Reducing the Influence of Tiny Normwise Relative Errors on Performance Profiles

Dingle, Nicholas J. and Higham, Nicholas J.

2011

MIMS EPrint: **2011.90**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: http://eprints.maths.manchester.ac.uk/
And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

Reducing the Influence of Tiny Normwise Relative Errors on Performance Profiles

NICHOLAS J. DINGLE, The University of Manchester
NICHOLAS J. HIGHAM, The University of Manchester

It is a widespread but little-noticed phenomenon that the normwise relative error $\|x - y\|/\|x\|$ of vectors x and y of floating point numbers of the same precision, where y is an approximation to x, can be many orders of magnitude smaller than the unit roundoff. We analyze this phenomenon and show that in the ∞-norm it happens precisely when x has components of widely varying magnitude and every component of x of largest magnitude agrees with the corresponding component of y. Performance profiles are a popular way to compare competing algorithms according to particular measures of performance. We show that performance profiles based on normwise relative errors can give a misleading impression due to the influence of zero or tiny normwise relative errors. We propose a transformation that reduces the influence of these extreme errors in a controlled manner, while preserving the monotonicity of the underlying data and leaving the performance profile unchanged at its left end-point. Numerical examples with both artificial and genuine data illustrate the benefits of the transformation.

Categories and Subject Descriptors: G.4 [**Mathematical Software**]: Algorithm design and analysis; G.1.0 [**General**]: Computer arithmetic; G.1.3 [**Numerical linear algebra**]

General Terms: Algorithms, Performance

Additional Key Words and Phrases: normwise relative error, performance profile, floating point arithmetic, forward error, backward error

1. INTRODUCTION

For a given norm $\|\cdot\|$ on $\mathbb{R}^n$, the normwise relative error of a vector $y \in \mathbb{R}^n$ as an approximation to $x \in \mathbb{R}^n$ is

$$\theta(x, y) = \frac{\|x - y\|}{\|x\|}.$$

In practical computation, x and y are vectors of floating point numbers of the same precision and x represents the true solution to some problem of interest. Usually, x is either exactly the solution (for example, because the problem has been specially constructed so that the solution is exactly representable in floating point arithmetic) or is an accurate approximation to the true solution, possibly computed using higher precision arithmetic. The approximation y to x will be subject to various possible sources of error, including rounding and discretization errors. For algorithms in which rounding errors are the only source of error, such as direct methods for solving linear systems, the aim is usually to achieve a normwise relative error of order u, the unit roundoff. We recall that for floating point arithmetic with base β and precision t, $u = \frac{1}{2}\beta^{1-t}$ [Higham 2002, Sec. 2.1].

This paper is motivated by our observation of a phenomenon that is widespread, yet hardly noticed: normwise relative errors are sometimes less than, and possibly several orders of magnitude less than, u. This phenomenon is seen, for example, in three papers on iterative refinement for linear systems [Demmel et al. 2006, Figs. 2,

Version of January 28, 2013.
This work was supported by Engineering and Physical Sciences Research Council grant EP/I006702/1. The work of the second author was also supported by Engineering and Physical Sciences Research Council grant EP/E050441/1 (CICADA: Centre for Interdisciplinary Computational and Dynamical Analysis) and European Research Council Advanced Grant MATFUN (267526). Author's addresses: Nicholas J. Dingle and Nicholas J. Higham, School of Mathematics, The University of Manchester, Manchester, M13 9PL, UK; email: higham@ma.man.ac.uk, nicholas.dingle@manchester.ac.uk.

3], [Higham 1997, Table 2], [Oishi et al. 2009, Tables 2, 3], and we recently noticed it in an experiment to compare the accuracy of different methods for computing the action of the matrix exponential on a vector, $e^A b$ [Al-Mohy and Higham 2011, Exp. 1]. We can illustrate the phenomenon with a simple experiment.

In MATLAB we use the function `matrix` from the Matrix Computation Toolbox [Higham] to generate 47 10×10 matrices A. These include most of the matrices provided by the MATLAB `gallery` function and a few others; we excluded matrices with 1-norm condition numbers greater than $u^{-1}/100$. MATLAB uses IEEE standard double precision arithmetic with unit roundoff $u = 2^{-53} \approx 1.1 \times 10^{-16}$, and our computations are carried out in MATLAB R2011b. We solve the system $Ax = b$, where $b_i = 11 - i$, using the backslash operator in double precision arithmetic, so that y is `A\b`. To obtain x, we solve the system in 100-digit arithmetic via `vpa(A,100)\vpa(b,100)`, where the `vpa` function is from the Symbolic Math Toolbox, and round the result to double precision. The vector $\theta_\infty(x, y)$ of normwise relative errors sorted in increasing order comprises 10 zeros followed by (to three significant digits)

```
1.14e-17  1.48e-17  2.29e-17  3.70e-17  4.60e-17  8.40e-17  1.57e-16
```

and then 30 larger errors (of maximum size around 10^{-5}). So here we have $\theta_\infty(x, y)$ nonzero but smaller than u in about 15 percent of the cases, by a factor as large as 10. Now we repeat the experiment, but with A replaced by its diagonal part. Since `A\b` now computes a correctly rounded approximation to each solution component we carry out one step of iterative refinement in fixed precision [Higham 1997] in order to introduce some rounding errors:

```
y = A\b; y = y + A\(b-A*y);
```

We now have sorted normwise relative errors comprising 32 zeros followed by

```
1.32e-22  3.39e-22  3.39e-21  8.67e-20  1.39e-18  4.36e-18  5.30e-18
5.83e-18  1.45e-17  3.76e-17  3.76e-17  4.27e-17
```

and four errors of order 10^{-16}. Now we see several instances with $0 < \theta_\infty(x, y) \ll u$.

We note that in this example we can alternatively compute the numerator of $\theta_\infty(x, y)$ by forming `vpa(A,100)\vpa(b,100) - vpa(y,100)` and then rounding this 100-digit difference back to double precision. This results in even more cases where $0 < \theta_\infty(x, y) \ll u$.

This behavior raises two questions. The first concerns the interpretation of the normwise relative error. The usual interpretation is that if $\theta(x, y) \approx \frac{1}{2} \times 10^{-p}$ then components x_i with $|x_i| \approx \|x\|_\infty$ have about p correct significant decimal digits [Higham 2002, Sec. 1.2]. Thus if $\theta(x, y) \approx u$ then the largest components of y are accurate in virtually all their digits. What, then, does it mean for $\theta(x, y)$ to be significantly smaller than u?

The second question is whether the occurrence of computed y with $\theta(x, y) \ll u$ affects how we assess the results of numerical experiments. Our particular concern is with performance profiles, proposed by Dolan and Moré [2002] as a way to compare a performance measure of different methods on a set of test problems. Although only introduced in 2002, performance profiles are already widely used, as indicated by the 419 citations to Dolan and Moré [2002] on the Web of Science in January 2013. When the performance measure is normwise relative error, can performance profiles be skewed by tiny normwise relative errors and, if so, what should be done to remedy this?

We begin, in Section 2, by analyzing the phenomenon of tiny normwise relative errors. In particular, we characterize when $\theta_\infty(x, y) \ll u$ holds for floating point vectors x and y. In Section 3 we show how zero and tiny normwise relative errors can produce misleading performance profiles and we propose a transformation of the data that produces more useful profiles.

Throughout this paper x and y are arbitrary vectors of floating point numbers and we make no assumptions about their provenance, though we have in mind scenarios such as those in the experiment and cited papers above.

2. NORMWISE RELATIVE ERRORS

We begin by obtaining a lower bound on the relative difference of two floating point numbers, for base β and precision t. We will need the machine epsilon $\epsilon_M = \beta^{1-t}$ [Higham 2002, Sec. 2.1].

LEMMA 2.1. *If $x \neq 0$ and y are distinct normalized floating point numbers then $|x-y|/|x| \geq \epsilon_M/\beta$ and this lower bound is attainable.*

Proof. The relative error is unaffected by shifting the exponent of x and y, so we can assume without loss of generality that $x \in [1, \beta)$. Recall that the spacing of the floating point numbers on $[1, \beta]$ is ϵ_M, while on $[\beta^{-1}, 1]$ it is ϵ_M/β [Higham 2002, Sec. 2.1]. The relative error $|x-y|/|x|$ will be minimized when $y \neq x$ is a floating point number adjacent to x, so we just need to consider $y = x \pm \epsilon_M$, for which $|x-y|/|x| = \epsilon_M/|x| > \epsilon_M/\beta$ in all but one case. The exceptional case is when $x = 1$ and $y = x - \epsilon_M/\beta$ is the next smaller floating point number: then $|x-y|/|x| = \epsilon_M/\beta$. □

From now on we specialize to base $\beta = 2$, which is the case of most practical importance. Then the lemma says that $|x-y|/|x| \geq u$.

In contrast to the scalar relative error, for vectors the relative error in the ∞-norm can be less than u, and the next lemma characterizes when this is the case.

LEMMA 2.2. *Let $x, y \in \mathbb{R}^n$ with $x \neq 0$ be vectors of normalized floating point numbers. If $\theta_\infty(x,y) < u$ then $x_k = y_k$ for all k such that $|x_k| = \|x\|_\infty$.*

Proof. For any k as defined in the statement of the lemma we have

$$\frac{|x_k - y_k|}{|x_k|} = \frac{|x_k - y_k|}{\|x\|_\infty} \leq \frac{\|x-y\|_\infty}{\|x\|_\infty} < u, \tag{1}$$

so by Lemma 2.1, $x_k = y_k$. □

Thus $\theta_\infty(x,y) < u$ for floating point vectors x and y only when the components of x and y of maximal magnitude are equal. For further insight we note that if $x_k \neq 0$ and $x_k \neq y_k$ then, by Lemma 2.1,

$$u \leq \frac{|x_k - y_k|}{|x_k|} \leq \frac{\|x-y\|_\infty}{\|x\|_\infty} \cdot \frac{\|x\|_\infty}{|x_k|} = \theta_\infty(x,y)\frac{\|x\|_\infty}{|x_k|},$$

and so $\theta_\infty(x,y) \geq u|x_k|/\|x\|_\infty$. It follows that tiny values $\theta_\infty(x,y) \ll u$ are possible only when x has components of widely varying magnitude. This situation is quite common; for example, the elements of an eigenvector of a tridiagonal matrix can have a very large dynamic range [Parlett 1998, Sec. 7.3], [Wilkinson 1965, p. 315 ff.].

For an example where $\theta_\infty(x,y) \ll u$, consider (in IEEE double precision arithmetic)

$$x = \begin{bmatrix} 1 \\ 2^{-16} \\ 2^{-32} \end{bmatrix}, \quad y^{(1)} = \begin{bmatrix} 1 \\ 2^{-16}(1+4u) \\ 2^{-32} \end{bmatrix}, \quad y^{(2)} = \begin{bmatrix} 1+2u \\ 2^{-16} \\ 2^{-32} \end{bmatrix}. \tag{2}$$

We have

$$\theta_\infty(x, y^{(1)}) = 4 \cdot 2^{-16}u \ll 2u = \theta_\infty(x, y^{(2)}).$$

Here, $y^{(1)}$ has a much smaller normwise relative error than $y^{(2)}$, even though $y^{(2)}$ has a smaller maximum componentwise relative error[1] than $y^{(2)}$ ($2u$ versus $4u$). This example illustrates the difficulty of capturing all the relevant information about the quality of the approximation $y \approx x$ in a single number.

Since any two norms on $\mathbb{R}^n$ differ by no more than a constant depending on n, we can obtain results similar to Lemma 2.2 for other norms. For example, for the 2-norm, if $\theta_2(x,y) < n^{-1/2}u$ then $\theta_\infty(x,y) < u$ and the conclusion of Lemma 2.2 holds.

One way to avoid the phenomenon of tiny normwise relative errors is to use a modified definition of relative error. For example, we can define

$$\operatorname{rel}(x,y) = \begin{cases} 0, & x = y, \\ \dfrac{\|x-y\|_\infty}{\max\{\,|x_k| : x_k \neq y_k\,\}}, & x \neq y. \end{cases}$$

This quantity is never smaller than u.

LEMMA 2.3. *Let $x, y \in \mathbb{R}^n$ be distinct vectors of normalized floating point numbers. Then* $\operatorname{rel}(x,y) \geq u$.

Proof. Let ℓ be an index for which $|x_\ell| = \max\{\,|x_k| : x_k \neq y_k\,\}$. Then $\operatorname{rel}(x,y) = \|x-y\|_\infty/|x_\ell| \geq |x_\ell - y_\ell|/|x_\ell| \geq u$, by Lemma 2.1. □

However, rel does not preserve monotonicity, in the sense that we can have $\theta_\infty(x,y^{(1)}) < \theta_\infty(x,y^{(2)})$ but $\operatorname{rel}(x,y^{(1)}) > \operatorname{rel}(x,y^{(2)})$. Indeed in the example (2) we have $\theta_\infty(x,y^{(1)}) \ll \theta_\infty(x,y^{(2)})$ but $4u = \operatorname{rel}(x,y^{(1)}) > \operatorname{rel}(x,y^{(2)}) = 2u$ (and rel is the same as the componentwise relative error in both cases). As the appropriate definition of relative error is problem and algorithm dependent, we will not pursue it further here. However, we note that the use of mixed absolute and relative errors in convergence tests for optimization codes and the effect on performance profiles is investigated by Dolan et al. [2006].

It is natural to try to extend this analysis to backward errors. Recall that the componentwise backward error of an approximate solution y to a linear system $Ax = b$ is given by [Higham 2002, Thm. 7.3], [Oettli and Prager 1964]

$$\begin{aligned} \omega_{E,f}(y) &:= \min\{\epsilon : (A+\Delta A)y = b + \Delta b,\ \ |\Delta A| \leq \epsilon E,\ \ |\Delta b| \leq \epsilon f\} \\ &= \max_i \frac{|r_i|}{(E|y| + f)_i}, \end{aligned}$$

where E and f are a nonnegative matrix and vector of tolerances, $|\Delta A| = (|\Delta a_{ij}|)$, and $r = b - Ay$ is the residual of y. Of most practical interest is the componentwise relative backward error with $E = |A|$ and $f = |b|$. Since for $a_{ij} \neq 0$ and $b_i \neq 0$ the inequalities in the definition of $\omega_{|A|,|b|}$ imply $\epsilon \geq |\Delta a_{ij}|/|a_{ij}|$ and $\epsilon \geq |\Delta b_{ij}|/|b_{ij}|$, it is tempting to argue that $\omega_{|A|,|b|} \geq u$, by Lemma 2.1. However, such an invocation is not valid because the lemma requires the data to be floating point numbers, and while the elements of A and b can be assumed to be floating point numbers those of ΔA and Δb cannot. Thus $\omega_{|A|,|b|}$ cannot be bounded below, and there is no simple characterization of when $\omega_{|A|,|b|} < u$ holds. Nevertheless we have observed that it is not unusual for componentwise or normwise backward errors to be orders of magnitude smaller than u (see, e.g., Davies et al. [2001, Sec. 5] and Khabou et al. [2012, Table 2.2]), so the development in the next section applies equally well to backward errors.

We turn now to the influence of tiny normwise relative errors on performance profiles.

[1]The componentwise relative error of y as an approximation to x is $\max_i(|x_i - y_i|/|x_i|)$.

Table I. Artificial data representing normwise relative errors.

Problem	Algorithm 1	Algorithm 2
1	4.0e-14	1.0e-16
2	6.0e-16	4.0e-16
3	1.0e-16	3.0e-16
4	9.0e-23	1.0e-17
5	6.0e-20	5.0e-17

3. PERFORMANCE PROFILES

Performance profiles provide a way to compare several different algorithms on a set of problems with respect to a performance measure such as run time, memory usage, or accuracy [Dolan and Moré 2002].

The performance ratio for an algorithm on a particular problem is the performance measure for that algorithm divided by the smallest performance measure for the same problem over all the algorithms. Here, we are assuming that the performance measure is one for which smaller is better. The performance profile is the set of functions $\{\, f_k(\alpha)\colon \alpha \in [1,\infty)\,\}$, where $f_k(\alpha)$ is the the proportion of problems where the performance ratio of the kth algorithm is at most α. Thus $f_k(\alpha)$ is a monotonically increasing function taking values in $[0,1]$. By plotting the curves $f_k(\alpha)$ on a single plot we can easily compare them and deduce information about the relative performance of the respective algorithms.

For $\alpha = 1$, $f_k(\alpha)$ reveals how often the kth algorithm was the best or joint best. Assume that failure to solve a problem (perhaps through non-convergence of an iterative method) is signalled by a value of ∞ for the performance measure. Then $\lim_{\alpha\to\infty} f(\alpha) < 1$ if the kth algorithm failed to solve at least one problem that was solved by another algorithm. Thus for large values of α the performance profile reveals the reliability of the algorithm.

Although originally introduced as a tool for evaluating and comparing optimization software, performance profiles are now widely used, and the second author has found them very useful in the context of algorithms for matrix functions (see, for example, Al-Mohy and Higham [2009; 2011; 2012], Higham [2005; 2008; 2009], Higham and Lin [2011]).

In the light of our observations in Section 1, it is natural to ask whether tiny normwise relative errors skew performance profiles. The following example shows that they can. Consider the artificial data in Table I, which represents normwise relative errors for two algorithms for solving $Ax = b$, implemented in IEEE double precision arithmetic. Specifically the data comprises plausible values of $\theta_\infty(x,y)$ where y is computed by the algorithm and the reference solution x is the true solution rounded to double precision. Suppose also that the systems are very well conditioned, so that for a numerically stable algorithm we can expect errors of order u. The performance profile is plotted in Figure 1. Since the curve for Algorithm 1 lies entirely above the curve for Algorithm 2 until the latter hits 1 at around $\alpha = 10^5$, it is tempting to conclude that Algorithm 1 has outperformed Algorithm 2 in the experiment. However, this is clearly not the case. The errors for Algorithm 2 are all less than $4u$, which is about the best that can be expected. However, Algorithm 1 performs unstably on problem 1, with an error of order $360u$, and it is also slightly less accurate than Algorithm 2 on problem 2. The performance profile has been unduly influenced by the tiny relative errors produced by Algorithm 1 on problems 4 and 5, for which Lemma 2.2 tells us that at least one component of the computed solution agrees exactly with that of the reference solution. The performance profile therefore provides an incomplete and possibly misleading impression of the relative merits of the two algorithms.

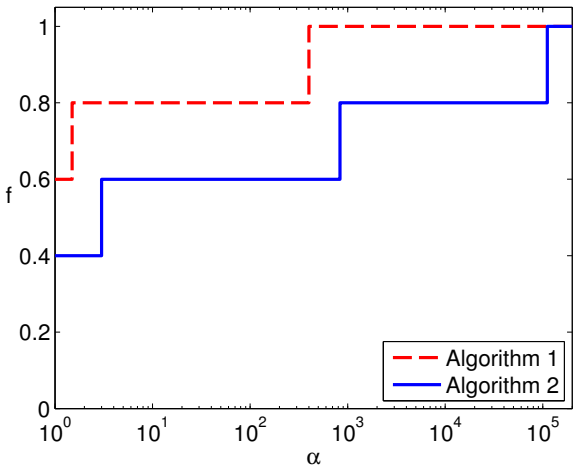

Fig. 1. Performance profile for data in Figure I.

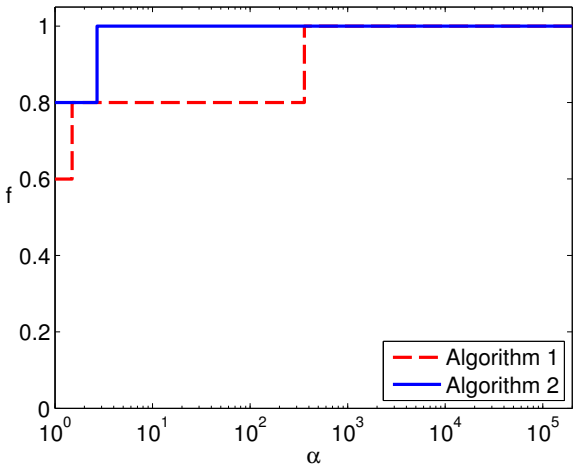

Fig. 2. Performance profiles for data in Figure I with entries below u rounded up to u.

One way to address this problem is to round up any value of $\theta_\infty(x, y)$ less than u up to u. Figure 2 shows the performance profile for the data transformed in this fashion. The curve for Algorithm 2 is now on or above that for Algorithm 1. At $\alpha = 1$ (where the curves intersect the y-axis) the relative ordering has changed; moreover, due to the introduction of ties in problems 4 and 5 the rounding results in $f_1(1) + f_2(1) > 1$, whereas $f_1(1) + f_2(1) = 1$ for the original data, as would usually be the case. Recall that the values at $\alpha = 1$ give the proportion of test problems on which a particular solver had the smallest performance measure, and so it is desirable not to change these values. This rounding strategy is too crude. Another possibility is to add u to every value of $\theta_\infty(x, y)$. However, while this preserves the ordering of the errors it makes all errors less than u cluster very close to u and so produces similar results to the previous approach in practice.

Table II. Transformed artificial data (rounded to two significant digits); numbers that have changed are denoted by *.

Problem	Algorithm 1	Algorithm 2
1	4.0e-14	1.0e-16*
2	6.0e-16	4.0e-16
3	1.0e-16*	3.0e-16
4	5.6e-18*	1.1e-17*
5	5.6e-18*	5.0e-17*

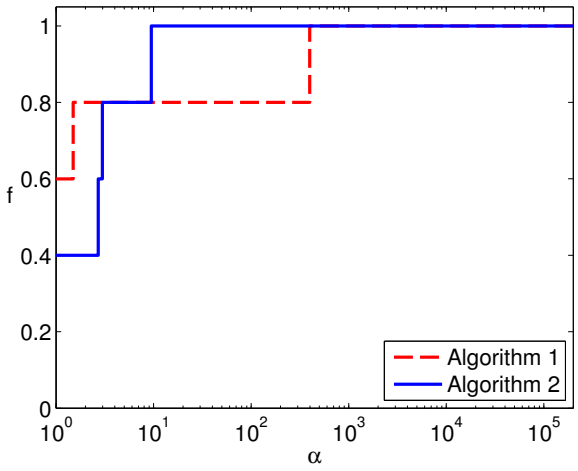

Fig. 3. Performance profile for data in Figure I with data transformed by f in (3).

A better strategy is to transform every data point s with $s < u$ to $f(s)$, where f is a strictly monotonically increasing function on $[0, u]$ with $a \le f(s) \le u$, for some positive parameter $a < u$, and $f(s) \ge s$. Such a transformation preserves the ordering of errors for each problem, ensures that no error less than u is decreased, and imposes a positive minimum value on the data points. We will take

$$f(s) = \begin{cases} q(s) & 0 \le s < u, \\ s, & s \ge u, \end{cases} \tag{3}$$

with q a linear function satisfying $q(0) = a$ and $q(u) = u$, so that $q(s) = s(1 - a/u) + a$. The user is therefore only required to choose the single parameter a. Setting $a = u$ corresponds to the crude rounding strategy discussed above.

With $a = 5 \times 10^{-2}u$ we applied the function (3) to the data in Table I, obtaining Table II. The corresponding performance profile is plotted in Figure 3. Note that the intersection of the curves with the y-axis is the same as for Figure 1. This performance profile gives a much clearer indication of the relative merits of the two algorithms: Algorithm 1 is most often the most accurate, but Algorithm 2 is at least as successful as Algorithm 1 at producing an error at most a factor α times the smallest (transformed) error for all $\alpha \ge 7$—in other words, Algorithm 2 is the more reliable.

Note that zero errors always present a dilemma in creating performance profiles, as they create an infinite performance ratio for any solver not having a zero error on the given problem. Transforming the data by f automatically solves this problem. To illustrate, we use the errors for `A\b` with the original matrices A reported in Section 1, along with corresponding errors for `A\b` followed by one step of iterative refinement

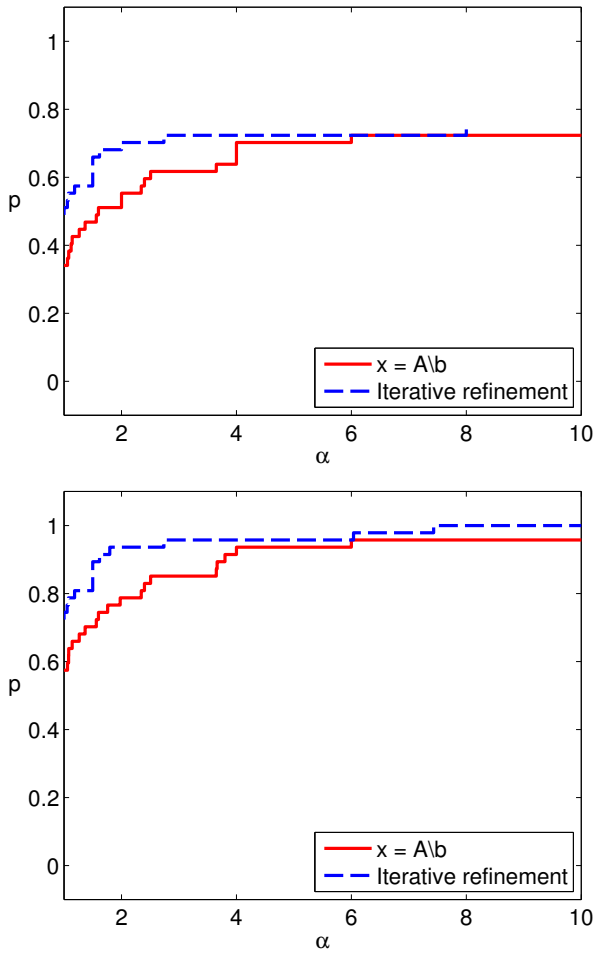

Fig. 4. Performance profiles for data from Section 1: raw data (top) and data transformed by (3) (bottom).

in fixed precision; this data includes a large number of zero errors. Figure 4 shows two performance profiles: the top one is for the raw data and the lower one is for the data transformed by (3) with $a = 5 \times 10^{-2}u$. Like all performance profiles in this paper, these were produced using function `perfprof` from Higham and Higham [2005, Sec. 22.4], which does not massage the input data in any way. The performance profile for the raw data suggests that the solvers are relatively unreliable, in that each solver is more than a factor 10 less accurate than the other on at least 20 percent of the problems. However, on 21 percent of the 47 problems both solvers produced zero error, and these cases are filtered out by `perfprof` as the performance ratio is undefined (as NaNs would be produced in the code's divisions), while in two problems one of the solvers produced a zero error and the other a nonzero error (a different solver in each case), resulting in two infinite performance ratios; all these cases result in an unreasonable lowering of the performance profile curves. Our scaling (3) leads to a performance profile that gives a truer representation of the performance of the solvers.

We give a final example that employs the data from Al-Mohy and Higham [2011, Exp. 1]. That experiment compares the normwise relative errors in the 2-norm of four different methods for computing the action of the matrix exponential on a vector, $e^A b$, on 155 problems. To simplify the interpretation we take just two of the methods, which we refer to as Algorithm 1 and Algorithm 2 (which are, respectively, Algorithm 3.1 and `expm_new` in Al-Mohy and Higham [2011], where `expm_new` is from Al-Mohy and Higham [2009]). There are 4 zero errors for Algorithm 1 and 16 for Algorithm 2, with zero errors for both on 3 problems. Algorithm 1 has error less than or equal to that of Algorithm 2 on 55 percent of the problems. Figure 5 shows three performance profiles: the top one is for the raw data, the middle one is for the data with all errors less than u rounded up to u, and the lower plot is for the data transformed by (3) with $a = 5\times10^{-2}u$. We make several observations.

— The performance profile for the raw data shows that for each algorithm the relative error is more than a factor 100 larger than that for the other algorithm in more than 10 percent of the cases. This statistic suggests unreliability, but it is misleading because it masks the fact that many of the errors in questions are much less than u.
— Rounding the errors up to u solves the reliability issue, but causes the performance profile for Algorithm 1 to lie entirely above that for Algorithm 2, which is also misleading.
— The transformation (3) leads to a performance profile giving a more balanced view. Algorithm 1 is most often the more accurate (as shown by the first and third plots, which have the same intercepts at $\alpha = 1$, namely 0.55 for Algorithm 1 and 0.47 for Algorithm 2), and although it is not as good as Algorithm 2 at producing errors within a factor 10–80 of the smallest, it is at least as good for factors 80 onwards.

Finally, we note that a performance profile is a cumulative distribution function of the performance ratio, and one could of course augment it with other statistical information for the data set, such as means and standard deviations. It may be preferable, however, to transform the data according to (3) before computing such statistics.

4. CONCLUSIONS

Transforming normwise relative errors via the function f in (3) prior to computing a performance profile based on the errors produces more meaningful profiles that greatly reduce the influence of tiny, or zero, errors, without affecting the monotonicity of the errors or the values at $\alpha = 1$ of the performance profile curves. This technique has been used successfully in [Al-Mohy et al. 2012] and [Higham and Lin 2013]. It can equally well be used on residuals, backward errors, and any other data where all values less than some tolerance (here u) can be regarded as roughly equally good.

Our transformation (3) contains a parameter a that must be chosen by the user. To aid the choice of a, we have developed a MATLAB GUI that allows a to be varied with a slider and redraws the performance profile as a changes. Experimentation with the GUI has shown that the precise value of a is not critical; the value $a = 5 \times 10^{-2}u$ used in our experiments works well in our experience.

Acknowledgements

We acknowledge the stimulating comments of the referees and Associate Editor that led to an improved paper.

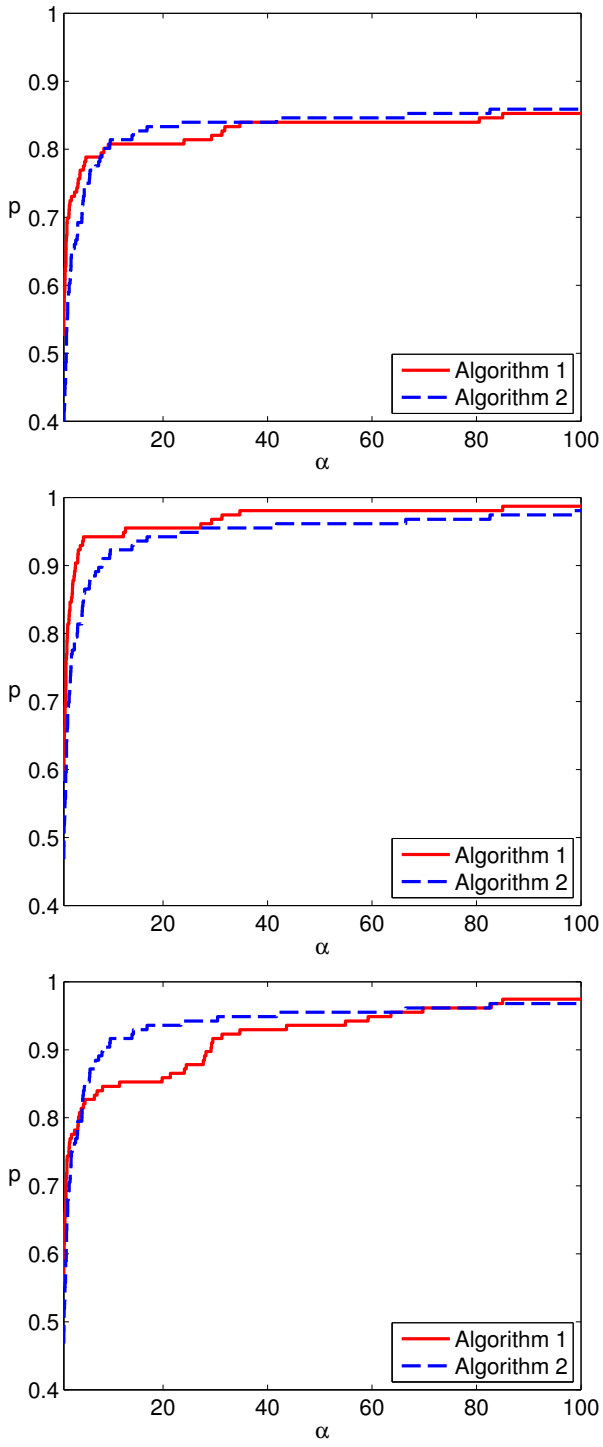

Fig. 5. Performance profiles for data from Al-Mohy and Higham [2011]: raw data (top), data with errors less than u replaced by u (middle), and data transformed by (3) (bottom).

REFERENCES

AL-MOHY, A. H. AND HIGHAM, N. J. 2009. A new scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl. 31,* 3, 970–989.

AL-MOHY, A. H. AND HIGHAM, N. J. 2011. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM J. Sci. Comput. 33,* 2, 488–511.

AL-MOHY, A. H. AND HIGHAM, N. J. 2012. Improved inverse scaling and squaring algorithms for the matrix logarithm. *SIAM J. Sci. Comput. 34,* 4, C153–C169.

AL-MOHY, A. H., HIGHAM, N. J., AND RELTON, S. D. 2012. Computing the Fréchet derivative of the matrix logarithm and estimating the condition number. MIMS EPrint 2012.72, Manchester Institute for Mathematical Sciences, The University of Manchester, UK. July. Revised December 2012.

DAVIES, P. I., HIGHAM, N. J., AND TISSEUR, F. 2001. Analysis of the Cholesky method with iterative refinement for solving the symmetric definite generalized eigenproblem. *SIAM J. Matrix Anal. Appl. 23,* 2, 472–493.

DEMMEL, J. W., HIDA, Y., KAHAN, W., LI, X. S., MUKHERJEE, S., AND RIEDY, E. J. 2006. Error bounds from extra-precise iterative refinement. *ACM Trans. Math. Software 32,* 2, 325–351.

DOLAN, E. D. AND MORÉ, J. J. 2002. Benchmarking optimization software with performance profiles. *Math. Programming 91*, 201–213.

DOLAN, E. D., MORÉ, J. J., AND MUNSON, T. S. 2006. Optimality measures for performance profiles. *SIAM J. Optim. 16,* 3, 891–909.

HIGHAM, D. J. AND HIGHAM, N. J. 2005. *MATLAB Guide* Second Ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

HIGHAM, N. J. The Matrix Computation Toolbox. `http://www.ma.man.ac.uk/~higham/mctoolbox`.

HIGHAM, N. J. 1997. Iterative refinement for linear systems and LAPACK. *IMA J. Numer. Anal. 17,* 4, 495–509.

HIGHAM, N. J. 2002. *Accuracy and Stability of Numerical Algorithms* Second Ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

HIGHAM, N. J. 2005. The scaling and squaring method for the matrix exponential revisited. *SIAM J. Matrix Anal. Appl. 26,* 4, 1179–1193.

HIGHAM, N. J. 2008. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

HIGHAM, N. J. 2009. The scaling and squaring method for the matrix exponential revisited. *SIAM Rev. 51,* 4, 747–764.

HIGHAM, N. J. AND LIN, L. 2011. A Schur–Padé algorithm for fractional powers of a matrix. *SIAM J. Matrix Anal. Appl. 32,* 3, 1056–1078.

HIGHAM, N. J. AND LIN, L. 2013. An improved Schur–Padé algorithm for fractional powers of a matrix and their Fréchet derivatives. MIMS EPrint 2013.1, Manchester Institute for Mathematical Sciences, The University of Manchester, UK. Jan.

KHABOU, A., DEMMEL, J., GRIGORI, L., AND GU, M. 2012. LU factorization with panel rank revealing pivoting and its communication avoiding version. Tech. Rep. UCB/EECS-2012-15, EECS Department, University of California, Berkeley. Jan.

OETTLI, W. AND PRAGER, W. 1964. Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides. *Numer. Math. 6*, 405–409.

OISHI, S., OGITA, T., AND RUMP, S. M. 2009. Iterative refinement for ill-conditioned linear systems. *Japan J. Indust. Appl. Math. 26*, 465–476.

PARLETT, B. N. 1998. *The Symmetric Eigenvalue Problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA. Unabridged, amended version of book first published by Prentice-Hall in 1980.

WILKINSON, J. H. 1965. *The Algebraic Eigenvalue Problem*. Oxford University Press.