A Local Radial Basis Function method for Advection-Diffusion-Reaction equations on complexly shaped domains

Scott A. Sarra
Marshall University

March 20, 2012

Abstract

Time-dependent advection-diffusion-reaction and diffusion-reaction equations are used as models in biology, chemistry, physics, and engineering. As representative examples, we focus on a chemotaxis model and a Turing system from biology and apply a local radial basis function method to numerically approximate the solutions. The numerical method can efficiently approximate large scale problems in complexly shaped domains.

keywords: Numerical Partial Differential Equations, Advection-Diffusion-Reaction equations, Diffusion-Reaction equations, Chemotaxis, Turing equations, Radial Basis Functions.

1 Introduction

Systems of advection-reaction-diffusion and reaction-diffusion equations model a large number of physical phenomena in various scientific disciplines. These areas include chemical reactions, population dynamics, flame propagation, and the evolution of concentrations in environmental and biological processes. In this work we focus on bacterial chemotaxis and on pattern formation in a Turing system as representative examples.

Chemotaxis is defined as the movement of a cell or organism in a direction corresponding to a gradient of increasing or decreasing concentration of a particular substance. In particular, a property of many bacteria is that in the presence of certain chemicals, called chemoattractants, they move preferentially towards higher concentration of the chemical [29]. Much of the modeling of chemotaxis is based on the original Keller-Segel chemotaxis model [22] and variations of the original model. Modified versions of the original Keller-Segel model are explored in [17].

Numerical simulations for chemotaxis models appearing in the literature have been typically carried out on rectangular domains. This geometry is chosen either to simplify the development of the numerical method or possibly because the particular numerical method is only applicable on such simple domains. Previously used numerical methods for chemotaxis problems on rectangular domains include a discontinuous Galerkin method [11] and various finite difference and finite volume methods [8, 7, 43, 49, 50]. In [6], a finite volume scheme is developed and applied to chemotaxis problems on circular domains. The method involves mapping a cartesian grid from a square domain to a circle. The method allows for numerical simulations to be performed which more closely resemble the results of experiments carried out in a circular petri dish.

Turing first described how a system of coupled reaction-diffusion equations could give rise to spatial patterns in chemical concentrations through a process of chemical instability [48]. Since that time, Turing systems have been used to model complex spatial patterns that are found in nature [1]. The applications include pattern formation in biological systems [29], for example patterns on fish [2] and butterflies [39]. Previously used numerical methods for Turing systems include various finite difference [2, 10, 51], finite element [25], and finite volume methods [6].

We note that other, more classical numerical methods, such as the Discontinuous Galerkin Method (DGM) and the Finite Element Method (FEM), may also be implemented in complexly shaped domains. However, like most classical numerical methods, the DGM and FEM both require a mesh (or grid) to be implemented on. In two dimensions, the meshes usually consist of elements with triangular or quadrilateral shapes. In higher dimensions, the boundaries of the elements become (possibly curved) surfaces, and the interiors become volumes instead of areas. Creating such meshes is a very sophisticated process and generating the mesh represents a significant overhead in the implementation of the methods. After the mesh is generated

the quality of the mesh must be assessed, as an "inferior" (reference [14] addresses mesh quality) mesh may adversely affect the accuracy of the method. In general, mesh generating software tend to be complex codes that are inaccessible to the average user. Thus, the user must give up control, and the software is used as a "black box" [31]. The RBF method does not require that a mesh be generated in a (potentially complex and computationally expensive) preprocessing stage. It is for this reason that the RBF method is referred to as a *meshfree* or *meshless* method.

The main features of the local RBF method that we describe are that it is simple, generally applicable, and efficient. The method places no restriction on the shape of the domain. Thus, the shape of the domain can be dictated by the particular application and not by the limitations of the numerical method. The method may be efficiently implemented with thousands of points in large domains.

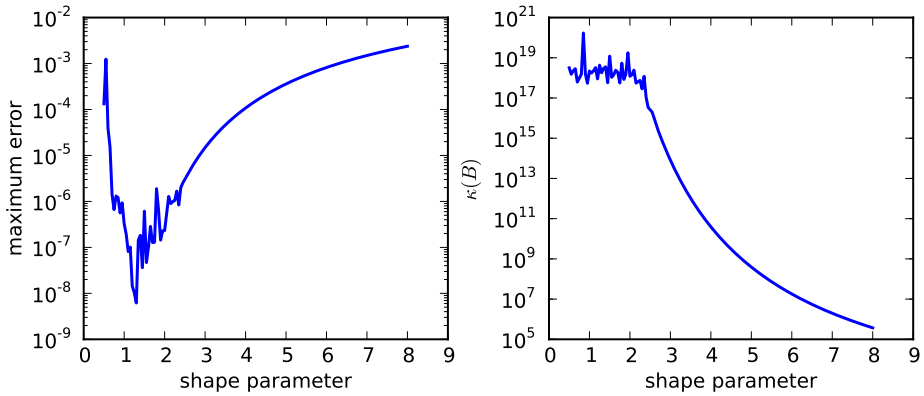

Figure 1: An illustration of the RBF uncertainty principle - the attainable error and the condition number of the system matrix cannot both be kept small. Left: accuracy versus the shape parameter. Right: condition number of the system matrix versus the shape parameter.

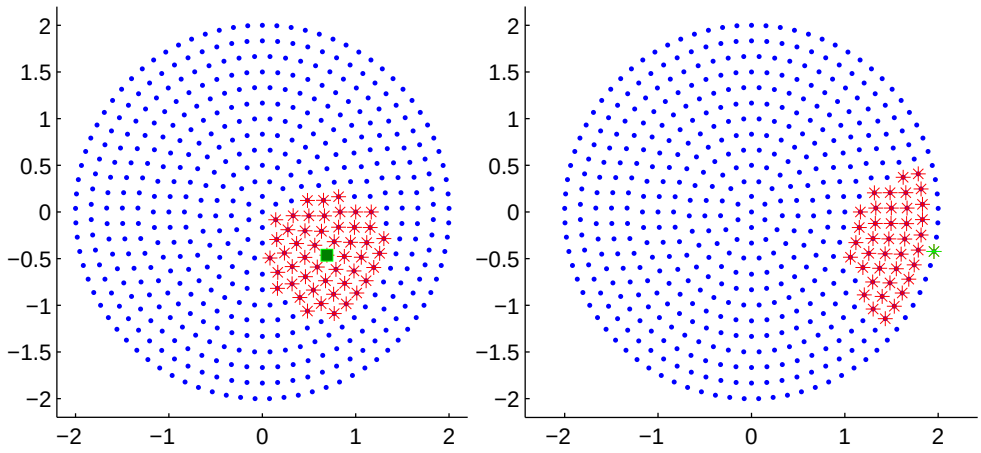

Figure 2: Example center distribution with $N = 500$ and stencils with $n = 50$. Left: An interior stencil with the base center marked with a square and supporting centers with an asterisk. Right: A boundary stencil for enforcing no flux boundary conditions. The stencil is based at the one center located on the boundary.

2 A local RBF method for time-dependent PDEs

In this section we describe a local RBF method for time-dependent PDEs. More detailed information on RBF methods in general can be found in recent research monographs on RBF methods [4, 12, 37, 52]. Variations of the local RBF methods for the solution of steady and time-dependent PDEs were first used in references [45, 46] and [42]. Recent applications of local RBF methods include [18, 40, 41, 34].

Each problem is discretized with a set of N distinct points $X = \{x_1^c, \ldots, x_N^c\}$ in $\mathcal{R}^d$ called centers. No restrictions are placed on the shape of problem domains or on the location of the centers. At each of the N centers, the local RBF method considers a local interpolant of the form

$$\mathcal{I}_n f(x) = \sum_{k \in I_i} a_k \phi\left(\|x - x_k^c\|_2, \varepsilon_i\right) \tag{1}$$

where a is a vector of expansion coefficients, ϕ is a RBF, and I_i is a vector associated with center i that contains the center number and the indices

of the n-1 nearest neighboring centers. Each center and its n-1 neighbors are called a stencil. A typical stencil is shown in the left image of figure 2. Enforcing the interpolation conditions

$$\mathcal{I}_n f(x_k) = f_k, \quad k \in I_i \tag{2}$$

on each stencil gives N, $n \times n$ linear systems

$$Ba = f \tag{3}$$

to be solved for the expansion coefficients. The matrix B is called the interpolation matrix or the system matrix. The local system matrices have elements

$$b_{jk} = \phi(\left\|x_j^c - x_k^c\right\|_2, \varepsilon_i), \qquad j,k = I_i(1), \ldots, I_i(n). \tag{4}$$

Our choice of RBF is the multiquadric (MQ)

$$\phi(r,\varepsilon) = \sqrt{1+\varepsilon^2 r^2} \tag{5}$$

which is popular in applications [20, 21] and is representative of the class of global, infinitely differentiable RBFs containing a free parameter called the shape parameter. The MQ system matrix B is guaranteed to be invertible [28], so the expansion coefficients are uniquely defined on each stencil.

For a fixed set of centers, the shape parameter affects both the accuracy of the method and the conditioning of the system matrix. A typical result from interpolating a smooth function is illustrated in figure 1. A condition number is used to quantify the sensitivity to perturbations of a linear system and to estimate the accuracy of a computed solution [9, 47]. Using the 2 norm, the matrix condition number is

$$\kappa(B) = \|B\|_2 \left\|B^{-1}\right\|_2 = \frac{\sigma_{\max}}{\sigma_{\min}} \tag{6}$$

where σ are the singular values of B. The RBF method is most accurate for smaller values of the shape parameter where the system matrix is ill-conditioned. The attainable error and the condition number of the system matrix cannot both be kept small. This relationship has been dubbed the uncertainty principle [38].

Algorithms have been recently developed that bypass solving the potentially ill-conditioned linear system (3). For instance, the RBF-QR algorithm [15, 13] evaluates a RBF interpolant (using the Gaussian RBF

$\phi(r,\varepsilon) = e^{-\varepsilon^2 r^2}$) in a stable manner for all values of the shape parameter. However, the RBF-QR approach has un-optimized parameters that the user must specify and brings with it a significant amount of computational cost [13]. At its current level of development, while promising, the RBF-QR method does not seem ready for serious applications. In reference [36] it is suggested, and examples are given, to illustrate that it may be more efficient to implement the RBF method in a more accurate floating point number system rather than to resort to methods that bypass solving the linear system (3).

To approximate derivatives of a function f at the center locations, a linear differential operator $\mathcal{L}$ is applied to (1) and it is evaluated at the center on which the stencil is based to get

$$\mathcal{L}f(x_i) = \sum_{k \in I_i} a_k \mathcal{L}\phi\left(\|x_i^c - x_k^c\|_2, \varepsilon_i\right). \tag{7}$$

Equation (7) can be written more concisely as a dot product

$$\mathcal{L}f(x_i) = h \cdot a \tag{8}$$

where a is the $n \times 1$ vector of RBF expansion coefficients and h is a $1 \times n$ vector containing the elements

$$h_i = \mathcal{L}\phi\left(\|x_i^c - x_k^c\|_2, \varepsilon_i\right), \qquad k \in I_i. \tag{9}$$

The dependence on the RBF expansion coefficients can be removed from (8) by noting that

$$\mathcal{L}f(x_i) = hB^{-1}f(I_i) = (hB^{-1})f(I_i) = w \cdot f(I_i) \tag{10}$$

where the stencil weights are

$$w = hB^{-1}. \tag{11}$$

That is, the weights are the solution of the linear system

$$wB = h. \tag{12}$$

Thus, space derivatives are approximated simply by multiplying the function values at the centers of the stencil by the weights.

Both the condition number (6) and the solution of the linear system (12) are calculated from the singular value decomposition (SVD) of the matrix B. The SVD of a $N \times N$ matrix B is $B = U\Sigma V^T$ where U and V are $N \times N$ orthogonal matrices and Σ is a $N \times N$ diagonal matrix with the N singular values of B as its elements [9, 47]. A property of orthogonal matrices is that the matrix inverse is simply the transpose of the matrix. Therefore, the solution of (12) is $w = h(V\Sigma^{-1}U^T)$.

A different value of the shape parameter may be used on each stencil. An effective way to select the shape parameter is based on the fact that RBF methods are most accurate when their system matrix is ill-conditioned. When implementing the method on computers that use double precision floating point arithmetic [30], the shape parameter can be specified on each stencil so that $10^{13} \leq \kappa(B) \leq 10^{15}$. If the error is considered as a function of the shape parameter, in most cases this results in a shape parameter being used that corresponds to a point on the error curve just before the curve begins to oscillate as is the case in figure 1. We note that the condition number range will be different if floating point numbers systems other than double precision are employed [36].

The shape parameter selection at each center is described in the following pseudocode:

```
kappa = 0
while kappa < kappaMin and kappa > kappaMax:
   form B
   U, S, V = svd(B)
   kappa = max(S)/min(S)
   if kappa <  kappaMin:
      shape = shape - shapeIncrement
   elseif kappa >  kappaMax:
      shape = shape + shapeIncrement
```

which adjusts the value of the shape parameter until the system matrix has a condition number in the desired range. When an acceptable matrix B is found, its SVD is then used to solve equation (12) for the stencil weights. The setup cost involves calculating the SVD of $N + R$, $n \times n$ linear systems, where R is the number of rejections of the shape parameter choice in the above loop. After setup, the local method approximates a space derivative with N dot products of length n vectors. The number of rejections R is discussed later in the numerical examples.

n	3	5	8	10	15	20	25	100
error	5.4e-3	3.7e-5	8.2e-7	8.1e-7	6.8e-7	9.3e-7	8.2e-7	6.7e-7

Table 1: Maximum error from differentiating the function $f(x) = \exp(\sin(\pi x))$ with $N = 100$ versus stencil size.

The optimal choice of the stencil size n is largely problem dependent, but some generalizations can be made and are illustrated by the following example. The function $f(x) = \exp(\sin(\pi x))$ is differentiated on $N = 100$ evenly spaced centers on the interval $[-1, 1]$. The results are summarized in table 1. The local method with a stencil size as small as $n = 8$ is about as accurate as the global method with $n = N$. The local method with $n \ll N$ is often just as accurate as the global method. Since the derivative is a local property, intuitively it seems reasonable that it can be accurately approximated without using information from the entire domain. The local method can have accuracy comparable to the global method while also enjoying both considerably smaller storage requirements and a smaller flop count than the global method. In two and three dimensions, computational evidence indicates that a stencil size in the range $20 \leq n \leq 100$ is adequate in most cases. If the solution is relatively smooth, $n \approx 20$ is usually sufficient. Larger n will be required if the solution is rapidly varying and/or features small detailed structure. The global RBF method theoretically converges at a spectral or exponential rate as the shape parameter and the minimum separation distance between centers are refined [24]. However, as the shape parameter and the minimum separation distance between centers are refined the condition number of the system matrix grows and the limitations of floating point arithmetic prevent the convergence trend from continuing beyond a certain point. At this point, the local RBF method is usually able to match the accuracy of the global method with some small stencil size.

After the PDE is discretized in space with the RBF method, the resulting system of ODEs

$$u_t = F(u)$$

is advanced in time with an ODE method, which is often called a method of lines approach. We have used a fourth-order Runge-Kutta method [5] in all the numerical examples. It is well known [32, 35] that RBF methods for time-dependent PDEs that are purely advective may have differentiation

n	error	density	time
400	3.4e-5	3.3	48.1
200	3.4e-5	1.3	25.0
100	4.0e-5	0.67	13.2
60	3.4e-5	0.4	8.4
40	3.4e-5	0.27	5.7
20	2.9e-5	0.13	2.6
12	3.0e-4	0.067	2.1

Table 2: The results of solving equation (13) with the local RBF method with various stencil sizes.

matrices that have eigenvalues will positive real parts which make stable time integration impossible. However, this is not the case for PDEs containing diffusive terms and RBF methods for advection-diffusion problems can be stably advanced in time with a suitable choice of time step size.

2.1 A motivating example

First we consider a two-dimensional advection-diffusion-reaction equation

$$u_t = \nu\left(u_{xx} + u_{yy}\right) - \alpha\left(u_x + u_y\right) + \gamma u^2(1-u). \tag{13}$$

with a known analytical solution in order to gauge accuracy versus the stencil size for this type of problem. The exact solution is

$$u(x,y,t) = \left[1 + \exp(a\left[x + y - bt\right] + c)\right]^{-1} \tag{14}$$

where $a = \sqrt{\gamma/4\nu}$, $b = 2\alpha + \sqrt{\gamma\nu}$, and $c = a(b-1)$. The initial conditions and Dirichlet boundary conditions are specified using the exact solution. This problem has been previously used as a test problem in [19]. The domain for the problem is a circular region of radius ten that is centered at the point $(5,5)$. The domain is discretized with $N = 15,000$ evenly spaced centers. The local RBF method is used with several different stencils sizes to advance the solution to time $t = 5$ with a time step size of $\Delta t = 0.005$. The results are summarized in table 2 which lists the stencil size (n), the maximum error at time $t = 5$, the percentage of non-zero elements (density) of the $N \times N$ differentiation matrix, and the execution time in seconds. Implementing the

method with $n = 20$ is just as accurate as the method with $n = 400$. In fact, the accuracy results are very similar over the stencil size range of 20 to 400. Of course with smaller stencils, are associated faster execution times as well as reduced memory requirements. The local method with $n = 20$ requires only about one-tenth of one percent of the computer memory than does the global method. We note that it is not possible to implement the global method with $n = 15,000$ on typical desktop computers due to excessive memory requirements.

2.2 No flux boundary conditions

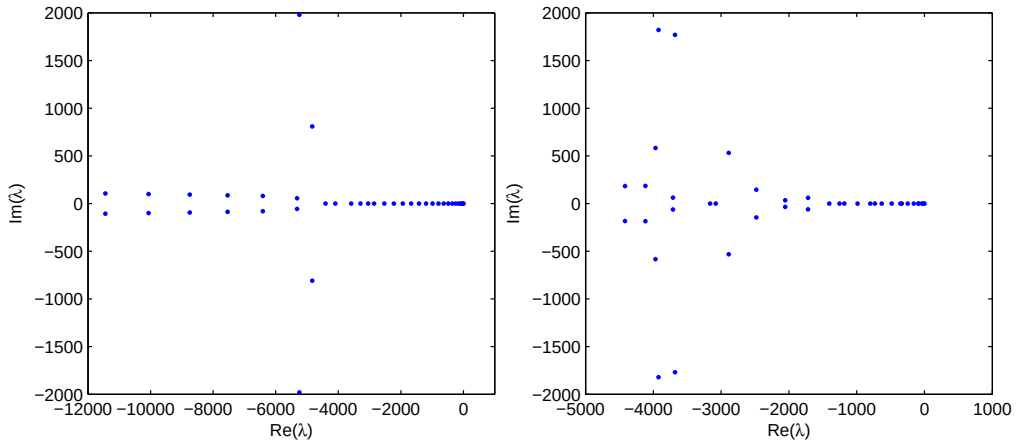

Figure 3: Eigenvalues of the differentiation matrix of the PDE (17). Left: global RBF method. Right: local RBF method with $n = 8$.

Many PDE problems in application areas such mathematical biology have zero Nuemann or no flux boundary conditions of the form

$$\nabla u \cdot \mathbf{n} = 0 \tag{15}$$

where $\mathbf{n}$ is a unit normal vector. A literature search only revealed one [42] implementation of a RBF method for a time-dependent PDE with Neumann boundary conditions. In [42], the authors arranged three layers of orthogonal grids near and including the boundaries and then used a polynomial based second-order finite difference scheme to find the value of the PDE solution on the boundary. Of course this approach requires the use of very

simply shaped domains such as rectangles and can not be applied in complexly shaped domains. Thus, we describe the implementation of Neumann boundary conditions in a more general setting.

As an example we consider a two dimensional problem with $\mathbf{n} = \langle n^1, n^2 \rangle$. The value of u on the boundary can be found from the Neumann boundary condition (15) as follows. Let w^x and w^y be the stencil weights that respectively discretize the first derivatives with respect to x and y on a stencil that only includes one boundary point (center number i) such as the stencil in the right image of figure 2. Then, the boundary value is specified as

$$u_i = \frac{1}{n_i^1 w_1^x + n_i^2 w_1^y} \left(-n_i^1 \left[w_2^x + \cdots w_n^x \right] - n_i^2 \left[w_2^y + \cdots w_n^y \right] \right) \tag{16}$$

which is found by solving for u_i in the discretization of equation (15).

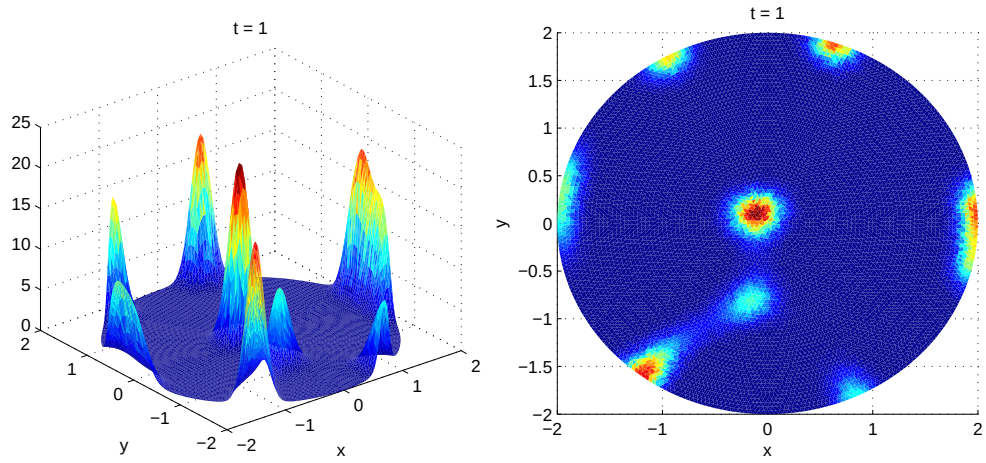

Figure 4: Cell density solution from the system (21) at $t = 1$. Left: surface plot. Right: density plot.

The enforcement of boundary conditions is important in numerical methods for PDEs as it affects both the accuracy and the stability of the method as it is advanced in time. To gain insight about the effect of applying a no flux boundary condition in a RBF method we consider the advection-diffusion equation

$$u_t + u_x = u_{xx} \tag{17}$$

on the interval $[0, 1]$ with no flux boundary conditions applied at both ends of the interval. The exact solution of the problem is $u(x, t) = \exp(-(n^2\pi^2 +$

$\frac{1}{4})t + \frac{x}{2})(-4\pi\cos(2\pi x) + \sin(2\pi x))$. The problem is discretized with $N = 40$ equally spaced centers. While equation (16) can be used to find the value of u on the boundary, it is of no use in examining eigenvalue stability. To examine stability, the boundary conditions need to be incorporated into the differentiation matrix of the PDE. This can be done for the advection-diffusion equation (17) as follows. Let D_1 and D_2 respectively discretize first and second order derivatives and d_{ij} be the elements of D_1.

Since $u_{xx} = D_1 D_1 u$ and $u_x(x_1) = u_x(x_N) = 0$

$$\begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}_{xx} = D_1 \begin{bmatrix} 0 \\ u_2 \\ \vdots \\ u_{N-1} \\ 0 \end{bmatrix}_x = \begin{bmatrix} d_{0,2} & \cdots & d_{0,N-1} \\ \vdots & \ddots & \vdots \\ d_{N,2} & \cdots & d_{N,N-1} \end{bmatrix} \begin{bmatrix} u_2 \\ \vdots \\ u_{N-1} \end{bmatrix}_x \tag{18}$$

where

$$\begin{bmatrix} u_2 \\ \vdots \\ u_{N-1} \end{bmatrix}_x = \begin{bmatrix} d_{2,0} & \cdots & d_{2,N} \\ \vdots & \ddots & \vdots \\ d_{N-1,0} & \cdots & d_{N-1,N} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}. \tag{19}$$

The $N \times N$ matrix

$$\bar{D}_2 = \begin{bmatrix} d_{0,1} & \cdots & d_{0,N-1} \\ \vdots & \ddots & \vdots \\ d_{N,1} & \cdots & d_{N,N-1} \end{bmatrix} \begin{bmatrix} d_{1,0} & \cdots & d_{1,N} \\ \vdots & \ddots & \vdots \\ d_{N-1,0} & \cdots & d_{N-1,N} \end{bmatrix} \tag{20}$$

discretizes the second derivative at the interior centers and also enforces zero Neumann boundary conditions at the endpoints. Now let $\bar{D}_1$ be the first order differentiation matrix that has been modified to account for the no flux boundary conditions by setting its first and last row to zero. Then the matrix $D = \bar{D}_2 - \bar{D}_1$ discretizes the space derivatives of the PDE and also enforces the boundary conditions.

The problem is discretized with $N = 40$ equally spaced centers and both the global and local RBF methods are applied with the solution being advanced in time to $T = 0.05$. The global method with shape parameter $\varepsilon = 4.3$ and the local method with a stencil size of $n = 8$ and $\varepsilon = 2.2$ both have a system matrix with a $\mathcal{O}(10^{14})$ condition number. The eigenvalues of both differentiation matrices, shown in figure 3, all have non-positive real

parts and thus may been stably advanced in time with explicit time stepping methods. Notice however, that for stability, the global method will require a time-step approximately 2.5 times smaller than the local method. The maximum error from the global method is 9.8e-4 while the local method has a slightly smaller maximum error of 3.8e-4.

3 Numerical Examples

All examples have been advanced in time with a fourth order Runge-Kutta method with a time step size of $\Delta t = 0.005$. The weights for all stencils were computed with system matrices that had condition numbers in the range $10^{13} \leq \kappa(B) \leq 10^{15}$.

3.1 example 1

The example is based on a biological experiment described in [3]. In the experiment, patterns were formed by Escherichia coli and Salmonella typhimurium when they were placed in a liquid medium and exposed to intermediates of the tricarboxylic acid cycle. The bacteria group together in large masses to form patterns and then randomly rearrange as time evolves. A mathematical model for the experiment was proposed in [49]. The model consists of a system of two nonlinear PDEs: an advection-diffusion equation for the cell density coupled with a reaction-diffusion equation for the chemoattractant concentration. The model is

$$\begin{aligned} u_t &= \frac{1}{3}\nabla^2 u - 80\,\nabla \cdot \left[\frac{u}{(1+v)^2}\nabla v\right] \\ v_t &= \nabla^2 v + \frac{u^2}{1+u^2} \end{aligned} \tag{21}$$

were u is the cell density and v represents the chemoattractant concentration. The no flux boundary conditions

$$\begin{aligned} \nabla u \cdot \mathbf{n} &= 0 \\ \nabla v \cdot \mathbf{n} &= 0 \end{aligned} \tag{22}$$

are enforced. The initial conditions are $v = 0$ for the chemoattractant and the initial values of the cell densities u are set to be uniformly distributed random numbers between 0.95 and 1.05.

The model has been evaluated numerically on square domains in [49] and [8]. Since the RBF method is applicable on domains of any shape, we take the domain to be a circle of radius 2 and uniformly distribute $N = 4000$ centers over the domain in a pattern such as in figure 2. A local stencil of size $n = 100$ is used. To indicate the extreme sparsity of the local method with these settings, the non-zero elements of the differentiation matrices are plotted in figure 5. Only 2.5 percent of the elements of the matrix are non-zero.

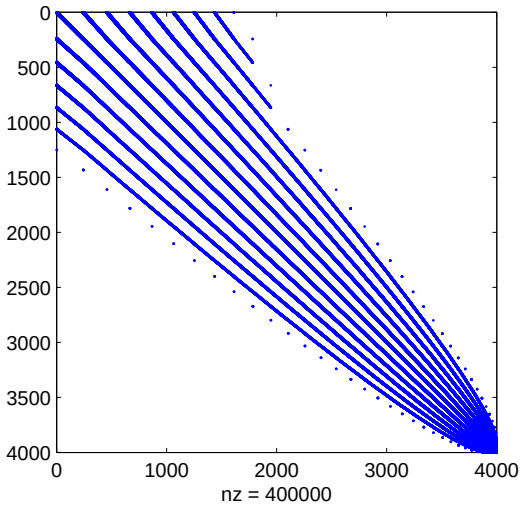

Figure 5: The non-zero elements of a differentiation matrix from the local RBF method with $N = 4000$ and $n = 100$.

The cell density at $t = 1$ is shown in figure 4. The local RBF method solution evolves with the same qualitative features as those in [49] and [8], regardless of the differently shaped domain and of course, different random initial conditions.

3.2 example 2

Next we consider the chemotaxis model

$$\begin{aligned} u_t &= \frac{1}{4}\nabla^2 u - \alpha\nabla\cdot\left[\frac{u}{(1+v)^2}\nabla v\right] + \frac{1}{100}u\,(20-u) \qquad (23)\\ v_t &= \nabla^2 v + \frac{1}{5}u^2 - uv \end{aligned}$$

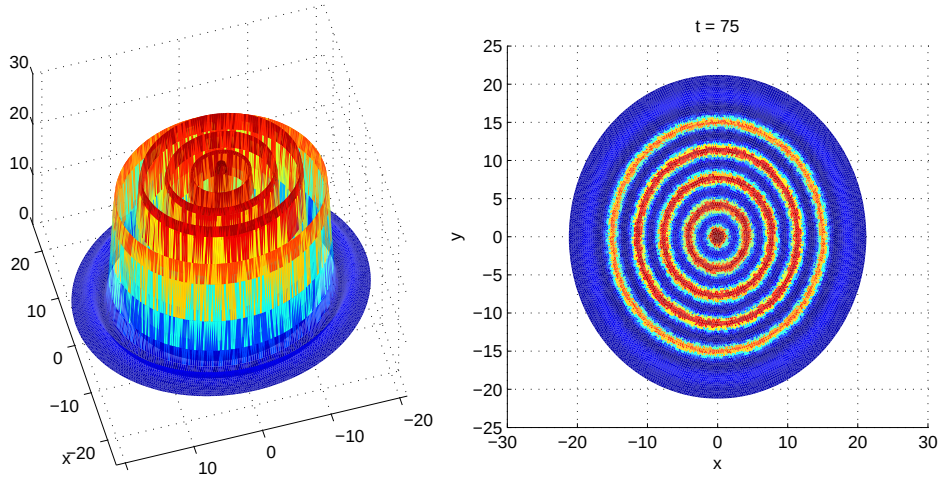

Figure 6: The cell density solution from the system (23) with $\alpha = 2.25$. The initial concentration evolves into a set of continuous, concentric rings. Left: surface plot. Right: density plot.

that is described in [29]. The domain is taken to be a circle of radius $15\sqrt{2}$ so that we may compare to results in [6] where the problem was solved with a finite volume scheme on a grid that was mapped from a square to a circle. No flux boundary conditions are applied and the initial conditions are taken to be $v = 0$ and

$$u(x, y, 0) = \begin{cases} 1 & \text{if } \sqrt{x^2 + y^2} \leq \frac{1}{2} \\ 0 & \text{otherwise.} \end{cases}$$

The parameter α is set to $\alpha = 2.25$ for which the correct qualitative behavior of the cell density solution is for the initial condition to develop into continuous, concentric rings. $N = 15{,}000$ uniformly distributed centers are used (50% less than in [6]) and a local stencil size of $n = 100$. The solution is advanced to $t = 75$ and the cell concentration is shown in figure 6.

For $\alpha = 5$, the correct qualitative behavior of the cell density solution is for the initial condition to develop into concentric spotted rings. With this value of α and the same initial condition, N, and n as the continuous rings example, the solution is advanced to $t = 75$. The results are shown in figure 7. Again, the correct qualitative behavior of the solution is observed.

In the setup phase, only $R = 2$ SVD are rejected. This small number is due to the uniform center distribution. The condition number of the system matrix is affected both by the value of the shape parameter and the smallest

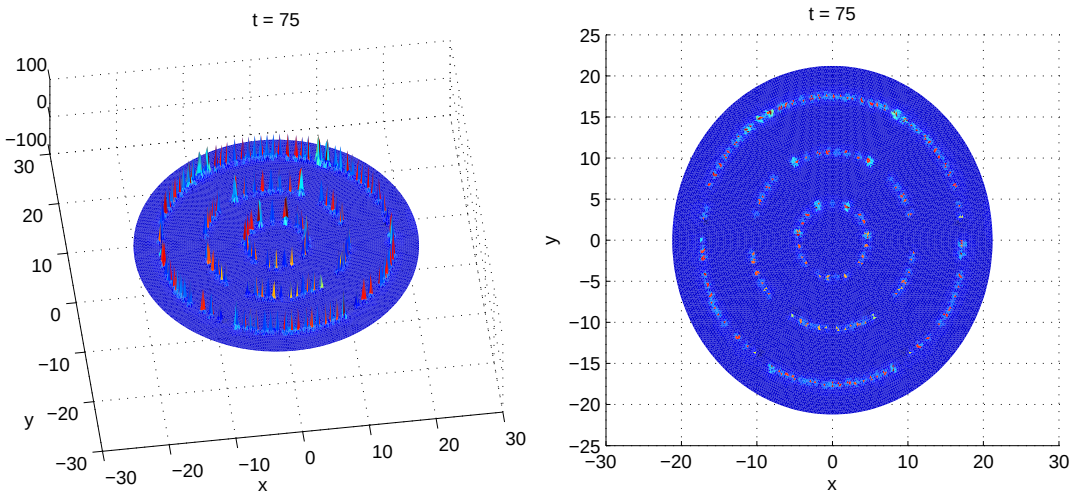

Figure 7: The cell density solution from the system (23) with $\alpha = 5$. The initial concentration evolves into a set of continuous, spotted rings. Left: surface plot. Right: density plot.

distance between any two centers on the stencil. Thus with uniform centers, after the shape is selected for the first stencil, the same shape parameter can be used with every other center in order to get a system matrix with a condition number in the desirable range.

3.3 example 3

The next example again considers the model (23) with $\alpha = 2.25$, but this time on a circular domain of radius 20 which is centered at the origin and has two holes removed. The first hole is a circle of radius 2 centered at $(4, 7)$ and the second hole is a circle of radius 1 centered at $(-3, 3)$. We note that on such a domain that the method described in [6] is not applicable. The domain is shown in figure 8. The initial bacterial concentration is zero everywhere except inside a circle of radius 0.5 that is centered at $(-5, 5)$ where the bacteria concentration is set to one. The chemoattractant is initially zero throughout the domain. Once again, zero flux boundary conditions are applied and $N = 15,000$ scattered centers are distributed throughout the domain as follows. The outer boundary is discretized with 600 evenly distributed centers, the radius two circle and the radius one circle are discretized

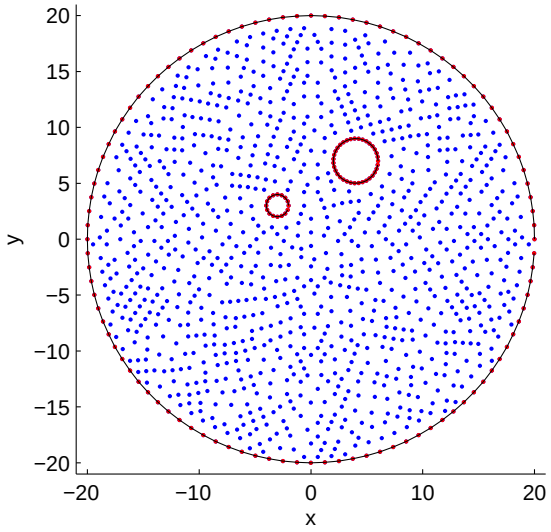

Figure 8: Domain and an example center distribution for example 3.

respectively with 60 and 30 evenly spaced centers. The remaining centers are located throughout the interior of the domain by adding centers one by one to the middle of the largest hole in the centers that have already been added to the domain. This methodology ensures that the centers cover the domain in a fairly uniform manner and that no two centers are close together relative to the spacing of the centers as a whole. The algorithm is described in detail in [27]. An example center distribution is illustrated in figure 8. In the setup phase, $R = 1493$ SVD are rejected which corresponds to one shape parameter adjustment being made at approximately 10% of the centers in the domain. This is due to the smallest distance between any two centers on a stencil not being the same for every stencil as was the case in the previous example.

The cell density solution is advanced to time $t = 100$ and the solution at several times is shown in figure 9. At $t = 64$, the continuity of one of the outer rings is broken as it runs into the two holes in the domain. By $t = 75$ the gaps in the ring are starting to close and the ring is again becoming continuous. This trend continues until the end of the simulation at $t = 100$.

3.4 example 4 - Turing equations

In [48], a reaction-diffusion model of the form

$$\begin{aligned} u_t &= D\,\nabla^2 u + f(u,v) \\ v_t &= \nabla^2 v + g(u,v) \end{aligned} \tag{24}$$

for two interacting chemicals with periodic boundary conditions was first considered. In the equations $u(x,y,t)$ and $v(x,y,t)$ represent chemical concentrations, D is the ratio of the diffusion coefficients of the two chemicals, and the nonlinear reaction terms f and g model the chemical kinetics. It well-known that for certain parameter settings that the equations have steady-state solutions in which u and v evolve into spatial patterns featuring objects such as dots and stripes [29]. After their introduction, Turing systems were later studied with no-flux boundary conditions and were used in biology to model pattern forming phenomena and in ecology where u and v become species densities [44]. Subsequently the equations were solved with mixed and Dirichlet boundary conditions and modified spatial patterns were observed [26]. The spatial patterns are affected by both the shape of the domain and by the type of boundary condition that is applied.

As a particular example we consider the Turing system

$$\begin{aligned} u_t &= D\,\delta\nabla^2 u + \alpha u(1-\tau_1 v^2) + v(1-\tau_2 u) \\ v_t &= \delta\nabla^2 v + \beta v\left(1+\frac{\alpha\tau_1}{\beta}uv\right) + u(\alpha+\tau_2 v) \end{aligned} \tag{25}$$

on a "butterfly" shaped domain (shown in the left image of figure 10) that has a boundary given by the equation

$$r(\theta) = 3e^{\sin\theta}\sin^2(2\theta) + 3e^{\cos\theta}\cos^2(2\theta), \qquad 0 \le \theta \le 2\pi. \tag{26}$$

The following parameters were used: $\delta = 0.0045$, $D = 0.516$, $\tau_1 = 0.02$, $\tau_2 = 0.2$, $\alpha = 0.899$, $\beta = -0.91$, and $\gamma = -\alpha$. References [2] and [51] can be consulted for an explanation of the choice of parameters. In references [2] and [51] the particular parameter choices were shown to produce spotted spatial patterns on rectangles and spheres.

The domain is discretized with 8125 total centers of which 425 are located on the boundary and zero Dirichlet boundary conditions are applied to both u and v. The initial conditions for both u and v are uniformly distributed

random numbers between -0.5 and 0.5. A local stencil of size $n = 100$ is used and the solution is advanced to time $t = 120$ at which a steady-state solution has been reached. The steady-state solution of u, in the right image of figure 10, features the expected spotted spatial pattern.

4 Conclusions

RBF methods have steadily gained in popularity in recent years due to their simplicity, ease of implementation, and flexibility. The method is essentially the same for three dimensional problems as it is for one dimensional problems due to its dependence on the distance between centers, and not the location of the centers. This leads to relatively simple computer code for high dimensional problems. Due to the complete freedom to locate centers, the method places no restriction on the shape of problem domains. Thus, the method is not restricted to rectangular domains and the shape of the domain can instead be dictated by the application.

For large problems requiring tens or hundreds of thousands of discretization points, it is not feasible to implement the global RBF method due to the dense structure of the associated matrices. Fortunately, local RBF methods with relatively small stencil sizes can produce solutions with accuracy comparable to the global method, but in a more efficient manner and without adding the complications of domain decomposition techniques which often must be resorted to in large scale problems. In the numerical examples, we reported results using a stencil size of $n = 100$. However, the solutions were visually indistinguishable when using a stencil size in the range $20 \leq n \leq 100$. Additionally, we have described a way to select the shape parameter on each stencil that is based on the condition number of the local system matrix.

Chemotaxis models and Turing systems have been used as examples, but the local RBF method is equally well applicable to other processes that are governed by advection-reaction-diffusion or reaction-diffusion type equations. As a template for such applications, the Matlab source code for example 2 is available at `http://www.scottsarra.org/math/math.html`.

The local RBF method is well suited to modern heterogeneous computer architectures that feature both CPU and graphical processing units (GPUs). GPUs have become massively parallel devices for implementing floating point operations. An example of a GPU accelerated computation is in reference [16] where a 45 to 75 time speedup was realized when compared to a CPU

alone approach. Others examples include a speedup factor of 85 in [23] and a 34-fold speedup in [33]. In the setup stage of the local RBF method, the calculation of the N sets of stencil weights could be distributed between computational cores. In a similar manner, the dot products required to approximate space derivatives at each center could be distributed among cores. Our current focus is on implementing such an approach that would allow problems in complexly shaped domains in three or more dimensions to be efficiently solved.

References

[1] P. Ball. *The self-made tapestry: pattern formation in nature*. Oxford University Press, 2001. 1

[2] R. Barrio, C. Varea, J. Aragon, and P. Maini. A two-dimensional numerical study of spatial pattern formation in interacting Turing systems. *Bulletin of Mathematical Biology*, 61:483–505, 1999. 1, 3.4

[3] E. O. Budrene and H. C. Berg. Complex patterns formed by motile cels of escherichia coli. *Nature*, 349:630–633, 1991. 3.1

[4] M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2003. 2

[5] J. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, 2003. 2

[6] D. Calhoun and C. Helzel. A finite-volume method for solving parabolic equations on logically cartesian curved surface meshes. *SIAM Journal on Scientific Computing*, 31(6):4066–4099, 2009. 1, 3.2, 3.3

[7] A. Chertock and A. Kurganov. A second-order positivity preserving central-upwind scheme for chemotaxis and haptotaxis models. *Numerische Mathematik*, 111:169–205, 2008. 1

[8] C. Chiu and J.-L. Yu. An optimal adaptive time-stepping scheme for solving reaction-diffusion-chemotaxis systems. *Mathematical Biosciences and Engineering*, 4(2):187–203, 2007. 1, 3.1, 3.1

[9] B. N. Datta. *Numerical Linear Algebra and Applications*. SIAM, second edition, 2010. 2, 2

[10] R. Dillon, P. Maini, and H. Othmer. Pattern formation in generalized Turing systems I: steady-state patterns in systems with mixed boundary conditions. *Journal of Mathematical Biology*, 32:345–393, 1994. 1

[11] Y. Epshteyn and A. Kurganov. New interior penalty discontinuous Galerkin methods for the Keller-Segel chemotaxis model. *SIAM Journal on Numerical Analysis*, 47:386–408, 2008. 1

[12] G. E. Fasshauer. *Meshfree Approximation Methods with Matlab*. World Scientific, 2007. 2

[13] G. E. Fasshauer and M. McCourt. Stable evaluation of Gaussian RBF interpolants. *Submitted for publication*, 2011. 2

[14] D. Field. Qualitative measures for initial meshes. *International Journal for Numerical Methods in Engineering*, 47:887906, 2000. 1

[15] B. Fornberg, E. Larsson, and N. Flyer. Stable computations with Gaussian radial basis functions in 2-d. *SIAM Journal of Scientific Computing*, 33:869–892, 2011. 2

[16] X. Gu, X. Jia, and S. Jiang. GPU-based fast gamma index calculation. *Physics in Medicine and Biology*, 56(5), 2011. 4

[17] T. Hillen and K. Painter. A user's guide to PDE models for chemotaxis. *Journal of Mathematical Biology*, 58:183–217, 2009. 1

[18] B. Hosseini and R. Hashemi. Solution of Burgers' equation using a local-RBF meshless method. *International Journal for Computational Methods in Engineering Science and Mechanics*, 12(1):44–58, 2011. 2

[19] W. Hundsdorfer. Accuracy and stability of splitting with stabilizing corrections. *Applied Numerical Mathematics*, 42:213–233, 2002. 2.1

[20] E. J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics I: Surface approximations and partial derivative estimates. *Computers and Mathematics with Applications*, 19(8/9):127–145, 1990. 2

[21] E. J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics II: Solutions to parabolic, hyperbolic, and elliptic partial differential equations. *Computers and Mathematics with Applications*, 19(8/9):147–161, 1990. 2

[22] E. F. Keller and L. A. Segel. Model for chemotaxis. *Journal of Theoretical Biology*, 30:225–234, 1971. 1

[23] G. Klingbeil, R. Erban, M. Giles, and P. K. Maini. STOCHSIMGPU: parallel stochastic simulation for the systems biology toolbox 2 for Matlab. *Bioinformatics*, 27(8):1170–1171, 2011. 4

[24] W. R. Madych and S. A. Nelson. Bounds on multivariate interpolation and exponential error estimates for multiquadric interpolation. *Journal of Approximation Theory*, 70:94–114, 1992. 2

[25] A. Madzvamuse, A. Wathen, and P. Maini. A moving grid finite element method applied to a model biological pattern generator. *Journal of Computational Physics*, 190:478–500, 2003. 1

[26] P. Maini and M. Myerscough. Boundary-driven instability. *Applied Mathematics Letters*, 10:1–4, 1997. 3.4

[27] S. De Marchi, R. Schaback, and H. Wendland. Near-optimal data-independent point locations for radial basis function interpolation. *Advances in Computational Mathematics*, pages 1–14, 2004. 3.3

[28] C. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986. 2

[29] J. Murray. *Mathematical Biology II: Spatial Models and Biomedial Applications*. Springer, 2003. 1, 3.2, 3.4

[30] M. Overton. *Numerical Computing with IEEE Floating Point Arithmetic*. SIAM, 2001. 2

[31] P. Persson and G. Strang. A simple mesh generator in Matlab. *SIAM Review*, 46:329–345, 2002. 1

[32] R. Platte and T. Driscoll. Eigenvalue stability of radial basis functions discretizations for time-dependent problems. *Computers and Mathematics with Applications*, 51:1251–1268, 2006. 2

[33] D. Rossinelli, B. Hejazialhosseini, D. G. Spampinato, and P. Koumoutsakos. Multicore/multi-GPU accelerated simulations of multiphase compressible flows using wavelet adapted grids. *SIAM Journal on Scientific Computing*, 33(2):512–540, 2011. 4

[34] Y. Sanyasiraju and G. Chandhini. A RBF based local gridfree scheme for unsteady convection-diffusion problems. *CFD Letters*, 1(2):59–67, 2009. 2

[35] S. A. Sarra. A numerical study of the accuracy and stability of symmetric and asymmetric RBF collocation methods for hyperbolic PDEs. *Numerical Methods for Partial Differential Equations*, 24(2):670 – 686, 2008. 2

[36] S. A. Sarra. Radial basis function approximation methods with extended precision floating point arithmetic. *Engineering Analysis with Boundary Elements*, 35(1):68–76, 2011. 2, 2

[37] S. A. Sarra and E. J. Kansa. Multiquadric radial basis function approximation methods for the numerical solution of partial differential equations. *Advances in Computational Mechanics*, 2, 2009. 2

[38] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Advances in Computational Mathematics*, 3:251–264, 1995. 2

[39] T. Sekimura, A. Madzvamuse, A. Wathen, and P. Maini. A model for colour pattern formation in the buttery wing of Papilio Dardanus. *Proceedings of the Royal Society London B*, 267(1):851–859, 2000. 1

[40] Y. Shan, C. Shu, and N. Qin. Multiquadric finite difference (MQ-FD) method and its application. *Advances in Applied Mathematics and Mechanics*, 5(1):615–638, 2009. 2

[41] Q. Shen. Numerical solution of the Sturm-Liouville problem with local rbf-based differential quadrature collocation method. *International Journal of Computer Mathematics*, 88(2):285–295, 2011. 2

[42] C. Shu, H. Ding, and K.S. Yeo. Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier-Stokes equations. *Computer methods in applied mechanics and engineering*, 192:941–954, 2003. 2, 2.2

[43] M. Smiely. An efficient implementation of a numerical method for a chemotaxis system. *International Journal of Computer Mathematics*, 86:219–235, 2009. 1

[44] C. Taubes. *Modeling Differential Equations in Biology*. Campbridge University Press, 2008. 3.4

[45] A. I. Tolstykh, M. V. Lipavskii, and D. A. Shirobokov. High-accuracy discretization methods for solid mechanics. *Archives of Mechanics*, 55:531–553, 2003. 2

[46] A. I. Tolstykh and D. A. Shirobokov. On using radial basis functions in finite difference mode with applications to elasticity problems. *Computational Mechanics*, 33:68–79, 2003. 2

[47] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, first edition, 1997. 2, 2

[48] A. Turing. The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society: B*, 237:37–72, 1952. 1, 3.4

[49] R. Tyson, S. Lubkin, and J. Murray. Model and analysis of chemotactic bacterial patterns in a liquid medium. *Journal of Mathematical Biology*, 38:359–375, 1999. 1, 3.1, 3.1, 3.1

[50] R. Tyson, L. Stern, and R. LeVeque. Fraction step methods applied to a chemotaxis model. *Journal of Mathematical Biology*, 41:455–475, 2000. 1

[51] C. Varea, J. Aragon, and R. Barriio. Turing patterns on a sphere. *Physical Review E*, 60(4):4588–4592, 1999. 1, 3.4

[52] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005. 2

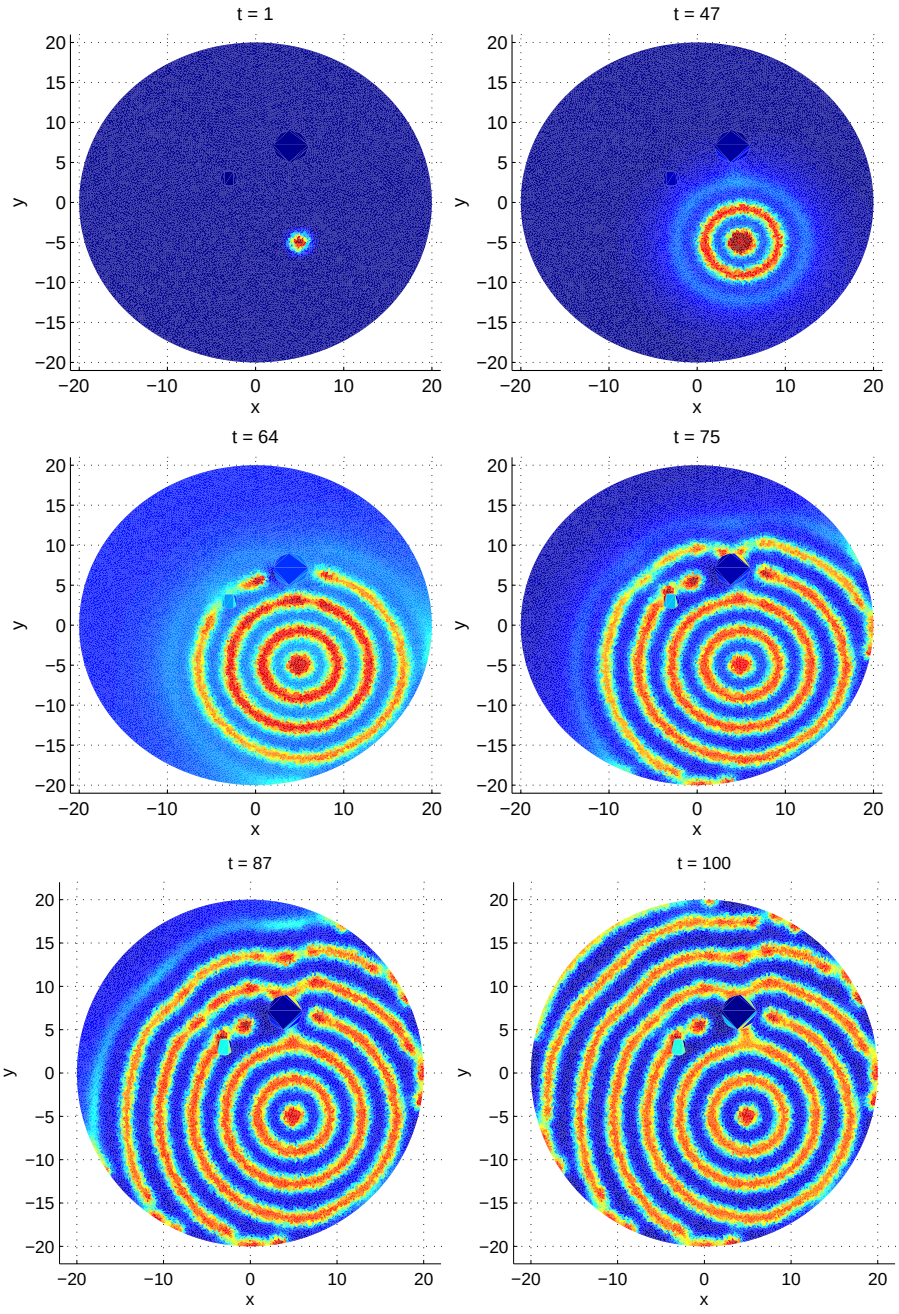

Figure 9: Cell density solution of the system (23) with $\alpha = 2.25$ on the domain shown in figure 8.

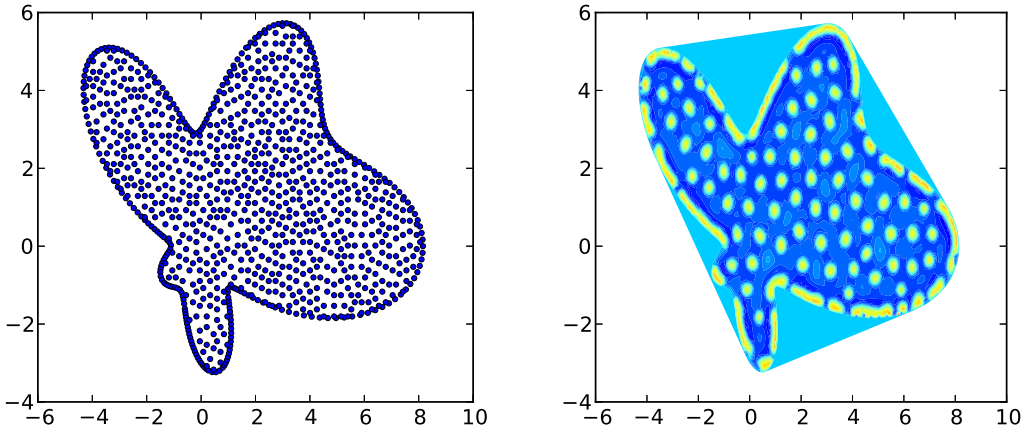

Figure 10: Left: An example discretization of the domain outlined by equation (26) with 1020 scattered centers. Right: Spotted spatial pattern in the steady-state solution of the Turing system (25).