NUMERICAL ALGEBRAIC GEOMETRY FOR OPTIMAL CONTROL APPLICATIONS

PHILIPP ROSTALSKI*, IOANNIS A. FOTIOU†, DANIEL J. BATES‡, A. GIOVANNI BECCUTI†, AND MANFRED MORARI†

Abstract. A new technique for solving polynomial nonlinear constrained optimal control problems is presented. The problem is reformulated into a parametric optimization problem, which in turn is solved in a two step procedure. First, in a pre-computation step, the equation part of the corresponding first order optimality conditions is solved for a generic value of the parameter. Relying on the underlying algebraic geometry, this first solution makes it possible to solve efficiently and in real time the corresponding optimal control problem at the measured parameter value for each subsequent time step. This approach has a probability one guarantee of finding the *global* optimal solution at each step. Controller synthesis for two applications from the area of power electronics featuring a dc-ac converter and a dc-dc converter are discussed to motivate the proposed approach.

Key words. Optimal control, model predictive control, receding horizon control, power electronics, controller synthesis, numerical algebraic geometry, homotopy continuation.

1. Introduction. Optimal control is a very active research area with broad attention from industry [23]. It is among the few control methodologies providing a systematic way to perform nonlinear control synthesis while handling constraints. To a great extent, it is thanks to this capability of handling constraints on system states, inputs and outputs that model predictive control (MPC) has proven to be very successful in practice [12], [19].

Model predictive control uses on-line optimization to obtain the solution of a finite time optimal control problem. For the sake of simplicity and because of implementation issues on a digital control unit, the optimal control problem is typically cast into a discrete time mathematical program, whose solution yields a sequence of optimal control moves. To ensure feedback, i.e. to deal with disturbances and model uncertainties, only the first move of the sequence is actually applied to the plant, and the optimization is repeated with new data measurements. This scheme is known as receding horizon control (RHC). The algorithm in this article is intended for such deterministic, discrete-time optimal control problems.

Unfortunately, technology and cost factors make the implementation of receding horizon control difficult if not, in certain cases, impossible. To circumvent these issues, the solution of the optimal control problem can, in some cases, be computed off-line, by solving the corresponding mathematical program parametrically. That is, the explicit formula giving the solution of the program (control inputs) as a closed-form expression of the problem parameters (measured state) is computed. The solution then is efficiently implemented on-line, e.g., as a lookup table [6]. This approach is also known as the "explicit" MPC solution technique.

While this approach is feasible for the class of linear systems and piece-wise affine (hybrid) systems, no closed form expression of the optimal solution exists in the general nonlinear polynomial case, as it necessarily involves implicit algebraic

*Department of Mathematics, University of California, Berkeley, CA 94720-3840 USA (Email: philipp@math.berkeley.edu). This work was performed while at the Automatic Control Laboratory, ETH Zurich, see below.

†Automatic Control Laboratory, ETH Zurich, Physikstrasse 3, 8092 Zurich, Switzerland (Email: {beccuti,morari}@control.ee.ethz.ch, ifotiou@gmail.com).

‡Department of Mathematics, 101 Weber Hall, Colorado State University, Fort Collins, CO 80528 USA (Email: bates@math.colostate.edu). Partially supported by NSF grant DMS–0914674.

functions. Nonetheless, by combining a pre-computation stage involving algebraic and numerical techniques with an on-line stage involving numerical computations, it is still possible to achieve efficient algorithms for computing the optimal solution for nonlinear polynomial problems [4], [11] and, to some extend, also for piecewise polynomial problems [10].

In contrast to other parametric optimization approaches based on continuation suggested mainly in the late nineties (see [1] or [30]), the approach followed in the authors' previous work [4] and in this article is based on polynomial homotopy continuation over the field of *complex* numbers. The present method also bears the advantage of having a deep and rich supporting theory which can be exploited in various ways [20], [22]. Even though the new approach is not necessarily faster than other homotopy continuation methods, it is guaranteed to find the globally optimal solution and yields a general purpose method for solving the class of polynomial nonlinear constrained optimal control problems considered here as well as more general parametric polynomial optimization problems.

In the present work we give a detailed description of the algorithm and the background material needed. In Section 2 we give a brief primer on modern algebraic continuation methods, before we set the stage of optimal control in Section 3, while elaborating on the proposed algorithm. Afterwards, in Sections 4 and 5, we describe the details of two applications and report the results of our numerical simulations.

1.1. Related Work. The author of [21] proposed a homotopy continuation scheme for the solution of nonlinear receding horizon control problems, making use of approximate Jacobian matrices and fast linear algebra (GMRES). While that method is tailored to receding horizon control problems, the one described here extends to the more general setting of parametric optimization. The main advantage of our approach is that it yields a probability one *guarantee* of convergence to the *global* optimal solution at each time step. The interested reader is referred to [24], Chapter 4 for precise statements regarding probability one guarantees.

The method in [21] differs also in that it deals with general nonlinear systems, while we restrict to the case of polynomial systems. Though more restrictive, this allows us to exploit the intrinsic structure of such systems [24]. This gives rise to the second main advantage of our method, which is the ability to deal with singularities on paths, whereas in [21] it must be assumed that homotopy paths remain nonsingular.

One further difference between the approaches is the fact that we use derivative information derived from the dynamics of the discrete-time system. The author of [21] uses the derivative information from the continuous-time system to Euler-predict the next step, with no Newton correction afterwards, which raises issues regarding the numerical accuracy of the method. In contrast, by using Newton correction steps, we are able to obtain arbitrarily high accuracy at the cost of more computational resources, as detailed in [5]. In general, Ohtsuka's method will be more efficient computationally since at each step only a single path needs to be traced. However the disadvantages of possibly not finding the global optimum or even getting stuck in numerical errors may outweigh this performance surplus.

The author of [30] provides a detailed repertoire of techniques for solving general optimal control problems of many different varieties. The existence of specialized methods for the case in which everything is polynomial is mentioned only in passing, and the supporting references provide no further details on any methods specific to this setting. The techniques of this paper therefore fill that gap in the literature by providing a specific procedure for harvesting the power of algebraic geometry to

solve problems of this type. The algebraic setting under consideration here is far more powerful, albeit more restrictive, than the general setting considered in [30], particularly since the ubiquitous convergence concerns of that article are of no concern in the algebraic setting.

2. Homotopy Continuation. In this section, we give a brief overview about polynomial homotopy continuation, a numerical technique for computing solutions of a system of polynomial equations. The book of Sommese and Wampler [24] provides a modern reference for this algebraic setting while the book of Allgower and Georg [2] treats continuation in the general nonlinear case.

2.1. Basic scenario. Consider a system of polynomial $f : \mathbb{C}^r \to \mathbb{C}^r$ with r equations $f_i(z)$, $i = 1, \ldots, r$ in r variables $z_i \in \mathbb{C}$, $i = 1, \ldots, r$, where $z = (z_1, \ldots, z_r) \in \mathbb{C}^r$ and $f_i(z_1, \ldots, z_r) \in \mathbb{C}[z_1, \ldots, z_r]$ for all i. Homotopy continuation computes the solutions of a given polynomial system $f(z) = 0$ (the target system), starting from the *known* solutions of another polynomial system $g(z) = 0$ (the start system) for some $g(z) : \mathbb{C}^r \to \mathbb{C}^r$. More specifically, homotopy continuation constructs a family of square polynomial systems $H(z, t) : \mathbb{C}^r \times \mathbb{R} \to \mathbb{C}^r$ (known as "homotopy" or "homotopy function") with

$$H(z, t) = (1 - t) \cdot f(z) + \gamma \cdot t \cdot g(z)\,, \tag{2.1}$$

where $z \in \mathbb{C}^r$ is the unknown variable, $t \in [0, 1]$ is a scalar called the *continuation parameter* and $\gamma \in \mathbb{C}$ is a non-zero random complex number that ensures certain numerical properties, see [24] for details.

For $t = 1$, $H(z, 1) = \gamma g(z)$, the solutions of which are known solutions of the start system $g(z) = 0$. As t decreases from 1 to 0, the homotopy function $H(z, t)$ is deformed to finally take the form of $H(z, 0) = f(z)$, the system that we aim to solve. As t varies from 1 to 0, the solutions of $H(z, t) = 0$ deform continuously so that there is one *solution curve* $z^{(i)}(t)$ extending from each isolated solution z_i of the start system $g(z) = 0$ to a solution of the target system, see Figure 2.1. For the sake of simplicity, we only consider here isolated solutions, for the general positive dimensional case, see [24].

2.2. Path Following. In order to follow the solution path $z(t) \in \mathbb{C}^r$, typically a prediction-correction scheme is employed. Therefore, we start by taking the absolute derivative of the homotopy equation $H(z(t), t) = 0$ with respect to parameter t, and obtain the so-called Davidenko differential equation [8]

$$\mathcal{J}(z(t), t) \cdot \frac{dz(t)}{dt} + \frac{\partial H(z(t), t)}{\partial t} = 0\ , t \in (0, 1]\,, \tag{2.2}$$

where $\mathcal{J}(z(t), t) = \left[\frac{\partial H_i(z,t)}{\partial z_j}\right] \in \mathbb{C}^{r \times r}$ is the Jacobian matrix of $H(z, t)$ with respect to $z \in \mathbb{C}^r$ and $\frac{dz(t)}{dt} = \left[\frac{dz_i(t)}{dt}\right] \in \mathbb{C}^r$ is the absolute derivative of $H(z, t)$ with respect to t. Equation (2.2) can be rewritten in the form

$$\dot{z}(t) = -\mathcal{J}^{-1}(z(t), t)\frac{\partial H(z(t), t)}{\partial t}\,, t \in (0, 1]\,, \tag{2.3}$$

where $\dot{z}(t) = \frac{dz(t)}{dt}$. The nonlinear differential equation (2.3) can now be solved to obtain points on the solution paths $z(t)$, $t \in (0, 1]$ of the homotopy equation $H(z(t), t) = 0$. We will simply use a (first order) Euler-Newton prediction-correction scheme, see e.g. [2], but higher order methods are obviously also applicable.

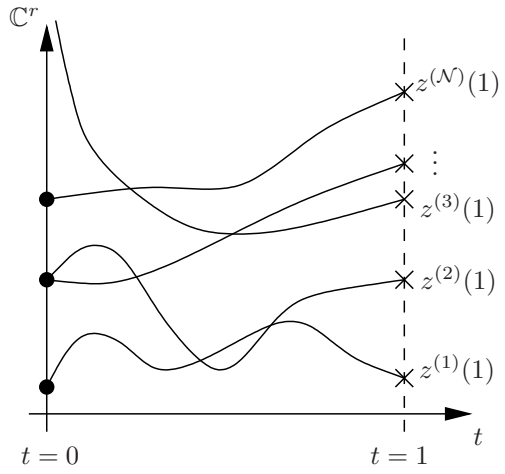


FIG. 2.1. *Solution paths: Each cross (×) represents a solution of the start system $g(z)$, black dots (•) those of the target system $f(z)$. As the solution paths are tracked from $t = 1$ to $t = 0$, some solution paths escape to infinity (which is made finite in practice by working over projective space), while others might arrive at identical solutions of the target system. With probability one, paths do not cross each other for $t \in [0, 1)$. (Note that the paths depicted above are drawn over $\mathbb{R}^1$ rather than $\mathbb{C}^r$, hence the depicted"crossings".)*

The Euler-Newton Scheme. After choosing an appropriate step size $\Delta t = t_j - t_{j+1}$ (either fixed or adaptively), we start from an approximation of a known solution $z^{(i)}(t_j)$ at the current value t_j of t and Euler-predicts the value $z^{(i)}(t_{j+1})$ of the solution curve $z^{(i)}(t)$ at $t_{j+1} = t_j - \Delta t$, stepping ahead along the tangent

$$z(t_{j+1}) = z(t_j) + \dot{z}(t_j)\Delta t\,, \tag{2.4}$$

In this way, we obtain an approximation of the solution $z^{(i)}(t)$ at $t = t_{j+1}$.

Subsequently, and within the same iteration step, we also perform a sequence of correction steps, in order to refine the initial approximation $z^{(i)}(t_{j+1})$. The correction is done by applying Newton's method for the solution of the nonlinear equation $H(z, t_{j+1}) = 0$, with t_{j+1} held constant.After a few Newton steps , we either obtain the corrected value of the solution $z(t)$ at point $t_{j+1} = t_j - \Delta t$ or we decide that convergence is too slow, and restart with a smaller step size.

Repeating these steps leads to a solution path from one solution of the start system $H(z, 1) = g(z)$ to one solutions of the system $H(z, 0) = f(z)$. By performing this procedure for all $\mathcal{N}$ solutions of the start system $g(z)$, we obtain $\mathcal{N}$ trajectories $z^{(i)}(t), i = 1, \ldots, \mathcal{N}$ for the homotopy (2.1).

2.3. Start Systems. In our discussion so far, we have assumed that we know a suitable start system $g(z)$ and its solutions. In practice, however, such a system is rarely given and if one is, its solutions are not known *a priori*.

Number of Solutions. In order to find all solutions of the target system, the start system must have at least as many solutions. On the other hand, a small number of solutions is desirable, to keep the computational burden low as every solution yields a path to be tracked. To ensure that, both $g(z)$ and $f(z)$ must share a common structure, i.e. $f(z)$ and $g(z)$ are both members of a family of polynomial systems $\mathcal{F}_\sigma$, parameterized by some parameter $\sigma \in \mathbb{C}^{n_\sigma}$ and $g(z)$ needs to be generic in $\mathcal{F}_\sigma$.

THEOREM 2.1 (Parameter continuation ([24], p. 92)). *Let $\mathcal{F}_\sigma(z)$ be a system of polynomials in r variables and n_σ parameters, $\mathcal{F}_\sigma(z) : \mathbb{C}^r \times \mathbb{C}^{n_\sigma} \to \mathbb{C}^r$, that is, $\mathcal{F}_\sigma(z) = \{f_1(z;\sigma), f_2(z;\sigma), \ldots, f_r(z;\sigma)\}$, and each $f_i(z;\sigma)$ is polynomial in both z and σ. Furthermore, let $\mathcal{N}_\sigma$ denote the number of non-singular solutions of $\mathcal{F}_\sigma(z)$ as a function of the parameter σ:*

$$\mathcal{N}_\sigma = \#\left\{z \in \mathbb{C}^r \,\middle|\, \mathcal{F}_\sigma(z) = 0,\ det\big(\frac{\partial \mathcal{F}_\sigma}{\partial z}(z;\sigma)\big) \neq 0\right\}.$$

Then, $\mathcal{N}_\sigma$ is finite and it is the same, say $\mathcal{N}_0$ for almost all $\sigma \in \mathbb{C}^{n_\sigma}$.

The set of parameters σ for which $\mathcal{N}_\sigma \neq \mathcal{N}_0$ lies on an algebraic set $\chi \in \mathbb{C}^{n_\sigma}$ of dimension less than n_σ and thus on a set of measure zero. Thus, a generic (random) choice of the parameter σ will (almost always) result in a polynomial system $\mathcal{F}_\sigma(z)$ having $\mathcal{N}_0$ non-singular solutions. Furthermore, for $\sigma \in \chi$, $\mathcal{N}_\sigma < \mathcal{N}_0$. Thus, to build a suitable start system for a given polynomial system $f(z)$, we need only to construct a parameterized family $\mathcal{F}_\sigma$ of systems for which $\mathcal{N}_0$ is certainly larger than the number of non-singular solutions of $f(z)$.

2.4. Homotopy Classes and Start Systems.

Naturally parameterized family. Sometimes the target system $f(z)$ comes with a *naturally parameterized family*, denoted by $\mathcal{F}_\sigma^{\text{nat}}$, i.e., we have a parametric system of polynomials and want to find the zero-set of one of its members. In this special case, we may use a *coefficient parameter homotopy*. However, a generic start system $g(z) \in \mathcal{F}_\sigma^{\text{nat}}$ for which the solutions to $g(z) = 0$ are known is not necessarily available, and it might be more difficult to find one than to solve the original problem.

The chief virtue of naturally parameterized families of polynomial systems is that once the solutions are known for a single (generic) instance, i.e., a single point in the parameter space, moving to another instance is more computationally efficient than any other kind of homotopy.

Furthermore, there are several classes of artificial families in the spectrum between so-called total degree start systems, which are particular easy to be solved but yield many redundant paths, and coefficient parameter polyhedral homotopies ([24], Chapter 8), which nearly fully exploit the structure of $f(z)$. Each achieves a different tradeoff between these two aspects. For the applications in this paper, we use multihomogeneous families which have shown to be a good compromise.

Multihomogeneous Homotopy. Consider the target system $f(z) : \mathbb{C}^r \to \mathbb{C}^r$ in r variables. We partition the r variables $z_1, \ldots, z_r$ into ν disjoint variable groups Z_i, $i = 1, \ldots, \nu$ of sizes $r_1, \ldots, r_\nu$ ($r = r_1 + \cdots + r_\nu$) as follows:

$$\{Z_1, \ldots, Z_\nu\},\ Z_i = \{z_j\}_{j \in \mathcal{I}_i},\ i = 1, \ldots, \nu,$$

where $\mathcal{I}_i$, $i = 1, \ldots, \nu$ is an index set indicating which of the variables z_j, $j = 1, \ldots, r$ belong to the group Z_i, $i = 1, \ldots, \nu$. Denote further with $d_{ki}, k = 1, \ldots, r$, $i = 1, \ldots, \nu$ the degree of the kth polynomial $f_k(z)$ in the target system with respect to the ith group of variables Z_i. Based on a partition $\{Z_1, \ldots, Z_\nu\}$, we adapt the notation from [24] and write every polynomial of the target system in the form

$$f_k(z) = \sum_{\substack{\{\alpha_1, \ldots, \alpha_\nu\} \\ |\alpha_i| \leq d_{ki}}} c_{\{\alpha_1, \ldots, \alpha_\nu\}} Z_1^{\alpha_1} \cdots Z_\nu^{\alpha_\nu}\,, \tag{2.5}$$

where each α_i is a vector of exponents. Expression (2.5) is simply a compact way of recording the polynomials $f_k(z), k = 1, \ldots, r$ together with its structure.

Let $\langle Z_i, 1\rangle$ denote any possible affine combination of the variables in group Z_i. The start system of a multihomogeneous homotopy associated with a target system $f(z)$ for a given partition of the variables $\{Z_1, \ldots, Z_\nu\}$ (called also a ν-homogeneous homotopy) is given by r polynomials

$$g_k(z) \in \langle Z_1, 1\rangle^{d_{k1}} \times \cdots \times \langle Z_\nu, 1\rangle^{d_{k\nu}},\ k = 1, \ldots, r\,. \tag{2.6}$$

That is, polynomial $g_k(z)$ is the product of affine factors $\langle Z_i, 1\rangle$, where we have d_{ki} (different) factors of the variables Z_i. In this way, we produce a start system that shares the same multidegree structure (with respect to a certain partition of the variables) as the original polynomial system. The target system (2.5) is a member of the family of (2.6), [24, p. 128] and the implication of Theorem 2.1 is that by starting from a generic system of the form (2.6), we will find all solutions of $f(z)$ (with probability one). It should be noted that the agreement in root counts does not imply that the actual number of roots will be the same. By construction the start system always has the maximum, Bézout-like, number of solutions while the target system usually will have considerably fewer.

It is essential to be able to find the solutions of the start system in an efficient manner. Fortunately, for start systems of the form (2.6), solving for the start points is simply a matter of combinatorially cycling through all ways of choosing one linear factor from each polynomial $g_k(z)$, $k = 1, \ldots, r$ of the start system. For each, we must only solve the corresponding linear system. Once we have done so for all possible combinations of linear factors, we have found all solutions of start system $g(z)$.

The choice of the partition $\{Z_1, \ldots, Z_\nu\}$ of the variables into groups in a multihomogeneous homotopy is crucial, because different partitions lead to different number of compatible linear systems to be solved and hence to different number of solution paths to be tracked. There exist both heuristic and systematic approaches for the choice of the partition, and there are also more sophisticated (and perhaps more costly) polyhedral methods that often have close to the minimum number of paths. For more details, the reader is referred to [28], [29] and ([24], Chapter 8).

3. Optimal Control.

3.1. Parametric optimization. In the optimal control context, we want to minimize a given polynomial objective function $J(z, x_0) \in \mathbb{R}[z, x_0]$ with respect to a decision-variable vector $z \in \mathcal{Z}$ for any given measurement of the initial state $x_0 \in \mathcal{X} \subseteq \mathbb{R}^n$, where $\mathcal{X}$ is the set of admissible states.

For the class of control problems we are looking at in this paper, i.e. systems described by polynomial vector fields and subject to (basic) semi-algebraic state or input constraints, the optimal control problem can be formulated as a polynomial parametric optimization problem

$$J^*(x_0) = \min_z\ J(z, x_0) \quad \text{s.t.} \quad \begin{cases} q(z, x_0) \le 0 \\ h(z, x_0) = 0\,, \end{cases} \tag{3.1}$$

where $q \in \mathbb{R}[z, x_0]^{n_q}$ and $h \in \mathbb{R}[z, x_0]^{n_\lambda}$ are vector polynomial functions representing the inequality and equality constraints, respectively, defining the set $\mathcal{Z}$ of feasible z for a given parameter x_0, see e.g. [11] for details.

The polynomial parametric optimization problem is to find an efficient computational procedure for evaluating the maps

$$\begin{array}{rccc} z^*(x_0): & \mathbb{R}^n & \longrightarrow & \mathbb{R}^r \\ & x_0 & \longmapsto & z^* \end{array} \qquad \begin{array}{rccc} J^*(x_0): & \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ & x_0 & \longmapsto & J^*, \end{array} \tag{3.2}$$

where $z^*(x_0) = \arg\min_{z \in \mathcal{Z}} J(z, x_0)$ and $J^*(x_0) = \min_{z \in \mathcal{Z}} J(z, x_0)$.

3.2. Posing the problem. We assume throughout the paper that the feasible set defined by $q(z, x_0)$ and $h(z, x_0)$ is compact and the minimum is attained. Furthermore we assume certain regularity conditions (so-called constraint qualifications*, see [14]) such that the optimizer is attained at a Karush-Kuhn-Tucker (KKT) point [7], which is a point in the solution set of the system

$$\begin{array}{rcl} \nabla_z J(z, x_0) + \sum_{i=1}^{n_q} \mu_i \nabla_z q_i(z, x_0) + \sum_{i=1}^{n_h} \lambda_i \nabla_z h_i(z, x_0) & = & 0 \\ h(z, x_0) & = & 0 \\ \mu_i q_i(z, x_0) & = & 0 \\ \mu_i & \geq & 0 \\ q(z, x_0) & \leq & 0\,. \end{array} \tag{3.3}$$

For the class of problems we consider, the first three relations of the KKT conditions (3.3) form a *square system of polynomial equations* in z, μ and λ.

3.3. Methods for solving polynomial equations. One way of solving the optimization problem (3.1) for all possible values of parameter x_0 would be to presolve the corresponding KKT-system parametrically – following the philosophy of moving as much as possible of the computational burden off-line, leaving an easy task for the on-line implementation.

Various methods have been proposed in the literature for solving systems of polynomial equations, both symbolic (e.g. [16], [27]) and numerical (e.g. [26], [24]). It should be noticed, though, that the KKT-system (3.3) contains not only variables, but also certain parameters $x_0 \in \mathbb{R}^n$.

Symbolic methods based on comprehensive Gröbner bases [31] or resultants can deal with parametric problems arising in parametric optimal control, but tend to be quite expensive when it comes to computations, [11]. Specifically, these techniques suffer from what is known as *coefficient blowup* and their pre-computation phase can take a considerable amount of time.

In the present work, we reduce computational complexity by using homotopy continuation for polynomial systems [24], [4] as the workhorse of the proposed method both in a (offline) precomputing phase as well as during the online execution.

3.4. The Approach.

Off-line Part. During the design phase, i.e. in the off-line part of the algorithm we concentrate on the first three relations in the KKT condition (3.3)

$$\mathcal{F}_{x_0}^{\text{nat}}(\underbrace{z, \mu, \lambda}_{\rho}): \quad \left\{ \begin{array}{rcl} \nabla_z J(z, x_0) + \sum_{i=1}^{n_q} \mu_i \nabla_z q_i(z, x_0) & & \\ + \sum_{i=1}^{n_h} \lambda_i \nabla_z h_i(z, x_0) & = & 0 \\ h(z, x_0) & = & 0 \\ \mu_i q_i(z, x_0) & = & 0\,, \end{array} \right. \tag{3.4}$$

*Constraint qualifications are non-trivial assumptions. Nonetheless, they are satisfied in many practical situations and are often assumed in parametric optimization. In case these assumptions are not satisfied, a similar approach is possible using the Fritz-John optimality conditions, [15].

where λ_i, $i = 1, \ldots, n_h$ and μ_i, $i = 1, \ldots, n_q$ are the Lagrange multipliers associated with the equality and inequality constraints, respectively. The system (3.4) forms a naturally parametrized family $\mathcal{F}_{x_0}^{\text{nat}}$ of square systems of $r + n_q + n_h$ polynomial equations parametrized by x_0, where $\rho = (z, \mu, \lambda) \in \mathbb{C}^{r+n_q+n_h}$. We now use homotopy continuation and compute all (real and complex) solutions of system (3.4).

ASSUMPTION 3.1 (Zero-dimensional assumption[†]). *We assume that the solution of system* (3.4) *is a set of isolated points. In other words, the corresponding variety* $\mathcal{V}$ *is zero-dimensional.*

The off-line part of the algorithm involves solving a generic instance $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$ in family $\mathcal{F}_{x_0}^{\text{nat}}$, for a generic (i.e., random complex) value $\tilde{x}_0 \in \mathbb{C}^n$ of the parameter x_0, thus we obtain an instance $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$. To solve $f_{\tilde{x}_0} = 0$, we have to embed it into a (larger) suitably parametrized family of polynomials, in which we know (or can easily find) a start system $g(\rho)$, whose solutions are available. We choose to embed it in the multihomogeneous family of systems $\mathcal{F}_{x_0}^{\text{m-hom}}$, as described in Section 2.3. Using homotopy continuation we track the solutions of $g(\rho) \in \mathcal{F}_{x_0}^{\text{m-hom}}$ to the set of solutions for $f_{\tilde{x}_0}(\rho) = 0$, denoted by $\tilde{\mathcal{P}} = \left\{\tilde{\rho}^{(j)},\, j = 1, \ldots, \tilde{n}_\rho\right\}$.

A graphical illustration of the off-line part of the proposed method can be seen in Figure 3.1(a) and is described in Algorithm 1.

Algorithm 1 Off-line part: Compute the solution of a generic instance $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$ of polynomial systems.

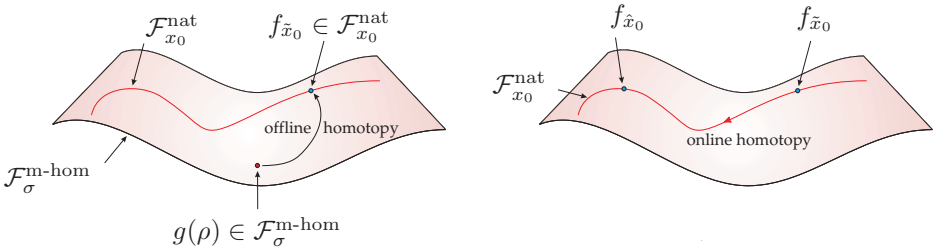
Input: Parameterized family of polynomial systems $\mathcal{F}_{x_0}^{\text{nat}}$
Output: Solutions $\tilde{\mathcal{P}} = \left\{\tilde{\rho}^{(j)} \in \mathbb{C}^{r+n_q+n_h},\, j = 1, \ldots, \tilde{n}_\rho\right\}$
1: Choose a random $\tilde{x}_0 \in \mathbb{C}^n$ and form instance $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$
2: Embed $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$ in a multihomogeneous family $\mathcal{F}_{\sigma}^{\text{m-hom}}$ and choose an appropriate start system $g(\rho) \in \mathcal{F}_{\sigma}^{\text{m-hom}}$.
3: Solve $g(\rho)$ (as in described in Section 2.4).
4: Construct and solve multihomogeneous homotopy from start system $g(\rho)$ to generic system $f_{\tilde{x}_0}(\rho)$, so as to obtain its solutions $\tilde{\rho}^{(j)}$, $j = 1, \ldots, \tilde{n}_\rho$.

On-line Part. Given the solutions $\tilde{\mathcal{P}} = \left\{\tilde{\rho}^{(j)},\, j = 1, \ldots, \tilde{n}_\rho\right\}$ of system $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$, the on-line part of the approach takes the measured value $x_0 = \hat{x}_0$ of the parameter x_0 and computes the optimum $J^*(\hat{x}_0)$ and an optimizer $z^*(\hat{x}_0)$ using the pre-computed solution from the offline part.

Given the on-line state measurement $x_0 = \hat{x}_0$, we can construct a coefficient parameter homotopy from $\tilde{x}_0$ to $\hat{x}_0$, thereby solving instance $f_{\hat{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$. This can be done efficiently, because we have already computed the solutions to $f_{\tilde{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$ in the same family during the off-line part of the algorithm and therefore only need to track a minimum number of solutions. In Figure 3.1(b) we graphically see the coefficient parameter homotopy that solves instance $f_{\hat{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$, the solutions of which we denote with $\mathcal{P} = \left\{\rho^{(j)},\, j = 1, \ldots, n_\rho\right\}$.

After system $f_{\hat{x}_0} \in \mathcal{F}_{x_0}^{\text{nat}}$ has been solved, we check all its solutions $\rho^{(j)} \in \mathcal{P} \subset \mathbb{C}^{r+n_q+n_h}$, $j = 1, \ldots, n_\rho$ and keep the ones that are both real and feasible with respect to the inequalities in (3.3).

[†]see ([24], Chapter 15) on how to employ homotopy continuation for positive-dimensional problems.



(a) Off-line part (b) On-line part

FIG. 3.1. *(a) Off-line part of the approach: After the naturally parametrized family of polynomials $\mathcal{F}^{nat}_{x_0}$ (curve) is embedded into the multihomogeneous family $\mathcal{F}^{m\text{-}hom}_{\sigma}$ (surface) containing a suitable start system $g(\rho)$, homotopy continuation is used off-line from $g(\rho)$ to the generic member $f_{\tilde{x}_0}(\rho) \in \mathcal{F}^{nat}_{x_0}$, in order to compute and store the latter's solutions $\tilde{\mathcal{P}} = \{\tilde{\rho}^{(j)},\, j = 1, \ldots, \tilde{n}_\rho\}$. (b) On-line part of the proposed approach: A coefficient-parameter homotopy is used to solve system $f_{\hat{x}_0} \in \mathcal{F}^{nat}_{x_0}$, starting from system $f_{\tilde{x}_0} \in \mathcal{F}^{nat}_{x_0}$, whose solutions have already been computed in the off-line part of the approach, and moving through $\mathcal{F}^{nat}_{x_0}$.*

Let

$$S = \{\rho^{(j)} = (z^{(j)}, \mu^{(j)}, \lambda^{(j)}) \in \mathcal{P} \cap \mathbb{R}^{r+n_q+n_h},\, j = 1, \ldots, n_\rho \mid q(z^{(j)}, \hat{x}_0) \leq 0,\, \mu^{(j)} \geq 0\}\,. \tag{3.5}$$

be the set of real feasible solutions with respect to the constraints in (3.3) at $x_0 = \hat{x}_0$ and S_z its projection on $z^{(j)}$- coordinates. The optimizer $z^*(\hat{x}_0)$ for the optimization problem (3.1) with $x_0 = \hat{x}_0$ can then be determined by selecting that $z \in S_z$, which induces the minimum cost in the objective function, i.e., $z^*(\hat{x}_0) = \arg\min_{z \in S_z} J(z, \hat{x}_0)$, whereas the optimal cost is $J^*(\hat{x}_0) = \min_{z \in S_z} J(z, \hat{x}_0)$. Note that this last step is nothing more but a search over a finite set, by the assumption of zero-dimensionality, see above. A summary of the on-line part of the proposed approach is given in Algorithm 2.

Algorithm 2 On-line part: Requires system $f_{\tilde{x}_0} \in \mathcal{F}^{\text{nat}}_{x_0}$ and its solutions $\tilde{\mathcal{P}} = \{\tilde{\rho}^{(j)},\, j = 1, \ldots, \tilde{n}_\rho\}$.

Input: Parameter value $\hat{x}_0$.
Output: Optimal cost $J^*(\hat{x}_0)$ and an optimizer $z^*(\hat{x}_0)$.

1: Construct the coefficient parameter homotopy from $f_{\tilde{x}_0} \in \mathcal{F}^{\text{nat}}_{x_0}$ to $f_{\hat{x}_0} \in \mathcal{F}^{\text{nat}}_{x_0}$ and solve it to obtain solutions $\mathcal{P} = \{\rho^{(j)},\, j = 1, \ldots, n_\rho\}$ of $f_{\hat{x}_0} = 0$.
2: Keep the real feasible solutions respecting the inequalities of the optimality conditions and store them in set S.
3: Form the discrete set S_z of all candidate optimizers.
4: Compare values $J(z, \hat{x}_0)$ for all $z \in S_z$ and select optimizer $z^*(\hat{x}_0)$ and corresponding optimal cost $J^*(x_0)$.

REMARK 3.2 (Test implementation). *The method described in this paper has been implemented in Matlab using the software package Bertini [5] for the homotopy continuation. Due to the lack of a Matlab interface the communication is done via text files which naturally introduces some read/write overhead.*

3.5. Illustrative example. Before we show the application of the proposed method in two power electronics case studies, we will illustrate the basic principle of the approach as well as potential extensions on a simple example.

We consider the Duffing oscillator, given by

$$\ddot{y}(t) + 2\zeta\dot{y}(t) + y(t) + y(t)^3 = u(t), \tag{3.6}$$

where $t \in \mathbb{R}^+$ is time, $y \in \mathbb{R}$ is the continuous state variable and $u \in \mathbb{R}$ the control input, see [13].

. This example is also considered in [4], but in a different setting. A corresponding state space description is given by

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -1 & -2\zeta \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ -x_1^3(t) \end{bmatrix}}_{f(x(t),u(t))}.$$

The parameter ζ is the damping coefficient and is assumed to be known (here $\zeta = 0.3$). The control objective is to regulate the state to the origin. To derive the discrete time model, we use a modified Euler approximation with a sampling period of $h = 0.05$ time units, i.e.

$$\begin{aligned} x(k+1) &= \tilde{f}(x(k), u(k), x(k+1), u(k+1)) \\ &= x(k) + \frac{h}{2}\left(f(x(k), u(k)) + f(x(k+1), u(k+1))\right) \end{aligned}$$

(by slight abuse of notation we denote by $x_i(t)$ the continuous time state and by $x_i(k)$ its discrete time counterpart).

An optimal control problem with prediction horizon $N = 3$, weight matrices $Q = \mathrm{diag}(1, 1)$, $R = [0.1]$ and state constraints $\|x(k+j)\|_\infty \leq 4 \;\; \forall j = 1, \ldots, N$, leads to the optimization problem

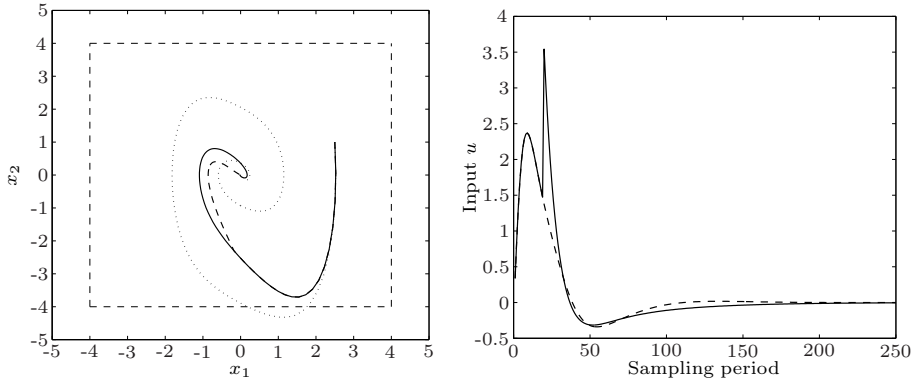
$$J^* = \min_{(u,x)} \sum_{i=1}^{N} \left[\, x_1(k+i),\; x_2(k+i) \,\right] Q \begin{bmatrix} x_1(k+i) \\ x_2(k+i) \end{bmatrix} + \sum_{i=0}^{N-1} u(k+i) R u(k+i)$$

$$\text{s.t.} \begin{cases} ||x(k+j)||_\infty \leq 4\,, \; \forall j = 1 \ldots N \\ x(k+1+j) = \tilde{f}(x(k+j), u(k+j), x(k+1+j), u(k+1+j))\,. \end{cases} \tag{3.7}$$

We can now apply the method outlined above and form the system of KKT conditions (3.3). The parameters in this conditions are the values of the initial state $x(0) = [x_1(0), x_2(0)]$. For the offline part, we fix $x(0) = \tilde{x}_0$ to random values and solve the resulting polynomial system using a certain multihomogeneous start system. This yields a total of 13251 paths out of which only one leads to a finite solution.

The system of KKT equations provides the start system for the online run. At every time step we track the solutions for this system to the solution of the KKT system for the actual parameter value by tracking a single path online (rather than 13251 each time).

Online retuning. Since the equations we aim to solve form a parametric family of equations, we can easily extend the method above and add additional parameters into the problem formulation. It might be desirable to change the tuning of the controller during the operation, e.g. change the constraints or the weight matrices Q and R online. In the proposed approach we can easily account for that by considering



(a) State-space diagram (b) Input evolution

FIG. 3.2. *Duffing oscillator, free response (...), control with fixed weight R (—) and a weight dropping after 1s (– –)*

the values we want to change at run-time as additional parameters and thus enlarging our parametric family. In the offline run, we also assign random values to those parameters and track the parameter online to desired values given by the operator.

In Figure 3.2(a) the dynamical response of the system is given for a prediction horizon of $N = 3$ and a fixed weight $R = \frac{1}{10}$ as well as for a variable weight R which switches from $R = \frac{1}{10}$ to $R = \frac{1}{30}$ at time $t = 1\,\mathrm{s}$. The corresponding optimal control inputs can be seen in Figure 3.2(b).

The affect of the rapidly changing weight can be clearly seen in both figures. Obviously as we decrease the weight on the input energy, the state is regulated to the origin much more direct, i.e. using a more aggressive input signal.

4. Power Electronics Application I.

4.1. System description. In this section the controller synthesis problem of a dc-ac converter is considered, for the specific case of a monophase voltage source inverter connected to a power grid through an LC filter, see Figure 4.1.

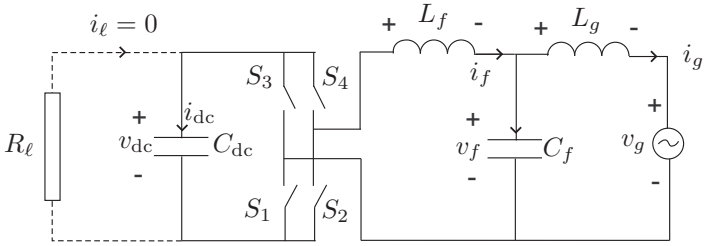


FIG. 4.1. *Topology of the dc-ac converter. C_{dc} is the dc-link capacitor, L_f, C_f denote the filter components and L_g accounts for the inductance of the connection to the grid, represented by the time-varying voltage generator v_g.*

The dc-link capacitor C_{dc} feeding the load R_ℓ supplies the slowly varying (ideally dc, i.e. constant) voltage v_{dc} that must be appropriately modulated by the switching stage (comprising switches S_1, S_2, S_3, and S_4) to track a reference value $i_{\mathrm{g,ref}}$ for the grid current i_g, which is injected into the external grid represented by the series connection of the inductor L_g with the ideal sinusoidal voltage generator v_g. The

filter elements L_f, C_f are interposed to eliminate the high frequency ripple on the resulting grid current. The load i_ℓ can be modeled at the simplest level as a constant current source: since this simply introduces a constant term in the system and model equations, without loss of generality it is assumed that $i_\ell = 0$ in the following. The semiconductor switches S_1 and S_4 are dually operated with respect to S_2 and S_3 so that the current $i_{\rm dc}$ features the two alternative circulation paths: mode 1 with switches S_2, S_3 on and S_1, S_4 off and mode 2, where S_2, S_3 are off and S_1, S_4 on.

The modulation pattern of the switches is shown in Figure 4.2, where in particular the duty cycle $d(k)$ at sampling instant k, i.e. the control input, defines *when* the transition is made between the two modes as a fraction of the constant switching period T_s. On the basis of this principle of operation, $d(k)$ is physically constrained to lie in the interval $[0, 1]$. At the beginning of each period k (i.e. the period between instants kT_s and $(k+1)T_s$) the converter is in mode 2, switches S_1 and S_4 are conducting, S_2 and S_3 are not, and the controller selects the duty cycle $d(k)$, determining the instant $k + \frac{1-d(k)}{2}$ when the transition to mode 1 takes place. Subsequently, at instant $k + \frac{1+d(k)}{2}$, S_1, S_4 are restored to conduction, S_2, S_3 are switched off and the procedure is periodically repeated at every sampling period. Notice therefore that the control action solely consists in deciding when the system is switched between its two possible operating topologies according to the prescribed modulation pattern.

It is then possible to steer the system by appropriately selecting $d(k)$ in order to track the selected time-varying reference value $i_{g,ref}(t)$ for i_g. The reference value is typically determined by a higher level controller so as to compensate for current harmonics which would otherwise be injected into the grid.

For the two possible modes of operation the evolution of the system can be respectively described in terms of the following set of differential equations:

$$\dot{x}(t) = F_1 x(t) + f_1 v_g(t), \qquad S_2, S_3 \text{ are on} \tag{4.1a}$$
$$\dot{x}(t) = F_2 x(t) + f_2 v_g(t), \qquad S_1, S_4 \text{ are on} \tag{4.1b}$$

the chosen state vector is $x(t) = [i_f(t), i_g(t),\, v_f(t),\, v_{\rm dc}(t)]^T$ and where matrices F_1, F_2 and vectors f_1, f_2, obtainable through elementary circuit theory by applying Kirchhoff's laws, are

$$F_1 = \begin{bmatrix} 0 & 0 & -\frac{1}{L_f} & -\frac{1}{L_f} \\ 0 & 0 & \frac{1}{L_g} & 0 \\ \frac{1}{C_f} & -\frac{1}{C_f} & 0 & 0 \\ \frac{1}{C_{\rm dc}} & 0 & 0 & 0 \end{bmatrix}, \; F_2 = \begin{bmatrix} 0 & 0 & -\frac{1}{L_f} & \frac{1}{L_f} \\ 0 & 0 & \frac{1}{L_g} & 0 \\ \frac{1}{C_f} & -\frac{1}{C_f} & 0 & 0 \\ -\frac{1}{C_{\rm dc}} & 0 & 0 & 0 \end{bmatrix},$$
$$f_1 = f_2 = \begin{bmatrix} 0 & -\frac{1}{L_g} & 0 & 0 \end{bmatrix}^T$$

4.2. Nonlinear control model. The principle of operation of the circuit naturally lends itself to formulating a discrete time model of the system with sampling interval T_s. In particular, through a standard zero-order-hold discretisation over T_s of (4.1a), (4.1b) one has the expressions

$$x(k+1)_{\rm m1} = A_1 x(k) + B_1 v_g(k) \tag{4.2a}$$
$$x(k+1)_{\rm m2} = A_2 x(k) + B_2 v_g(k)\,, \tag{4.2b}$$

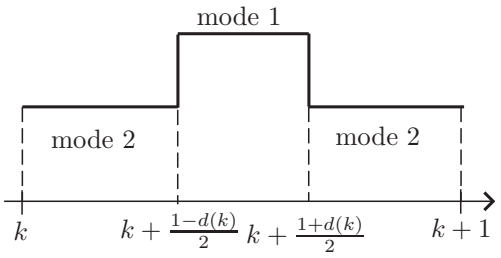


FIG. 4.2. *Employed modulation scheme.*

where k denotes the physical time instant $t = kT_s$. In particular, it holds that

$$A_1 = e^{T_s F_1}, \quad B_1 = F_1^{-1}(I_4 - e^{T_s F_1})g_1$$
$$A_2 = e^{T_s F_2}, \quad B_2 = F_2^{-1}(I_4 - e^{T_s F_2})g_2$$

in which I_4 denotes the 4×4 identity matrix. The equations featured in (4.2) are respectively valid for the system update if the duty cycle were 1 (subscript $m1$, circuit always in mode 1) or 0 (subscript $m2$, circuit always in mode 2), where the only approximation is that the grid voltage is assumed to remain constant at $v_g(k)$ throughout the k-th sampling period (this assumption is reasonable in view of the diversity in time scale of T_s and of the period corresponding to the grid frequency).

By applying the classic averaging approach [9], [18] the control model dynamics then assume the form

$$\begin{aligned} x(k+1) &= x(k+1)_{\mathrm{m1}} d(k) + x(k+1)_{\mathrm{m2}}(1-d(k)) \\ &= (A_1 - A_2)x(k)d(k) + (B_1 - B_2)v_g(k)d(k) \end{aligned} \tag{4.3}$$

with $0 \leq d(k) \leq 1$, wherein the discrete time updates valid for the two modes are correspondingly *averaged* over the period according to their effective duration and where it is manifest that the resulting expression is nonlinear.

4.3. Control scheme.

Control objectives. The control objective is to track the grid current reference $i_{g,\mathrm{ref}}$, which for the scope of the present work will be assumed to be a sinusoidal waveform of varying frequency and amplitude. This has to be achieved in the presence of the hard constraints on the manipulated variable (the duty cycle) which is bounded between 0 and 1. Moreover, the controller should yield a duty cycle devoid of higher order frequency components, i.e. subharmonic oscillations which could cause fast-scale instabilities and electromagnetic interference issues.

Constrained finite time optimal control. The control objective is to track the grid current to its reference, or equivalently, to minimize the quadratic grid current error $i_{g,\mathrm{err}}(k) = (i_g(k) - i_{g,\mathrm{ref}}(k))^2$. The reference is by nature time-varying, as is the grid voltage $v_g(k)$, so that both must be featured among the state-parameter variables, which with a slight abuse of notation shall still be denoted by $x(k)$. Although measured and updated at each sampling instant k these two values will be considered to be constant over the horizon of the optimal control problem.

Consider the objective function

$$J(z, x(k)) = \sum_{j=0}^{N-1} i_{g,\mathrm{err}}(k+j|k) \ , \tag{4.4}$$

which penalizes the predicted evolution of $i_{g,\mathrm{err}}(k+j|k)$ from time-instant k on over the finite horizon of length N, where $z = \left[X_1^N,\, D(k)\right]$, with $D(k) := [d(k), \ldots, d(k+N-1)]$ and $X_1^N := [x(k+1), \ldots, x(k+N)]$. Vector $z \in \mathbb{R}^{(m+n)N}$ is the optimization vector containing all the decision variables (present and future states and control inputs) and $x_0 = [x(k)^T, i_{g,\mathrm{ref}}(k), v_g(k)]^T$ is the initial state-parameter (on-line measurement) vector of the discrete-time system.

The control input at time-instant k is then obtained by minimizing the objective function (4.4) over the sequence of control moves $D(k) = [d(k), \ldots, d(k+N-1)]^T$ (in practice over the space of the decision vector z) subject to the related system equations and constraints for the model (4.3); the resulting nonlinear optimization program is referred to as the constrained finite time optimal control (CFTOC) problem and is solved by means of the algorithm presented in Section 3.

4.4. Control scheme based on linear model. As a term of comparison it is interesting to consider the case for which, with an additional simplifying assumption, a linear control model is derived for the system and a corresponding optimal control problem is set up. In particular, as mentioned in Section 4.1, the dc-link would ideally be assumed to supply a constant voltage v_{dc}, which is a reasonable hypothesis if C_{dc} is large enough (notice that this increases the cost of the capacitor considerably). In such a case the state vector would be $x(t) = [i_f(t),\, i_g(t),\, v_f(t)]^T$ whereas the system equations would become

$$\begin{aligned} \dot{x}(t) &= Fx(t) + fv_g(t) + gv_{\mathrm{dc}}, && S_2, S_3 \text{ are on} \\ \dot{x}(t) &= Fx(t) + fv_g(t) - gv_{\mathrm{dc}}, && S_1, S_4 \text{ are on} \end{aligned}$$

where F, f and g are

$$F = \begin{bmatrix} 0 & 0 & -\frac{1}{L_f} \\ 0 & 0 & \frac{1}{L_g} \\ \frac{1}{C_f} & -\frac{1}{C_f} & 0 \end{bmatrix}, \ f = \begin{bmatrix} 0 \\ -\frac{1}{L_g} \\ 0 \end{bmatrix}, \ g = \begin{bmatrix} -\frac{1}{L_f} \\ 0 \\ 0 \end{bmatrix}$$

By employing the same discretization procedure one would obtain for the two modes

$$\begin{aligned} x(k+1)_{\mathrm{m1}} &= Ax(k) + Bv_g(k) + Cv_{\mathrm{dc}} \\ x(k+1)_{\mathrm{m2}} &= Ax(k) + Bv_g(k) - Cv_{\mathrm{dc}} \ , \end{aligned}$$

where

$$A = e^{T_s F}, \quad B = F^{-1}(I_3 - e^{T_s F})f, \quad C = F^{-1}(I_3 - e^{T_s F})g$$

having denoted the 3×3 identity matrix by I_3. By averaging over one period the discrete-time state update would result in being

$$\begin{aligned} x(k+1) &= x(k+1)_{\mathrm{m1}} d(k) + x(k+1)_{\mathrm{m2}}(1 - d(k)) \\ &= Ax(k) + Bv_g(k) + Cv_{\mathrm{dc}}(2d(k) - 1) \end{aligned} \tag{4.5}$$

with $0 \leq d(k) \leq 1$ and would thus be a linear expression.

A control problem identical to the one presented in Section 3 could then be formulated, with the only difference being that $v_{\rm dc}$ would not be part of the state vector as it is a constant and that (4.5) would now be the prediction model. This formulation leads to a linear programming problem that can efficiently be solved using existing linear programming solver or one could even explicitly compute the solution as a function of the parameters.

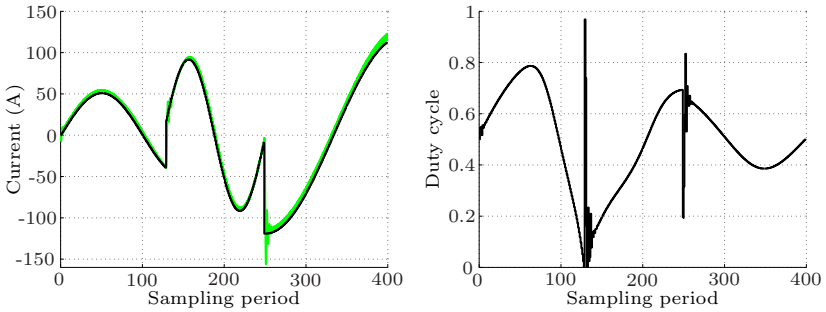
4.5. System parameters. In this section, simulation results are presented demonstrating the performance of the proposed control methodology. The circuit parameters are given by $C_{\rm dc} = 0.35\,mF$, $C_f = 4.7\mu F$, $L_f = 0.8\,mH$, and $L_g = 0.65\,mH$. The grid voltage is assumed to be an ideal sine wave of amplitude $\sqrt{2}\ 230\,V$ and frequency $50\,Hz$; the grid current reference is set to $i_{g,ref}(t) = I_{g,ref} sin(2\pi\omega t)$ and a switching frequency of $10\,kHz$ is assumed so that the switching period is $100\,\mu s$. The prediction horizon length is $N = 4$ and as a simplifying assumption it is set $d(k+1|k) = d(k+2|k)$ in the optimal control problem presented in Section 4.3, i.e. a partial move blocking scheme is employed.

4.6. Closed loop simulation results. Starting from the physical state vector $x(0) = [0, 0, 0, 800]$, that is with the filter and grid at zero initial conditions and with the dc-link capacitor charged at $800\,V$, the grid current reference amplitude is set to $I_{g,ref} = \sqrt{2}\ 36\,A$ with $\omega = 50\,Hz$. Figure 4.3(a) depicts the evolution of the grid current as it tracks the selected reference. As can be seen the controller manages to follow the desired value closely until at sampling instant $k = 130$ a step variation is imposed on the reference so that $I_{g,ref} = \sqrt{2}\ 65\,A$ and $\omega = 80\,Hz$ thereafter. The duty cycle is then brusquely modified (Figure 4.3(b)) in order to follow the newly imposed reference value as quickly as possible. Successively at instant $k = 250$ a second step in the reference occurs after which $I_{g,ref} = \sqrt{2}\ 84\,A$ and $\omega = 30\,Hz$; again, the controller can be seen to rapidly manoeuvre the duty cycle to quickly regain effective tracking of the grid current. The overall evolution of the system variables are portrayed in Figure 4.3(c): as can be seen the voltage of the dc-link varies considerably and it is therefore advantageous to feature its evolution in the control model. Additionally, the filter current closely mirrors the grid current in behavior although it also exhibits a higher frequency ripple superimposed on it.

4.7. Comparison with control scheme based on linear model. As a term of comparison the system response achievable with the controller based on the linear model described in Section 4.4 is presented for the same choice of parameters and operating scenario, as shown in Figure 4.4. In particular it is assumed that the voltage of the dc-link is fixed to its initial value $800\,V$ in (4.5). It is manifest that after an initial transient during which the performance of the two schemes is comparable (since the dc-link is in any case close to its initial value), the quality of the tracking deteriorates for the controller based on the linear model, with a noticeable error and phase lag. The response is additionally poorer for the variations in the reference, after which the grid current is more or less restored to its desired value only with a certain delay and with a relatively limited degree of accuracy.

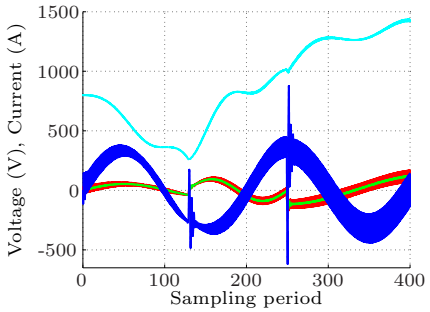
The quality of the achievable performance is therefore substantially inferior in this case; it could be reasonably ameliorated by increasing the size of the capacitor $C_{\rm dc}$ in order to make the linear prediction model (4.5) more accurate, thus incurring however in higher converter costs[‡].

[‡]Notice that a few issues have been ignored in the presented control problem for the sake of sim-



(a) Grid current i_g (green) and reference $i_{g,ref}$ (black)

(b) Control input



(c) Grid current i_g (green), filter current i_f (red), filter voltage v_f (blue) and dc-link voltage v_{dc} (cyan)

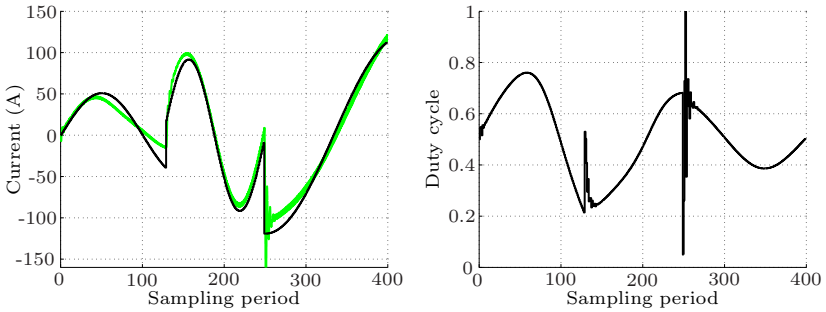
FIG. 4.3. *Simulation results for the proposed control scheme*

4.8. Computations. Using the implementation of the algorithm described in Section 3, the presented case study could be solved. Below we give some details about the characteristics of this application using the proposed algorithm. Note that timings depend highly on several tracking dependent parameter such as accuracy etc., also see the discussion in the conclusion section. It is merely meant to give an idea about the order of magnitude of the computation time rather than precise indication about the timings on a target hardware.

With a careful choice of a multihomogenous start system, the offline algorithm involved tracking of 2571 paths, leading to 22 finite, multiplicity one solutions. The total offline computation time was 475.21 seconds on a 3.2 GHz Pentium 4 machine. Depending on the actual target parameters the average online tracking time for a single path took between 150 milliseconds and 2.5 seconds. In particular, the number of real solutions is dependent upon the choice of parameter value (as is expected from theory, but more significantly than in other applications).

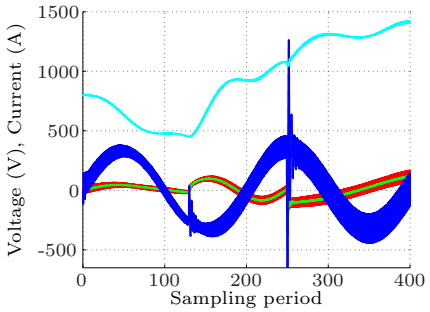
5. Power Electronics Application II.

plicity, the most significant being the delay introduced by the computation and state measurability, i.e. the calculated optimal control input will typically have to be applied at the successive sampling instant due to the computation time and not all system variables are actually accessible. Adequate provisions however can be made in terms of delay compensation and state estimation techniques: the interest herein rather lies, all other things being equal, in analyzing the benefit derived from the explicit inclusion of the nonlinear dynamics.



(a) Grid current i_g (green) and reference $i_{g,ref}$ (black)

(b) Control input



(c) Grid current i_g (green), filter current i_f (red), filter voltage v_f (blue) and dc-link voltage $v_{\rm dc}$ (cyan)

FIG. 4.4. *Simulation results for the control scheme based on linear model*

5.1. Problem description and simulation results. We will now consider the controller synthesis problem of a fixed-frequency buck dc-dc converter, see Figure 5.1. The voltage source v_s feeds the circuit through the switching stage and the low pass LC filter to supply power to the output resistive load R_o. The dually operated switches S_1, S_2 must be commutated appropriately in order that the requested output voltage reference $v_{o,\rm ref}$ (lower in value than the available dc voltage source v_s, hence the name *buck*) be applied to the output load. The switches are operated by a pulse sequence with constant switching frequency f_s (resp. period T_s) through a duty cycle $0 \leq d(k) \leq 1$ similarly defined as in Section 4.1. Additionally, the considered system along with its associated control problem description, nonlinear model derivation and optimal control formulation have been extensively described in [3], whereto the interested reader is referred for full details: in the following a summary of the obtained results will be given. In particular, a control model of the form

$$\begin{aligned} x(k+1) &= x(k+1)_{\rm m1} d(k) + x(k+1)_{\rm m2}(1-d(k)) \\ &= \Phi(R_o)x(k) + \Psi(R_o)d(k) \end{aligned} \tag{5.1a}$$

with $0 \leq d(k) \leq 1$ can be derived through a classical averaging approach [9], [18], where it has been set $x(k) = [\frac{i_L(k)}{v_s} \ \ \frac{v_o(k)}{v_s}]^T$ and where it can be easily verified that matrix $\Phi(R_o)$ and vector $\Psi(R_o)$ depend on a nonlinear (rational) function of the load

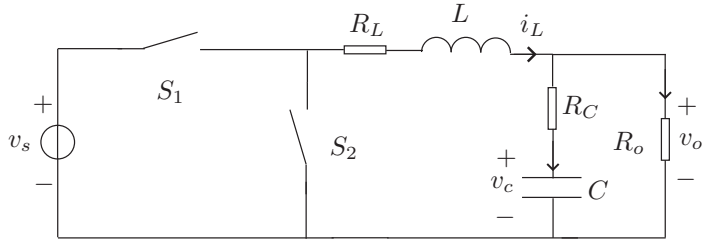


FIG. 5.1. *Topology of the buck converter. R_o denotes the output load resistance, R_C and R_L are the parasitic resistances of the capacitor C and inductor L.*

R_o.[§] The scaling of the state over the voltage source has been performed in order to ensure that the controller to be derived is independent of (and therefore *valid for any value of*) v_s.

The main control objective for the buck dc-dc converter is to regulate v_o to its reference $v_{o,\text{ref}}$ with an accuracy of $\pm 1\%$, while satisfying the constraints $0 \leq d(k) \leq 1$ and $i_L(k) \leq i_{L,\max}$, even under (measurable) variations in v_s and the load R_o[¶].

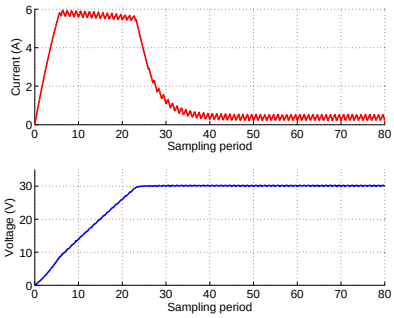
By augmenting model (5.1) with the appropriate constraint for $i_L(k)$ and formulating a cost function reflecting the desired control objectives over the chosen horizon it is then possible to derive an optimal control problem which can be recast in the form (3.1). Notice that in this case the vector x_0 featured therein includes R_o as a varying parameter [3], whereas as mentioned v_s need not be accounted for in this manner as all quantities and equations have been scaled over it.

For the closed loop simulations the output voltage reference is set to $v_{o,\text{ref}} = 30\,V$ and $i_{L,\max} = 6\,A$. The first case to be analyzed in Figure 5.2(a) and Figure 5.2(b) is that of the transient behavior during startup, i.e. when $x(0) = [0, 0]^T$. The proposed optimal control scheme yields an output voltage that reaches its desired value within 25 switching periods. The hard constraint on the inductor current is respected, in that it is always kept below $6\,A$. For the second case, results stemming from a 50% increase in the voltage source v_s during the previously attained steady state operation and from a 90% decrease in the output load R_o are shown in Figures 5.3(a)-5.3(b); more specifically, the voltage source increase occurs at the 90^{th} switching period and on top of that, at the 145^{th} switching period, the load decrease kicks in. In the simulation diagrams we can see that the controller can cope with both disturbances, guaranteeing a constant regulated output voltage of $30\,V$.

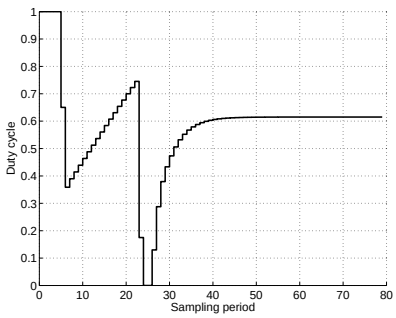
5.2. Computations. This case study was also solved using the implementation of the algorithm described in section 3. For the off-line part of the application discussed in this paper, there were 1392 paths to be tracked, leading to 19 finite, multiplicity one solutions. The tracking for the off-line part of the algorithm took 88 seconds on a 3.2 GHz Pentium 4 machine. The on-line part of the algorithm involved tracking the 19 finite solutions from the off-line part through a coefficient parameter homotopy. The tracking of a single path took approximately 10ms on the same ma-

[§]To deal with the rational expressions appearing (5.1) and make them conform to the framework of Section 3, we clear denominators from their right-hand side. This does not pose any limitations, since these are physical quantities that do not assume the value of zero. We then obtain polynomial equations, representing the state-update rational relation (5.1a), which are in turn used as equalities in the problem formulation (3.1).

[¶]The assumption that the load R_o is effectively measurable is not trivial in effective industrial practice and would for example require the implementation of an extended Kalman filter.

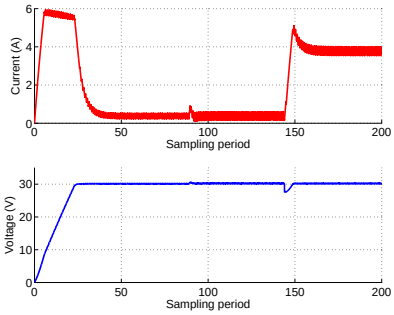


(a) Inductor current (above) and output voltage (below)

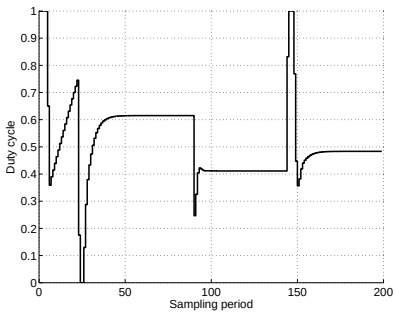


(b) Duty cycle during the startup phase

FIG. 5.2. *Simulation results for the startup scenario*



(a) Inductor current (above) and output voltage (below)



(b) Duty cycle

FIG. 5.3. *Simulation results for the scenario featuring a 50% increase of v_s and a 90% decrease in the output load R_o*

chine, but depends highly on the chosen accuracy. All 19 finite paths stayed finite, although in every case, two paths landed at complex solutions while the other 17 were real. Removing solutions not satisfying the inequality constraints in the KKT condition (3.3) and a finite search over all remaining points yields the global optimum.

6. Conclusion. In this paper we have proposed a new pre-computation-based method for polynomial optimal control. In contrast to many other optimization routines used in non-linear optimal control, based e.g. on SQP solvers ([25], [17]), our approach has the guarantee of finding the global optimum. By pre-solving a generic instance of the underlying parametric programming problem using homotopy continuation with a multihomogeneous start system, we shift a large portion of the computational burden offline, allowing an easier task for the online computation. While the offline part might take hours or even days to be completed, the online implementation usually requires only fractions of a second of computation time.

Among the applications that would benefit most from this new approach are power electronics systems, with sampling times of fractions of milliseconds. Motivated by this fact, the technique has been illustrated on two such power electronics case studies.

All timings given in this papers are highly dependent upon the settings of various tracking-related tolerances (which are too numerous to detail here) and also include

the file read-write overhead, but they already give an idea about the order of magnitude. However, with the current implementation and hardware setup there is still considerable work to be done in order to bring these timings down to an acceptable level for practical applications.

The curious reader may wonder just how far these methods can be pushed. In particular, what is the largest polynomial system that can be solved by Bertini or similar software? While a few problems involving around 1000 variables have been solved with Bertini, it is unrealistic to hope for more than 100 equations or so at this point. There are many factors that impact the size of the problem that can be solved – too many, in fact, to carefully enumerate here – but the main concerns are the number of paths to be tracked off-line, the number of paths to be tracked for each step on-line, the size of the polynomial system, (since the Jacobian matrix figures prominently in the method), the degrees of the polynomials, (high degree causes numerical instability, which is countered by high precision at the cost of a significant reduction in efficiency), and other factors. Since the number of states, the horizon length, and the number of constraints of an MPC problem all effect the number of variables in the associated KKT conditions, it is difficult to estimate exactly what number of states, horizon lengths, or number of constraints may be handled with this method.

While this paper focuses primarily on the proof of concept, new algorithms and more advanced implementations might soon lead to solutions that allow the computation of optimal control inputs at the desired sampling time on the target hardware. The path tracking step can be easily parallelized and it would be interesting to investigate how much one can improve the timings with, e.g., a parallel hardware implementation using Field Programmable Gate Array (FPGA) technology or GPU's.

On the software side there are also many unexploited mechanisms to significantly speed up the online tracking. The most promising approach is to replace the fixed start system by a start system that is continuously updated and moved along with the parameter to be tracked online.

Acknowledgments. Partially supported by the Institute for Mathematics and Its Applications (IMA). We express our gratitude also to Professor Andrew Sommese for useful conversations and the use of various computational resources and to Professor Pablo Parrilo for interesting discussions.

REFERENCES

[1] E.L. Allgower and K. Georg. Continuation and Path Following. In *Acta Numerica*, pages 1–64. Cambridge University Press, Cambridge, UK, 1993.

[2] E.L. Allgower and K. Georg. *Introduction to Numerical Continuation Methods*, volume 45 of *Classics in Applied Mathematics*. SIAM, 2003.

[3] D.J. Bates, A.G. Beccuti, I.A. Fotiou, and M. Morari. An Optimal Control Application in Power Electronics Using Numerical Algebraic Geometry. In *Proceedings of the American Control Conference*, Seattle, Washington, June 2008.

[4] D.J. Bates, I.A. Fotiou, and P. Rostalski. A Numerical Algebraic Geometry Approach to Nonlinear Contrained Optimal Control. In *IEEE Conference on Decision and Control*, pages 6256–6261, New Orleans, LA, December 2007.

[5] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Adaptive multiprecision path tracking. *SIAM J. Numer. Anal.*, 46(2):722–746, 2008.

[6] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The Explicit Linear Quadratic Regulator for Constrained Systems. *Automatica*, 38:3–20, 2002.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[8] D.F. Davidenko. On a New Method of Numerical Solution of Systems of Nonlinear Equations. *Doklady Akademii Nauk SSSR*, 88:601–602, 1953.

[9] R.W. Erickson, S. Cuk, and R.D. Middlebrook. Large Signal Modeling and Analysis of Switching Regulators. In *IEEE Power Electronics Specialists Conference Records*, pages 240–250, Cambridge, MA, June 1982.

[10] I.A. Fotiou, A.G. Beccuti, G. Papafotiou, and M. Morari. Optimal Control of Piecewise Polynomial Hybrid Systems Using Cylindrical Algebraic Decomposition. In *Hybrid Systems: Computation and Control*, volume 3927 of *Lecture Notes in Computer Science*, pages 227–241, Santa Barbara, CA, 2006.

[11] I.A. Fotiou, P. Rostalski, P.A. Parrilo, and M. Morari. Parametric Optimization and Optimal Control Using Algebraic Geometry. *International Journal of Control*, 79:1340–1358, 2006.

[12] C.E. Garcia, D.M. Prett, and M. Morari. Model Predictive Control: Theory and Practice – A Survey. *Automatica*, 25:335–348, 1989.

[13] D.W. Jordan and P. Smith. *Nonlinear Ordinary Differential Equations*. Oxford Applied Mathematics and Computer Science. Oxford University Press, Oxford, UK, 1987.

[14] H.W. Kuhn and A.W. Tucker. Nonlinear Programming. In *2nd Berkeley Symposium*, pages 481–492, Berkeley, CA, 1951.

[15] O.L. Mangasarian and S. Fromovitz. The Fritz John necessary optimality conditions in the presence of equality and inequality constraints. *J. Math. Anal. Appl.*, 17:37–47, July 1967.

[16] D. Manocha. Solving Systems of Polynomial Equations. *IEEE Computer Graphics and Applications*, 14:46–55, 1994.

[17] F. Martinsen, L. Biegler, and B.A. Foss. A new optimization algorithm with application to MPC. *Journal of Process Control*, 14(8):853–865, 2004.

[18] R.D. Middlebrook and S. Cuk. A General Unified Approach to Modeling Switching Power Converter Stages. In *IEEE Power Electronics Specialists Conference*, pages 18–34, Cleveland, OH, June 1976.

[19] M. Morari and J.H. Lee. Model Predictive Control: Past, Present and Future. *Computers and Chemical Engineering*, 23:667–682, 1999.

[20] A. Morgan and A. Sommese. A Homotopy for Solving General Polynomial Systems that Respects m-homogeneous Structures. *Appl. Math. Comput.*, 24:101–113, 1987.

[21] T. Ohtsuka. A Continuation/GMRES Method for Fast Computation of Nonlinear Receding Horizon Control. *Automatica*, 40:563–574, 2004.

[22] A.B. Poore. Bifurcation Problems for Some Parametric Nonlinear Programs in Banach Spaces. *SIAM Journal on Control and Optimization*, 34:1947–1971, 1996.

[23] S.J. Qin and T.A. Badgwell. A Survey of Industrial Model Predictive Control Technology. *Control Engineering Practice*, 11:733–764, 2003.

[24] A.J. Sommese and C.W. Wampler. *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*. World Scientific Publishing, Hackensack, NJ, 2005.

[25] M.C. Steinbach. A structured interior point SQP method for nonlinear optimal control problems. In R. Bulirsch and D. Kraft, editors, *Computational Optimal Control*, volume 115 of *International Series of Numerical Mathematics*, pages 213–222. Birkhäuser, 1994.

[26] H.J. Stetter. *Numerical Polynomial Algebra*. SIAM, Philadelphia, PA, 2004.

[27] B. Sturmfels. *Solving Systems of Polynomial Equations*. Number 97 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 2002.

[28] J. Verschelde. *Homotopy Continuation Methods for Solving Polynomial Systems*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1996.

[29] C.W. Wampler. Bezout Number Calculations for Multi-Homogeneous Polynomial Systems. *Applied Mathematics and Computation*, 51:143–157, 1992.

[30] L.T. Watson. Theory of Globally Convergent Probability-One Homotopies for Nonlinear Programming. *SIAM Journal on Optimization*, 11:761–780, Nov. /Feb. 2000.

[31] V. Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14:1–29, 1992.