SIAM J. SCI. COMPUT.
Vol. 35, No. 5, pp. B1034–B1054

A RANDOMIZED SUBDIVISION ALGORITHM FOR DETERMINING THE TOPOLOGY OF NODAL SETS*

GREGORY S. COCHRAN†, THOMAS WANNER‡, AND PAWEŁ DŁOTKO§

Abstract. Topology is a natural mathematical tool for quantifying complex structures. In many applications, such as, for example, in the context of phase-field models in materials science, the structures of interest arise as sub- or superlevel sets of continuous functions, i.e., as nodal domains. From a computational point of view, any attempt at constructing a truthful representation of the topology of nodal domains has to involve a discretization step, and it is natural to wonder whether this step introduces topological artifacts. In this paper, we present a randomized subdivision algorithm which, given a smooth function, constructs an adaptive rectangular grid containing the essential information necessary for approximating nodal domains. Furthermore, under mild regularity assumptions the algorithm will also provide a computer-assisted proof for the correctness of the approximation by showing that the rectangular grid can be used to construct rectangular complexes which are homotopy equivalent to the nodal domains of the function. Our method extends the results of [S. Day, W. D. Kalies, and T. Wanner, *Multiscale Model. Simul.*, 7 (2009), pp. 1695–1726], by employing a more accurate and efficient interval arithmetic range enclosure algorithm, as well as developing a randomized subdivision technique to virtually eliminate grid alignment effects.

Key words. homology, nodal domain, pattern characterization, nonuniform rectangular grid, topological information

AMS subject classifications. 55-04, 55N99, 65C99, 60G15, 60G60

DOI. 10.1137/120903154

1. Introduction.

In recent years, computational topology has increasingly been used as a tool for understanding high-dimensional and complex data generated both from experiment and from numerical simulations [8, 9, 10, 11, 12, 14, 15, 18, 27, 29, 30]. Many of these studies involve point-cloud data sets or networks, and they have demonstrated that topological methods can provide significant new insight into applied problems. Also in the context of evolution equations, such as deterministic and stochastic partial differential equations, computational topology has been used successfully, both for spatial-temporal chaos and for model validation [16, 17, 20]. In almost all of the above cases, the studies rely on the computability of the homology of sets, i.e., they determine the Betti numbers of the involved structures. These novel uses of computational homology in data analysis raise the fundamental mathematical question of the validity of the homology computations given error, noise, and the finite nature of the input data.

To the best of our knowledge the first work towards quantifying this uncertainty is due to Nyogi, Smale, and Weinberger [26]. They consider the problem of computing

*Submitted to the journal's Computational Methods in Science and Engineering section December 19, 2012; accepted for publication (in revised form) June 24, 2013; published electronically September 26, 2013. This research was supported in part by the Institute for Mathematics and its Applications at the University of Minnesota with funds provided by the National Science Foundation.
http://www.siam.org/journals/sisc/35-5/90315.html

†Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030 and ASM Research, Fairfax, VA 22030 (gregory.scott.cochran@gmail.com). This author's work was partially supported by a Ph.D. completion grant from the Provost's office at George Mason University.

‡Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030 (twanner@gmu.edu). This author's work was partially supported by NSF grants DMS-0639300, DMS-0907818, and DMS-1114923.

§Institute of Computer Science, Jagiellonian University, ul. St. Łojasiewicza 6, 30-348 Kraków, Poland (pawel.dlotko@uj.edu.pl).

the homology of embedded manifolds by a process of random sampling. More precisely, assume that $\mathcal{M}$ is a compact manifold in some Euclidean space, and randomly select N points $x_1, \ldots, x_N$ from $\mathcal{M}$ according to the uniform probability measure on $\mathcal{M}$. For suitable $\varepsilon > 0$, consider the union $\mathcal{U}_\varepsilon$ of closed balls of radius ε centered at the sampling points. How likely is it that the homology of $\mathcal{U}_\varepsilon$, which can easily be determined using the nerve lemma [14, section III.2] and the associated Cech complex, agrees with the homology of $\mathcal{M}$? For this, choose an arbitrary small constant $\delta > 0$. Then it was shown in [26] that one can derive an explicit lower bound on the necessary number N of randomly chosen sampling points such that the homology of $\mathcal{U}_\varepsilon$ coincides with the homology of $\mathcal{M}$ with probability at least $1 - \delta$. The bound on N depends on the allowable error δ, as well as on a crucial manifold parameter τ which encodes both local curvature and global separation information of the underlying manifold.

In certain applications, however, the above approach is not immediately applicable. Consider for example the problem of studying the topology of evolving microstructures which are generated through phase-field models in materials science. Models of this type are given by deterministic or stochastic partial differential equations over a base domain $\Omega \subset \mathbb{R}^d$ which describe the evolution of one or more phase variables $u(t, x)$ for times $t \geq 0$ and the spatial coordinate $x \in \Omega$. These phase variables define the associated microstructures via a thresholding procedure as sublevel or superlevel sets of u. In other words, if u is a typical phase variable and $\mu \in \mathbb{R}$ an appropriate threshold, then we are interested in the topology of the sets

$$N^{\pm}(t) = \{x \in \Omega \ : \ \pm (u(t, x) - \mu) \geq 0\} \ , \tag{1.1}$$

i.e., in the topology of the nodal domains of u. From now on, for the sake of notational simplicity, we will usually drop the threshold μ and consider the equivalent problem of computing the nodal domain for threshold 0—one just has to replace the function u by $u - \mu$. Solving phase-field models numerically can be quite involved, and already furnishes some discretization of the underlying domain Ω. Such discretizations could be a regular rectangular grid in the case of finite-difference methods, or more complicated meshes if finite elements are used. In order to keep the computational effort minimal, it would therefore be advantageous to use the provided discretization of the domain for the homology computation, rather than having to introduce another discrete set of randomly chosen sampling points. In addition, the nodal sets are only given implicitly through the phase variable which makes it harder to generate random samples with respect to the uniform probability measure and to obtain good estimates on the crucial manifold parameter τ from u. Thus, even though under suitable conditions on the phase variable u the nodal domains $N^{\pm}(t)$ are embedded manifolds with boundary, so that in principle the results of [26] are applicable, one does encounter serious natural limitations.

For the case of stochastic evolution equations, Mischaikow and Wanner have recently developed a probabilistic framework for assessing the correctness of homology computations via uniform discretizations [21, 22]. The approach considers the homology of nodal domains of random fields in one and two space dimensions and assumes the following setting. Consider a rectangular domain $\Omega \subset \mathbb{R}^d$, a probability space $(F, \mathcal{F}, \mathbb{P})$, and a random field $u : \Omega \times F \to \mathbb{R}$. Furthermore, consider a grid of N^d equispaced grid points in Ω. In order to determine the topology of the random nodal domains $N^{\pm}(\omega) = \{x \in \Omega \ : \ \pm u(x, \omega) \geq 0\}$, where $\omega \in F$, we approximate these sets by two cubical complexes $Q^{\pm}(\omega)$ in the following way. Depending on the sign of u at each grid point, a rectangular region around the grid point is either added

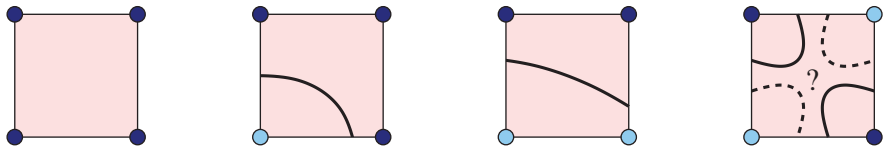

FIG. 1.1. *The three images from the left show all possible corner sign configurations of a square in the final adaptive grid produced by the algorithm in* [7]*, up to rotation, symmetry, and sign inversion. The sign configuration shown in the rightmost image will never validate; if such a square is encountered it will automatically be subdivided.*

to $Q^+(\omega)$ or $Q^-(\omega)$. This results in a raster image version of the nodal domains. For the cubical complexes $Q^{\pm}(\omega)$, one can easily and correctly compute the homology, for example with the software package CHomP [3]. Thus, in order to determine whether the above discretization furnishes the correct homology of the nodal domains $N^{\pm}(\omega)$, one needs to make sure that the cubical approximations do not change the topology of the nodal domains. It was demonstrated in [21, 22] that one can derive explicit bounds on the probability that the topology of the nodal sets $N^{\pm}(\omega)$ agrees with the topology of their cubical approximations $Q^{\pm}(\omega)$. The error bounds are a function of the underlying discretization size N and averaged Sobolev norms of the random field. It can be shown [6] that these estimates are sharp, and that they provide a priori bounds for the suitability of certain uniform grid sizes.

Finally, a more a posteriori point of view of the homology verification problem was taken by Day, Kalies, and Wanner in [7], where the fundamental idea is to provide a computer-assisted proof that the homology of a considered nodal domain is correct, while one is actually computing this homology information. It was demonstrated that this can be accomplished for functions $u : \Omega \to \mathbb{R}$ which are defined on rectangular two-dimensional domains by employing an adaptive cubical refinement procedure which combines a binary subdivision method with interval arithmetic and mean-value–type arguments. The method of [7] produces a decomposition of the two-dimensional domain into a union of squares[1] in such a way that the precise topology of the nodal lines of the function u, which delineate the two nodal domains, can be deduced. To achieve this, the squares in the final grid are determined in such a way that the function values on the corners of each square Q correctly describe the location of the nodal line within Q in the following sense. If the function values at the corners are all positive, then the function u is positive on all of Q; if exactly one corner has a negative function value, while the remaining three have positive u-values, then the zero set of u in Q consists of a simple non-self-intersecting curve which originates on one edge of the square and ends at another edge. If two corners of Q are u-negative and two corners are u-positive, then we have two possibilities: In one case, the u-positive (and consequently u-negative) corners are connected by an edge in Q, and in this case the zero set of u in Q is a simple non-self-intersecting curve which originates on one of these connecting edges and ends in the other connecting edge. On the other hand, if there is no edge in Q joining u-positive (and consequently u-negative) corners of Q, then further subdivision is necessary to determine the zero set of u. See Figure 1.1 for an illustration. Once the final adaptive cubical grid has been determined,

[1] In this paper, we denote two-dimensional cells obtained as a result of uniform subdivisions by *squares*, and we refer to two-dimensional cells obtained as a result of nonuniform subdivisions as *rectangles*. Note that this notation is not always consistent with the standard geometric meaning of squares and rectangles.

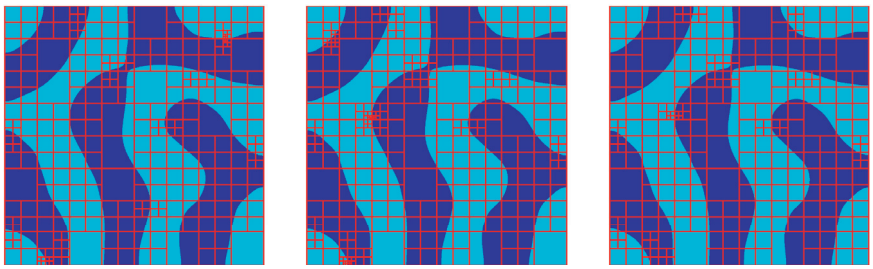

FIG. 1.2. *Sample images of the final grid produced by the homology verification algorithm of* [7]. *From left to right, the images show three time snapshots of nodal domains of a solution to the Cahn–Hilliard equation, for increasing time.*

a cubical complex with the same homology as the nodal domain is constructed and can be used to compute the Betti numbers of the nodal domains. This algorithm has been implemented in one and two space dimensions [6, 7]. Examples of the resulting final adaptive validation grids are shown in Figure 1.2. We would like to point out, however, that this algorithm does not always succeed. For practical purposes one has to establish a lower bound on the size of the squares in the final adaptive grid, and there are cases where this minimal size is not sufficient for validation. The algorithm will also fail whenever it is impossible to validate the sign of the u-value at one of the corners of a grid square.

While the method presented in [7] was applied successfully to a number of specific situations, these proof-of-concept applications also uncovered a number of serious shortcomings of the algorithm. Its limitations became particularly evident for random trigonometric polynomials of high degree and for evolution equations. In the latter case, the temporal evolution of the nodal lines causes the algorithm to break down whenever a nodal line comes close to and then crosses a dyadic subdivision line, as can clearly be seen in the three time snapshots of Figure 1.2. In [7] this problem could only be addressed by considering additional initial subdivisions of the underlying domain, and then rerunning the algorithm a number of times. As a result, only moderately complicated evolution problems could be considered due to time constraints. In addition, in the case of trigonometric polynomials of high degree the validation procedure was frequently unable to find a final adaptive grid whose squares were larger than a prescribed minimum size, which is related to the machine precision of the hardware used.

In the current paper we present a significant improvement of this method which addresses the above shortcomings. One of the main bottlenecks of the method in [7] is its use of range enclosure methods. For example, it order to validate a cube whose corners all have positive u-values, the original algorithm employs a simple range enclosure method using interval arithmetic and a mean value form; see, for example, [23, 24]. While this method is better than the straightforward use of interval arithmetic to find an enclosure for the range $u(Q)$, particularly for random trigonometric polynomials of high degree it leads to sizable overestimations, which then lead to more subdivisions, and thus to larger final adaptive grids—or even the failure of the algorithm. In the current paper, we show that one can extend the method of Skelboe, which was introduced in the context of optimization, to improve the range enclosure techniques which are used in the validation step of the algorithm. The resulting algorithm is a branch-and-bound–type algorithm which uses adaptive refinements to obtain suitable range enclosures using interval arithmetic. The second, and even more important, ex-

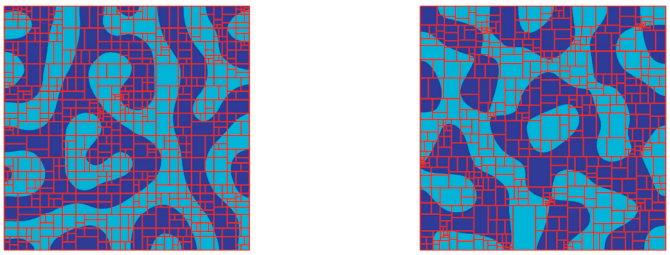

FIG. 1.3. *Two nonuniform cubical approximations for patterns generated by the Cahn–Hilliard equation. In the left image the subdivisions are created using the golden ratio, in the right image the subdivision ratios are taken uniformly from the interval* $[0.3, 0.7]$.

tension presented in this paper is concerned with the grid alignment problems which were described in the context of evolving patterns. For this we present a randomized rectangular subdivision algorithm which leads to final nonuniform rectangular validation grids. Rather than employing exclusively dyadic subdivisions, we randomize the subdivision process and break the inherent symmetries through noncentered partitions. Even though most existing homology codes for rectangular grids require regular cubical grids as input [3], the recent extension of the coreduction homology algorithm of [25] to regular CW-complexes obtained in [13] is directly applicable to the present situation. Two sample randomized adaptive grids are shown in Figure 1.3; a more detailed description of the terminology in the caption will be given in later sections.

The remainder of this paper is organized as follows. In section 2 we describe the original algorithm of [7] in more technical detail, as well as its two major deficiencies. In the following section 3 we propose to resolve one of these issues by developing an improved range enclosure method which is based on Skelboe's approach. These interval arithmetic considerations are then combined with a randomized subdivision algorithm to address the second deficiency. The final randomized validation algorithm is based on a preliminary version which was discussed in [4]. Finally, the effectiveness of the resulting new approach to validating nodal domains of smooth functions via rigorous computations is illustrated in section 4, which is devoted to two case studies. First, we consider nodal domains created by a double-well potential with thresholds close to the critical values of the potential. The second case study is concerned with time-evolving patterns generated by the Cahn–Hilliard model for phase separation.

2. Rigorous approximation of nodal domains. In this section we give a more technical outline of the validation algorithm developed in [7]. For this, let $\Omega \subset \mathbb{R}^2$ denote a closed square domain, where we implicitly assume from now on that the adjective "square" also implies that the sides of the square are parallel to the coordinate axes. Furthermore, let $u : \Omega \to \mathbb{R}$ denote a smooth function. We assume that u, as well as its derivatives, can be evaluated using interval arithmetic. This means that there exist computable real-valued functions $\underline{u}$ and $\overline{u}$ which are defined on the collection of compact subsets of Ω such that $u(R) \subset [\underline{u}(R), \overline{u}(R)]$ for all square subsets $R \subset \Omega$. In other words, for every square subset R we can determine an interval, namely, $[\underline{u}(R), \overline{u}(R)]$, which encloses the actual range $u(R) \subset \mathbb{R}$. We would like to point out that these assumptions could certainly be relaxed. For example, assuming that the rectangles are parallel to the coordinate axes is made for convenience. Also, one could use techniques from automatic differentiation to provide the derivatives of u. For the sake of simplicity, we adhere to the above setting.

Using the above notation, the validation algorithm of [7] can be described as follows. Let $\mathcal{Q}$ denote an initial collection of squares with nonoverlapping interiors whose union equals Ω, for example, simply choose $\mathcal{Q} = \{\Omega\}$. Furthermore, let $\mathcal{V} = \emptyset$. Then for any square $R \in \mathcal{Q}$ the following steps are performed.

(S1) Remove R from $\mathcal{Q}$.

(S2) For each of the four corners of R, validate the sign of the function value of u using interval arithmetic, i.e., determine an interval enclosure for the value of u at the corner and test whether it contains zero. If it does for any of the corners, the algorithm fails.

(S3) Based on the rigorous signs of the function values of u at the corners of R, determine which of the four cases shown in Figure 1.1 occurs (up to sign negation and rotation). Then perform one of the following *validation tests*.

  (a) If all four signs are the same, compute the enclosure $[\underline{u}(R), \overline{u}(R)]$ for the range $u(R) \subset \mathbb{R}$. If it contains zero, then R fails the validation test, otherwise it passes.

  (b) If all but one of the signs are the same, use interval arithmetic as above to show that both $u_x(R)$ and $u_y(R)$ do not contain zero. If this is the case, R passes the test, otherwise it fails.

  (c) If the signs are as in the third image in Figure 1.1, use interval arithmetic to verify that the range of u on the top and bottom horizontal edges of R does not contain zero, and that $u_y(R)$ does not contain zero. If this is the case, R passes the test, otherwise it fails. Similarly treat the case when the corners of the left vertical edge have the same sign, and the ones on the right edge have the opposite sign by enclosing the range of u on these edges, and the range of u_x over R.

  (d) In this last case, i.e., the rightmost image of Figure 1.1, R automatically fails the validation test.

  We would like to point out that if R passes the validation step, then the nodal domain geometry of u in R is qualitatively as shown in the respective image of Figure 1.1.

(S4) If the validation test in step (S3) fails, subdivide R into four equal subsquares via binary subdivision, and add each of the squares to $\mathcal{Q}$. If R passes the test, add R to $\mathcal{V}$.

This iteration ends if either $\mathcal{Q} = \emptyset$ is reached, or if one square in $\mathcal{Q}$ is smaller than a prespecified minimal size. While in the latter case the algorithm fails, in the former case the algorithm produces a subdivision of Ω which allows one to exactly determine the location of the nodal lines within each square in the final grid.

It is clear from the form of the algorithm that upon successful completion, one can easily construct a complete representation of the nodal domains of u which is amenable to a subsequent homology computation. This has been described in more detail in [7], and is employing software for cubical homology based on [19]. Since this software requires uniform cubical complexes as input, the validation procedure had to produce a final adaptive grid of squares which was created through binary subdivisions. As we mentioned in the introduction, the use of these binary subdivisions in turn led to significant grid alignment issues—which is the first major deficiency of the method in [7].

But also from a more technical point of view the original algorithm was far from optimal. In order to compute the enclosure $[\underline{u}(R), \overline{u}(R)]$ for the range $u(R) \subset \mathbb{R}$ over a square R, the approach in [7] used a method which is based on the mean value

theorem. If we assume that $R = [a_1, b_1] \times [a_2, b_2]$, then one can easily see that

$$u(R) \subset u\left(\frac{a_1+b_1}{2}\,,\,\frac{a_2+b_2}{2}\right) + \frac{b_1-a_1}{2}\cdot u_x(R)\cdot[-1,1] + \frac{b_2-a_2}{2}\cdot u_y(R)\cdot[-1,1].$$

In other words, the range enclosure is computed by interval enclosures of function values of u at points (which can usually be obtained with fairly tight bounds), combined with bounds for the gradient of u over the square R. The latter bounds, unfortunately, can be extremely large. In particular, if one assumes that the function u is given as a Fourier-type series and the size of R is larger than the wavelength of modes with nontrivial coefficients, then the above formula can lead to large overestimations, even if the contributions of the respective small wavelength Fourier modes to the nodal domain structure are small. This leads both to a large number of unsuccessful validation attempts, and also to final grids which contain considerably more squares than necessary.

We close this section with a brief comment on the general applicability of the validation algorithm. In principle, the algorithm can only be expected to work if the function u is at least twice continuously differentiable, and if zero is a regular value for u, i.e., at every zero of the function u the gradient ∇u is nonzero. This ensures that the zero set of u in Ω consists of nonintersecting C^1-curves, and it guarantees that small perturbations of u lead to small perturbations of the nodal domain which do not change their topology. This robustness under perturbations is needed. For example, if the zero set contains two intersecting curves, infinitesimally small perturbations could break the crossing in two fundamentally different ways. Notice that this necessarily implies that zero is not a regular value for u. Thus, in a generic sense the algorithm can be expected to apply "almost always."

3. A randomized subdivision algorithm. Due to the two shortcomings described in the previous section, the validation approach presented in [7] worked only for proof-of-concept case studies. The improvements of the algorithm presented in the current paper center around two concepts: significantly improving the employed range enclosure methods, and moving away from deterministic subdivisions using squares as the basic building blocks of the adaptive grid.

3.1. Rigorous positivity verification. As our discussion in section 2 showed, the basic validation step boils down to answering the following question of positivity:

(P) Given a smooth function v which is defined on a compact domain R, is the function v strictly positive on R?

For example, if in (S3(a)) in section 2 the signs of u at the vertices are all positive, then it is not necessary to obtain a full range enclosure for $u(R)$. It suffices to find a rigorous positive lower bound on the range. Similarly, in (S3(b)) it suffices to establish the positivity of either u_x or $-u_x$, as well as of either u_y or $-u_y$. Therefore, depending on the case, we consider (P) with $v \in \{u, \pm u_x, \pm u_y\}$.

In the context of rigorously solving optimization problems, Skelboe [28] proposed a branch-and-bound–type algorithm to determine a lower bound on the range of a function over a rectangular domain, which is correct up to a prespecified tolerance ε. This algorithm is described in detail in [23, 24], and at first glance it seems to be perfectly suited for answering the basic question (P) mentioned above. Note, however, that we are not interested in a precise lower bound if the function v is indeed strictly positive on R. In fact, any rigorous positive lower bound will suffice. Moreover, if the function does take negative values on R, we do not require a lower bound on

the range, but rather a proof that there are negative function values. We therefore propose a modification of Skelboe's algorithm which is guided by the following two design goals.

(G1) If the function v takes negative values on R, find a rigorous upper bound for such a function value that is less than 0 as quickly as possible.

(G2) At the same time, discard subdomains of R over which interval arithmetic shows v to be strictly positive.

These principles lead to an algorithm which is considerably faster than a direct implementation of Skelboe's method, and which significantly improves on the method presented in [7]. In anticipation of the next section, from now on we focus on rectangular domains R, where we implicitly assume that the sides of the involved rectangles are parallel to the coordinate axes.

In order to describe our algorithm, let v be a smooth function, and let R denote a rectangular domain in $\mathbb{R}^2$. In addition, we require three input parameters which affect the performance of the algorithm, are described in more detail below, and are given by

$$\begin{aligned} \varrho_{\mathrm{mach}} &: \text{a positive constant related to the machine precision,} \\ f_\varrho &: \text{a scaling factor between 0 and 1 used in (3.2), and} \\ M_{\mathrm{depth}} &: \text{a maximal subdivision depth.} \end{aligned} \tag{3.1}$$

The main data structure for our method is an ordered list $\mathcal{L}$, which consists of triples $\mathcal{X} = (\mathcal{X}_{\mathrm{rect}}, \mathcal{X}_{\mathrm{low}}, \mathcal{X}_{\mathrm{depth}}) \in 2^R \times \mathbb{R} \times \mathbb{N}$ with the following entries:

- The first entry $\mathcal{X}_{\mathrm{rect}} \subset \mathbb{R}^2$ denotes a rectangular subdomain of the rectangle R;
- the second entry $\mathcal{X}_{\mathrm{low}} \in \mathbb{R}$ contains a rigorously determined lower bound on the range $v(\mathcal{X}_{\mathrm{rect}})$ of v over $\mathcal{X}_{\mathrm{rect}}$; and
- the third entry $\mathcal{X}_{\mathrm{depth}} \in \mathbb{N}$ denotes the current subdivision depth (see below).

The list $\mathcal{L}$ is ordered with respect to the second component, and throughout the algorithm the second components of triples in $\mathcal{L}$ are all nonpositive. The algorithm starts with the following initialization steps.

(P1) Find rigorous enclosures for the function values of v at the four corners and at the center of R, and let $M_v \in \mathbb{R}$ denote the smallest of the five rigorous upper bounds. In other words, we are guaranteed that v attains function values less than or equal to M_v over R. In addition, find a rigorous enclosure $[\underline{v}(R), \overline{v}(R)] \supset v(R)$ for the range of v over R.

(P2) If $M_v \leq 0$, we have shown that v attains nonpositive values on R, and the algorithm exits with the answer `false` to question (P).

(P3) If $\underline{v}(R) > 0$, we have shown that v is strictly positive on R, and the algorithm exits with the answer `true`.

(P4) If on the other hand we have $M_v > 0$ and $\underline{v}(R) \leq 0$, then we initialize the list $\mathcal{L}$ with the triple $\mathcal{X} = (R, \underline{v}(R), 1)$, and we define the threshold

$$\varrho = \max\{f_\varrho \cdot M_v \ , \ \ \varrho_{\mathrm{mach}}\} > 0 \ . \tag{3.2}$$

The constant $\varrho_{\mathrm{mach}} > 0$ in this definition is usually chosen as a small multiple of the machine precision of the underlying hardware. It provides a lower bound on when numbers are considered to be close enough to zero to be treated as zero. In addition, the scaling factor f_ϱ links the actual validation threshold to the absolute size M_v defined in (P1). This relative fine tuning

of the threshold ϱ is necessary to increase the performance of the algorithm, as will be demonstrated in the numerical examples later on, at the end of section 4.1.

The constant $\varrho > 0$ defined in (3.2) serves as a threshold variable which is used to decide whether a computed range enclosure over a subdomain of R is sufficiently tight. This will be discussed in more detail below. After the initialization step, the algorithm performs the following main loop as long as $\mathcal{L} \neq \emptyset$.

(P5) (a) Remove the first triple $(\mathcal{X}_{\text{rect}}, \mathcal{X}_{\text{low}}, \mathcal{X}_{\text{depth}})$ from the list $\mathcal{L}$. Recall that this implies $\mathcal{X}_{\text{low}} \leq 0$ is a rigorous lower bound for the range of v over the rectangle $\mathcal{X}_{\text{rect}} \subset R$.

(b) If the subdivision depth $\mathcal{X}_{\text{depth}}$ exceeds the prespecified maximal subdivision depth M_{depth}, exit the algorithm with `false`.

(c) Subdivide the rectangle $\mathcal{X}_{\text{rect}}$ into two subrectangles $\mathcal{X}^{(1)}_{\text{rect}}$ and $\mathcal{X}^{(2)}_{\text{rect}}$ by subdividing the longer edge of $\mathcal{X}_{\text{rect}}$ into two equal pieces. After determining range enclosures for v over the two newly created rectangles, form the two triples

$$\mathcal{X}^{(k)} = \left(\mathcal{X}^{(k)}_{\text{rect}},\ \underline{v}\left(\mathcal{X}^{(k)}_{\text{rect}}\right),\ \mathcal{X}_{\text{depth}} + 1\right) \qquad \text{for} \qquad k = 1, 2\,.$$

(d) Find rigorous enclosures for the function value of v at the centers of $\mathcal{X}^{(1)}_{\text{rect}}$ and $\mathcal{X}^{(2)}_{\text{rect}}$, and exit with `false` if the upper bound of either of these is nonpositive.

(e) For $k = 1, 2$ perform the following steps:

* If $\mathcal{X}^{(k)}_{\text{low}} \leq 0$ and $\overline{v}(\mathcal{X}^{(k)}_{\text{rect}}) - \underline{v}(\mathcal{X}^{(k)}_{\text{rect}}) \leq \varrho$, then exit with `false`;
* if $\mathcal{X}^{(k)}_{\text{low}} \leq 0$ and $\overline{v}(\mathcal{X}^{(k)}_{\text{rect}}) - \underline{v}(\mathcal{X}^{(k)}_{\text{rect}}) > \varrho$, then add the triple $\mathcal{X}^{(k)}$ to the list $\mathcal{L}$, ordered with respect to $\mathcal{X}^{(k)}_{\text{low}}$.

Notice that if we have $\mathcal{X}^{(k)}_{\text{low}} > 0$, then the triple $\mathcal{X}^{(k)}$ will no longer be considered. In this case, we have rigorously verified that $v > 0$ on $\mathcal{X}^{(k)}_{\text{rect}}$, and this rectangle is discarded from further considerations.

If at some point $\mathcal{L} = \emptyset$, then we have verified that $v > 0$ on R, and the algorithm exits with the answer `true` to question (P). Throughout the algorithm, range enclosures for the function v are determined using the monotonicity test form described in [24, section 6.4], rather than the purely mean-value based approach used in [7].

One can easily see that if the above algorithm returns `false`, then either we have reached the maximal recursion depth M_{depth}, or we have rigorously established the existence of a function value of v on R which is less than or equal to ϱ. On the other hand, if the algorithm returns `true`, we obtain a computer-assisted proof which shows that v is strictly positive on R. In terms of realizing our two design goals, part (P5(e)) of the algorithm ensures that our goal (G2) is satisfied, by discarding subdomains over which v has shown to be positive. Furthermore, goal (G1) is achieved by always working on the rectangle $\mathcal{X}_{\text{rect}}$ with the smallest lower range bound $\mathcal{X}_{\text{low}} \leq 0$, i.e., by concentrating on the rectangle which is most likely to contain negative function values, if in fact they exist. Finally, while the choice of ϱ as a fixed percentage of M_v made in (P4) might seem strange at first, this adaptive definition is necessary to achieve an efficient algorithm. This will be discussed in more detail in the last section of this paper. At this point we would merely like to mention that this is due to the fact that if v takes only large positive values on R, choosing too small a tolerance ϱ leads to unnecessarily tight enclosure conditions in (P5(e)).

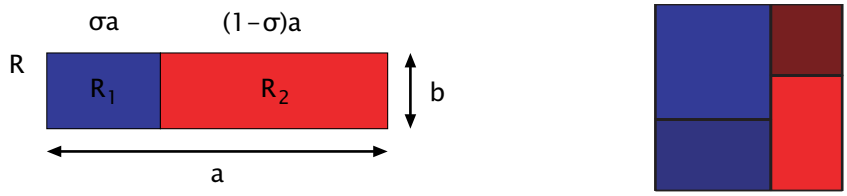

FIG. 3.1. *The two panels illustrate Definition* 3.1. *Given a rectangle with side lengths* $a \geq b$ *and a subdivision parameter* $\sigma \in (0,1)$*, the rectangle is subdivided into two subrectangles by splitting the longer side into two segments of lengths* σa *and* $(1-\sigma)a$*. This is illustrated in the left panel. The right panel shows the three types of rectangles which are obtained after subdivision using the golden ratio, as introduced at the end of Definition* 3.1.

3.2. Randomized subdivisions. While the algorithm for deciding the positivity question presented in the last section addresses inefficiencies associated with the use of interval arithmetic in [7], we still have to resolve the grid alignment issues with tracking the topology of evolving patterns. As mentioned in the introduction, these alignment issues are due to the fact that the method of [7] exclusively relied on binary subdivisions. Reliance on this subdivision method was in turn a consequence of the homology code which was available at the time for large data sets, and which required an underlying uniform cubical grid. Recently it has been shown in [13] that the fast coreduction method for homology computation developed in [25] for the uniform cubical setting can in fact be extended to the case of regular CW-complexes, and Dłotko et al. [13] have implemented this method for the case of rectangular, not necessarily uniform complexes.

The availability of fast homology code for nonuniform rectangular complexes makes it possible to change the subdivision rule of [7] in such a way that in practice all grid alignment problems are eliminated. For this, assume that we are given a smooth function u on a rectangular domain $\Omega \subset \mathbb{R}^2$. Our improved validation algorithm still maintains the basic structure as outlined in (S1) through (S4) in section 2, but if a rectangle R fails the subdivision step, we randomly divide it into two *noncongruent* subrectangles. To describe this in more detail, we first need the following definition.

DEFINITION 3.1. *Let* R *be a rectangle with side lengths* $a \geq b$*, and let* $\sigma \in (0,1)$ *be arbitrary. Then we say that* R is subdivided into two rectangles using the subdivision parameter σ*, if* R *is divided into a rectangle* R_1 *with side lengths* σa *and* b*, and a rectangle* R_2 *with side lengths* $(1-\sigma)a$ *and* b*.*

One special case of this definition is of particular interest later on. We say that a rectangle R is subdivided using the golden ratio*, if the subdivision parameter is given by either* $\sigma = (\sqrt{5}-1)/2$ *or by* $\sigma = (3-\sqrt{5})/2$*. As will be shown in Proposition* 3.4*, these choices lead to subdivisions with special properties which are based on properties of the golden ratio* $(1+\sqrt{5})/2$*.*

This subdivision procedure is illustrated in the left panel of Figure 3.1. It also forms the basis for the following randomized version of the validation algorithm of [7].

(V) RANDOMIZED VALIDATION ALGORITHM: Assume we are given a smooth function u defined on a closed rectangle $\Omega \subset \mathbb{R}^2$. Furthermore, assume that $\mathcal{S} \subset (0,1)$ is a set of possible subdivision ratios with $\mathcal{S} = 1 - \mathcal{S}$. Then the randomized validation algorithm proceeds as follows:
Let $\mathcal{Q} = \{\Omega\}$ and let $\mathcal{V} = \emptyset$. As long as $\mathcal{Q}$ is not empty, perform for any rectangle $R \in \mathcal{Q}$ steps (S1) through (S3) from section 2. However, step (S4) in section 2 is replaced by the following.

(V4) If a rectangle R fails the validation step in (S3), randomly pick a subdivision ratio $\sigma \in \mathcal{S}$ according to the uniform measure on $\mathcal{S}$, subdivide the rectangle R into two rectangles R_1 and R_2 using the subdivision parameter σ as described in Definition 3.1, and then add the two new rectangles to $\mathcal{Q}$. If R passes the test, add R to $\mathcal{V}$.

We would like to point out that the condition $\mathcal{S} = 1 - \mathcal{S}$ is imposed only to make sure that there is no directional bias in the rectangle subdivision process. It could easily be removed in the following.

It will be shown in the next section that this randomized version of the validation algorithm does indeed take care of grid alignment effects. Yet, before turning our attention to some case studies which demonstrate the applicability of the method (V), we still need to address a technical issue. While the method produces a nonuniform rectangular decomposition of the base domain Ω, it is not immediately clear what type of rectangles are produced through the random selection of subdivision ratios from $\mathcal{S}$. To quantify this question, recall that if R denotes a rectangle with side lengths $a \geq b$, then its *aspect ratio* is given by $a/b \geq 1$. Can we obtain a priori information on the aspect ratios that are produced by (V)? In particular, can the aspect ratios become arbitrarily large?

In the remainder of this section it will be shown that it is possible to control the set of possible aspect ratios of rectangles in the final decomposition of Ω by suitably choosing the set $\mathcal{S}$ in (V). For this, we first need the following lemma.

LEMMA 3.2. *Consider a rectangle R with side lengths $a \geq b$ and let $0 < \sigma \leq 1/2$ be arbitrary. Assume that R is subdivided into two new rectangles R_1 and R_2 using the subdivision parameter σ as introduced in Definition* 3.1. *Then the following hold:*

- *If the aspect ratio a/b of R is larger than $1/\sigma$, then both R_1 and R_2 have aspect ratios strictly smaller than a/b;*
- *if the aspect ratio a/b of R is less than or equal to $1/\sigma$, then both R_1 and R_2 have aspect ratios at most $1/\sigma$.*

Proof. We first discuss the rectangle R_1. One can easily see that if the aspect ratio of R satisfies $1 \leq a/b \leq 1/\sigma$, then R_1 has the aspect ratio $b/(\sigma a) \geq 1$, which due to $b \leq a$ is bounded above by $1/\sigma$. If on the other hand we have $a/b \geq 1/\sigma$, then R_1 has the aspect ratio $\sigma a/b \geq 1$, which due to $\sigma < 1$ is strictly less than a/b. Thus, the assertions of the lemma hold for R_1.

We now turn our attention to the second rectangle R_2. Notice that due to our choice of σ we have $0 < \sigma \leq 1/2$, and therefore

$$0 < \frac{1}{1-\sigma} \leq \frac{1}{\sigma}\,. \tag{3.3}$$

Now assume that the aspect ratio of R satisfies $1 \leq a/b \leq 1/(1-\sigma)$. Then R_2 has the aspect ratio $b/((1-\sigma)a) \geq 1$, which due to $b \leq a$ is bounded above by $1/(1-\sigma)$. The latter bound is less than or equal to $1/\sigma$ according to (3.3). On the other hand, if $a/b \geq 1/(1-\sigma)$, then R_2 has the aspect ratio $(1-\sigma)a/b \geq 1$, which due to $\sigma > 1$ is strictly less than a/b. Together with (3.3), this shows that the lemma also holds for R_2. ∎

Using this lemma one can now easily deduce a tight bound on the set of possible aspect ratios which are produced by (V). This is the subject of the following result.

PROPOSITION 3.3. *Let $\mathcal{S} \subset (0,1)$ denote a compact and nonempty set which satisfies $\mathcal{S} = 1 - \mathcal{S}$. Furthermore, assume that R is a rectangle whose aspect ratio r lies in the interval $[1, 1/\min \mathcal{S}]$. If one then subdivides R into two rectangles using*

a subdivision parameter $\sigma \in \mathcal{S}$, then the aspect ratios of both new rectangles are contained in $[1, 1/\min \mathcal{S}]$.

In particular, if an initial square is subdivided recursively using arbitrarily selected subdivision parameters in the set $\mathcal{S}$, then all subrectangles which are created in this way have aspect ratios in the interval $[1, 1/\min \mathcal{S}]$.

Proof. According to our assumptions we have $1 \leq r \leq 1/\min \mathcal{S}$. Without loss of generality we can assume that $\sigma \in \mathcal{S} \cap (0, 1/2]$, since otherwise we consider $1 - \sigma$ instead. According to Lemma 3.2, in the case $r \geq 1/\sigma$ the two generated rectangles R_1 and R_2 have aspect ratios strictly less than $r \leq 1/\min \mathcal{S}$. On the other hand, if we have $1 \leq r \leq 1/\sigma$, then the lemma shows that both R_1 and R_2 have aspect ratios less than or equal to $1/\sigma \leq 1/\min \mathcal{S}$. This completes the proof of the proposition. ∎

The above result shows that the rectangles occurring in the final rectangular subdivision of a square Ω cannot be arbitrarily narrow. In fact, by choosing the subdivision ratios carefully, one can minimize the number of possible aspect ratios even more. Note for example that if one chooses $\mathcal{S} = \{1/2\}$, then the final grid would only consist of squares and rectangles of aspect ratio two. However, this reduces the algorithm (V) basically to the method of [7] with the ensuing grid alignment problems. These can be avoided with the following choice.

PROPOSITION 3.4. *Consider the set*

$$\mathcal{S} = \left\{ \frac{\sqrt{5}-1}{2} \approx 0.618\,, \quad \frac{3-\sqrt{5}}{2} \approx 0.382 \right\}\,,$$

which is based on the golden ratio. Furthermore, assume that an initial square is subdivided recursively using randomly selected subdivision parameters in the set $\mathcal{S}$. Then all subrectangles which are created in this way have aspect ratios in the set

$$\left\{ 1\,, \quad \frac{1+\sqrt{5}}{2} \approx 1.61803\,, \quad \frac{3+\sqrt{5}}{2} \approx 2.61803 \right\}\,.$$

In other words, the subdivision process leads to only three types of rectangles, as shown in Figure 3.1.

Proof. Without loss of generality we only consider the σ-value in $\mathcal{S} \cap (0, 1/2]$, i.e., the case $\sigma = (3 - \sqrt{5})/2$. Subdividing a square with this subdivision parameter σ leads to two rectangles with aspect ratios

$$\frac{1}{\sigma} = \frac{3+\sqrt{5}}{2} \quad \text{and} \quad \frac{1}{1-\sigma} = \frac{1+\sqrt{5}}{2}\,,$$

respectively. Furthermore, subdividing a rectangle with aspect ratio $1/\sigma$ with the subdivision parameter $\sigma = (3 - \sqrt{5})/2$ leads to two rectangles with aspect ratios

$$\frac{\sigma}{\sigma} = 1 \quad \text{and} \quad \frac{1-\sigma}{\sigma} = \frac{1+\sqrt{5}}{2}\,,$$

respectively, and subdividing a rectangle with aspect ratio $1/(1 - \sigma)$ with the subdivision parameter $\sigma = (3 - \sqrt{5})/2$ leads to two rectangles with aspect ratios

$$\frac{1-\sigma}{\sigma} = \frac{1+\sqrt{5}}{2} \quad \text{and} \quad \frac{1-\sigma}{1-\sigma} = 1\,,$$

respectively. This completes the proof of the proposition. ∎

We will see in the next section that even though this choice might not seem that different from the choice $\mathcal{S} = \{1/2\}$, it suffices to eliminate grid alignment issues for all practical purposes. To close this section, we would further like to point out that one can easily adapt the proof of [7, section 3] to show that the adaptive rectangular grid furnished by the validation algorithm naturally gives rise to rectangular complexes which are homotopy equivalent to the nodal domains of the function. For more details, we refer the reader to [7].

4. Numerical case studies. In this final section of the paper we demonstrate that the Randomized Validation Algorithm (V) described in section 3 does indeed address the deficiencies of the method introduced in [7]. For this, we concentrate on two case studies. The first considers nodal domains generated by a standard double-well potential, close to the critical points of the potential. In addition, we present extensive simulations of the Cahn–Hilliard model for phase separation. This latter example makes use of trigonometric polynomial representations of the function u, and we therefore refrain from separately discussing trigonometric polynomials as in [7].

4.1. Double-well nodal domains. As a first test case we focus on a simple setting which allows us to get an idea of the basic performance parameters of the Randomized Validation Algorithm. For this, consider the inverted standard double-well potential

$$H_C(x,y) = \frac{2x^2 - x^4 - 2y^2}{4} + C \ ,$$

where C denotes a real parameter. One can easily see that for $C < -1/4$ the positive nodal domain of H_C is the empty set. For $-1/4 < C < 0$ the positive nodal domain consists of two open connected components, which form close to the global maxima $(\pm 1, 0)$ of the potential for $C \approx -1/4$, and which both approach the origin $(0,0)$ as $C \to 0^-$. Finally, for $C > 0$ the positive nodal domain is simply connected. Notice that in the critical cases $C = -1/4$ and $C = 0$, the zero set of H_C is singular: While in the former case it consists of two isolated points, in the latter case it is in the shape of a figure eight. It was mentioned in [7], that at either of these values the validation algorithm necessarily has to fail. It cannot rigorously resolve the topology of the nodal domains. As a consequence, one would expect the algorithm to encounter difficulties as C approaches these critical values.

In order to test the performance of the Randomized Validation Algorithm we consider rotated versions of the double-well potential as in [7]. At the time, this procedure was necessary to avoid alignment effects due to the original strict binary subdivision approach. For comparison purposes we still employ this setting here, even though it would be no longer necessary for the randomized algorithm. Thus, we consider rotated versions of the double-well potential around the point $r_c = (3\sqrt{3}/10,\ 2\sqrt{2}/5)$ with uniformly distributed random angles $\theta \in [0, 2\pi)$. In other words, we consider the positive nodal domain of the θ-dependent potentials

$$H_{C,\theta}(x,y) = H_C\left(5R_\theta^{-1}\left((x,y) - r_c\right)^t\right) \qquad \text{with} \quad R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} .$$

Notice that rather than using the original potential H_C, we use the scaled version $H_C(5\cdot)$ to ensure that the positive nodal domain is contained in the unit square. Some typical images of these nodal domains are shown in Figure 4.1, together with the grids obtained from our verification algorithm. The top row contains nodal sets for C-values close to the critical value $C \approx -1/4$, while the bottom row is for $C \approx 0$.

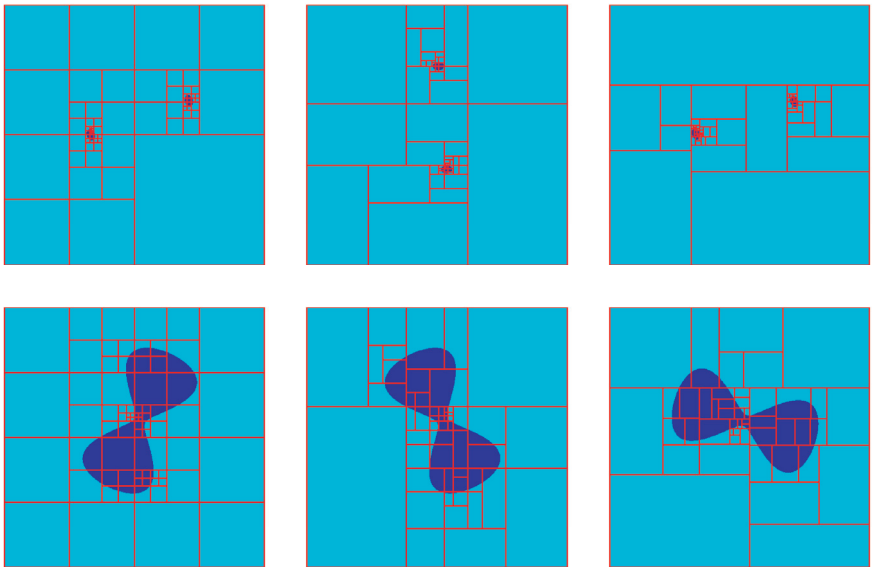

FIG. 4.1. *Sample nonuniform grids produced by the validation algorithm for the double-well potential. The top row is for* $C = -0.25 + 0.00625$*, while the bottom row is for* $C = 0.00625$*. From left to right, the columns correspond to using standard binary subdivisions, subdivisions created using the golden ratio, and random subdivision ratios taken uniformly from the interval* $\mathcal{S} = [0.3, 0.7]$.

For comparison reasons, we test the algorithm proposed in this paper with three different types of subdivision procedures:

(i) The first type uses the original binary subdivision technique of [7] in combination with the improved interval arithmetic methods, in order to establish their efficiency. Sample grids obtained with this method are shown in the left column of Figure 4.1;

(ii) the second type uses the golden ratio set $\mathcal{S}$ from Proposition 3.4; sample resulting grids are shown in the middle column of Figure 4.1;

(iii) the third and final type uses the subdivision interval $\mathcal{S} = [0.3, 0.7]$, and leads to grids as depicted in the rightmost column of Figure 4.1.

For each of these subdivision types, we performed 1000 simulations each for randomly selected angles $\theta \in [0, 2\pi)$ and C-values close to the two critical values. Specifically, we considered $C = c_0 + c_s\gamma$ with $\gamma = 2^{-k}/5$, for $k = 1, \ldots, 44$, and $(c_0, c_s) \in \{(-1/4, 1), (0, -1), (0, 1)\}$. The three choices for (c_0, c_s) are abbreviated by $C = -1/4^+$, $C = 0^-$, and $C = 0^+$ in the following. Finally, the three main input parameters of the algorithm described in (3.1) are chosen as $\varrho_{\text{mach}} = 10^{-14}$, $M_{\text{depth}} = 25$, as well as $f_\varrho \in \{0,\ 0.05,\ 0.1,\ 0.2,\ 0.5\}$. If the algorithm verifies the topology of the nodal domains, two key performance parameters are recorded: the total number of boxes in the final nonuniform adaptive grid, as well as the total number of interval evaluations which were necessary to obtain this grid.

For the first set of simulations, we consider the case $C = 0^+$, fix the parameter $f_\varrho = 0.1$, and consider the three subdivision types (i), (ii), and (iii) described above. In Figure 4.2, the dependence of the averaged two key performance parameters of the random verification algorithm on the absolute value $\gamma = |C - c_0|$ are depicted. The left panel shows the total number of boxes in the final grid, while the right panel contains the associated total number of interval evaluations. The blue, red, and green curves correspond to subdivision types (i), (ii), and (iii), respectively.

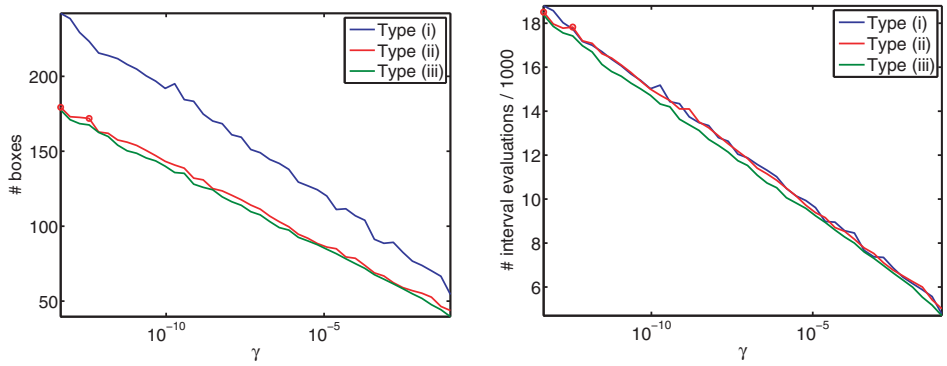

FIG. 4.2. *Dependence of averaged key performance parameters of our random verification algorithm on the absolute value* $\gamma = |C - c_0|$. *The two panels show the total number of boxes in the final grid, as well as the total number of interval evaluations for the case* $C = 0^+$ *and* $f_\varrho = 0.1$. *The blue, red, and green curves correspond to subdivision types* (i), (ii), *and* (iii), *respectively. Circled data points indicate that not all of the* 1000 *simulations for this* γ*-value led to successful validation.*

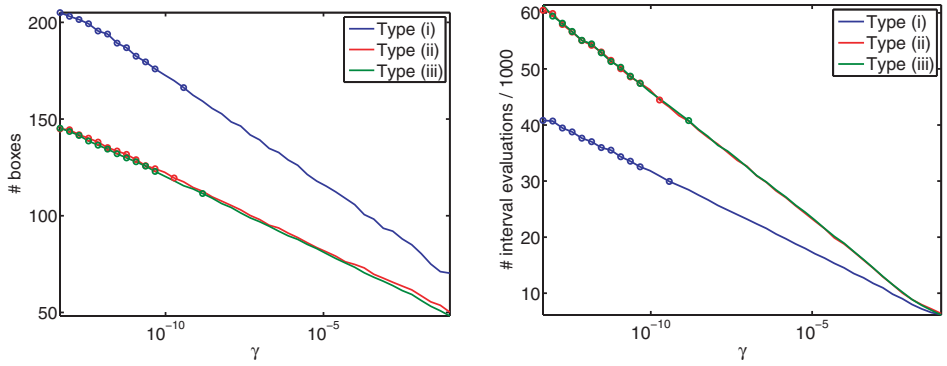

FIG. 4.3. *Dependence of averaged key performance parameters of our random verification algorithm on the absolute value* $\gamma = |C - c_0|$. *The two panels show the total number of boxes in the final grid, as well as the total number of interval evaluations for the case* $C = -1/4^+$ *and* $f_\varrho = 0.1$. *The blue, red, and green curves correspond to subdivision types* (i), (ii), *and* (iii), *respectively. Circled data points indicate that not all of the* 1000 *simulations for this* γ*-value led to successful validation.*

Circled data points indicate that not all of the 1000 simulations for the respective γ-value led to successful validation, and the average computation ignored these unsuccessful attempts. These simulations show that just as for the original algorithm of [7], both performance parameters depend linearly on the logarithm of γ and improve on its performance. In fact, considering the simplicity of the double-well nodal domains, the improvements of the randomized algorithm are substantial. These simulations also indicate that the specific form of the set $\mathcal{S}$ in the random subdivision types (ii) and (iii) has little impact on the algorithm performance. However, the randomized subdivisions on average lead to smaller grid sizes than the standard binary subdivisions, for the same computational effort. We will address this point in more detail below. While Figure 4.2 only considers the case $C = 0^+$, the results are almost identical for $C = 0^-$, and we therefore omit the corresponding graphs.

The situation is slightly different if we consider the case $C = -1/4^+$ shown in Figure 4.3, again for fixed parameter $f_\varrho = 0.1$ and the three subdivision types (i), (ii), and (iii). While the randomized algorithm maintains both the linear dependence of the

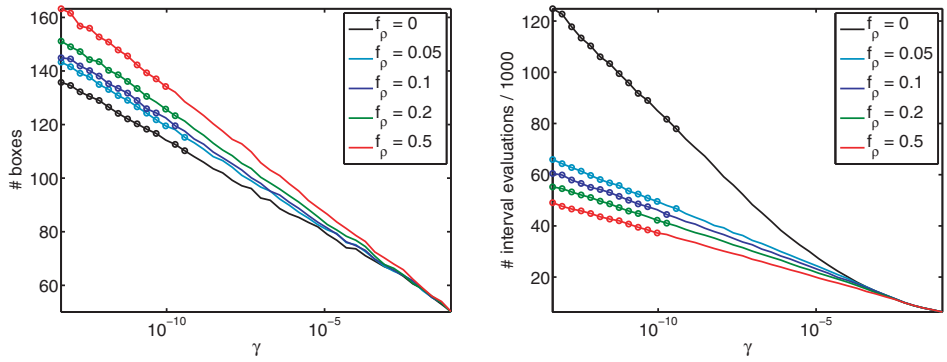

FIG. 4.4. *Dependence of averaged key performance parameters of our random verification algorithm on the absolute value* $\gamma = |C - c_0|$ *for the case* $C = -1/4^+$ *with type* (ii) *subdivisions, i.e., using the golden ratio. The two panels show the total number of boxes in the final grid, as well as the total number of interval evaluations for* $f_\varrho = 0, 0.05, 0.1, 0.2, 0.5$*, as indicated in the legend. Circled data points indicate that not all of the* 1000 *simulations for this* γ*-value led to successful validation.*

two performance parameters on $\log \gamma$ and the significant improvement over the method of [7], one now observes different behavior among the subdivision types. The random types (ii) and (iii) do lead to smaller final grids than the binary subdivision type (i), yet at a *higher* computational cost. It turns out that this behavior is an artifact of our subdivision strategies, which were chosen in order to facilitate comparison with the previous results. While (i) divides every square into four congruent subsquares, the random types (ii) and (iii) divide a rectangle into two subrectangles. This disparity is responsible for differences in behavior observed in Figure 4.3, as well as in the left panel of Figure 4.2. Nevertheless, in all of these cases the new randomized algorithm significantly outperforms the one of [7].

To close this section we briefly address the impact of the input parameter f_ϱ defined in (3.1) on the algorithm performance. For this, Figure 4.4 contains simulation results for the case $C = -1/4^+$ and type (ii) subdivisions. The two panels show the average total number of boxes in the final grid, as well as the average total number of interval evaluations for $f_\varrho = 0, 0.05, 0.1, 0.2, 0.5$. Notice that for the case $f_\varrho = 0$ (which was basically considered in [4]), the total number of interval evaluations no longer exhibits the characteristic linear dependence on $\log \gamma$, but rather quadratic growth. This choice of f_ϱ implies that ϱ in (3.2) is kept constant at ϱ_{mach}, regardless of the actual behavior of u on the considered box—leading to significant waste of computational effort. These unnecessary computations can be avoided by linking the threshold ϱ to the behavior of u on the considered box, as defined in (3.2) with $f_\varrho \in (0, 1)$, and for such values of f_ϱ we recover the linear dependence on $\log \gamma$. In fact, the simulations of Figure 4.4 indicate that as f_ϱ increases from 0, the computational effort decreases. Yet, since at the same time the size of the final grid increases, one has to find the right balance between computational effort and resulting grid size. For most purposes the choice $f_\varrho = 0.1$ leads to sufficiently small grids, while retaining a linear dependence on $\log \gamma$ which outperforms the algorithm in [7].

4.2. Pattern evolution in the Cahn–Hilliard model. As a second example we consider time-evolving patterns generated by a nonlinear partial differential equation. As in our previous paper [7] we consider spinodal decomposition patterns generated by the celebrated Cahn–Hilliard model [1, 2] and its stochastic extension

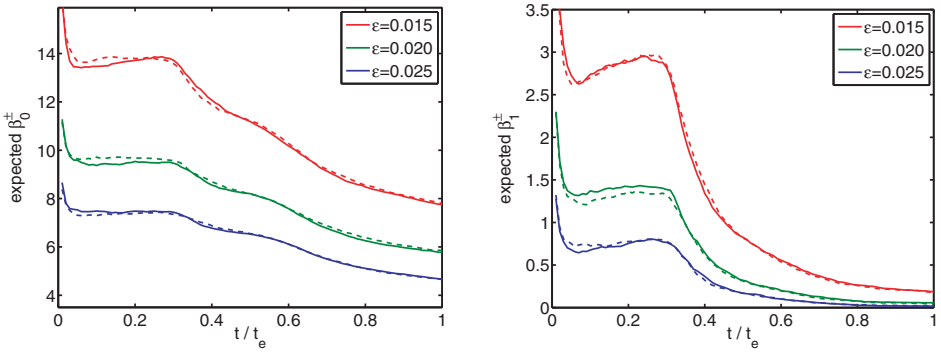

FIG. 4.5. *Betti number evolutions for the deterministic Cahn–Hilliard model* (4.1) *with* $\sigma = 0$*, and varying values of the interaction length* ε*. The left panel shows the averaged behavior of the zeroth Betti numbers* $\beta_0^{\pm}$ *of the nodal domains* $N^{\pm}(t)$*, while the right panel depicts the one-dimensional Betti numbers* $\beta_1^{\pm}$*. Averages are computed from* 1024 *simulations, solid lines indicate Betti numbers for* $N^{+}(t)$*, and dashed lines for* $N^{-}(t)$*.*

due to [5]. Both models are given by

$$\frac{\partial u}{\partial t} = -\Delta\left(\varepsilon^2 \Delta u + f(u)\right) + \sigma \cdot \xi \quad \text{for} \quad x \in \Omega \quad \text{and} \quad t \geq 0\,, \tag{4.1}$$

subject to no-flux boundary conditions for both u and Δu, where f is the negative derivative of a double-well potential, $\varepsilon > 0$ is a small parameter modeling interaction length, ξ denotes the generalized derivative of an infinite-dimensional Wiener process, and σ denotes the intensity of the noise. For $\sigma = 0$ the equation reduces to the classical Cahn–Hilliard model, for $\sigma \neq 0$ it is the stochastic Cahn–Hilliard–Cook equation.

The Cahn–Hilliard-Cook equation (4.1) determines the evolution of a phase variable $u(t, x)$ as a function of time t and a spatial variable x which describes the composition of the underlying material through its function values: Values of $u(t, x)$ close to $+1$ indicate that at time t and position x the material consists almost exclusively of the first material, values close to -1 correspond to the second material, and values in between represent mixtures of the two materials. Thus, values of u close to zero correspond to an equal mixture of the two involved materials. In general one is interested in the microstructures created through the phase separation process, and this naturally leads to the study of the time-evolving nodal domains (1.1). For the classical Cahn–Hilliard model in two dimensions, these nodal domains typically have the form shown in Figures 1.2 and 1.3. It has been demonstrated that their topology can provide important information on the microstructure evolution. In fact, in [17] it is shown that by studying the evolution of the Betti numbers of the nodal domains $N^{\pm}(t)$ one can uncover quantitative microstructure differences between the classical Cahn–Hilliard model and its stochastic extension, the Cahn–Hilliard–Cook model. Sample evolution curves determined with the Randomized Validation Algorithm can be found in Figure 4.5.

In order to study the performance of the Randomized Validation Algorithm when applied to the Cahn–Hilliard model (4.1), we consider the deterministic case $\sigma = 0$. For interaction lengths $\varepsilon = 0.015$, 0.020, and 0.025 we compute solutions originating at a small random perturbation of the homogeneous state $u \equiv 0$ over a time interval $[0, t_e]$, which includes both the spinodal decomposition and the early coarsening regime. The partial differential equation (4.1) is solved by a spectral Galerkin

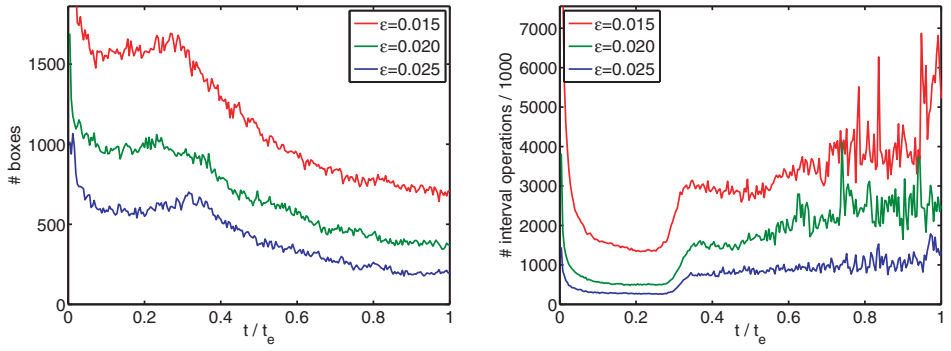

FIG. 4.6. *Evolution of the key performance parameters of the Randomized Validation Algorithm applied to the deterministic Cahn–Hilliard model* (4.1) *with* $\sigma = 0$. *For three different* ε*-values the figure shows the evolution of the total number of boxes in the final nonuniform grid (left panel) and of the total number of interval evaluations (right panel) needed to achieve the validation.*

method with linearly implicit time stepping and using the two-dimensional cosine basis $\cos(k\pi x)\cos(\ell\pi y)$ with $k, \ell = 0, \ldots, 31$, i.e., we use 1024 eigenfunctions in the solution expansion. The corresponding Fourier coefficients are then used to apply the Randomized Validation Algorithm with golden ratio subdivisions (ii) at times $k \cdot t_e/100$ for $k = 1, \ldots, 100$, and with parameters $f_\varrho = 0.1$, $M_{\text{depth}} = 12$, and $\varrho_{\text{mach}} = 10^{-9}$.

The resulting evolution of the two basic performance parameters for sample solution paths are shown in Figure 4.6. The left panel shows the temporal evolution of the total number of rectangles in the final validated grid, which seems to vary only mildly from one time step to the next, and generally follows the actual topology evolution shown in Figure 4.5. The computational effort as measured by the total number of interval evaluations is depicted in the right panel of Figure 4.6. While for early times the evolution curves again follow the topology dynamics, the effort sharply increases at around $t/t_e \approx 0.35$, and later on becomes fairly oscillatory with large excursions. This effect will be discussed in more detail below. Since we are interested in the typical behavior of the Randomized Validation Algorithm, we also determined the temporal evolution of averaged performance parameters, from 1024 runs for each of the three interaction lengths. These averaged curves are depicted in Figure 4.7, and they show that the sample path behavior is indeed indicative of a disparity between grid size and computational effort during coarsening. While the average total number of boxes in the final grid closely tracks the topology evolution, the computational effort is significantly increased for later evolution times.

The sudden increase in computational effort can be partly explained by studying the effects of the input parameter M_{depth} on the behavior of the algorithm. To this end, Figure 4.8 shows the averaged evolution of the performance parameters for interaction length $\varepsilon = 0.015$ and varying M_{depth}. Recall that the latter parameter specifies a maximal recursion depth within the routine which tests for positivity on a rectangle. A glance at the left panel which depicts the evolution of the average number of boxes in the final grid shows that as M_{depth} is increased from $M_{\text{depth}} = 6$, the grid size decreases significantly, and hardly changes anymore for $M_{\text{depth}} \geq 12$. The story is considerably different for the computational effort shown in the right panel of Figure 4.8. While for $t/t_e \leq 0.6$ the effort decreases with increasing M_{depth}, due to the fact that rectangles validated earlier are therefore removed from further consideration, for larger times the effort actually increases as M_{depth} changes from 12

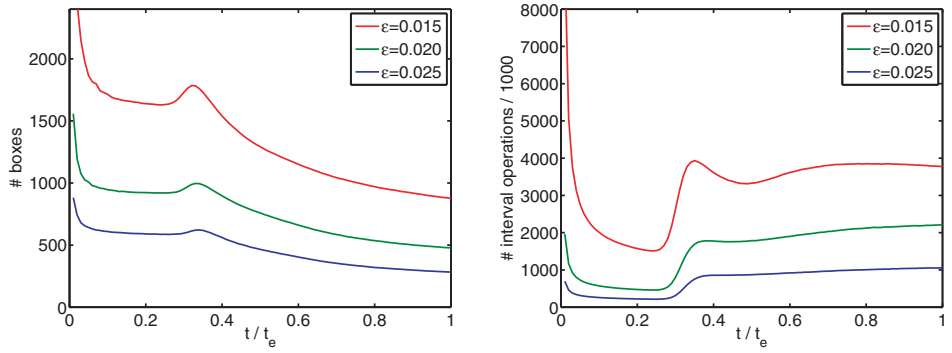

FIG. 4.7. *Evolution of averages of the key performance parameters of the Randomized Validation Algorithm applied to the deterministic Cahn–Hilliard model* (4.1) *with* $\sigma = 0$. *For three different* ε*-values the figure shows the evolution of the average total number of boxes in the final nonuniform grid (left panel) and of the average total number of interval evaluations (right panel) needed to achieve the validation. Averages were computed from* 1024 *independent runs in each case.*

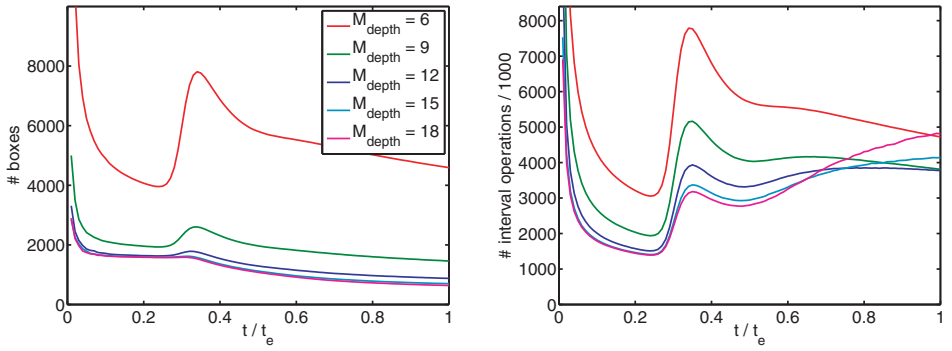

FIG. 4.8. *Evolution of averages of the key performance parameters of the Randomized Validation Algorithm applied to the deterministic Cahn–Hilliard model* (4.1) *with* $\sigma = 0$ *and* $\varepsilon = 0.015$. *For five different choices of the input parameter* M_{depth} *the figure shows the evolution of the average total number of boxes in the final nonuniform grid (left panel) and of the average total number of interval evaluations (right panel) needed to achieve the validation. Averages were computed from* 1024 *independent runs in each case, and the algorithm used golden ratio subdivisions.*

to 18. It seems that this increase is due to the particular structure of the solutions of (4.1) during the coarsening regime. Upon entering this regime, solutions to the Cahn–Hilliard equation have developed fairly sharp transition layers, whose width is on the order of ε. During coarsening, these layers start to move, and this motion can lead to high frequency Fourier modes with sizable coefficients. Such large coefficients, in combination with our use of interval arithmetic, lead to overestimations of the range of $v \in \{u, \pm u_x, \pm u_y\}$ during the validation step, i.e., the interval arithmetic range enclosures become less efficient. Since the large coefficients correspond to high frequency modes, one therefore has to significantly subdivide a rectangle before one can achieve validation. This fact is illustrated in some sense in the right panel of Figure 4.6. As the transition layers start to move, the number of interval evaluations exhibits localized spiking behavior. These spikes correspond exactly to the occurrence of large coefficients in high frequency modes. Consider now again Figure 4.8 which

shows the effects of changing M_{depth}. Recall that this parameter limits the maximal subdivision depths for our positivity verification algorithm for $v \in \{u, \pm u_x, \pm u_y\}$. As this parameter is increased, one initially sees both a decrease in the total number of boxes and the total number of interval evaluations, as expected. Yet, at a certain value of M_{depth}, any further increase leads to hardly any change in the number of boxes, but a significant increase in the number of interval evaluations. This is due to the fact that the positivity test algorithm creates additional subdivisions in an attempt to verify the positivity of v, while in fact this positivity cannot be verified numerically. Rather, the validation step has to subdivide the whole rectangle and do a recursion on the new subrectangles. In other words, in this regime, a lower value of M_{depth} leads to the earlier abandonment of a positivity validation effort, which is bound to fail anyway. For practical purposes, the choice $M_{\text{depth}} = 12$ seems to be a reasonable trade-off between final grid size and computational effort.

To close this section, we briefly comment on one of the main reasons for developing a randomized validation algorithm. As we mentioned in the introduction, the original algorithm of [7] suffered from grid alignment effects, which led to a large failure rate for the validation when using only binary subdivisions without further initial subdivisions. The randomized algorithm completely removes this deficiency. To show this, we specifically chose the fairly large input parameter $\varrho_{\text{mach}} = 10^{-9}$, so as to increase the possibility of validation failure. For each of the three ε-values mentioned above, and each of the five values of M_{depth} indicated in Figure 4.8, we computed 1024 solution paths, and along each solution path 100 patterns were validated. For $\varepsilon = 0.025$, validation was achieved during the first validation attempt in *all* cases. For $\varepsilon = 0.020$ and $\varepsilon = 0.015$ there were patterns which failed to validate in one attempt. In those cases, the algorithm was restarted, possibly several times. For $\varepsilon = 0.020$, every single pattern was validated in at most four attempts, and in fact 99.999219% of validations succeeded during the first attempt. Only for $\varepsilon = 0.015$ did we encounter patterns which did not validate in ten attempts, and these patterns accounted for 0.009765625% of all patterns. In other words, out of 512000 nodal domains only 50 did not validate after 10 runs of the algorithm. In fact, 99.973242% validated during the first attempt. These numbers show that for all practical purposes, randomizing the subdivision procedure leads to the elimination of grid alignment effects.

Acknowledgment. The authors would like to thank the anonymous referees for their careful reading of the manuscript and their helpful comments.

REFERENCES

[1] J. W. Cahn, *Free energy of a nonuniform system.* II. *Thermodynamic basis*, J. Chem. Phys., 30 (1959), pp. 1121–1124.

[2] J. W. Cahn and J. E. Hilliard, *Free energy of a nonuniform system* I. *Interfacial free energy*, J. Chem. Phys., 28 (1958), pp. 258–267.

[3] CHomP, *Computational Homology Project*, http://chomp.rutgers.edu, (2002–2010).

[4] G. S. Cochran, *Optimal Sampling of Random Fields for Topological Analysis*, Ph.D. thesis, George Mason University, Fairfax, VA, 2011.

[5] H. Cook, *Brownian motion in spinodal decomposition*, Acta Metallurgica, 18 (1970), pp. 297–306.

[6] S. Day, W. D. Kalies, K. Mischaikow, and T. Wanner, *Probabilistic and numerical validation of homology computations for nodal domains*, Electronic Res. Announc. Amer. Math. Soc., 13 (2007), pp. 60–73.

[7] S. Day, W. D. Kalies, and T. Wanner, *Verified homology computations for nodal domains*, Multiscale Model. Simul., 7 (2009), pp. 1695–1726.

[8] V. de Silva and G. Carlsson, *Topological estimation using witness complexes*, in Eurographics Symposium on Point-Based Graphics, M. Alexa and S. Rusinkiewicz, eds., The Eurographics Association, Aire-la-Ville, Switzerland, 2004.
[9] V. de Silva and R. Ghrist, *Coverage in sensor networks via persistent homology*, ATG Algebr. Geom. Topol., 7 (2007), pp. 339–358.
[10] V. de Silva and R. Ghrist, *Homological sensor networks*, Notices Amer. Math. Soc., 54 (2007), pp. 10–17.
[11] C. J. A. Delfinado and H. Edelsbrunner, *An incremental algorithm for Betti numbers of simplicial complexes on the* 3*-sphere*, Comput. Aided Geom. Design, 12 (1995), pp. 771–784.
[12] T. K. Dey, H. Edelsbrunner, and S. Guha, *Computational topology*, in Advances in Discrete and Computational Geometry, Contemp. Math. 223, AMS, Providence, RI, 1999, pp. 109–143.
[13] P. Dłotko, T. Kaczynski, M. Mrozek, and T. Wanner, *Coreduction homology algorithm for regular CW-complexes*, Discrete Comput. Geom., 46 (2011), pp. 361–388.
[14] H. Edelsbrunner and J. L. Harer, *Computational Topology*, AMS, Providence, RI, 2010.
[15] H. Edelsbrunner, D. Letscher, and A. Zomorodian, *Topological persistence and simplification*, Discrete Comput. Geom., 28 (2002), pp. 511–533.
[16] M. Gameiro, K. Mischaikow, and W. Kalies, *Topological characterization of spatial-temporal chaos*, Phys. Rev. E (3), 70 (2004), 035203.
[17] M. Gameiro, K. Mischaikow, and T. Wanner, *Evolution of pattern complexity in the Cahn-Hilliard theory of phase separation*, Acta Materialia, 53 (2005), pp. 693–704.
[18] R. Ghrist, *Barcodes: The persistent topology of data*, Bull. Amer. Math. Soc. (N. S.), 45 (2008), pp. 61–75.
[19] T. Kaczynski, K. Mischaikow, and M. Mrozek, *Computational Homology*, Appl. Math. Sci. 157, Springer-Verlag, New York, 2004.
[20] K. Krishan, M. Gameiro, K. Mischaikow, M. Schatz, H. Kurtuldu, and S. Madruga, *Homology and symmetry breaking in Rayleigh-Benard convection: Experiments and simulations*, Phys. Fluids, 19 (2007), 117105.
[21] K. Mischaikow and T. Wanner, *Probabilistic validation of homology computations for nodal domains*, Ann. Appl. Probab., 17 (2007), pp. 980–1018.
[22] K. Mischaikow and T. Wanner, *Topology-guided sampling of nonhomogeneous random processes*, Ann. Appl. Probab., 20 (2010), pp. 1068–1097.
[23] R. E. Moore, *Methods and Applications of Interval Analysis*, SIAM Studies Appl. Math. 2, SIAM, Philadelphia, 1979.
[24] R. E. Moore, R. Baker Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*, SIAM, Philadelphia, 2009.
[25] M. Mrozek and B. Batko, *Coreduction homology algorithm*, Discrete Comput. Geom., 41 (2009), pp. 96–118.
[26] P. Niyogi, S. Smale, and S. Weinberger, *Finding the homology of submanifolds with high confidence from random samples*, Discrete Comput. Geom., 39 (2008), pp. 419–441.
[27] V. Robins, P. J. Wood, and A. P. Sheppard, *Theory and algorithms for constructing discrete morse complexes from grayscale digital images*, IEEE Trans. Pattern Anal. Machine Intelligence, 33 (2011), pp. 1646–1658.
[28] S. Skelboe, *Computation of rational interval functions*, BIT, 14 (1974), pp. 87–95.
[29] T. Wanner, E. R. Fuller Jr., and D. M. Saylor, *Homological characterization of microstructure response fields in polycrystals*, Acta Materialia, 58 (2010), pp. 102–110.
[30] A. Zomorodian and G. Carlsson, *Computing persistent homology*, Discrete Comput. Geom., 33 (2005), pp. 249–274.