

## LOPSIDED APPROXIMATION OF AMOEBAS

JENS FORSGÅRD, LAURA FELICIA MATUSEVICH, NATHAN MEHLHOP,  
AND TIMO DE WOLFF

**ABSTRACT.** The amoeba of a Laurent polynomial is the image of the corresponding hypersurface under the coordinatewise log absolute value map. In this article, we demonstrate that a theoretical amoeba approximation method due to Purbhoo can be used efficiently in practice. To do this, we resolve the main bottleneck in Purbhoo’s method by exploiting relations between cyclic resultants. We use the same approach to give an approximation of the Log preimage of the amoeba of a Laurent polynomial using semi-algebraic sets. We also provide a SINGULAR/SAGE implementation of these algorithms, which shows a significant speedup when our specialized cyclic resultant computation is used, versus a general purpose resultant algorithm.

### 1. INTRODUCTION

Consider a *Laurent polynomial*

$$(1.1) \quad f(\mathbf{z}) = \sum_{j=1}^d b_j \mathbf{z}^{\alpha(j)},$$

where  $\mathbf{z} := (z_1, \dots, z_n)$ . We denote by  $\text{Var}(f)$  the hypersurface defined by  $f$  in the maximal open torus  $(\mathbb{C}^*)^n := (\mathbb{C} \setminus \{0\})^n$  of  $\mathbb{C}^n$ .

**Definition 1.1.** The *log absolute value map* is given by

$$(1.2) \quad \text{Log } |\cdot|: (\mathbb{C}^*)^n \rightarrow \mathbb{R}^n, \quad (z_1, \dots, z_n) \mapsto (\log |z_1|, \dots, \log |z_n|).$$

The *amoeba*  $\mathcal{A}(f)$  of  $f$  is defined as  $\text{Log } |\text{Var}(f)|$ . The *unlog amoeba*  $\mathcal{U}(f)$  of  $f$  is defined as  $|\text{Var}(f)|$ , where

$$|\cdot|: (\mathbb{C}^*)^n \rightarrow \mathbb{R}^n, \quad (z_1, \dots, z_n) \mapsto (|z_1|, \dots, |z_n|). \quad \square$$

Gelfand, Kapranov, and Zelevinsky [GKZ94, Definition 6.1.4] introduced amoebas in the context of toric geometry. Since then, amoebas have been used in different areas such as complex analysis [FPT00, PR04], real algebraic curves [Mik00], statistical thermodynamics [PPT13], and nonnegativity of real polynomials [IdW16]. Overviews on amoeba theory include [dW17, Mik04, PT05].

The usefulness of amoebas motivates the problem of finding an efficient algorithm for computing them. Since  $\text{Log } |\cdot|$  is a non-algebraic map,  $\mathcal{A}(f)$  is not a semi-algebraic set in general. The amoeba  $\mathcal{A}(f)$  is, however, a semi-analytic set. The

---

Received by the editor April 28, 2017, and, in revised form, August 10, 2017.

2010 *Mathematics Subject Classification.* Primary 13P15, 14Q20, 14T05; Secondary 90C59, 90C90.

*Key words and phrases.* Amoeba, amoeba computation, cyclic resultant, lopsided amoeba, resultant, SAGE, SINGULAR.

The second author was partially supported by NSF Grant DMS 1500832.

absolute value map, on the other hand, is a real algebraic map. Therefore, the unlog amoeba  $\mathcal{U}(f)$  is a real semi-algebraic set. Hence, the ideal solution to the problem of amoeba computation can be described as follows.

Given a Laurent polynomial  $f \in \mathbb{C}[\mathbf{z}^{\pm 1}]$ , efficiently compute a real semi-algebraic description of the unlog amoeba  $\mathcal{U}(f)$ .

Theobald was the first to tackle computational aspects of amoebas in [The02]. He described, in the case  $n = 2$ , methods to compute a superset of the boundary of  $\mathcal{A}(f)$  called the *contour* of  $\mathcal{A}(f)$ . Later, his method was extended by Schroeter and the fourth author in [SdW13], who provide a method to test whether a contour point of  $\mathcal{A}(f)$  belongs to the boundary of  $\mathcal{A}(f)$  in the case when the hypersurface defined by  $f$  is smooth.

An approach, different than Theobald's, for computing amoebas arises from the following problem.

**Problem 1.2** (The membership problem). Let  $f \in \mathbb{C}[\mathbf{z}^{\pm 1}]$  and  $|\mathbf{v}| \in \mathbb{R}^n$ . Provide a certificate  $C$  such that if  $C(|\mathbf{v}|)$  is true, then  $\text{Log}|\mathbf{v}| \notin \mathcal{A}(f)$ .

The membership problem was addressed by Purbhoo in [Pur08], using the notion of lopsidedness; see Definition 1.3. A second approach for certifying non-membership of a point in  $\mathcal{A}(f)$  was provided by Theobald and the fourth author in [TdW15] using semi-definite programming and sums of squares. A rough but quick method to approximate amoebas was given by Avendaño, Kogan, Nisse, and Rojas [AKNR18] via tropical geometry. In [BKS16], Bogdanov, Kytmanov and Sadykov computed amoebas in dimensions two and three, with focus on amoebas with the most complicated topology possible. We remark that even for univariate polynomials deciding the membership problem is NP-hard [AKNR18, Section 1.2].

The main aim of this article is to make the results in [Pur08] effective and efficient in practice.

**Definition 1.3** ([Rul03, p. 24] and [Pur08, Definition 1.2]). Let  $f$  be as in (1.1). We say that  $f$  is *lopsided* at a point  $\text{Log}|\mathbf{v}| \in \mathbb{R}^n$  if there exists an index  $k \in \{1, \dots, d\}$  such that

$$(1.3) \quad \left| b_k \mathbf{v}^{\alpha(k)} \right| > \sum_{j \in \{1, \dots, d\} \setminus \{k\}} \left| b_j \mathbf{z}^{\alpha(j)} \right|.$$

The *lopsided amoeba* of  $f$  is defined by

$$\mathcal{L}(f) := \{ \text{Log}|\mathbf{v}| \in \mathbb{R}^n \mid f \text{ is not lopsided at } \text{Log}|\mathbf{v}| \}. \quad \square$$

It is not hard to see that  $\mathcal{A}(f) \subseteq \mathcal{L}(f)$ ; the special case  $n = 1$  of this result was proven in the nineteenth century, in an equivalent formulation, by Pellet [Pel81]. We remark that, in general,  $\mathcal{A}(f) \neq \mathcal{L}(f)$ .

Following Purbhoo, we introduce

$$(1.4) \quad \begin{aligned} \text{CycRes}(f; r) &:= \prod_{k_1=0}^{r-1} \cdots \prod_{k_n=0}^{r-1} f \left( e^{2\pi i k_1/r} z_1, \dots, e^{2\pi i k_n/r} z_n \right) \\ &= \text{Res}_{u_n} \left( \text{Res}_{u_{n-1}} \left( \dots \text{Res}_{u_1} \left( f(u_1 z_1, \dots, u_n z_n), \right. \right. \right. \\ &\quad \left. \left. \left. u_1^r - 1, \dots, u_{n-1}^r - 1 \right), u_n^r - 1 \right), \right. \end{aligned}$$

where  $\text{Res}_x(g, h)$  denotes the resultant of the polynomials  $g$  and  $h$  with respect to the variable  $x$ . If  $g = g(x)$  is a univariate polynomial, then  $\text{Res}_x(g, x^r - 1)$  is called a *cyclic resultant* of  $g$ .

Note that  $\text{CycRes}(f; r)$  is a Laurent polynomial in  $z_1, \dots, z_n$ . The informal version of [Pur08, Theorem 1] states that the limit as  $r \rightarrow \infty$  of  $\mathcal{L}(\text{CycRes}(f; r))$  is  $\mathcal{A}(f)$ ; see Section 2.2 for more information.

The main obstacle in turning Purbhoo's result into an efficient approximation method for amoebas is the difficulty in computing the polynomials  $\text{CycRes}(f; r)$ . The degree and the number of terms of  $\text{CycRes}(f; r)$  grow exponentially with  $r$ , but more importantly, the methods used by computer algebra systems to find resultants fail to take advantage of the sparseness of the polynomials  $u_i^r - 1$ , and are therefore manifestly inefficient when applied to  $\text{CycRes}(f; r)$ .

Our main results are as follows.

- (1) We give a fast method to compute the cyclic resultant  $\text{CycRes}(f; 2r)$  from  $\text{CycRes}(f; r)$  omitting all intermediate steps  $\text{CycRes}(f; r + 1), \dots, \text{CycRes}(f; 2r - 1)$ ; see Section 3 for details. We provide an experimental comparison of the runtimes using these quick resultants versus a general purpose resultant algorithm in Table 2. We also give the complexity of our specialized algorithm for computing (certain) cyclic resultants, verifying our experimental conclusion that this method is significantly faster than the best resultant algorithm for real polynomials (Remark 3.5).
- (2) We provide an algorithm to approximate the unlog amoeba by semi-algebraic sets; see Theorem 6.1 and Algorithm 6.2.

Purbhoo's Amoeba Approximation Theorem is the main ingredient in the proof of Theorem 6.1 and in Algorithm 6.2. Using toric geometry, we also show that there exists a natural correspondence between the boundary components of lopsided amoebas and the boundary components of linear amoebas, where the latter are well understood due to Forsberg, Passare, and Tsikh [FPT00]; see Section 5 for further details.

Lemma 3.3 allows us to compute cyclic resultants for powers  $r = 2^k$  using a divide and conquer algorithm. Note that several prominent algorithms are built on a similar approach, for example, the Cooley–Tukey algorithm for the Fast Fourier Transformation [CT65].

As a companion to this work we provide the first implementation of lopsided approximation of amoebas, available here:

<http://www.math.tamu.edu/research/dewolff/LopsidedAmoebaApproximation>.

We also provide the data presented in this article on this website. The algorithms in this article are implemented in the computer algebra system SINGULAR [DGPS15] and scripts to provide graphical outputs use the computer algebra system SAGE [Dev16].

**Outline.** This paper is organized as follows. Section 2 contains background on amoebas and lopsided amoebas. Section 3 outlines a fast way of computing certain cyclic resultants. In Section 4 we describe how the algorithm from Section 3 can be used to approximate amoebas. Section 5 provides a geometric interpretation of the lopsided amoeba as the intersection of the amoeba of a toric variety and the amoeba of a hyperplane. In Section 6 we describe how to use results from Sections 3 and

5 to compute semi-algebraic sets approximating the amoeba. Finally, Section 7 is devoted to examples.

## 2. PRELIMINARIES

In this section, we review the theory of amoebas and lopsided amoebas which is used in this work.

Recall that, throughout this article,  $f \in \mathbb{C}[\mathbf{z}^{\pm 1}]$  denotes the Laurent polynomial (1.1). We denote by  $A$  the *support* of  $f$ , namely,

$$A := \{\alpha \in \mathbb{Z}^n \mid \mathbf{z}^\alpha \text{ appears with non-zero coefficient in } f\}.$$

The *Newton polytope* of  $f$ , denoted  $\text{New}(f)$ , is the convex hull in  $\mathbb{R}^n$  of the support set  $A$  of  $f$ .

**2.1. Amoebas.** The *complement* of the amoeba  $\mathcal{A}(f)$  is the set  $\mathbb{R}^n \setminus \mathcal{A}(f)$ . The connected components of  $\mathbb{R}^n \setminus \mathcal{A}(f)$  are referred to as the *components of the complement* of  $\mathcal{A}(f)$ .

**Theorem 2.1** ([GKZ94, Chapter 6.1.A, Proposition 1.5, Corollary 1.8]). *For a nonzero Laurent polynomial  $f$ , the complement  $\mathbb{R}^n \setminus \mathcal{A}(f)$  is non-empty. Every component of  $\mathbb{R}^n \setminus \mathcal{A}(f)$  is convex and open (with respect to the standard topology).*

Forsberg, Passare, and Tsikh [FPT00] showed that every component of the complement of an amoeba  $\mathcal{A}(f)$  can be associated with a specific lattice point in the Newton polytope  $\text{New}(f)$  via the *order map*:

(2.1)

$\text{ord}: \mathbb{R}^n \setminus \mathcal{A}(f) \rightarrow \mathbb{R}^n, \quad \mathbf{w} \mapsto (u_1, \dots, u_n), \quad \text{where}$

$$u_j := \frac{1}{(2\pi\sqrt{-1})^n} \int_{\text{Log}|\mathbf{z}|=\mathbf{w}} \frac{z_j \partial_j f(\mathbf{z})}{f(\mathbf{z})} \frac{dz_1 \cdots dz_n}{z_1 \cdots z_n} \quad \text{for all } 1 \leq j \leq n.$$

**Theorem 2.2** ([FPT00, Propositions 2.4 and 2.5]). *The image of the order map is contained in  $\text{New}(f) \cap \mathbb{Z}^n$ . Let  $\mathbf{w}, \mathbf{w}' \in \mathbb{R}^n \setminus \mathcal{A}(f)$ . Then  $\mathbf{w}$  and  $\mathbf{w}'$  belong to the same component of the complement of  $\mathcal{A}(f)$  if and only if  $\text{ord}(\mathbf{w}) = \text{ord}(\mathbf{w}')$ .*

As a consequence of Theorem 2.2, it is possible to define *the component of order  $\alpha$  of the complement of  $\mathcal{A}(f)$* . We use the following notation:

$$(2.2) \quad E_\alpha(f) := \{\mathbf{w} \in \mathbb{R}^n \setminus \mathcal{A}(f) \mid \text{ord}(\mathbf{w}) = \alpha\}.$$

**2.2. Lopsided amoebas.** We now give a more precise statement of the main result in [Pur08], which was alluded to in the introduction.

**Theorem 2.3** ([Pur08, Theorem 1 “rough version”]). *For  $r \rightarrow \infty$  the family  $\mathcal{L}(\text{CycRes}(f; r))$  converges uniformly to  $\mathcal{A}(f)$ . More precisely, for every  $\varepsilon > 0$  there exists an integer  $N$  such that for all  $r > N$  the lopsided amoeba  $\mathcal{L}(\text{CycRes}(f; r))$  is contained in an  $\varepsilon$ -neighborhood of  $\mathcal{A}(f)$ .*

*Remark 2.4.* The integer  $N$  in Theorem 2.3 depends only on  $\varepsilon$  and the Newton polytope (or degree) of  $f$ , and can be computed explicitly from this data. For simplicity, assume that  $f$  is a polynomial of degree  $d$ . Let  $\text{Log}|\mathbf{v}| \in \mathbb{R}^n \setminus \mathcal{A}(f)$  be

of distance at least  $\varepsilon$  from the amoeba  $\mathcal{A}(f)$ . Then, by [Pur08, Theorem 1], if  $r$  is chosen such that

$$r\varepsilon \geq (n-1)\log r + \log((n+3)2^{n+1}d),$$

then it holds that  $\text{CycRes}(f; r)$  is lopsided at the point  $\text{Log}|\mathbf{v}|$ . We will typically be interested in the case  $r = 2^k$ . Then, it suffices that

$$\frac{2^k}{k} \geq \frac{C(n, d)}{\varepsilon},$$

where  $C(n, d) := (n-1)\log 2 + \log((n+3)2^{n+1}d)$  is constant.

If  $f$  is lopsided at a point  $\text{Log}|\mathbf{v}| \in \mathbb{R}^n$ , then it is straightforward to determine the order of the component of the complement of  $\mathcal{A}(f)$  which contains  $\text{Log}|\mathbf{v}|$ .

**Theorem 2.5** ([FPT00, Proposition 2.7] and [Pur08, Proposition 4.1]). *Let  $A \subset \mathbb{Z}^n$  and let  $f(\mathbf{z})$  be as in (1.1). Assume that  $f(\mathbf{z})$  is lopsided at  $\text{Log}|\mathbf{v}|$  with dominating term  $|b_{\alpha'} \mathbf{z}^{\alpha'}|$ . Then  $\text{ord}(\text{Log}|\mathbf{v}|) = \alpha'$ . In particular, if  $\text{CycRes}(f, r)$  is lopsided at  $\text{Log}|\mathbf{v}|$  with dominating term  $|b_{\alpha'} \mathbf{z}^{\alpha'}|$ , then  $\text{ord}(\text{Log}|\mathbf{v}|) \cdot r^n = \alpha'$ .*

The following useful criterion for lopsidedness is a consequence of the triangle inequality.

**Lemma 2.6.** *The Laurent polynomial  $f(\mathbf{z})$  is not lopsided at  $\text{Log}|\mathbf{z}|$  if and only if there exist arguments  $\theta \in (S^1)^d$  such that  $\sum_{j=1}^d b_j e^{\theta_j \sqrt{-1}} \mathbf{z}^{\alpha(j)} = 0$ .*

### 3. A FAST ALGORITHM TO COMPUTE CYCLIC RESULTANTS

In this section, we provide a fast way of computing certain cyclic resultants.

We recall the *Poisson formula* for the resultant. If  $f(z)$  and  $g(z)$  are univariate polynomials, and  $g$  has leading coefficient  $b$ , then

$$(3.1) \quad \text{Res}_z(f, g) = b^{\deg(f)} \prod_{\{\xi \mid g(\xi)=0\}} f(\xi).$$

**Lemma 3.1.** *Let  $g \in \mathbb{C}[z]$  be a univariate polynomial of degree  $\delta$  with leading coefficient  $b$ , and let  $r$  be a positive integer. Then there exists a univariate polynomial  $h$  of degree  $\delta$  such that*

$$\text{Res}_u(g(zu), u^r - 1) = h(z^r).$$

*Proof.* By (3.1),

$$\text{Res}_u(g(zu), u^r - 1) = \prod_{j=0}^{r-1} g(e^{2\pi\sqrt{-1}j/r} z).$$

Writing  $g(z) = b \prod_{\ell=1}^{\delta} (z - \xi_{\ell})$ , we have

$$\begin{aligned} \text{Res}_u(g(zu), u^r - 1) &= \prod_{j=0}^{r-1} b \prod_{\ell=1}^{\delta} (e^{2\pi\sqrt{-1}j/r} z - \xi_{\ell}) \\ &= b^r \prod_{\ell=1}^{\delta} \prod_{j=0}^{r-1} (e^{2\pi\sqrt{-1}j/r} z - \xi_{\ell}) \\ &= b^r \prod_{\ell=1}^{\delta} (z^r - \xi_{\ell}^r). \end{aligned}$$

Now use  $h(z) = b^r \prod_{\ell=1}^{\delta} (z - \xi_{\ell}^r)$ . □

**Lemma 3.2.** *Let  $h(z^{2^k}) = \text{Res}_u(g(zu), u^{2^k} - 1)$  be as in Lemma 3.1. Then*

$$\text{Res}_u(g(zu), u^{2^k} + 1) = h(-z^{2^k}).$$

*Proof.* Note that the roots of  $u^{2^k} + 1$  are exactly the roots of  $u^{2^k} - 1$  multiplied by  $e^{2\pi\sqrt{-1}/(2^{k+1})}$ . Therefore, by (3.1),

$$\begin{aligned} \text{Res}_u(g(zu), u^{2^k} + 1) &= \prod_{j=0}^{2^k-1} g\left(e^{2\pi\sqrt{-1}j/2^k} e^{2\pi\sqrt{-1}/2^{k+1}} z\right) \\ &= h\left((e^{2\pi\sqrt{-1}/2^{k+1}} z)^{2^k}\right) \\ &= h(-z^{2^k}). \end{aligned} \quad \square$$

**Lemma 3.3.** *The following identity holds:*

$$\text{Res}_u(g(zu), u^{2^{k+1}} - 1) = \text{Res}_u(g(zu), u^{2^k} - 1) \cdot \text{Res}(g(zu), u^{2^k} + 1).$$

*Proof.* The equality follows from the Poisson formula, since  $u^{2^{k+1}} - 1 = (u^{2^k} - 1)(u^{2^k} + 1)$ , and these factors have disjoint sets of roots.  $\square$

#### Algorithm 3.4.

```

1: Input: a polynomial  $f \in \mathbb{Q}[\sqrt{-1}][\mathbf{x}]$  in  $n$  variables of degree  $d$  and  $k \in \mathbb{N}$ 
2: Output:  $\text{CycRes}(f; 2^k)$ 
3: /* Initialize a polynomial */
4:  $\text{CycResult} := f$ 
5: /* Different variables are handled subsequently */
6: for  $j = 1$  to  $n$  do
7:   for  $l = 1$  to  $k$  do
8:     /* Initialize multiplier polynomial */
9:      $\text{Multiplier} := \text{CycResult}$ 
10:    Change all signs of terms in multiplier, which have an exponent whose
     $j$ th entry is not divisible by  $2^l$ .
11:     $\text{CycResult} = \text{CycResult} \cdot \text{Multiplier}$ 
12:   end for
13: end for
14: return  $\text{CycResult}$ 

```

*Proof of correctness of Algorithm 3.4.* This follows from the definition of cyclic resultants (1.4), Lemma 3.2, and Lemma 3.3.  $\square$

*Complexity analysis of Algorithm 3.4.* In order to perform the complexity analysis we will impose the assumption that  $n = 1$ . This is not a severe restriction, as for a multivariate polynomial our algorithm iterates the univariate case, however, it greatly simplifies the computations.

We count complexity as the number of arithmetic operations performed in a field containing the coefficients of  $f$ . Here, we consider addition and multiplication to have the same cost. In the comparison with the signed resultant algorithm we need to assume that the coefficients are real; however, this does not affect our complexity analysis.

By Lemma 3.1 we have that, for each  $l$ , the polynomial `CycResult` has at most  $d+1$  terms. Multiplying the coefficient of every second term (i.e., the terms whose exponents are not divisible by  $2^l$ ) by  $-1$  requires  $\lfloor d/2 \rfloor$  arithmetic operations.

Multiplying two univariate (real) polynomials of degree  $d$  has complexity  $2d^2 + 2d + 1$ ; see [BPR03, Algorithm 8.3].

Since we perform this task  $k$  times, the total complexity is less than

$$k \left( \frac{d}{2} + 2d^2 + 2d + 1 \right) = k \left( 2d^2 + \frac{5d}{2} + 1 \right).$$

Thus, the complexity is  $O(kd^2)$ .  $\square$

*Remark 3.5.* Let  $f$  be a real, univariate polynomial. The fastest general resultant algorithm which the authors are aware of is the *signed subresultant algorithm*; see, e.g., [BPR03, Algorithm 8.73]). It computes the resultant of two univariate polynomials of degrees  $d_1$  and  $d_2$  with arithmetic complexity  $O(d_1 d_2)$ . In our situation  $d_1 = d$  and  $d_2 = 2^k$ ; hence the signed subresultant algorithm computes the cyclic resultant of order  $k$  with arithmetic complexity  $O(d2^k)$ .

The algorithm we propose is, hence, a vast improvement. In particular since, in the typical situation, the degree  $d$  is fixed while one varies the parameter  $k$  in order to obtain an improved approximation. Overall, it reduces the runtime from exponential to polynomial in one variable and from double to single exponential in arbitrary many variables.

We point out that Hillar and Levine [Hil05, HL07] have shown that the sequence of cyclic resultants  $\{\text{Res}_z(g(z), z^k - 1)\}_{k=1}^\infty$  satisfies a linear recurrence of order  $2^{\deg(g)}$ . While this recurrence can be used to compute *all* cyclic resultants, it is, in general, slower than our method for computing  $\text{Res}_u(g(zu), u^{2^k} - 1)$ , and the latter is sufficient for our purposes.

#### 4. APPROXIMATING AN AMOEBEA USING CYCLIC RESULTANTS

In this section, we describe how Algorithm 3.4 can be applied to approximate an amoeba of a given polynomial. There are two obstacles in obtaining an algorithm to approximate amoebas from the results of Section 3. First, we can only compute cyclic resultants of finite level  $k$ . That is, according to Theorem 2.3 and Remark 2.4, we can only test for membership in some  $\varepsilon$  neighborhood of the amoeba, where  $\varepsilon$  determines  $k$ . Second, it is not effective to test all points of  $\mathbb{R}^n$  for membership.

Neither of these obstacles are insurmountable. Our approach will be as follows. Let us fix some  $\varepsilon > 0$ , which determines the integer  $k$  in accordance with Remark 2.4. Also, we will construct a grid  $\mathcal{G} := [s, t]^n \cap (\ell \cdot \mathbb{Z})^n \subseteq \mathbb{Q}^n$  for some  $\ell \in \mathbb{Q}^n$ . Here,  $\ell$  should be chosen small enough so that any ball of radius  $\varepsilon$  inside  $[s, t]^n$  contains at least one point of  $\mathcal{G}$ . Then, testing the finitely many points of  $\mathcal{G}$  on the level  $k$ , we are assured to find points in every component of the complement of the amoeba whose intersection with  $[s, t]^n$  contains a ball of radius  $\varepsilon$ . Since the complement of the amoeba consists of a finite number of open sets, this ensures that we will find all components of the complement if  $\varepsilon$  is chosen sufficiently small. Though, we remark that there is currently no explicit expression describing how small  $\varepsilon$  should be chosen. In practice we will choose the grid  $\mathcal{G}$  rather than  $\varepsilon$ ; the latter can then be determined from the former.

**Algorithm 4.1.**

```

1: Input: a polynomial  $f \in \mathbb{Q}[\sqrt{-1}][\mathbf{x}]$ , start-/endpoints of grid  $s, t$ , steplength  $\ell$ 
   for grid,  $k \in \mathbb{N}$ 
2: Output: List of gridpoints together with a bit indicating membership in  $\varepsilon$ 
   neighborhood of  $\mathcal{A}(f)$  and the order of the component of the complement if
   applicable
3: /* Create grid */
4: Create grid  $\mathcal{G} := [s, t]^n \cap (\ell \cdot \mathbb{Z})^n$ .
5: /* Compute  $2^j$ th cyclic resultants for  $j = 1, \dots, k$  via
   Algorithm 3.4 */
6: for  $j = 1$  to  $k$  do
7:    $g_j := \text{CycResult}(f, 2^j)$ 
8: end for
9: /* Create empty list */
10:  $L := []$ 
11: /* Test gridpoints for lopsidedness */
12: for every  $\mathbf{p}$  in  $\mathcal{G}$  do
13:    $\text{IsLopsided} := \text{false}$ 
14:    $j := 1$ 
15:   while  $\text{IsLopsided} = \text{false}$  and  $j \leq k$  do
16:     if  $g_j$  is lopsided at  $\mathbf{p}$  then
17:        $\text{IsLopsided} := \text{true}$ 
18:     else
19:        $j := j + 1$ 
20:     end if
21:   end while
22:   if  $\text{IsLopsided} = \text{true}$  then
23:     /* Compute the order of  $f$  at  $\mathbf{p}$  via Theorem 2.5 */
24:      $\alpha :=$  exponent of dominating term of  $g_j$  at  $\mathbf{p}$  divided by  $2^j$ 
25:     Adjoin  $(\mathbf{p}, 0, \alpha)$  to  $L$ 
26:   else
27:     Adjoin  $(\mathbf{p}, 1, \text{NaN})$  to  $L$ 
28:   end if
29: end for
30: return  $L$ 

```

*Proof of Algorithm 4.1.* Follows from Algorithm 3.4 and Theorem 2.5.  $\square$

We remark that computing the orders allows to determine from the finite list  $L$  a set of polyhedra contained in the complement of the amoeba. By Theorem 2.1, the components of the complement of an amoeba are convex. Similarly, the recession cone of a component of the complement with order  $\alpha$  is equal to the normal cone of  $\alpha$  in the Newton polytope of  $f$ ; see [PT05].

## 5. GEOMETRY OF THE LOPSIDED AMOEBA

The main result of this section, Theorem 5.2, gives a geometric interpretation of the lopsided amoeba of a Laurent polynomial.



Let  $f(\mathbf{z})$  be as in (1.1), and denote by  $\mathbf{b}$  the coefficient vector  $(b_1, \dots, b_d)$ . We introduce the auxiliary  $2d$ -variate polynomial

$$F(\mathbf{w}, \mathbf{x}) := \sum_{j=1}^d w_j x_j,$$

which coincides with the polynomial  $f(\mathbf{z})$  when evaluated at  $\mathbf{w} = \mathbf{b}$  and  $x_j = \mathbf{z}^{\alpha(j)}$ . The polynomial  $F$  has amoeba  $\mathcal{A}(F) \subset \mathbb{R}^d \times \mathbb{R}^d$ . We denote by  $\text{pr}_2$  the projection of  $\mathbb{R}^d \times \mathbb{R}^d$  onto the second factor, and set  $F_{\mathbf{b}}(\mathbf{x}) := F(\mathbf{b}, \mathbf{x})$ .

For convenience, we denote the coordinates of  $\mathbb{R}^d \times \mathbb{R}^d$  by  $(\text{Log}|\mathbf{w}|, \text{Log}|\mathbf{x}|)$ .

**Lemma 5.1.** *Let  $H_{\mathbf{b}}$  be the  $d$ -dimensional affine space defined by  $\text{Log}|\mathbf{w}| = \text{Log}|\mathbf{b}|$ . Then  $\mathcal{A}(F_{\mathbf{b}}) = \text{pr}_2(\mathcal{A}(F) \cap H_{\mathbf{b}})$ .*

*Proof.* By Lemma 2.6,  $(\text{Log}|\mathbf{b}|, \text{Log}|\mathbf{x}|) \in \mathcal{A}(F)$  if and only if there exist arguments  $(\theta_{\mathbf{b}}, \theta_{\mathbf{x}}) \in (S^1)^d \times (S^1)^d$  such that  $F(\mathbf{b}e^{\sqrt{-1}\theta_{\mathbf{b}}}, \mathbf{x}e^{\sqrt{-1}\theta_{\mathbf{x}}}) = 0$ . We have

$$F(\mathbf{b}e^{\sqrt{-1}\theta_{\mathbf{b}}}, \mathbf{x}e^{\sqrt{-1}\theta_{\mathbf{x}}}) = F(\mathbf{b}, \mathbf{x}e^{\sqrt{-1}(\theta_{\mathbf{x}} + \theta_{\mathbf{b}})}) = F_{\mathbf{b}}(\mathbf{x}e^{\sqrt{-1}(\theta_{\mathbf{x}} + \theta_{\mathbf{b}})}),$$

and the result follows.  $\square$

Recall that the *affine toric variety*  $X_A$  associated to  $A = \{\alpha(1), \dots, \alpha(d)\} \subset \mathbb{Z}^n$ , is the Zariski closure in  $\mathbb{C}^d$  of the image of the monomial map  $(\mathbb{C}^*)^n \rightarrow (\mathbb{C}^*)^d$  given by  $\mathbf{z} \mapsto (\mathbf{z}^{\alpha(1)}, \dots, \mathbf{z}^{\alpha(d)})$ .

The amoeba  $\mathcal{A}(X_A)$  of the toric variety  $X_A$  is a linear subvariety of  $\mathbb{R}^d$ , parameterized by  $\text{Log}|\mathbf{z}| \mapsto (\alpha(1) \cdot \text{Log}|\mathbf{z}|, \dots, \alpha(d) \cdot \text{Log}|\mathbf{z}|)$ . We denote by  $s_A: \mathcal{A}(X_A) \rightarrow \mathbb{R}^n$  the inverse of this mapping.

We embed  $\mathcal{A}(X_A)$  in the second factor  $\mathbb{R}^d \times \mathbb{R}^d$ , in order to be able to compare this set to the amoeba  $\mathcal{A}(F)$ . Let  $\overline{\text{pr}}_A = s_A \circ \text{pr}_2$ . Note that, given  $\mathbf{b}$ , the induced map

$$\overline{\text{pr}}_A: H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A)) \rightarrow \mathbb{R}^n$$

is an affine isomorphism.

**Theorem 5.2.** *We have that  $\mathcal{L}(f) = \overline{\text{pr}}_A(\mathcal{A}(F) \cap H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A)))$ .*

More precisely, if we identify  $\mathbb{R}^n$  with  $H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A)) \subset \mathbb{R}^d \times \mathbb{R}^d$  by the map  $\overline{\text{pr}}_A$ , then  $\mathcal{L}(f)$  is identified with  $\mathcal{A}(F)$  (or rather,  $\mathcal{L}(f)$  is identified with the part of  $\mathcal{A}(F)$  which is contained in  $H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A))$ ).

*Proof.* Let  $\mathbf{b}$  be fixed. Since  $\overline{\text{pr}}_A: H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A)) \rightarrow \mathbb{R}^n$  is an affine isomorphism, it has an inverse  $\overline{\text{pr}}_A^{-1}$  given by

$$\begin{aligned} \overline{\text{pr}}_A^{-1}: \mathbb{R}^n &\rightarrow H_{\mathbf{b}} \cap (\mathbb{R}^d \times \mathcal{A}(X_A)), \\ \text{Log}|\mathbf{z}| &\mapsto (\text{Log}|\mathbf{b}|, \alpha(1) \cdot \text{Log}|\mathbf{z}|, \dots, \alpha(d) \cdot \text{Log}|\mathbf{z}|). \end{aligned}$$

By Lemma 2.6,  $f$  is not lopsided at  $\text{Log}|\mathbf{z}|$  if and only if there exist angles  $\theta_{\mathbf{b}} \in (S^1)^d$  such that

$$\sum_{j=1}^d b_j e^{\theta_{b_j} \sqrt{-1}} \mathbf{z}^{\alpha(j)} = 0,$$

which is equivalent to  $F(\mathbf{b}e^{\theta_{\mathbf{b}} \sqrt{-1}}, \mathbf{z}) = 0$ . But, the latter is equivalent to

$$\overline{\text{pr}}_A^{-1}(\text{Log}(\mathbf{z})) \in \mathcal{A}(F),$$

and the proof is complete.  $\square$

## 6. APPROXIMATING UNLOG AMOEBAS BY SEMI-ALGEBRAIC SETS

Recall that the unlog amoeba  $\mathcal{U}(f)$  is a semi-algebraic set. In this section we give an algorithm which provides an approximation of the unlog amoeba  $\mathcal{U}(f)$  as a sequence of semi-algebraic sets.

The *boundary* of the amoeba  $\mathcal{A}(f)$  is denoted by  $\partial\mathcal{A}(f)$ ; similarly, the boundary of a lopsided amoeba  $\mathcal{L}(f)$  by is denoted by  $\partial\mathcal{L}(f)$ . Our computation is based on the following statement, which uses the notation introduced in (2.2).

**Theorem 6.1.** *Let  $f(\mathbf{z})$  be as in (1.1) and let  $\text{CycRes}(f; r) = \sum_{\beta \in A(r)} b_{r,\beta} \mathbf{z}^\beta$ . Then, the boundary  $\partial\mathcal{L}(\text{CycRes}(f; r))$  converges to the boundary  $\partial\mathcal{A}(f)$  as  $r \rightarrow \infty$ , in the same sense as in Theorem 2.3. In particular, if  $\mathbb{R}^n \setminus \mathcal{A}(f) = \bigcup_{j=1}^k E_{\alpha(j)}(f)$  and if  $r$  is sufficiently large, then the semi-algebraic set  $\mathcal{U}(f)$  belongs to an  $\varepsilon$ -neighborhood of the semi-algebraic set*

$$(6.1) \quad \begin{array}{ccc} \sum_{\beta \in A(r) \setminus \{r^n \cdot \alpha(1)\}} |b_{r,\beta}| \mathbf{x}^\beta & \geq & |b_{r,r^n \cdot \alpha(1)}| \mathbf{x}^{r^n \cdot \alpha(1)}, \\ & \vdots & \\ \sum_{\beta \in A(r) \setminus \{r^n \cdot \alpha(k)\}} |b_{r,\beta}| \mathbf{x}^\beta & \geq & |b_{r,r^n \cdot \alpha(k)}| \mathbf{x}^{r^n \cdot \alpha(k)}, \\ x_1, \dots, x_n & \geq & 0. \end{array}$$

*Proof.* Since each connected component of the complement of the set defined by (6.1) is contained in the complement of  $\mathcal{U}(f)$ , we have that the Hausdorff distance of  $\partial\mathcal{L}(\text{CycRes}(f; r))$  and  $\partial\mathcal{A}(f)$  is dominated by the Hausdorff distance of  $\mathcal{L}(\text{CycRes}(f; r))$  and  $\mathcal{A}(f)$ . In particular, the statement regarding convergence of the boundaries follows immediately from Theorem 2.3. Using Theorem 5.2, we see that a component of the complement of  $\mathcal{L}(\text{CycRes}(f; r))$  of order  $\alpha'$  is given by  $\text{Log}|\cdot|$  image of the semi-algebraic set

$$(6.2) \quad \begin{array}{ccc} |b_{r,\alpha'}| \mathbf{x}^{\alpha'} & > & \sum_{\beta \in A(r) \setminus \{\alpha'\}} |b_{r,\beta}| \mathbf{x}^\beta, \\ x_1, \dots, x_n & \geq & 0. \end{array}$$

By Theorem 2.3 we know that for all sufficiently large  $r$  there exists a bijection between the set of all components of the complement of  $\mathcal{A}(f)$  and the set of all components of the complement of  $\mathcal{L}(\text{CycRes}(f; r))$ . By Theorem 2.5 it follows that if a component of the complement of  $\mathcal{A}(f)$  has order  $\alpha'$ , then the corresponding component of the complement of  $\mathcal{L}(\text{CycRes}(f; r))$  has order  $\alpha' \cdot r^n$ . This and (6.2) complete the proof.  $\square$

This theorem allows us to compute a sequence of semi-algebraic sets approximating the unlog amoeba  $\mathcal{U}(f)$  as follows.

**Algorithm 6.2.**

- 1: **Input:**  $f \in \mathbb{Q}[\sqrt{-1}][\mathbf{x}]$ ,  $\varepsilon > 0$
- 2: **Output:** An approximation of  $\mathcal{U}(f)$  by a semi-algebraic set.
- 3: Choose  $r \in \mathbb{N}$  sufficiently large with respect to  $\varepsilon$  according to Remark 2.4.
- 4: **/\* Compute all potential images of  $\text{ord}(\mathcal{A}(f))$  \*/**
- 5:  $M := \text{New}(f) \cap \mathbb{Z}^n$
- 6: **/\* Define a zero polynomial \*/**
- 7:  $g := \mathbf{0}$

```

8:  /* Construct  $g$  as sum of abs. values of terms of  $\text{CycRes}(f;r)$  */
9:  for  $b_\alpha \mathbf{x}^\alpha$  term in  $\text{CycRes}(f;r)$  do
10:     $g := g + |b_\alpha| \mathbf{x}^\alpha$ 
11:  end for
12:  /* Construct inequalities for semi-alg. description of  $\mathcal{U}(f)$  */
13:  for  $\alpha/r^n \in M$  do
14:     $g(\alpha) := g - 2 \cdot |b_\alpha| \mathbf{x}^\alpha$ 
15:  end for
16:  /* Construct semi-alg. description of  $\mathcal{U}(f)$  */
17:   $L := \bigcup_{j=1}^n \{x_j \geq 0\} \cup \bigcup_{\alpha/r^n \in M} \{g(\alpha)\}$ 
18:  return  $L$ 

```

*Proof of correctness of Algorithm 6.2.* This follows from Theorem 2.3 and Remark 2.4.  $\square$

Note that it is sufficient to consider the image of the order map of  $\mathcal{A}(f)$  as the set  $M$  in Algorithm 6.2, if the image of the order map is known. For example, if  $\text{supp}(f)$  is a *circuit*, that is, a minimal affinely dependent set, then it is well-known that the image of the order map is at most  $\text{supp}(f)$ ; see [Rul03, Theorem 12, p. 36].

## 7. EXPERIMENTAL COMPARISONS AND COMPUTATIONS

In this section we show experimentally that our fast method for computing cyclic resultants allows us to efficiently tackle the membership problem, as well as approximate an unlog amoeba using semi-algebraic sets. We compare this approach to a general purpose resultant algorithm: the comparison of the runtimes of both methods is given in Table 2.

**7.1. Outline, input, and assumptions.** As we mentioned in the Introduction, the two methods we use in this article to compute the amoeba of a Laurent polynomial are:

- (1) an approximation based on solving the Membership Problem 1.2, and
- (2) an approximation given by semi-algebraic sets converging to the unlog amoeba  $\mathcal{U}(f)$ .

We use the computer algebra system SINGULAR [DGPS15] for the computation of resultants and for the lopsidedness test. We produce pictures by transferring our SINGULAR output to the computer algebra system SAGE [Dev16].

For technical reasons we encode, in SINGULAR, coefficients over a rational ring with  $n$  variables which, in order to mimic complex numbers, has a parameter  $I$  with a minimal polynomial  $I^2 + 1$ .

When computing  $\text{CycRes}(f; 2^k)$ , we refer to the number  $k$  as the *level* of the resultant. In theory, our SINGULAR code can be used to compute cyclic resultants of an arbitrary level, and also to test lopsidedness of a single point; see Sections 7.2 and 7.3. In order to produce pictures, we need to restrict our computations to a grid as described in Section 4. Depending on the approach used, we handle this issue differently; see Sections 7.3 and 7.4.

**7.2. The computation of cyclic resultants.** In order to compute amoebas via membership tests or via a sequence of semi-algebraic sets converging to the unlog amoeba  $\mathcal{U}(f)$  for a given Laurent polynomial  $f$  we need to compute the  $r$ th cyclic resultant  $\text{CycRes}(f;r)$ . The larger we choose  $r \in \mathbb{N}^*$  the more accurate we expect

TABLE 1. The size of  $\text{CycRes}(z_1^3 + z_1z_2 + z_2^3 + 1; r)$  with  $r = 2^k$ .

$r$	2	4	8	16	32	64
number of terms	10	31	109	409	1585	6241
degree	12	48	192	786	3072	12288
coefficient magnitude	9	860	$> 10^{12}$	$> 10^{51}$	$10^{204}$	$10^{811}$

TABLE 2. A comparison of runtimes for the computation of cyclic resultants.

Polynomial	Level	Runs	Quick resultant	Iterated resultant	Factor
$f_1$	3	100	0.0062	0.0263	4.24
$f_1$	4	100	0.0520	1.7120	32.92
$f_1$	5	5	8.0400	221.6040	27.56
$f_1$	6	1	28.1400	118397.8600	4207.46
$f_2$	3	100	0.0354	0.6680	18.87
$f_2$	4	100	0.3952	88.9828	301.42
$f_2$	5	3	5.0433	67927.7433	13468.91
$f_3$	3	10	0.0080	245.5500	30693.75

the approximation to be, but also more resources are required, and the output obtained increase in complexity. Indeed,  $\text{CycRes}(f; r)$  is typically extremely large by any measure. For instance, consider  $f(z_1, z_2) = z_1^3 + z_1z_2 + z_2^3 + 1$ ; a polynomial in two variables of degree three with four terms. Table 1 provides a rough view of the size of  $\text{CycRes}(f; r)$  for different values of  $r$ .

In Table 2 we provide a comparison of the runtimes of our resultant algorithm and a standard computation using a built in resultant command for the following polynomials with real and complex coefficients in two and three variables:

$$\begin{aligned} f_1(z_1, z_2) &= z_1^3 + z_1z_2 + z_2^3 + 1, \\ f_2(z_1, z_2) &= (5 + \sqrt{-1}) \cdot z_1^3 + \sqrt{-1} \cdot z_1z_2 + (4 + \sqrt{-1}) \cdot z_2^3 + 1, \\ f_3(z_1, z_2, z_3) &= z_1^4z_2 + z_1z_2z_3^5 + z_1^2z_2^4 + z_1z_2^2 + z_1z_2z_3 + z_1z_2z_3^3 + 1. \end{aligned}$$

In Table 2, **Level** denotes the level of the cyclic resultant. **Runs** denotes the number of test runs we made for the particular polynomial and level. **Quick resultant** denotes the average runtime (in seconds) for the computation of cyclic resultants using our new recursive method, while **Iterated resultant** denotes the average runtime (in seconds) of a procedure using (1.4) and the **SINGULAR resultant** command. Finally, **Factor** denotes the ratio between the latter and former runtimes.

For instances, which can be computed very quickly, we took up to 100 test runs and computed the average runtime of both algorithms <sup>1</sup>.

**7.3. Solving the membership problem via lopsidedness.** In this section we describe how amoebas can be approximated in practice by solving the membership problem using lopsidedness certificates. Again, we use the following polynomial as a running example

$$(7.1) \quad f(z_1, z_2) = z_1^3 + z_1 z_2 + z_2^3 + 1.$$

The level two cyclic resultant equals (where we point out that, for notational convenience, the variables are denoted  $\mathbf{x}, \mathbf{y}, \mathbf{z}$  in SINGULAR output):

```
1 x48+28*x40y4-4*x36y12-4*x36+246*x32y8-156*x28y16-156*x28y4+
2 6*x24y24+576*x24y12+6*x24-860*x20y20-860*x20y8-156*x16y28+
3 969*x16y16-156*x16y4-4*x12y36+576*x12y24+576*x12y12-4*x12+
4 246*x8y32-860*x8y20+246*x8y8+28*x4y40-156*x4y28-156*x4y16+
5 28*x4y4+y48-4*y36+6*y24-4*y12+1
```

In order to compute the intersection of  $\mathcal{A}(f)$  with  $[s, t]^n \subseteq \mathbb{R}^n$ , the idea is to define a grid on  $[s, t]^n$ , and to test  $f$  and  $\text{CycRes}(f; r)$  for lopsidedness on every grid point up to a predefined  $r$ . According to Theorem 2.3 all grid points  $\mathbf{w} \in \mathbb{R}^n$  which do not belong to  $\mathcal{A}(f)$  are lopsided for  $\text{CycRes}(f; r)$  if  $r$  is sufficiently large. For our running example we choose the grid given by  $[-2, 2]^2 \cap (\frac{1}{10}\mathbb{Z})^2 \subseteq \mathbb{R}^2$ .

For every point  $\mathbf{w}$  on the grid, we need to evaluate every monomial of  $f$  or  $\text{CycRes}(f; r)$  at  $\text{Log}^{-1}(\mathbf{w}) = (\exp(w_1), \dots, \exp(w_n))$ . We test all points in our list for lopsidedness using  $f = \text{CycRes}(f; 2^0)$  and  $\text{CycRes}(f; 2^1), \dots, \text{CycRes}(f; 2^k)$ ; here  $k = 4$ . The algorithm successively searches for lopsidedness certificates for  $\mathbf{w} \notin \mathcal{L}(\text{CycRes}(f; j))$  for  $j \in \{0, \dots, k\}$ . If a certificate is found, then we save the level  $j$ . Otherwise  $j$  is increased by 1. If  $\text{CycRes}(f; r)$  is not lopsided at  $\mathbf{w}$ , then we presume that  $\mathbf{w} \in \mathbb{R}^n$  is a point in  $\mathcal{A}(f)$  since no certificate was found. This does not preclude further testing being performed, that may eliminate some of these points at a higher level. Results of the computation described in this section are displayed in Figure 1.

**7.4. Approximating unlog amoebas using semi-algebraic sets.** In this last section we demonstrate how to use our software to obtain an approximation of the unlog amoeba  $\mathcal{U}(f)$  for a given  $f \in \mathbb{C}[\mathbf{z}^{\pm 1}]$  using semi-algebraic sets. Moreover, we can draw an implicit plot of (the boundary of) the image of this semi-algebraic description.

Returning to our running example (7.1) we compute the semi-algebraic description of  $\mathcal{U}(f)$  in SINGULAR. In this example, we use levels 1, 2, and 3 given by the first three non-trivial instances of `quickcyclicresultant` for  $f$ . The output is shown in Figure 2. An interesting observation is that, according to this plot, the approximation of the sought semi-algebraic description of  $\mathcal{U}(f)$  does not necessarily improve monotonically. One can see that there exist certain areas where the `level 2` approximation is better than the `level 3` approximation. This behavior does not seem to be a plotting artifact, and raises theoretical questions on the nature of the convergence of this approximation.

<sup>1</sup>All computations were performed with SINGULAR Version 3-0-4 and on a Desktop computer, Distribution: Linux Mint 13 Maya, Kernel: 3.8.0-44-generic x86\_64, CPU: Quad core Intel Core i7-2600 CPU, RAM: 16GB.

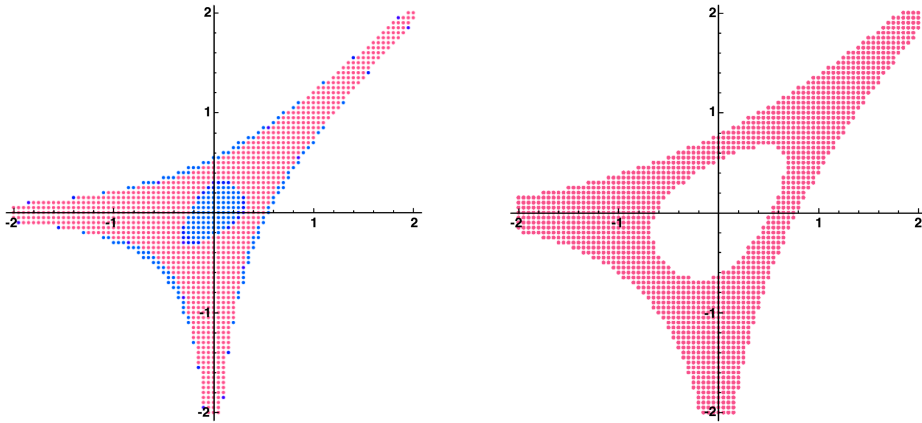


FIGURE 1. Plot of the amoeba of the polynomial  $f(z_1, z_2) = z_1^3 + z_2^3 + b \cdot z_1 z_2 + 1$  for  $b = 2$  and  $b = -4$  on a  $[-2, 2]^2$  grid with  $81 \times 81$  grid points and relaxation level at most 4. On the left-hand side the white points are the complement of the lopsided amoeba of  $f$ , light blue (light grey) points are lopsided on relaxation level 3 and dark blue points (dark grey; very few, mainly at the boundary of hole in the middle) at relaxation level 4. Red (black) points were never lopsided and hence are presumed to belong to the amoeba of  $f$ . On the right-hand side the amoeba coincides with the lopsided amoeba in this plot.

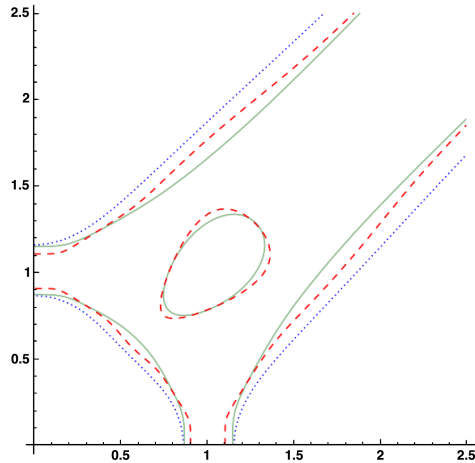


FIGURE 2. A plot of the semi-algebraic approximation of the unlog amoeba  $\mathcal{U}(f)$  for  $f(z_1, z_2) = z_1^3 + z_2^3 + 2 \cdot z_1 z_2 + 1$  using relaxed polynomials given by `quickcyclicresultant` on level 1 (blue/dotted), 2 (green/solid), and 3 (red/dashed).

## ACKNOWLEDGMENTS

We are very grateful to Alicia Dickenstein for her helpful suggestions, especially on resultants. We also thank Luis Felipe Tabera for his helpful comments. We are grateful to the anonymous referee for their helpful suggestions.

## REFERENCES

- [AKNR18] M. Avendaño, R. Kogan, M. Nisse, and J. M. Rojas, *Metric estimates and membership complexity for Archimedean amoebae and tropical hypersurfaces*, J. Complexity **46** (2018), 45–65, DOI 10.1016/j.jco.2017.11.008. MR3774347
- [BKS16] D. V. Bogdanov, A. A. Kytmanov, and T. M. Sadykov, *Algorithmic computation of polynomial amoebas*, Computer algebra in scientific computing, Lecture Notes in Comput. Sci., vol. 9890, Springer, Cham, 2016, pp. 87–100, DOI 10.1007/978-3-319-45641-6\_7. MR3593787
- [BPR03] S. Basu, R. Pollack, and M.-F. Roy, *Algorithms in Real Algebraic Geometry*, 2nd ed., Algorithms and Computation in Mathematics, vol. 10, Springer-Verlag, Berlin, 2006. MR2248869
- [CT65] J. W. Cooley and J. W. Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Comp. **19** (1965), 297–301, DOI 10.2307/2003354. MR0178586
- [Dev16] The Sage Developers, *Sagemath, the Sage Mathematics Software System (Version 6.4.1)*, 2016, <http://www.sagemath.org>.
- [DGPS15] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann, *SINGULAR 4-0-2 — A computer algebra system for polynomial computations*, <http://www.singular.uni-kl.de>, 2015.
- [dW17] T. de Wolff, *Amoebas, Nonnegative Polynomials and Sums of Squares Supported on Circuits*, Oberwolfach Rep. **23** (2015), 1308–1311.
- [FPT00] M. Forsberg, M. Passare, and A. Tsikh, *Laurent determinants and arrangements of hyperplane amoebas*, Adv. Math. **151** (2000), no. 1, 45–70, DOI 10.1006/aima.1999.1856. MR1752241
- [GKZ94] I. M. Gel'fand, M. M. Kapranov, and A. V. Zelevinsky, *Discriminants, Resultants, and Multidimensional Determinants*, Mathematics: Theory & Applications, Birkhäuser Boston, Inc., Boston, MA, 1994. MR1264417
- [Hil05] C. J. Hillar, *Cyclic resultants*, J. Symbolic Comput. **39** (2005), no. 6, 653–669, DOI 10.1016/j.jsc.2005.01.001. MR2167674
- [HL07] C. J. Hillar and L. Levine, *Polynomial recurrences and cyclic resultants*, Proc. Amer. Math. Soc. **135** (2007), no. 6, 1607–1618, DOI 10.1090/S0002-9939-06-08672-2. MR2286068
- [IdW16] S. Ilman and T. de Wolff, *Amoebas, nonnegative polynomials and sums of squares supported on circuits*, Res. Math. Sci. **3** (2016), Paper No. 9, 35, DOI 10.1186/s40687-016-0052-2. MR3481195
- [Mik00] G. Mikhalkin, *Real algebraic curves, the moment map and amoebas*, Ann. of Math. (2) **151** (2000), no. 1, 309–326, DOI 10.2307/121119. MR1745011
- [Mik04] G. Mikhalkin, *Amoebas of algebraic varieties and tropical geometry*, Different faces of geometry, Int. Math. Ser. (N. Y.), vol. 3, Kluwer/Plenum, New York, 2004, pp. 257–300, DOI 10.1007/0-306-48658-X\_6. MR2102998
- [Pel81] A. E. Pellet, *Sur un mode de séparation des racines des équations et la formule de Lagrange*, Darboux Bull. (2) **5** (1881), 393–395, In French.
- [PPT13] M. Passare, D. Pochekutov, and A. Tsikh, *Amoebas of complex hypersurfaces in statistical thermodynamics*, Math. Phys. Anal. Geom. **16** (2013), no. 1, 89–108, DOI 10.1007/s11040-012-9122-x. MR3029256
- [PR04] M. Passare and H. Rullgård, *Amoebas, Monge-Ampère measures, and triangulations of the Newton polytope*, Duke Math. J. **121** (2004), no. 3, 481–507, DOI 10.1215/S0012-7094-04-12134-7. MR2040284
- [PT05] M. Passare and A. Tsikh, *Amoebas: their spines and their contours*, Idempotent mathematics and mathematical physics, Contemp. Math., vol. 377, Amer. Math. Soc., Providence, RI, 2005, pp. 275–288, DOI 10.1090/conm/377/06997. MR2149010

- [Pur08] K. Purbhoo, *A Nullstellensatz for amoebas*, Duke Math. J. **141** (2008), no. 3, 407–445, DOI 10.1215/00127094-2007-001. MR2387427
- [Rul03] H. Rullgård, *Topics in geometry, analysis and inverse problems*, Ph.D. thesis, Stockholm University, 2003.
- [SdW13] F. Schroeter and T. de Wolff, *The boundary of amoebas*, 2013, Preprint, [arXiv:1310.7363](#).
- [TdW15] T. Theobald and T. de Wolff, *Approximating amoebas and coamoebas by sums of squares*, Math. Comp. **84** (2015), no. 291, 455–473, DOI 10.1090/S0025-5718-2014-02828-7. MR3266970
- [The02] T. Theobald, *Computing amoebas*, Experiment. Math. **11** (2002), no. 4, 513–526 (2003). MR1969643

DEPARTMENT OF MATHEMATICS, TEXAS A&M UNIVERSITY, COLLEGE STATION, TEXAS 77843  
*Email address:* [jensf@math.tamu.edu](mailto:jensf@math.tamu.edu)

DEPARTMENT OF MATHEMATICS, TEXAS A&M UNIVERSITY, COLLEGE STATION, TEXAS 77843  
*Email address:* [laura@math.tamu.edu](mailto:laura@math.tamu.edu)

DEPARTMENT OF MATHEMATICS, TEXAS A&M UNIVERSITY, COLLEGE STATION, TEXAS 77843  
*Email address:* [mehl144@tamu.edu](mailto:mehl144@tamu.edu)

DEPARTMENT OF MATHEMATICS, TEXAS A&M UNIVERSITY, COLLEGE STATION, TEXAS 77843  
*Current address:* Institut für Mathematik, Technische Universität Berlin, Straße des 17. Juni 136, 10623 Berlin, Germany  
*Email address:* [dewolff@math.tamu.edu](mailto:dewolff@math.tamu.edu)