[3] B. G. Agee, "The least squares CMA: a new technique for rapid correction of constant modulus signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1986, pp. 953–956.
[4] T. E. Biedka, W. H. Tranter, and J. H. Reed, "Convergence analysis of the least squares constant modulus algorithm in interference cancellation applications," *IEEE Trans. Signal Processing*, vol. 48, pp. 491–501, Mar. 2000.
[5] H. Furukawa, Y. Kamio, and H. Sasaoka, "Cochannel interference reduction and path-diversity reception technique using CMA adaptive array antenna in digital land mobile communications," *IEEE Trans. Veh. Technol.*, vol. 50, pp. 605–616, Mar. 2001.
[6] A. Mathur, A. V. Keerthi, and J. J. Shynk, "A variable step-size CM array algorithm for fast fading channels," *IEEE Trans. Signal Processing*, vol. 45, pp. 1083–1087, Apr. 1997.
[7] J. J. Shynk, A. V. Keethi, and A. Mathur, "Steady-state analysis of the multistage constant modulus array," *IEEE Trans. Signal Processing*, vol. 44, pp. 948–962, Apr. 1996.
[8] B. Yang, "Projection approximation subspace tracking," *IEEE Trans. Signal Processing*, vol. 43, pp. 95–107, Jan. 1995.
[9] Y. X. Chen, Z. Y. He, T. S. Ng, and P. C. K. Kwok, "RLS adaptive blind beamforming algorithm for cyclostationary signals," *Electron. Lett.*, vol. 35, no. 14, pp. 1136–1138, July 1999.
[10] S. Haykin, *Adaptive Filter Theory*, 2nd ed. NJ: Printice-Hall, 1991.
[11] M. Rupi, P. Tsakalides, C. L. Nikias, and E. Del Re, "Robust constant modulus arrays based on fractional lower-order statistics," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1999, pp. 2945–2948.
[12] L. Ljung and T. Soderstrom, *Theory and Practice of Recursive Identification*. Cambridge, MA: MIT Press, 1983.

# Comments on "Numerical Evaluation of the Lambert $W$ Function and Application to Generation of Generalized Gaussian Noise with Exponent 1/2"

D. A. Barry, L. Li, and D.-S. Jeng

***Abstract*—The Lambert $W$ function appears in a wide variety of circumstances, including the recent application to signal processing referred to in the paper under discussion. Besides applications, a sizable body of mathematical analysis has been reported. The original paper presented a numerical algorithm for computation of $W_{-1}$. An existing, similar algorithm is presented. Iterative improvement of the $W_{-1}$ estimates is also discussed, and issues concerning computational efficiency and possible sources of rounding error in fixed precision computational environments are identified. Existing, public-domain software takes into account all the identified numerical issues and produces estimates of $W$ to near the precision available on the host machine.**

***Index Terms*—Algorithms, approximation methods, error estimation, finite word length effects, iterative methods, round-off errors.**

## I. Introduction

The Lambert $W$ function arises in a wide variety of mathematical, physical, chemical, and engineering contexts [1]. A recent summary of these is available [2]. In [3], a new use of the lower real branch of the Lambert $W$ function $W_{-1}$ is described; it arises in the inverse distribution function of generalized Gaussian noise with power 1/2. The main contributions of [3] were, first, to identify the relationship between $W_{-1}$ and a particular case of the generalized Gaussian noise distribution and, second, to provide a numerical algorithm for computation of $W_{-1}$. Here, our focus is on the second contribution.

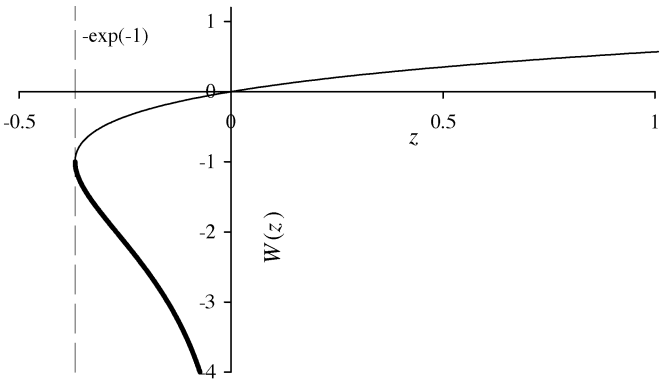


Fig. 1. Branches of the Lambert $W$ function. The lower branch $W_{-1}$ is given by the thick line; the upper branch $W_0$ is given by the thin line.

Given the widespread applications of the Lambert $W$ function referred to above, further utilization of it in signal processing is a reasonable prospect. It is thus timely to mention the significant body of research pertaining to the $W$ function in the mathematical literature, including numerical algorithms for its calculation. Specifically, this comment is intended first to extend and clarify some material presented in [3] regarding the $W$ function and its approximations and, second, to alert readers that well-tested, arbitrary-precision numerical software is available for computation of *both* real branches of the Lambert $W$ function, i.e., $W_{-1}(z)$, $-\exp(-1) \le z < 0$ with $W_{-1}(z) \le -1$, and the principal branch $W_0(z)$, $-\exp(-1) \le z$ with $W_0(z) \ge -1$ (see Fig. 1 for a plot of these branches). The available software has been thoroughly tested and, as such, is not prone to numerical problems such as rounding error that could occasionally result from the naïve application of the algorithm presented in [3].

## II. Detailed Remarks

### A. Available Software

We wish to make three remarks regarding existing software and algorithms used therein.

1) Software to compute real values of the Lambert $W$ function is available in the public domain mathematical library *Netlib* as Algorithm 743 of the TOMS database (www.netlib.org/toms/743). The software, which is written in *FORTRAN* (both single and double precision versions), computes both real branches of the Lambert $W$ function to whatever precision is available on the computing platform used. This algorithm and details of its application were described in [4] and [5]. A *FORTRAN*-to-*C* converter such as *f2c* (www.netlib.org/f2c) could be employed to help derive *C* or *C++* versions of the TOMS software.
2) The approximation in [4], which is simpler than that presented in [3], produces "one-shot" estimates of $W_{-1}$ with

Manuscript received October 1, 2002; revised June 18, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Fredrik Gustafsson.

D. A. Barry is with the University of Edinburgh, Edinburgh EH9 3JL, U.K. (e-mail: d.a.barry@ed.ac.uk).
L. Li is with the School of Engineering, University of Queensland, St. Lucia, 4072, Australia (e-mail: l.li@uq.edu.au).
D.-S. Jeng is with the Department of Civil Engineering, the University of Sydney, Sydney, NSW 2006, Australia (e-mail: d.jeng@civil.usyd.edu.au).

Digital Object Identifier 10.1109/TSP.2004.826154

a maximum relative error of $O(10^{-4})$. This algorithm is repeated here for easy comparison with that in [3, Fig. 2]:

$$W_{-1}(z) = \begin{cases} -1-t-2x^{-1} \\ \quad \times\left\{1-\left[1+x\left(\frac{t}{2}\right)^{\frac{1}{2}}\right]^{-1}\right\}, & -\exp(-1)\le z\le -\exp(-9) \\ \ln\left[\frac{z}{\ln(-zg^{-1})}\right], & -\exp(-9)<z<0 \end{cases} \quad (1)$$

where $t = -1-\ln(-z)$, $x = 3^{-1}-t(270+127.047\,138\,134\,921\,9t^{1/2})^{-1}$, $\eta = 2[1+z\exp(1)]$, and $g = 1+(1+0.504\,392\,132\,306\,845\,7t)(\eta^{1/2}+3^{-1}\eta)$. It is unclear that the new algorithm offers any advantages over that given in [4].

3) As explained in [4], the algorithm in (1) was developed for use with the fourth-order iterative improvement scheme of [6] such that a single iteration gives results that have a relative error of $O(10^{-16})$, i.e., the results are accurate to the number of digits typically available in fixed-precision numerical environments. On the other hand, in [3], the third-order Halley iterative improvement scheme used in the *Maple* software is recommended. However, using an initial guess with a relative error of $O(10^{-4})$, two iterations of the Halley scheme would be needed to achieve an estimate of $W_{-1}$ with a relative error of $O(10^{-16})$, in which case, it would be less efficient than the fourth-order scheme of [6].

### B. Rounding Error

There are two specific numerical issues involving rounding error that concern the computation of the Lambert $W$ function near $z = -\exp(-1)$:

4) Like the fourth-order iterative improvement scheme used in [5], the Halley scheme (see [3, (9)]) suffers from rounding error near the branch point $z = -\exp(-1)$ when operated in a fixed-precision computing environment. The problem is that updated estimates are not produced because of rounding. Likewise, the error control criterion (see [3, (11)]) will fail. To see this, note that both [3, (9) and (11)] contain the term $w_{approx}\exp(w_{approx})-z$, where $w_{approx}$ is an estimate of $W_{-1}(z)$. As an example, consider $z = -\exp(-1)$ for which $W_{-1}(z) = -1$. Suppose that ten digits are available for computation and that the current estimate of $W_{-1}$ is $w_{approx} = -1-10^{-5}$. The relative error is $|1-1+10^{-5}| = 10^{-5}$. However, the expression $w_{approx}\exp(w_{approx})-z$ evaluates to $-1.000\,010\,000\times -0.367\,875\,762\,4-(-0.367\,879\,441\,2) = 0.000\,000\,000$. Thus, although ten digits are available, the scheme in [3] will give only five correct digits in this case.

The fourth-order iterative improvement scheme used in [4] was examined closely in that paper. To avoid iteration and the associated rounding error, a more intricate continued-fraction approximation was proposed for noniterative estimation of $W_{-1}$ in the region $-\exp(-1) < z < -\exp(-1)[1-17\times 22(1-b)/7]$, where $b$ is the number of bits in the floating-point mantissa of the host machine. In-built calculation of $b$ is included in the *Netlib* software.

5) The second problem of rounding error arises due to round-off error for arguments $z$ that are near $-\exp(-1)$. If $z+\exp(-1) = \Delta z$ yields an increment (i.e., $\Delta z$) less than the machine precision, then the increment will be ignored in the calculations, and the argument used will simply be $z = -\exp(-1)$. The value of the Lambert $W$ function computed will be $W_{-1} = -1$. However, as is clear from [3, (16)], the value of $W_{-1}[-\exp(-1)+\Delta z] \approx -1-[2\Delta z\times\exp(1)]^{1/2}$; therefore, the correct value of $W_{-1}(z)$ will be different from $-1$ to within the precision of the machine. For example, again suppose the ten digits are available for floating point computations. Then, to ten digits, $-\exp(-1)+10^{-11} = -\exp(-1)$ for which $W_{-1}$ would be $-1.000\,000\,000$. However, the correct value of $W_{-1}[-\exp(-1)+10^{-11}] = -1.000\,007\,373$, and therefore, only five correct digits result in the estimate. In [5], it is pointed out that when values of $W_{-1}(z)$ or $W_0(z)$ are needed in this region, it is necessary to compute the offset $\Delta z$ from $-\exp(-1)$ and use this offset in the (suitably modified) calculations.

### C. Other Approximations

Other simple analytical approximations to $W_{-1}(z)$ are available. For example, two approximations were presented in [7] that are valid for the entire range of $z$, i.e., $-\exp(-1)\le z<0$. Both are single equations of the form given in the first section of (1). One has a maximum relative error of $3.5\times 10^{-3}$:

$$W_{-1}(z) = -1-t-2x_0^{-1}\times\left\{1-\left[1+x_0\left(\frac{t}{2}\right)^{\frac{1}{2}}\right]^{-1}\right\}, \quad x_0 = 0.3205 \quad (2)$$

whereas the other's maximum relative error is $2.5\times 10^{-4}$:

$$W_{-1}(z) = -1-t-2x_1^{-1}\left\{1-\left[1+\frac{x_1\left(\frac{t}{2}\right)^{\frac{1}{2}}}{1+x_2t\exp\left(x_3t^{\frac{1}{2}}\right)}\right]^{-1}\right\} \quad (3)$$

where $x_1 = 0.3361$, $x_2 = -0.0042$, and $x_3 = -0.0201$. The expressions in (2) and (3) apply over the entire range of $z$ and, depending on the circumstances, might be useful for rapid computation of $W_{-1}(z)$.

For $W_0(z)$, analytical approximations are also available in [1], [4], [7], and [8]. The maximum relative errors of these approximations range vary with their complexity. For example, the maximum relative error of the approximation in [4] is $O(10^{-4})$.

## III. Conclusions

A key contribution of [3], viz., an algorithm to compute $W_{-1}(z)$ to $O(10^{-4})$ maximum relative error, gives results that correspond (in the sense that the same relative error is achieved) to an existing, somewhat simpler algorithm presented in [4]. Improved estimates can be computed using the third-order Halley scheme, as recommended in [3]. However, several possible concerns arise when computations are performed with a fixed number of digits. First, because it requires two iterations to achieve the typical $O(10^{-16})$ results desired in fixed-precision computational environments, the Halley scheme will be more computationally expensive than a single pass of the fourth-order scheme used in [4]. Second, there are two possible sources of rounding error that affect calculations near $z = -\exp(-1)$. As quantified in [4], how "near" is near depends on the precision used in the computations. All these issues are accounted for in the public domain TOMS algorithm available at the Netlib repository.

## Acknowledgment

The authors thank the Associate Editor, Prof. Fredrik Gustafsson, and the three anonymous reviewers for their helpful and constructive comments and suggestions.

## REFERENCES

[1] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth, "On the Lambert $W$ function," *Adv. Comput. Math.*, vol. 5, pp. 329–359, 1996.

[2] D. A. Barry, J.-Y. Parlange, L. Li, H. Prommer, C. J. Cunningham, and F. Stagnitti, "Analytical approximations for real values of the Lambert $W$-function," *Math. Comput. Simul.*, vol. 53, pp. 95–103, 2000.

[3] F. Chapeau-Blondeau and A. Monir, "Numerical evaluation of the Lambert $W$ function and application to generation of generalized Gaussian noise with exponent 1/2," *IEEE Trans. Signal Processing*, vol. 50, pp. 2160–2165, Sept. 2002.

[4] D. A. Barry, P. J. Culligan-Hensley, and S. J. Barry, "Real values of the $W$-function," *ACM Trans. Math. Softw.*, vol. 21, pp. 161–171, 1995.

[5] D. A. Barry, S. J. Barry, and P. J. Culligan-Hensley, "Algorithm 743: WAPR: a FORTRAN routine for calculating real values of the $W$-function," *ACM Trans. Math. Softw.*, vol. 21, pp. 172–181, 1995.

[6] F. N. Fritsch, R. E. Shafer, and W. P. Crowley, "Algorithm 443, solution of the transcendental equation $we^w = x$," *Commun. ACM*, vol. 16, pp. 123–124, 1973.

[7] D. A. Barry, J.-Y. Parlange, G. C. Sander, and M. Sivaplan, "A class of exact solutions for Richards' equation," *J. Hydrol.*, vol. 142, pp. 29–46, 1993.

[8] D. A. Barry, J.-Y. Parlange, L. Li, H. Prommer, C. J. Cunningham, and F. Stagnitti, "Erratum to 'Analytical approximations for real values of the Lambert $W$-function' [mathematics and computers in simulation 53(2000) 95–103]," *Math. Comput. Simul.*, vol. 59, p. 543, 2002.


# The Relationship of Transform Coefficients for Differing Transforms and/or Differing Subblock Sizes

Brynmor J. Davis and S. Hamid Nawab

***Abstract*—Jiang and Feng have developed a relationship between the discrete cosine transform (DCT) coefficients of a block and those of its subblocks. Their derivation of this result can be significantly simplified. The new derivation also generalizes to all linear, invertible transforms and any separable subblock geometry.**

***Index Terms*—Separable transform, subblock transform, transform coefficient relationship, transform domain processing.**

## I. INTRODUCTION

A relationship between the discrete cosine transform (DCT) [1] coefficients of an image and the DCT coefficients of its subblocks has been derived by Jiang and Feng [2]. This direct relationship is a computationally efficient alternative to first applying an inverse DCT (to obtain the image in the pixel domain) and then taking the forward DCT over the subblocks. Direct coefficient manipulations such as this facilitate image processing in the transform domain, which is currently an active area of research [3]–[6].

The derivation contained in [2] can be significantly simplified using a matrix representation of the transforms. In addition, this approach allows the relationship to be generalized from the DCT to any linear, invertible transform (and even certain mixes of transforms across the subblocks). The constraints on the geometries of the subblocks are also relaxed—while [2] only deals with $A:1$ ratios between the subblocks (e.g., a $12 \times 20$ image being divided into four $6 \times 10$ subblocks), the new derivation can be applied to $A:B$ ratios (e.g. four $6 \times 10$ subblocks being related to 15 $4 \times 4$ subblocks) or even certain nonuniform ratios (e.g. four $6 \times 10$ subblocks being related to a $12 \times 8$ subblock and a $12 \times 12$ subblock).

A related approach is nonuniform transform domain filtering (NTDF) [7], which is an extension of transform domain filtering (TDF) [8]. Theoretically, NTDF can be used to relate coefficients where there is a constant $A:B$ ratio between the subblock sizes. However, [7] does not include a clear relation between the coefficients, nor does it describe explicitly which subblock and transform geometries can be handled. Additionally, NTDF is designed specifically for a pipelining architecture.

In comparison, the results presented here give a fundamental theoretical relation, along with a well-defined set of sufficient conditions for its validity. These conditions are more general than those described in either [2] or [7]. The result is presented first in one dimension and then in two dimensions through the use of separable transforms and separable subblock geometries. An example is also provided.

## II. ONE-DIMENSIONAL RESULT

The majority of useful discrete transforms are linear and invertible (e.g., the DCT, the discrete Fourier transform (DFT) [9], and the discrete wavelet transform (DWT) [10]). This means their operation on a $N \times 1$ vector $x$ can be represented by a matrix multiplication (see [9] for examples). The transform matrix will be $N \times N$ and invertible. The result of this matrix multiplication gives a vector of transform coefficients.

$$X = Tx \tag{1}$$

$$x = T^{-1}X. \tag{2}$$

If the vector $x$ is split into $P$ contiguous subvectors with lengths $N_1, N_2, \cdots, N_P$ ($\sum_{i=1}^{P} N_i = N$) and a (possibly different) transform applied to each subvector, then the resulting set of transform coefficients can still be found through multiplication with a $N \times N$ matrix $\hat{T}$.

$$\hat{X} = \hat{T}x \tag{3}$$

$$x = \hat{T}^{-1}\hat{X}. \tag{4}$$

The $\hat{}$ indicates that transforms have been taken over a set of subvectors. In addition, the matrices $\hat{T}$ and $\hat{T}^{-1}$ can be constructed from the transform matrices for each subvector. They will exhibit a block diagonal structure, which is shown as

$$\hat{T} = \begin{bmatrix} T_1 & 0 & & \cdots & 0 \\ 0 & T_2 & 0 & \cdots & 0 \\ 0 & 0 & T_3 & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & T_P \end{bmatrix} \tag{5}$$

$$\hat{T}^{-1} = \begin{bmatrix} T_1^{-1} & 0 & & \cdots & 0 \\ 0 & T_2^{-1} & 0 & \cdots & 0 \\ 0 & 0 & T_3^{-1} & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & T_P^{-1} \end{bmatrix}. \tag{6}$$

Manuscript received February 13, 2003; revised May 22, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sheila S. Hemami.

The authors are with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215 USA (e-mail: bryn@bu.edu; hamid@bu.edu).

Digital Object Identifier 10.1109/TSP.2004.826165