Comput Stat
DOI 10.1007/s00180-009-0148-x

ORIGINAL PAPER

Reliability and efficiency of algorithms for computing the significance of the Mann–Whitney test

Niranjan Nagarajan · Uri Keich

Received: 21 March 2008 / Accepted: 10 March 2009

Abstract Motivated by recent applications of the Mann–Whitney U test to large data sets we took a critical look at current methods for computing its significance. Surprisingly, we found that the two fastest and most popular tools for exact computation of the test significance, Dinneen and Blakesley's and Harding's, can exhibit large numerical errors even in moderately large datasets. In addition, another method proposed by Pagano and Tritchler also suffers from a similar numerical instability and can produce inaccurate results. This motivated our development of a new algorithm, mw-sFFT, for the exact computation of the Mann–Whitney test with no ties. Among the class of exact algorithms that are numerically stable, mw-sFFT has the best complexity: $O(m^2 n)$ versus $O(m^2 n^2)$ for others, where m and n are the two sample sizes. This asymptotic efficiency is also reflected in the practical runtime of the algorithm. In addition, we also present a rigorous analysis of the propagation of numerical errors in mw-sFFT to derive an error guarantee for the values computed by the algorithm. The reliability and efficiency of mw-sFFT make it a valuable tool in compuational applications and we plan to provide open-source libraries for it in C++ and Matlab.

Keywords Numerical error · Exact computation · FFT

N. Nagarajan (✉)
CBCB, UMIACS, University of Maryland, College Park, MD 20742, USA
e-mail: niranjan@umiacs.umd.edu

U. Keich
Department of Computer Science, Cornell University, Ithaca, NY 14850, USA
e-mail: keich@cs.cornell.edu

Present Address:
U. Keich
School of Mathematics and Statistics, University of Sydney, Sydney, NSW 2006, Australia
e-mail: uri@maths.usyd.edu.au

Published online: 29 March 2009

1 Introduction

Computing the p value, or more generally the probability mass function (pmf) of the Mann–Whitney test statistic[1] has been an active area of research since its introduction in Mann and Whitney (1947). Some of the popular approaches to evaluating the test's p value are derived from large sample approximation: normal (Mann and Whitney 1947), Edgeworth corrected normal (Fix and Hodges 1955; Hodges et al. 1990), uniform (Buckle et al. 1969), and saddlepoint methods (Jin and Robinson 1999; Froda and van Eeden 2000; Jin and Robinson 2003). Others rely on exact methods where the pmf is computed directly from the underlying uniform distribution on permutations (Mann and Whitney 1947; Dinneen and Blakesley 1973; Pagano and Tritchler 1983; Harding 1984; Mehta et al. 1984; Streitberg and Rohmel 1984; Di Bucchianico 1999).

While the large-sample approximations have all been shown to be valid in some asymptotic domain, a user could find it difficult to figure out which method is applicable for a given set of parameters: the size of the two samples m, n and the value of the statistic k. Moreover, even if a certain asymptotic approximation is deemed "valid" it is typically difficult to give a bound on the error. An example of the problems with such approximations can be seen in Fig. 1. Exact methods are attractive as they offer to eliminate this uncertainty by replacing the approximation schemes with an exact computation of the p value. Algorithms for the Mann–Whitney test in this category typically have a runtime complexity of $O(m^2n^2)$ which can be rather slow for large m and n (Mann and Whitney 1947; Pagano and Tritchler 1983; Mehta et al. 1984; Streitberg and Rohmel 1984). The methods proposed in Dinneen and Blakesley (1973) and Harding (1984) have a more favorable runtime of $O(m^2n)$ and are the methods of choice in many statistical packages (STATISTICA, UNISTAT, SPSS, etc.).

As part of this study, we evaluated the various exact algorithms for the accuracy of their results. To our suprise, we found that both Dineen and Blakesley's and Harding's algorithm can be severly affected by accumulating roundoff errors. For example, when $m = n = 300$ and $k = 44,554$, Dineen and Blakesley's algorithm reports a p value that is *greater than 1* and more than 20 times the correct answer. In addition, we found that the relative error tends to become worse with increasing m and n. Similar problems were also found with Harding's algorithm and some illustrative examples of this can be seen in Table 1. The problems with numerical errors in both algorithms are due to the fact that they rely on subtle cancellations to compute their values and we discuss this in more detail in Sect. 2.1. Other algorithms such as one based on Mann and Whitney's original recursion formula [(1) of Mann and Whitney (1947) referred to as `REC` below] and the algorithms in Mehta et al. (1984) and Di Bucchianico (1999) were found to be relatively accurate. However, these algorithms all have a runtime complexity of $O(m^2n^2)$ or higher. In this work, we present the design of an algorithm which is robust to numerical errors while maintaining the desirable runtime complexity of $O(m^2n)$. In addition, we also show how an upper bound on the numerical error in our algorithm can be computed. Typically, the presentation of an algorithm

[1] As in the original work of Mann and Whitney we assume that ties occur with probability 0 and can be ignored.

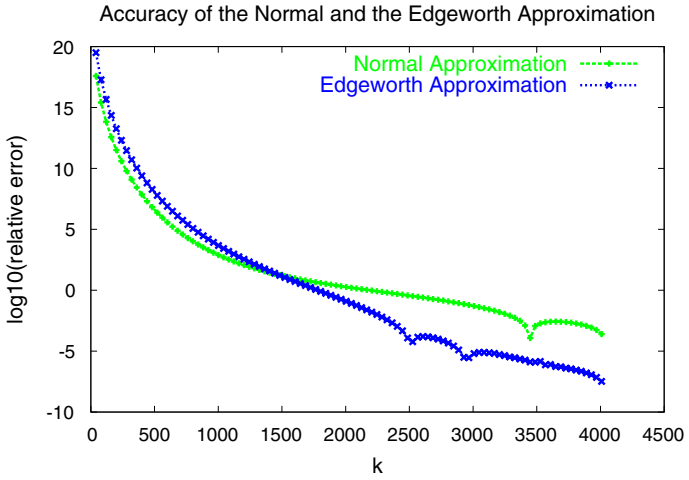

Fig. 1 Approximation error in the normal and edgeworth approximations. Here $m = n = 90$ and k varies over $\{\lfloor \frac{i}{101} * mn/2 \rfloor \mid i \in [1,\ldots, 100]\}$. As it can be seen, while the normal and edgeworth approximations are reliable for some values of k, they are particularly poor for others

Table 1 Numerical error in Harding's algorithm

m	n	k	p value	p value from Harding's algorithm
120	120	6,487	0.09	0.02
120	180	10,158	0.19	2.51
180	180	12,831	3.10×10^{-4}	0.45
540	540	95,275	8.10×10^{-24}	0.32

in computational statistics is not coupled with analysis of its numerical stability. Our results here suggest that this might be important and the approach presented in this work can serve as a useful template for other applications.

An important motivation for our work here is the recent development of various computational applications for the Mann–Whitney test. Some of the popular applications include, detecting changes in data streams (Kifer et al. 2004), identifying differentially expressed genes (Bickel 2004; Troyanskaya et al. 2002), identifying transcription factor binding sites (Karanam and Moreno 2004) and studying DNA copy number changes (Van de Wiel et al. 2005). In these applications m and n can be quite large (values greater than 1,000 are not uncommon) and the large amounts of data that need to be analyzed creates a need for fast algorithms. In applications such as data-stream analysis there is even a need for realtime response. In addition, to correct for the multiple hypotheses that are tested, these applications typically demand that even very small p values are accurately computed. For example, in tools for finding binding sites in DNA, this correction can easily be 10^{50} or larger (this is the size of

the search space). In particular, Monte–Carlo methods are unsatisfactory here as they require prohibitively large sample sizes to estimate the p values that are of interest.

The rest of this article is organized as follows: we first present a new, exact and reliable algorithm, mw-sFFT [runtime complexity of $O(m^2 n)$] for computing the significance of the Mann–Whitney test (Sects. 2.1, 2.2). This algorithm extends the characteristic function based approach of Pagano and Tritchler (1983) and uses novel techniques (Nagarajan et al. 2005; Keich and Nagarajan 2006) to avoid numerical errors in FFT based algorithms. In Sect. 2.3, we discuss the reliability of mw-sFFT and present a careful analysis of the propagation of errors in the algorithm and a proof of an upper bound on the numerical error. We then present results from our experiments that empirically confirm the reliability of mw-sFFT (Sect. 3.1), the utility of its error bounds (Sect. 3.2) and its efficiency in practice (Sect. 3.3). We conclude with some remarks about research directions to explore and the general applicability of our methods.

2 Methods

2.1 Preliminaries and Pagano and Tritchler's algorithm

Given two samples $X_1, \ldots, X_m$ and $Y_1, \ldots, Y_n$, the Mann–Whitney U test is a non-parametric test for the null hypothesis $F_X = F_Y$ against the alternative that X is stochastically smaller ($F_X \geq F_Y$; there are trivial extensions for the other two obvious alternatives). The test is based on the Mann–Whitney U statistic (or equivalently, the Wilcoxon rank sum statistic) (Mann and Whitney 1947) which is defined as

$$M_{m,n} := \sum_{i=1}^{m} \sum_{j=1}^{n} 1_{Y_j < X_i}$$

where

$$1_{Y_j < X_i} = \begin{cases} 1, & \text{if } Y_j < X_i \\ 0, & \text{otherwise.} \end{cases}$$

Our goal here is to compute the significance of the test under H_0 ($F_X = F_Y$) which is defined as

$$p\text{ value}(m, n, k) := P_0(M_{m,n} \leq k) = \sum_{s=0}^{k} P(M_{m,n} = s). \tag{1}$$

Since under the null-assumption, all orderings of $\{X_1, \ldots, X_m, Y_1, \ldots, Y_n\}$ are equally likely, one can verify that $P_0(M_{m,n} \leq k) = P_0(M_{m,n} \geq mn - k)$. Thus, we can assume without loss of generality that:

1. $m \leq n$ and
2. $k \leq mn/2$.

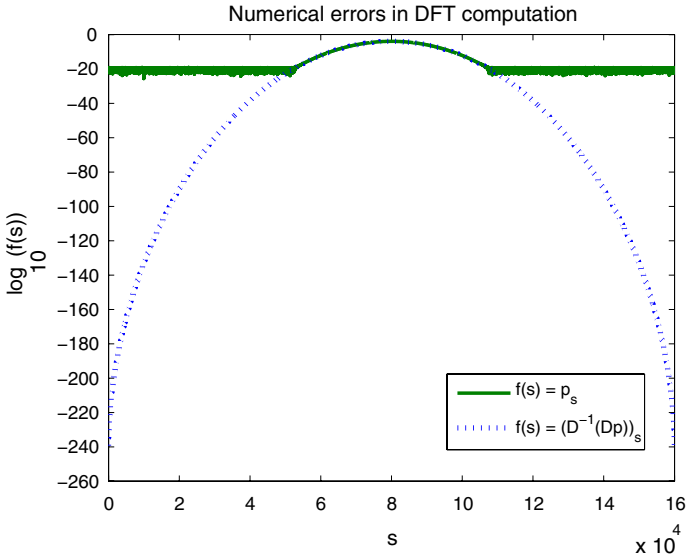

Fig. 2 The parameters used in this comparison are $m = n = 400$. While theoretically identical the two curves differ because the DFT's are computed using fixed precision arithmetic

Let:

- $p = (p_0, \ldots, p_{Q-1})$ be the pmf of $M_{m,n}$, where $Q = mn + 1$
- $M(\theta) := \sum_{s=0}^{Q-1} p_s e^{\theta s}$ be its moment generating function
- $\phi = (\phi_0, \ldots, \phi_{Q-1})$ be its characteristic function evaluated at the points $\{2\pi l/Q : l \in [0..Q-1]\}$.

Then from the definition of ϕ, $\phi = Dp$ where D is the Discrete Fourier Transform operator.[2]

Pagano and Tritchler (1983) proposed an algorithm that computes p and therefore the p value as follows: it first computes ϕ and then applies an FFT (Press et al. 1992) implemented D^{-1} to recover $p = D^{-1}\phi$. Their algorithm uses a recursive procedure to compute $\phi_l = M(i \cdot 2\pi l/Q)$ (where $i := \sqrt{-1}$) for each l in $O(mn)$ time and thus its overall runtime complexity is $O(m^2n^2)$. While Pagano and Tritchler's algorithm is not the most efficient algorithm in terms of runtime it suffers from a more serious drawback in that numerical errors in DFT calculations can dominate the p value that is computed. An example of this can be seen for $m = n = 400$ and $k = 4 \times 10^4$ where the algorithm yields a p value *greater than* 500 while the correct answer is $\approx 8.4 \times 10^{-37}$.

The basic source of errors in Pagano and Tritchler's algorithm can be seen in Fig. 2 where we plot p against $D^{-1}(Dp)$. Here, the DFT of p produces a weighted sum of entries in the range 10^{-5} to 10^{-250}. As machine precision for double precision arithmetic is only $\varepsilon_0 \approx 2.22 \times 10^{-16}$, the precision of entries less than $\varepsilon_0 \max_s p_s$ is not

[2] For a vector v of size Q, $(Dv)_l := \sum_{s=0}^{Q-1} v_s e^{i \cdot 2\pi ls/Q}$ and $(D^{-1}v)_l := 1/Q \sum_{s=0}^{Q-1} v_s e^{-i \cdot 2\pi ls/Q}$ and D, D^{-1} are linear transforms such that $v = D^{-1}(Dv)$ (Press et al. 1992).

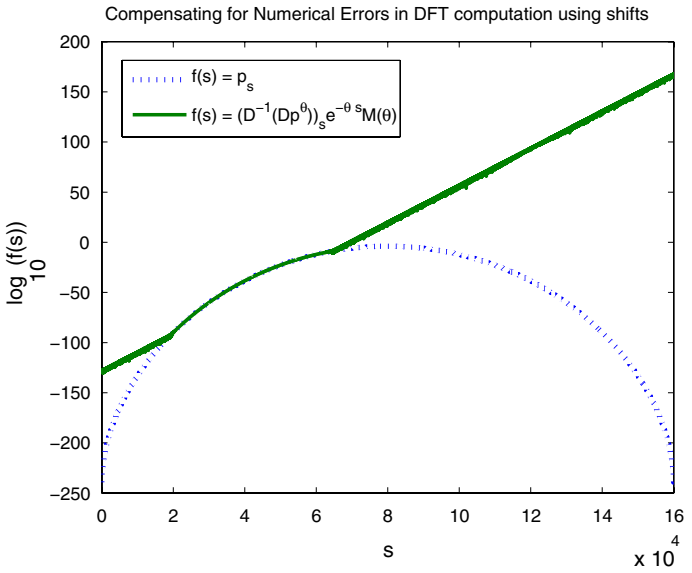

Fig. 3 The parameters used in this comparison are $m = n = 400$ and $\theta = -0.0043$. Note that while the curves in Fig. 2 diverge substantially around $k = 4 \times 10^4$, the curves here agree quite well in this region. This agreement suffices to accurately compute the p value for $k = 4 \times 10^4$

preserved in the sum. Correspondingly, these entries are not recovered using D^{-1} and computing, in this example, the p values of $k \in [0, 5 \times 10^4]$ cannot be achieved with Pagano and Tritchler's algorithm. This, in a sense, is also the source of numerical errors in Dineen and Blakesley's and Harding's algorithm; both the algorithms rely on subtle cancellations between large sums to compute smaller entries in p and hence numerical errors overwhelm these values. In the next section, we present an algorithm that avoids this pitfall while being as efficient as Dineen and Blakesley's and Harding's algorithm.

2.2 Avoiding roundoff errors: the mw-sFFT algorithm

The problem of numerical errors in Pagano and Tritchler's algorithm cannot be easily avoided; working with higher precision arithmetic only delays the onset of numerical errors to smaller p values and in addition adds a significant time penalty.[3] A more elegant and practical approach is to adopt the following shifted-FFT (or sFFT) solution from Keich (2005): consider a "shifted" version of p, denoted by p^θ, where

$$p^\theta_s = p_s e^{\theta s} / M(\theta).$$

We alter our goal to that of computing p^θ. To see the benefit of this consider Fig. 3 where we compare p_s with $(D^{-1}(Dp^\theta))_s e^{-\theta s} M(\theta)$. As can be seen, for the given

[3] Arbitrary precision arithmetic does even worse; it increases the time complexity of the algorithm, with a runtime that now grows with the size of the numbers as well. In addition, the lack of hardware support on most machines increases the constant that is hidden in the big-O notation.

Fig. 4 The Mann–Whitney sFFT algorithm

MW-sFFT(m, n, k)
1 $\theta \leftarrow \theta(k)$ (by solving Eq. 3)
2 $Q \leftarrow mn + 1$
3 for $l \leftarrow 0$ to $Q - 1$
4 do $\phi_l^\theta \leftarrow M(\theta + i \cdot 2\pi l/Q)/M(\theta)$ (using Eq. 4)
5 $p^\theta \leftarrow D^{-1}\phi^\theta$ (using FFT)
6 return $\sum_{s=0}^{k} p_s^\theta e^{-\theta s} M(\theta)$

value of θ, we can recover p accurately for entries around $k = 4 \times 10^4$ and thus hope to compute p value$(400, 400, 4 \times 10^4)$. The magic here is that the shift in the distribution redistributes its mass in a way such that the numerical errors introduced by the DFT do not affect the values of interest to us.

But this begs the question, "How do we compute p^θ in the first place?" The solution is provided by the following procedure: let $\phi^\theta := Dp^\theta$. Then it can be shown with a bit of algebra that

$$\phi_l^\theta = M(\theta + i \cdot 2\pi l/Q)/M(\theta). \tag{2}$$

We can then compute p^θ as $D^{-1}\phi^\theta$ and $p_s = p_s^\theta e^{-\theta s} M(\theta)$.

An important part of this solution is to choose θ wisely. Here we rely on the following large deviation procedure[4] (Dembo and Zeitouni 1998):

$$\theta(k) = \text{argmin}_\theta \log M(\theta) - \theta k. \tag{3}$$

This choice can be shown to center the shifted distribution on k and intuitvely this should help to minimize the numerical error in recovering the values of p^θ about k. Further support for the robustness of this procedure is also provided by the experiments in Sect. 3.1.

The outline of the sFFT algorithm for the Mann–Whitney test (or mw-sFFT) is presented in Fig. 4. This algorithm improves over the algorithm by Pagano and Tritchler by compensating for the numerical roundoff errors inherent in the DFT calculations. In addition, we propose the use of the following formula (van Dantzig 1947–1950) to compute the moment generating function:

$$M(\theta) = \prod_{j=1}^{m} \frac{j}{n+j} \frac{1 - e^{\theta(n+j)}}{1 - e^{\theta j}}. \tag{4}$$

This reduces the cost of computing the moment generating function from $O(mn)$ in Pagano and Tritchler's algorithm (based on a recursive computation) to $O(m)$. However, there are several practical issues regarding numerical stability that need to be addressed in order to use this formula and we discuss these in the appendix. The corresponding improved runtime for mw-sFFT is $O(m^2 n + mn \log mn)$, and typically

[4] The minimization in the procedure can be carried out numerically by using, for example, Brent's method (Press et al. 1992). Also, it is not difficult to see that for $k < \mu = mn/2$, $\theta(k) < 0$.

this is $O(m^2 n)$, making it asymptotically the fastest known, numerically robust, exact algorithm. The space complexity for mw-sFFT is $O(mn)$ which is the same as that for Dineen and Blakesley's and Harding's algorithm but an improvement over the $\Omega(m^2 n)$ space complexity for Mann and Whitney's original REC [and related algorithms such as Mehta et al. (1984), Streitberg and Rohmel (1984) and Di Bucchianico (1999)].

2.3 Analyzing the numerical error and providing error bounds

As we proposed in the introduction, the numerical stability of algorithms computing statistical significance is an important aspect that we ignore at our own peril. Proving that an algorithm is numerically stable can be tricky and can sometimes aid in uncovering significant problems in it. While empirical evidence for the reliability of an algorithm can be reassuring, theoretical error bounds help guard against special cases. In the case of mw-sFFT we were able to conduct a careful analysis of the propagation of errors in the algorithm and derive an upper bound on the error in the p value computed. The essence of this bound is the following lemma:

Lemma 1

$$|fl(p_s) - p_s| \leq (mC_3 + C_5 \log Q)\varepsilon_0 e^{\log M(\theta) - \theta s} + fl(p_s)\varepsilon_0 + O({\varepsilon_0}^2)$$

where $C_3 \leq 83, C_5 \leq 20$ *are universal constants,* $fl(p_s)$ *is the approximation for* p_s *computed using mw-sFFT (for any* $\theta < 0$*) and* $\varepsilon_0 \approx 2.2 \times 10^{-16}$ *using double precision arithmetic.*

Remark 1 The lemma provides additional justification for our choice of θ as the error bound in the lemma is essentially minimized [the relative error term of $fl(p_s)\varepsilon_0$ is typically negligible] at k when θ is set to $\theta(k)$.

We provide a detailed proof of this important lemma in the appendix. The lemma is basically derived by proving that if we compute ϕ_l^θ (characteristic function of the shifted distribution) carefully then the numerical error can be bounded by $mC_3\varepsilon_0$ (Lemma 6). Recall here that machine precision is only ε_0. Combining this result with a bound on the errors introduced by the DFT provides us with a good error guarantee for p^θ and correspondingly for p. A direct corollary of Lemma 1 is the following error bound on the computed p value:

Corollary 1

$$|fl(p\,value(m,n,k)) - p\,value(m,n,k)| \leq \sum_{s=0}^{k} (mC_3 + C_5 \log Q)\varepsilon_0 e^{\log M(\theta(k)) - \theta(k)s} + k \times fl(p\,value(m,n,k))\varepsilon_0 + O({\varepsilon_0}^2). \tag{5}$$

where $fl(p\,value(m,n,k))$ *is the approximation for* $p\,value(m,n,k)$ *computed using mw-sFFT.*

As we show in Sect. 3.2, these error bounds are not too conservative and are useful in practice to guarantee that the reported p values are sufficiently accurate. In the next section, we report our results from the empirical evaluation of mw-sFFT and competing algorithms, starting with an analysis of their reliability.

3 Results

3.1 Empirical accuracy of mw-sFFT

In order to test the accuracy of the different algorithms for computing the p value of the Mann–Whitney test we need a provably accurate algorithm that would establish the gold standard. One such solution is offered by a significantly slower version of `REC` that we implemented using arbitrary precision integer arithmetic. We then tested mw-sFFT over a range of test parameters as given in Table 2. On this test set of 3,600 parameter combinations, we found that the p value computed using mw-sFFT had a relative error of less than $10^{-11.8}$ in all cases (we report the absolute value of the relative error, i.e., the relative error of a in comparison to b is $|(a-b)/b|$). This high level of accuracy demonstrated by mw-sFFT is reassuring and is also confirmed by the error bounds (see Sect. 3.2). We are currently working on techniques to make mw-sFFT more flexible and allow a trade-off of some of this accuracy for an improved runtime, based on a user-determined parameter.

As a more unstructured test of our algorithm, we also used the local search procedure `fminsearch` in MATLAB to search the space of parameters and find parameters where the relative error in the p value is maximized. Based on 100 random start points (obtained by first choosing $m \in [0, \ldots, 200]$ and $n \in [m, \ldots, 200]$ uniformly and then choosing $k \in [0, \ldots, mn/2]$ uniformly), the worst cases found were $m = 45, n = 133, k = 2,992$ with a relative error of $10^{-11.5}$. Interestingly, this is quite close to our worst case error from the structured test as well.

Further evidence of the stability of mw-sFFT can be found in Fig. 5a where we plot mean relative error as a function of $m = n$ and in Fig. 5b where we plot relative error as a function of k for $m = n = 90$. Here we also compared mw-sFFT against `REC` implemented using standard double precision arithmetic (not to be confused with the "gold standard" version which uses arbitrary precision arithmetic) and found that they have similar accuracy, where mw-sFFT is actually more accurate in many cases.

Table 2 Range of test parameters

Parameter	Values
m	15, 30, 45, 60, 90, 120, 180, 300, 540
n	15, 30, 45, 60, 90, 120, 180, 300, 540
k	$\lfloor \frac{i}{101} * mn/2 \rfloor,\ i \in [1,\ldots,100]$

Note that the condition $m \leq n$ is always enforced

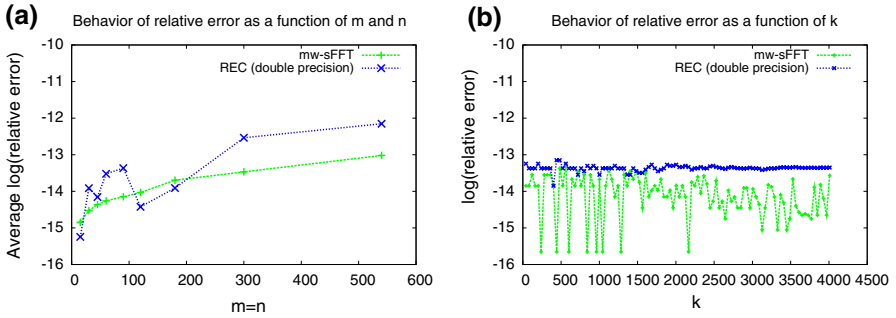

Fig. 5 Relative error in p values computed by mw-sFFT as a function of the parameters. The logarithms were taken with base 10. For comparison, we also present the results for REC implemented with double precision arithmetic. **a** The mean here was taken over the k values in Table 2. **b** Here $m = n = 90$

3.2 Employing the error bounds

The error bound in Corollary 1 helps us derive a bound on the relative error in the p value computed by mw-sFFT as follows: Let the bound from Corollary 1 be α. Then assuming $fl(p\,\text{value}(m, n, k)) > \alpha$, we have:

$$\text{Relative error} \leq \left| \frac{\alpha}{fl(p\,\text{value}(m, n, k)) - \alpha} \right|.$$

Testing these bounds on the set of parameters described in Table 2 we found that they guarantee that the relative error for the computed p value is less than $10^{-6.2}$ in the worst case (this also agrees with the worst case relative error of $10^{-11.8}$ in practice). Note that using the precomputed value for $M(\theta)$ in mw-sFFT, these error bounds can be calculated in constant time. Thus, they provide a quick check for the accuracy of the computed p value. As a sanity check, we also verified that the error bounds are indeed greater than the observed relative errors in all the tests that we performed.

A limitation of the tests that we performed for parameters in Table 2 was that they were restricted to $m, n < 600$. This was due to the fact that the computational and space complexity of known reliable algorithms (such as REC with arbitrary precision arithmetic) make it infeasible to run extensive tests beyond this point. To work around this, we used the analytical error bounds to establish a conservative estimate of the accuracy of mw-sFFT. We tested mw-sFFT over the set of large parameters given in Table 3 and found that the error bounds guarantee the relative error to be less than

Table 3 Range of test parameters with large m and n

Parameter	Values
m	800, 1,200, 2,500, 5,000
n	800, 1,200, 2,500, 5,000
k	$\lfloor \frac{i}{6} * mn/2 \rfloor$, $i \in [1 \ldots 5]$

Note that, as before, the condition $m \leq n$ is enforced

$10^{-4.5}$ in all cases. However, as before, the actual values reported by mw-sFFT are likely to be much more accurate in this range.

3.3 Runtime comparison of algorithms

As discussed in Sect. 2.2, mw-sFFT provides a clear improvement in runtime complexity compared to other reliable algorithms [$O(m^2n)$ versus $O(m^2n^2)$ for the algorithms in Mann and Whitney (1947), Mehta et al. (1984) and Streitberg and Rohmel (1984)]. This efficiency is also evident in our empirical results as can be seen in Fig. 6a. Here we used REC (implemented with double precision arithmetic) as a representative of the family of algorithms with an $O(m^2n^2)$ complexity (these algorithms are essentially variations on a dynamic programming algorithm and they should have similar runtime behavior in optimized implementations). For the sake of comparison we also included runtimes for an implementation of Dineen and Blakesley's algorithm in the figure. Not surprisingly, mw-sFFT and Dineen and Blakesley's algorithm have similar runtimes stemming from their similar runtime complexity of $O(m^2n)$. The important difference between them is that mw-sFFT is much more reliable.

The runtimes for Dineen and Blakesley's algorithm and REC vary with k and so we report average values in Fig. 6a. The runtime curves in Fig. 6b illustrate the typical behavior of these algorithms for fixed m and n and varying k. As can be seen here, both REC and Dineen and Blakesley's algorithm exploit optimizations in runtime that are possible for some values of k. Similar improvements are also possible in mw-sFFT [based on the idea in Nagarajan et al. (2005)] and we are currently working to acheive this. In Table 4 we also present some runtime examples that demonstrate the utility of mw-sFFT over algorithms with an $O(m^2n^2)$ complexity, especially for large m and n. In applications where large datasets are being processed these runtime gains can make the difference between a feasible and an infeasible solution.

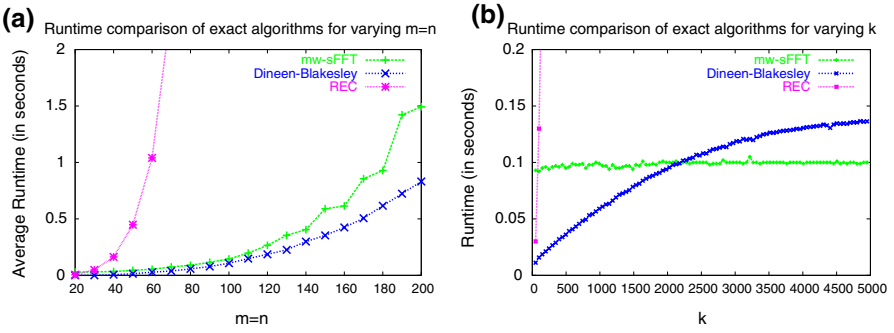

Fig. 6 Runtime comparison of mw-sFFT, REC and Dineen and Blakesley's algorithm for varying m, n and k. The comparison here was done using optimized C implementations on an Intel Xeon 2GHz processor. **a** The runtimes reported here are averaged over ten evenly spaced values for k in the range $0, \ldots, mn/2$. **b** Here $m = n = 100$

Table 4 Runtime examples for mw-sFFT and REC

m	n	k	mw-sFFT	REC
10	10	10	0.01	<0.01
100	100	100	0.10	0.12
100	100	4,000	0.10	12.02
500	500	1,00,000	18.77	2096.33
1,000	1,000	4,00,000	180.19	$>3\times 10^4$

Note that the runtimes reported here are in seconds

4 Concluding remarks

The increasing use of statistical tests in computational applications has created a new demand for reliable and efficient algorithms that can handle a much wider range of parameters of interest. In this context, the somewhat neglected aspect of "numerical error of an algorithm" also becomes relevant. In our work here, we have tried to address both these issues in the context of the Mann–Whitney test. There is much work that remains to be done in extending these ideas to other tests of interest in computational applications.

There are several features that we would like to add to mw-sFFT. In particular, the ability to tradeoff its high accuracy for improved runtimes could lead to a significantly faster and still reliable algorithm. In addition, exploiting optimizations similar to those in Dineen and Blakesley's algorithm could make the algorithm even more efficient. Finally, it should be noted that mw-sFFT can also be applied to the tied-case of the Mann–Whitney test (the moment generating function can be computed using the procedure in Pagano and Tritchler's algorithm), albeit without the error guarantees shown here. However, we believe that proving similar error bounds should be a feasible task. This is an important direction for our future work as existing algorithms for this problem do not come with guarantees on the numerical error.

5 Appendix: Error bound for mw-sFFT

The main result for this section is the proof of Lemma 1 which bounds the absolute error in computing the entries of the pmf p using the mw-sFFT algorithm. In what follows we use the notation $fl(x)$ for the machine estimator of x, $\varepsilon_x := x - fl(x)$ for the error in the machine estimate and $rel(x) := |(fl(x) - x)/x|$ for the relative error in the estimate [both $fl(x)$ and $rel(x)$ implicitly assume a specific computational procedure].

As is common, we assume that for $x, y \in \mathbb{R}$ with $x = fl(x)$ and $y = fl(y)$ and for any arithmetical operation $x \circ y$ where $\circ \in \{+, -, \times, \div\}$, $rel(x \circ y) \leq \varepsilon_0$, where ε_0 is the machine precision (in particular, $fl(x \circ y) = 0$ if $x \circ y = 0$). Based on these assumptions it can be shown that for x as above and for $w, z \in \mathbb{Z}$ with $w = fl(w)$ and $z = fl(z)$:

$$\begin{aligned} rel(w+z) &\le \varepsilon_0 \\ rel(wz) &\le C_1(\varepsilon_0+\varepsilon_0^2) = C_1\varepsilon_0 + O(\varepsilon_0^2), \\ rel(xz) &\le \varepsilon_0 \end{aligned} \tag{6}$$

where $C_1 = 4\sqrt{3}/3 \approx 2.31$ (Tasche and Zeuner 2002). Since for $z \neq 0$, $w/z = w\bar{z}/|z|^2$, where $\bar{z}$ is the complex conjugate of z, it follows that

$$rel(w/z) \le (C_1+3)\varepsilon_0 + O(\varepsilon_0^2). \tag{7}$$

An analysis of the accumulated error concentrates on the propagation of the roundoff errors throughout the computation. The following basic lemmas, presented here for completeness, facilitate this analysis.

Lemma 2 *For $x, y \in \mathbb{R}$ and $w = x + iy, z \in \mathbb{C}$ and assuming the relative error of all these variables is $O(\varepsilon_0)$ we have:*

$$\begin{aligned} rel(w) &\le \max\{rel(x), rel(y)\} \\ rel(xz) &\le rel(x) + rel(z) + \varepsilon_0 + O(\varepsilon_0^2) \\ rel(wz) &\le rel(w) + rel(z) + C_1\varepsilon_0 + O(\varepsilon_0^2) \\ rel(w/z) &\le rel(w) + rel(z) + (C_1+3)\varepsilon_0 + O(\varepsilon_0^2). \end{aligned}$$

Finally, if $w, z \in \mathbb{C}$ with ε_z and ε_w bounded by $O(\varepsilon_0)$:

$$|\varepsilon_{w-z}| \le \varepsilon_0 |w-z| + |\varepsilon_w| + |\varepsilon_z| + O(\varepsilon_0^2).$$

Proof Using Eq. 6 and Eq. 7 the proof is elementary and analogous, for example, to that for Lemma 3 in Keich (2005). □

For a function f of a variable x let $\Delta_f(x) := fl(f(fl(x))) - f(x)$ so that the relative error in computing $f(x)$ is given by $rel(f(x)) := \left|\Delta_f(x)/f(x)\right|$. We assume that for standard operators, such as $f \in \mathcal{F} := \{\sin, \cos, \sinh, \cosh, \exp\}$, $rel(f(x)) \le \varepsilon_0$ when $\varepsilon_x = 0$, however in the typical case of interest when $\varepsilon_x \neq 0$ these operators might significantly increase the relative error. The following lemma addresses this issue for the operators that we use to compute ϕ^θ (as defined in Eq. 2).

Lemma 3 *Suppose that for $x \in \mathbb{R}$ $|\varepsilon_x| = O(\varepsilon_0)$, then:*

1. *For $f \equiv \exp$ or $f \equiv \cosh$,*

$$rel(f(x)) \le \varepsilon_0 + |\varepsilon_x| + O(\varepsilon_0^2)$$

2. *For $f \equiv \sinh$, $x \in (-1, 1)$*

$$rel(f(x)) \le \varepsilon_0 + C_{\sinh} rel(x) + O(\varepsilon_0^2),$$

where $C_{\sinh} \le \cosh(1)$ (the rhs is not optimal).

3. *For* $f \equiv \sin$ *and* $x \in [-\pi/2, \pi/2]$,

$$rel(f(x)) \leq \varepsilon_0 + C_{\sin} rel(x) + O(\varepsilon_0^2),$$

where $C_{\sin} \leq \pi/2$ (*again, the rhs is by no means optimal*).

Remark 2 It follows from the proof below that the $O(\varepsilon_0^2)$ remainder term depends on x only possibly through ε_x (for item 2 this requires assuming $rel(x) = |\varepsilon_x/x| \leq C$ which holds in our calculations).

Proof For any standard real operator $f \in \mathcal{F}$ we have

$$\begin{aligned} \left|\Delta_f(x)\right| &\leq |f(x) - f(fl(x))| + |f(fl(x)) - fl(f(fl(x)))| \\ &\leq |f(x) - f(fl(x))| + \varepsilon_0 \, |f(fl(x))| \\ &\leq |f(x) - f(fl(x))| \, (1 + \varepsilon_0) + \varepsilon_0 \, |f(x)| \, . \end{aligned}$$

It follows that

$$rel(f(x)) \leq \varepsilon_0 + \underbrace{\left| \frac{f(x) - f(fl(x))}{f(x)} \right|}_{Q} (1 + \varepsilon_0).$$

The lemma follows from a straightforward analysis of Q:

1. For $f \equiv \exp$

$$Q = |1 - \exp(-\varepsilon_x)| = |\varepsilon_x| + O(\varepsilon_0^2).$$

For $f \equiv \cosh$, using the identity $\cosh(x + y) = \cosh x \cosh y + \sinh x \sinh y$ we have

$$Q = |(1 - \cosh \varepsilon_x) - \tanh x \sinh \varepsilon_x| \leq |\varepsilon_x| + O(\varepsilon_0^2).$$

2. For $f \equiv \sinh$, using the expansion of $\sinh(x + y)$ and assuming $x \in (-1, 1)$ we have

$$Q = \left| (1 - \cosh \varepsilon_x) - \cosh x \cdot \frac{x}{\sinh x} \cdot \frac{\sinh \varepsilon_x}{\varepsilon_x} \cdot \frac{\varepsilon_x}{x} \right| \leq \cosh(1) rel(x) + O(\varepsilon_0^2).$$

3. Similarly, for $f \equiv \sin$ and $x \in (-\pi/2, \pi/2)$ we use the analogous expansion of $\sin(x + y)$ to get

$$Q = \left| (1 - \cos \varepsilon_x) - \cos x \cdot \frac{x}{\sin x} \cdot \frac{\sin \varepsilon_x}{\varepsilon_x} \cdot \frac{\varepsilon_x}{x} \right| \leq \frac{\pi}{2} rel(x) + O(\varepsilon_0^2).$$

□

Lemma 4 *Let $l, Q \in \mathbb{N}$ and let $f_Q(l) = \sin(\pi l/Q)$ and let $g_Q(l) = \cos(\pi l/Q)$. We can compute $f_Q(l)$ and $g_Q(l)$ with a relative error less than $(\pi+1)\varepsilon_0 + O(\varepsilon_0^2)$ (ignoring the possibility of either l or Q creating an integer type overflow).*

Proof Consider first $f_Q(l)$ and to simplify the notations assume that $Q \equiv 0 \pmod 2$. Since $f_Q(l)$ depends only on $l \bmod Q$ we can assume that $l \in \{0, 1, \ldots, 2Q\}$. Moreover, using the trigonometric identities $\sin(x) = \sin(\pi - x)$ and $\sin x = -\sin(\pi + x)$ we can further assume without loss of generality that $l \in \{0, 1, \ldots, Q/2\}$. Let $x = \pi l/Q \in [0, \pi/2]$ then $rel(x) \le 2\varepsilon_0$ and by Lemma 3

$$rel(f_Q(l)) \le \varepsilon_0 + \pi\varepsilon_0 + O(\varepsilon_0^2).$$

Finally, we can compute $g_Q(l)$ from the identity $g_Q(l) = f_Q(l + Q/2)$. □

Remark 3 In addition to computing $f_Q(l)$ and $g_Q(l)$ as indicated above, for numerical stability, we compute $f(z) = e^z - 1$ for $z = x + iy$, $y = 2\pi l/Q$, $x \in (-\infty, 0)$ and $l, Q \in \mathbb{N}$ as follows: for $x \le -1$ we use

$$f(z) = e^x(\cos(y) + i\sin(y)) - 1 \tag{8}$$

while for $x \in (-1, 0)$ we use

$$\begin{aligned} f(z) &= 2e^{z/2}\sinh(z/2) \\ &= 2e^{x/2}(\cos(y/2) + i\sin(y/2))\cdot \\ &\quad (\sinh(x/2)\cos(y/2) + i\cosh(x/2)\sin(y/2)). \end{aligned} \tag{9}$$

Note that Eq. 8 is used for $x \le -1$ as it is more efficient to compute than Eq. 9.

Lemma 5 *Let x, y, z and $f(z)$ be as defined above in Remark 3 and assume that $rel(x) \le \varepsilon_0 + O(\varepsilon_0^2)$. Then,*

$$rel(f(z)) \le C_2\varepsilon_0 + O(\varepsilon_0^2)$$

where $C_2 < 18$ is a universal constant.

Proof If $x \in (-1, 0)$, we compute $f(z)$ using Eq. 9. From Lemma 3 and $\varepsilon_{x/2} = \varepsilon_x/2$

$$rel(e^{x/2}) \le \varepsilon_0 + |\varepsilon_x|/2 + O(\varepsilon_0^2) \le \varepsilon_0(1 + |x|/2) + O(\varepsilon_0^2) < 2\varepsilon_0 + O(\varepsilon_0^2).$$

By Lemmas 4 and 2

$$rel(e^{iy/2}) \le (\pi+1)\varepsilon_0 + O(\varepsilon_0^2), \tag{10}$$

and therefore by Lemma 2

$$rel(e^{z/2}) \le (\pi+4)\varepsilon_0 + O(\varepsilon_0^2). \tag{11}$$

Similarly, from Lemma 3

$$\begin{aligned} rel(\sinh(x/2)) &\le \varepsilon_0(1+\cosh(1)) + O(\varepsilon_0^2) \\ rel(\cosh(x/2)) &\le \varepsilon_0(1+|x|) + O(\varepsilon_0^2) \le 2\varepsilon_0 + O(\varepsilon_0^2), \end{aligned}$$

and therefore from Lemmas 4 and 2

$$\begin{aligned} rel\left(\sinh(x/2)\cos(y/2)\right) &\le \varepsilon_0(3+\pi+\cosh(1)) + O(\varepsilon_0^2) \\ rel\left(\cosh(x/2)\sin(y/2)\right) &\le \varepsilon_0(4+\pi) + O(\varepsilon_0^2). \end{aligned}$$

Hence by Lemma 2 and Eq. 11 above for $x \in (-1, 0)$:

$$rel(f(z)) = rel\left(e^{z/2}\sinh(z/2)\right) \le \left(C_1 + 7 + 2\pi + \cosh(1)\right)\varepsilon_0 + O(\varepsilon_0^2).$$

For $x \le -1$, $rel(e^x)$ can be rather large so we have to take a slightly different route. Again, let $\Delta_h(u) := h(u) - fl(h(fl(u)))$. It follows from Lemma 3 and from $e^x < 1$ and $|xe^x| < 1$ that

$$\left|\Delta_{\exp}(x)\right| \le e^x\left[\varepsilon_0 + |x|\varepsilon_0 + O(\varepsilon_0^2)\right] \le 2\varepsilon_0 + O(\varepsilon_0^2).$$

Analagous to Eq. 10 we have

$$\left|\Delta_{\exp}(iy)\right| \le (\pi+1)\varepsilon_0 + O(\varepsilon_0^2),$$

and therefore from Lemma 2

$$\left|\Delta_{\exp}(z)\right| \le (\pi+4)\varepsilon_0 + O(\varepsilon_0^2).$$

Hence by the last part of Lemma 2:

$$\left|\Delta_f(z)\right| \le (\pi + 4 + |f(z)|)\varepsilon_0 + O(\varepsilon_0^2),$$

and since $|f(z)| \ge 1 - e^{-1}$ we find that for $x \le -1$

$$rel\left(f(z)\right) \le \left(\frac{\pi+4}{1-e^{-1}} + 1\right)\varepsilon_0 + O(\varepsilon_0^2).$$

□

From Eq. 2 and Eq. 4, ϕ^θ, the DFT of the shifted pmf, p^θ, is given by:

$$\phi_l^\theta = \prod_{j=1}^{m} \underbrace{\frac{1-e^{(n+j)(i\cdot 2\pi l/Q+\theta)}}{1-e^{j(i\cdot 2\pi l/Q+\theta)}}\frac{1-e^{j\theta}}{1-e^{(n+j)\theta}}}_{\phi_{l,j}^\theta}.$$

Lemma 6 *If* $\theta < 0$

$$|fl(\phi_l^\theta) - \phi_l^\theta| \leq mC_3\varepsilon_0 + O(\varepsilon_0^2),$$

where $C_3 < 83$ *is a universal constant.*

Proof Note that the error in the numerically computed θ using Eq. 3 might be relatively large. However, since θ is only an auxillary variable whose exact value is of no interest to us we can assume without loss of generality that $rel(\theta) = 0$. Applying Lemmas 5 and 2 we get:

$$rel(\phi_{l,j}^\theta) \leq C_4\varepsilon_0 + O(\varepsilon_0^2),$$

where $C_4 = 4C_2 + C_1 + 5 < 80$. So from $m - 1$ applications of Lemma 2 and the fact that $|\phi_l^\theta| \leq 1$, we have,

$$\begin{aligned} |fl(\phi_l^\theta) - \phi_l^\theta| &\leq mC_4\varepsilon_0 + (m-1)C_1\varepsilon_0 + O(\varepsilon_0^2) \\ &\leq mC_3\varepsilon_0 + O(\varepsilon_0^2) \end{aligned}$$

where $C_3 = C_4 + C_1$. □

We are now ready to prove Lemma 1,

Proof for Lemma 1 Completely analogous to Lemma 5 in Keich (2005) the bound of Lemma 6 above yields:

$$|fl(p_s^\theta) - p_s^\theta| \leq (mC_3 + C_5 \log Q)\varepsilon_0 + O(\varepsilon_0^2).$$

The proof now follows from the identity,

$$p_s = p_s^\theta e^{-\theta s} M(\theta)$$

□

References

Bickel DR (2004) Degrees of differential gene expression: detecting biologically significant expression differences and estimating their magnitudes. Bioinformatics 20(5):682–688

Buckle N, Kraft C, Eeden Cvan (1969) An approximation to the Wilcoxon–Mann–Whitney distribution. J Am Stat Assoc 64:591–599

Dembo A, Zeitouni O (1998) Large deviation techniques and applications. Springer, New York

Di Bucchianico A (1999) Combinatorics, computer-algebra and the Wilcoxon–Mann–Whitney test. J Stat Plann Infer 79:349–364

Dinneen LC, Blakesley BC (1973) Algorithm AS 62: a generator for the sampling distribution of the Mann–whitney U statistic. Appl Stat 22(2):269–273

Fix E, Hodges JL (1955) Significance probabilities of the Wilcoxon test. Ann Math Stat 26(2):301–312

Froda S, Eeden Cvan (2000) A uniform saddlepoint expansion for the null-distribution of the Wilcoxon–Mann–Whitney statistic. Can J Stat 1:137–149

Harding EF (1984) An efficient, minimal-storage procedure for calculating the Mann–Whitney U, generalized U and similar distributions. Appl Stat 33(1):1–6

Hodges JL, Ramsey P, Wechsler S (1990) Improved significance probabilities of the Wilcoxon test. J Educ Stat 15(3):249–265

Jin R, Robinson J (1999) Saddlepoint approximation near the endpoints of the support. Stat Prob Lett 45(4):295–303

Jin R, Robinson J (2003) Saddlepoint approximations of the two-sample Wilcoxon statistic. Institute of Mathematical Statistics, pp 149–158

Karanam S, Moreno CS (2004) CONFAC: automated application of comparative genomic promoter analysis to DNA microarray datasets. Nucleic Acids Res 32(Web Server issue):W475–W484

Keich U (2005) Efficiently computing the p-value of the entropy score. J Comput Biol 12(4):416–430

Keich U, Nagarajan N (2006) A fast and numerically robust method for exact multinomial goodness-of-fit test. J Comput Graph Stat 15(4):779–802

Kifer D, Ben-David S, Gehrke J (2004) Detecting change in data streams. In: Proceedings of the 30th VLDB conference, pp 180–191

Mann H, Whitney D (1947) On a test whether one of two random variables is stochastically larger than the other. Ann Math Stat 18:50–60

Mehta CR, Patel NR, Tsiatis AA (1984) Exact significance testing to establish treatment equivalence with ordered categorical data. Biometrics 40(3):819–825

Nagarajan N, Jones N, Keich U (2005) Computing the p-value of the information content from an alignment of multiple sequences. In: Proceedings of the 13th ISMB conference, pp 311–318

Pagano M, Tritchler D (1983) On obtaining permutation distribution in polynomial time. Am Stat Assoc 83:435–440

Press W, Teukolsky S, Vetterling W, Flannery B (1992) Numerical recipes in C. The art of scientific computing, 2nd edn. Cambridge University Press, New York

Streitberg B, Rohmel J (1984) Exact nonparametrics in APL. In: Proceedings of the APL conference, ACM, New York, pp 313–325

Tasche M, Zeuner H (2002) Improved roundoff error analysis for precomputed twiddle factors. J Comput Anal Appl 4(1):1–18

Troyanskaya OG, Garber ME, Brown PO, Botstein D, Altman RB (2002) Nonparametric methods for identifying differentially expressed genes in microarray data. Bioinformatics 18(11):1454–1461

van Dantzig D (1947–1950) Kader cursus Mathematische Statistiek, Mathematisch Centrum, pp 301–304

Van de Wiel MA, Smeets SJ, Brakenhoff RH, Yistra B (2005) CGHMultiArray: exact p-values for multiarray comparative genomic hybridization data. Bioinformatics 21:3193–3194