An Algorithm for the Matrix Lambert W Function

Fasi, Massimiliano and Higham,
Nicholas J. and Iannazzo, Bruno

2014

MIMS EPrint: **2014.58**

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

AN ALGORITHM FOR THE MATRIX LAMBERT W FUNCTION*

MASSIMILIANO FASI†, NICHOLAS J. HIGHAM‡, AND BRUNO IANNAZZO§

Abstract. An algorithm is proposed for computing primary matrix Lambert W functions of a square matrix A, which are solutions of the matrix equation $We^W = A$. The algorithm employs the Schur decomposition and blocks the triangular form in such a way that Newton's method can be used on each diagonal block, with a starting matrix depending on the block. A natural simplification of Newton's method for the Lambert W function is shown to be numerically unstable. By reorganizing the iteration a new Newton variant is constructed that is proved to be numerically stable. Numerical experiments demonstrate that the algorithm is able to compute the branches of the matrix Lambert W function in a numerically reliable way.

Key words. Lambert W function, primary matrix function, Newton method, matrix iteration, numerical stability, Schur–Parlett method

AMS subject classifications. 65F60, 15A15

1. Introduction.

The Lambert W function of the complex number a is defined implicitly through the scalar equation

$$we^w = a, \tag{1.1}$$

which has a countably infinite set of solutions for $a \neq 0$ and just one solution for $a = 0$.

Any solution $w \neq -1$ of (1.1) can be extended to an analytic function w, such that $we^w = a$, defined in all of the complex plane except a suitable curve, the branch cut. Any such function is said to be an analytic branch of the Lambert W function and can be further extended on the branch cut (excluding the singular points) in order to be continuous and differentiable (for $w \neq 1$) on one side of the branch cut.

For background on the Lambert W function and details of applications we refer the reader to [10], [13]. We follow the numbering of the branches, the notation, and the branch cuts of [10] (see Figure 1.1). The branches are denoted by $W_k(a)$ for $k \in \mathbb{Z}$, where $W_0(z)$ is the principal branch whose branch cut is $(-\infty, -1/e]$, while for $k \neq 0$, the branch cut is $(-\infty, 0]$. The extension to the branch cuts is made using the "counter-clockwise continuity" rule [26], that is, considering the one-sided limit from above the branch cut.

For each $a \in \mathbb{C} \setminus \{0\}$ the set $\{W_k(a) : k \in \mathbb{Z}\}$ is the set of solutions of (1.1), while for $a = 0$ the unique solution is $W_0(0) = 0$. The latter implies that $W_k(0)$ is undefined for $k \neq 0$, namely, 0 is a singular point.

In the matrix case, for $A \in \mathbb{C}^{n\times n}$ any solution of the equation $We^W = A$ can be called a matrix Lambert W function [11]. The solutions of this matrix equation break into two types.

---

*Version of March 28, 2015.

†Dipartimento di Informatica—Scienza e Ingegneria, Università di Bologna, Via Mura Anteo Zamboni 7, 40126 Bologna, Italy (massimiliano.fasi@studio.unibo.it). The work of this author was supported by a grant of Collegio Superiore di Bologna.

‡School of Mathematics, University of Manchester, Manchester, M13 9PL, UK (nick.higham@manchester.ac.uk, http://www.maths.manchester.ac.uk/~higham). The work of this author was supported by European Research Council Advanced Grant MATFUN (267526) and Engineering and Physical Sciences Research Council grant EP/I01912X/1.

§ Dipartimento di Matematica e Informatica, Università di Perugia, Via Vanvitelli 1, 06123 Perugia, Italy (bruno.iannazzo@dmi.unipg.it). The work of this author was supported by the Istituto Nazionale di Alta Matematica, INdAM–GNCS Project 2014.

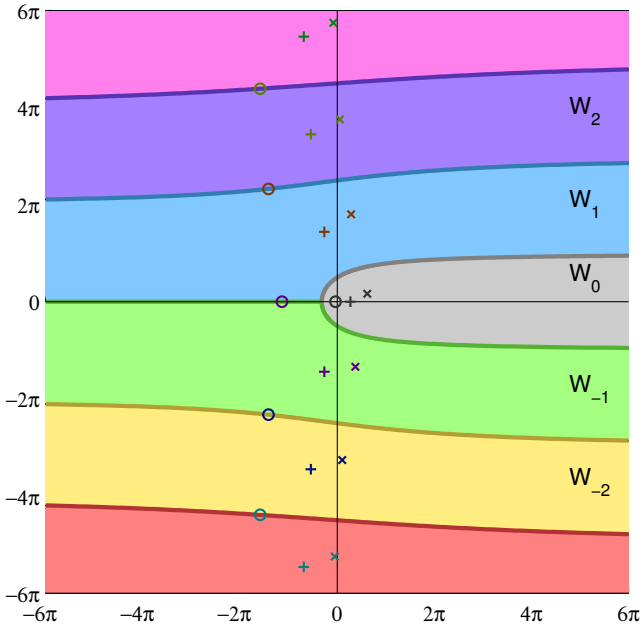

FIG. 1.1. *The ranges of the branches of the Lambert W function and the values of* $W_k(1)$ (+), $W_k(10+10i)$ (×) *and* $W_k(-0.1)$ (o). *The colors of the curves that separate two adjacent regions denote the branches that the curves belong to. The corresponding plot for the logarithm consists of horizontal strips of height* 2π *with boundaries at odd multiples of* π.

- Primary matrix Lambert W functions, defined using the theory of primary matrix functions [20, sec. 1.2]. Each such function can be written as a polynomial in A.
- Nonprimary matrix Lambert W functions, none of which can be written as a polynomial in A.

It can be shown that the eigenvalues of any matrix Lambert W function of A are Lambert W functions of the eigenvalues of A. A matrix Lambert W function is nonprimary if and only if its spectrum contains two different branches of the same eigenvalue of A. This may happen only when A has an eigenvalue appearing in two different Jordan blocks. Our focus here is on primary matrix Lambert W functions.

Primary matrix Lambert W functions can be described by specifying a branch for each distinct eigenvalue of A. For an eigenvalue λ, the branch W_k can be freely chosen with the following exceptions:

- if $\lambda = 0$, then we must choose $k = 0$, since elsewhere the scalar function is not defined;
- if $\lambda = -1/e$ appears in a nontrivial Jordan block then we must choose $k \notin \{0, -1\}$; for $k \in \{0, -1\}$ the scalar function is defined at $-1/e$ (in fact, $W_0(-1/e) = W_{-1}(-1/e) = -1$) but it cannot be extended to a differentiable function.

Let $\lambda_1, \dots, \lambda_t$ be the distinct eigenvalues of A in a specific order. With the constraints given above, we define $W_{k_1,k_2,\dots,k_t}(A)$ as the primary Lambert W function whose eigenvalues are $W_{k_1}(\lambda_1), \dots, W_{k_t}(\lambda_t)$.

If we want the same branch W_k for every eigenvalue it is convenient to use the notation $W_k(A)$ instead of $W_{k,\dots,k}(A)$. In this case we say that $W_k(A)$ is an *unmixed*

branch of the matrix Lambert W function.

The matrix Lambert W function arises in the numerical solution and stability analysis of delay differential (systems of) equations [5], [9], [25], [32], [33], where the principal Lambert W function of a matrix, $W_0(A)$, is used to deduce properties of the stability of the system. Cepeda-Gomez and Michiels [9] show with examples that the principal branch is not sufficient to determine the stability of all systems, but rather $W_{-1}(A)$ is needed as well. Thus an algorithm that can compute any branch is required. The matrix Lambert W function has been recently considered in a problem of quantum computing [31], where the matrix argument A is normal ($A^*A = AA^*$).

The purpose of this work is to introduce an algorithm for computing the unmixed branches of the Lambert W function (provided they are well-defined) of a general matrix $A \in \mathbb{C}^{n\times n}$. The algorithm is based on the Newton iteration applied to the matrix equation $We^W - A = 0$.

We follow the usual idea of considering the scalar function $f(w) = we^w - a$, applying Newton's method to obtain the iteration

$$y_{k+1} = \frac{y_k^2 + ae^{-y_k}}{y_k + 1}, \tag{1.2}$$

and then translating from scalars to matrices. The resulting matrix iteration coincides with Newton's method applied to $We^W - A = 0$ when the initial value is a polynomial of A and both are well defined. However, this iteration suffers from two problems. First, it is numerically unstable in finite precision arithmetic. Second, it is not easy to find a starting matrix such that the Newton iteration converges to the desired matrix Lambert W function. We develop a heuristic strategy for the initial value based on two different series expansions of the scalar Lambert W function, at ∞ (and 0) and around $-1/e$, respectively. For any scalar a few terms of one of these two expansions always yields a good initial value. In the matrix case we would like to use one of the two expansions applied to A, but if A has two eigenvalues for which two different series expansions need to be used as starting points neither of these expansions can be used.

We are able to resolve both these problems. We identify a new iteration that is equivalent to Newton's method but numerically stable. Moreover, we consider an ordered Schur form T of A such that the leading eigenvalues are the ones for which the approximation at ∞ (and 0) yields a suitable initial value for Newton's method while the trailing eigenvalues are the ones for which the approximation at $-1/e$ gives a good initial value. Blocking T as a 2×2 block matrix and using the Schur–Parlett algorithm we obtain the correct evaluation of the Lambert W function on the whole matrix A. The resulting algorithm for computing the matrix Lambert W function requires $O(n^3)$ arithmetic operations and is much more numerically reliable than an algorithm based on diagonalization used in [5].

The paper is organized as follows. In section 2 we give necessary background on Fréchet derivatives and the branch cuts of the Lambert W function. Section 3 treats the choice of the initial value for Newton's method in the scalar case. In section 4 we derive a numerically stable form of a simplified version of Newton's method for the matrix Lambert W function and combine it with a suitably blocked Schur decomposition to construct a complete algorithm for the Lambert W function of a matrix. Section 5 is devoted to numerical experiments.

2. Preliminaries.

2.1. The Fréchet derivative. We first give some notation and properties of the Fréchet derivative of a matrix function that will be needed in the following sections. The Fréchet derivative of f at $A \in \mathbb{C}^{n\times n}$, when it exists, is the unique linear mapping $Df(A)[E]$ satisfying

$$f(A+E) = f(A) + Df(A)[E] + o(\|E\|)$$

for all $E \in \mathbb{C}^{n\times n}$. Since $Df(A)[E]$ is linear in E, the vec operator, which converts a matrix to a vector by stacking the columns on top of each other, yields $\operatorname{vec}(Df(A)[E]) = K(A)\operatorname{vec}(E)$, for an $n^2 \times n^2$ matrix $K(A)$ called the Kronecker matrix. The vec operator interacts nicely with the Kronecker product; in particular, $\operatorname{vec}(AXB) = (B^T \otimes A)\operatorname{vec}(X)$.

The Fréchet derivative of a matrix function at a matrix A in the direction H is usually much more complicated than its scalar counterpart, but when H commutes with A there is a simple expression, as shown in the following result [20, Prob. 3.8].

LEMMA 2.1. *Let $f : \Omega \to \mathbb{C}$ be analytic, let $\mathcal{U}_n$ be the subset of $\mathbb{C}^{n\times n}$ comprising matrices whose spectrum belongs to Ω, and let $f_n, f_n' : \mathcal{U}_n \to \mathbb{C}^{n\times n}$ be the matrix functions induced by f and f' respectively. Then for any H commuting with $A \in \mathcal{U}_n$, $Df_n(A)[H] = f_n'(A)H$.*

For instance, if A commutes with H, then $D\exp(A)[H] = \exp(A)H$, which mimics the scalar case. A formula for the derivative of the exponential function in a general direction is not so simple. We will use the following expression of the Kronecker matrix associated with the derivative of the exponential [20, Thm. 10.13]:

$$K(A) = (I \otimes \exp(A))f_1(A^T \otimes I - I \otimes A), \tag{2.1}$$

where $f_1(z) = (e^z - 1)/z$ for $z \neq 0$ and $f_1(0) = 1$.

2.2. Branch cuts of the Lambert W function. As noted in section 1, the branch cut for the principal branch W_0 is $(-\infty, -1/e]$, while for W_k, with $k \neq 0$, it is $(-\infty, 0]$. This choice of branch cuts is convenient because it allows simple asymptotic expansions for $W_k(z)$ as $z \to \infty$ based on the branches of the complex logarithm [10]. On the other hand, it makes it complicated to understand the behaviour of the branches $W_1(z)$ and $W_{-1}(z)$ in a neighborhood of the point $z = -1/e$.

In fact, the image of a small circle around $-1/e$ under $W_{-1}(z)$ comprises two disjoint curves: the half circle below the real axis is mapped into a curve adjacent to the branch -2 region, while the half circle above the real axis is mapped into a curve adjacent to the branch 0 and 1 regions. Thus, on the one hand, for ε sufficiently small, the function defined as $W_{-1}(z)$ in the half-circle $\{|z + 1/e| \leqslant \varepsilon : \operatorname{Im} z > 0\}$ cannot be extended analytically in a neighborhood of $-1/e$. On the other hand, the function defined as $W_{-1}(z)$ in the half circle $\{|z + 1/e| \leqslant \varepsilon : \operatorname{Im} z < 0\}$ can be extended analytically in a neighborhood of $-1/e$. This explains the lack of symmetry in treating the branch cut.

The situation for the branch W_1 is similar (see Figure 2.1): on one hand, for ε sufficiently small, the function defined as $W_1(z)$ in the half-circle $\{|z + 1/e| \leqslant \varepsilon : \operatorname{Im} z > 0\}$ can be extended analytically in a neighborhood of $-1/e$. On the other hand, the function defined as $W_1(z)$ in the half circle $\{|z + 1/e| \leqslant \varepsilon : \operatorname{Im} z < 0\}$ cannot be extended analytically in a neighborhood of $-1/e$.

While $W_{-1}(-1/e) = -1$ by the counter-clockwise continuity rule, we have $W_1(-1/e) \neq -1$, and thus, strictly speaking, $-1/e$ is not a branch point for the branch 1 (while it is a branch point for the branch -1); nevertheless, in a neighborhood of $-1/e$ the function W_1 behaves as if it had a branch point there.

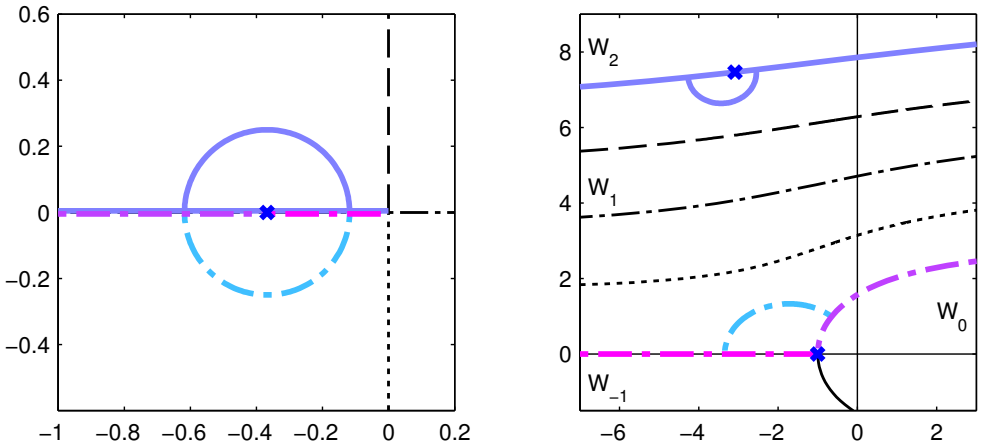

FIG. 2.1. *The image through $W_1(z)$ (right) of the complex plane with a highlighted circle around the branch point $-1/e$ (left). The image of the branch cut $(-\infty, 0)$ when we approach it from above is the regular curve separating the range of $W_1(z)$ from the range of $W_2(z)$; while the image of the same half line when we approach it from below splits in two curves meeting at the point -1: the left one separates the range of $W_1(z)$ from that of $W_{-1}(z)$, the right one separates the same range from that of $W_0(z)$. This illustrates why we treat in a different way the branch $W_1(z)$ when we approach the negative real axis clockwise or counter-clockwise; we can loosely say that there is one branch cut from above and two from below. The colors have been used to show what is the image of any (part of a) branch cut.*

Finally, the point 0 turns out to be singular for each branch different from W_0 and, in particular, $\lim_{z\to 0} |W_k(z)| = \infty$ for $|k| > 0$.

3. Computing the scalar Lambert W function. The Lambert W function has a qualitative behaviour comparable to the logarithm, but it cannot be computed with the same techniques because of the lack of suitable identities. The customary methods for computing the scalar Lambert W function are based on a suitable root-finding method applied to the equation $we^w = a$, such as the Newton or the Halley methods, which guarantee local superlinear convergence for $w \neq -1$ and linear convergence for $w = -1$. These methods are based on iterations of the form $z_{i+1} = \varphi(z_i)$ and require determination of z_0 such that z_i converges to the value $W_k(a)$ for the k of interest. For this purpose some heuristic techniques have been proposed and implemented. They are based on two asymptotic expansions of the function $W_k(z)$, which we briefly recall.

It is known that [10, eq. (4.18)] for each k one has, as $z \to \infty$,

$$W_k(z) = \ell_{1,k} - \ell_{2,k} + \sum_{h=0}^{\infty}\sum_{m=1}^{\infty} c_{hm}\frac{\ell_{2,k}^m}{\ell_{1,k}^{m+h}}, \qquad c_{hm} = \frac{(-1)^{m+h-1}}{m!}S(h+m, h+1), \quad (3.1)$$

where $\ell_{1,k} = \ln z + 2\pi i k =: \ln_k(z)$, $\ell_{2,k} = \ln \ell_{1,k}$, and $S(h+m, h+1)$ is a (signed) Stirling cycle number of first kind [28, p. 631]. Here, for a nonzero $z \in \mathbb{C}$, $\ln z$ denotes the principal logarithm, which is the solution of the equation $e^x = z$ having imaginary part in the strip $\{z \in \mathbb{C} : -\pi < \mathrm{Im}(z) \leqslant \pi\}$. It has been proved that the same expansion holds for $z \to 0$ and $k \neq 0$ (see [10] and the references therein). The

expansion implies

$$W_k(z) = \ell_{1,k} - \ell_{2,k} + \frac{\ell_{2,k}}{\ell_{1,k}} + \frac{\ell_{2,k}(\ell_{2,k}-2)}{2\ell_{1,k}^2} + \frac{\ell_{2,k}(2\ell_{2,k}^2 - 9\ell_{2,k}+6)}{6\ell_{1,k}^3} + O\left(\left|\frac{\ell_{2,k}}{\ell_{1,k}}\right|^4\right), \tag{3.2}$$

which holds either as $z \to 0$ and $k \neq 0$ or as $z \to \infty$.

Another expansion is known for $z \to -1/e$, which converges to $W_0(z)$ in a neighborhood of $-1/e$, to W_{-1} when $\mathrm{Im}(z) \geqslant 0$ and z approaches $-1/e$, and to W_1 when $\mathrm{Im}(z) < 0$ and z approaches $-1/e$ (compare the discussion in section 2.2 about the choice of the branches). The way to construct the series can be found in [10, p. 350]; the first few terms are

$$W_k(z) = -1 + p - \frac{1}{3}p^2 + \frac{11}{72}p^3 + \cdots, \qquad p = (-1)^{|k|}\sqrt{2(ez+1)}. \tag{3.3}$$

A widely used strategy [10], [30] is to choose as initial guess for the selected root-finding algorithm one of the two expansions (up to a certain term) that is hopefully sufficiently close to $W_k(z)$ to guarantee convergence.

In our implementation we use Newton's method as the root-finding algorithm and we choose as initial value one of the two functions

$$\varphi_k(z) = \ell_{1,k} - \ell_{2,k} + \ell_{2,k}/\ell_{1,k}, \qquad \psi_k(z) = -1 + (-1)^{|k|}\sqrt{2(ez+1)}, \tag{3.4}$$

the former being a truncation of (3.2) and the latter a truncation of (3.3). In particular,

- for $k = 0$, we choose $z_0 = \psi_0(z)$ for $|z - 1/2| < 3/2$, and $z_0 = \varphi_0(z)$ otherwise;
- for $k = 1$, we choose $z_0 = \psi_1(z)$ for $|z + 1/2| < 1/3$ with $\mathrm{Im}\, z < 0$, and $z_0 = \varphi_1(z)$ otherwise;
- for $k = -1$, we choose $z_0 = \psi_{-1}(z)$ for $|z + 1/2| < 1/3$ with $\mathrm{Im}\, z \geqslant 0$, and $z_0 = \varphi_{-1}(z)$ otherwise;
- for $|k| \geqslant 1$, we choose $z_0 = \varphi_k(z)$ for any $z \in \mathbb{C} \setminus \{0\}$.

These choices are based on numerical experiments. We describe just the choice for $k = 0$; the others follow from similar experiments.

We consider a discretization $\widetilde{Q}$ of the subset $Q = \{z = a + ib : |a|, |b| \leqslant 3\}$ of the complex plane and run Newton's method for each $z \in \widetilde{Q}$ with $z_0 = \varphi_k(z)$ (or with $z_0 = \psi_k(z)$). Let N be a fixed positive integer. If the value $W_k(z)$ is approximated within a specific tolerance in $m < N$ steps then we set $s_k(z) = m$, otherwise $s_k(z)$ is set to N. In this way, we have a function $s_k : \widetilde{Q} \to \{0, 1, \ldots, N\}$, and assigning a color to each element of $\{0, \ldots, N\}$ we obtain an illustration of the convergence behaviour of the Newton method.

In Figure 3.1 we show the result of the experiment for $k = 0$, together with the circle $|z - 1/2| = 3/2$. As one can see, choosing the approximation $z_0 = \varphi_k(z)$ outside the circle gives convergence in no more than 7 steps (blue and green area of left plot), while choosing the approximation $z_0 = \psi_k(z)$ for z inside the circle gives convergence in no more than 8 steps (light green area of the right plot).

Notice that the chosen areas for the initial values are somewhat arbitrary. For instance, for $k = 0$, we have put the center at $z = 1/2$ and the radius $R = 3/2$ for simplicity, but any $1.35 \leqslant R \leqslant 1.60$ would give the same convergence properties. A similar argument can be used for $|k| = 1$, yielding a range of useful radii which includes the interval $0.25 \leqslant R \leqslant 0.40$. This fact gives some flexibility when designing the algorithm for computing the matrix Lambert W function.

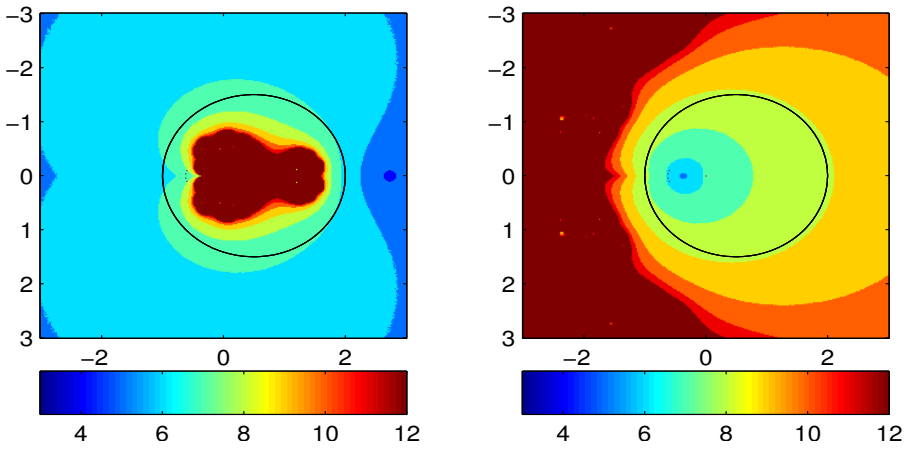

FIG. 3.1. *Level curves revealing the convergence of Newton's method to $W_0(z)$ with $|\operatorname{Re} z| \leqslant 3$ and $|\operatorname{Im} z| \leqslant 3$ by choosing $z_0 = \varphi_0(z)$ (left) and $z_0 = \psi_0(z)$ (right), together with the circle $|z - 1/2| \leqslant 3/2$. Dark blue means less than four iterations for convergence, dark red more than eleven or lack of convergence.*

4. Newton's method for the matrix Lambert W function.

4.1. Local convergence. Let $F(W) = We^W - A$, where $W, A \in \mathbb{C}^{n\times n}$. To compute the Lambert W function we consider Newton's method applied to the equation $F(W) = 0$. The sequence is obtained through the iteration

$$X_{k+1} = X_k - DF(X_k)^{-1}[F(X_k)], \quad k = 0, 1, 2, \ldots, \tag{4.1}$$

for an initial guess X_0. Let W_* be a solution of $F(W) = 0$. From the standard theory of Newton's method we know that if $DF(W_*)$ is nonsingular then the sequence (4.1) converges quadratically to W_* for any X_0 sufficiently close to W_*.

The computation of the Newton step using (4.1) is not recommended since it would require dealing with $n^2 \times n^2$ matrices, in view of Lemma 4.2 below. Hence the Newton method cannot be used in this form.

However, assuming that the starting matrix is a polynomial in A it is possible to simplify Newton's method to a more computationally useful formula, which directly generalizes Newton's method in the scalar case:

$$Y_{k+1} = \left(Y_k^2 + Ae^{-Y_k}\right)(Y_k + I)^{-1} =: G(Y_k), \quad k = 0, 1, 2, \ldots, \tag{4.2}$$

for an initial guess $Y_0 \in \mathcal{P}(A)$, where $\mathcal{P}(A)$ is the set of polynomials in the matrix A. We call this iteration *simplified Newton method*. A condition for the equivalence of the regular and simplified Newton methods is proved in the next result.

THEOREM 4.1. *Let $A \in \mathbb{C}^{n\times n}$ and $X_0 = Y_0 \in \mathcal{P}(A)$. If both sequences X_k of (4.1) and Y_k of (4.2) are well defined then $X_k = Y_k \in \mathcal{P}(A)$ for all $k \geqslant 0$.*

Proof. We use an induction argument. Let $X_k = Y_k \in P(A)$. We know that $H_k := X_{k+1} - X_k$ is the unique solution of the linear system $DF(X_k)[Z] = -F(X_k)$. We prove that $K_k := Y_{k+1} - Y_k = (Ae^{-Y_k} - Y_k)(Y_k + I)^{-1}$ solves the same linear system, whence $Y_{k+1} = X_{k+1}$. Using Lemma 2.1, we have

$$\begin{aligned} DF(X_k)[K_k] = DF(Y_k)[K_k] &= K_k e^{Y_k} + Y_k e^{Y_k} K_k = K_k(Y_k + I)e^{Y_k} \\ &= (Ae^{-Y_k} - Y_k)e^{Y_k} = -F(Y_k) = -F(X_k), \end{aligned}$$

and the proof is completed. ∎

The equivalence of the two methods on the set $\mathcal{P}(A)$ does not imply that they are equivalent in a neighborhood of a solution of $F(W) = 0$. This is problematic because in finite precision arithmetic we cannot guarantee that for formula (4.2) the computed iterates $\widetilde{Y}_k$ stay in $\mathcal{P}(A)$. The components of $\widetilde{Y}_k$ in the subspace complementary to $\mathcal{P}(A)$, which are created by rounding errors, will be small initially but they can potentially grow greatly.

The theory in [20, sec. 4.9.4] says that for stability of an iteration with iteration function g at the fixed point W we need $Dg(W)$ to have bounded powers. The latter condition holds if $\rho(Dg(W)) < 1$ (where ρ denotes the spectral radius), or if $Dg(W)$ is idempotent. In this case errors will not be amplified to first order when iterates are in the neighborhood of W. To test the stability of the iteration G we find the Kronecker matrix form of the Fréchet derivative.

LEMMA 4.2. *Let W be a solution of $F(W) = 0$, and assume that -1 is not an eigenvalue of W. Then the Kronecker matrix representation of $DG(W)$ is*

$$K(W) = \big((W+I)^{-T} \otimes I\big)\big(W^T \otimes I - (I \otimes W) f_1(I \otimes W - W^T \otimes I)\big),$$

where $f_1(z) = (e^z - 1)/z$ for $z \neq 0$ and $f_1(0) = 1$.

Proof. First we compute the derivative of the function G, using the product and chain rules:

$$\begin{aligned} G(Y)[H] = {} & (YH + HY + ADe^{-Y}[-H])(Y+I)^{-1} \\ & - (Y^2 + Ae^{-Y})(Y+I)^{-1} H (Y+I)^{-1}. \end{aligned}$$

At a fixed point W we have $(W^2 + Ae^{-W})(W+I)^{-1} = W$ and thus

$$DG(W)[H] = (HW - ADe^{-W}[H])(W+I)^{-1}.$$

Using (2.1) we obtain

$$\begin{aligned} K(W) = {} & \big((W+I)^{-T} \otimes I\big) \\ & \times \big[W^T \otimes I - (I \otimes A)(I \otimes \exp(-W)) f_1(I \otimes W - W^T \otimes I)\big], \end{aligned}$$

which implies the result. ∎

Using Lemma 4.2 we can express the stability condition in terms of the spectrum $\Lambda(W)$ of W. A sufficient condition for stability is that

$$h(\lambda, \mu) < 1, \qquad h(\lambda, \mu) = \max_{\lambda, \mu \in \Lambda(W)} \left| \frac{\lambda - \mu f_1(\mu - \lambda)}{\lambda + 1} \right|. \tag{4.3}$$

It is not difficult to find examples where the condition is violated, so we conclude that Newton's method can be unstable in a rather large number of cases.

In our previous work we have on several occasions been able to manipulate an unstable iteration into an equivalent stable iteration [18], [21], [22], [23], [24]. Inspired by these results, we were able to find a stable variant of the simplified Newton method, namely,

$$\begin{cases} Z_0 \in \mathbb{C}^{n \times n}, \quad H_0 = (Ae^{-Z_0} - Z_0)(Z_0 + I)^{-1}, \\ Z_{k+1} = Z_k + H_k, & k = 0, 1, \ldots \\ H_{k+1} = \big((Z_k + (Z_k + I)H_k)e^{-H_k} - Z_{k+1}\big)(Z_{k+1} + I)^{-1}. \end{cases} \tag{4.4}$$

The equivalence of (4.2) and (4.4) and the stability of (4.4) are proved in the following result. When the sequences are converging we have $Z_k \to W$ and $H_k \to 0$ and thus we are interested in fixed points of the iteration of the type $[W^T\ 0]^T$.

THEOREM 4.3. *Let $A \in \mathbb{C}^{n\times n}$ and $Y_0 = Z_0 \in \mathcal{P}(A)$. The sequence Y_k of (4.2) is well defined if and only if the sequence Z_k of (4.4) is well defined and we have $Y_k = Z_k$ for $k \geqslant 0$. Moreover, if*

$$\varphi\left(\begin{bmatrix} Z \\ H \end{bmatrix}\right) = \begin{bmatrix} Z+H \\ \big((Z+(Z+I)H)e^{-H} - Z - H\big)(Z+H+I)^{-1} \end{bmatrix}$$

then for any $E, F \in \mathbb{C}^{n\times n}$ and fixed point $[W^T\ 0]^T$ such that -1 is not an eigenvalue of W we have

$$D\varphi\left(\begin{bmatrix} W \\ 0 \end{bmatrix}\right)\begin{bmatrix} E \\ F \end{bmatrix} = \begin{bmatrix} E+F \\ 0 \end{bmatrix},$$

and thus $D\varphi\left(\left[\begin{smallmatrix} W \\ 0 \end{smallmatrix}\right]\right)$ is idempotent.

Proof. The equivalence between the two sequences is proved by induction. First, observe that, when -1 is not in the spectrum of Y_0, we have $Y_1 - Y_0 = H_0$ and thus $Z_1 = Y_1$. Then, assuming that $k \geqslant 1$ steps have been performed and that $Y_h = Z_h$ for $h \leqslant k$, we can construct Y_{k+1} if and only if we can construct Z_{k+1}, that is when -1 is not an eigenvalue of Y_k. To prove that $Y_{k+1} = Z_{k+1}$ it is enough to prove that $H_k = Y_{k+1} - Y_k$, and this follows from

$$\begin{aligned} H_k &= \big((Y_{k-1} + (Y_{k-1}+I)H_{k-1})e^{-H_{k-1}} - Y_k\big)\,(Y_k+I)^{-1} \\ &= (Ae^{-Y_{k-1}}e^{-H_{k-1}} - Y_k)(Y_k+I)^{-1} = Y_{k+1} - Y_k, \end{aligned}$$

where we have used the equality $Ae^{-Y_k} = (Y_k+I)(Y_{k+1}-Y_k) + Y_k$, which follows from (4.2), the fact that $e^{U+V} = e^U e^V$ when U and V commute, and the inductive hypothesis.

The statement about $D\varphi$ can be proved by a straightforward but tedious computation that we will omit. ∎

The subtlety of the stabilization process can be seen from the fact that the variant of (4.4)

$$\begin{cases} Z_0 \in \mathbb{C}^{n\times n}, \quad H_0 = (Ae^{-Z_0} - Z_0)(Z_0+I)^{-1}, \\ Z_{k+1} = Z_k + H_k, & k = 0, 1, \dots \\ H_{k+1} = \big((Z_k + H_k(Z_k+I))e^{-H_k} - Z_{k+1}\big)\,(Z_{k+1}+I)^{-1} \end{cases} \tag{4.5}$$

is unstable. Notice that (4.5) differs from the stable variant (4.4) only in that the equation for H_{k+1} contains the product $H_k(Z_k+I)$ instead of $(Z_k+I)H_k$. This small change in the iteration results in the derivative not always having a spectral radius less than or equal to 1 at a fixed point of the type $[W^T\ 0]^T$.

4.2. Choice of the initial value. To design an algorithm for computing a primary matrix Lambert W function we need to devise a strategy to find a starting matrix for (4.4).

Let A be a square matrix whose distinct eigenvalues are $\lambda_1, \dots, \lambda_t$. Consider a matrix iteration of the type

$$X_{k+1} = R(X_k, A), \quad k = 0, 1, \dots,$$

where $R(x,a)$ is a rational function of both of its arguments. It is known [20, Thm. 4.15] (see also [24]) that with $X_0 \in \mathcal{P}(A)$ the iteration converges if the scalar sequences $x_{k+1}^{(j)} = R(x_k^{(j)}, \lambda_j)$, with $x_0^{(j)} = P(\lambda_j)$, converge for $j = 1, \dots, t$. It is not difficult to show that theorem is true also for $R(x,a) = (x^2 + ae^{-x})/(x+1)$. In particular, if $x_k^{(j)}$ converges to $W_{k_j}(\lambda_j)$, for any j, then the matrix sequence X_k converges to the solution of the matrix equation $We^W = A$ whose eigenvalues are $W_{k_1}(\lambda_1), \dots, W_{k_t}(\lambda_t)$, which is the primary matrix function $W_{k_1,\dots,k_t}(A)$.

In order to compute the matrix Lambert W function $W_{k_1,\dots,k_t}(A)$ through Newton's method it is therefore sufficient to choose a matrix in $\mathcal{P}(A)$ whose eigenvalues yield sequences converging to $W_{k_1}(\lambda_1), \dots, W_{k_t}(\lambda_t)$. In general this is very complicated, but it can be simplified if one needs an unmixed branch $W_k(A)$.

We have seen in section 3 a heuristic strategy for the choice of the initial values such that the Newton method should converge to any branch of $W_k(z)$. The initial value is chosen as one of the two approximations $\varphi_k(z)$ and $\psi_k(z)$ defined in (3.4). In particular, for $|k| > 1$ and for any $z \neq 0$ the initial value $x_0 = \varphi_k(z)$ gives a sequence x_k converging to $W_k(z)$. Thus, in order to obtain a sequence converging to $W_k(A)$ it is enough to choose $X_0 = \varphi_k(A)$.

For $|k| \leqslant 1$ the situation is more complicated, since we have two different strategies for the initial value depending on where z lies. In particular we have determined a region where we will choose $x_0 = \varphi_k(z)$ and a region where we will choose $x_0 = \psi_k(z)$. In the matrix case, we have a stable method for computing $W_k(A)$ for A whose eigenvalues all belong to $\mathcal{U}_k$, where we choose $X_0 = \varphi_k(A)$, and for A whose eigenvalues all belong to $\mathcal{V}_k$, where we choose $X_0 = \psi_k(A)$. Nevertheless, it may happen that A has eigenvalues in both regions and then neither $X_0 = \varphi_k(A)$ nor $X_0 = \psi_k(A)$ give the desired branch of the matrix Lambert W function.

To overcome this problem we use a strategy borrowed from the Schur–Parlett algorithm [14], [29]. We consider a Schur form T of the matrix A, ordered such that T is a 2×2 block matrix whose $(1,1)$ block contains the eigenvalues in $\mathcal{U}_k$ and whose $(2,2)$ block contains the eigenvalues in $\mathcal{V}_k$. Hence we write the matrix and its Lambert W function as

$$T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}, \qquad W_k(T) = \begin{bmatrix} X_{11} & X_{12} \\ 0 & X_{22} \end{bmatrix}, \tag{4.6}$$

where the size of T_{11} and X_{11} equals the number of eigenvalues λ of A belonging to $\mathcal{U}_k$, say m. Notice that m can be n or 0, in which case $T = T_{11}$ and $T = T_{22}$, respectively.

Since all the eigenvalues of T_{11} and T_{22} belong to the region of convergence of Newton's method with initial value $X_0 = \varphi(T)$ and $X_0 = \psi(T)$, respectively, we can use Newton's method to calculate $X_{11} = W_k(T_{11})$ and $X_{22} = W_k(T_{22})$, and since $W(T)$ is a primary function of T we have the commutativity condition

$$\begin{bmatrix} X_{11} & X_{12} \\ 0 & X_{22} \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{bmatrix} X_{11} & X_{12} \\ 0 & X_{22} \end{bmatrix}. \tag{4.7}$$

We can determine X_{12} by equating $(1,2)$ blocks to obtain the Sylvester equation

$$T_{11}X_{12} - X_{12}T_{22} = X_{11}T_{12} - T_{12}Z_2, \tag{4.8}$$

which has a unique solution since the matrices T_{11} and T_{22} have no eigenvalues in common [19, chap. 16].

This algorithm has one weak point: if an eigenvalue of T_{11} is near to an eigenvalue of T_{22} then the Sylvester equation may be ill-conditioned [19, sec. 16.3] even if the Lambert W function is well-conditioned. The problem can be addressed by observing that the circle which separates the two regions is not a sharp division of the two regions of convergence (see the discussion at the end of section 3), so a slightly smaller or larger circle yields the same convergence results. If two eigenvalues of A are very near to the circle but in different regions it is possible to reduce or increase the radius of the circle in order to let the two eigenvalues belong to the same region. Nevertheless, there is no a priori lower bound independent of A on the gap between the eigenvalues in T_{11} and those in T_{22}, while a lower bound dependent on the size n of A is $b_n = (R_M - R_m)/(n-1)$, where R_M and R_m are the largest and the smallest radii, respectively, that may be chosen for the splitting. In our experiments, even for matrices of medium size, say $n = 1000$, in the worst case we have not noticed a significant lack of accuracy due to ill conditioning of the Sylvester equation.

4.3. The algorithm. We now state our algorithm for computing the matrix Lambert W function using the stabilized Newton method. The algorithm is for the unmixed case. It can be adapted for the mixed case by a more sophisticated blocking strategy, but we will not consider that here.

ALGORITHM 1. Input: an integer k, a matrix $A \in \mathbb{C}^{n\times n}$ such that $W_k(A)$ exists. Output: $W_k(A)$.

1 Compute the (real) Schur form $A = Q^*TQ$ of A.

2a. If $k = 0$, choose $1.35 \leqslant r_0 \leqslant 1.60$ in order to maximize the gap between the eigenvalues in the two regions $\mathcal{V} = \{|z-1/2| < r_0\}$ and $\mathcal{U} = \{|z-1/2| \geqslant r_0\}$.

2b. If $k = 1$, choose $0.25 \leqslant r_0 \leqslant 0.40$ in order to maximize the gap between the eigenvalues in the two regions $\mathcal{V} = \{|z+1/2| < r_0,\ \operatorname{Im} z < 0\}$ and $\mathcal{U} = \mathbb{C} \setminus \mathcal{V} = \{|z+1/2| \geqslant r_0\} \cup \{|z+1/2| < r_0,\ \operatorname{Im} z \geqslant 0\}$.

2c. If $k = -1$, choose $0.25 \leqslant r_0 \leqslant 0.40$ in order to maximize the gap between the eigenvalues in the two regions $\mathcal{V} = \{|z+1/2| < r_0,\ \operatorname{Im} z \geqslant 0\}$ and $\mathcal{U} = \mathbb{C} \setminus \mathcal{V} = \{|z+1/2| \geqslant r_0\} \cup \{|z+1/2| < r_0,\ \operatorname{Im} z < 0\}$.

2d. If $|k| > 1$ set $\mathcal{U} = \mathbb{C}$ and $\mathcal{V} = \emptyset$.

3. Reorder the Schur form of A in order to have T as in (4.6), such that the eigenvalues of T_{11} (possibly an empty matrix) belong to $\mathcal{U}$ and the eigenvalues of T_{22} (possibly an empty matrix) belong to $\mathcal{V}$ (this task can be accomplished by applying a unitary similarity as explained in [6]).

4a. If T_{11} is nonempty then compute $X_{11} = W_k(T_{11})$ as the limit of (4.4) with $Z_0 = L_1 - L_2 + L_3$, where $L_1 = \ln T_{11} + 2\pi i k I$, $L_2 = \ln L_1$, and $L_3 = L_2 L_1^{-1}$. For better accuracy, H_0 should be evaluated as $H_0 = (L_1 e^{-L_3} - W)(I+W)^{-1}$, where $W = L_1 + L_2 - L_3$.

4b. If T_{22} is nonempty (which implies $|k| \leqslant 1$) then compute $X_{22} = W_k(T_{22})$ as the limit of (4.4) with $Z_0 = (-1)^k(2eT_{22} + 2I)^{1/2} - I$.

5. Solve the Sylvester equation (4.8) for X_{12} (this task can be accomplished using the Bartels and Stewart algorithm [7]).

6. Form $W_k(T) = \begin{bmatrix} X_{11} & X_{12} \\ 0 & X_{22} \end{bmatrix}$ and recover $W_k(A)$ by reversing the similarities of steps 3 and 1.

The matrix exponentials and logarithms in step 4 are evaluated using the algorithms of Al-Mohy and Higham [1], [2], which exploit triangularity. The matrix square roots in step 4b are evaluated using the Björck–Hammarling recurrence [8], with the efficient blocked implementation of [16].

It is worth pointing out that, in spite of the similarities between the Lambert W function and the logarithm, the computation of the former is much more expensive than that of the latter. In particular, two matrix logarithms can be required to obtain the initial value for the Newton iteration (see line 4a in Algorithm 1), while each step of the Newton iteration requires the computation of one matrix exponential. The total cost of the algorithm is still $O(n^3)$ operations, but with a constant much larger than that for the logarithm using the algorithm of [2].

We also note that Newton iterations for matrix functions are invariably started with a scalar multiple of I or A [20]; our choice of nontrivial functions of A as starting matrices is novel.

When A is real and has no real eigenvalues on $(-\infty, -1/e)$, it can be proved that $W_0(A)$ is real (for instance, using [21, Thm 3.2]). In this case, a real Schur decomposition can be employed in the algorithm, since the sets $\mathcal{U}$ and $\mathcal{V}$ are symmetric about the real axis, so a nonreal eigenvalue and its complex conjugate belong to the same convergence region and hence can go in the same block. (In contrast, for the Schur–Parlett algorithm usually only the complex Schur form can be used [4], [14]). The logarithm evaluations are now done using the algorithm of Al-Mohy, Higham, and Relton [3], which is designed for real matrices and works entirely in real arithmetic.

5. Numerical experiments. We present some numerical experiments to illustrate the behaviour of our algorithm in finite precision arithmetic. The tests are performed using MATLAB R2011b, for which the unit roundoff is $u = 2^{-53} \approx 1.1 \times 10^{-16}$.

The algorithm is compared with a diagonalization approach that computes $A = M \operatorname{diag}(\lambda_1, \dots, \lambda_n) M^{-1}$ and then forms $W_k(A) = M \operatorname{diag}\big(W_k(\lambda_1), \dots, W_k(\lambda_n)\big) M^{-1}$. This approach has been used in earlier work [5] but is obviously applicable only when A has a complete set of eigenvectors.

The Schur–Parlett algorithm (implemented in the MATLAB function `funm`) is of limited use, since it is designed for entire functions. For multivalued functions it could group eigenvalues across a branch cut, leading to a wrong branch. The possibility of constructing a suitable modification of the Schur–Parlett algorithm for multivalued function is still under investigation.

As a measure of the accuracy of a computed approximation $\widehat{W}$ to $W_k(A)$ we consider the Frobenius norm relative error $\|\widetilde{W} - \widehat{W}\|_F / \|\widetilde{W}\|_F$, where $\widetilde{W}$ is a reference solution computed by using the diagonalization approach at high precision (with the variable precision arithmetic of the Symbolic Math Toolbox) and then rounding the result to double precision.

Another measure of quality is the relative residual of $\widehat{W}$,

$$\rho(\widehat{W}, A) = \frac{\|\widehat{W} \exp(\widehat{W}) - A\|_F}{\|\widehat{W} \exp(\widehat{W})\|_F + \|A\|_F}. \tag{5.1}$$

As pointed out by Deadman and Higham [15], the difficulty with such residuals is knowing how small we can reasonably expect them to be. Rather than carry out a perturbation analysis as in [15], here we will take advantage of the availability of $\widetilde{W}$ and we will simply compare the residual of $\widehat{W}$ with that of $\widetilde{W}$.

Experiment 1. To compare our Algorithm 1 with the diagonalization approach, we consider the matrix

$$A(\varepsilon) = \begin{bmatrix} 1 & 1 \\ 0 & 1+\varepsilon \end{bmatrix},$$

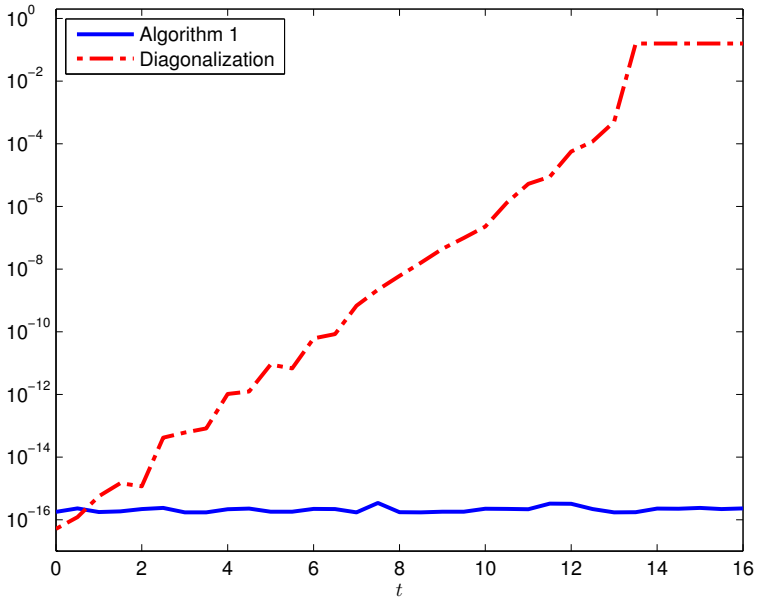

FIG. 5.1. *Relative errors in computing $W_{-1}(A)$, with A of (5.1), with $\varepsilon = 10^{-t}$, for Algorithm 1 and the eigenvector approach.*

whose eigenvector matrix becomes increasingly ill conditioned as $\varepsilon \to 0$. In Figure 5.1 we plot the relative error of the two methods for computing $W_{-1}(A)$, for $\varepsilon = 10^{-t}$ with t varying from 1 to 16. As expected the diagonalization approach loses accuracy as ε tends to zero while our method has accuracy independent of the eigenvector conditioning.

A similar figure is obtained considering any other branch of the Lambert W function.

EXPERIMENT 2. We pick 47 matrices of size 10×10 from the MATLAB `gallery` function. We compare the error in computing $W_0(A)$ for Algorithm 1 and the diagonalization approach. In Figures 5.2 and 5.3 we show the relative residuals and relative errors, respectively. Figure 5.3 also shows an estimate of $\mathrm{cond}(W_0, A)u$, where $\mathrm{cond}(W_0, A)$ is a 1-norm condition number for W_0 defined in the usual way for matrix functions [20, sec. 3.1] and estimated using the code `funm_condest1` from the Matrix Function Toolbox [17], [20, Alg. 3.20].

As one can see in Figure 5.2 the residual for Algorithm 1 is never more than a couple of orders of magnitude larger than that of the reference solution, whereas diagonalization gives much larger residuals than Algorithm 1 on a number of problems. Consistent with this behavior, Figure 5.3 shows that Algorithm 1 has error less than $\mathrm{cond}(W_0, A)u$ in almost every case, whereas the error for diagonalization greatly exceeds $\mathrm{cond}(W_0, A)u$ in several cases. For all the matrices tested the Newton iteration required no more than 9 steps to converge for each of the (at most two) diagonal blocks.

EXPERIMENT 3. We repeat the previous test but now computing $W_{-1}(A)$, in order to show that our method is able to compute also the non-principal branches of the Lambert W function. We have removed singular matrices from the test, since $W_{-1}(z)$ has a singularity for $z = 0$. The residuals are plotted in Figure 5.4, which

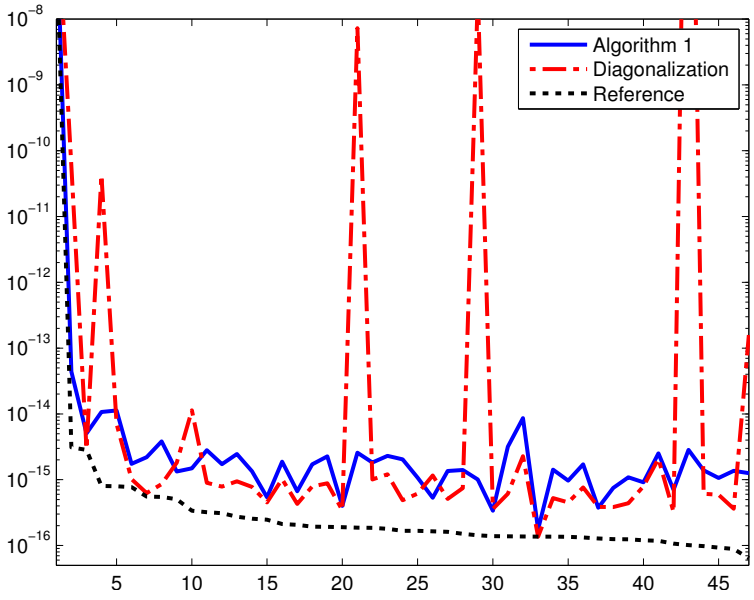

FIG. 5.2. *Residuals in computing $W_0(A)$, where $A \in \mathbb{C}^{n\times n}$ ranges over a set of 47 matrices chosen from the MATLAB* `gallery` *function. The results are ordered by nonincreasing residual of the reference solution.*

shows a similar behaviour to the case of $W_0(A)$.

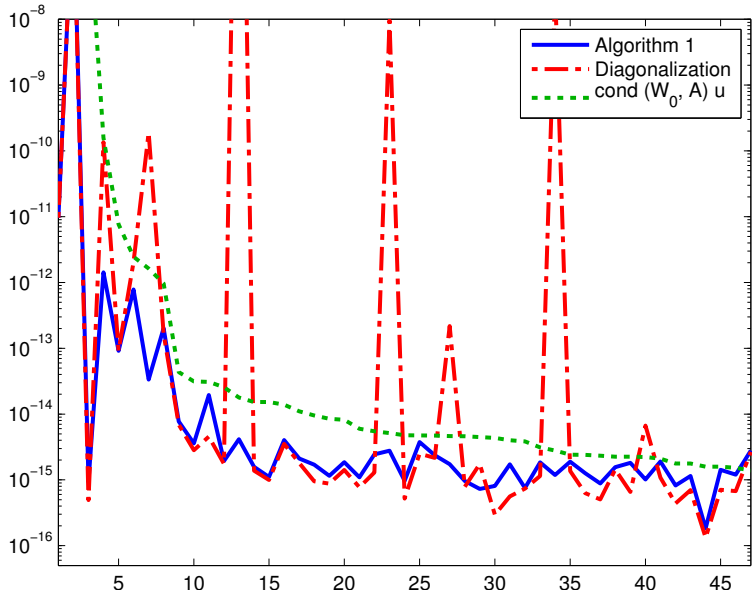

FIG. 5.3. *Relative errors in computing* $W_0(A)$*, where* $A \in \mathbb{C}^{n\times n}$ *ranges over* 47 *matrices chosen from the MATLAB* `gallery` *function. The results are ordered by nonincreasing value of* $\mathrm{cond}(W_0, A)u$.

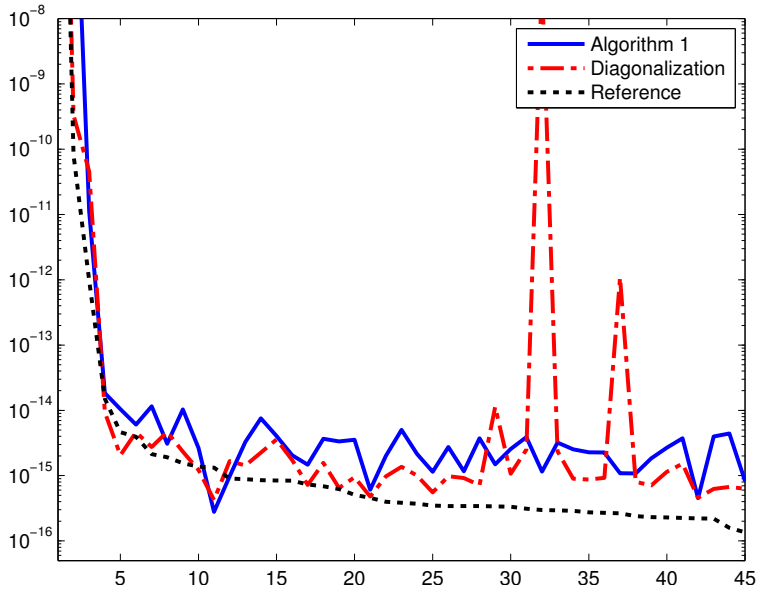

FIG. 5.4. *Residuals in computing* $W_{-1}(A)$*, where* $A \in \mathbb{C}^{n\times n}$ *varies over* 45 *matrices chosen from the MATLAB* `gallery` *function. The results are ordered by nonincreasing residual of the reference solution.*

6. Concluding remarks. Corless and Jeffrey [12] call the Lambert W function "the first nontrivial example of a multivalued function", since the logarithm (its close relative) and the inverse trigonometric functions have such a regular branch structure that a new notation is not needed to specify the branch. We have presented the first numerically reliable algorithm for computing an arbitrary branch of the matrix Lambert W function. Corless and Jeffrey also comment that "the multivalued nature of W 'stress tests' naming conventions, numerics on branches, computer-aided analysis, and the results of series computations". To that list we can add the evaluation of a non-entire function of a triangular matrix. To derive our algorithm we had to construct a numerically stable Newton iteration, formulate starting matrices that are nontrivial functions of the matrix of interest (in contrast to other Newton iterations for matrix functions), and use a Schur form with a novel blocking. Our algorithm builds on previous work on matrix functions in that it employs algorithms for the matrix exponential, logarithm, and square root. Empirically, the algorithm produces residuals within a couple of orders of magnitude of those of a reference solution.

Given the wide range of situations in which the scalar Lambert W function arises [10], we can expect to see more of the matrix version, both in practical applications, such as those cited in section 1, and in more theoretical studies, such as that in [27] concerning chain rules for matrix functions.

Acknowledgements. We wish to thank Rob Corless and an anonymous referee whose suggestions improved the presentation of the paper. We would also like to thank Nicola Guglielmi and Nick Schraudolph for useful discussions on the application to delay differential equations and the algorithms for the scalar Lambert W function, respectively.

REFERENCES

[1] Awad H. Al-Mohy and Nicholas J. Higham. A new scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 31(3):970–989, 2009.

[2] Awad H. Al-Mohy and Nicholas J. Higham. Improved inverse scaling and squaring algorithms for the matrix logarithm. *SIAM J. Sci. Comput.*, 34(4):C153–C169, 2012.

[3] Awad H. Al-Mohy, Nicholas J. Higham, and Samuel D. Relton. Computing the Fréchet derivative of the matrix logarithm and estimating the condition number. *SIAM J. Sci. Comput.*, 35(4):C394–C410, 2013.

[4] Mary Aprahamian and Nicholas J. Higham. The matrix unwinding function, with an application to computing the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 35(1):88–109, 2014.

[5] Farshid Maghami Asi and A. Galip Ulsoy. Analysis of a system of linear delay differential equations. *Journal of Dynamic Systems, Measurement, and Control*, 125:215–223, 2003.

[6] Zhaojun Bai and James W. Demmel. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl.*, 186:73–95, 1993.

[7] R. H. Bartels and G. W. Stewart. Algorithm 432: Solution of the matrix equation $AX + XB = C$. *Comm. ACM*, 15(9):820–826, 1972.

[8] Åke Björck and Sven Hammarling. A Schur method for the square root of a matrix. *Linear Algebra Appl.*, 52/53:127–140, 1983.

[9] Rudy Cepeda-Gomez and Wim Michiels. Some special cases in the stability analysis of multi-dimensional time-delay systems using the matrix Lambert W function. *arXiv preprint arXiv:1406.4738*, 2014. To appear in Automatica.

[10] 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.*, 5(4):329–359, 1996.

[11] Robert M. Corless, Hui Ding, Nicholas J. Higham, and David J. Jeffrey. The solution of $S\exp(S) = A$ is not always the Lambert W function of A. In *ISSAC '07: Proceedings of the* 2007 *International Symposium on Symbolic and Algebraic Computation*, New York, 2007, pages 116–121. ACM Press.

[12] Robert M. Corless and David J. Jeffrey. The Wright ω function. In *Artificial Intelligence, Automated Reasoning, and Symbolic Computation*, Jacques Calmet, Belaid Benhamou,

Olga Caprotti, Laurent Henocque, and Volker Sorge, editors, volume 2385 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2002, pages 76–89.

[13] Robert M. Corless and David J. Jeffrey. The Lambert W function. In *The Princeton Companion to Applied Mathematics*, Nicholas J. Higham, Mark Dennis, Paul Glendinning, Paul Martin, Fadil Santosa, and Jared Tanner, editors, Princeton University Press, Princeton, NJ, USA, 2015.

[14] Philip I. Davies and Nicholas J. Higham. A Schur–Parlett algorithm for computing matrix functions. *SIAM J. Matrix Anal. Appl.*, 25(2):464–485, 2003.

[15] Edvin Deadman and Nicholas J. Higham. Testing matrix function algorithms using identities. MIMS EPrint 2014.13, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, March 2014. 15 pp. Revised January 2015. To appear in ACM Trans. Math. Software.

[16] Edvin Deadman, Nicholas J. Higham, and Rui Ralha. Blocked Schur algorithms for computing the matrix square root. In *Applied Parallel and Scientific Computing: 11th International Conference, PARA 2012, Helsinki, Finland*, P. Manninen and P. Öster, editors, volume 7782 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2013, pages 171–182.

[17] Nicholas J. Higham. The Matrix Function Toolbox. `http://www.maths.manchester.ac.uk/~higham/mftoolbox`.

[18] Nicholas J. Higham. Stable iterations for the matrix square root. *Numer. Algorithms*, 15(2): 227–242, 1997.

[19] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Second edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002. xxx+680 pp. ISBN 0-89871-521-0.

[20] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008. xx+425 pp. ISBN 978-0-898716-46-7.

[21] Nicholas J. Higham, D. Steven Mackey, Niloufer Mackey, and Françoise Tisseur. Functions preserving matrix groups and iterations for the matrix square root. *SIAM J. Matrix Anal. Appl.*, 26(3):849–877, 2005.

[22] Bruno Iannazzo. On the Newton method for the matrix pth root. *SIAM J. Matrix Anal. Appl.*, 28(2):503–523, 2006.

[23] Bruno Iannazzo. *Numerical solution of certain nonlinear matrix equations*. PhD thesis, Università di Pisa, 2007.

[24] Bruno Iannazzo. A family of rational iterations and its application to the computation of the matrix pth root. *SIAM J. Matrix Anal. Appl.*, 30(4):1445–1462, 2008.

[25] Elias Jarlebring and Tobias Damm. The Lambert W function and the spectrum of some multidimensional time-delay systems. *Automatica J. IFAC*, 43(12):2124–2128, 2007.

[26] W. Kahan. Branch cuts for complex elementary functions or much ado about nothing's sign bit. In *The State of the Art in Numerical Analysis*, A. Iserles and M. J. D. Powell, editors, Oxford University Press, 1987, pages 165–211.

[27] Wen-Xiu Ma and Boris Shekhtman. Do the chain rules for matrix functions hold without commutativity? *Linear and Multilinear Algebra*, 58(1):79–87, 2010.

[28] Frank W. J. Olver, Daniel W. Lozier, Ronald. F. Boisvert, and Charles W. Clark, editors. *NIST Handbook of Mathematical Functions*. Cambridge University Press, Cambridge, UK, 2010. xv+951 pp. `http://dlmf.nist.gov`. ISBN 978-0-521-14063-8.

[29] B. N. Parlett. A recurrence among the elements of functions of triangular matrices. *Linear Algebra and Appl.*, 14(2):117–121, 1976.

[30] Nic Schraudolph. Personal communication.

[31] Willi-Hans Steeb and Yorick Hardy. Exponential of a matrix, a nonlinear problem, and quantum gates. *J. Math. Phys.*, 56(1):012201, 2015.

[32] Sun Yi, Patrick W. Nelson, and A. Galip Ulsoy. Delay differential equations via the matrix Lambert W function and bifurcation analysis: application to machine tool chatter. *Math. Biosci. Eng.*, 4(2):355–368, 2007.

[33] Sun Yi, Patrick W. Nelson, and A. Galip Ulsoy. Controllability and observability of systems of linear delay differential equations via the matrix Lambert W function. *IEEE Trans. Automat. Control*, 53(3):854–860, 2008.