MANCHESTER
1824
The University of Manchester

An Algorithm for the Matrix Lambert W Function

Massimiliano Fasi* Nicholas J. Higham* Bruno Iannazzo+

*The University of Manchester, UK

+Università di Perugia, Italy

Atlanta, 26 October 2015

The Lambert W function

The Lambert W function for any $z \in \mathbb{C}$ is any of the solutions of

$$z = W(z)\,\mathrm{e}^{W(z)}.$$

It is in some sense the inverse of

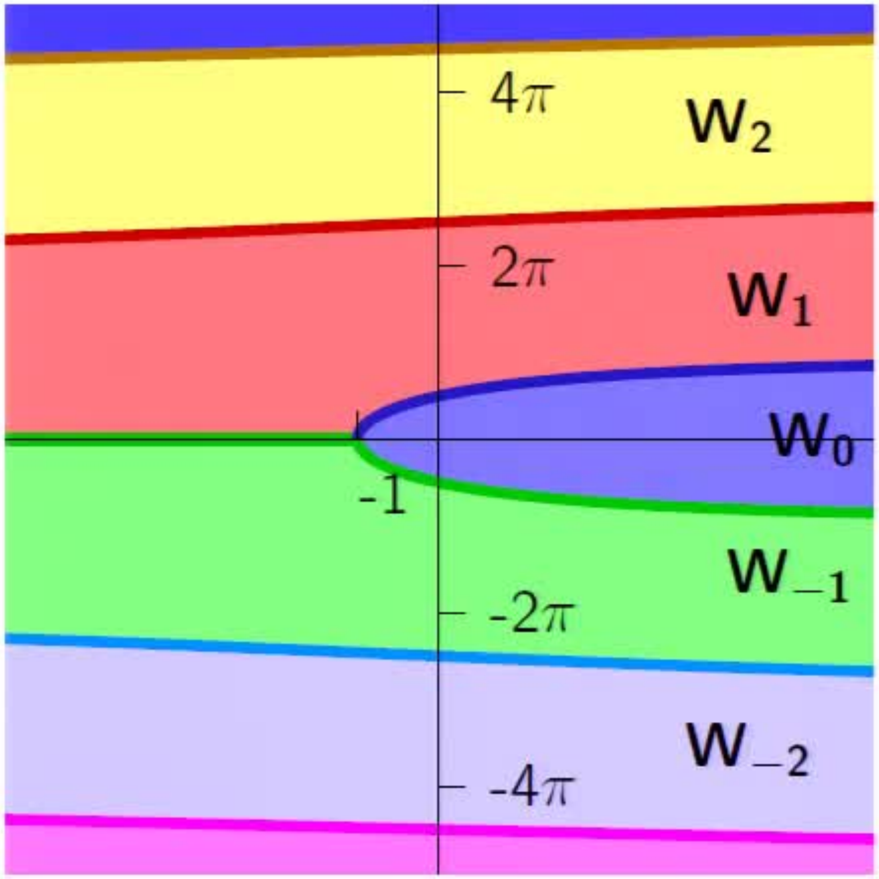
$$w \mapsto w\,\mathrm{e}^{w}.$$

Edward Maitland Wright.
Solution of the equation $z\,\mathrm{e}^{z} = a$.
Proceedings of the Royal Society of Edinburg (A):193-203, 1959.

Branches of the Lambert W function

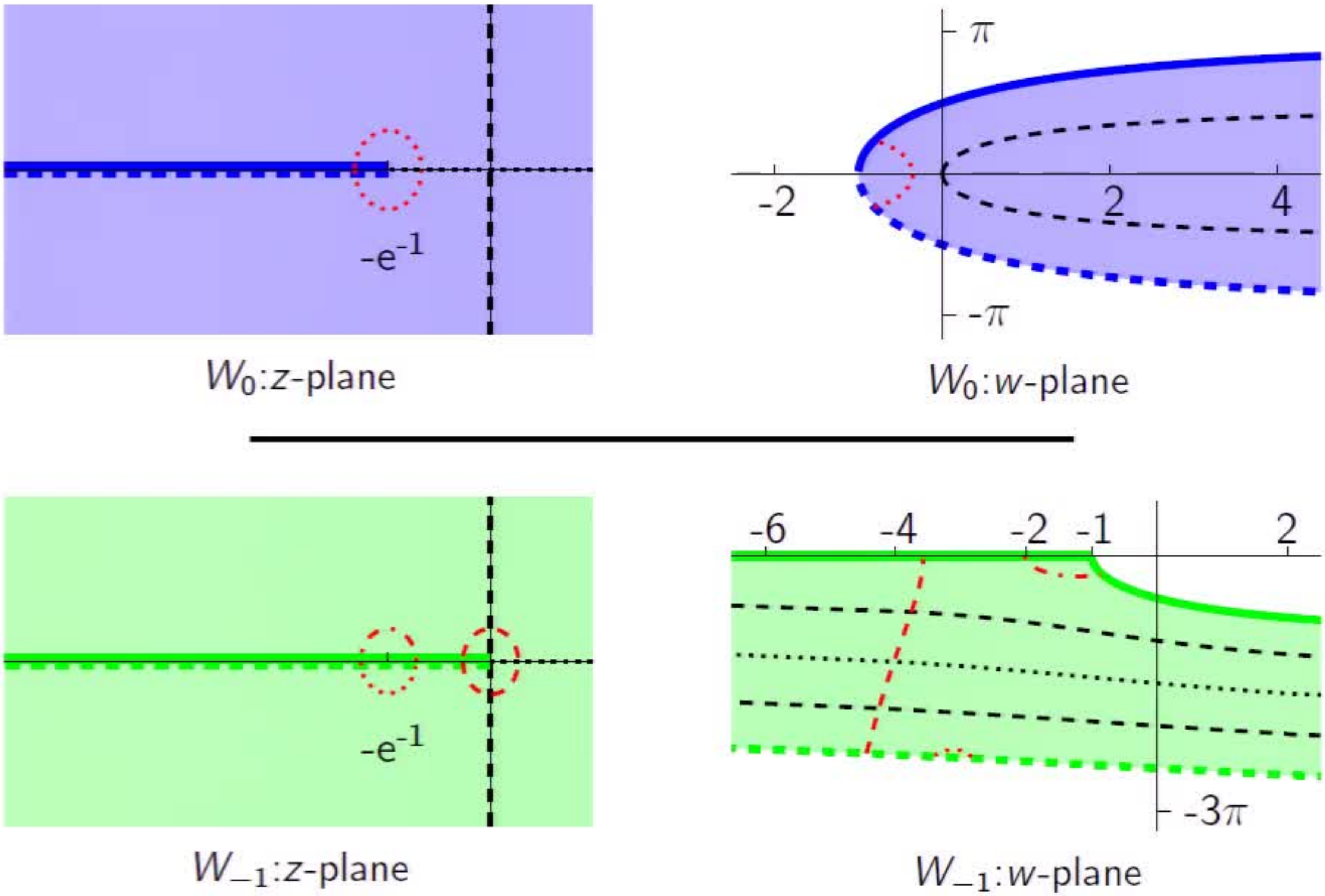
$$W_k : \mathcal{I}_k \to \mathbf{W}_k, \quad k \in \mathbb{Z}$$

$$\mathcal{I}_k = \begin{cases} \mathbb{C} - \{-e^{-1}\}, & k = 0, \\ \mathbb{C} - \{0\}, & k \neq 0, \end{cases}$$

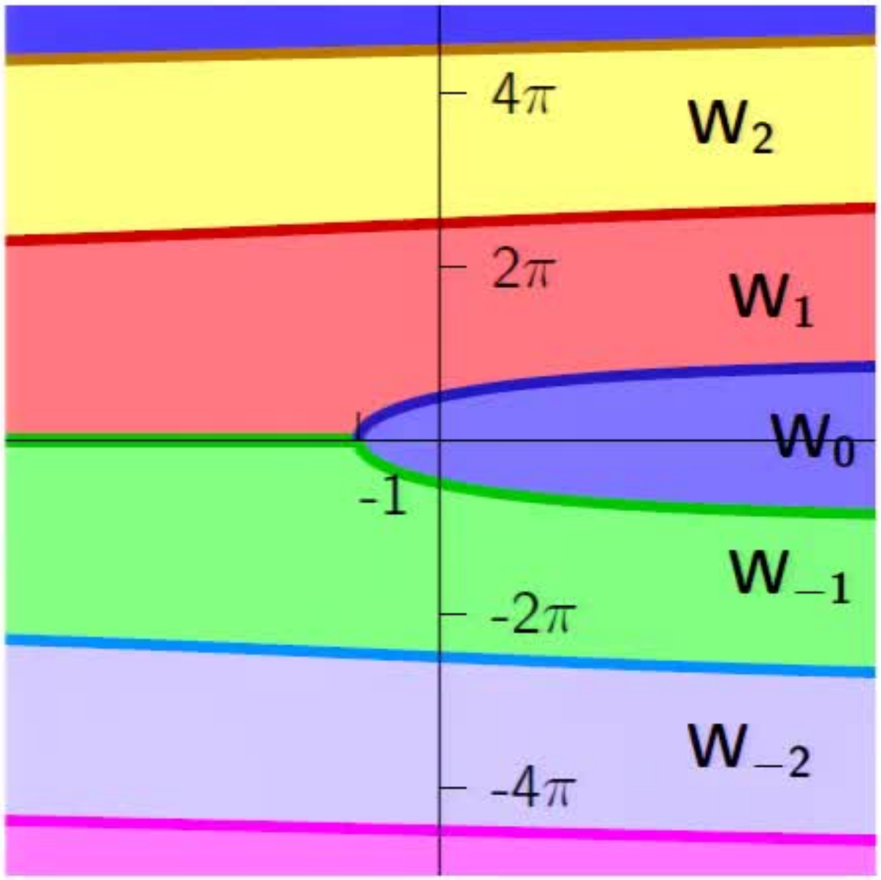


The w-plane

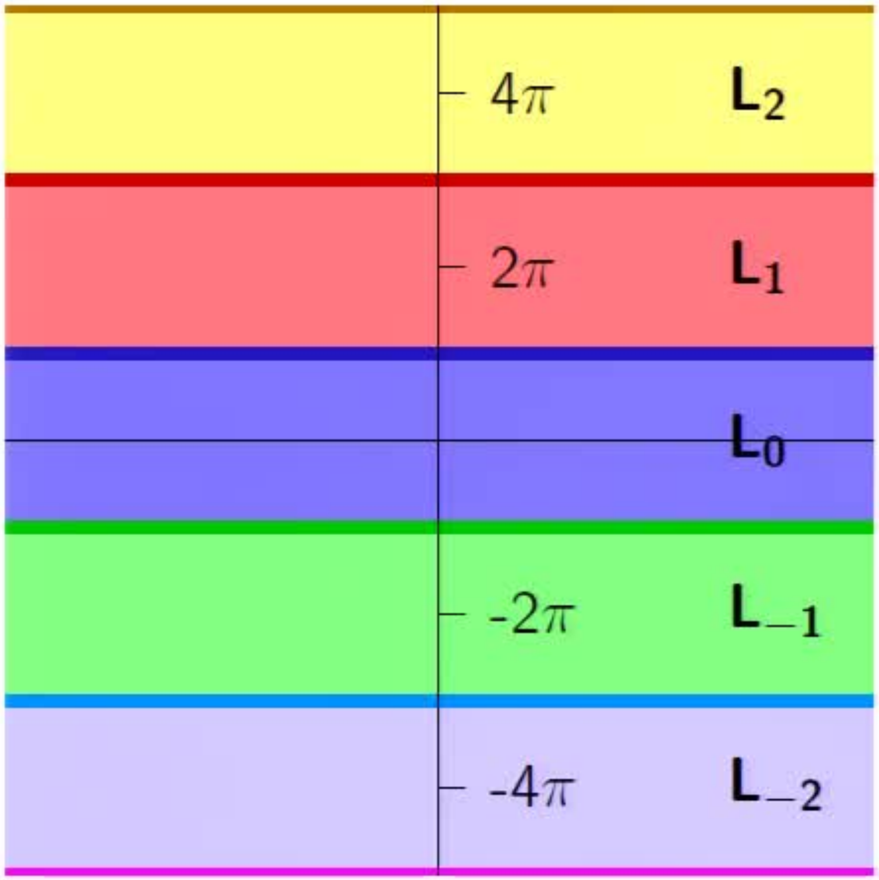
Two interesting branches



Lambert W and logarithm



Lambert W



Logarithm

Rootfinding algorithms

Alternative formulation

Any branch of W is solution of the non-linear equation

$$w e^{w} - z = 0.$$

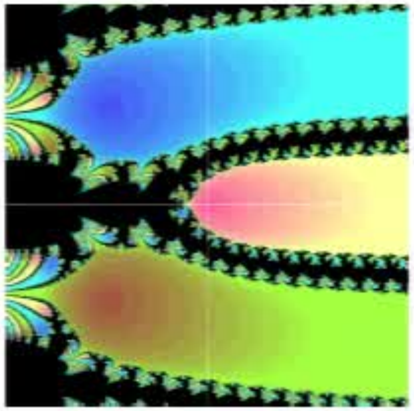
- Newton's method

$$g_n(x_k) = x_k - \frac{x_k e^{x_k} - z}{(x_k + 1) e^{x_k}}, \qquad k = 0, 1, \ldots$$

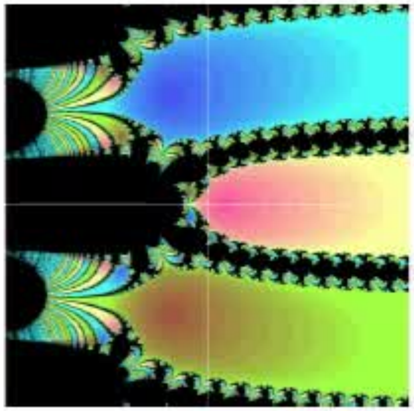
- Halley's method

$$g_h(x_k) = x_k - \frac{2\,(x_k e^{x_k} - z)}{2(x_k + 1) e^{x_k} - (x_k e^{x_k} - z)\,\frac{(x_k+2)}{(x_k+1)}}, \qquad k = 0, 1, \ldots$$

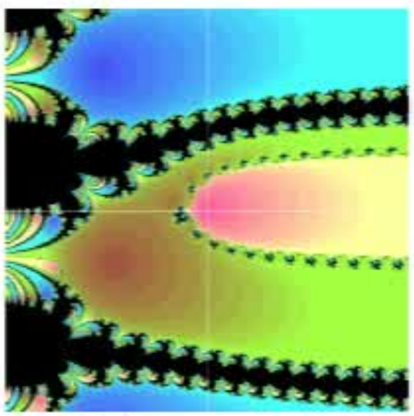
Basins of attraction



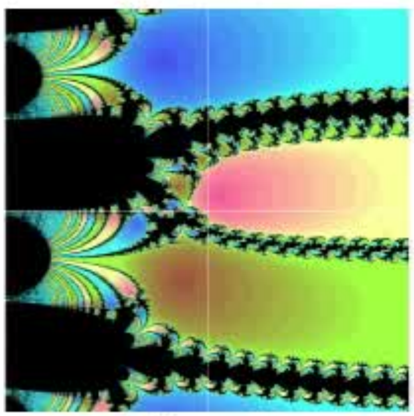
(a) $z = 0.1$



(b) $z = 1$

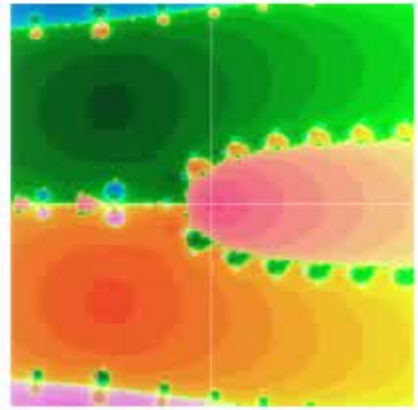


(c) $z = 0.1\mathrm{i}$

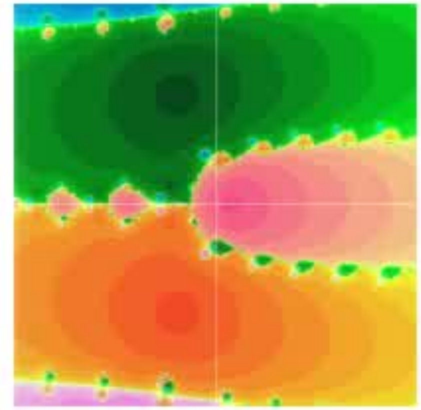


(d) $z = \mathrm{i}$

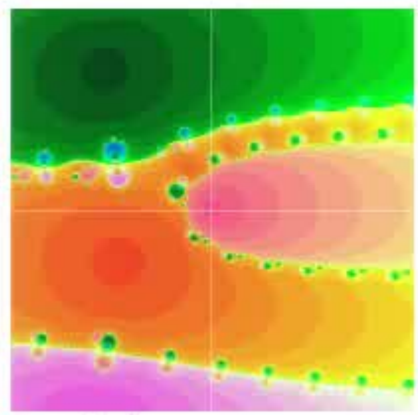
Newton's method



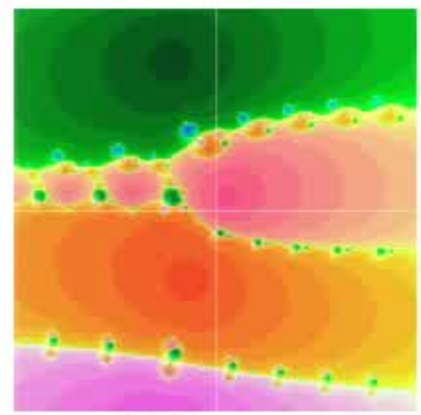
(e) $z = 0.1$



(f) $z = 1$



(g) $z = 0.1\mathrm{i}$



(h) $z = \mathrm{i}$

Halley's method

Asymptotic expansions

Well known series expansions of W_k are

- for $z \to -\mathrm{e}^{-1}$

$$W_k(z) = \sum_{\ell=0}^{\infty} \mu_\ell p_k^\ell = -1 + p_k - \frac{1}{3}p_k^2 + \frac{11}{72}p_k^3 + \dots, \quad k = -1, 0, 1$$

$$p_0 = \sqrt{2(ez+1)}, \quad p_{-1} = p_1 = -\sqrt{2(ez+1)}$$

- for $z \to \infty$

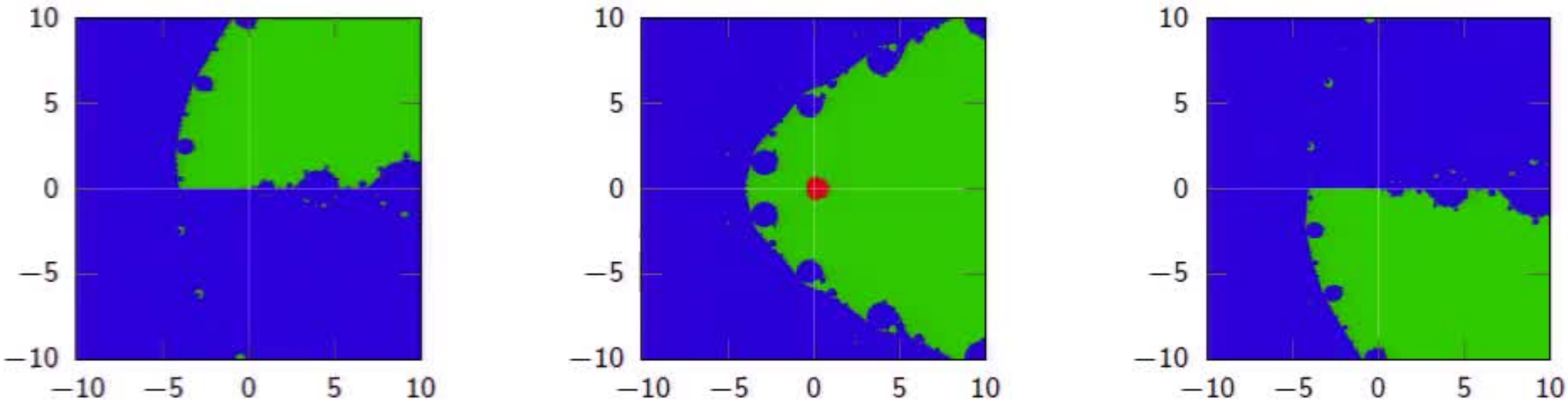
$$W_k(z) = \log z + 2\pi ik - \log(\log z + 2\pi ik) + \sum_{j=0}^{\infty}\sum_{m=1}^{\infty} c_{jm} \log^m(\log z + 2\pi ik)(\log z + 2\pi ik)^{-j-m}$$

R. M. Corless, G. H. Gonnet, D. E. Hare, D. J. Jeffrey, D. E. Knuth.
On the Lambert W function.
Advances in Computational mathematics 5(4):329-359, 1996.

Newton method – convergence regions

Calling $\psi(z)$ and $\varphi(z)$ are the first two terms of the expansions at ∞ and $-\mathrm{e}^{-1}$ respectively, we define

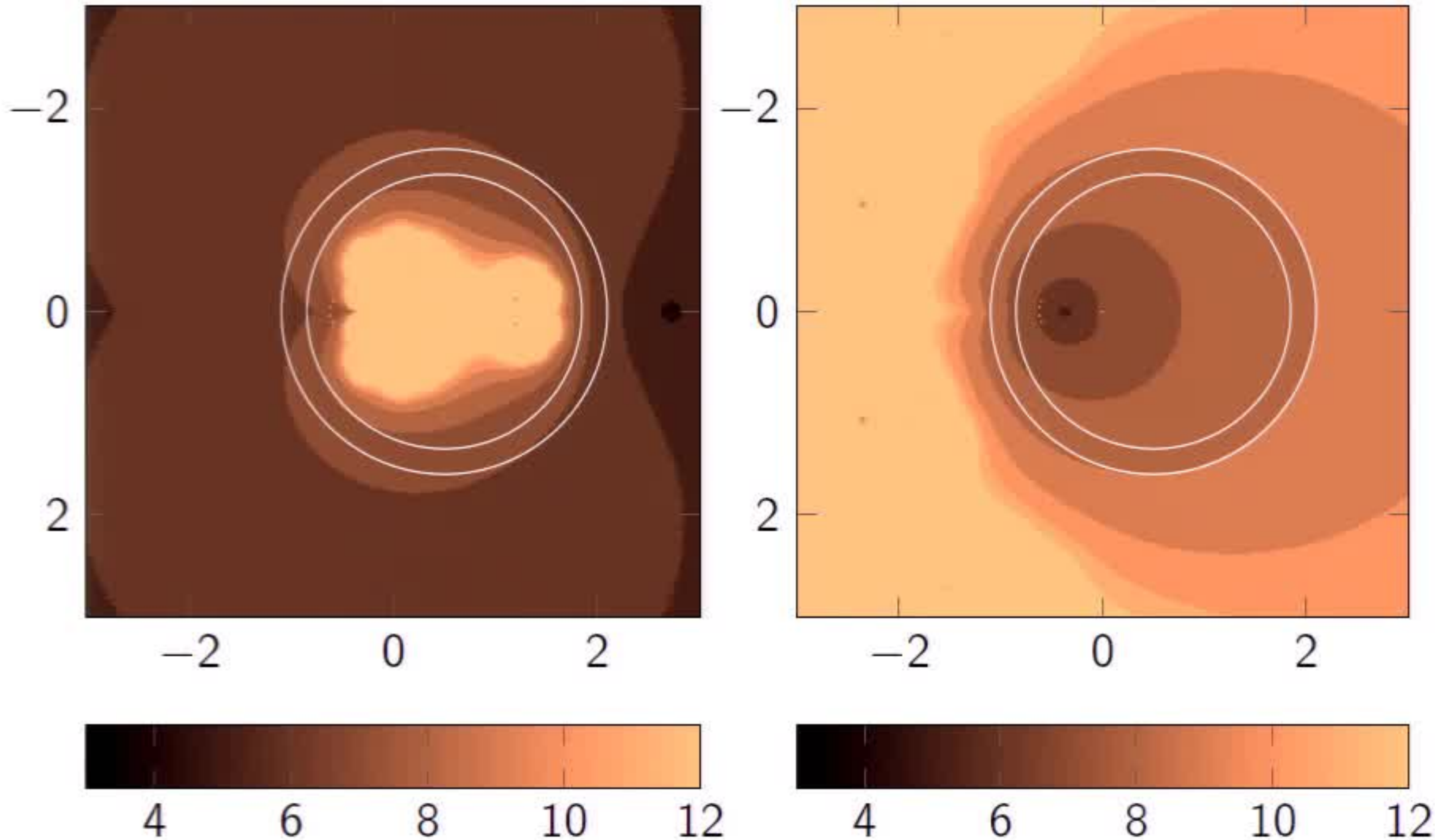
$$\mathcal{V}_k = \{z \in \mathbb{C} : \{x_i\}_i \to W_k(z),\ x_0 = \psi(z)\}$$
$$\mathcal{U}_k = \{z \in \mathbb{C} : \{x_i\}_i \to W_k(z),\ x_0 = \varphi(z)\}$$
$$\mathcal{S}_k = \mathcal{V}_k \cap \mathcal{U}_k$$



(a) Branch W_{-1} (b) Branch W_0 (c) Branch W_1

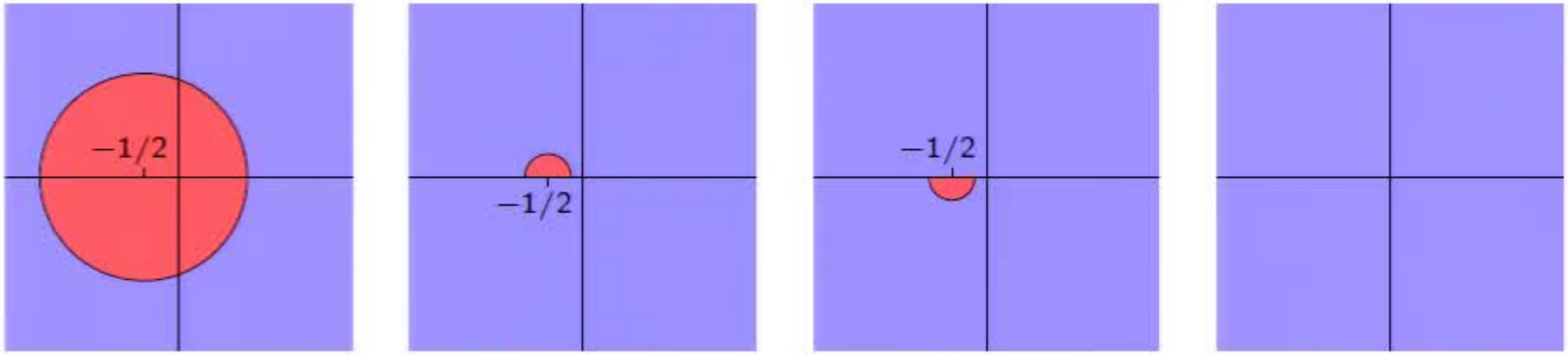
Convergence regions

Newton method – convergence speed



Number of iterations using expansions at ∞ (left) and at $-e^{-1}$ (right)

Initial values



(a) $k = 0$ (b) $k = -1$ (c) $k = 1$ (d) $|k| > 1$

Initial value choice strategies for Newton's methods ($z \to -\mathrm{e}^{-1}$, $z \to \infty$).

Nicol N. Schraudolph.
Implementation of the W function.
ftp://ftp.idsia.ch/pub/nic/W.m.

True and Simplified Newton's methods

Matrix function: $F : \mathcal{I} \subset \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$, $W \mapsto F_A(W) = W\mathrm{e}^{W} - A$

Desired solution: $W \in \mathbb{C}^{n\times n}$ such that $W \exp(W) = A$

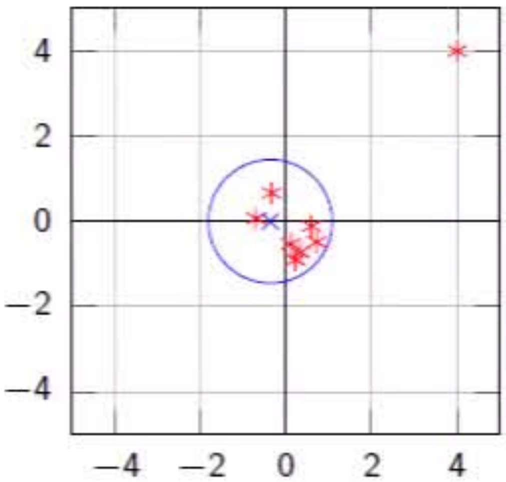
Newton's method
$$\begin{cases} X_{k+1} &= X_k - DF_{X_0}^{-1}[F(X_0)] \\ X_0 &= \widetilde{W} \end{cases}$$

Simplified version
$$\begin{cases} Y_{k+1} &= \left(Y_k^2 + A\exp(-Y_k)\right)\left(Y_k + I\right)^{-1} \\ Y_0 &= \widetilde{W} \end{cases}$$

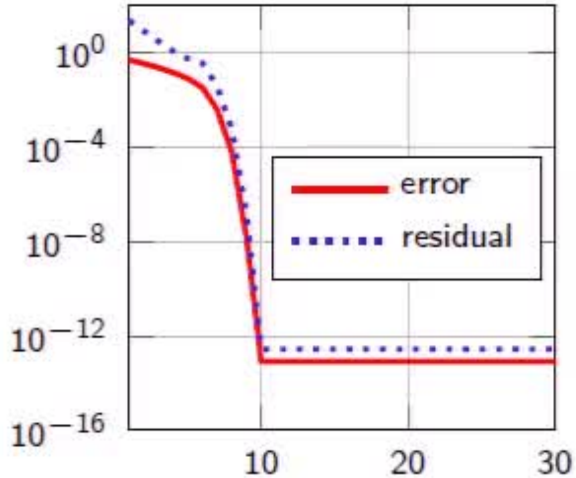
Equivalence of the iterations

Let $A \in \mathbb{C}^{n\times n}$, $X_0 = Y_0 \in \mathbb{C}^{n\times n}$. If both $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ are well defined and $X_0 \in \mathcal{P}(A)$, then $X_k = Y_k \in \mathcal{P}(A)$ for all $k \geq 0$.

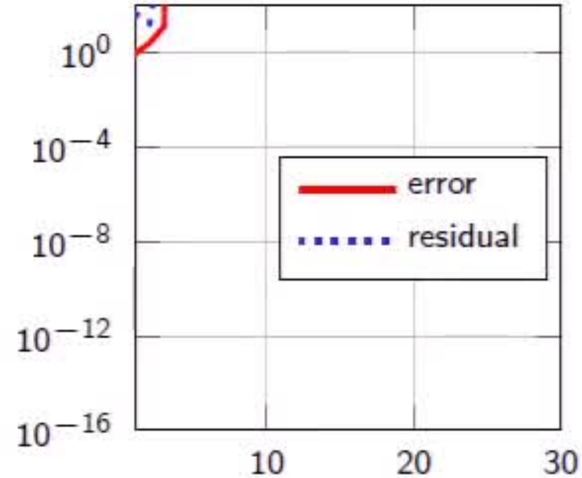
Spectrum and convergence - both regions



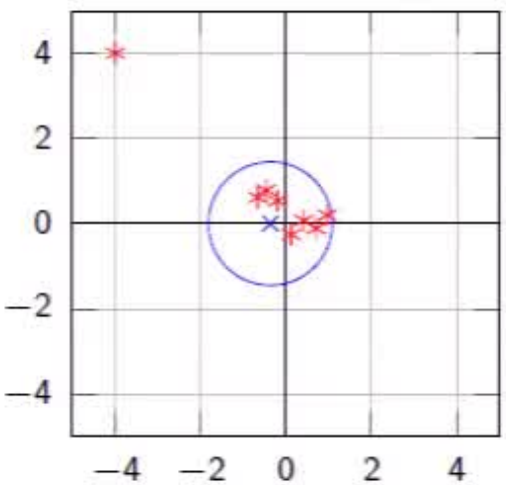
(m) $\sigma(A)$



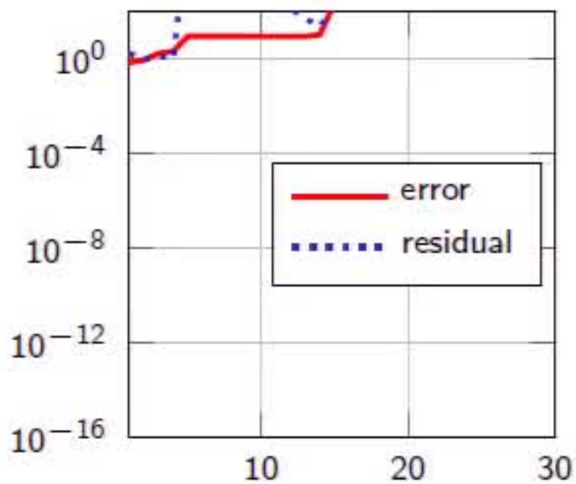
(n) $Y_0 = \varphi(A)$



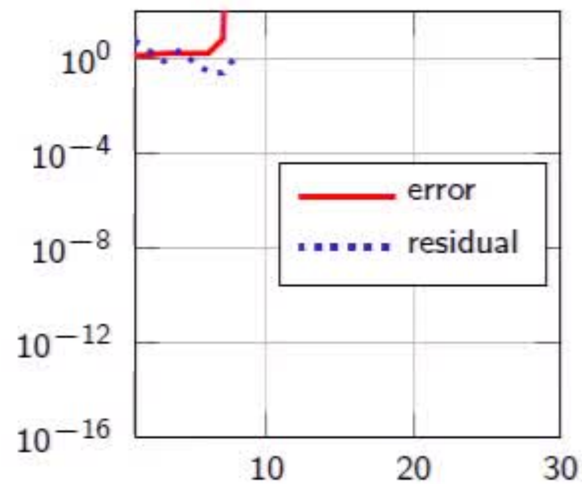
(o) $Y_0 = \psi(A)$



(p) $\sigma(A)$



(q) $Y_0 = \varphi(A)$



(r) $Y_0 = \psi(A)$

A Newton based Schur–Parlett algorithm

Approach to compute $W_b(A)$ when $|b| < 2$

- split matrix into two blocks
- order eigenvalues by region
- apply simplified iteration twice
- compute matrix

Advantages

- accuracy for ill-conditioned eigenvectors
- possibility of using both series expansions for a same matrix
- possibility of clustering eigenvalues across a branch cut

Eigenvectors conditioning for W_0

Methods

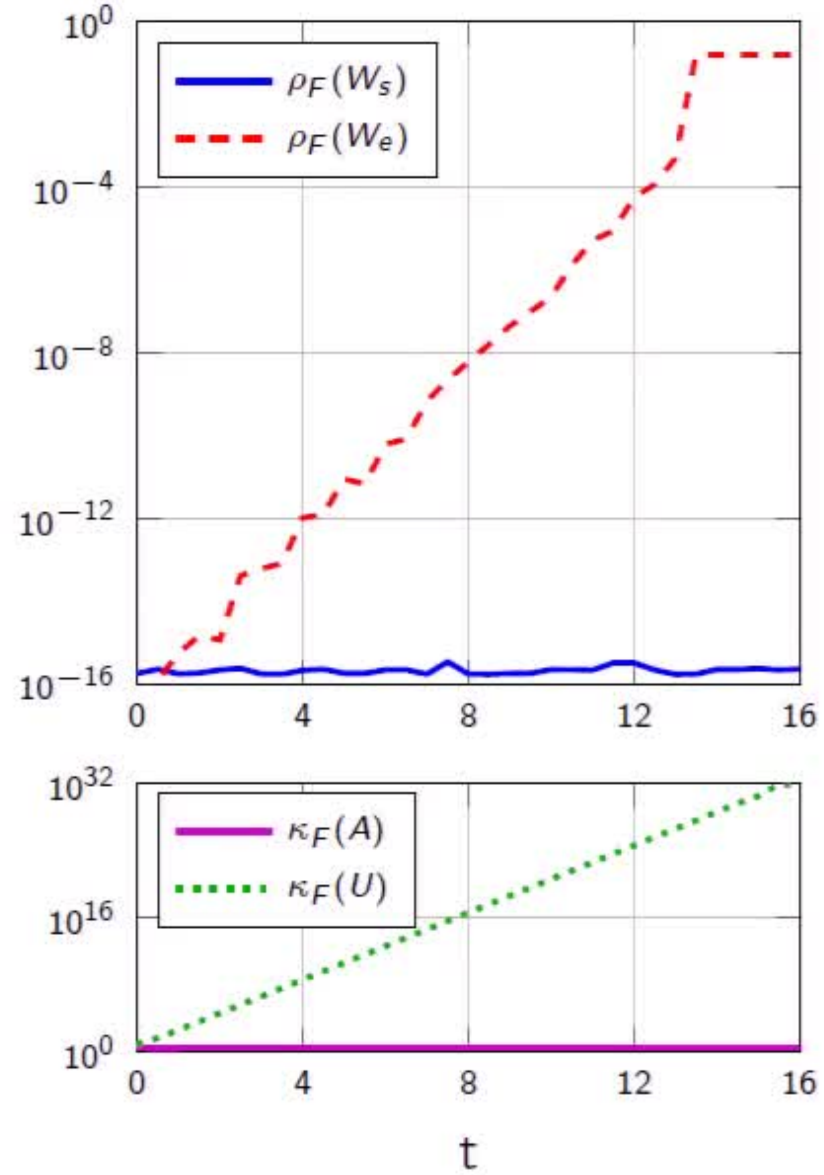
- $\mathbf{W_s}$: Newton–Schur–Parlett
- $\mathbf{W_e}$: diagonalization

Residual

$$\rho_F(W) = \frac{\|W\exp(W) - A\|_F}{\|W\exp(W)\|_F + \|A\|_F}$$

Matrix

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 + 10^{-t} \end{bmatrix} = UDU^{-1}$$



Residual for W_0

	$\kappa_F(A)$	$\kappa_F(U)$	$\rho_F(W_e)$	$\rho_F(W_s)$
gcdmat	1.09e+02	1.00e+00	**4.87e-16**	2.05e-15
minij	1.75e+02	1.00e+00	3.86e-16	**3.79e-16**
pascal	4.16e+09	1.00e+00	1.81e-15	**9.63e-16**
cauchy	6.23e+13	1.00e+00	3.59e-16	**3.12e-16**
lotkin	2.77e+13	1.79e+01	**6.12e-16**	1.77e-15
riemann	6.28e+03	2.26e+01	**7.87e-16**	1.68e-15
dramadah	5.05e+02	7.22e+01	**2.28e-15**	6.98e-15
lesp	6.69e+00	4.41e+02	1.14e-14	**5.36e-15**
kahan	2.99e+01	4.74e+04	1.58e-13	**4.87e-16**
frank	2.85e+07	7.16e+05	4.23e-11	**4.47e-14**
forsythe	6.71e+07	1.11e+07	5.84e-11	**4.63e-14**
redheff	4.68e+01	1.17e+09	2.37e-08	**9.19e-16**

Relative residual for diagonalization and Newton–Schur–Parlett