(66553471917369195.*x.^8)./576460752303423488 -
(4436898127824613.*x.^10)./18014398509481984 +
(5243606878338179.*x.^12)./18014398509481984 -
(403354375256783.*x.^14)./2251799813685248 + (403354375256783.*x.^16)./9007199254740992
+ 403354375256783./295147905179352825856) +
((1170159453644025.*x.^2)./36893488147419103232 -
1170159453644025./73786976294838206464).*(19.*y - 1140.*y.^3 + 20064.*y.^5 -
160512.*y.^7 + 695552.*y.^9 - 1770496.*y.^11 + 2723840.*y.^13 - 2490368.*y.^15 +
1245184.*y.^17 - 262144.*y.^19) - ((4979284124573925.*x)./295147905179352825856 -
(4979284124573925.*x.^3)./73786976294838206464 +
(995856824914785.*x.^5)./18446744073709551616).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1) + (5.*y - 20.*y.^3 +
16.*y.^5).*((59012090608842783.*x.^2)./147573952589676412928 -
(19670696869614261.*x.^4)./18446744073709551616 +
(6556898956538087.*x.^6)./9223372036854775808 -
6556898956538087./295147905179352825856) + (3.*y -
4.*y.^3).*((10998159305752809.*x)./147573952589676412928 -
(18330265509588015.*x.^3)./18446744073709551616 +
(32994477917258427.*x.^5)./9223372036854775808 -
(10998159305752809.*x.^7)./2305843009213693952 +
(1222017700639201.*x.^9)./576460752303423488) + (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((6440622452265651.*x)./295147905179352825856 -
(45084357165859557.*x.^3)./73786976294838206464 +
(45084357165859557.*x.^5)./9223372036854775808 -
(19321867356796953.*x.^7)./1152921504606846976 +
(32203112261328255.*x.^9)./1152921504606846976 -
(6440622452265651.*x.^11)./288230376151711744 +
(495432496328127.*x.^13)./72057594037927936) + ((54975581388801.*x)./576460752303423488
- (18325193796267.*x.^3)./144115188075855872).*(5.*y - 20.*y.^3 + 16.*y.^5) -
((171908985079438O9.*x)./73786976294838206464 -
(85954492539719045.*x.^3)./18446744073709551616 +
(120336289555606663.*x.^5)./4611686018427387904 -
(171908985079438O9.*x.^7)./288230376151711744 +
(171908985079438O9.*x.^9)./288230376151711744 -
(1562808955267619.*x.^11)./72057594037927936).*(19.*y - 1140.*y.^3 + 20064.*y.^5 -
160512.*y.^7 + 695552.*y.^9 - 1770496.*y.^11 + 2723840.*y.^13 - 2490368.*y.^15 +
1245184.*y.^17 - 262144.*y.^19) + ((20351800053188303.*x)./295147905179352825856 -
(20351800053188303.*x.^3)./36893488147419103232 +
(20351800053188303.*x.^5)./18446744073709551616 -
(2907400007598329.*x.^7)./4611686018427387904).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
```

5th_bernoulli_with_20x20_rand.txt

```
(4830997968581349.*x.^7)./4611686018427387904).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) +
((685392803002797.*x)./9223372036854775808 -
(228464267667599.*x.^3)./2305843009213693952).*(50.*y.^2 - 400.*y.^4 + 1120.*y.^6 -
1280.*y.^8 + 512.*y.^10 - 1) + y.*((87233756689189845.*x)./590295810358705651712 -
(203545432274776305.*x.^3)./36893488147419103232 +
(109914533428379204 7.*x.^5)./18446744073709551616 -
(1308506350337847675.*x.^7)./4611686018427387904 +
(1599285539301813825.*x.^9)./2305843009213693952 -
(261701270067569535.*x.^11)./288230376151711744 +
(87233756689189845.*x.^13)./144115188075855872 -
(5815583779279323.*x.^15)./36028797018963968) -
y.*((2953220325438383.*x)./147573952589676412928 -
(20672542278068681.*x.^3)./36893488147419103232 +
(20672542278068681.*x.^5)./4611686018427387904 -
(8859660976315149.*x.^7)./576460752303423488 +
(14766101627191915.*x.^9)./576460752303423488 -
(2953220325438383.*x.^11)./144115188075855872 +
(227170794264491.*x.^13)./36028797018963968) - (2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 +
1).*((43464390942259089.*x)./147573952589676412928 -
(130393172826777267.*x.^3)./9223372036854775808 +
(912752209787440869.*x.^5)./4611686018427387904 -
(1434324901094549937.*x.^7)./1152921504606846976 +
(2390541501824249895.*x.^9)./576460752303423488 -
(565037082249368157.*x.^11)./72057594037927936 +
(304250736595813623.*x.^13)./36028797018963968 -
(43464390942259089.*x.^15)./9007199254740992 +
(2556728878956417.*x.^17)./2251799813685248) - (5.*y - 20.*y.^3 +
16.*y.^5).*((122384865141792057.*x.^4)./4611686018427387904 -
(5827850721037717.*x.^2)./4611686018427387904 -
(122384865141792057.*x.^6)./576460752303423488 +
(961595368971223305.*x.^8)./1152921504606846976 -
(64106357931414887.*x.^10)./36028797018963968 +
(75762059373490321.*x.^12)./36028797018963968 -
(5827850721037717.*x.^14)./4503599627370496 +
(5827850721037717.*x.^16)./18014398509481984 + 5827850721037717./590295810358705651712)
- ((12196477230903801.*x)./295147905179352825856 -
(60982386154519005.*x.^3)./73786976294838206464 +
(85375340616326607.*x.^5)./18446744073709551616 -
(12196477230903801.*x.^7)./1152921504606846976 +
(12196477230903801.*x.^9)./1152921504606846976 -
(1108770657354891.*x.^11)./288230376151711744).*(2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 + 1) -
((4484444688301515.*x)./147573952589676412928 -
(4484444688301515.*x.^3)./18446744073709551616 +
(4484444688301515.*x.^5)./9223372036854775808 -
(640634955471645.*x.^7)./2305843009213693952).*(2688.*y.^4 - 128.*y.^2 - 21504.*y.^6 +
84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 + 32768.*y.^16 + 1) + (7.*y
- 56.*y.^3 + 112.*y.^5 - 64.*y.^7).*((518242915971887925.*x.^4)./36893488147419103232 -
(44420821369018965.*x.^2)./36893488147419103232 -
(34549527731459195.*x.^6)./576460752303423488 +
(133262464107056895.*x.^8)./1152921504606846976 -
(14806940456339655.*x.^10)./144115188075855872 +
(4935646818779885.*x.^12)./144115188075855872 +
4935646818779885./295147905179352825856) + (5.*y - 20.*y.^3 +
16.*y.^5).*((239266112373203481.*x.^2)./147573952589676412928 -
(398776853955339135.*x.^4)./9223372036854775808 +
(2047054516970740893.*x.^6)./4611686018427387904 -
(2631927236105238291.*x.^8)./1152921504606846976 +
(3801672674374233087.*x.^10)./576460752303423488 -
(806415415776352473.*x.^12)./72057594037927936 +
(398776853955339135.*x.^14)./36028797018963968 -
(265851235970226 09.*x.^16)./4503599627370496 +
(2953902621891401.*x.^18)./2251799813685248 - 2953902621891401./295147905179352825856)
+ (162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 - 228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12
+ 1105920.*y.^14 - 589824.*y.^16 + 131072.*y.^18 -
```

5th_bernoulli_with_20x20_rand.txt

```
(545602348521603.*x.^10)./576460752303423488 - 545602348521603./295147905179352825856)
+ (8.*y.^4 - 8.*y.^2 + 1).*((22607502939428165.*x.^4)./9223372036854775808 -
(4521500587885633.*x.^2)./9223372036854775808 -
(4521500587885633.*x.^6)./1152921504606846976 +
(4521500587885633.*x.^8)./2305843009213693952 +
4521500587885633./295147905179352825856) - (2.*y.^2 -
1).*((39913783077890755.*x.^4)./18446744073709551616 -
(7982756615578151.*x.^2)./18446744073709551616 -
(7982756615578151.*x.^6)./2305843009213693952 +
(7982756615578151.*x.^8)./461168601842738790 4 +
7982756615578151./590295810358705651712) + ((4538751574118201.*x)./73786976294838206464
- (22693757870591005.*x.^3)./18446744073709551616 +
(31771261018827407.*x.^5)./4611686018427387904 -
(4538751574118201.*x.^7)./288230376151711744 +
(4538751574118201.*x.^9)./288230376151711744 -
(412613779465291.*x.^11)./72057594037927936).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) -
((2903883535242265.*x)./73786976294838206464 -
(2903883535242265.*x.^3)./18446744073709551616 +
(580776707048453.*x.^5)./4611686018427387904).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) + (160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1).*((32772188340803879.*x)./295147905179352825856 -
(163860941704019395.*x.^3)./73786976294838206464 +
(229405318385627153.*x.^5)./18446744073709551616 -
(32772188340803879.*x.^7)./1152921504606846976 +
(32772188340803879.*x.^9)./1152921504606846976 -
(2979289849163989.*x.^11)./288230376151711744) - (5.*y - 20.*y.^3 +
16.*y.^5).*((23230056806591585.*x)./147573952589676412928 -
(69690170419774755.*x.^3)./9223372036854775808 +
(487831192938423285.*x.^5)./4611686018427387904 -
(766591874617522305.*x.^7)./1152921504606846976 +
(1277653124362537175.*x.^9)./576460752303423488 -
(301990738485690605.*x.^11)./72057594037927936 +
(162610397646141095.*x.^13)./36028797018963968 -
(23230056806591585.*x.^15)./9007199254740992 +
(1366473929799505.*x.^17)./2251799813685248) +
((12073849079555799.*x)./295147905179352825856 -
(20123081799259665.*x.^3)./36893488147419103232 +
(36221547238667397.*x.^5)./18446744073709551616 -
(12073849079555799.*x.^7)./4611686018427387904 +
(1341538786617311.*x.^9)./1152921504606846976).*(162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1) + ((60604951800452067.*x.^2)./147573952589676412928 -
(20201650600150689.*x.^4)./18446744073709551616 +
(6733883533383563.*x.^6)./9223372036854775808 -
6733883533383563./295147905179352825856).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) + (9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9).*((2605436868519675.*x)./18446744073709551616 -
(6079352693212575.*x.^3)./1152921504606846976 +
(32828504543347905.*x.^5)./576460752303423488 -
(39081553027795125.*x.^7)./144115188075855872 +
(47766342589527375.*x.^9)./72057594037927936 -
(7816310605559025.*x.^11)./9007199254740992 +
(2605436868519675.*x.^13)./4503599627370496 -
(173695791234645.*x.^15)./1125899906842624) + (162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1).*((2764659776788251.*x.^2)./18446744073709551616 -
(4607766294647085.*x.^4)./1152921504606846976 +
(23653200312521703.*x.^6)./576460752303423488 -
(30411257544670761.*x.^8)./144115188075855872 +
(43927372008968877.*x.^10)./72057594037927936 -
(9317927395841883.*x.^12)./9007199254740992 +
(4607766294647085.*x.^14)./4503599627370496 - (307184419643139.*x.^16)./562949953421312
+ (34131602182571.*x.^18)./281474976710656 - 34131602182571./36893488147419103232) +
(9.*y - 120.*y.^3 + 432.*y.^5 - 576.*y.^7 +
256.*y.^9).*((15434499961995555.*x.^4)./36893488147419103232 -
```

5th_bernoulli_with_20x20_rand.txt

```
(921756462888235689.*x.^10)./144115188075855872 -
(1955240981884136 31.*x.^12)./18014398509481984 +
(96687740862402345.*x.^14)./9007199254740992 -
(6445849390826823.*x.^16)./1125899906842624 + (716205487869647.*x.^18)./562949953421312
- 716205487869647./73786976294838206464) +
((4128978264310679.*x.^4)./36893488147419103232 -
(4128978264310679.*x.^2)./36893488147419103232 +
4128978264310679./295147905179352825856).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) + (11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 - 1024.*y.^11).*((54802207136292693.*x.^4)./2305843009213693952
- (2609628911252033.*x.^2)./2305843009213693952 -
(54802207136292693.*x.^6)./288230376151711744 +
(430588770356585445.*x.^8)./576460752303423488 -
(28705918023772363.*x.^10)./18014398509481984 +
(33925175846276429.*x.^12)./18014398509481984 -
(2609628911252033.*x.^14)./2251799813685248 +
(2609628911252033.*x.^16)./9007199254740992 + 2609628911252033./295147905179352825856)
- ((24813279488784245.*x)./295147905179352825856 -
(24813279488784245.*x.^3)./73786976294838206464 +
(4962655897756849.*x.^5)./18446744073709551616).*(8.*y.^4 - 8.*y.^2 + 1) +
((46726877568666603.*x)./295147905179352825856 -
(233634387843333015.*x.^3)./73786976294838206464 +
(327088142980666221.*x.^5)./18446744073709551616 -
(46726877568666603.*x.^7)./1152921504606846976 +
(46726877568666603.*x.^9)./1152921504606846976 -
(4247897960787873.*x.^11)./288230376151711744).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) -
((8869543491138855.*x.^4)./36893488147419103232 -
(760246584954759.*x.^2)./36893488147419103232 -
(591302899409257.*x.^6)./576460752303423488 +
(2280739754864277.*x.^8)./1152921504606846976 -
(253415528318253.*x.^10)./144115188075855872 +
(84471842772751.*x.^12)./144115188075855872 +
84471842772751./295147905179352825856).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) - (18.*y.^2 - 48.*y.^4 +
32.*y.^6 - 1).*((48552848663561935.*x)./295147905179352825856 -
(242764243317809675.*x.^3)./73786976294838206464 +
(339869940644933545.*x.^5)./18446744073709551616 -
(48552848663561935.*x.^7)./1152921504606846976 +
(48552848663561935.*x.^9)./1152921504606846976 -
(4413895333051085.*x.^11)./288230376151711744) -
((1765438848080257.*x)./4611686018427387904 -
(5296316544240771.*x.^3)./288230376151711744 +
(37074215809685397.*x.^5)./144115188075855872 -
(58259481986648481.*x.^7)./360287970189639 68 +
(97099136644414135.*x.^9)./18014398509481984 -
(22950705025043341.*x.^11)./2251799813685248 +
(12358071936561799.*x.^13)./1125899906842624 -
(1765438848080257.*x.^15)./281474976710656 +
(103849344004721.*x.^17)./70368744177664).*(17.*y - 816.*y.^3 + 11424.*y.^5 -
71808.*y.^7 + 239360.*y.^9 - 452608.*y.^11 + 487424.*y.^13 - 278528.*y.^15 +
65536.*y.^17) - ((90201496901085765.*x)./295147905179352825856 -
(631410478307600355.*x.^3)./73786976294838206464 +
(631410478307600355.*x.^5)./9223372036854775808 -
(270604490703257295.*x.^7)./1152921504606846976 +
(451007484505428825.*x.^9)./1152921504606846976 -
(90201496901085765.*x.^11)./288230376151711744 +
(6938576684698905.*x.^13)./72057594037927936).*(15.*y - 560.*y.^3 + 6048.*y.^5 -
28800.*y.^7 + 70400.*y.^9 - 92160.*y.^11 + 61440.*y.^13 - 16384.*y.^15) +
((11661531997312053.*x)./73786976294838206464 -
(19435886662186755.*x.^3)./9223372036854775808 +
(34984595991936159.*x.^5)./4611686018427387904 -
(11661531997312053.*x.^7)./1152921504606846976 +
(1295725777479117.*x.^9)./288230376151711744).*(15.*y - 560.*y.^3 + 6048.*y.^5 -
28800.*y.^7 + 70400.*y.^9 - 92160.*y.^11 + 61440.*y.^13 - 16384.*y.^15) +
((3216273726223863.*x)./147573952589676412928 -
(5360456210373105.*x.^3)./18446744073709551616 +
```

F.5 4 term cosine with added random Chebyshev coefficients

4_term_cosine_rand_15x15.txt

```
Original function
*
1.*(1.*cos(2.*pi.*x./4) + 1./4.*cos(2.*pi.*x./2)  + 1./2.*cos(2.*pi.*y./1) +
1./3.*cos(2.*pi.*y./10))
*
Polynomial function
*
(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1).*((2608878552014205.*x.^4)./1125899906842624 -
(521775710402841.*x.^2)./1125899906842624 - (521775710402841.*x.^6)./140737488355328 +
(521775710402841.*x.^8)./281474976710656 + 521775710402841./36028797018963968) -
(461977591100978925.*y)./576460752303423488 -
(98296263189533021.*x)./288230376151711744 -
y.*((3957706007584465.*x)./36028797018963968 -
(3957706007584465.*x.^3)./9007199254740992 + (791541201516893.*x.^5)./2251799813685248)
+ (50.*y.^2 - 400.*y.^4 + 1120.*y.^6 - 1280.*y.^8 + 512.*y.^10 -
1).*((30113595657792129.*x.^4)./4056481920730334084789450257 2032 -
(1433980745609149.*x.^2)./4056481920730334084789450257 2032 -
(30113595657792129.*x.^6)./5070602400912917605986812821504 +
(236606823025509585.*x.^8)./10141204801825835211973625643008 -
(15773788201700639.*x.^10)./3169126500570573503741758013 44 +
(18641749692918937.*x.^12)./3169126500570573503741758013 44 -
(1433980745609149.*x.^14)./3961408125713216879677197 5168 +
(1433980745609149.*x.^16)./1584563250285286751870879 00672 +
1433980745609149./5192296858534827628530496329220096) -
y.*((33655227599540475.*x.^4)./9007199254740992 -
(6731045519908095.*x.^2)./9007199254740992 - (6731045519908095.*x.^6)./1125899906842624
+ (6731045519908095.*x.^8)./2251799813685248 + 6731045519908095./288230376151711744) +
((10596168374263325.*x.^2)./72057594037927936 -
(10596168374263325.*x.^4)./9007199254740992 +
(14834635723968655.*x.^6)./4503599627370496 - (2119233674852665.*x.^8)./562949953421312
+ (423846734970533.*x.^10)./281474976710656 -
423846734970533./144115188075855872).*(50.*y.^2 - 400.*y.^4 + 1120.*y.^6 - 1280.*y.^8 +
512.*y.^10 - 1) + ((7047534008753543.*x.^2)./144115188075855872 -
7047534008753543./288230376151711744).*(11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7 +
2816.*y.^9 - 1024.*y.^11) +
((798102260769309195.*x.^4)./3402823669209384634633746074317682114 56 -
(68408765208797931.*x.^2)./3402823669209384634633746074317682114 56 -
(53206817384620613.*x.^6)./5316911983139663491615228241121378304 +
(205226295626393793.*x.^8)./1063382396627932698323045648224275 6608 -
(22802921736265977.*x.^10)./1329227995784915872903807060280344576 +
(7600973912088659.*x.^12)./1329227995784915872903807060280344576 +
7600973912088659./2722258935367507707706996859454145691648).*(242.*y.^2 - 9680.*y.^4 +
151008.*y.^6 - 1208064.*y.^8 + 5637632.*y.^10 - 16400384.*y.^12 + 30638080.*y.^14 -
36765696.*y.^16 + 27394048.*y.^18 - 11534336.*y.^20 + 2097152.*y.^22 - 1) - (5.*y -
20.*y.^3 + 16.*y.^5).*((12183535553757225.*x.^2)./72057594037927936 -
(12183535553757225.*x.^4)./9007199254740992 +
(17056949775260115.*x.^6)./4503599627370496 - (2436707110751445.*x.^8)./562949953421312
+ (487341422150289.*x.^10)./281474976710656 - 487341422150289./144115188075855872) -
((968335726773157.*x.^4)./4056481920730334084789450257 2032 -
(968335726773157.*x.^2)./4056481920730334084789450257 2032 +
968335726773157./3245185536584267267831560205 76256).*(13728.*y.^4 - 288.*y.^2 -
256256.*y.^6 + 2471040.*y.^8 - 14057472.*y.^10 + 50692096.*y.^12 - 120324096.*y.^14 +
190513152.*y.^16 - 199229440.*y.^18 + 132120576.*y.^20 - 50331648.*y.^22 +
8388608.*y.^24 + 1) + ((10183364336582277.*x)./144115188075855872 -
(3394454778860759.*x.^3)./36028797018963968).*(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7)
+ (7.*y - 56.*y.^3 + 112.*y.^5 -
64.*y.^7).*((12865948015192125.*x.^4)./4503599627370496 -
(1102795544159325.*x.^2)./4503599627370496 - (857729867679475.*x.^6)./70368744177664 +
(3308386632477975.*x.^8)./140737488355328 - (367598514719775.*x.^10)./17592186044416 +
(122532838239925.*x.^12)./17592186044416 + 122532838239925./36028797018963968) -
((36767902063004937.*x)./144115188075855872 -
(183839510315024685.*x.^3)./36028797018963968 +
(257375314441034559.*x.^5)./9007199254740992 -
(36767902063004937.*x.^7)./562949953421312 + (36767902063004937.*x.^9)./562949953421312
- (3342536551182267.*x.^11)./140737488355328).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) -
((213546387842757.*x)./288230376151711744 - (355910646404595.*x.^3)./36028797018963968
```

4_term_cosine_rand_15x15.txt

```
+ (640639163528271.*x.^5)./18014398509481984 -
(213546387842757.*x.^7)./4503599627370496 +
(23727376426973.*x.^9)./1125899906842624).*(13.*y - 364.*y.^3 + 2912.*y.^5 - 9984.*y.^7
+ 16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) +
((2962050285610549.*x.^4)./18014398509481984 -
(2962050285610549.*x.^2)./18014398509481984 +
2962050285610549./144115188075855872).*(5.*y - 20.*y.^3 + 16.*y.^5) + (160.*y.^4 -
32.*y.^2 - 256.*y.^6 + 128.*y.^8 + 1).*((80402762310964475.*x.^2)./72057594037927936 -
(80402762310964475.*x.^4)./9007199254740992 +
(112563867235350265.*x.^6)./4503599627370496 -
(16080552462192895.*x.^8)./562949953421312 + (3216110492438579.*x.^10)./281474976710656
- 3216110492438579./144115188075855872) - (162.*y.^2 - 4320.*y.^4 + 44352.*y.^6 -
228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 - 589824.*y.^16 +
131072.*y.^18 - 1).*((1618958484071775.*x.^4)./83076749736557242056487941267521536 -
(49059348002175.*x.^2)./83076749736557242056487941267521536 -
(323791696814355.*x.^6)./1298074214633706907132624082305024 +
(4209292058586615.*x.^8)./2596148429267413814265248164610048 -
(1964336294007087.*x.^10)./32451855365842672678315602057625 6 +
(4464400668197925.*x.^12)./324518553658426726783156020576256 -
(49059348002175.*x.^14)./25353012004564588802993406410752 +
(166801783207395.*x.^16)./10141204801825835211973625643008 -
(9811869600435.*x.^18)./12676506002282294014967032053 76 +
(1962373920087.*x.^20)./126765060022822940149670320537 6 +
1962373920087./66461399789245793645190353014017228 8) +
((120795628932333.*x.^2)./10633823966279326983230456482242756608 -
(201326048220555.*x.^4)./66461399789245793645190353014017228 8 +
(1033473714198849.*x.^6)./33230699894622896822595176507008 6144 -
(1328751918255663.*x.^8)./83076749736557242056487941267521536 +
(1919308326369291.*x.^10)./41538374868278621028243970633760768 -
(407126008623789.*x.^12)./519229685853482762853049632922009 6 +
(201326048220555.*x.^14)./25961484292674138142652481646100 48 -
(13421736548037.*x.^16)./32451855365842672678315602057625 6 +
(1491304060893.*x.^18)./16225927682921336339157801028812 8 -
1491304060893./21267647932558653966460912964485513216).*(6600.*y.^4 - 200.*y.^2 -
84480.*y.^6 + 549120.*y.^8 - 2050048.*y.^10 + 4659200.*y.^12 - 6553600.*y.^14 +
5570560.*y.^16 - 2621440.*y.^18 + 524288.*y.^20 + 1) +
((1421712268898133.*x)./36028797018963968 -
(473904089632711.*x.^3)./9007199254740992).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) + (11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 - 1024.*y.^11).*((56953791899779625.*x.^2)./72057594037927936 -
(56953791899779625.*x.^4)./9007199254740992 +
(79735308659691475.*x.^6)./4503599627370496 -
(11390758379955925.*x.^8)./562949953421312 + (2278151675991185.*x.^10)./281474976710656
- 2278151675991185./144115188075855872) -
((735918091981820895.*x.^4)./72057594037927936 -
(63078693598441791.*x.^2)./72057594037927936 -
(49061206132121393.*x.^6)./1125899906842624 +
(189236080795325373.*x.^8)./2251799813685248 -
(21026231199480597.*x.^10)./281474976710656 +
(7008743733160199.*x.^12)./281474976710656 +
7008743733160199./576460752303423488).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) -
((13337887460798637.*x.^2)./36028797018963968 -
(4445962486932879.*x.^4)./4503599627370496 + (1481987495644293.*x.^6)./2251799813685248
- 1481987495644293./72057594037927936).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1) - (3.*y - 4.*y.^3).*((4075271170391145.*x)./288230376151711744 -
(4075271170391145.*x.^3)./72057594037927936 +
(815054234078229.*x.^5)./18014398509481984) - (3.*y -
4.*y.^3).*((121347041712851325.*x.^4)./36028797018963968 -
(10401175003958685.*x.^2)./36028797018963968 -
(8089802780856755.*x.^6)./562949953421312 + (31203525011876055.*x.^8)./1125899906842624
- (3467058334652895.*x.^10)./140737488355328 +
(1155686111550965.*x.^12)./140737488355328 + 1155686111550965./288230376151711744) -
(7.*y - 56.*y.^3 + 112.*y.^5 - 64.*y.^7).*((4453512560324503.*x)./18014398509481984 -
(31174587922271521.*x.^3)./4503599627370496 +
(31174587922271521.*x.^5)./562949953421312 - (13360537680973509.*x.^7)./70368744177664
+ (2267562801622515.*x.^9)./70368744177664 - (4453512560324503.*x.^11)./17592186044416
```

4_term_cosine_rand_15x15.txt

```
(26986970381723313.*x.^7)./562949953421312 + (44978283969538855.*x.^9)./562949953421312
- (8995656793907771.*x.^11)./140737488355328 +
(6919735995531367.*x.^13)./35184372088832) -
((3623597380759497.*x.^2)./81129638414606681695789005144064 -
3623597380759497./16225927682921336339157801028812 8).*(2688.*y.^4 - 128.*y.^2 -
21504.*y.^6 + 84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 +
32768.*y.^16 + 1) + (840.*y.^4 - 72.*y.^2 - 3584.*y.^6 + 6912.*y.^8 - 6144.*y.^10 +
2048.*y.^12 + 1).*((272562585660712875.*x.^4)./12980742146337069071326240823050 24 -
(8259472292748875.*x.^2)./12980742146337069071326240823050 24 -
(54512517132142575.*x.^6)./20282409603651670423947251286016 +
(708662722717853475.*x.^8)./40564819207303340847894502572032 -
(330709270601664955.*x.^10)./50706024009129176059868128215 04 +
(751611978640147625.*x.^12)./50706024009129176059868128215 04 -
(8259472292748875.*x.^14)./39614081257132168796771975168 +
(28082205795346175.*x.^16)./15845632502852867518708790067 2 -
(1651894458549775.*x.^18)./19807040628566084398385987584 +
(330378891709955.*x.^20)./19807040628566084398385987584 +
330378891709955./10384593717069655257060992658440192) + (840.*y.^4 - 72.*y.^2 -
3584.*y.^6 + 6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 +
1).*((82958757588064085.*x.^2)./72057594037927936 -
(82958757588064085.*x.^4)./4503599627370496 +
(248876272764192255.*x.^6)./2251799813685248 -
(177768766260137325.*x.^8)./562949953421312 +
(130363761924100705.*x.^10)./281474976710656 -
(11851251084009155.*x.^12)./35184372088832 + (1693035869144165.*x.^14)./17592186044416
- 1693035869144165./144115188075855872) + (8.*y.^4 - 8.*y.^2 +
1).*((27714916864420373.*x.^2)./18014398509481984 -
(27714916864420373.*x.^4)./1125899906842624 +
(83144750593261119.*x.^6)./562949953421312 - (59389107566615085.*x.^8)./140737488355328
+ (43552012215517729.*x.^10)./70368744177664 - (3959273837774339.*x.^12)./8796093022208
+ (565610548253477.*x.^14)./4398046511104 - 565610548253477./36028797018963968) + (5.*y
- 20.*y.^3 + 16.*y.^5).*((140683536830216191.*x.^2)./72057594037927936 -
(140683536830216191.*x.^4)./4503599627370496 +
(422050610490648573.*x.^6)./2251799813685248 -
(301464721779034695.*x.^8)./562949953421312 +
(221074129304625443.*x.^10)./281474976710656 -
(20097648118602313.*x.^12)./35184372088832 + (2871092588371759.*x.^14)./17592186044416
- 2871092588371759./144115188075855872) +
((10978287616806597.*x.^2)./144115188075855872 -
(3659429205602199.*x.^4)./18014398509481984 +
(1219809735200733.*x.^6)./9007199254740992 -
1219809735200733./288230376151711744).*(13.*y - 364.*y.^3 + 2912.*y.^5 - 9984.*y.^7 +
16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) + (98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 -
1).*((81244630794742401.*x.^4)./20769187434139310514121985316880384 -
(3868791942606781.*x.^2)./20769187434139310514121985316880384 -
(81244630794742401.*x.^6)./25961484292674138142652481646100 48 +
(638350670530118865.*x.^8)./51922968585348276285304963292200 96 -
(42556711368674591.*x.^10)./16225927682921336339157801028812 8 +
(50294295253888153.*x.^12)./16225927682921336339157801028812 8 -
(3868791942606781.*x.^14)./20282409603651670423947251286016 +
(3868791942606781.*x.^16)./81129638414606681695789005144064 +
3868791942606781./26584559915698317458076141205606891 52) +
((4682884029127873.*x.^4)./64903710731685345356631204115 2512 -
(4682884029127873.*x.^2)./64903710731685345356631204115 2512 +
4682884029127873./51922968585348276285304963292200 96).*(338.*y.^2 - 18928.*y.^4 +
416416.*y.^6 - 4759040.*y.^8 + 32361472.*y.^10 - 141213696.*y.^12 + 412778496.*y.^14 -
825556992.*y.^16 + 1133117440.*y.^18 - 1049624576.*y.^20 + 627048448.*y.^22 -
218103808.*y.^24 + 33554432.*y.^26 - 1) - ((32572631553376005.*x.^4)./36028797018963968
- (2791939847432229.*x.^2)./36028797018963968 -
(2171508770225067.*x.^6)./562949953421312 + (8375819542296687.*x.^8)./1125899906842624
- (930646615810743.*x.^10)./140737488355328 + (310215538603581.*x.^12)./140737488355328
+ 310215538603581./288230376151711744).*(13.*y - 364.*y.^3 + 2912.*y.^5 - 9984.*y.^7 +
16640.*y.^9 - 13312.*y.^11 + 4096.*y.^13) +
((127668687979205667.*x.^4)./21267647932558653966460912964485513216 -
(6079461332343127.*x.^2)./21267647932558653966460912964485513216 -
(127668687979205667.*x.^6)./26584559915698317458076141205606891 52 +
```

4_term_cosine_rand_15x15.txt

```
(8873906961158811.*x.^10)./5444517870735015415413993718908291383296 -
(1882343900851869.*x.^12)./68056473384187692692674921486353642291 2 +
(930829401520155.*x.^14)./34028236692093846346337460743176821145 6 -
(62055293434677.*x.^16)./42535295865117307932921825928971026432 +
(6895032603853.*x.^18)./21267647932558653966460912964485513216 -
6895032603853./27875931498163278926919647840810451882475 52).*(338.*y.^2 - 18928.*y.^4 +
416416.*y.^6 - 4759040.*y.^8 + 32361472.*y.^10 - 141213696.*y.^12 + 412778496.*y.^14 -
825556992.*y.^16 + 1133117440.*y.^18 - 1049624576.*y.^20 + 627048448.*y.^22 -
218103808.*y.^24 + 33554432.*y.^26 - 1) +
((8863354844111763.*x.^2)./17840596158824498513228574618118686892047843328 -
(14772258073519605.*x.^4)./11150372599265311570767859136324180752990208 +
(75830924777400639.*x.^6)./55751862996326557853839295681620903764951 04 -
(97496903285229393.*x.^8)./13937965749081639463459823920405225941237 76 +
(140828860300886901.*x.^10)./69689828745408197317299119602026129706188 8 -
(29872788548672979.*x.^12)./87112285931760246646623899502532662132736 +
(14772258073519605.*x.^14)./43556142965880123323311949751266331066368 -
(984817204901307.*x.^16)./5444517870735015415413993718908291383296 +
(109424133877923.*x.^18)./27222589353675077077069968594541456916 48 -
109424133877923./35681192317648997026457149236237378409568665 6).*(25480.*y.^4 -
392.*y.^2 - 652288.*y.^6 + 8712704.*y.^8 - 69701632.*y.^10 + 361181184.*y.^12 -
1270087680.*y.^14 + 3111714816.*y.^16 - 5369233408.*y.^18 + 6499598336.*y.^20 -
5402263552.*y.^22 + 2936012800.*y.^24 - 939524096.*y.^26 + 134217728.*y.^28 + 1) +
((1775307832739585.*x)./288230376151711744 -
(1775307832739585.*x.^3)./72057594037927936 +
(355061566547917.*x.^5)./18014398509481984).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) +
((5944435031774035.*x)./72057594037927936 - (5944435031774035.*x.^3)./18014398509481984
+ (1188887006354807.*x.^5)./4503599627370496).*(5.*y - 20.*y.^3 + 16.*y.^5) +
((5331193095977445.*x)./144115188075855872 -
(1777064365325815.*x.^3)./36028797018963968).*(11.*y - 220.*y.^3 + 1232.*y.^5 -
2816.*y.^7 + 2816.*y.^9 - 1024.*y.^11) + ((14842241724596229.*x)./144115188075855872 -
(24737069540993715.*x.^3)./18014398509481984 +
(44526725173788687.*x.^5)./9007199254740992 -
(14842241724596229.*x.^7)./2251799813685248 +
(1649137969399581.*x.^9)./562949953421312).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) - (540578779569263.*x.*(3.*y -
4.*y.^3))./72057594037927936 + (3.*y -
4.*y.^3).*((7054086739057981.*x.^4)./72057594037927936 -
(7054086739057981.*x.^2)./72057594037927936 + 7054086739057981./576460752303423488) -
((1805439523700093.*x.^2)./144115188075855872 -
1805439523700093./288230376151711744).*(18.*y.^2 - 48.*y.^4 + 32.*y.^6 - 1) - (98.*y.^2
- 1568.*y.^4 + 9408.*y.^6 - 26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 -
1).*((32967746058252225.*x.^4)./25961484292674138142652481646100 48 -
(999022607825825.*x.^2)./25961484292674138142652481646100 48 -
(6593549211650445.*x.^6)./40564819207303340847894502572032 +
(85716139751455785.*x.^8)./81129638414606681695789005144064 -
(40000865217346033.*x.^10)./10141204801825835211973625643008 +
(90911057312150075.*x.^12)./10141204801825835211973625643008 -
(999022607825825.*x.^14)./79228162514264337593543950336 +
(3396676866607805.*x.^16)./31691265005705735037417580 1344 -
(199804521565165.*x.^18)./39614081257132168796771975168 +
(39960904313033.*x.^20)./39614081257132168796771975168 +
39960904313033./20769187434139310514121985316880384) -
((384558502468497.*x)./18014398509481984 -
(128186167489499.*x.^3)./4503599627370496).*(8.*y.^4 - 8.*y.^2 + 1) -
((52841473958172053.*x.^2)./34028236692093846346337460743176821145 6 -
(52841473958172053.*x.^4)./21267647932558653966460912964485513216 +
(158524421874516159.*x.^6)./10633823966279326983230456482242756608 -
(113231729910368685.*x.^8)./26584559915698317458076141205606891 52 +
(83036601934270369.*x.^10)./13292279957849158729038070602803445 76 -
(7548781994024579.*x.^12)./16615349947311448411297588253504307 2 +
(1078397427717797.*x.^14)./83076749736557242056487941267521536 -
1078397427717797./68056473384187692692674921486353642291 2).*(162.*y.^2 - 4320.*y.^4 +
44352.*y.^6 - 228096.*y.^8 + 658944.*y.^10 - 1118208.*y.^12 + 1105920.*y.^14 -
589824.*y.^16 + 131072.*y.^18 - 1) - ((1575646852315167.*x.^2)./144115188075855872 -
1575646852315167./288230376151711744).*(2.*y.^2 - 1) + (18.*y.^2 - 48.*y.^4 + 32.*y.^6
- 1).*((58994381170394145.*x)./576460752303423488 -
```

4_term_cosine_rand_15x15.txt

```
((2583772977544029.*x.^4)./18014398509481984 -
(2583772977544029.*x.^2)./18014398509481984 +
2583772977544029./144115188075855872).*(8.*y.^4 - 8.*y.^2 + 1) -
((78994568151759825.*x.^2)./5316911983139663491615228241121378304 -
(78994568151759825.*x.^4)./664613997892457936451903530140172288 +
(110592395412463755.*x.^6)./33230699894622896822595176507008614 4 -
(15798913630351965.*x.^8)./41538374868278621028243970633760768 +
(3159782726070393.*x.^10)./20769187434139310514121985316880384 -
3159782726070393./10633823966279326983230456482242756608).*(2688.*y.^4 - 128.*y.^2 -
21504.*y.^6 + 84480.*y.^8 - 180224.*y.^10 + 212992.*y.^12 - 131072.*y.^14 +
32768.*y.^16 + 1) - ((29060673606014247.*x.^2)./72057594037927936 -
(9686891202004749.*x.^4)./9007199254740992 + (3228963734001583.*x.^6)./4503599627370496
- 3228963734001583./144115188075855872).*(11.*y - 220.*y.^3 + 1232.*y.^5 - 2816.*y.^7 +
2816.*y.^9 - 1024.*y.^11) - (50.*y.^2 - 400.*y.^4 + 1120.*y.^6 - 1280.*y.^8 +
512.*y.^10 - 1).*((20341409948986775.*x)./144115188075855872 -
(101707049744933875.*x.^3)./36028797018963968 +
(142389869642907425.*x.^5)./9007199254740992 -
(20341409948986775.*x.^7)./562949953421312 + (20341409948986775.*x.^9)./562949953421312
- (1849219086271525.*x.^11)./140737488355328) -
((93349156342681925.*x.^2)./21267647932558653966460912964485513216 -
(93349156342681925.*x.^4)./265845599156983174580761 4120560689152 +
(130688818879754695.*x.^6)./13292279957849158729038070060280344576 -
(18669831268536385.*x.^8)./16615349947311448411297588253504 3072 +
(3733966253707277.*x.^10)./83076749736557242056487941267521536 -
3733966253707277./42535295865117307932921825928971026432).*(338.*y.^2 - 18928.*y.^4 +
416416.*y.^6 - 4759040.*y.^8 + 32361472.*y.^10 - 141213696.*y.^12 + 412778496.*y.^14 -
825556992.*y.^16 + 1133117440.*y.^18 - 1049624576.*y.^20 + 627048448.*y.^22 -
218103808.*y.^24 + 33554432.*y.^26 - 1) + ((219973490175389.*x.^4)./1125899906842624 -
(219973490175389.*x.^2)./1125899906842624 +
219973490175389./9007199254740992).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 + 6912.*y.^8 -
6144.*y.^10 + 2048.*y.^12 + 1) - ((936161915774427.*x.^2)./144115188075855872 -
936161915774427./288230376151711744).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 + 128.*y.^8 +
1) - y.*((18325843453117035.*x)./72057594037927936 -
(91629217265585175.*x.^3)./18014398509481984 +
(128280904171819245.*x.^5)./4503599627370496 -
(18325843453117035.*x.^7)./281474976710656 + (18325843453117035.*x.^9)./281474976710656
- (1665985768465185.*x.^11)./70368744177664) + (9.*y - 120.*y.^3 + 432.*y.^5 -
576.*y.^7 + 256.*y.^9).*((80357024176939965.*x.^4)./9007199254740992 -
(6887744929451997.*x.^2)./9007199254740992 - (5357134945129331.*x.^6)./140737488355328
+ (20663234788355991.*x.^8)./281474976710656 -
(2295914976483999.*x.^10)./35184372088832 + (765304992161333.*x.^12)./35184372088832 +
765304992161333./72057594037927936) + ((2221544235015819.*x)./18014398509481984 -
(2221544235015819.*x.^3)./2251799813685248 + (2221544235015819.*x.^5)./1125899906842624
- (317363462145117.*x.^7)./281474976710656).*(840.*y.^4 - 72.*y.^2 - 3584.*y.^6 +
6912.*y.^8 - 6144.*y.^10 + 2048.*y.^12 + 1) -
((13936134893078493.*x.^2)./72057594037927936 -
(4645378297692831.*x.^4)./9007199254740992 + (1548459432564277.*x.^6)./4503599627370496
- 1548459432564277./144115188075855872).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) + (9.*y - 120.*y.^3 +
432.*y.^5 - 576.*y.^7 + 256.*y.^9).*((35483950863425855.*x.^2)./18014398509481984 -
(35483950863425855.*x.^4)./1125899906842624 +
(106451852590277565.*x.^6)./562949953421312 -
(76037037564483975.*x.^8)./140737488355328 + (55760494213954915.*x.^10)./70368744177664
- (5069135837632265.*x.^12)./8796093022208 + (724162262518895.*x.^14)./4398046511104 -
724162262518895./36028797018963968) - ((5199389315207145.*x)./288230376151711744 -
(1733129771735715.*x.^3)./72057594037927936).*(98.*y.^2 - 1568.*y.^4 + 9408.*y.^6 -
26880.*y.^8 + 39424.*y.^10 - 28672.*y.^12 + 8192.*y.^14 - 1) -
((3743819539070925.*x)./72057594037927936 -
(1247939846356975.*x.^3)./18014398509481984).*(160.*y.^4 - 32.*y.^2 - 256.*y.^6 +
128.*y.^8 + 1) + (5.*y - 20.*y.^3 +
16.*y.^5).*((303555943583881695.*x.^4)./18014398509481984 -
(26019080878618431.*x.^2)./18014398509481984 -
(20237062905592113.*x.^6)./281474976710656 + (78057242635855293.*x.^8)./562949953421312
- (8673026959539477.*x.^10)./70368744177664 + (2891008986513159.*x.^12)./70368744177664
+ 2891008986513159./144115188075855872) - (8.*y.^4 - 8.*y.^2 +
1).*((14663016286434327.*x.^4)./6338253001141147007 48351602688 -
(698238870782587.*x.^2)./6338253001141147007 48351602688 -
```