National Physical Laboratory

Report

NPL REPORT
DEM-ES 016

Testing functions for calculating the discrete Fourier transform and its inverse

K J Lines, F O Onakunle and I M Smith

NOT RESTRICTED

March 2007

dti
FUNDED BY THE DTI

National Physical Laboratory | Hampton Road | Teddington | Middlesex | United Kingdom | TW11 0LW

Switchboard 020 8977 3222 | **NPL Helpline** 020 8943 6880 | **Fax** 020 8943 6458 | **www.npl.co.uk**

Testing functions for calculating the discrete Fourier transform and its inverse

K J Lines, F O Onakunle and I M Smith
Division of Enabling Metrology

ABSTRACT

This report describes the application of a general methodology for testing the numerical correctness of scientific software to functions for the calculations of the discrete Fourier transform and its inverse. The functions tested are taken from a number of proprietary software packages and libraries.

This report constitutes one of the deliverables of Project 3.2 *Numerical Software Testing* within the UK Department of Industry's National Measurement System *Software Support for Metrology* Programme 2004–2007.

ISSN 1744-0475

National Physical Laboratory
Hampton Road, Teddington, Middlesex, TW11 0LW

We gratefully acknowledge the financial support of the UK Department of Trade and Industry (National Measurement System Policy Unit)

Approved on behalf of the Managing Director, NPL
by Jonathan Williams, Knowledge Leader of the Electrical and Software Team

Contents

1 Introduction

The work described here constitutes a continuation of work undertaken as part of the first and second Software Support for Metrology (SS*f*M) programmes concerned with testing the numerical correctness of software for computations identified as important to metrology and provided as components within proprietary software packages and libraries. It builds upon previous work [1, 2, 3, 4, 5] by testing additional calculations implemented by recent releases of some of the software packages previously tested.

In the first SS*f*M programme, a general methodology for testing the correctness of scientific software was promoted. The methodology is based on the design and use of reference data sets and corresponding reference results to undertake "black box" testing of the software, together with the use of quality metrics and performance measures to make objective comparisons between reference results and the results returned by the software under test for the reference data sets.

In the second SS*f*M programme, a web-based facility was provided to enable users to generate, for a number of key computations, reference data sets and corresponding reference results appropriate to their own applications. The facility takes the form of data generators implemented in Java in such a way as to provide portability of the generators (across computer platforms), as well as flexibility and reproducibility of the reference data sets generated. The facility was further developed in the third SS*f*M programme to provide the functionality of a testing service. The facility has two modes of operation. In the first mode, the user is provided with reference data sets with corresponding reference results. In the second mode, the user is provided only with reference data sets, but may upload test results to be compared with the corresponding reference results (which are hidden from the user).

In the third (and current) SS*f*M programme, testing has been carried out on software for the following computations:

- Calculation of sample (arithmetic) mean and standard deviation,
- Straight-line (ordinary) regression, and
- Polynomial (ordinary) regression.

This report describes the application of the methodology to software for the following additional computations:

- Discrete Fourier transform, and
- Inverse discrete Fourier transform.

Reference pairs, comprising reference data sets and corresponding reference results, have been generated using Matlab as reference software. Results are presented for recent releases of the Microsoft Excel and LabVIEW software packages; Origin, a spreadsheet-based program for data analysis and presentation; Mathcad, a calculation package widely used in metrology and engineering; and the NAG and IMSL Fortran libraries.

The report is organised as follows. Section 2 gives an overview of the methodology employed for evaluating the numerical correctness of the results produced by scientific software. Section 3 provides a specification of the computations that are the subject of this report, as well as listing the particular software (functions or modules) for undertaking these computations that are tested. Section 4 considers issues of implementation arising from the application of the methodology for the variety of software languages and packages covered. For each computation, sections 5, 6 and 7 provide, respectively, specifications of the performance parameters used to define the reference data sets, the performance measures used to compare reference and test results, and the procedures used to generate reference data sets and corresponding reference results. Section 8 presents, and provides an interpretation of, the results of the testing while section 9 contains conclusions.

2 Methodology

The procedure employed in evaluating the test software[1] is described in this section. A general methodology for evaluating the numerical correctness of the results produced by scientific software has been used. The basis of the approach is the design and use of *reference data sets* and corresponding *reference results* to undertake "black box" testing of the software. The reference data sets and results are generated in a manner that is consistent with the functional specification of the test software, and the results returned by the test software for the reference data sets are then compared objectively with the reference results using quality metrics or performance measures. Finally, the performance measures are interpreted in order to decide whether the test software meets the requirements and is fit for its intended purpose. The methodology is described in several reports and papers [1, 6, 7, 8, 9], and is illustrated by a case study [10].

The testing procedure is divided into the following stages:

1. Documenting the specification of the test software,
2. Specifying reference data sets,
3. Specifying performance measures and testing requirements,
4. Generating reference pairs, i.e., reference data and corresponding reference results,
5. Interfacing to the test software and providing test results, and
6. Determining, presenting and interpreting values of the performance measures.

These stages are based upon previous recommendations [1] but modified to suit testing from the perspective of a *user* of proprietary software packages and libraries. During the software development process, the first stage would be *specifying the test software*, and the fifth *implementing the test software and providing test results*, though in practice these are carried out with varying degrees of formality. The application of the procedure by a software developer is presented in a case study [10]. Recording the results of the first stage is important because it serves to define the basis of the testing undertaken, and to make clear any assumptions made about the test software.

Note that the testing process involves the use of software in addition to the test software itself, for example, for generating reference data sets and corresponding reference results, and evaluating and presenting quality metrics and performance measures. Therefore, the correctness of the results of the testing (and conclusions inferred from the results) depends on the quality (correctness) of this additional software.

[1] The term *test software* used throughout this report refers to the software under test, and not the software employed to do the testing.

3 Specification of Test Software

This section provides specifications for the computations that are the subject of this report, viz.,

1. Discrete Fourier transform, and
2. Inverse discrete Fourier transform.

It also lists the software (functions or modules) for undertaking these computations that have been tested. Matlab [11] is used as reference software for the generation of reference data and corresponding reference results for the calculations of both the discrete Fourier transform and the inverse discrete Fourier transform (section 6).

The Fourier transform is a widely-used tool that defines a relationship between a signal $h(t)$ in the time (or space) domain and a representation $Y(f)$ of the signal in the frequency domain. The essence of the Fourier transform of a signal is to decompose the signal into a sum of sinusoids of different frequencies. The Fourier transform is defined by [12, 13, 14]

$$Y(f) = \int_{-\infty}^{+\infty} h(t) e^{-2\pi jft} \, \mathrm{d}t,$$

and the inverse transform by

$$h(t) = \int_{-\infty}^{+\infty} Y(f) e^{2\pi jft} \, \mathrm{d}f,$$

where $j = \sqrt{-1}$. These relationships show that $h(t)$ and $Y(f)$ may be considered to be a (transform) pair, and as two different representations of the same function, one based in the time domain and the other in the frequency domain. In general, the Fourier transform $Y(f)$ is a complex quantity, and is expressed either in terms of Cartesian (real and imaginary) or polar (magnitude and phase angle) components.

In many applications of the Fourier transform, the signal $h(t)$ is sampled, i.e., the value of the signal is recorded, at uniformly spaced intervals in time (or space). The discrete Fourier transform (DFT) provides estimates of the Fourier transform of $h(t)$ at discrete values of frequency, with the estimates calculated in terms of a finite number of sampled values of the signal.

Let the vector

$$\boldsymbol{h} = \begin{bmatrix} h_1 \\ \vdots \\ h_m \end{bmatrix}$$

of length m contain values $h_l = h(t_l)$ of $h(t)$ sampled at $t_l = (l-1)\Delta$, $l = 1, \ldots, m$, where Δ is the sampling interval. Then, the DFT of $\boldsymbol{h}$ is a vector

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

of Fourier coefficients defined by

$$y_k = \sum_{l=1}^{m} h_l e^{-2\pi j(l-1)(k-1)/m}, \quad k = 1, \ldots, m.$$

The relationship between the Fourier transform $Y(f)$ of $h(t)$ and the DFT $\boldsymbol{y}$ of $\boldsymbol{h}$ is given by

$$Y(f_k) \approx \Delta y_k,$$

where f_k are the discrete values of frequency

$$f_k = \frac{k}{m\Delta}, \quad k = -\frac{m}{2}, \ldots, \frac{m}{2}.$$

The inverse discrete Fourier transform (IDFT) provides a means of recovering the sampled values $\boldsymbol{h}$ from the DFT $\boldsymbol{y}$. The formula for the IDFT is

$$h_k = \frac{1}{m} \sum_{l=1}^{m} y_l e^{2\pi j(l-1)(k-1)/m}, \ k = 1, \ldots, m,$$

and differs from the formula for the DFT only in terms of the sign of the exponential and the division by m.

The problem of evaluating the DFT (or IDFT) can be expressed as a matrix-vector multiplication. However, the algorithm of choice is the fast Fourier transform (FFT) [12], which significantly reduces the number of arithmetic operations necessary to perform the evaluations. Most (if not all) of the software tested in this work implements the FFT algorithm for evaluating the DFT and IDFT. Note that the aim of this work is to test the numerical correctness of software implementations of the DFT and IDFT, and not the computational efficiency of those implementations.

The Fourier transform is equally applicable to real and complex signals and many software packages can calculate the DFT for both types of signal. The testing described in this report, however, is concerned only with real signals.

3.1 Discrete Fourier transform

The following test software for the computation of the DFT is considered:

1. IMSL [15] function **DFFTRF**.
2. Microsoft Excel [16] **Fourier Analysis tool** (part of the Analysis ToolPak).
3. Origin [17] **FFT tool**.
4. NAG [18] function **C06FAF**.
5. Mathcad [19] function **fft**.
6. LabVIEW [20] function **fft**.

3.2 Inverse discrete Fourier transform

The following test software for the computation of the IDFT is considered:

1. IMSL function **DFFTRB**.
2. Microsoft Excel **Fourier Analysis tool** (part of the Analysis ToolPak).
3. Origin **FFT tool**.
4. NAG function **C06FBF**.
5. Mathcad function **ifft**.

6. LabVIEW function **ifft**.

4 Interfacing to the test software

Software implementations of the DFT and IDFT may differ in:

- The length of signal vector (for the DFT) or number of Fourier coefficients (for the IDFT) that are permitted, and
- The format that is taken by the output of the DFT and/or the input to the IDFT, i.e., the representation of the signal in the frequency domain.

When carrying out the testing described in this report, it is important to be aware of the differences between the software tested to ensure that:

- Software for the DFT and IDFT are properly used and results provided by the software are properly understood, and
- The calculation of performance measures is implemented in terms of quantities that are directly comparable.

This section describes, for each software package, the constraints that apply to the input parameters of the functions tested, and details of the formats of the input and output parameters of the functions and how those formats relate to those used by the Matlab (reference) functions. An overview of the implementation of the testing methodology is also given.

4.1 Constraints on inputs

While Matlab, IMSL and LabVIEW place no (explicit) constraint on the number m of sampled values (or Fourier coefficients), other software packages require that m is a power of 2 and/or lies between stated lower and upper bounds as follows:

- NAG.

  The largest prime factor of m must be less than or equal to 19 and the total number of prime factors of m, counting repetitions, must not exceed 20.

- Microsoft Excel.

  m must be a power of 2 not greater than 4096.

- Origin.

  No constraint is placed on m, but truncation or padding is applied to produce a signal (for the DFT) or vector of Fourier coefficients (for the IDFT) whose length is a power of 2.

- Mathcad.

  m must be a power of 2 greater than or equal to 8.

4.2 Conversion to "standard format"

The input vectors required for and output vectors generated by the calculations are not in the same format for all the packages.[2] In some cases, therefore, pre-processing is required to convert the reference data sets to the format required by the test software, and post-processing is required to convert the test results to a standard format in order to ensure that the performance measures are calculated in terms of quantities that are directly comparable.

4.2.1 Discrete Fourier transform

Consider the DFT result returned by Matlab. For a real signal $\boldsymbol{h}$ consisting of an even number m of sampled values,[3] the Matlab **fft** function returns a vector consisting of m complex Fourier coefficients. There is redundancy in the output – the output vector $\boldsymbol{y}$ can be written as

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{m/2} \\ y_{m/2+1} \\ y_{m/2+2} \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} + jy_2^{\mathrm{I}} \\ \vdots \\ y_{m/2}^{\mathrm{R}} + jy_{m/2}^{\mathrm{I}} \\ y_{m/2+1}^{\mathrm{R}} \\ y_{m/2}^{\mathrm{R}} - jy_{m/2}^{\mathrm{I}} \\ \vdots \\ y_2^{\mathrm{R}} - jy_2^{\mathrm{I}} \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 e^{j\phi_2} \\ \vdots \\ A_{m/2} e^{j\phi_{m/2}} \\ A_{m/2+1} \\ A_{m/2} e^{-j\phi_{m/2}} \\ \vdots \\ A_2 e^{-j\phi_2} \end{bmatrix}, \qquad (1)$$

and, therefore, may be represented compactly by the m real numbers $y_1^{\mathrm{R}},\ldots,y_{m/2+1}^{\mathrm{R}},y_2^{\mathrm{I}},\ldots,y_{m/2}^{\mathrm{I}}$, or, equivalently, by the $(m/2 + 1)$ amplitudes $A_1,\ldots,A_{m/2+1}$ and the $(m/2 – 1)$ phase angles $\phi_2,\ldots,\phi_{m/2}$.

While the outputs of the DFT calculations in Microsoft Excel, LabVIEW and Origin are of identical format to that of Matlab[4, 5], the outputs from Mathcad, NAG and IMSL are different.

Mathcad outputs a scaled and more compact version:

$$\frac{1}{\sqrt{m}} \begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} - jy_2^{\mathrm{I}} \\ \vdots \\ y_{m/2}^{\mathrm{R}} - jy_{m/2}^{\mathrm{I}} \\ y_{m/2+1}^{\mathrm{R}} \end{bmatrix}.$$

NAG outputs the vector of real numbers

---

[2] The Numerical Mathematics Consortium (www.nmconsortium.org) is a network whose aim is to standardise on a core set of mathematical functions, including the (fast) Fourier transform.

[3] Throughout this report, the number m of data points used is *even*.

[4] From the Origin FFT graphical user interface, under the "Settings" tab, the output options "Normalize Amplitude" and "Shift Results" must both be *unticked* and the "Exponential Phase Factor" "–1 (Electrical Engineering)" radio button must be selected in order to obtain results in the same format as returned by Matlab.

[5] Origin returns two columns containing the real and imaginary parts of the DFT rather than a single column of complex values.

$$\frac{1}{\sqrt{m}}\begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} \\ \vdots \\ y_{m/2+1}^{\mathrm{R}} \\ y_{m/2}^{\mathrm{I}} \\ \vdots \\ y_2^{\mathrm{I}} \end{bmatrix}.$$

IMSL outputs the vector of real numbers

$$\begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} \\ y_2^{\mathrm{I}} \\ \vdots \\ y_{m/2}^{\mathrm{R}} \\ y_{m/2}^{\mathrm{I}} \\ y_{m/2+1}^{\mathrm{R}} \end{bmatrix}.$$

In order that the performance measures for the DFT are calculated in terms of quantities that are directly comparable, Mathcad, NAG and IMSL outputs (test results) are converted to the same format as the reference results provided by Matlab.

4.2.2 Inverse discrete Fourier transform

Consider the IDFT result returned by Matlab. For a given input vector $\boldsymbol{y}$ of m Fourier coefficients that are conjugate symmetric as in equation (1), the Matlab **ifft** function returns a vector $\boldsymbol{h}$ consisting of m real values:

$$\boldsymbol{h} = \begin{bmatrix} h_1 \\ \vdots \\ h_m \end{bmatrix}.$$

While the inputs to the IDFT calculations in Microsoft Excel, LabVIEW and Origin are of identical format to that of Matlab, i.e., as given by equation (1), Mathcad, NAG and IMSL require input vectors having formats

$$\begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} - jy_2^{\mathrm{I}} \\ \vdots \\ y_{m/2}^{\mathrm{R}} - jy_{m/2}^{\mathrm{I}} \\ y_{m/2+1}^{\mathrm{R}} \end{bmatrix}, \begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} \\ \vdots \\ y_{m/2+1}^{\mathrm{R}} \\ y_{m/2}^{\mathrm{I}} \\ \vdots \\ y_2^{\mathrm{I}} \end{bmatrix} \text{ and } \begin{bmatrix} y_1^{\mathrm{R}} \\ y_2^{\mathrm{R}} \\ y_2^{\mathrm{I}} \\ \vdots \\ y_{m/2}^{\mathrm{R}} \\ y_{m/2}^{\mathrm{I}} \\ y_{m/2+1}^{\mathrm{R}} \end{bmatrix},$$

respectively, and all three packages output scaled versions of $\boldsymbol{h}$:

$$\sqrt{m}\begin{bmatrix} h_1 \\ \vdots \\ h_m \end{bmatrix}.$$

When testing the calculation of the IDFT using Mathcad, NAG and IMSL, inputs (reference data sets) are converted from the format required by Matlab to the format required by the test software.

In order that the performance measures for the IDFT are calculated in terms of quantities that are directly comparable:

- Complex outputs (test results) returned by the test software are converted to real quantities by ignoring their (numerically small) imaginary part.
- Mathcad, NAG and IMSL outputs are converted to the same format as the reference results provided by Matlab.

4.2.3 Implementation of the testing methodology

For each function tested, a reference data set and corresponding reference results are generated in Matlab. A "program" (or equivalent) is then implemented in the language or package of which the function is a part to undertake the following generic operations:

1. Load or import the reference data set and reference results,
2. Convert (if required) the reference data set to the format required by the test function,[6, 7]
3. Apply the (test) function to the reference data set to obtain test results,
4. Convert (if required) the test results to standard (Matlab) format,
5. Compare the test and reference results by the evaluation of a performance measure, and
6. Write the value of the performance measure to (an ASCII text) file.

These operations are repeated for a number of reference data sets (see sections 5 and 7). The performance measures used to compare the test and reference results are defined in section 6.

5 Specification of Reference Data Sets

Performance parameters are used to capture the properties of data sets that would be encountered in practice and to describe the range of admissible inputs to the test software. By varying an individual performance parameter, sequences of data sets may be generated, with the sequence forming a *graded* sequence in cases where the performance parameter relates directly to the condition or "degree of difficulty" of the problem represented by the data. By investigating the performance of the test software for such graded sequences, it is possible to identify cases where the test software is

[6] No conversion is required when testing the calculation of the DFT.

[7] For convenience, when testing the calculation of the IDFT in Mathcad, the conversion of the reference data set is implemented in Matlab, i.e., the reference data set loaded into Mathcad has *already* been converted to the required format.

based upon a poor choice of mathematical algorithm.

Reference pairs, comprising reference data sets and corresponding reference results, are generated by starting with a reference problem and applying reference software to it to produce corresponding reference results. Matlab is used as reference software with the functions **fft** and **ifft** applied (see below) to reference data to produce reference results. The Matlab functions are based on the FFTW library[8] [21], a library of high quality software for the DFT and IDFT, which won the Wilkinson Prize for Numerical Software in 1999.[9]

For the testing described in this report, reference pairs are generated in two ways:

- Assign parameters that define a signal, generate the reference signal and then generate the corresponding reference Fourier coefficients.
- Assign parameters that define amplitudes and phases, generate the reference Fourier coefficients and then generate the corresponding reference signal.

For each of the above methods, three types of signal are considered, giving a total of six different reference data types:

1. Signal is the sum of a polynomial and random noise.
2. Signal is the sum of periodic functions and random noise.
3. Signal is made up of samples drawn randomly from a probability distribution.
4. Fourier coefficients consist of random amplitudes and random phases.
5. Fourier coefficients consist of decaying amplitudes and random phases.
6. Fourier coefficients consist of a limited number of non-zero amplitudes and phases.

For each reference data type, a reference signal and a reference set of Fourier coefficients are generated in the manner described in section 8. Tables 1 to 6 list the performance parameters used to define each reference data type. The table also gives the values assigned to each performance parameter. A number of sequences of reference data sets are generated, where each sequence corresponds to setting the values for all but one of the performance parameters equal to default values and varying the value of the remaining parameter according to the information provided in the table.

6 Specification of Performance Measures

Performance measures or quality metrics are used to quantify the performance of the test software for the reference data sets to which the test software is applied. Furthermore, by relating the values of these metrics to the requirements of the user, it is possible to assess objectively whether the test software meets these requirements and is therefore "fit for purpose".

---

[8] See www.fftw.org. The web site contains much information about algorithms and software for the FFT, including **benchFFT**, a program to benchmark FFT software, by measuring the performance and accuracy of publicly available FFT implementations (see www.fftw.org/benchfft/). The concern here, however, is with the accuracy (rather than the performance) of software provided with proprietary software packages and libraries (rather than specific algorithms running on particular computing architectures).

[9] The Wilkinson Prize is sponsored by the National Physical Laboratory, the Numerical Algorithms Group, Ltd., and the Argonne National Laboratory.

Performance parameter	*Values defining sequences of reference data sets*					
Number of points	2^7	$\mathbf{2^8}$	2^9	2^{10}	2^{11}	10^4
Polynomial order	1	3	**5**	7	9	11
Noise standard deviation	0	10^{-4}	10^{-3}	10^{-2}	$\mathbf{10^{-1}}$	10^0

Table 1: Values for performance parameters for the specification of type 1 reference data sets for the computation of the DFT and IDFT. Default values for the performance parameters are shown in bold.

Performance parameter	*Values defining sequences of reference data sets*					
Number of points	2^7	$\mathbf{2^8}$	2^9	2^{10}	2^{11}	10^4
Number of periodic functions	1	5	**9**	13	17	21
Noise standard deviation	0	10^{-4}	10^{-3}	10^{-2}	$\mathbf{10^{-1}}$	10^0

Table 2: As Table 1 except for type 2 reference data sets.

<table>
<tr><td>Performance parameter</td><td colspan="30">Values defining sequences of reference data sets</td></tr>
<tr><td>Number of samples</td><td colspan="5">2^7</td><td colspan="5">$\mathbf{2^8}$</td><td colspan="5">2^9</td><td colspan="5">2^{10}</td><td colspan="5">2^{11}</td><td colspan="5">10^4</td></tr>
<tr><td>Probability distribution type</td><td colspan="10">Rectangular</td><td colspan="10">Triangular</td><td colspan="10">Gaussian</td></tr>
<tr><td>Sample mean</td><td colspan="6">0</td><td colspan="6">$\mathbf{10^2}$</td><td colspan="6">10^4</td><td colspan="6">10^6</td><td colspan="6">10^8</td></tr>
<tr><td>Sample standard deviation</td><td colspan="6">1</td><td colspan="6">$\mathbf{10^2}$</td><td colspan="6">10^4</td><td colspan="6">10^6</td><td colspan="6">10^8</td></tr>
</table>

Table 3: As Table 1 except for type 3 reference data sets.

Performance parameter	*Values defining sequences of reference data sets*					
Number of points	2^7	$\mathbf{2^8}$	2^9	2^{10}	2^{11}	10^4
Amplitude mean	1	10	$\mathbf{10^2}$	10^3	10^4	10^5
Amplitude standard deviation	0	0.005	**0.05**	0.5	5	50

Table 4: As Table 1 except for type 4 reference data sets.

Performance parameter	*Values defining sequences of reference data sets*					
Number of points	2^7	$\mathbf{2^8}$	2^9	2^{10}	2^{11}	10^4
Decay constant	0.5	**1**	1.5	2	4	8
Noise standard deviation	0	10^{-4}	10^{-3}	10^{-2}	$\mathbf{10^{-1}}$	10^0

Table 5: As Table 1 except for type 5 reference data sets.

Performance parameter	*Values defining sequences of reference data sets*					
Number of points	2^7	$\mathbf{2^8}$	2^9	2^{10}	2^{11}	10^4
Number of non-zero amplitudes and phases	1	5	**9**	13	17	21

Table 6: As Table 1 except for type 6 reference data sets.

For the computation of the DFT, the following performance measure is used:

$$P(\boldsymbol{h}) = \frac{\left\| \mathrm{abs}\left(\boldsymbol{y}^{\mathrm{test}} - \boldsymbol{y}^{\mathrm{ref}}\right) \right\|}{\left\| \mathrm{abs}\left(\boldsymbol{y}^{\mathrm{ref}}\right) \right\|}, \qquad \boldsymbol{y}^{\mathrm{ref}} \neq \boldsymbol{0}, \tag{2}$$

where

$$\| \boldsymbol{y} \| = \sqrt{\sum_{i=1}^{m} y_i^2}$$

and

$$\mathrm{abs}(\boldsymbol{y}) = \begin{bmatrix} \mathrm{abs}(y_1) \\ \vdots \\ \mathrm{abs}(y_m) \end{bmatrix} = \begin{bmatrix} \sqrt{(\mathrm{Re}(y_1))^2 + (\mathrm{Im}(y_1))^2} \\ \vdots \\ \sqrt{(\mathrm{Re}(y_m))^2 + (\mathrm{Im}(y_m))^2} \end{bmatrix}.$$

$\boldsymbol{h}$ denotes the input reference data set and $\boldsymbol{y}^{\mathrm{test}}$ and $\boldsymbol{y}^{\mathrm{ref}}$ are, respectively, the test and reference results comprising complex Fourier coefficients (section 4.2.1).

For the computation of the IDFT, the following performance measure is used [5]:

$$P(\boldsymbol{y}) = \frac{\left\| \boldsymbol{h}^{\mathrm{test}} - \boldsymbol{h}^{\mathrm{ref}} \right\|}{\left\| \boldsymbol{h}^{\mathrm{ref}} \right\|}, \qquad \boldsymbol{h}^{\mathrm{ref}} \neq \boldsymbol{0}, \tag{3}$$

where $\boldsymbol{y}$ denotes the input reference data set and $\boldsymbol{h}^{\mathrm{test}}$ and $\boldsymbol{h}^{\mathrm{ref}}$ are, respectively, the test and reference results comprising real signal values (section 4.2.2).

The performance measure $P(\boldsymbol{x})$ provides an indication, for the reference data set $\boldsymbol{x}$, of the relative accuracy of the test result compared with the reference result.

7 Generation of Reference Pairs

This section provides specifications for the calculation of reference data sets and corresponding reference results for the computations that are the subject of this report. In each case, the inputs, outputs and the procedure for the calculation of reference data sets and corresponding reference results are given.

Reference data sets for both the DFT and IDFT are generated at the same time. For each reference data set type, the outputs of the data generator are:

- Reference signal $\boldsymbol{h}^{\mathrm{REF}}$, and
- Reference Fourier coefficients $\boldsymbol{y}^{\mathrm{REF}}$.

For reference data sets of types 1, 2 and 3, a reference signal is first defined. The corresponding reference Fourier coefficients are obtained by applying Matlab's **fft** function to the reference signal. For reference data sets of types 4, 5 and 6, reference Fourier coefficients are first defined. The corresponding reference signal is obtained by applying Matlab's **ifft** function to the reference Fourier coefficients.

7.1 Type 1 reference data sets

The inputs for the data generator are:

- Number of points *m*,
- Polynomial order *n*,
- Noise standard deviation σ, and
- Random number seed *S*.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate linearly spaced values x_i, $i = 1,\ldots,m,$ in the range $[-1,1]$.
2. Generate a vector $\boldsymbol{a}$ of polynomial coefficients a_l, $l = 1,\ldots,n,$ sampled from the Gaussian distribution with mean zero and standard deviation one.
3. Calculate the vector $\boldsymbol{p}$ of polynomial values

$$p_i = p(x_i, \boldsymbol{a}) = \sum_{l=1}^{n} a_l T_{l-1}(x_i), \; i = 1,\ldots,m,$$

   where $T_l(x)$ is the Chebyshev polynomial of degree *l* (order *l* + 1) defined by the recurrence relation [22]

$$T_0(x) = 1, \; T_1(x) = x, \; T_l(x) = 2xT_{l-1}(x) - T_{l-2}(x), \; l \geq 2.$$

4. Generate a vector $\boldsymbol{r}$ of random numbers r_i, $i = 1,\ldots,m,$ sampled from the Gaussian distribution with mean zero and standard deviation σ.
5. Set $\boldsymbol{h}^{\text{REF}} = \boldsymbol{p} + \boldsymbol{r}$.
6. Apply the Matlab **fft** function to $\boldsymbol{h}^{\text{REF}}$ to obtain reference Fourier coefficients $\boldsymbol{y}^{\text{REF}}$.

7.2 Type 2 reference data sets

The inputs for the data generator are:

- Number of points *m*,
- Number of periodic functions *n*,
- Noise standard deviation σ, and
- Random number seed *S*.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate linearly spaced values x_i, $i = 1,\ldots,m,$ in the range $[-1,1]$.
2. Generate random numbers a_l, $l = 1,\ldots,n,$ sampled from the rectangular distribution $\mathrm{R}(0,1)$.
3. Generate random numbers f_l, $l = 1,\ldots,n,$ sampled from the Gaussian distribution

with mean zero and standard deviation 0.1.

4. Generate random numbers δ_l, $l = 1,\dots,n$, sampled from the rectangular distribution $\mathrm{R}(0,2\pi)$.
5. Calculate the vectors $\boldsymbol{p}_l$ of values $p_{l,i} = a_l \sin(2\pi f_l x_i + \delta_l)$, $i = 1,\dots,m$, $l = 1,\dots,n$.
6. Generate a vector $\boldsymbol{r}$ of random numbers r_i, $i = 1,\dots,m$, sampled from the Gaussian distribution with mean zero and standard deviation σ.
7. Set $\boldsymbol{h}^{REF} = \sum_{l=1}^{n} \boldsymbol{p}_l + \boldsymbol{r}$.
8. Apply the Matlab **fft** function to $\boldsymbol{h}^{\mathrm{REF}}$ to obtain reference Fourier coefficients $\boldsymbol{y}^{\mathrm{REF}}$.

7.3 Type 3 reference data sets

The inputs for the data generator are:

- Number of samples *m*,
- Probability type *P*,
- Sample mean $\bar{x}$,
- Sample standard deviation σ, and
- Random number seed *S*.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate a vector $\boldsymbol{r}$ of random numbers r_i, $i = 1,\dots,m$, sampled from the distribution of probability type *P* with mean $\bar{x}$ and standard deviation σ.
2. Set $\boldsymbol{h}^{\mathrm{REF}} = \boldsymbol{r}$.
3. Apply the Matlab **fft** function to $\boldsymbol{h}^{\mathrm{REF}}$ to obtain reference Fourier coefficients $\boldsymbol{y}^{\mathrm{REF}}$.

7.4 Type 4 reference data sets

The inputs for the data generator are:

- Number of points *m*,
- Amplitude mean $\overline{A}$,
- Amplitude standard deviation σ, and
- Random number seed *S*.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate random numbers A_i, $i = 1,\dots,m/2+1$, sampled from the rectangular distribution with mean $\overline{A}$ and standard deviation σ.
2. Generate random numbers ϕ_i, $i = 2,\dots,m/2$, sampled from the rectangular

distribution $\mathrm{R}(0,2\pi)$.

3. Set $\boldsymbol{y}^{\mathrm{REF}} = \begin{bmatrix} A_1 \\ A_2 e^{j\phi_2} \\ \vdots \\ A_{m/2} e^{j\phi_{m/2}} \\ A_{m/2+1} \\ A_{m/2} e^{-j\phi_{m/2}} \\ \vdots \\ A_2 e^{-j\phi_2} \end{bmatrix}$.

4. Apply the Matlab **ifft** function to $\boldsymbol{y}^{\mathrm{REF}}$ to obtain reference signal $\boldsymbol{h}^{\mathrm{REF}}$.

7.5 Type 5 reference data sets

The inputs for the data generator are:

- Number of points *m*,
- Decay constant *D*,
- Noise standard deviation σ, and
- Random number seed *S*.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate random numbers r_i, $i = 2,\ldots,m/2+1$, sampled from the Gaussian distribution with mean zero and standard deviation σ.
2. Set $A_1 = 1$, $A_i = e^{-iD/m} - r_i$, $i = 2,\ldots,m/2+1$.
3. Generate random numbers ϕ_i, $i = 2,\ldots,m/2$, sampled from the rectangular distribution $\mathrm{R}(0,2\pi)$.
4. Set $\boldsymbol{y}^{\mathrm{REF}} = \begin{bmatrix} A_1 \\ A_2 e^{j\phi_2} \\ \vdots \\ A_{m/2} e^{j\phi_{m/2}} \\ A_{m/2+1} \\ A_{m/2} e^{-j\phi_{m/2}} \\ \vdots \\ A_2 e^{-j\phi_2} \end{bmatrix}$.
5. Apply the Matlab **ifft** function to $\boldsymbol{y}^{\mathrm{REF}}$ to obtain reference signal $\boldsymbol{h}^{\mathrm{REF}}$.

7.6 Type 6 reference data sets

The inputs for the data generator are:

- Number of points m,
- Number of non-zero amplitudes and phases n, and
- Random number seed S.

The procedure implemented (in Matlab) by the data generator for the computation of the DFT and IDFT is as follows:

1. Generate random indices $1 \le i_l \le m/2+1,\ l = 1,\ldots,n$.
2. Generate random numbers $\tilde{A}_l,\ l = 1,\ldots,n,$ sampled from the rectangular distribution $\mathrm{R}(0,10)$.
3. Generate random numbers $\tilde{\phi}_l,\ l = 1,\ldots,n,$ sampled from the rectangular distribution $\mathrm{R}(0,2\pi)$.
4. Set $A_i = 0,\ i = 1,\ldots,m/2+1,$ and $\phi_i = 0,\ i = 2,\ldots,m/2.$
5. Set $A_{i_l} = \tilde{A}_l,\ l = 1,\ldots,n,$ and $\phi_{i_l} = \tilde{\phi}_l,\ i = 2,\ldots,n.$
6. Set $\boldsymbol{y}^{\mathrm{REF}} = \begin{bmatrix} A_1 \\ A_2 e^{j\phi_2} \\ \vdots \\ A_{m/2} e^{j\phi_{m/2}} \\ A_{m/2+1} \\ A_{m/2} e^{-j\phi_{m/2}} \\ \vdots \\ A_2 e^{-j\phi_2} \end{bmatrix}.$
7. Apply the Matlab **ifft** function to $\boldsymbol{y}^{\mathrm{REF}}$ to obtain reference signal $\boldsymbol{h}^{\mathrm{REF}}$.

8 Presentation and interpretation of results

In testing the calculations of the DFT or IDFT, given:

- A test software package,
- A reference data set type, and
- A performance parameter,

up to six values of a performance measure may be obtained (one for each value of the performance parameter).

In the results below, only the *maximum* value of a performance measure is presented as:

1. There is no requirement to compare the values of the performance measure corresponding to different values of a performance parameter.
2. The values of the performance measure corresponding to different values of a

performance parameter are generally of the same order[10] and, for the purpose of the testing in this report, it is only this order that is of interest.

Particular points to note include:

1. For packages (Mathcad, Excel 2003) that are unable to process reference data sets with 10^4 points (section 4.1), no value of the performance measure is available for that number of points.
2. Although Origin will process reference data sets containing 10^4 points, the data set is padded to produce a data set containing $2^{14} = 16\,384$ (the next highest power of 2) points before processing occurs. As the result of the processing will also contain 2^{14} points, the performance measures given by equations (2) and (3) cannot be calculated. No performance measure is therefore available for 10^4 points for Origin.

8.1 Discrete Fourier transform

Tables 7 to 12 show, for each reference data set type, the maximum values of the performance measure for each test software package for the calculation of the DFT.

Package	*Performance parameter*		
	Number of points	*Polynomial order*	*Noise standard deviation*
Matlab	0	0	0
Mathcad	1×10^{-12}	1×10^{-12}	1×10^{-12}
Excel	7×10^{-15}	6×10^{-15}	5×10^{-15}
Origin	7×10^{-15}	4×10^{-15}	4×10^{-15}
NAG	5×10^{-16}	4×10^{-16}	3×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	3×10^{-16}	3×10^{-16}	3×10^{-16}

Table 7: Maximum values of the performance measure $P(\boldsymbol{h})$ for type 1 reference data sets for calculation of the DFT.

Package	*Performance parameter*		
	Number of points	*Number of periodic functions*	*Noise standard deviation*
Matlab	0	0	0
Mathcad	2×10^{-12}	2×10^{-12}	2×10^{-12}
Excel	6×10^{-15}	4×10^{-15}	6×10^{-15}
Origin	7×10^{-15}	3×10^{-15}	3×10^{-15}
NAG	4×10^{-16}	3×10^{-16}	3×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	4×10^{-16}	3×10^{-16}	3×10^{-16}

Table 8: As Table 7 except for type 2 reference data sets.

[10] For example, for testing of the calculation of the DFT for Excel, reference data set type 1 and performance parameter polynomial order, the six values of performance measure obtained are 6×10^{-15}, 4×10^{-15}, 5×10^{-15}, 4×10^{-15}, 5×10^{-15} and 4×10^{-15}, with the maximum value of 6×10^{-15} displayed in the table of results.

Package	*Performance parameter*			
	Number of samples	*Probability distribution type*	*Sample mean*	*Sample standard deviation*
Matlab	0	0	0	0
Mathcad	1×10^{-12}	1×10^{-12}	2×10^{-12}	2×10^{-12}
Excel	5×10^{-15}	4×10^{-15}	4×10^{-15}	4×10^{-15}
Origin	4×10^{-15}	2×10^{-15}	3×10^{-15}	3×10^{-15}
NAG	4×10^{-16}	2×10^{-16}	3×10^{-16}	3×10^{-16}
IMSL	3×10^{-16}	2×10^{-16}	2×10^{-16}	3×10^{-16}
LabVIEW	5×10^{-16}	2×10^{-16}	2×10^{-16}	3×10^{-16}

Table 9: As Table 7 except for type 3 reference data sets.

Package	*Performance parameter*		
	Number of points	*Amplitude mean*	*Amplitude standard deviation*
Matlab	4×10^{-16}	3×10^{-16}	3×10^{-16}
Mathcad	2×10^{-12}	2×10^{-12}	2×10^{-12}
Excel	5×10^{-15}	4×10^{-15}	5×10^{-15}
Origin	6×10^{-15}	3×10^{-15}	3×10^{-15}
NAG	5×10^{-16}	3×10^{-16}	4×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	7×10^{-16}	3×10^{-16}	3×10^{-16}

Table 10: As Table 7 except for type 4 reference data sets.

Package	*Performance parameter*		
	Number of points	*Decay constant*	*Noise standard deviation*
Matlab	4×10^{-16}	3×10^{-16}	3×10^{-16}
Mathcad	2×10^{-12}	2×10^{-12}	2×10^{-12}
Excel	5×10^{-15}	4×10^{-15}	4×10^{-15}
Origin	7×10^{-15}	3×10^{-15}	3×10^{-15}
NAG	5×10^{-16}	3×10^{-16}	4×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	7×10^{-16}	3×10^{-16}	3×10^{-16}

Table 11: As Table 7 except for type 5 reference data sets.

Package	*Performance parameter*	
	Number of points	*Number of non-zero amplitudes and phases*
Matlab	3×10^{-16}	3×10^{-16}
Mathcad	2×10^{-12}	2×10^{-12}
Excel	5×10^{-15}	4×10^{-15}
Origin	7×10^{-15}	4×10^{-15}
NAG	5×10^{-16}	4×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}
LabVIEW	7×10^{-16}	3×10^{-16}

Table 12: As Table 7 except for type 6 reference data sets.

The values of the performance measure for Matlab for the calculation of the DFT for reference data set types 1, 2 and 3 are all zero. This is to be expected as the reference and test results are identical (with both generated using the Matlab **fft** function). The values of the performance measure for Matlab for the calculation of the DFT for reference data set types 4, 5 and 6 provide a measure of the consistency of Matlab's **ifft** and **fft** functions, i.e., the extent to which application of the **ifft** to reference Fourier coefficients to obtain a reference signal followed by application of **fft** to the reference signal gives the original reference Fourier coefficients. For all reference data sets and all other test software except Mathcad, the values of the performance measure are of the order of 10^{-15} or 10^{-16}. These results indicate that the numerical accuracy of the test software is high for this calculation and these data sets. For Mathcad, the values of the performance measure are typically of the order of 10^{-12} or 10^{-13}.

8.2 Inverse discrete Fourier transform

Tables 13 to 18 show, for each reference data set type, the maximum values of the performance measure for each test software package for the calculation of the IDFT.

Package	*Performance parameter*		
	Number of points	*Polynomial order*	*Noise standard deviation*
Matlab	3×10^{-16}	4×10^{-16}	4×10^{-16}
Mathcad	1×10^{-12}	1×10^{-12}	1×10^{-12}
Excel	5×10^{-15}	5×10^{-15}	3×10^{-15}
Origin	7×10^{-15}	2×10^{-15}	2×10^{-15}
NAG	4×10^{-16}	4×10^{-16}	3×10^{-16}
IMSL	3×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	5×10^{-16}	4×10^{-16}	3×10^{-16}

Table 13: Maximum values of the performance measure $P(\boldsymbol{y})$ for type 1 reference data sets for the calculation of the IDFT.

Package	*Performance parameter*		
	Number of points	*Number of periodic functions*	*Noise standard deviation*
Matlab	4×10^{-16}	3×10^{-16}	3×10^{-16}
Mathcad	8×10^{-13}	8×10^{-13}	8×10^{-13}
Excel	4×10^{-15}	4×10^{-15}	5×10^{-15}
Origin	5×10^{-15}	2×10^{-15}	1×10^{-15}
NAG	5×10^{-16}	4×10^{-16}	3×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	1×10^{-15}	3×10^{-16}	3×10^{-16}

Table 14: As Table 13 except for type 2 reference data sets.

Package	*Performance parameter*			
	Number of samples	*Probability distribution type*	*Sample mean*	*Sample standard deviation*
Matlab	3×10^{-16}	3×10^{-16}	3×10^{-16}	3×10^{-16}
Mathcad	7×10^{-13}	7×10^{-13}	1×10^{-12}	1×10^{-12}
Excel	3×10^{-15}	3×10^{-15}	4×10^{-15}	4×10^{-15}
Origin	3×10^{-15}	1×10^{-15}	1×10^{-15}	2×10^{-15}
NAG	4×10^{-16}	3×10^{-16}	3×10^{-16}	3×10^{-16}
IMSL	3×10^{-16}	3×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	5×10^{-16}	3×10^{-16}	3×10^{-16}	3×10^{-16}

Table 15: As Table 13 except for type 3 reference data sets.

Package	*Performance parameter*		
	Number of points	*Amplitude mean*	*Amplitude standard deviation*
Matlab	0	0	0
Mathcad	1×10^{-12}	1×10^{-12}	1×10^{-12}
Excel	4×10^{-15}	4×10^{-15}	4×10^{-15}
Origin	5×10^{-15}	1×10^{-15}	1×10^{-15}
NAG	5×10^{-16}	3×10^{-16}	3×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	7×10^{-16}	3×10^{-16}	3×10^{-16}

Table 16: As Table 13 except for type 4 reference data sets.

Package	*Performance parameter*		
	Number of points	*Decay constant*	*Noise standard deviation*
Matlab	0	0	0
Mathcad	1×10^{-12}	1×10^{-12}	1×10^{-12}
Excel	4×10^{-15}	4×10^{-15}	3×10^{-15}
Origin	5×10^{-15}	2×10^{-15}	1×10^{-15}
NAG	5×10^{-16}	3×10^{-16}	3×10^{-16}
IMSL	4×10^{-16}	3×10^{-16}	3×10^{-16}
LabVIEW	7×10^{-16}	3×10^{-16}	3×10^{-16}

Table 17: As Table 13 except for type 5 reference data sets.

Package	*Performance parameter*	
	Number of points	*Number of non-zero amplitudes and phases*
Matlab	0	0
Mathcad	1×10^{-12}	1×10^{-12}
Excel	3×10^{-15}	4×10^{-15}
Origin	5×10^{-15}	2×10^{-15}
NAG	5×10^{-16}	4×10^{-16}
IMSL	3×10^{-16}	3×10^{-16}
LabVIEW	6×10^{-16}	3×10^{-16}

Table 18: As Table 13 except for type 6 reference data sets.

The values of the performance measure for Matlab for the calculation of the IDFT for reference data set types 4, 5 and 6 are all zero. This is to be expected as the reference and test results are identical (with both generated using the Matlab **ifft** command). As for the testing of the DFT, the values of the performance measure for Matlab for the calculation of the IDFT for reference data set types 1, 2 and 3 provide a measure of the consistency of Matlab's **fft** and **ifft** functions. For all reference data sets and all other test software except Mathcad, the values of the performance measure are of the order of 10^{-15} or 10^{-16}. The results indicate that the numerical accuracy of the test software is high for this calculation and these data sets. For Mathcad, the values of the performance measure are typically of the order of 10^{-12} or 10^{-13}.

9 Conclusions

In this report we have described the application of a general methodology [1] for testing the numerical correctness of scientific software to functions for the calculations of the discrete Fourier transform and inverse discrete Fourier transform. The functions tested are taken from a number of proprietary software packages and libraries, including the NAG and IMSL (FORTRAN) libraries, Microsoft Excel, Origin, LabVIEW and Mathcad. Matlab has been used as reference software for the generation of reference pairs, comprising reference data sets and corresponding reference results.

Each stage of the methodology, from documenting the specifications of the functions tested through the definition of performance parameters and performance measures to the presentation and interpretation of the results of the testing, has been described. In this way, and by stating any assumptions made in the application of the methodology, the testing undertaken is made repeatable and as objective as possible given the ("black-box") nature of the testing.

The results obtained and conclusions drawn from the testing undertaken must be interpreted in the context of the particular calculations considered, and do not necessarily relate to other functions of the software libraries and packages considered. The black-box testing described here has been carried out in such a way that the functions have been used without taking account of information elsewhere (where it exists), for example, as contained in publications or posted on the internet. This mode of use is deliberate, since we believe it accords with that adopted by most users generally, and within metrology in particular. Furthermore, it allows for the consistent testing of the functions given the differing quality and quantity of information provided with each.

Some particular conclusions are as follows:

- The results for Matlab that are non-zero (DFT for reference data set types 4, 5 and 6, and IDFT for types 1, 2 and 3) indicate a high consistency between Matlab's functions **fft** for the DFT and **ifft** for the IDFT.
- All the software tested for the calculations of the DFT and IDFT, apart from Mathcad, return test results for the reference data sets that can be considered to have a numerical accuracy comparable to that expected from reference software.
- For Mathcad, the values of the performance measures are generally 10^3 or 10^4 times greater than those for the other packages.

Acknowledgements

This report constitutes one of the deliverables of Project 3.2 of the 2004–2007 NMS *Software Support for Metrology* Programme, and has been funded by the National Measurement System Directorate of the UK Department of Trade and Industry.

References

[1] H.R. Cook, M.G. Cox, M.P. Dainton and P.M. Harris. A methodology for testing spreadsheets and other packages used in metrology. *NPL Report CISE 25/99*, 1999.

[2] M.G. Cox, M.P. Dainton and P.M. Harris. Testing functions for the calculation of standard deviation. *NPL Report CMSC 07/00*, 2000.

[3] M.G. Cox, M.P. Dainton and P.M. Harris. Testing functions for linear regression. *NPL Report CMSC 08/00*, 2000.

[4] M.G. Cox, P.M. Harris, E.G. Johnson, P.D. Kenward and G.I. Parkin. Testing the numerical correctness of software. *NPL Report CMSC 34/04*, 2004.

[5] P.M. Harris, K.J. Lines and I.M. Smith. Testing the numerical correctness of software. *NPL Report DEM-ES 004 (CMSC 34/04 revised)*, National Physical Laboratory, 2006.

[6] S.L.R. Ellison, M.G. Cox, A.B. Forbes, B.P. Butler, S.A. Hannaby, P.M. Harris, and S.M. Hodson. Development of data sets for the validation of analytical instrumentation. *Journal of AOAC International*, **77**(3), pp 1-5, 1994.

[7] Statistics Software Qualification. Edited by B.P. Butler, M.G. Cox, S.L.R. Ellison and W.A. Hardcastle. The Royal Society of Chemistry, 1996.

[8] M.G. Cox and P.M. Harris. Design and use of reference data sets for testing scientific software. *Analytica Chimica Acta* **380**, pp 339 – 351, 1999.

[9] R.M. Barker, P.M. Harris and L. Wright. Software Support for Metrology Good Practice Guide No. 16: Testing algorithms and software. *NPL Report DEM-ES 003*, National Physical Laboratory, 2005.

[10] H.R. Cook, M.G. Cox, M.P. Dainton and P.M. Harris. Testing spreadsheets and other packages used in metrology: A case study. *NPL Report CISE 26/99*, September 1999.

[11] Matlab (Version 7.2.0.232 (R2006a)). Copyright © 1984–2006, The MathWorks, Inc.

[12] J.W. Cooley, and O.W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* **19**, pp 297-301, 1965.

[13] E.O. Brigham. *The fast Fourier transform and its applications*. Prentice-Hall, 1988.

[14] W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery. *Numerical Recipes in Fortran 90*. Cambridge University Press, 1996.

[15] IMSL Fortran 90 MP Library (Version 4.01). Copyright © 1999, Visual Numerics, Inc.

[16] Microsoft ® Office Excel 2003 (11.5612.5606). Copyright © 1985–2003, Microsoft Corporation.

[17] Origin (OriginPro 7.5 SR1 v7.5776 B776). Copyright © 1991–2003, OriginLab Corporation.

[18] The NAG Fortran Library (Mark 20). Copyright © 2001, Numerical Algorithms Group Ltd.

[19] Mathcad (Version 13.0 (509121419)). Copyright © 1986-2005, Mathsoft Engineering & Education, Inc.

[20] LabVIEW (Version 8.2). Copyright © 2006, National Instruments.

[21] M. Frigo and S.G. Johnson. The Design and Implementation of FFTW3. *Proceedings of the IEEE* **93** (2), pp 216–231, 2005.

[22] L. Fox and I.B. Parker. *Chebyshev polynomials in numerical analysis*. Oxford University Press, 1968.