A domain decomposition algorithm for computing multiple steady states of differential equations

Wenrui Hao* Jonathan D. Hauenstein† Bei Hu‡
Andrew J. Sommese§

January 19, 2011

Abstract

We present a new technique for solving polynomial systems derived from the discretization of differential equations based on domain decomposition and homotopy continuation. This method divides the domain into subdomains and solves the polynomial system arising from each subdomain. Homotopy continuation then uses these solutions to build solutions for the original domain. We illustrate the method on both one-dimensional and two-dimensional space problems.

Keywords: Domain decomposition, homotopy continuation, differential equations, multiple solutions.
AMS Subject Classification: 65N55, 65M22,65H10,65L10.

1 Introduction

Discretization of systems of differential equations often lead to systems of polynomials. Though using modern numerical codes for the complete solu-

*Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556 (whao@nd.edu). This author was supported by the Duncan Chair of the University of Notre Dame and NSF grant DMS-0712910.

†Department of Mathematics, Mailstop 3368, Texas A&M University, College Station, TX 77843 (jhauenst@math.tamu.edu, www.math.tamu.edu/∼jhauenst). This author was supported by Texas A&M University and NSF grant DMS-0915211.

‡Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556 (b1hu@nd.edu, www.nd.edu/∼b1hu).

§Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556 (sommese@nd.edu, www.nd.edu/∼sommese). This author was supported by the Duncan Chair of the University of Notre Dame and NSF grant DMS-0712910.

tion of polynomial systems can yield new solutions [5], the systems of polynomials (arising even from very sparse grids) are usually much too large for direct solution by these codes. The realization underlying this article is that domain decomposition gives excellent guidance on how to "bootstrap" from the solutions of many small systems of polynomials to often large numbers of solutions of a system of polynomials arising from a discretization with a realistic grid.

Domain decomposition is a powerful tool for devising parallel methods to solve partial differential equations. The basic idea of domain decomposition to decompose the domain into subdomains, and solve on all of the subdomains independently in parallel. Since computing approximate values to the subdomain boundary points is a major difficulty, there is a rich literature (see, for example, [4, 7, 9] and the references therein), which constructs schemes to approximate them for time dependent systems of PDEs. These parallel schemes are mainly focused on parabolic equations.

This article introduces a new domain decomposition algorithm which we call the *bootstrapping method*. This method computes multiple steady states of time dependent systems and is naturally parallelizable. We will introduce the bootstrapping method for solving problems consisting of one and two space dimensions in Sections 2 and 4, respectively. We provide numerical examples and examine the algorithm in Sections 3 and 5. Section 6 summarizes the results.

2 One dimensional bootstrapping

We introduce the one dimensional bootstrapping method using Laplace's equation. Suppose that $u(x)$ is a solution to

$$\begin{cases} u_{xx} &= f(u) \quad \text{on } (0,1), \\ u(0) &= u_0, \\ u(1) &= u_1. \end{cases} \tag{2.1}$$

We decompose the interval $[0,1]$ with grid points $x_i = iH$ for $i = 0, \ldots, N$ where $H = \frac{1}{N}$ and N is a positive integer. Each subinterval $[x_i, x_{i+1}]$ is then decomposed with grid points $x_{i,j} = x_i + jh$ for $j = 0, \ldots, M$ where $h = \frac{1}{NM}$ and M is a positive integer. The points $x_{i,0} = x_i$ and $x_{i,M} = x_{i+1}$ are the boundary points of the subintervals while $x_0 = x_{0,0} = 0$ and $x_N = x_{N-1,M} = 1$ are the boundary points of the original domain $[0,1]$. The other points are called interior points.

Define $u_{i,j} = u(x_{i,j})$. The goal is to compute a numerical approximation $U_{i,j}$ of $u_{i,j}$ where these approximations form a solution to a polynomial

system constructed by discretizing (2.1). To define and compute solutions, we need the following two discretized operators

$$\partial_i^2 U = \frac{U_{i+1,0} - 2U_{i,0} + U_{i-1,0}}{H^2}$$

and

$$\partial_{i,j}^2 U = \begin{cases} \dfrac{U_{i,1} - 2U_{i,0} + U_{i-1,M-1}}{h^2} & \text{if } j = 0 \\ \dfrac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{h^2} & \text{if } j = 1, \ldots, M-1. \end{cases}$$

There are four discretized polynomial systems that we will consider. The first and second polynomial systems, denoted F_H and F_h, are the polynomial systems that arise from the grid points in $[0, 1]$ which are separated by H and h, respectively. In particular,

$$F_H(U) = \begin{cases} \partial_i^2 U - f(U_{i,0}) & \text{for } i = 1, \ldots, N-1 \\ U_{0,0} - u_0 \\ U_{N,0} - u_1. \end{cases} \tag{2.2}$$

and, if $X = \{(i, j) \mid i = 0, \ldots, N-1 \text{ and } j = 0, \ldots, M-1\} \setminus \{(0, 0)\}$, then

$$F_h(U) = \begin{cases} \partial_{i,j}^2 U - f(U_{i,j}) & \text{for } (i, j) \in X \\ U_{0,0} - u_0 \\ U_{N,0} - u_1. \end{cases} \tag{2.3}$$

The third polynomial system, denoted $F_{i,h}$, is the polynomial system that arises from the domain $[x_i, x_{i+1}]$ where $U_{i,0}$ and $U_{i+1,0}$ are fixed, namely

$$F_{i,h}(U) = \partial_{i,j}^2 U - f(U_{i,j}) \quad \text{for } j = 1, \ldots, M-1. \tag{2.4}$$

The fourth polynomial system, denoted $F_{h,H}$, is the polynomial system that arises based on domain decomposition, namely

$$F_{h,H}(U) = \begin{cases} \partial_i^2 U - f(U_{i,0}) & \text{for } i = 1, \ldots, N-1 \\ \partial_{i,j}^2 U - f(U_{i,j}) & \text{for } i = 0, \ldots, N-1, j = 1, \ldots, M-1 \\ U_{0,0} - u_0 \\ U_{N,0} - u_1. \end{cases} \tag{2.5}$$

The basic idea of the bootstrapping method is to solve F_H for small values of $N = \frac{1}{H}$ and use these solutions to solve $F_{h,H}$. Solutions to F_h are computed using the homotopy

$$F(U, t) = (1 - t)F_h(U) + \gamma t F_{h,H}(U), \tag{2.6}$$

where γ is a random complex number (see [8] for more details).

We now describe the bootstrapping method in detail.

1. For small N, F_H defined by (2.2) can be solved directly. This computes discretized solutions $U_{i,0}$ on the coarse grid defined by $H = \frac{1}{N}$. We denote the set of solutions as $S_{N,0}$.

2. Each solution in $S_{N,0}$ defines Dirichlet "boundary conditions" on each subdomain. Solve $F_{i,h}$ for $i = 0, \ldots, M-1$ and let the set of solution be $S_{N,M}$. Typically, M is taken to be small, say $M = 2$ or $M = 3$.

3. Discard unreasonable solutions of $S_{N,M}$ using a filtering condition. Our filtering condition ensures that the approximation solution is reasonably close to the real solution. Some filtering conditions are following:

   (a) **Variational form:** Let $v \in V_h$ be the set of continuous piecewise linear functions that vanish at $x = 0$ and $x = 1$. Then, the solutions satisfy the form

   $$-\int_0^1 u'v'dx = \int_0^1 f(u)vdx \text{ for all functions v,}$$

   corresponding to the residual $u'' - f(u)$ being orthogonal to the test function space V_h. We can pick a sequence $\{v_n\}_{n=1}^{\infty}$, such as $v_n = \frac{\sin(n\pi x)}{\sqrt{n}}$ or $v_n = \frac{x^n(1-x)}{\sqrt{n}}$, to obtain a series of filtering conditions. (Such a filtering approach is related to finite element methods). We use

   $$|\int_0^1 u'v'dx + \int_0^1 f(u)vdx| \leq \epsilon \text{ for } \|v\|_{H^1([0,1])} \leq 1,$$

   with a reasonably small ϵ.

   (b) **Bounded condition:** A filter forcing $\|U_{i,j}\| < K$, where $K > 0$ is a upper bound of the PDE solutions. Clearly, a bounded filter can be applied to PDE systems for which bounds can be estimated.

   (c) **Other conditions:** Other filters can be found in [1], which is derived from known properties of the solutions of the problem at hand, such as derivatives, symmetry, positive or some other easily-detected behavior.

4. By construction, the points in $S_{N,M}$ are solutions of $F_{h,H}$ which are start points for the homotopy F defined by (2.6) at $t = 1$. Track the solutions paths defined by F starting at the points in $S_{N,M}$. Each endpoint is a solution of F_h which we can consider as $F_{\widehat{H}}$ where $\widehat{H} = \frac{1}{\widehat{N}}$ and $\widehat{N} = N \cdot M$. In particular, the set of endpoints can be considered as $S_{\widehat{N},0}$. If the mesh size is sufficiently small, output the solutions; otherwise, go to step 2.

With the proper modifications of the polynomial systems, this method can be applied to other problems.

3 One dimensional examples

We apply the bootstrapping method to a modification of (2.1), namely

$$\begin{cases} u_{xx} &= f(u) \quad \text{on } (0,1), \\ u'(0) &= 0, \\ u(1) &= 0. \end{cases} \tag{3.7}$$

The Neumann boundary condition $u'(0) = 0$ is discretized using a second-order one-sided scheme

$$u'(0) \approx \frac{u(2h) - 4u(h) + 3u(0)}{-2h} \text{ or } u'(0) \approx \frac{u(2H) - 4u(H) + 3u(0)}{-2H}.$$

The "variational form" filtering condition was used in Example 1 while Example 2 utilized the "bounded condition" filtering since the solutions are bounded by $\sqrt{p}$ from [3].

3.1 Example 1

In the first example, we consider $f(u) = -\lambda(1 + u^p)$ where $\lambda \geq 0$ is a real parameter and p is a nonnegative integer. Multiplying by u' and integrating over $[0, x]$, we obtain

$$\frac{1}{2}(u')^2(x) + F(u(x)) - F(u_0) = 0, \tag{3.8}$$

where $F(u) = \int_0^u \lambda(1 + s^p)ds$ and $u_0 = u(0)$. Since $u' < 0$ for $x > 0$, we have

$$\int_0^{u(x)} \frac{ds}{\sqrt{F(u_0) - F(s)}} = \sqrt{2}(1 - x). \tag{3.9}$$

By taking $x = 0$, we obtain

$$G(u_0) := \int_0^{u_0} \frac{ds}{\sqrt{F(u_0) - F(s)}} - \sqrt{2} = 0. \tag{3.10}$$

In particular, there exists λ^* such that

(i) for $0 < \lambda < \lambda^*$, there are two solutions,

(ii) the two solutions merge at $\lambda = \lambda^*$, and

(iii) for $\lambda > \lambda^*$, there are no solutions.

Figure 1 presents $G(u_0)$ for different values of λ with $p = 4$. With this setup, one can verify that $\lambda^* \approx 1.30107$. Exact values of the real solutions can be obtained from (3.9) which we compare with the numerical bootstrapping method. We implemented this method utilizing Bertini [2], which is a software package for solving polynomial system and tracking homotopy paths. To solve this problem, we used 12 computing nodes each containing two Xeon 5410 processors running 64-bit Linux, i.e., each node consists of 8 processing cores.

This computation yielded two real solutions for $\lambda = 1.2$ and one solution for $\lambda = \lambda^*$, which are displayed in Figure 2. Table 1 lists the number of PDE solutions, N, M, numerical errors, and the time required for the computation.

Table 1: Summary of solutions

λ	(N, M)	# solns	# $S_{N,M}$	1^{st} soln Err.	2^{nd} soln Err.	Time
1.2	$(7, 3)$	2	4	1.105364e-4	3.714641e-4	4m5s
	$(21, 2)$	2	10	3.236483e-5	1.133685e-4	31m16s
	$(42, 2)$	2	16	8.384076e-6	3.029060e-5	5h39m58s
λ^*	$(7, 3)$	1	2	1.674890e-4		1m56s
	$(21, 2)$	1	6	3.862634e-5		14m46s
	$(42, 2)$	1	10	1.087360e-5		2h45s42s

3.2 Example 2

In the second example, we consider $f(u) = -\frac{\pi^2}{4}u^2(u^2 - p)$ where $p \geq 0$ is a real parameter [3]. We utilized the shooting method to confirm our

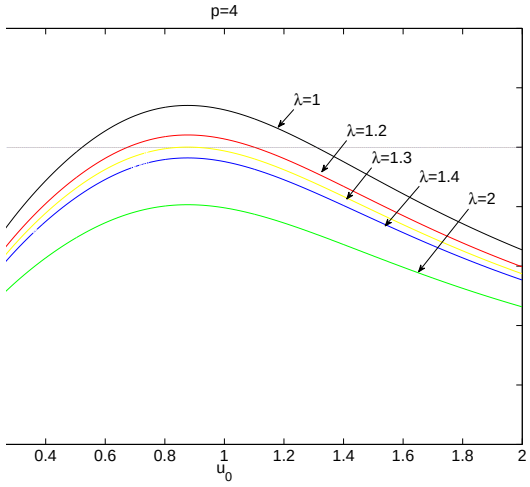

Figure 1: $G(u_0)$ for $p = 4$

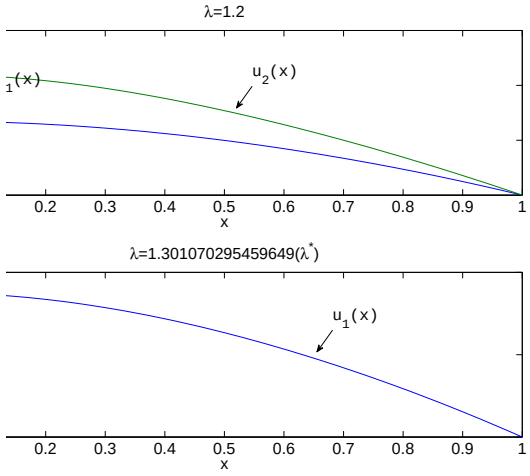

Figure 2: Two solutions for $\lambda = 1.2$ and unique solution for $\lambda = \lambda^*$

results regarding the number of solutions. In particular, let $v(t;\beta)$ denote the solution of the initial value problem

$$\begin{cases} -v_{xx} &= \frac{\pi^2}{4} v^2(v^2 - p) \quad \text{on } (0,1), \\ v'(0) &= 0, \\ v(0) &= \beta. \end{cases} \tag{3.11}$$

Define $F(\beta) = v(1;\beta)$ and note that the roots of F correspond to solutions of (3.7). Figure 3 plots F for p equal to 1, 8, and 10. This clearly shows that F has 2, 4, and 8 roots, respectively for these values of p.

Using the same setup utilized in Section 3.1, we used the bootstrapping method to approximate the solutions for these values of p. This computation produced the same number of solutions as the shooting method. Figure 4 plots these solutions and Table 2 summarizes the data of this computation.

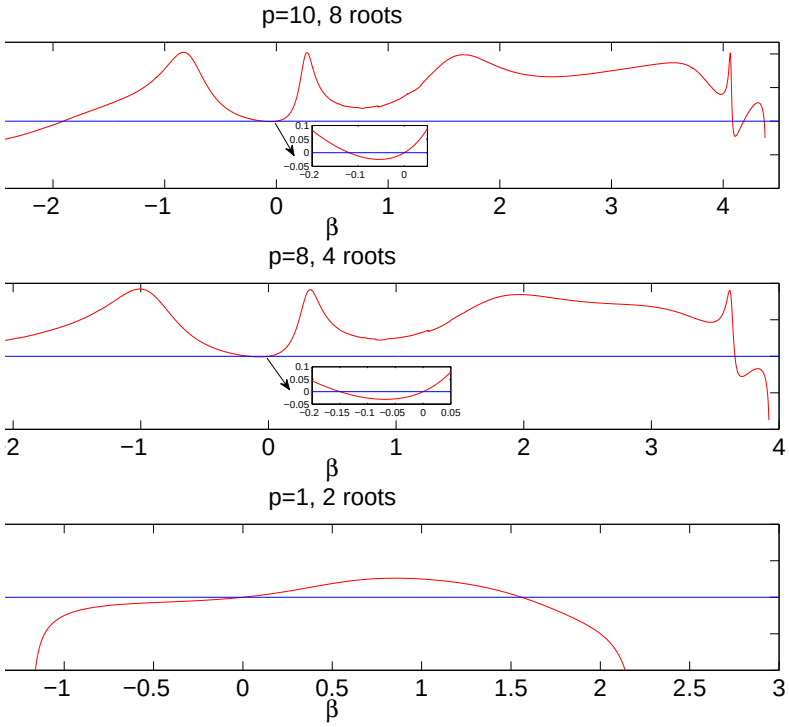

Figure 3: Plots of F using the shooting method

4 Two dimensional bootstrapping

We extend the bootstrapping method from Section 2 to two space dimensional problems by consider Laplace's equation on the unit square. Let

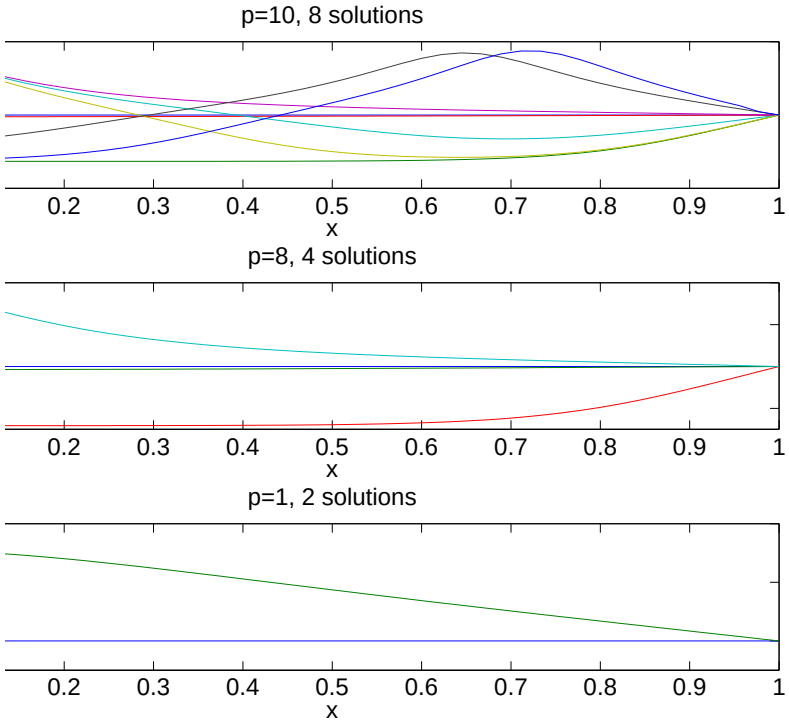

Figure 4: Solutions for different values of p

Table 2: Summary of solutions

p	(N, M)	# solns	# $S_{N,M}$	Time
1	(10,3)	2	48	1h23m
	$(30, 3)$	2	19	12m23s
8	$(10, 3)$	4	120	1h57m
	$(30, 3)$	4	48	27m56s
10	$(10, 3)$	12	490	3h56m
	$(30, 3)$	8	108	56m42s

$\Omega = (0, 1) \times (0, 1)$ and suppose that $u(x, y)$ defined on $\overline{\Omega}$ is a solution to

$$\begin{cases} \Delta u &= f(u) & \text{on } \Omega, \\ u(x, y) &= g(x, y) & \text{on } \partial\Omega. \end{cases} \tag{4.12}$$

Analogous to Section 2, we decompose $\overline{\Omega}$ into equal subdomains and then decompose each subdomain. Let N_x and N_y be positive integers with $H_x = \frac{1}{N_x}$ and $H_y = \frac{1}{N_y}$, and grid points $(x_{i_1}, y_{i_2}) = (i_1 H_x, i_2 H_y)$. The grid points in each domain $[x_{i_1}, x_{i_1+1}] \times [y_{i_2}, y_{i_2+1}]$ are

$$(x_{i_1,j_1}, y_{i_2,j_2}) = (i_1 H_x + j_1 h_x, i_2 H_y + j_2 h_y)$$

where $h_x = \frac{1}{N_x M_x}$ and $h_y = \frac{1}{N_y M_y}$ with positive integers M_x and M_y. To simplify notation, we will write $(i, j) = ((i_1, i_2), (j_1, j_2))$. As before, the values $U_{i,j}$ will be numerical approximations of $u_{i,j} = u(x_{i_1,j_1}, y_{i_2,j_2})$ which form a solution to a polynomial system arising by discretizing the system of differential equations.

The first step of the bootstrapping method solves the polynomial system arising from the discretization with respect to H_x and H_y. For the grid displayed in Figure 5, this computes approximations at the red star points.

The second step uses the solutions computed in step 1 to setup "boundary conditions" for the subdomains. Consider the subdomain $[x_i, x_{i+1}] \times [0, 1]$ with M_x and N_y points in the x and y directions, respectively. For the grid displayed in Figure 5, solving on these grids yield approximations at the green circle points. Now, consider the subdomain $[0, 1] \times [y_i, y_{i+1}]$ with $N_x M_x$ and M_y points in the x and y directions, respectively. For the grid displayed in Figure 5, solving on these grids yield approximations at the black square points.

The third step applies a filter to discard unreasonable solutions. The fourth step uses a homotopy to compute approximations on $\overline{\Omega}$ with $N_x M_x$ and $N_y M_y$ grid points in the x and y directions, respectively, using the polynomial system and approximations constructed in the second step.

5 Two dimensional example

We apply the two space dimensional bootstrapping method to a tumor model without a necrotic core that was discussed in [6]. Let Ω denote the tumor region, σ denote the concentration of nutrients, p denote the pressure, $\widetilde{\sigma}$ denote the concentration of nutrients needed for sustainability, μ denote the aggressiveness of the tumor, and κ denote the mean curvature. A normalized steady state system of a free boundary problem modeling tumor growth is

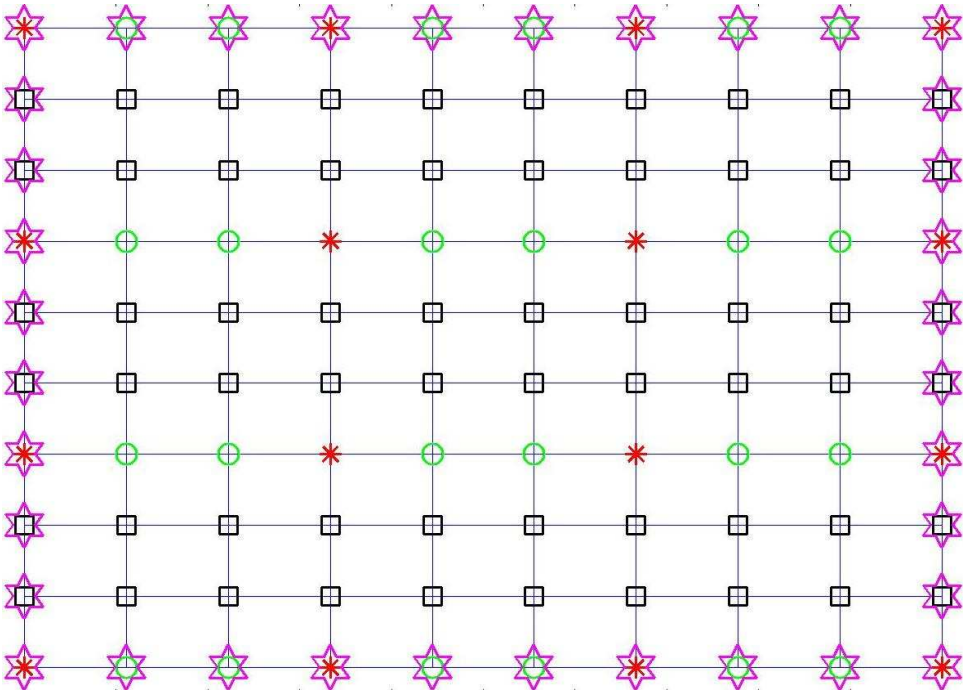

Figure 5: Mesh grid with $N_x = N_y = M_x = M_y = 3$

given by

$$\begin{cases} -\Delta\sigma &= -\sigma & \text{in } \Omega \\ -\Delta p &= \mu(\sigma - \widetilde{\sigma}) & \text{in } \Omega \\ \\ \sigma &= 1 & \text{on } \partial\Omega \\ p &= \kappa & \text{on } \partial\Omega \\ \frac{\partial p}{\partial n} &= 0 & \text{on } \partial\Omega, \end{cases} \tag{5.13}$$

It should be emphasized here that Ω is unknown in advance yielding a free boundary problem. In [6], nonradial solutions were computed by using parameter continuation with respect to μ starting from bifurcation branching points. A more difficult question is to compute other nonradial solutions for a given value of μ. The bootstrapping method allows us to compute such solutions with the goal of computing as many as possible.

We built a filtering condition based on the fact that the distance to the boundary in each angular direction, denoted $R(\theta)$, must be positive. This was enforced by requiring $R(\theta) > -\epsilon$. Here we choose $\epsilon = 1$ by taking numerical error into consideration. Using the same floating grids described in

[6] along with a second-order difference scheme, we employed the bootstrapping algorithm utilizing Bertini on 20 computing nodes of the type described in Section 3.1.

Let N_R and N_θ denote the number of subdomains in the radial, R, and angular, θ, direction, respectively. Each subdomains was decomposed using M_R and M_θ grid points in the R and θ directions, respectively. For small values of N_R and N_θ, e.g., $N_R = 2$ and $N_\theta = 3$, Bertini computed all solutions of the polynomial system arising from the discretization. This polynomial system has a natural multihomogeneous structure which we utilized to reduce the bound on the number of isolated solutions over $\mathbb{C}$ from 3^{17} =129,140,163 to 923,440. This was the starting point for the bootstrapping method which built numerical solutions on the whole domain using more grid points. Table 3 lists number of grid points, number of solutions, and time needed for the computation.

Table 3: Summary for solving (5.13)

μ	N_R	N_θ	M_R	M_θ	Num. of solns	Time
3	2	3	3	1	56	1h46m41s
	8	5	1	1	117	2d2h47m50s
2	2	3	3	1	34	1h12m13s
	8	5	1	1	61	23h35m45s
1	2	3	3	1	4	54m34s
	8	5	1	1	6	11h56m23s
0.9	2	3	3	1	2	44m21s
	8	5	1	1	2	7h3m25s
0.8	2	3	3	1	1	22m13s
	8	5	1	1	1	3h46m12s
0.5	2	3	3	1	1	22m11s
	8	5	1	1	1	3h45m50s

After obtaining a nonradial solution using the bootstrapping method, we utilized higher order difference schemes and finer grids to refine the solutions. The solutions for the higher order schemes were computed using a homotopy starting from the second order scheme. After refinement, some "fake" solutions were found and discarded. For example, with $\mu = 3$, we found 24 solutions using this process which are presented in Figure 7. We note that solution 21 presented in Figure 7 is the radial solution for $\mu = 3$. Starting from solutions 2 and 4 using a parameter continuation with respect

to μ, we found that these solutions arise from the bifurcation at μ_2 (see [6]) for more details regarding this bifurcation point). Figure 6 shows the solution behavior of these paths starting from solutions 2 and 4 which intersect the radial solution branch at μ_2.

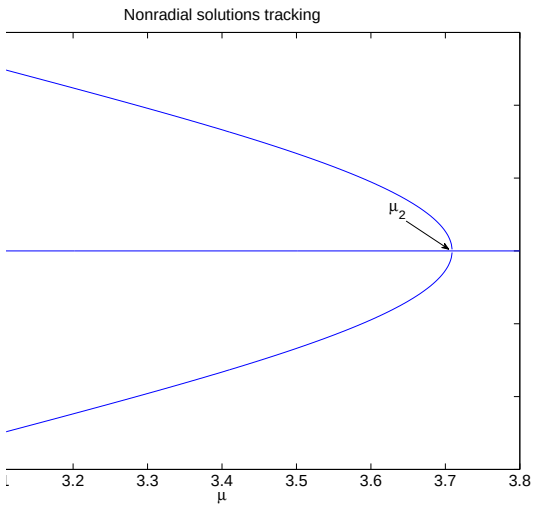

Figure 6: Tracking solution paths back to the bifurcation at μ_2

For smaller values of μ, e.g., $\mu = 0.5$ and $\mu = 0.8$, this bootstrapping process only yielded the radial solution. This suggests that no nonradial solutions exist for these values of μ, which is consistent with the theoretical results regarding this tumor model [6].

6 Conclusion

A bootstrapping method for computing solutions to systems of differential equations has been presented for one and two space dimensional problems. The computations needed in this method are parallelizable and allows one to compute solutions to extremely large polynomial systems. The bootstrapping algorithm for three dimensional case shall be considered in the future along with a detailed analysis of how to pick the starting value of N as well as M in the algorithm. Another line of research is developing and testing filtering conditions.

References

[1] E. L. Allgower, D. J. Bates, A. J. Sommese, and C. W. Wampler, Solution of Polynomial Systems Derived from Differential Equations, *Computing*, Vol. 76, 1–10 (2005).

[2] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Bertini: Software for numerical algebraic geometry. Available at `www.nd.edu/∼sommese/bertini`.

[3] C. Chen, Z. Xie, Structure of muliple solutions for nonlinear differential equations , *Science in China Ser. A Mathematics*, Vol. 47, 172–180 (2004).

[4] C.N. Dawson, Q. Du, and T. F. Dupont, A finite difference domain decomposition algorithm for numerical solution of the Heat equation,*Math. Comp.*, Vol. 57, 63–71 (1991).

[5] W. Hao, J.D. Hauenstein, B. Hu, Y. Liu, A.J. Sommese, and Y.-T. Zhang, Multiple stable steady states of a reaction-diffusion model on zebrafish dorsal-ventral patterning, Discrete and Continuous Dynamical Systems - Series S, Vol. 4, 1413–1428 (2011):`doi:10.3934/dcdss.2011.4.1413`.

[6] W. Hao, J.D. Hauenstein, B. Hu, Y. Liu, A.J. Sommese, and Y.-T. Zhang, Bifurcation for a free boundary problem modeling the growth of a tumor with a necrotic core, preprint available at `www.nd.edu/∼sommese/preprints`.

[7] W. Hao and S. Zhu, Parallel iterative methods for parabolic equations, *Int. J. Comput. Math*, Vol. 86, 431–440 (2009).

[8] A.J. Sommese and C.W. Wampler *Numerical solution of systems of polynomials arising in engineering and science*, World Scientific, Singapore (2005).

[9] G. Yuan, S. Zhu and L. Shen, Domain decomposition algorithm based on the group explicit formula for the heat equation, *Int. J. Comput. Math*, Vol. 82, 1295–1306 (2005).

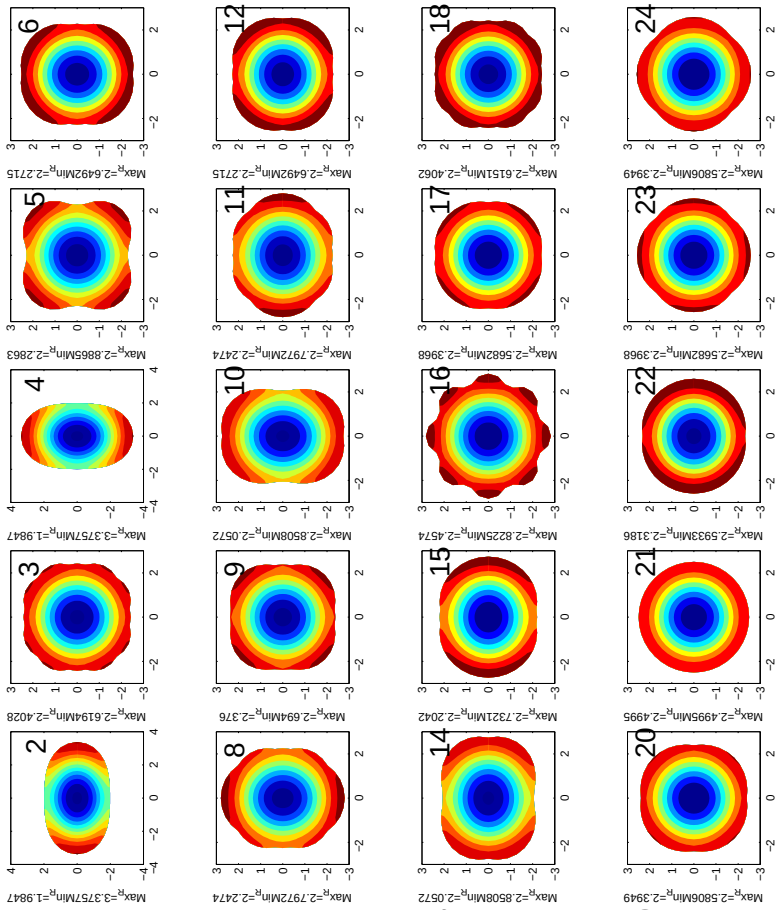

Figure 7: Solutions for $\mu = 3$