XXI IMEKO World Congress ”Measurement in Research and Industry”
August 30 - September 4, 2015, Prague, Czech Republic

GENERATION OF REFERENCE DATA FOR SOFTWARE VALIDATION

A B Forbes [1], *I M Smith* [2], *G J P Kok* [3]

[1] National Physical Laboratory, Teddington, United Kingdom, alistair.forbes@npl.co.uk
[2] National Physical Laboratory, Teddington, United Kingdom, ian.smith@npl.co.uk
[3] Van Swinden Laboratorium, Delft, The Netherlands, gkok@vsl.nl

Abstract - A common approach in the validation of metrology software involves the use of reference data comprising reference input data and reference output data. Software to be validated is applied to the reference input data to generate test output data that is compared, in an appropriate way, with the reference output data. This paper considers reference data generation in the context of a Joint Research Project currently being undertaken under the European Metrology Research Programme.

Keywords: reference data generation, software validation

1. INTRODUCTION

Within metrology, the use of software to undertake computation is widespread and it is imperative that such software can be shown to be operating correctly. The European Metrology Research Programme (EMRP) [1] funded the Joint Research Project (JRP) NEW06 “Traceability for computationally-intensive metrology” (referred to as “TraCIM”) [2]. The JRP was concerned with meeting the requirement to establish traceability of metrology software at the point of use and a key output is an information and communications technology infrastructure that allows online software validation.

In order to undertake software validation, a number of steps, of which reference data generation is one, must be undertaken. Section 2 begins by summarising those steps, while Section 3 describes how reference data may be generated, including details of how to provide statements of “numerical uncertainty” to accompany reference data and which are used to assess software under test. Although the focus of the paper is on data generation, choices of performance metrics are briefly discussed in Section 4. Section 5 considers the application of data generation techniques to polynomial regression before conclusions are given in Section 6.

2. STEPS IN SOFTWARE VALIDATION

Fig. 1 captures the main steps that must be undertaken to validate software:

- Write a clear, unambiguous statement of the computational aim to be addressed by the software. The statement acts as both the user and functional requirements for the software developer, and provides a basis for verification and validation of the software implementation. Its provision is therefore fundamental for all subsequent steps. Within the TraCIM project, a database [3] has been created that allows computational aims to be specified and stored.

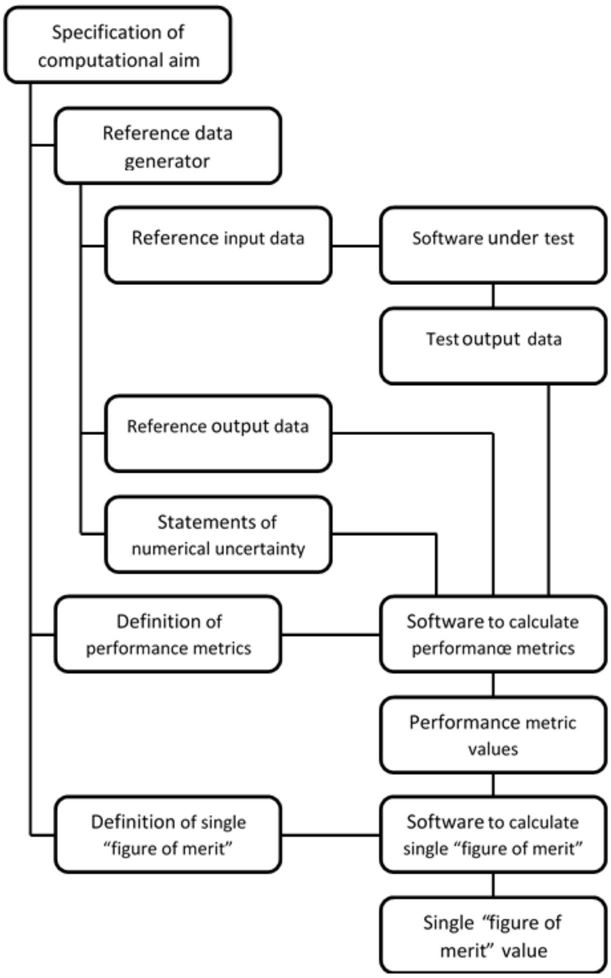
Specification of computational aim
Reference data generator
Reference input data
Software under test
Test output data
Reference output data
Statements of numerical uncertainty
Definition of performance metrics
Software to calculate performance metrics
Performance metric values
Definition of single “figure of merit”
Software to calculate single “figure of merit”
Single “figure of merit” value

Fig. 1. Steps in software validation.

- Develop software to generate reference data, also referred to as “reference pairs” and comprising reference input data and reference output data. Each reference pair must be accompanied by a statement of its “numerical uncertainty” that reflects the fact that it is generally not possible for the reference output data to be the exact solution to the specified computational aim for the corresponding reference input data. The reference input data is to be processed by software under test to obtain test output data that is compared, in an appropriate way, with the reference output data. Section 3 provides more details on the generation of reference data and the provision of statements of numerical uncertainty.

- Define performance metrics that allow test output data and reference output data to be compared. The performance metrics must take into account the numerical uncertainty statement that accompanies each reference pair. Performance metrics are discussed in Section 4.
- Develop software to evaluate performance metrics.
- Define a single "figure of merit" that combines the values of the performance metrics obtained by applying test software to a (suitably large) number of reference pairs.
- Develop software to evaluate the single figure of merit.
- For each reference pair, apply the software under test to the reference input data to obtain test output data, and evaluate the performance metric(s) for that reference pair.
- Evaluate the single figure of merit to provide a numerical assessment of the software under test.

Note that the definition of a single figure of merit is not considered in this paper.

3. REFERENCE DATA GENERATION

Reference data generation (RDG) is a key step in the assessment of software. An appropriate number of reference pairs must be generated to ensure (a) that the full range of problems intended to be solved by the software is adequately covered, and (b) confidence that a fair assessment of the software can be provided.

For many computational aims, approaches for reference data generation are well known. RDG is typically implemented in one of the following ways:

- Forward RDG, in which reference input data is taken and used to produce corresponding reference output data.
- Reverse RDG, in which reference output data is taken and used to produce corresponding reference input data.

Forward RDG typically requires the availability of reference software that processes reference input data to produce reference output data. Whereas in the past this process could be both complicated and costly, nowadays forward RDG may be more easily implemented, e.g., through the use of software that supports extended precision arithmetic such as [4, 5].

Reverse RDG generally requires an analysis of the computational aim to be undertaken so that reference output data can be processed to obtain reference input data, and in some cases is much more simple to implement than forward RDG.

3.1. Numerical uncertainty

In order to obtain an assessment of the numerical performance of software intended to implement a specified computational aim, the provision of reference pairs alone is insufficient. For each reference pair, additional information, in the form of a statement of "numerical uncertainty", must be provided. The numerical uncertainty associated with a reference result, i.e., a single element of a reference output data set, is expressed in two ways:

- By the accuracy of the reference result (see Section 3.2).
- By the sensitivity of the reference result to perturbations in the reference input data (see Section 3.3).

The statements of numerical uncertainty that accompany a reference pair capture the fact that in finite precision it is generally not possible for the reference input data and reference output data to satisfy exactly the required mathematical relationship. These statements of numerical uncertainty should be taken into account when evaluating metrics to evaluate the performance of software under test.

Let $\boldsymbol{b}$ and $\boldsymbol{t}$ represent sets of input and output data, respectively. In practice, $\boldsymbol{b}$ and $\boldsymbol{t}$ are formed by aggregating a number of elements having different properties, e.g., real and integer values, vector and matrices, etc.

Let $\hat{t}_k = t_k^{\mathrm{f}}(\boldsymbol{b})$ represents a value for the kth element of output data, e.g., as might be returned by a reference or test software implementation of the computational aim. The closeness of agreement of the value with the corresponding reference value t_k is given by

$$d(t_k, \hat{t}_k) = t_k - \hat{t}_k.$$

It is assumed that:

- The variables defining the reference data and the reference results are stored as floating-point numbers to a defined working precision (usually double precision with a floating-point relative accuracy of $\epsilon_{\mathrm{w}} = 2^{-52}$).
- A function $\rho_{\mathrm{e}}[.]$ is available to convert floating-point numbers stored in the working precision to floating-point numbers stored in extended precision.
- A function $\rho_{\mathrm{w}}[.]$ is available to convert floating-point numbers stored in extended precision to floating-point numbers stored in the working precision.
- An extended precision software implementation of the computational aim is available represented by the functions
$$t_k^{\mathrm{f}}, k = 1, \ldots, N,$$

where N is the total number of elements of output data. This software implementation is considered to behave as a reference software implementation of the computational aim.

3.2. Accuracy

The accuracy of a reference result is expressed by its closeness of agreement with a result obtained using the extended precision arithmetic software implementation of the computational aim. The accuracy $A(t_k)$ of t_k is given by

$$A(t_k) = |d(t_k, \hat{t}_k)|,$$

where

$$\hat{t}_k = \rho_{\mathrm{w}}[t_k(\rho_{\mathrm{e}}[\boldsymbol{b}])],$$

i.e., $\hat{t}_k$ is obtained by converting reference input data $\boldsymbol{b}$ to extended precision, applying the extended precision software implementation of the computational aim, and converting the result to the working precision. $A(t_k)$ should be independent of the number of digits used in the extended precision calculations.

3.3. Sensitivity

The sensitivity of each reference result is expressed in terms of the sensitivity coefficient measuring the variation in the result due to variations in the input data values. Calculations of the sensitivity coefficients are implemented using the extended precision software implementation of the computational aim. The sensitivity coefficient is determined using a Monte Carlo calculation in which random perturbations are applied to the reference data [6]. Specifically, given a number M of Monte Carlo trials and an integer m to control the size of the relative perturbations that are applied, with $1 \leq m \leq 1 + |\log_{10} \epsilon_{\mathrm{w}}|/2$, the sensitivities $S(t_k)$, $k = 1, \ldots, N$, are determined in the following steps:

1. Set $\delta = 10^m \epsilon_{\mathrm{w}}$.
2. For $q = 1, \ldots, M$:
   - Form the vector $\boldsymbol{e}_b$ with elements $e_{b,j} = z_{b,j}\delta$, where the $z_{b,j}$ are random draws from the rectangular distribution $\mathrm{R}(-1, 1)$;
   - Calculate $\boldsymbol{b}_q = \boldsymbol{b} + \boldsymbol{e}_b$.
   - Calculate

     $$\alpha_{kq} = d(t_k, \widehat{t}_{kq}), \; k = 1, \ldots, N,$$

     where $\widehat{t}_{kq} = \rho_{\mathrm{w}}\left[t_k^{\mathrm{f}}\left(\rho_{\mathrm{e}}[\boldsymbol{b}_q]\right)\right]$.
3. Calculate, for $k = 1, \ldots, N$,

   $$S(t_k) = \max\left\{1, \frac{\frac{1}{M-1}\sum_{q=1}^{M}(\alpha_{kq} - \alpha_{k\cdot})^2}{\delta/\sqrt{3}}\right\},$$

   where

   $$\alpha_{k\cdot} = \frac{1}{M}\sum_{q=1}^{M}\alpha_{kq}.$$

Provided the floating-point relative accuracy ϵ_{e} for the extended precision software implementation of the computational aim is sufficiently small compared with ϵ_{w} (for example, $\epsilon_{\mathrm{e}} < \epsilon_{\mathrm{w}}^2$), the values calculated for the sensitivities can be expected to provide information about the computational aim, viz., its "condition", rather than about the algorithm and software used to undertake the computational aim, viz., their "stability". It is recommended that the above procedure is applied for integers m satisfying $1 \leq m \leq 4$, say, in order to determine whether there is any dependence of the sensitivity coefficients on the relative size of the perturbations applied to the reference data. It is also recommended that the above procedure is applied for a number M of trials satisfying $M \geq 10^4$ in order to ensure the reliability of the estimates of the sensitivity coefficients is adequate.

4. PERFORMANCE METRICS

It is important that performance metrics take into account the statement of numerical uncertainty associated with the reference results.

Let t_k^{t}, $k = 1, \ldots, N$, be the results returned by a test software implementation of the computational aim operating with floating-point relative accuracy ϵ_{w}. The test results returned may be compared with corresponding reference results as follows.

- Calculate

  $$D(t_k^{\mathrm{t}}) = |d(t_k, t_k^{\mathrm{t}})|, \; k = 1, \ldots, N.$$

  Then calculate

  $$D_A(t_k^{\mathrm{t}}) = \max\left\{D(t_k^{\mathrm{t}}), A(t_k)\right\}, \; k = 1, \ldots, N,$$

  which accounts for the numerical accuracy associated with the reference results.
- Calculate

  $$P(t_k^{\mathrm{t}}) = \log_{10}\left(1 + \frac{D_A(t_k^{\mathrm{t}})}{S(t_k)\epsilon_{\mathrm{w}}}\right), \; k = 1, \ldots, N,$$

  which accounts for the sensitivity of the reference results.

For a test result t_k^{t} for which the reference result is t_k, the performance metric $P(t_k^{\mathrm{t}})$ measures the number of decimal digits of accuracy lost by the test software *in addition to* the number that would be expected to be lost by reference software for the computational aim operating with the same floating-point relative accuracy. Here, the number of decimal digits of accuracy that would be expected to be lost by reference software is measured by the sensitivity of the reference result.

For example, suppose

$$A(t_k) = 10^a \epsilon_{\mathrm{w}}, \; S(t_k) = 10^b, \; D(t_k^{\mathrm{t}}) = 10^c \epsilon_{\mathrm{w}}.$$

Consider first the case $c \geq a$, i.e., the reference result has more significant digits of accuracy than the test result. Then,

$$P(t_k^{\mathrm{t}}) = \log_{10}\left(1 + \frac{10^c \epsilon_{\mathrm{w}}}{10^b \epsilon_{\mathrm{w}}}\right) = \log_{10}\left(1 + 10^{c-b}\right),$$

and:

- $P(t_k^{\mathrm{t}}) \approx 0$ when $c < b$, or
- $P(t_k^{\mathrm{t}}) \approx c - b$ when $c \geq b$.

In the alternative case when $c < a$, i.e., the numerical accuracy of the reference result is not adequate to distinguish numerically between the test and reference results, then

$$P(t_k^{\mathrm{t}}) = \log_{10}\left(1 + \frac{10^a \epsilon_{\mathrm{w}}}{10^b \epsilon_{\mathrm{w}}}\right) = \log_{10}\left(1 + 10^{a-b}\right),$$

and the performance of the test software is measured by the numerical accuracy of the reference result.

5. EXAMPLE

5.1. Polynomial regression

Regression problems occur frequently in many metrology disciplines. A common problem involves determining estimates of the parameters defining the best-fit calibration function to (x, y) data, accounting for uncertainty associated with the measured y-values. The problem is referred to as one of "weighted least squares (WLS) polynomial regression".

5.2. Statement of problem

Given measured x-values $\boldsymbol{x} = (x_1, \ldots, x_m)^\top$, measured y-values $\boldsymbol{y} = (y_1, \ldots, y_m)^\top$, standard uncertainties $\boldsymbol{u_y} = (u(y_1), \ldots, u(y_m))^\top$ associated with $\boldsymbol{y}$, and polynomial degree n:

- Calculate estimates $\boldsymbol{a} = (a_1, \ldots, a_{n+1})^\top$ of the polynomial coefficients $\boldsymbol{A} = (A_1, \ldots, A_{n+1})^\top$ that minimise
$$\sum_{i=1}^{m}\left[\frac{y_i - p(\boldsymbol{A}, x_i)}{u(y_i)}\right]^2,$$
where
$$p(\boldsymbol{A}, x) = \sum_{j=1}^{n+1} A_j x^{j-1}.$$
- Calculate the covariance matrix $V_{\boldsymbol{a}}$ associated with $\boldsymbol{a}$ by applying the (generalised) law of propagation of uncertainty [7].
- Calculate values $\boldsymbol{r} = (r_1, \ldots, r_m)^\top$ of the weighted residuals $\boldsymbol{R} = (R_1, \ldots, R_m)^\top$, where
$$r_i = \frac{y_i - p(\boldsymbol{a}, x_i)}{u(y_i)}.$$
- Calculate the covariance matrix $V_{\boldsymbol{r}}$ associated with $\boldsymbol{r}$ by applying the (generalised) law of propagation of uncertainty [7].
- Calculate values $\boldsymbol{p} = (p_1, \ldots, p_m)^\top$ of the best-fit polynomial function $\boldsymbol{P} = (P_1, \ldots, P_m)^\top$ at the x-coordinates of the data points, where
$$p_i = p(\boldsymbol{a}, x_i).$$
- Calculate the covariance matrix $V_{\boldsymbol{p}}$ associated with $\boldsymbol{p}$ by applying the (generalised) law of propagation of uncertainty [7].

5.3. Analysis of computational aim

The problem of WLS polynomial regression belongs to the class of problems referred to as "linear least squares fitting" whose properties are well-known [8].

Let C be the (observation) matrix of dimension $m \times (n+1)$ given by

$$C = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^n \\ 1 & x_2 & x_2^2 & \ldots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^n \end{bmatrix},$$

and

$$\boldsymbol{f}(\boldsymbol{A}) = \boldsymbol{y} - C\boldsymbol{A}.$$

The problem is to determine estimates $\boldsymbol{a}$ of $\boldsymbol{A}$ that minimise

$$F(\boldsymbol{A}) = \boldsymbol{f}^\top(\boldsymbol{A}) V_{\boldsymbol{y}}^{-1} \boldsymbol{f}(\boldsymbol{A}), \qquad (1)$$

where

$$V_{\boldsymbol{y}} = \mathrm{diag}(u^2(y_1), \ldots, u^2(y_m)).$$

Expression (1) can be rewritten

$$F(\boldsymbol{A}) = \left(L_{\boldsymbol{y}}^{-1}\boldsymbol{f}(\boldsymbol{A})\right)^\top \left(L_{\boldsymbol{y}}^{-1}\boldsymbol{f}(\boldsymbol{A})\right) = \tilde{\boldsymbol{f}}^\top(\boldsymbol{A})\tilde{\boldsymbol{f}}(\boldsymbol{A}),$$

where

$$L_{\boldsymbol{y}} = \mathrm{diag}(u(y_1), \ldots, u(y_m))$$

is the Cholesky factor of $V_{\boldsymbol{y}}$ [9], i.e., $V_{\boldsymbol{y}} = L_{\boldsymbol{y}} L_{\boldsymbol{y}}^\top$.

At the solution,

$$\left.\frac{\partial F}{\partial A_j}\right|_{\boldsymbol{A}=\boldsymbol{a}} = 0, \quad j = 1, \ldots, n+1,$$

which gives rise to the system of linear equations of order $n+1$,

$$D^\top D\boldsymbol{a} = D^\top L_{\boldsymbol{y}}^{-1}\boldsymbol{y},$$

where $D = L_{\boldsymbol{y}}^{-1}C$. The system of linear equations is referred to as the *normal equations*.

If D is full rank, so that $D^\top D$ is invertible, the estimates $\boldsymbol{a}$ can then be expressed (mathematically) as a linear function of the measured y-values:

$$\boldsymbol{a} = \left(D^\top D\right)^{-1} D^\top L_{\boldsymbol{y}}^{-1}\boldsymbol{y} = D^\dagger L_{\boldsymbol{y}}^{-1}\boldsymbol{y},$$

where $D^\dagger = \left(D^\top D\right)^{-1} D^\top$ is the *pseudo-inverse* of D and satisfies

$$DD^\dagger D = D, \quad D^\dagger D D^\dagger = D^\dagger, \quad D^\dagger \left(D^\dagger\right)^\top = \left(D^\top D\right)^{-1}.$$

In practice, matrix inversion is avoided for reasons of numerical accuracy, and an alternative approach is used to solve the system of equations. One such approach involves QR factorisation [9] as follows. Let D have QR factorisation

$$D = Q_{12} R_{12} = [Q_1 \;\; Q_2] \left[\begin{array}{c} R_1 \\ 0 \end{array} \right],$$

where Q_{12} is orthogonal, R_{12} is upper-triangular, Q_1 and Q_2 comprise the first $(n+1)$ and last $m-(n+1)$ columns of Q_{12}, and R_1 comprises the first $(n+1)$ rows of R_{12}. Let $\boldsymbol{q}_1$ be the vector containing the first $(n+1)$ elements of $Q_{12}^\top \left(L_{\boldsymbol{y}}^{-1}\boldsymbol{y}\right)$, i.e.,

$$\boldsymbol{q}_1 = Q_1^\top \left(L_{\boldsymbol{y}}^{-1}\boldsymbol{y}\right).$$

The estimates $\boldsymbol{a}$ are the solution of the upper-triangular system

$$R_1 \boldsymbol{a} = \boldsymbol{q}_1,$$

i.e.,

$$\boldsymbol{a} = R_1^{-1} Q_1^\top \left(L_{\boldsymbol{y}}^{-1}\boldsymbol{y}\right) = A\boldsymbol{y},$$

where $A = R_1^{-1} Q_1^\top L_{\boldsymbol{y}}^{-1}$.

The covariance matrix $V_{\boldsymbol{a}}$ associated with $\boldsymbol{a}$ is given by

$$V_{\boldsymbol{a}} = A V_{\boldsymbol{y}} A^\top = R_1^{-1} R_1^{-\top}.$$

The values $\boldsymbol{r}$ of the weighted residuals $\boldsymbol{R}$ are given by

$$\boldsymbol{r} = L_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - C\boldsymbol{a}) = L_{\boldsymbol{y}}^{-1}\left(I - CA\right)\boldsymbol{y} = R\boldsymbol{y},$$

where $R = L_{\boldsymbol{y}}^{-1}\left(I - CA\right)$.

The covariance matrix $V_{\boldsymbol{r}}$ associated with $\boldsymbol{r}$ is given by

$$V_{\boldsymbol{r}} = R V_{\boldsymbol{y}} R^\top.$$

The values $\boldsymbol{p}$ of the best-fit polynomial function $\boldsymbol{P}$ are given by

$$\boldsymbol{p} = C\boldsymbol{a} = CA\boldsymbol{y} = P\boldsymbol{y},$$

where $P = CA$.

The covariance matrix $V_{\boldsymbol{p}}$ associated with $\boldsymbol{p}$ is given by

$$V_{\boldsymbol{p}} = P V_{\boldsymbol{y}} P^\top.$$

5.4. Reference data generation

A common approach used to generate reference data for least squares regression problems is the nullspace method [8]. The method implements reverse RDG.

Generation of a reference pair requires a number of choices to be made or values to be assigned, including:

- The seed for random number generation.
- The number of measured data points.
- The polynomial degree.
- The nature of the spacing of x-values, e.g., evenly spaced, randomly spaced.
- The polynomial coefficient values.
- The standard uncertainties associated with the y-values.

Accuracy and sensitivity values can be calculated for each element of vectors $\boldsymbol{a}$, $\boldsymbol{r}$ and $\boldsymbol{p}$, and of matrices $V_{\boldsymbol{a}}$, $V_{\boldsymbol{r}}$ and $V_{\boldsymbol{p}}$. The calculations are undertaken using an extended precision (number of digits set to 100) software implementation of the computational aim.

Fig. 2 plots the reference input data of two example reference pairs. In each plot, for each point i, the simulated measured values (x_i, y_i) is denoted by a cross, while the standard uncertainty $u(y_i)$ associated with y_i is denoted by a straight line between $(x_i, y_i - u(y_i))$ and $(x_i, y_i + u(y_i))$. The upper plot shows five points corresponding to a value of $n = 1$, i.e., a straight line, while the lower plot shows twenty points corresponding to $n = 5$. In both cases, the x-values are randomly spaced.

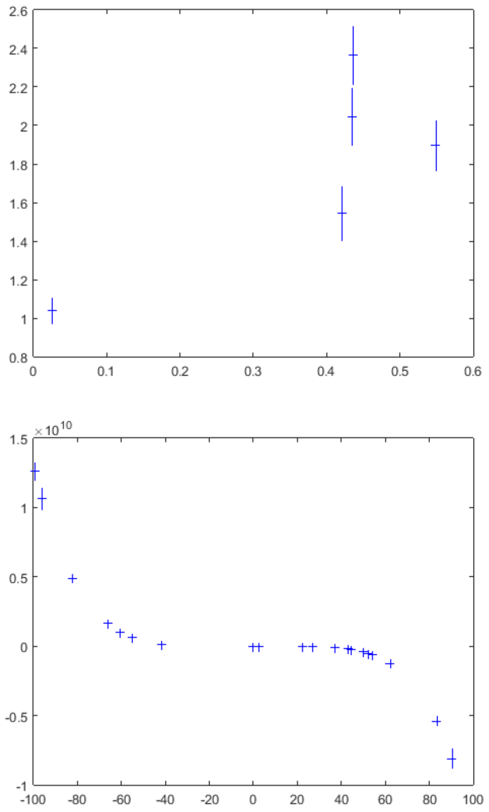

Fig. 2. Example reference input data for polynomial degrees 1 (upper plot) and 5 (lower plot).

5.5. Performance metrics

Performance metrics are calculated for each element of vectors $\boldsymbol{a}$, $\boldsymbol{r}$ and $\boldsymbol{p}$, and of matrices $V_{\boldsymbol{a}}$, $V_{\boldsymbol{r}}$ and $V_{\boldsymbol{p}}$ by applying two software implementations of WLS polynomial regression:

- Software 'S1', which undertakes all calculations in double precision.
- Software 'S2', which converts the reference input data to extended precision (number of digits set to 30), undertakes calculations in extended precision, and converts the test results from extended precision to double precision.

Table 1 shows the maximum value of the performance metric $P(t_k^{\mathrm{t}})$ obtained for $\boldsymbol{a}$, $V_{\boldsymbol{a}}$, $\boldsymbol{r}$, $V_{\boldsymbol{r}}$, $\boldsymbol{p}$ and $V_{\boldsymbol{p}}$ for the reference pairs introduced in Section 5.4. The metric compares the numerical performance of test software against reference software, whose numerical performance is described by the sensitivities of the reference results to perturbations in the reference data.

	Reference pair 1		**Reference pair 2**	
	S1	**S2**	**S1**	**S2**
$\boldsymbol{a}$	0.4	0.2	0.4	0.1
$V_{\boldsymbol{a}}$	0.1	0.0	2.4	0.2
$\boldsymbol{r}$	0.8	0.1	4.2	0.1
$V_{\boldsymbol{r}}$	0.3	0.2	3.4	0.2
$\boldsymbol{p}$	0.2	0.0	0.2	0.1
$V_{\boldsymbol{p}}$	0.0	0.0	2.6	0.0

Table 1. Summary of performance metrics for the reference pairs introduced in Section 5.4.

For $n = 1$, the results suggest that software S1 and S2 both perform close to reference software for the computational aim. For $n = 5$, however, the numerical performance of S2 is better than that of software S1, and only software S2 performs close to reference software.

6. CONCLUSIONS

Reference data generation is a key step when assessing software intended to implement a specified computational aim. While the data generation process returns reference input data and reference output data, additional information, in the form of numerical uncertainty information, is required to undertake a quantitative assessment of software. The paper describes how such information can be provided and incorporated into performance metrics that can be evaluated for software assessment. The presented ideas are applied to the case of weighted least squares polynomial regression.

ACKNOWLEDGMENTS

This work has been undertaken as part of the EMRP Joint Research Project NEW06 "Traceability for computationally-intensive metrology", co-funded by the UK's National Measurement Office Programme for Materials and Modelling and the European Union. The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union.

REFERENCES

[1] European Metrology Research Programme (EMRP). www.emrponline.eu.

[2] TraCIM website. www.ptb.de/emrp/1390.html.

[3] TraCIM Computational Aims Database. www.tracim-cadb.npl.co.uk/.

[4] Mathematica 10.2, Wolfram Research, Inc.

[5] ADVANPIX Multiprecision Computing Toolbox for MATLAB. www.advanpix.com.

[6] M. G. Cox, P. M. Harris and I. M. Smith, "Software specifications for uncertainty evaluation", National Physical Laboratory Technical Report MS 7, 2010.

[7] Joint Committee for Guides in Metrology, JCGM:102:2011 Evaluation of measurement data – Supplement 2 to the "Guide to the expression of uncertainty in measurement" – Extension to any number of output quantities.

[8] R. M. Barker, M. G. Cox, A. B. Forbes, P. M. Harris, Software Support for Metrology Best Practice Guide No. 4: Discrete Modelling and Experimental Data Analysis, National Physical Laboratory Technical Report DEM-ES 018, 2007.

[9] G. H. Golub and C. F. Van Loan, "Matrix Computations", John Hopkins University Press, Baltimore, third edition, 1996.