Probability-One Homotopy Maps for Mixed Complementarity Problems

Kapil Ahuja

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science & Applications

Layne T. Watson (Chairman)
Ekkehard W. Sachs (Committee Member)
Calvin J. Ribbens (Committee Member)

March 19, 2007
Blacksburg, VA

Keywords: globally convergent, Newton homotopy, nonlinear embedding,
complementarity, optimization

Probability-One Homotopy Maps for Mixed Complementarity Problems

Kapil Ahuja

ABSTRACT

Probability-one homotopy algorithms have strong convergence characteristics under mild assumptions. Such algorithms for mixed complementarity problems (MCPs) have potentially wide impact because MCPs are pervasive in science and engineering. A probability-one homotopy algorithm for MCPs was developed earlier by Billups and Watson based on the default homotopy mapping. This algorithm had guaranteed global convergence under some mild conditions, and was able to solve most of the MCPs from the MCPLIB test library. This thesis extends that work by presenting some other homotopy mappings, enabling the solution of all the remaining problems from MCPLIB. The homotopy maps employed are the Newton homotopy and homotopy parameter embeddings.

Acknowledgments

I would like to thank Dr. Watson for being my advisor and teaching me how to do research. I am grateful to him for the long hours he spent with me in technical discussions as well as in editing the paper. I thank Dr. Watson also for supporting me over Summer 2005.

I would also like to thank Dr. Billups from University of Colorado at Denver. He helped me in numerous ways including environment setup at a machine at CU Denver. Although I never met him in person, he was always there to help me when needed.

I am thankful to Dr. Sachs for having technical discussions with me that made me rethink certain concepts. I am thankful to Dr. Ribbens for being on my committee.

I would like to express my gratitude to Edtech (Learning Technologies at Virginia Tech) where I have had an assistantship since 2004. Thanks especially to Aaron Zeckoski for giving me flexible hours so that I could manage both my studies and work.

Thanks to my mother and sister for their constant support, encouragement, and love.

Contents

List of Figures

List of Tables

1. Introduction

Probability-one homotopy methods have proven to be extremely successful in solving smooth systems of highly nonlinear equations. A fundamental reason is that these methods do not depend on descent of a merit function and so are insensitive to local fluctuations that can cause descent-based methods to get stuck in a local minimum. Homotopy methods work by following the zero curve of an appropriately defined homotopy mapping. As long as the function satisfies certain global characteristics (which depend on the choice of homotopy mapping), the zero curve is guaranteed, with probability one, to reach a solution.

Motivated by this theoretical advantage and the practical successes of these methods, a number of papers have extended probability-one homotopy methods to solve complementarity problems. Early work addressed linear complementarity problems (LCPs) [27] and nonlinear complementarity problems (NCPs) [26]. Probability-one homotopies were first applied to mixed complementarity problems in [5]. There, the homotopy algorithm only provided a better restarting point after a failed Newton iteration. A more in-depth study of homotopy algorithms for MCPs was done in [6]. There, a probability-one homotopy algorithm based on the default homotopy mapping ((2.9) below) was developed. Under some mild conditions (which are easily satisfied), the algorithm is guaranteed to solve any MCP having finite lower and upper bounds. This algorithm solved most of the MCPs from the MCPLIB [9] test library, including, as expected, all of the problems with finite bounds.

The failures reported in [6] indicate that the default homotopy mapping is not always a reliable choice when some bounds are not finite. However, part of the power of homotopy methods is that there is a great deal of flexibility in constructing the homotopy mapping. Thus, if one mapping is not well-suited for the problem, it is usually not too difficult to construct a mapping that better exploits the global characteristics of the problem. Thus, experienced users of homotopy methods usually have no trouble solving many problems that are intractable by other means.

This thesis demonstrates that a similar strategy can be successful for complementarity problems. The work done in [6] is extended by presenting some other probability-one homotopy algorithms that solve the remaining unsolved problems in [6]. The algorithms presented here fall under two categories—(1) algorithms based on the Newton homotopy [17], [25], and (2) algorithms that embed the homotopy parameter λ inside the problem [29], [31].

The thesis is organized as follows. Section 2 covers some basics of complementarity problems and homotopy maps. In Section 3, related work is summarized along with a review of software

libraries used. The Newton homotopy map is described in Section 4, and the approach of embedding the homotopy parameter λ in the problem is discussed in Section 5. Section 6 is dedicated to implementation details. Results are presented in Section 7. Finally, Section 8 concludes the thesis with a discussion of possible future work.

2. Complementarity problems and homotopy

The mixed complementarity problem (MCP) is defined by a nonlinear function and bounds on its variables. Given a nonlinear function $H : R^n \to R^n$ and lower and upper bound n-vectors l, u, respectively, where $-\infty \leq l_i < u_i \leq \infty$ for $i = 1, \ldots, n$, the problem MCP$(H, \mathcal{B}_{l,u})$ [11] is to find a vector $x \in R^n$ such that $x \in \mathcal{B}_{l,u} = \prod_{i=1}^{n}[l_i, u_i]$ and

$$l_i < x_i < u_i \Rightarrow H_i(x) = 0, \tag{2.1}$$

$$x_i = l_i \Rightarrow H_i(x) \geq 0, \tag{2.2}$$

$$x_i = u_i \Rightarrow H_i(x) \leq 0. \tag{2.3}$$

The nonlinear complementarity problem (NCP) and a system of nonlinear equations are both special cases of an MCP. The NCP corresponds to $l_i = 0$ and $u_i = +\infty$ for all i, and a system of nonlinear equations corresponds to $l_i = -\infty$ and $u_i = +\infty$ for all i. A linear complementarity problem (LCP) is a special case of an NCP where $H(x) = q + Mx$ [27], where $q \in R^n$ and M is an $n \times n$ matrix. Since MCPs subsume all types of complementarity problems, their applications are quite diverse, ranging from mechanics and chemical engineering to economics [27].

The homotopy methods described in this thesis, like those in [5] and [6], require reformulating the MCP as a nonlinear system of equations. Thus, Section 2.1 covers how to formulate the MCP as a nonlinear system. In Section 2.2 some basics of homotopy methods, including the default homotopy mapping and its application to MCPs, are covered. Many homotopy maps for solving LCPs involve homotopy parameter embeddings, and thus such maps are briefly revisited in Section 2.3. Section 2.4 describes different approaches of using the predictor-corrector algorithm for homotopy zero curve tracking. The following notational conventions are used. For a function $F : R^n \to R^m$, $\nabla F(x)$ is defined as the $m \times n$ Jacobian matrix. Given a set Ω, $\partial\Omega$ represents its boundary and $\bar{\Omega}$ represents its closure.

2.1. Formulation of the MCP as a nonlinear system

One approach to solving MCPs is to reformulate them as systems of equations. One such reformulation, used in [5] and [6], is to define the MCP function $F : R^n \to R^n$ by

$$F_i(x) = \phi^{FB}(x_i - l_i, -\phi^{FB}(u_i - x_i, -H_i(x))) = 0, \qquad i = 1, \ldots, n, \tag{2.4}$$

where ϕ^{FB} is the Fischer-Burmeister function defined by

$$\phi^{FB}(a,b) = a + b - \sqrt{a^2 + b^2}. \tag{2.5}$$

F is called an *MCP function* because solving the MCP is equivalent to solving the nonlinear equation $F(x) = 0$. That is, x solves the MCP (2.1)–(2.3) if and only if $F(x) = 0$. Solving nonsmooth equations is usually difficult, so the nonsmooth system (2.4) is often smoothed. A C^2 smoother of ϕ^{FB} with smoothing parameter $\mu > 0$ is

$$\phi^{BW}(a,b,\mu) = a + b - \sqrt{a^2 + b^2 + \mu^2}. \tag{2.6}$$

Applying this C^2 smoother to (2.4) gives the smoothed MCP function

$$F_i^{\mu}(x) = \phi^{BW}(x_i - l_i, -\phi^{BW}(u_i - x_i, -H_i(x), \mu), \mu) = 0, \qquad i = 1, \ldots, n. \tag{2.7}$$

If H is C^2, then F^{μ} is also C^2, and $\lim_{\mu \to 0+} F^{\mu}(x) = F(x)$. Numerical computation is done with F^{μ} rather than with F.

2.2. Continuation and homotopy

Continuation is a technique in numerical analysis for solving a system of nonlinear equations where one starts from a simple problem whose solution is known ($G(x) = 0$) and ends up with the solution of the problem at hand ($F(x) = 0$) [18], [31]. This is done by tracking the solution of a family of problems ($\rho(\lambda, x) = 0$) with respect to a parameter λ such that at $\lambda = 0$, $\rho(0, x) = G(x) = 0$, and at $\lambda = 1$, $\rho(1, x) = F(x) = 0$. A typical choice for the function ρ (known as a homotopy map) is

$$\rho(\lambda, x) = \lambda F(x) + (1 - \lambda)G(x), \qquad \lambda \in [0, 1]. \tag{2.8}$$

Continuation methods, as traditionally implemented (incrementing λ monotonically from 0 to 1), have some limitations [31]. The continuation process may fail because of

(1) turning points in the path,

(2) bifurcations in the path,

(3) the curve of solutions returns to $\lambda = 0$,

(4) the curve of solutions wanders off to infinity,

(5) the curve of solutions might not exist for all $\lambda \in$ [0,1].

Many authors use the terms continuation and homotopy interchangeably, but technically a homotopy method is a generalization of the continuation technique designed to address directly turning points and bifurcations (limitations (1) and (2)) [31]. A type of homotopy algorithm, in which most of the limitations mentioned above are avoided, is the probability-one homotopy algorithm. These probability-one homotopies satisfy certain assumptions, which are summarized in the following theorem (Proposition 2.2 from [6]):

Theorem 2.2.1. *Let $F : R^n \to R^n$ be a Lipschitz continuous function and suppose there is a C^2 map*

$$\rho : R^m \times [0, 1) \times R^n \to R^n$$

such that

(1) *$\nabla\rho(a, \lambda, x)$ has rank n on the set $\rho^{-1}(\{0\})$,*

(2) *the equation $\rho_a(0, x) = 0$, where $\rho_a(\lambda, x) = \rho(a, \lambda, x)$, has a unique solution $x^a \in R^n$ for every fixed $a \in R^m$,*

(3) *$\nabla_x\rho_a(0, x^a)$ has rank n for every $a \in R^m$,*

(4) *ρ is continuously extendible (in the sense of Buck [7]) to the domain $R^m \times [0, 1] \times R^n$, and $\rho_a(1, x) = F(x)$ for all $x \in R^n$ and $a \in R^m$, and*

(5) *γ_a, the connected component of $\rho_a^{-1}(\{0\})$ containing $(0, x^a)$, is bounded for almost every $a \in R^m$.*

Then for almost every $a \in R^m$ there is a zero curve γ_a of ρ_a, along which $\nabla\rho_a$ has rank n, emanating from $(0, x^a)$ and reaching a zero $\bar{x}$ of F at $\lambda = 1$. Further, γ_a does not intersect itself and is disjoint from any other zeros of ρ_a. Also, if γ_a reaches a point $(1, \bar{x})$ and F is strongly regular at $\bar{x}$, then γ_a has finite arc length.

In the statement of the above theorem, a function is *strongly regular* if all elements of the Clark-subdifferential are nonsingular (see [6] for details).

Theorem 2.2.1 also holds for $\rho : U \times [0, 1) \times V \to R^n$ where $U \subset R^m$, $V \subset R^n$ are nonempty open sets. A well studied homotopy map that satisfies most of the above assumptions is the default homotopy map (so called because this is the default map used in the software package HOMPACK90 [30]) and is given by

$$\rho_a(\lambda, x) = \lambda F(x) + (1 - \lambda)(x - a). \tag{2.9}$$

If F is C^2, then assumptions (1)–(4) are easily satisfied by the default homotopy map (2.9) [6]. Assumption (5) is satisfied by the default homotopy map (2.9) if for some $\tilde{x} \in R^n$ and $r > 0$ (Theorem 2.3 from [6]),

$$(x - \tilde{x})^T F(x) \geq 0 \text{ whenever } ||x - \tilde{x}|| = r, \tag{2.10}$$

and $||a - \tilde{x}|| < r$. Condition (2.10) guarantees the existence of a path from almost any starting point a, $||a - \tilde{x}|| < r$, to a solution, and has thus been studied extensively for nonsmooth functions and MCPs [6]. For the homotopy mapping described below, condition (2.10) is guaranteed whenever the bounds on the MCP are finite [6].

The default homotopy map (2.9) was used in [6] to solve MCPs by linking the smoothing parameter μ in (2.7) to the homotopy parameter λ. Specifically, μ was chosen to be a strictly decreasing function of λ such that $\mu(\lambda) > 0$ for $0 \leq \lambda < 1$, and $\mu(1) = 0$. The homotopy mapping was then defined by

$$\rho_a^\mu(\lambda, x) = \lambda F^{\mu(\lambda)}(x) - (1 - \lambda)(x - a), \qquad \lambda \in [0, 1], \tag{2.11}$$

where F^μ is the smoothed MCP function (2.7). By Theorem 4.1 from [6], the zero curve of this mapping emanating from $(\lambda, x) = (0, a)$ is guaranteed, with probability one, to lead to a solution of MCP$(H, \mathcal{B}_{l,u})$ when the following conditions are satisfied:

(1) $H(x)$ is C^2,

(2) the starting point a is chosen in the interior of $\mathcal{B}_{l,u}$,

(3) $\mathcal{B}_{l,u}$ is compact, and

(4) the smoother μ is C^2 and satisfies

$$\mu(\lambda)^2 \leq 2\frac{1 - \lambda}{\lambda}(u_i - a_i)(u_i - l_i), \qquad i = 1, \ldots, n, \qquad \lambda \in (0, 1]. \tag{2.12}$$

2.3. Homotopy methods for LCPs

As discussed earlier, for a specific value of bounds (l, u) and function (H), in Equations (2.1) through (2.3), a MCP becomes a LCP. Specifically the problem LCP is to find a vector $x \in R^n$ such that [27]:

$$x \geq 0, \qquad H \geq 0, \qquad H^t x = 0, \tag{2.13}$$

where $H = (q + Mx) \in R^n$, $q \in R^n$, and M is an $n \times n$ matrix. This section focuses on LCPs because, as described before, homotopy maps for solving LCPs involve techniques similar to embedding a homotopy parameter (λ) in the problem.

Rich theory exists for solving LCPs by homotopy methods and a number of homotopy maps have been successfully used in solving them. A few simple ones are touched upon here. For detailed theory and numerical results please refer to [27]. All the homotopy maps discussed here are based on a reformulation developed by Managasarian [19], in which the LCP is converted to a system of nonlinear equations. This system is given by:

$$F_i(x) = -|M_{i.}x + q_i - x_i|^3 + (M_{i.}x + q_i)^3 + {x_i}^3, \qquad i = 1, \ldots, n. \tag{2.14}$$

Thus, solving $F(x) = 0$ is equivalent to solving the LCP and vice versa. Equation (2.14) can now be used to develop a number of homotopy maps. Some of them are as follows:

(1) Use of default homotopy map: The default homotopy map given by Equation (2.9) is used here. $F(x)$ in Equation (2.9) is set equal to $F(x)$ defined by Equation (2.14). The advantage of this map is that with certain constraints on M (positive definiteness, etc.), a zero curve of $\rho_a(\lambda, x)$ can be successfully tracked from $\lambda = 0$ to $\lambda = 1$. The disadvantage of using this homotopy map is that the feasibility ($x \geq 0$, $H \geq 0$) and complementarity ($H^t x = 0$) conditions are not satisfied for all points along $\rho_a(\lambda, x) = 0$.

(2) Relaxation of M: Here the homotopy map is obtained by embedding the homotopy parameter (λ) inside $F(x) = 0$ as follows: $(1 - \lambda)I + \lambda M$. This embedding gives the following homotopy map ($i = 1, \ldots, n$):

$$\varphi_i(\lambda, x) = -|[(1 - \lambda)I + \lambda M]_{i.}x + q_i - x_i|^3 + ([(1 - \lambda)I + \lambda M]_{i.}x + q_i)^3 + {x_i}^3. \tag{2.15}$$

This homotopy map avoids the problem of not having the feasibility and complementarity conditions satisfied along the zero curve, but has the problem that no single zero curve is guaranteed to reach $\lambda = 1$ from $\lambda = 0$.

(3) Relaxation of q: This is quite similar to the homotopy map (2.15). The difference here is that the homotopy parameter (λ) here is embedded with q rather than M. This gives the following map ($i = 1, \ldots, n$):

$$\vartheta_i(\lambda, x) = -|M_{i.}x + \lambda q_i + (1 - \lambda)||q||_\infty - x_i|^3 + (M_{i.}x + \lambda q_i + (1 - \lambda)||q||_\infty)^3 + {x_i}^3. \tag{2.16}$$

The advantages and disadvantages for this map are similar to those of the homotopy map (2.15), the difference being in proofs related to the number of Jacobian matrix singularities (of $\varphi(\lambda, x)$ and $\vartheta(\lambda, x)$) along the zero curve.

(4) Relaxation of complementarity: The homotopy map here is defined as ($i = 1, \ldots, n$):

$$\varrho_i(\lambda, x) = -\lambda|M_{i.}x + q_i - x_i|^3 + \lambda(M_{i.}x + q_i)^3 + {x_i}^3 - (1-\lambda){a_i}^3. \tag{2.17}$$

This map retains the advantage of the default homotopy map. That is, under some mild constraints the zero curve of $\varrho(\lambda, x)$ can be successfully tracked from $\lambda = 0$ to $\lambda = 1$. The disadvantages of homotopy maps (2.15) and (2.16) are to some extent removed here. This map maintains feasibility of x for all points along $\varrho(\lambda, x) = 0$. However, the complementarity condition is imposed only at $\lambda = 1$.

2.4. Predictor-corrector path following

Predictor-corrector path following is an approach for tracking the zero curve of the given homotopy map [2], [23], [30]. For the discussion below, assume that the zero curve of a homotopy map for a given starting point a is defined by $\rho_a(\lambda, x) = 0$, and is denoted by γ_a. Also, assume that the zero curve γ_a can be parameterized by the arc length denoted by s. The predictor-corrector algorithm, as the name suggests, consists of two steps.

(1) Predictor phase. Here an approximate step is taken on the curve in the direction of the tangent at the pertinent point. That is, given a current point (λ, x) the unit tangent vector $T = (d\lambda/ds, dx/ds)^t$ at this point is computed, and a small step ϵ is taken along the tangent vector direction to obtain the predicted point (λ_p, x_p). That is:

$$(\lambda_p, x_p) = (\lambda, x) + \epsilon(d\lambda/ds, dx/ds), \tag{2.18}$$

where $x, x_p, dx/ds \in R^n$ and $\lambda, \lambda_p, d\lambda/ds \in R$. Equation (2.18) represents a linear predictor. Frequently, a Hermite cubic predictor is used for improved accuracy of the predicted points. In the case of industry-level software packages, such as HOMPACK90 [30], a linear predictor is used at the start, in order to obtain the second point (with the starting point as the first point). After two points on the curve have been obtained, the algorithm switches to a Hermite cubic predictor. A similar approach is applied in estimating the step size ϵ; that is, a suitable step size is chosen at the start. As the algorithm progresses, an optimal step

size is estimated, based on a combination of mathematics, computational experience, and mathematical software principles [30].

(2) Corrector phase. The predicted point does not lie on the zero curve usually, so a series of correction steps are applied with the aim of bringing the predicted point back on the curve. This is done iteratively. If the point after applying a corrector step at k^{th} iteration is denoted as $(\lambda^{(k+1)}, x^{(k+1)})$, then

$$(\lambda^{(k+1)}, x^{(k+1)}) = (\lambda^{(k)}, x^{(k)}) + (\delta_\lambda, \delta_x). \tag{2.19}$$

For $k = 1$, $(\lambda^{(k)}, x^{(k)}) = (\lambda_p, x_p)$. A simple representation of predictor and corrector phases is given in Figure 1.

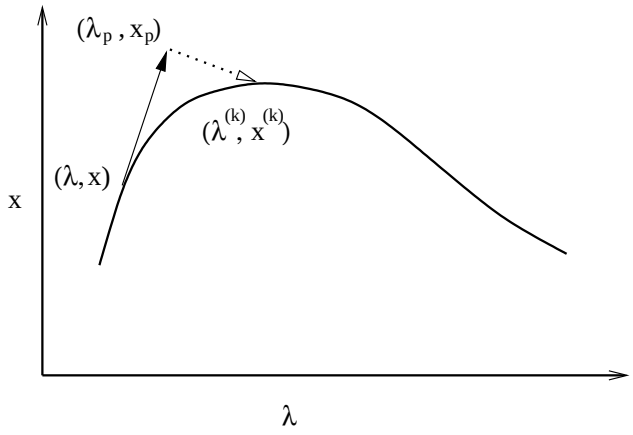

Figure 1. Predictor-corrector procedure.

Computation of the tangent vector T in the predictor phase and the corrector step $(\delta_\lambda, \delta_x)$ in the corrector phase forms an important part of the predictor-corrector algorithms. There are many ways to compute these two values, and some accepted techniques are discussed in the next two subsections.

2.4.1. Finding the tangent vector T for the predictor phase

As $\rho_a(\lambda(s), x(s)) = 0$ for all $s \geq 0$, we can take the derivative of this expression, leading to the following equation:

$$\frac{\partial \rho_a}{d\lambda}\frac{d\lambda}{ds} + \frac{\partial \rho_a}{dx}\frac{dx}{ds} = 0. \tag{2.20}$$

Another form of Equation (2.20) is $\nabla\rho_a(\lambda, x) \cdot T = 0$, that is, Jacobian matrix of ρ_a times the tangent vector. We are considering the unit tangent vector, so the normalization condition gives

$$|d\lambda/ds|^2 + ||dx/ds||_2^{\,2} = 1. \tag{2.21}$$

Equations (2.20) and (2.21) can be solved to obtain the tangent vector T. To ensure that progress is made along the zero curve, the tangent vector should point in the direction of the increasing arc length s. Such a guarantee can be obtained by choosing the orientation of the tangent vector T at the current point (λ, x), so that T makes an acute angle with the tangent vector $T_0 = (d\lambda_0/ds, dx_0/ds)^t$ at the previous point (λ_0, x_0).

Another technique, that ensures that the tangent vector T at the current point (λ, x) makes an acute angle naturally with the tangent vector T_0 at the previous point (λ_0, x_0), is to use the following equation (instead of Equation (2.21)):

$$T_0{}^t \cdot z = 1, \tag{2.22}$$

where $T = \frac{z}{||z||}$. Thus, the complete linear system (i.e., combining Equations (2.20) and (2.22)) can now be written as:

$$\begin{pmatrix} \nabla\rho_a(\lambda, x) \\ T_0{}^t \end{pmatrix} z = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{2.23}$$

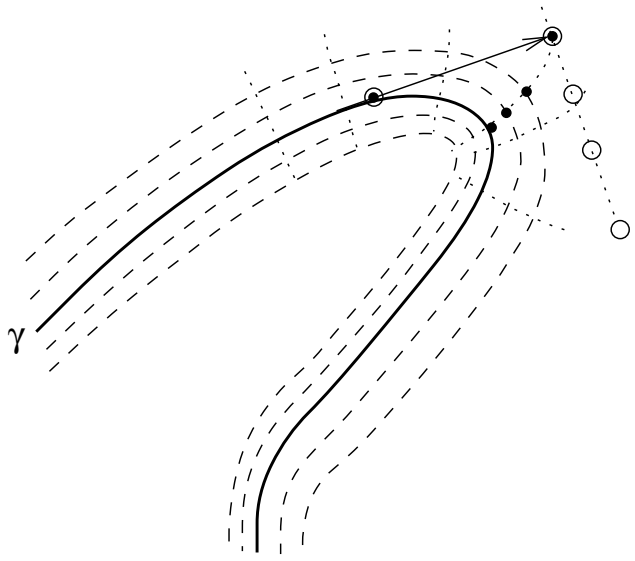

Figure 2. Davidenko flow, normal flow iterates •, and augmented Jacobian matrix iterates ∘ [30] (c) 1997 ACM, Inc. Reprinted by permission.

2.4.2. Finding the corrector step $(\delta_\lambda, \delta_x)$ for the corrector phase

The corrector step $(\delta_\lambda, \delta_x)$ is usually obtained by either applying pure Newton iterations or quasi-Newton iterations. This section focuses on applying Newton iterations. Details on usage of quasi-Newton iterations (and other sophisticated techniques) are given in [30]. In its simplest form, Newton iteration leads to the following equation:

$$(\delta_\lambda, \delta_x)^t = -[\nabla\rho(\lambda^{(k)}, x^{(k)})]^\dagger \rho(\lambda^{(k)}, x^{(k)}), \tag{2.24}$$

where $\rho : R \times R^n \rightarrow R^n$ and $[\nabla\rho(\lambda^{(k)}, x^{(k)})]^\dagger$ denotes the Moore-Penrose pseudoinverse of the $n \times (n+1)$ Jacobian matrix of $\rho(\lambda^{(k)}, x^{(k)})$. Equation (2.24) is computationally equivalent to finding the unique minimum norm solution of the following equation:

$$[\nabla\rho(\lambda^{(k)}, x^{(k)})](\delta_\lambda, \delta_x)^t = -\rho(\lambda^{(k)}, x^{(k)}). \tag{2.25}$$

For different values of starting point a, a family of zero curves γ_a exists. This family of trajectories is referred as the Davidenko flow. The flow along which the corrector points obtained by Equation (2.24) return to the zero curve is normal to the Davidenko flow. This is illustrated in Figure 2 [30].

Instead of finding the pseudoinverse, another approach is to augment the Jacobian matrix $\nabla\rho(\lambda^{(k)}, x^{(k)})$ with another row, leading to a square Jacobian matrix. Thus, the Newton iteration results in solving a square system of equations. In the field of mechanics, this technique is popularly known as the Riks-Wempner method and is given by the following system of equations:

$$\begin{pmatrix} \nabla\rho(\lambda^{(k)}, x^{(k)}) \\ T^t \end{pmatrix} \begin{pmatrix} \delta_\lambda \\ \delta_x \end{pmatrix} = \begin{pmatrix} -\rho(\lambda^{(k)}, x^{(k)}) \\ 0 \end{pmatrix}, \tag{2.26}$$

where, as before, $T = (d\lambda/ds, dx/ds)^t$ is the unit tangent vector at the current point (λ, x) on the zero curve γ_a. Here the corrector points lie on a hyperplane that passes through the predicted point (λ_p, x_p) and is perpendicular to the tangent vector T. This is illustrated in Figure 2 as the augmented Jacobian matrix iterates [30].

3. Related work and review of software libraries

This section is divided into three subsections. Section 3.1 gives a short summary of available MCP solvers. The General Algebraic Modeling System (used to model MCPs in the MCPLIB test library) is described in detail in Section 3.2. Section 3.3 summarizes the mathematical libraries used in developing the algorithms for this thesis.

3.1. MCP solvers

Over the years many different algorithms have been developed for solving large scale MCPs. The current collection of MCP solvers mostly revolve around some variation of Newton's method for a system of nonlinear equations. A brief theoretical summary of these solvers is as follows [4]:

(1) PATH and MILES: In both these solvers, a sequence of linearized subproblems is solved. They are discussed in more detail in Section 3.2.2.

(2) NE/SQP: Here the MCP is first formulated as a system of nonsmooth equations, and then the Gauss-Newton method is applied to minimize a merit function. The merit function used is the squared 2-norm of the nonsmooth function.

(3) QPCOMP: This extends the NE/SQP algorithm by adding a perturbation strategy that avoids the local minima of the merit function.

(4) PROXI: This solver follows a strategy similar to NE/SQP and QPCOMP, that is, the MCP is first converted to a system of nonsmooth equations; however, the subsequent system is solved using a nonsmooth version of Newton's method rather than the Gauss-Newton method.

A detailed comparison of the above solvers along with descriptions of some other solvers, like SMOOTH, SEMISMOOTH, and SEMICOMP, is given in [4].

3.2. GAMS

The General Algebraic Modeling System (GAMS) is designed for solving optimization problems [15]. The system consists of a modeling language, it's compiler, and a whole suite of in-house as well as third-party high performance solvers. The complete environment is free of cost for small problems (e.g., for the number of constraints less than or equal to 300), however for big problems a paid license is required.

As discussed before, the MCPs of the MCPLIB test library are formulated in the GAMS language. The next subsection (Section 3.2.1) gives a short summary of how this language works.

Section 3.2.2 talks about PATH and MILES (the two most popular MCP solvers in GAMS). Finally, CPLIB, an extension of the GAMS I/O library that is used by MCP solver developers, is discussed in Section 3.2.3.

3.2.1. How GAMS works

Consider the following linear programming problem [20]:

$$Max \quad 109 * Xcorn + 90 * Xwheat + 115 * Xcotton, \tag{3.1}$$

$$\text{such that} \quad Xcorn + Xwheat + Xcotton \leq 100 \quad (land), \tag{3.2}$$

$$6 * Xcorn + 4 * Xwheat + 8 * Xcotton \leq 500 \quad (labor), \tag{3.3}$$

$$Xcorn \geq 0, \quad Xwheat \geq 0, \quad Xcotton \geq 0 \quad (nonnegativity), \tag{3.4}$$

where $Xcorn$, $Xwheat$, and $Xcotton$ denote the land devoted to the cultivation of corn, wheat, and cotton, respectively. The optimization problem consists of maximizing the total profit that is given by the Equation (3.1). Equations (3.2), (3.3), and (3.4) specify constraints imposed by total land available, total labor available, and a nonnegativity constraint respectively. This problem is formulated in the GAMS language as follows [20]:

```
VARIABLES Z;
POSITIVE VARIABLES Xcorn, Xwheat, Xcotton;

EQUATIONS OBJ, land, labor;

OBJ.. Z =E= 109 * Xcorn + 90 * Xwheat + 115 * Xcotton;
land.. Xcorn + Xwheat + Xcotton =L= 100;
labor.. 6*Xcorn + 4 * Xwheat + 8 * Xcotton =L= 500;

MODEL farmPROBLEM /ALL/;
SOLVE PROBLEM USING LP MAXIMIZING Z;
```

3.2.2. MCP solvers with GAMS

The two most popular MCP solvers available with GAMS are PATH [12] and MILES [24]. GAMS has one other MCP solver called NLPEC [13]. Technically NLPEC solves a superset of MCP models, the MPEC (Mathematical Program with Equilibrium Constraints) problem.

As discussed before, PATH and MILES both solve a sequence of linearized subproblems [16]. The main difference between PATH and MILES can be summed up by the fact that PATH is a sophisticated solver, useful for all kinds of problems, while MILES is usually good for easy problems only. This makes PATH the most sought after solver, but the disadvantage is that it is not free (while MILES is free, as it comes with any standard GAMS installation). Some of the PATH features that make it good even for hard problems are: It tries to fix singular models by itself; it has techniques to improve the starting point of the linearized subproblems; it has a restart procedure if an option fails for a difficult problem.

3.2.3. CPLIB

CPLIB stands for Callable Program Library [8]. It is a set of Fortran subroutines that extend the functionality of the GAMS I/O library for variational and complementarity solvers. With CPLIB, a solver can be relatively easily connected with GAMS. When viewed from a MCP solver's perspective, CPLIB acts as a service that provides function and derivative evaluations. CPLIB hides the machine dependent details so that the solver developer can focus on the mathematics rather than porting details. Most solver's I/O requests are fulfilled by CPLIB, however, for certain kinds of input/output operations, a solver may directly call the GAMS I/O library.

To use CPLIB, a MCP solver (subsequently referred to as a solver) needs to perform the following tasks. The solver needs to create two subroutines, CORERQ (core-requirement) and SOLVER. CORERQ is used by CPLIB to communicate with the solver for tasks like requests for workspace. CPLIB expects the solver to perform the main computation inside the SOLVER subroutine. The SOLVER subroutine is also used when the solver needs function and derivative evaluations from CPLIB.

3.3. Mathematical libraries

Use of mathematical libraries in scientific computing is ubiquitous. This is because most of these libraries are stable and thoroughly tested components that perform independent tasks. This helps the developers to concentrate on the main algorithm. For this thesis three mathematical libraries (HOMPACK90, LAPACK, and MINOS) were used. These three libraries are summarized in the following subsections.

3.3.1. HOMPACK90

HOMPACK90 is a software package for solving systems nonlinear of equations [30]. The library is developed in Fortran 90 and is based on globally convergent probability-one homotopy

algorithms. HOMPACK90 builds on HOMPACK, a FORTRAN 77 software package providing similar functionality. HOMPACK is organized in two different ways — by subroutine level and algorithm/problem type. By subroutine level, HOMPACK90 consists of three levels. The first level provides drivers that interface with the user code. The second level contains the actual homotopy algorithm. Numerical linear algebra is contained in the third level. Algorithm/problem type organization is based on three different types of algorithms — ordinary differential equation based algorithm, normal flow algorithm, and augmented Jacobian matrix algorithm. For each algorithm, there are separate routines for dense and sparse Jacobian matrix problems.

For the expert user who prefers to use HOMPACK90 only for homotopy algorithms rather than as a black box that does everything, two routines based on the reverse call paradigm have been provided. Here, whenever HOMPACK90 needs a function value at a point, a Jacobian matrix at a point, or some linear algebra operation, it returns to the calling program. HOMPACK90 also provides a routine for the special case of polynomial systems.

3.3.2. LAPACK

LAPACK (Linear Algebra PACKage) is one of the most widely used software packages in the domain of mathematical software and high performance computing [3]. It is consists of FORTRAN 77 routines for solving a system of linear equations, linear least square problems, eigenvalue problems, and computing singular value decompositions. LAPACK was primarily developed to take advantage of the multi-layer memory hierarchy present in computers, and make the previous numerical libraries (EISPACK and LINPACK) more efficient. The idea was that the algorithms should spend more time in performing computation and less time in moving data between different computer memory levels.

LAPACK routines are organized in three levels based on the granularity of the task performed. Driver routines form the first level and each of these routines fully solve a problem (for example, solving a linear system of equations). The second level is comprised of computational routines that perform independent computational tasks (for example, LU and QR factorizations). Auxiliary routines that perform basic linear algebra tasks similar to BLAS (Basic Linear Algebra Subroutines) form the third level. LAPACK is only available for dense and banded matrices. It is not suitable for sparse matrix problems.

3.3.3. MINOS

MINOS (Modular In-core Nonlinear Optimization System) is a software package developed in FORTRAN 77 and is used for solving large scale linear and nonlinear optimization problems

[22]. It consists of numerically stable algorithms and can be called from C and MATLAB using driver programs. One important limitation of this software is that it can only find locally optimal solutions. If the problem is not structured such that any locally optimal solution is also globally optimal, then MINOS might return any one of the many local optimum points. Another disadvantage of using MINOS is that it is not free. Nevertheless, the most popular algebraic modeling systems like GAMS and AMPL use MINOS as their nonlinear solver.

To use MINOS the user has to supply a couple of subroutines (FUNOBJ, FUNCON, and MATMOD). MINOS also requires the user to provide some files (SPECS, MPS, BASIS, and files containing first and last entry of data read by FUNOBJ and FUNCON). The user should define the nonlinear objective function in the FUNOBJ subroutine and the constraint function in the FUNCON subroutine. MATMOD is needed only when there is a sequence of problems, some of which are related to each other. Run-time parameters should be defined in the SPECS file, while constraints and variables are defined in the MPS file. A BASIS file containing information such as a better starting point for a slightly different problem or steps defining how to restart if the algorithm stops before completion may be provided.

4. Newton homotopy

The Newton homotopy method [17] for solving a system of nonlinear equations is another form of the global Newton method [25]. The global Newton method is represented by the ordinary differential equation

$$\nabla F(x)\frac{dx}{dt} = -\alpha F(x), \tag{4.1}$$

where $F : \bar{\Omega} \to R^n$ is C^2, $\Omega \subset R^n$ is a bounded open set with a smooth connected boundary $\partial\Omega$, and α is a positive real number. With the typical choice $\alpha = 1$, (4.1) can be rewritten as

$$\frac{dx}{dt} = -\big(\nabla F(x)\big)^{-1} F(x), \tag{4.2}$$

(assuming $\nabla F(x)$ is nonsingular). The global Newton method is strongly connected with Newton's method for solving a system of nonlinear equations, because applying Euler's method to (4.2), with a step size of one, gives the classical Newton's method.

The Newton homotopy method (also called the global homotopy method) handles singularities more elegantly than the global Newton method [1] and is defined by the homotopy map

$$\rho(a, \lambda, x) = \rho_a(\lambda, x) = F(x) - (1 - \lambda)F(a), \tag{4.3}$$

where $\rho : \partial\Omega \times R \times \bar{\Omega} \to R^n$, and $x = a$ at $\lambda = 0$. When $\nabla F(x)$ is nonsingular, differentiation of the equation $\rho_a(\lambda, x) = 0$ with respect to arc length s and a bit of algebra gives the equation

$$\frac{dx}{ds} = -\frac{1}{(1-\lambda)}\frac{d\lambda}{ds}\big(\nabla F(x)\big)^{-1} F(x). \tag{4.4}$$

Equations (4.2) and (4.4) show the similarity between the global Newton method and the Newton homotopy method.

The following theorem (which combines Theorem 2.4 and Lemma 2.15 from [17]) establishes the probability-one nature of the Newton homotopy map.

Theorem 4.1.1. *Let $\Omega \subset R^n$ be a bounded open set with a smooth connected boundary $\partial\Omega$. Let $F : \bar{\Omega} \to R^n$ be a C^2 function satisfying*

(1) *$F(x) \neq 0$ and $\nabla F(x)$ is nonsingular for all $x \in \partial\Omega$,*

(2) *rank $\nabla F(x) \geq n - 1$ for all $x \in \Omega$ such that $F(x) = 0$,*

(3) *$F(x) = 0$ has at most countably many solutions in Ω,*

(4) *the Newton direction* $-\nabla F(x)^{-1}F(x)$ *either points into* Ω *for all* $x \in \partial\Omega$*, or points out of* Ω *for all* $x \in \partial\Omega$*.*

Let ρ_a *be defined by (4.3), and let* γ_a *be the connected component of* $\rho_a(\{0\})^{-1}$ *emanating from* $(\lambda, x) = (0, a)$*. Then for almost all* $a \in \partial\Omega$*, the curve* γ_a *reaches a point* $(\lambda, x) = (1, x^*)$*, where* $F(x^*) = 0$ *and* $x^* \in \Omega$*.*

To apply the Newton homotopy to solve MCPs, the smoothed MCP-function (2.7) can again be used with the smoothing parameter μ linked to the homotopy parameter λ. Specifically, by defining the smoothing parameter μ to be a C^2 strictly decreasing function of λ such that $\mu(\lambda) > 0$ for $0 \leq \lambda < 1$, and $\mu(1) = 0$, the following Newton homotopy mapping can be defined

$$\rho_a^{\mu}(\lambda, x) = F^{\mu(\lambda)}(x) - (1-\lambda)F^{\mu(0)}(a), \qquad \lambda \in [0, 1], \tag{4.5}$$

where F^{μ} is the smoothed MCP function (2.7).

For LCPs a homotopy map, closely related to the Newton homotopy map, was developed by Kojima and Saigal [27]. A small discussion of this homotopy map is helpful here because this map has strong convergence properties for a certain kind of LCP matrix M. This provides insights into developing convergence proofs for complementarity problems using the Newton homotopy. This homotopy, referred to as the Kojima-Saigal homotopy in [27], converts the LCP (q, M) (given by Equation (2.13)) into a system of nonlinear of equations using the following formulation:

$$F(H, x) = \begin{pmatrix} H - Mx - q \\ H_1 x_1 \\ \vdots \\ H_n x_n \end{pmatrix}, \tag{4.6}$$

where $F : R^n \times R^n \to R^{2n}$. The homotopy map is given as:

$$K(\lambda, H, x) = \begin{pmatrix} H - Mx - q \\ H_1 x_1 - (1-\lambda){H_1}^{(0)}{x_1}^{(0)} \\ \vdots \\ H_n x_n - (1-\lambda){H_n}^{(0)}{x_n}^{(0)} \end{pmatrix}, \tag{4.7}$$

where $K : [0, 1) \times R^n \times R^n \to R^{2n}$. In the above equation, $x^{(0)} \in R^n$, $x^{(0)} > 0$, and $H^{(0)} = (q + Mx^{(0)}) > 0$ are obtained using phase 1 of the simplex algorithm applied to a slightly modified version of (2.13).

The simplex method for linear programming belongs to a class of optimization problems called active set methods [23]. These methods divide the inequality constraints into two sets. The

first set contains indices of the constraints that are active at a point and the other set contains indices of constraints that are inactive at the point. The constraints for which strict inequality holds at a point are termed inactive constraints. The algorithm progresses by performing small changes to these index sets. That is, at each step an index is transferred from one set to another.

It is important to understand that this Kojima-Saigal homotopy map (Equation (4.7)) is not globally convergent in the sense described by Theorem 2.2.1, because it requires a feasible interior point to start. A feasible interior point satisfies $(H, x) > 0$ and $H = Mx + q$. A convergence proof for the Kojima-Saigal homotopy map for certain types of matrices M is given in [27](Theorem 9.1).

5. Embedding homotopy parameter λ in the problem

Standard homotopy maps (default and Newton) guarantee global convergence under special conditions. For the default homotopy, these conditions, although easily satisfiable, guarantee global convergence only for MCPs having finite lower and upper bounds. The reason is that if either of the bounds are infinite, then $\mathcal{B}_{l,u}$ is not compact, and the zero curve (lying in $[0,1] \times \mathcal{B}_{l,u}$) may wander off to infinity (limitation (4) discussed in Section 2.2). This is illustrated in Figure 3 where the zero curve asymptotically approaches a hyperplane $\lambda = \bar{\lambda} < 1$. For the default homotopy map (2.11), this asymptotic behavior can occur if $F^{\mu(\lambda)}(x)$ and $(x-a)$ are qualitatively dissimilar (in the sense of topological degree). The MCPs of MCPLIB that were not solved by the probability-one homotopy algorithm of [6] had the homotopy zero curve behavior illustrated in Figure 3.

Similar behavior can occur when the Newton homotopy maps of Section 4 are used if the conditions of Theorem 4.1.1 are not satisfied by the MCP reformulations (2.4) or (2.7). The remedy for this zero curve behavior (going to infinity) is to construct the homotopy map as a family of qualitatively similar functions by embedding the homotopy parameter λ inside the problem. For the standard homotopy maps, this means using a modified function $F(\lambda, x)$ instead of $F(x)$ in Equations (2.11), (4.3), and (4.5). The technique for embedding λ in the problem is somewhat of an art but not too difficult with experience.

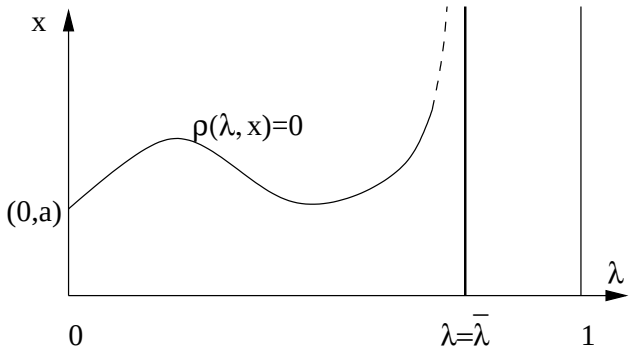

Figure 3. Diverging homotopy zero curve.

Over the years numerous science and engineering problems have been solved by this approach [29], [31]. For example, [21] solves a DC circuit operating point problem very efficiently by this approach. The basic idea used in [21] is to have a transistor model without any nonlinear effects at $\lambda = 0$, and slowly bring these nonlinear effects back into the model as λ approaches 1. Specific to MCPs also, this technique of first solving a mildly nonlinear system, and then gradually bringing the strong nonlinearity back as λ is increased from 0 toward 1, was successful in solving some

MCPs that did not yield to the "standard" homotopy maps. This technique, also termed nonlinear homotopy parameter embedding, is illustrated with a simple example in Section 5.1.

Another example, where the approach of embedding the homotopy parameter λ inside the problem has been useful, is in [28]. However, the embedding technique here (in [28]) is different from the nonlinear embedding described in the previous paragraph. In [28], a general nonlinear programming problem is solved by a homotopy approach. Usually the functions in a nonlinear programming problem contain a parameter vector c, and for a specific $c = c^{(0)}$ a solution to the problem is known. If the problem at hand has a parameter vector $c = c^{(1)}$, then embedding the homotopy parameter λ as

$$c = \lambda \cdot c^{(1)} + (1 - \lambda) \cdot c^{(0)}, \qquad \lambda \in [0, 1], \tag{5.1}$$

helps to solve the problem. For many MCPs (those not solved by "standard" homotopy maps and nonlinear embedding) this technique was quite effective. For example, the elasto-hydrodynamic lubrication problem [10] (ehl_kost in MCPLIB) was solved this way. In elasto-hydrodynamic lubrication there are two elastic cylinders in line contact (Figure 4), and the problem consists of calculating the pressure and lubricant film gap between the two. This problem has three runs with two varying parameters *alph* and *lam* (that are dependent on load and speed related to the two cylinders). The algorithm from [6], that uses the default homotopy map (2.11), was able to solve Run 1 but failed for Runs 2 and 3. Thus, the homotopy parameter λ was embedded in the problem parameter *lam* in the following way (similar to Equation (5.1)):

$$lam = \lambda \cdot lam_2 + (1 - \lambda) \cdot lam_1, \qquad \text{for Run 2,}$$

$$lam = \lambda \cdot lam_3 + (1 - \lambda) \cdot lam_1, \qquad \text{for Run 3,}$$

where lam_k represents the value of *lam* given for Run k.

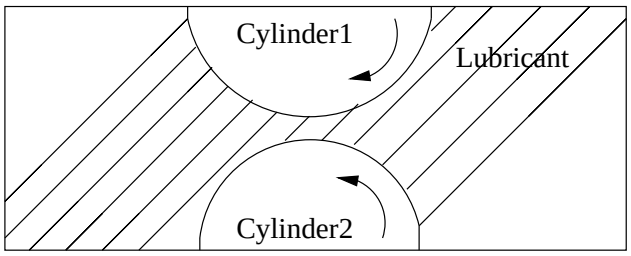

Figure 4. Elasto-hydrodynamic lubrication.

5.1. Example of nonlinear homotopy parameter embedding

To illustrate the nonlinear homotopy parameter embedding approach, a four variable simplified form of Powell's 16 variable nonlinear program [9] is used to demonstrate the process. The MCP (NCP in this example) is defined by variables x_i, their bounds $l_i = 0, u_i = +\infty,\ i = 1, 2, 3, 4$, and corresponding function $H(x)$ given by

$$H_1(x) = e^{(x_1-x_3)(x_2-x_4)}(x_2 - x_4) + c_1(x),$$

$$H_2(x) = e^{(x_1-x_3)(x_2-x_4)}(x_1 - x_3) + c_2(x),$$

$$H_3(x) = -e^{(x_1-x_3)(x_2-x_4)}(x_2 - x_4) - c_1(x),$$

$$H_4(x) = -e^{(x_1-x_3)(x_2-x_4)}(x_1 - x_3) - c_2(x),$$

where $c_1(x)$ and $c_2(x)$ are some slowly varying terms. Application of the smoothed MCP function (2.6)–(2.7) to this NCP gives the nonlinear system

$$F_i^{\mu}(x) = \phi^{BW}(x_i, H_i(x), \mu) = 0, \quad i = 1, 2, 3, 4.$$

The default homotopy map applied to the above nonlinear system gives the system

$$\rho_i^H(\lambda, x) = \lambda\big(\phi^{BW}(x_i, H_i(x), \mu)\big) + (1 - \lambda)(x_i - a_i), \quad i = 1, 2, 3, 4, \tag{5.2}$$

where a_i is the starting value for x_i. The zero curve of (5.2) diverges to infinity as λ is increased from 0 toward 1. However, if the nonlinear function $H(x)$ is simplified to $M(x)$ defined by

$$M_1(x) = e^{(x_1-x_3)(x_2-x_4)} + c_1(x),$$

$$M_2(x) = e^{(x_1-x_3)(x_2-x_4)} + c_2(x),$$

$$M_3(x) = -e^{(x_1-x_3)(x_2-x_4)} - c_1(x),$$

$$M_4(x) = -e^{(x_1-x_3)(x_2-x_4)} - c_2(x),$$

then the zero curve defined by

$$\rho_i^M(\lambda, x) = \lambda\big(\phi^{BW}(x_i, M_i(x), \mu)\big) + (1 - \lambda)(x_i - a_i), \quad i = 1, 2, 3, 4, \tag{5.3}$$

reaches $\lambda = 1$. Thus based on the above behaviors, the homotopy parameter λ is embedded in the NCP such that at $\lambda = 0$, the nonlinear function is $M(x)$, and at $\lambda = 1$, the nonlinear function is $H(x)$. That is, the homotopy map used is

$$\rho_i^N(\lambda, x) = \lambda\big(\phi^{BW}(x_i, N_i(\lambda, x), \mu)\big) + (1 - \lambda)(x_i - a_i), \quad i = 1, 2, 3, 4, \tag{5.4}$$

where

$$N_1(\lambda, x) = e^{(x_1-x_3)(x_2-x_4)}\big(\lambda \cdot (x_2 - x_4) + (1-\lambda)\cdot 1\big) + c_1(x),$$

$$N_2(\lambda, x) = e^{(x_1-x_3)(x_2-x_4)}\big(\lambda \cdot (x_1 - x_3) + (1-\lambda)\cdot 1\big) + c_2(x),$$

$$N_3(\lambda, x) = -e^{(x_1-x_3)(x_2-x_4)}\big(\lambda \cdot (x_2 - x_4) + (1-\lambda)\cdot 1\big) - c_1(x),$$

$$N_4(\lambda, x) = -e^{(x_1-x_3)(x_2-x_4)}\big(\lambda \cdot (x_1 - x_3) + (1-\lambda)\cdot 1\big) - c_2(x).$$

Thus the new homotopy map $\rho^N(\lambda, x)$ has the homotopy parameter λ embedded in the nonlinear function $N(\lambda, x)$.

Tracking of this new zero curve requires code changes, which can be avoided by reformulating the problem. This involves creating new variables y_1, y_2 with infinite bounds, and adding new equations:

$$y_1 - (x_2 - x_4) = 0, \qquad y_2 - (x_1 - x_3) = 0.$$

These new constraints lead to an MCP (since now y_1, y_2 are unbounded) with function

$$P_1(x, y) = e^{(x_1-x_3)(x_2-x_4)}y_1 + c_1(x),$$

$$P_2(x, y) = e^{(x_1-x_3)(x_2-x_4)}y_2 + c_2(x),$$

$$P_3(x, y) = -e^{(x_1-x_3)(x_2-x_4)}y_1 - c_1(x),$$

$$P_4(x, y) = -e^{(x_1-x_3)(x_2-x_4)}y_2 - c_2(x),$$

$$P_5(x, y) = y_1 - (x_2 - x_4),$$

$$P_6(x, y) = y_2 - (x_1 - x_3),$$

and corresponding homotopy map

$$\rho_i^P(\lambda, x, y) = \lambda\big(\phi^{BW}(x_i, P_i(x, y), \mu)\big) + (1-\lambda)(x_i - a_i), \quad i = 1, 2, 3, 4, \tag{5.5}$$

$$\rho_5^P(\lambda, x, y) = \lambda P_5(x, y) + (1-\lambda)(y_1 - 1), \tag{5.6}$$

$$\rho_6^P(\lambda, x, y) = \lambda P_6(x, y) + (1-\lambda)(y_2 - 1). \tag{5.7}$$

A bit of algebra shows that the homotopy map obtained from (5.5)–(5.7) is equivalent to the homotopy map in (5.4).

6. Implementation

For problems where the default homotopy (Section 2.2) was used, the code from the implementation in [6] was reused. Code for the Newton homotopy (Section 4) was developed in C. All linear algebra computations were done using LAPACK [3] (except for two problems for which MINOS [22] was used because a sparse factorization routine was needed). The overall architecture was based on the code developed in [6]. That is, STEPNX (from HOMPACK90 [30]) was used to find the next point on the curve, and CPLIB [8] (from GAMS) was used to calculate the function value and Jacobian matrix at a given point. While calling STEPNX, the unit tangent vector at the current point (λ, x) on the zero curve was computed using Equation (2.23) as discussed in Section 2.4.1. Equation (2.26), from Section 2.4.2, formed the basis of computing the corrector step for the corrector phase.

Problems in MCPLIB were solved using the following strategy. First, the default homotopy mapping was tried. If it was not successful, then the Newton homotopy was used. Finally, if neither of these mappings was successful, then the embedding method described in Section 5 was employed, using the default mapping and then the Newton mapping.

Some of the problems in MCPLIB are not technically MCPs. Instead, they are constrained nonlinear systems (CNS). They are basically square systems of equations ($F(x) = 0$) with bounds on the variables [14]. These bounds ensure certain behaviors, like avoidance of errors in function evaluation, etc. To solve these problems using the Newton homotopy, the bounds were ignored and the simple Newton homotopy map (4.3) was used. The map is called simple because the MCP was neither converted to a nonlinear system, nor smoothed. Finally, the bounds were checked at the solution to ensure they were satisfied. The three problems that were solved by this approach are: hydroc06, hydroc20, and methan08. Details relating to error tolerances, number of homotopy curve tracking steps, etc., for these problems are presented in Section 7.

For legal MCPs that were not solved by the default mapping, the Newton homotopy mapping (4.5) was used. The smoothing parameter μ was defined as $\mu(\lambda) = 0.1(1 - \lambda)$. The problems that were solved by this approach are: electric, scarfasum (1–4), and trafelas (1). Details about these problems are presented in Section 7.

For both the default and Newton homotopy maps, when the homotopy parameter had to be embedded in the problem (Section 5), the problem was reformulated (in GAMS) instead of changing the (homotopy definition) code. This was accomplished using the method described in Section 5.1. The reason is that the exact manner in which λ has to be embedded may not always be clear,

and changing code is cumbersome compared to reformulation. Section 5.1 described the nonlinear embedding technique with a simplified example. An actual MCPLIB example with embedding technique of changing the parameter value (that is using Equation (5.1)), is detailed in Appendix A. The problems that were solved by this approach of embedding λ are: duopoly, ehl_kost (2,3), forcebsm, forcedsa, lincont, powell (5), shubik (4, 10, 34, 36), and trafelas (2). Details about these problems are presented in Table 2 of Section 7.

7. Results

This section gives details pertaining to the homotopy map, curve tracking tolerance, curve tracking function evaluations, and $\|F\|_\infty/(1+\|x\|_\infty)$ for the problems solved using the algorithms discussed in Sections 2.2, 4, and 5. Table 1 gives the details for problems that did not require embedding of the homotopy parameter λ in the problem, while Table 2 gives the details for problems that were solved by embedding the homotopy parameter λ in the problem.

The problem names in the tables are from MCPLIB, and all problems from MCPLIB *not* mentioned in Tables 1, 2 were solved by the "default" homotopy from [6] (except for three black box MPSGE problems for which there was no reasonable way to embed λ because of lack of analytic descriptions).

Table 1. Problems that did not require embedding of homotopy parameter.

Problem (run)	Homotopy map	ARCTOL	NFE	$\frac{\|F\|_\infty}{1+\|x\|_\infty}$
bert_oc (1–4)*	default	10^{-04}	784	5.83×10^{-09}
electric	Newton	10^{-10}	33277	3.07×10^{-07}
hydroc06	Newton (CNS)	10^{-04}	68	2.72×10^{-08}
hydroc20	Newton (CNS)	10^{-04}	157	1.33×10^{-09}
methan08	Newton (CNS)	10^{-04}	25	3.45×10^{-08}
olg†‡	default	10^{-10}	239	6.71×10^{-07}
scarfasum (1–4)	Newton	10^{-04}	39	2.57×10^{-07}
shubik (1–3, 5–9, 11–33, 35, 37–48)‡	default	10^{-11}	9042⋆	9.99×10^{-07}
trafelas (1)	Newton	10^{-05}	1194	1.25×10^{-07}

∗ This was solved by using the same code and settings as in [6] but on a faster computer.

† An algebraic version of this MPSGE problem, also available in MCPLIB, was solved.

‡ These were solved by using the same code as in [6] but with slightly different settings.

⋆ HMAX = 20.

In the tables, the names default, Newton (CNS), and Newton refer to the homotopy maps described by Equations (2.11), (4.3), and (4.5) respectively. ARCTOL is the curve tracking tolerance used by the HOMPACK90 routine STEPNX [30] (ARCTOL = RELERR = ABSERR in

Table 2. Problems requiring embedding of homotopy parameter.

Problem (run)	Homotopy map	ARCTOL	NFE	$\frac{\|F\|_\infty}{1+\|x\|_\infty}$
duopoly	Newton	10^{-05}	27056	7.05×10^{-08}
ehl_kost (2)	default	10^{-04}	3333	1.84×10^{-07}
ehl_kost (3)	default	10^{-04}	7608	2.75×10^{-08}
forcebsm	Newton	10^{-04}	100	4.43×10^{-08}
forcedsa	default	10^{-06}	10289	3.57×10^{-07}
lincont	Newton	10^{-06}	$2528^{\pm}$	5.71×10^{-13}
powell (5)	default	10^{-04}	272	4.75×10^{-07}
shubik (4, 10, 34, 36)	default	10^{-11}	$6914^{\pm}$	9.66×10^{-07}
trafelas (2)	Newton	10^{-05}	1079	6.51×10^{-09}

$\pm$ HMAX = 20.

STEPNX). NFE is the number of function (Jacobian matrix) evaluations required. For problems with multiple cases (runs), the minimum value of ARCTOL over all the runs required for success is reported. Also for such cases, maximum values of NFE and $\|F\|_\infty/(1+\|x\|_\infty)$, between different runs, are listed in the tables.

For the optimal step size estimation, default values of all the curve tracking parameters were used (as set in STEPNX). This included the maximum step size (HMAX), which is set to one in STEPNX. Results for HMAX $\neq 1$ are indicated by footnotes in the tables.

8. Conclusion and future work

This thesis extends the work done in [6] on probability-one homotopy algorithms for solving mixed complementarity problems (MCPs). The focus here is on algorithms for solving MCPs (from the MCPLIB test library) that were not solved by the probability-one homotopy algorithm of [6]. The algorithms used here are based on the Newton homotopy map and homotopy parameter embeddings. Using these methods to solve all of the remaining problems in MCPLIB demonstrates the effectiveness of homotopy methods for solving difficult MCPs. Many problems can be solved using standard mappings. When those fail, there is tremendous flexibility for choosing more sophisticated homotopy mappings that exploit qualitative characteristics of the problem at hand.

Convergence theory for both the default homotopy map and the Newton homotopy map hinges on Sard's Theorems from differential geometry. The idea behind homotopy parameter embeddings is to build the homotopy map as a family of qualitatively similar functions. This approach is similar to algorithms discussed in [21], [28], [29], [31], and is attractive because it extends the standard homotopy maps' guarantees of global convergence to a much wider class of problems.

A potential criticism of the embedding approach is that in cases where the standard mappings failed, there was a high cost, in terms of human time, in developing the more sophisticated homotopy mappings used in the homotopy parameter embedding approach. While handcrafted homotopy maps are not general, and their creation is somewhat of an art, it *is* usually the case that a particular handcrafted map works for a large *class* of problems, not just a single problem. So, within a particular application domain, economics, say, one could imagine building up over time an arsenal of homotopy maps that cover more and more classes of economic problems. Users could then easily select the appropriate mapping for the particular class of problems. Exactly this has been done in circuit design and robotics, where certain types of homotopy maps are known to work for certain types of circuits or certain types of robot mechanisms. For less sophisticated users, a reasonable strategy would be to incorporate the various mappings into a "polyalgorithm" in which all mappings would be tried simultaneously until one of them succeeds (or all fail).

Future work involves finding conditions on MCPs under which the Newton homotopy map would be globally convergent (similar to the conditions in Theorem 4.1 of [6] for the default homotopy map discussed in Section 2.2). Another avenue of research is to address difficulties associated with unbounded MCPs. This might involve developing specialized techniques for adding artificial bounds on the variables and barriers on the components of $H(x)$ in order to force a bounded zero curve.

9. Bibliography

1. E.L. Allgower and K. Georg, Numerical Continuation Methods: An Introduction, *Springer Series in Computational Mathematics*, vol. 13, 1990.
2. E. L. Allgower and K. Georg, Continuation and path following, *in Acta Numerica, A. Iserles (Ed.), Cambridge Univ. Press, Cambridge*, pp. 1–64, 1993.
3. E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, LAPACK Users' Guide, *Society for Industrial and Applied Mathematics: Philadelphia*, 3rd ed., 1999.
4. S.C. Billups, S.P. Dirkse, and M.C. Ferris, A comparison of large scale mixed complementarity problem solvers, *Computational Optimization and Applications*, vol. 7, no. 1, pp. 3–25, 1997.
5. S.C. Billups, A homotopy-based algorithm for mixed complementarity problems, *SIAM Journal on Optimization*, vol. 12, no. 3, pp. 583–605, 2002.
6. S.C. Billups and L.T. Watson, A probability-one homotopy algorithm for nonsmooth equations and mixed complementarity problems, *SIAM Journal on Optimization*, vol. 12, no. 3, pp. 606–626, 2002.
7. C. Buck, Advanced Calculus, *McGraw–Hill: New York*, 3rd ed., 1978.
8. S.P. Dirkse, M.C. Ferris, P.V. Preckel, and T.F. Rutherford, The GAMS callable program library for variational and complementarity solvers, *Mathematical Programming Technical Report 94-07, Computer Sciences Department, University of Wisconsin, Madison*, 1994.
9. S.P. Dirkse and M.C. Ferris, MCPLIB: a collection of nonlinear mixed complementarity problems, *Optimization Methods and Software*, vol. 5, pp. 319–345, 1995.
10. D. Dowson and G.R. Higginson, Elasto-Hydrodynamic Lubrication, SI Edition *Pergamon Press: Oxford*, 1977.
11. M.C. Ferris and J. Pang, Complementarity and Variational Problems: State of the Art, *Society for Industrial and Applied Mathematics: Philadelphia*, 1997.
12. M.C. Ferris and T.S. Munson, PATH 4.6, *http://www.gams.com/solvers/path.pdf*.
13. M.C. Ferris and GAMS Development Corporation, NLPEC *http://www.gams.com/dd/-docs/solvers/nlpec.pdf*.
14. GAMS On-line Documentation, Solving Constrained Nonlinear System (CNS) with GAMS, *http://www.gams.com/docs/pdf/cns.pdf*.
15. GAMS On-line Documentation, The General Algebraic Modeling System, *http://www.-gams.com/docs/intro.htm*.
16. GAMS On-line Documentation, PATH versus MILES, *http://www.gams.com/docs/pathv-smiles.htm*.
17. H.B. Keller, Global homotopies and Newton methods, *in Recent Advances in Numerical Analysis, C. De Boor and G.H. Golub (Eds.), Academic Press, New York*, pp. 73–94, 1978.
18. D. Kincaid and W. Cheney, Numerical Analysis: Mathematics of Scientific Computing, *Brooks/Cole Thomson Learning: Pacific Grove*, 3rd ed., 2002.
19. O.L. Mangasarian, Equivalence of the complementarity problem to a system of nonlinear equations, *SIAM Journal on Applied Mathematics*. vol. 31, no. 1, pp. 89–92, 1976.
20. B.A. McCarl and GAMS Development Corporation, Quick start tutorial drawn from GAMS user guide:2002, *http://www.gams.com/mccarl/tutorial.pdf*.

21. R.C. Melville, Lj. Trajković, S.-C. Fang, and L.T. Watson, Artificial parameter homotopy methods for the DC operating point problem, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 6, pp. 861–877, 1993.

22. B.A. Murtagh and M.A. Saunders, MINOS 5.0 user's guide, *Technical Report SOL 83.20, Stanford University, Stanford*, 1983.

23. J. Nocedal and S. J. Wright, Numerical Optimization, *Springer: New York*, 1st ed., 2000.

24. T. Rutherford, MILES: A mixed inequality and nonlinear equation solver, *http://www.gams.com/dd/docs/solvers/miles.pdf*.

25. S. Smale, A convergent process of price adjustment and global Newton methods, *Journal of Mathematical Economics*, vol. 3, pp. 107–120, 1976.

26. L.T. Watson, Solving the nonlinear complementarity problem by a homotopy method, *SIAM Journal on Control and Optimization*, vol. 17, no. 1, pp. 36–46, 1979.

27. L.T. Watson, J.P. Bixler, and A.B. Poore, Continuous homotopies for the linear complementarity problem, *SIAM Journal on Matrix Analysis and Applications*, vol. 10, no. 2, pp. 259–277, 1989.

28. L.T. Watson and R.T. Haftka, Modern homotopy methods in optimization, *Computer Methods in Applied Mechanics and Engineering*, vol. 74, no. 3, pp. 289–305, 1989.

29. L.T. Watson, Globally convergent homotopy algorithms for nonlinear systems of equations, *Nonlinear Dynamics*, vol. 1, no. 2, pp. 143–191, 1990.

30. L.T. Watson, M. Sosonkina, R.C. Melville, A.P. Morgan, and H.F. Walker, Algorithm 777: HOMPACK90: A suite of Fortran 90 codes for globally convergent homotopy algorithms, *ACM Transactions on Mathematical Software, http://doi.acm.org/10.1145/279232.279235*, vol. 23, pp. 514–549, 1997.

31. L.T. Watson, Probability-one homotopies in computational science, *Journal of Computational and Applied Mathematics*, vol. 140, no. 1-2, pp. 785–807, 2002.

10. Appendix A: Duopoly pricing model

Original problem

```
sets I /1*7/;
alias (J,I);
scalar delta /.9/;

parameters pi(I,J);
table pi(I,J)
        1      2      3      4      5      6      7
2              2.5    5      5      5      5      5
3                     4      8      8      8      8
4                            4.5    9      9      9
5                                   4.     8      8
6                                          2.5    5;

positive variables v(I), w(I), x(I,J);

equations f(I), g(I), h(I,J);

f(I).. sum(J, x(I,J)) - 1. =g= 0;
g(I).. w(i) - sum(J, pi(I,J)*x(I,J)) - delta *sum(J, v(J)*x(I,J)) =g= 0;
h(I,J).. v(I) - delta*w(J) - pi(J,I) =g= 0;

model duop /f.v, g.w, h.x /;

v.l(I) = 1.;
w.l(I) = 1.;
x.l(I,J) = 1.;

duop.iterlim = 40000;
solve duop using mcp;
```

Modified problem

```
sets I /1*7/;
alias (J,I);
scalar delta /.9/;

parameters pi(I,J);
table pi(I,J)
        1       2       3       4       5       6       7
2               2.5     5       5       5       5       5
3                       4       8       8       8       8
4                               4.5     9       9       9
5                                       4.      8       8
6                                               2.5     5;

variables temp(I,J);
positive variables v(I), w(I), x(I,J);

equations tempeq(I,J), f(I), g(I), h(I,J);

tempeq(I,J).. temp(I,J) - pi(J,I) =e= 0;
f(I).. sum(J, x(I,J)) - 1. =g= 0;
g(I).. w(i) - sum(J, pi(I,J)*x(I,J)) - delta *sum(J, v(J)*x(I,J)) =g= 0;
h(I,J).. v(I) - delta*w(J) - temp(I,J) =g= 0;

model duop /tempeq.temp, f.v, g.w, h.x /;

temp.l(I,J) = 0;
v.l(I) = 1.;
w.l(I) = 1.;
x.l(I,J) = 1.;

duop.iterlim = 40000;
solve duop using mcp;
```

Vita

Kapil Ahuja earned his bachelor's degree in Mechanical Engineering in May 2001 from Institute of Technology - Banaras Hindu University (IT-BHU), India. In July 2001 he joined Infosys Technologies Limited (India) as a software engineer, and subsequently worked in the software industry for three years. He started graduate studies in August 2004 towards a Master of Science degree in Computer Science and Applications at Virginia Polytechnic Institute and State University. After completing master's degree in May 2007, he starts PhD work in Mathematics at Virginia Polytechnic Institute and State University.