REPORT DOCUMENTATION PAGE

AFOSR-SR-BL-TR-98-
0283

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	3/23/98	Final Report, 3/95 to 3/98

4. TITLE AND SUBTITLE
Nonlinear Control Theory for Aerospace Vehicles

5. FUNDING NUMBERS
Contract Number
F49620-95-C-0015

6. AUTHOR(S)
Blaise Morton and Michael Elgersma

7 PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research /NM
Bolling Air Force Base, DC 20332

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTAL NOTES

12A. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This report contains two related manuscripts on the subject of mechanism geometry. The first paper addresses general theory, the second addresses practical computations. Together, they suggest a general approach for efficient parametrization of spatial mechanism configuration spaces.

14. SUBJECT TERMS
Mechanisms, Configuration Space, Algebraic Equations, Generalized Eigenvalue Problems

15. NUMBER OF PAGES
76

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

1 Introduction

This report contains two related manuscripts on the subject of mechanism geometry. The first paper addresses general theory, the second addresses practical computations. Together they suggest a general approach for efficient parametrization of spatial mechanism configuration spaces. Before describing the contents of these papers we discuss some of the steps that led us to this point.

About ten years ago we started developing a general control architecture for launch vehicles as part of the Adaptive Guidance, Navigation and Control contract. The objective was to reduce the time and cost of producing flight software for launch vehicles.

One of the difficult features of launch vehicles, from a control perspective, is their limited margins in both performance and robustness. Because of limited performance margin, the ascent trajectory must be carefully tailored for each flight, depending on payload and mission. Because of limited robustness margin, the control laws must be reanalyzed and perhaps changed depending on payload and mission. In order to reduce the time and effort required for the flight software development, we decided to automate the guidance and control design tasks in a software-development environment that included a multibody simulation of the launch vehicle. The multibody model reflected propellant slosh, engine gimbaling and vehicle flex dynamics. For this reason we wanted to find an efficient simulation of our multibody system.

The multibody model we adopted for the launch vehicle had one feature that made the kinematic problem especially easy – there were no closed loops. In such a case the configuration space is simply the product of the individual configuration spaces of the joints linking the bodies together. We were using revolute and spherical joints only, so the configuration space turned out to be a product of (closed subsets of) circles and orthogonal groups. Easy to parametrize. The key technical insight for this problem was the realization that the dynamic computations could be reduced from order N^3 to order N, where N is the number of degrees of freedom. Having figured out how to do that for our launch vehicle simulation, we survived our first serious encounter with multibody systems.

The next multibody problem we encountered was a true mechanism problem – a six-degree-of-freedom hand controller. The hand controller mechanism was a general Stuart platform with three legs connecting the movable

top plate to the base. The details of the configuration are not important for our discussion here, what is critical is that the motions of the various bodies making up the mechanism were constrained by loop-closure conditions. In this situation is it is no longer true that the configuration space is the product of the individual configuration spaces of the joints connecting the bodies. In fact, for general mechanisms of this type, the configuration spaces may be very complicated and need not be easy to parametrize. This state of affairs can make it difficult to design a mechanism for an engineering application, because it is not easy to write down and analyze dynamic equations without first parametrizing the configuration space (methods do exist, but they are not so easy to use for system design). In any case, after some effort we were able to parametrize our hand-controller mechanism and so we survived our second encounter with multibody systems.

But our curiosity was piqued. The analysis of the hand controller took longer than we thought it should, and the approach that finally worked was unsystematic. Surely there was a better approach, one that we could learn before our third encounter. On looking into the matter, however, we found that no general, systematic approach was known. In fact, there were single-loop one-DOF mechanisms (the 7R) for which the most authoritative reference we could find (Duffy's book) had no solution. In the 1970's, the 7R problem was called the Mount Everest of mechanism problems by one of the experts in the field. As luck would have it, the Chinese researchers Lee and Liang had just solved the 7R problem in 1986 (it was now the early 1990's). Ignorant of their results, we set out to develop a more systematic approach to general mechanism problems, and we decided to use the 7R as our acid test for the general method.

After a little work we found an approach that was at the same time systematic and computationally tractable. We worked out the details for the special case of the 7R mechanism and submitted our results for publication in June of 1993. It was then we learned of the Chinese solution of the 7R problem, so we could not claim primacy for the 7R solution. But our approach to the problem was original enough to warrant publication, so we revised the manuscript and resubmitted it. It appeared as " A New Computational Algorithm for 7R Spatial Mechanisms " in Mech. Mach. Theory Vol. 31, No. 1, pp. 23-43, 1996.

In the five years since we first applied our method to the 7R problem we have continued developing both the theory and the computational methods

associated with the general approach. The fundamental theory is described in full generality in the first paper included in this report. This first paper is complete as it stands – ready for publication. The second paper is not ready for publication, it is the current draft of a working document on the computational technique. All of the methods discussed in the second paper have been tested on numerical examples with experimental software, so we are confident of useful results even though we do not yet know the full range of applicability. We expect to continue research into this area in the years to come.

1.1 Discussion of Key Results

The first paper is titled " Multi-Affine Modular Systems for Mechanism Families." A summary of contents can be found in the abstract and introduction of that paper. Here we outline the key steps.

The first step is to define precisely what a mechanism of (R,P,S)-type is. A minimal set of parameters needed to describe a mechanism of this type is then defined. We then describe what is meant by the configuration space of a mechanism, and develop ideas for computing the dimension of the configuration space from the parameters describing the mechanism.

Some illustrative examples are then developed to show the difficulty with the usual formula for the dimension of the configuration space from crude features of the mechanism data alone. This difficulty motivates consideration of families of mechanisms, and leads to the distinction between complete and overconstrained families. A general theory of complete families is then developed, and it is shown that the usual formula for the dimension of the configuration space applies to a generic mechanism in such a family. It is worth pointing out that nongeneric mechanisms in complete families and even overconstrained mechanisms are often (approximately) encountered in real-world systems (e.g. door-locks, axles), so the mathematical ideas developed here are of practical significance, even though they are not usually considered.

Next, for a generic mechanism in a complete family we describe the structure of a branched cover for the configuration space over a base space composed of a product of joint configuration spaces (this structure often arises, but not always – consider three bodies connected by spherical joints for a counterexample). The physical significance of the branch locus and the covering degree of this branched cover (when it exists) are discussed. It is an

interesting, open problem to find a formula for the covering degree directly from the mechanism parameters and the joint-data of the base space. In this situation, to compute coordinates on the configuration space, we can compute local sections of the branched cover over the base space.

Whether or not the branched cover over the product of joints exists, we can find a covering set of local parametrizations. Using a special set of joint coordinates, we show how a modular system of multiaffine equations describing the configuration space can be generated. The generation of the multiaffine equations is the central theme of the third section of this paper. Methods for solving equations of this type are the topic of the second paper in this report.

The second paper is titled "A Polynomial-Runtime Algorithm for Multi-Affine Equations." The central theme of this second paper is a polynomial-runtime algorithm for solving the system of equations developed in the first paper. In this case, the polynomial time condition means polynomial in the number of solutions to the system of equations. In general, the number of solutions of the system of equations grows combinatorially with the number of equations, so the runtime of the algorithm grows faster than any polynomial function of the number of equations. Given the growth in the number of solutions, however, there is no avoiding this problem (just writing down the solutions to the equations has superpolynomial growth). So, an algorithm polynomial in the number of solutions is the best one can hope for. It is worth noting that Groebner basis methods are worse than polynomial in the number of solutions, so the algorithm described here is an improvement over Groebner basis.

The text is divided into two chapters. The first chapter concerns solution methods when the number of equations is the same as the number of unknowns. In this case the number of solutions for a generic system is given by a known formula, and the solution method uses that formula.

The two basic techniques are:

1. Dilation – generating many equations of higher degree from the original multiaffine system

2. Deflation – removing extraneous roots from the higher-order system

When these two methods are applied to systems of multiaffine equations, the result is a generalized eigenproblem of size equal to the number of solu-

tions. The eigenproblem can be solved in polynomial time, hence the polynomial runtime of the overall algorithm.

The second chapter concerns solution methods for multiaffine systems when the number of equations exceeds the number of unknowns. Such systems are called overdetermined. Systems of this type are generated by the method described in the first paper, hence our interest in them. When coefficients are generic, overdetermined systems have no solutions, so the coefficients that arise in the mechanism problem must satisfy some special set of relations that allows solutions to exist.

Overdetermined systems in n variables tend to have fewer solutions than systems with n equations in n variables, so in theory the solution algorithm could be quicker. In practice, the additional structure usually makes the solution more difficult – especially in those cases where the number of solutions is not know a priori.

For the overdetermined systems arising in mechanism analysis we have developed a third technique described in section 2.3. It is an iterative method for reducing the size of a nonsquare generalized eigenproblem to a smaller square one. A tentative name for this procedure is ablation, but a better name might be found when the theory is better developed.

The theory in the case of overdetermined systems clearly needs more work. By implementing these algorithms in experimental code and applying them to examples we know the methods work at least for some problems, but we have yet to determine the complete range of applicability.

MULTI-AFFINE MODULAR SYSTEMS FOR MECHANISM FAMILIES

Blaise Morton and Michael Elgersma

Honeywell Technology Center

MN65-2500

3660 Technology Drive

Minneapolis, MN 55418

Abstract

This paper describes part of our six-step, general approach for mechanism kinematic analysis. The details for a particular application for the 7R spatial mechanism are presented in [ME]. Here we concentrate on the derivation of the modular system for a general class of spatial mechanisms.

The quaternion-pair approach described below can be applied to spatial mechanisms endowed with spherical, prismatic and cylindrical joints as well as the revolute joints illustrated in [ME]. Systems of multi-affine equations are obtained in the same way for single and multi-loop mechanisms. In these equations, the number of independent variables is equal to the sum over all joints of the joint degrees of freedom, so such a system is minimal in that sense.

1. INTRODUCTION

The algebraic analysis of mechanism kinematics is an old subject that remains an active area of research today. Some of the current theoretical interest has been motivated by the growing power in today's computers, to analyze examples that were simply too complicated to examine by hand. A case in point is the general 1-DOF, spatial 7R mechanism that was recently solved by Lee and Liang [Lee1,2] in the late 1980s. The subject of this paper is a new approach to kinematic analysis of mechanisms that addresses a special but important class of examples, including the 7R mechanism. Here we describe our general approach from a mathematical point of view. The main result is a computational technique for generating modular systems of equations [Macaulay] of a very special type.

Before embarking on the general theory we present a simple example, to demonstrate the nature of the the problem being solved.

Consider a 4-bar, planar linkage. One of the bars is assumed fixed (ground), while the other three are joined in pairs by revolute joints to form a closed circuit. For general values of the bar lengths, this simple mechanism has one degree of freedom. Any one of the joint angles can be used as a local coordinate for the one-dimensional configuration space, pick one of the angles on the base and call it θ_3. The problem is to find the values of θ_0, θ_1, and θ_2 as a function of θ_3.

There is a standard technique for converting this problem into a system of algebraic equations. For each fixed value of θ_k there is a simple geometric construction to find the allowed configurations. The problem reduces to finding the points of intersection of two circles in the plane, and from the geometry it is clear that there are only two configurations possible for each value of θ_k. The obvious way to compute the coordinates of these two points (and

thereby determine the other joint angles) is to write down the pair of quadratic equations in the coordinates (x,y) of the intersection point and find all solutions. An interesting feature arises from the algebraic viewpoint -- by Bezout's theorem we might expect to find four solutions rather than two. This apparent inconsistency is resolved by a well-known explanation. Two of the four algebraic solutions are finite, possibly corresponding to real geometric solutions (if real solutions exist), while the remaining two solutions are always meaningless for the kinematic problem -- they are the circular points at infinity. This problem of extraneous roots is easily handled in the simple examples, but the analogous problem for more complex mechanisms has not been solved in general. The work by Duffy [Duffy1] is the most comprehensive work in which this type of problem has been addressed, but all the spatial mechanisms analyzed there are 1-DOF, single-loop with prismatic and revolute joints. The difficulties presented by the 7R problem, for example, can be traced in the evolutionary path [Freudenstein], [Roth], [Duffy2], [Lee 1,2] that led to the correct solution in this case.

The solution presented in [Lee 1,2] for the 7R problem was not very systematic, leaving open the question of how to proceed for more complicated (e.g. multiloop) spatial algorithm. To address the problem of general multiloop spatial mechanisms, we have developed the alternative approach presented in this paper. This method was first applied to the 7R mechanism in [ME]. The general mathematical theory is presented here.

To give a flavor of our new approach, let us return to the four-bar example. By the methods described in Section 3, we derive systems of multi-affine equations in the four variables z_1, z_2, z_3, z_4 where

$$z_j = \tan(\frac{\theta_j}{2}) \tag{1.1}$$

In the special case of planar 4-bars, we obtain three equations for each of the four angle

parameters -- obviously these are not independent. It turns out that only two of these three (the translation equations) represent real constraints for the parameters z_j. To show the symmetry of the construction process we use a variable index notation z_{k+3}, z_{k+2}, z_{k+1}, z_k and consider the subscripts in the integers mod 4 for each integer value of k.

For each fixed value of z_{k+3}, we can solve for z_{k+2}, z_{k+1}, and z_k. Let the leg lengths be l_1, l_2, l_3, and l_4. Two components of the translation equations (starting at joint k) are:

$$\left[A(z_{k+3}) + z_{k+2}\, B(z_{k+3})\right] \begin{bmatrix} 1 \\ z_{k+1} \end{bmatrix} = 0 \in R^2 \tag{1.2}$$

which is a generalized eigenvalue problem with:

$$A(z_{k+3}) = \begin{bmatrix} l_{k+4}+l_{k+3}+l_{k+2}+l_{k+1} & z_{k+3}(-l_{k+4}+l_{k+3}+l_{k+2}-l_{k+1}) \\ z_{k+3}(-l_{k+4}+l_{k+3}+l_{k+2}+l_{k+1}) & -l_{k+4}-l_{k+3}-l_{k+2}+l_{k+1} \end{bmatrix} \tag{1.3}$$

$$B(z_{k+3}) = \begin{bmatrix} z_{k+3}(-l_{k+4}+l_{k+3}-l_{k+2}-l_{k+1}) & -l_{k+4}-l_{k+3}+l_{k+2}-l_{k+1} \\ -l_{k+4}-l_{k+3}+l_{k+2}+l_{k+1} & z_{k+3}(l_{k+4}-l_{k+3}+l_{k+2}-l_{k+1}) \end{bmatrix} \tag{1.4}$$

To obtain the remaining variable, z_k, we can use an additional affine equation obtained by starting the translation at another joint. By starting at each of the 4 joints, the translation equations give a set of two scalar multi-affine equations for each choice of k=1,2,3,4 for a total of 8 equations. Since the k^{th} equations do not involve z_k, we can obtain 8 more multi-affine equations by multiplying by z_k, for a total of 16 multiaffine equations. This set of multi-affine equations is an example of our main result.

In any application there is more work to be done once the system of equations has been constructed. Let us consider the last step for the planar four-bar. From the 16 multi-affine

equations there are many ways to select a set of four independent ones to obtain a 4×4 generalized eigenvalue problem involving all the z_j.

$$\left[L(z_k) + z_{k+3}\, N(z_k)\right] \begin{bmatrix} 1 \\ z_{k+1} \\ z_{k+2} \\ z_{k+2}z_{k+1} \end{bmatrix} = 0 \in R^4 \tag{1.5}$$

The determinant of the 4×4 matrix on the left-hand side of equation (1.5) must vanish, giving us a single degree-4 polynomial relation between z_k and z_{k+3}. There are several ways to proceed from here.

First, by proper choice of multi-affine equations in constructing the matrix, one can find special structure in the matrices L and N that leads to degeneracies in the determinantal polynomial. The determinant becomes the square of a degree-two polynomial so only two distinct solutions exist.

Alternately, one can take the point of view that there are many such determinantal polynomials in the same two variables. The polynomial we want is the greatest common divisor of the complete set.

In any case, when all the multiaffine equations are considered together, it is found that only two solutions exist.

The reader is invited to try his hand in exploring this simple case computationally. The general situation for more complicated mechanisms remains an interesting research problem.

2. PROBLEM STATEMENT AND NOTATION

We will be considering spatial mechanisms consisting of rigid bodies connected by joints of the following type:

1) revolute (type R)
2) prismatic (type P)
3) spherical (type S)

We assume the reader is familiar with the above terminology -- the mathematical characterization of these joints is discussed in the next paragraph. The standard enginnering perspective is presented in [Duffy]. It should be observed that some other joint types (e.g. cylindric), sometimes considered separately by engineers, can be constructed from the basic joint types above.

The common feature shared by the above three joint types can be stated succinctly: the relative motions allowed by these joints form real-algebraic subgroups of the group E(3) of Euclidean motions in R^3. The relative motions allowed by an R-joint are those that fix a point in space and a line containing that point. The relative motions allowed by a P-joint form a 1-parameter translation group. The relative motions allowed by an S-joint are those that fix a point in space.

2.1 Mechanism Definition

To specify a mechanism completely (for our purposes), we need the following data:

1) A set W of rigid bodies
2) A subset J of $W \times W$; points in J are pairs of bodies that are connected

3) A joint (R, P, or S) for each point in J

4) Data specifying the relative constraint imposed by each joint on its body-pair

The number of bodies (cardinality of W) is denoted N.

We explain "mechanism specification" by a practical analogy. Imagine a box, full of parts, on which is written "some assembly required." Included in the box is an instruction booklet that tells how to identify each part uniquely (identify the points in W). The booklet also tells which part should be attached to another part (identify the points in J), and the means of attachment is specified as well (joint type for each point in J). On each body, the manufacturer has thoughtfully provided fixtures appropriate for the specified joints. The data specifying the relative constraint imposed by each joint is exactly the information required by the machine operator who endowed each body with its joint fixtures. The reader can imagine assembling the mechanism, one joint at a time (perhaps according to some recommended order), until all the bodies are connected by the appropriate joints.

Anyone who has bought a box full of parts on which is written "some assembly required" will no doubt be wondering if the parts in our hypothetical box really do fit together. In answer to that question we, like most manufacturers, merely give assurance that the parts were designed to fit together. We leave some construction details to the customer who should find that, for some configurations of pieces during assembly, the parts really do fit together as advertised (provided the dimensions of the pieces are within their design tolerances).

2.2 The Configuration Space of a Mechanism

Once the mechanism is assembled, the individual bodies within it may be moved continuously through a range of distinct positions and orientations relative to one another. When we consider motions, one specified body is assumed fixed in space (attached to ground) to eliminate

motions of the entire mechanism as a rigid body. The set C of all possible distinct, relative positions of the bodies in the assembled mechanism is called the configuration space. Let us clarify what we mean by the set of possible distinct, relative positions of the bodies.

We return to our practical analogy. Unlike most manufacturers, we explicitly state that the pieces might not fit together if their dimensions are outside design tolerances. This can happen, for example, in a 4-bar planar linkage where the length of one leg is greater than the sum of the lengths of the other three. A more subtle concern is that different (i.e. nonisotopic) embeddings of the mechanism in R^3 could result if two identical kits are assembled by procedures involving different intermediate configurations. Moving the mechanism too far in one direction will cause the bodies to collide. Problems of this sort can be very complicated if considered in the context of global embeddings -- the topological questions alone lead to problems in knot and braid theory. We look instead at the simpler problem in which intersections are allowed to occur and the various bodies are allowed to pass through each other. That is, we allow mechanism configurations into the set C even if they are immersed but not necessarily embedded. In addition, we allow some complex solutions. Allowing complex solutions (at least for some values of the part dimensions) is an artifice mandated by our computational techniques. The set of real, embedded (i.e. physically realizable) configurations is a subset of the configuration space C we have defined here.

2.3 Families of Mechanisms

We now define the concept of a family of mechanisms.

Definition: By a family of mechanisms, we mean a collection of mechanisms having the same number of bodies, isomorphic sets J, and the same joint type for the corresponding points in J.

In other words, the only differences between two mechanisms in the same family are:

1) sizes and shapes of the N bodies (independent of joint data)

2) data involving fixed-point location and line orientation at each R-joint

3) direction of translation for each P-joint

4) fixed-point location for each S-joint.

Because we are only concerned with immersed mechanisms, we really do not need to consider the exact sizes and shapes of the bodies (parameters of the first type). Each body might as well be all of R^3. From our point of view, all that matters is the finite-dimensional space of parameters associated with the positions and orientations of the R-, P- and S-joints. Each body serves to fix the relative geometry of the joints attached to it. Using this criterion as an equivalence relation, we turn attention to equivalence classes of families. We observe that the equivalence class of a complete family is a finite-dimensional space. From now on we use the term "family" to denote the term "equivalence class of families," the latter being a more precise description of our object of attention.

There is an important special class of families that merit special attention.

Definition: A family is called complete if all the parameters of the above four types are allowed to vary independently.

We have been concentrating on the theory of spatial mechanisms, but all of our statements so far apply equally well to planar mechanisms. The adjustment required is to leave out spherical joints and consider only bodies in the plane z=0 connected by R- and P- joints that preserve that plane. Now that we are considering families of mechanisms, however, we must recognize the distinction between the planar and spatial cases.

Example 1a: Consider a four-bar linkage in the plane. There are four real parameters: the distances between the revolute joints on each of the four bodies. In this case, the complete

family is the four-dimensional family of all four-bar linkages in the plane.

Example 1b: Consider a three-bar spatial mechanism with spherical joints connecting each bar with the other two. There are three real parameters: the distances between the S-joints on each of the three bodies. In this case, the complete family is the three-dimensional family of all spatial mechanisms of this type.

Note that the four-dimensional family in example 1a is not a complete family when the 4-bar is viewed as a spatial mechanism. In fact, unless the geometric parameters of the mechanism are perturbed in special ways, there are no solutions to the system of algebraic equations describing the perturbed mechanism. The angles and positions of the pairs of fixed axes associated with the R-joints on the four bodies must satisfy compatibility conditions if the bodies are to fit together in any way at all. We will return to this discussion later.

There are practical reasons for considering families of mechanisms rather than individual mechanisms, though we do not elaborate this point here. From a theoretical point of view we shall see that the "family perspective" helps to clarify some of the peculiar mathematical features of special mechanism types.

Before proceeding, we require some more definitions. We consider the planar and spatial cases in parallel.

For a joint j, we define the number of degrees of freedom D_j of that joint. In both the planar and spatial cases, both the R-joint and the P-joint have a single degree of freedom. In the spatial case only, an S-joint has three degrees of freedom.

In the following we let N_j denote the number of constrains imposed by joint j.

In the planar case $N_j = 3 - D_j$. Each body in the plane starts with three degrees of freedom, but this number is reduced by N_j independent algebraic equations relating the positions and orientations of the two bodies connected by j.

In the spatial case $N_j = 6 - D_j$. Each body in space starts with six degrees of freedom, but this number is reduced by N_j independent algebraic equations relating the positions and orientations of the two bodies connected by j.

A little thought leads to the following formula for DOF, where DOF is defined to be the dimension of the configuration space of a mechanism:

$$\text{Planar Mechanism:} \quad \text{DOF} = 3\,(N-1) - \sum_{j \in J} N_j \tag{2.1a}$$

$$\text{Spatial Mechanism:} \quad \text{DOF} = 6\,(N-1) - \sum_{j \in J} N_j \tag{2.1b}$$

The number (N-1) appears because one of the bodies is fixed to ground.

A little more thought reveals that these formulas do not always work. Considering example 1a as a spatial mechanism provides one counterexample but it is of a very special type. We provide another, more general counterexample to clarify the problem.

Example 2: Consider a spatial mechanism consisting of two bodies, each with two ball joints. The two bodies are constructed so that that their ball joints are separated by the same distance, so the mechanism can be assembled in R^3. Note that DOF = 1, because the ungrounded body is free to spin about the axis through the two ball joints. The right-hand side of

equation (2.1b) is 0.

In the following we restrict our discussion again to spatial mechanisms. The reader should be able to make the proper adjustments for the restricted planar case.

The formula of equation (2.1b) does not work for example 2 because the mechanism type is overconstrained. By overconstrained, we mean that a special relation must exist among the parameters of the complete family in order for the mechanism to be realizable. If the distances between ball joints on the two bodies are different, no solutions (real or complex) exist to the system of algebraic equations describing the mechanism.

To address this situation, we consider the configuration space C of the mechanism as a subset of the product space $E(3)^{(N-1)}$ cut out by the intersection of the constraint equations [recall that one of the bodies is grounded, hence the exponent (N-1)].

Definition: A mechanism is called simply-constrained if its configuration space C is the transverse intersection of the $\sum_{j \varepsilon J} N_j$ algebraic constraint equations imposed by its joints.

For a spatial mechanism, the transversality condition means that the rank of the subspace of the cotangent space of $E(3)^{(N-1)}$ spanned by the differentials of the constraint equations at each point of C is equal to $\sum_{j \varepsilon J} N_j$. From this definition the following consequences are immediate for a simply-constrained mechanism:

1) The configuration space C is a smooth manifold

2) The dimension of C is DOF, which is $6\,(N-1) - \sum_J N_J$

3) A small neighborhood of the complete family is realizable by simply constrained mechanisms

Now consider the parameter space X of the complete family of a simply-constrained mechanism M. The mechanism M represents a point in X at which the constraint equations intersect transversely in $E(3)^{(N-1)}$. This transversality condition is an open condition, hence is true for a Zariski-open set U in X. The mechanisms corresponding to points in U are simply-constrained. Therefore, it makes sense to speak of a family of simply-constrained mechanisms; we call such a family a simple family. Note that not every mechanism in a simple family need be simply-constrained -- in fact, there will be closed sets in X for which the configuration spaces are singular algebraic sets. On the other hand, every Zariski-open set of mechanisms containing one of these singular algebraic sets will contain simply-constrained mechanisms, so the singular behavior inside a simple family is, in some sense, not so bad.

The difference between example 1b and 2 is now clear -- example 1b lies in a simple family while example 2 does not. A mechanism that does not lie in a simply family is called over-constrained.

The general concept of mechanism families seems natural from a mathematical viewpoint, though our current understanding of them is quite limited. In his undergraduate thesis [Walker], Walker analyzed the class of simple families of single-loop planar linkages. For this class the complete families, when normalized by a geometric scale factor, are geometric simplexes (higher-dimensional tetrahedra) of dimension one less than the number of bars in the linkage. Walker determined the structure of the algebraic subsets corresponding to singular mechanisms within the families, and constructed Morse functions on the corresponding real configuration spaces to determine the changes in topology for families of mechanisms passing through those singular sets. He also completed the classification of two-DOF configuration spaces for mechanisms of this type, and obtained partial results for the higher-dimensional cases. The local parametrization problem appears tractable for planar linkages of arbitrary loop-structure, though Walker did not address it. We are not aware of any other references along these lines.

2.4 Statement and Discussion of the Main Problem

We are interested in the following problem:

MAIN PROBLEM: Determine, by algebraic methods, explicit parametrizations of the configuration space for a family of specified mechanisms. This parametrization should be of minimal degree.

The rest of this section is devoted to discussion of the main problem.

As we have already seen, for most mechanisms in a simple family (i.e. a Zariski-open set), the configuration spaces are smooth manifolds of dimension DOF as defined by equation (1b). When C is a manifold, the explicit parametrizations we have in mind form a special, algebraic coordinate atlas for C.

In general (simply-constrained or not), the construction is as follows.

From the set J of all joints, select (if possible) a subset Q such that:

1) $\sum_{j \in Q} D_j = DOF$

2) the degrees of freedom of joints in Q are independent

For many practical examples we have analyzed, subsets Q satisfying the first condition do exist (see example 2 above for a counterexample). The meaning of the second condition will become apparent immediately.

Consider the situation where all the free parameters associated with the joints in Q are frozen

at some fixed values. When C is a smooth manifold, the two conditions above are those required to make the corresponding set of configurations a discrete, zero-dimensional set (i.e. a set of points). This discrete set is the zero set of the ideal generated by finitely many polynomial equations and so is a finite set. The number of points in that set (counting multiplicities) is denoted $D_{C/Q}$, the degree of C over Q. It is not always the case that sets Q can be found satisfying the above two conditions even for simple families (e.g. example 1b), but there are enough examples to make this special condition worth consideration.

For the rest of this subsection we assume a set of joints Q satsifying the above two conditions can be found.

The geometric situation can viewed in another way as follows. Suppose the configuration space C of the mechanism M is a smooth manifold and let Q be a set of joints as above. Let H_Q be the product manifold of all the geometric subgroups of E(3) corresponding to the relative motions of each joint in Q. Then there is a finite branched covering map $\pi : C \to H_Q$ [Gunning]. The fiber over each point h of H_Q is the finite set of points in C for which the joints in Q have the values specified by h. The number $D_{C/Q}$ is the branching order of this covering. The set B_π defined to be the union of multiple points in C with respect to π is called the branch locus of π.

Now we observe that the construction above for the configuration space C of a single mechanism extends naturally to the configuration spaces of all mechanism in a family (assuming the family is realizable, as in the case of simple families). For any configuration-space manifold C' in that family the number $D_{C'/Q}$ is the same. Because the number $D_{C'/Q}$ is independent of the choice of C', we change notation and denote this number by D_Q.

We are finally able to provide a precise statement of the main problem. Consider a family of mechanisms and a subset Q satisfying the above conditions 1 and 2. Determine a mapping (in

the form of a computational algorithm) from the product space H_Q to a set of polynomials in flag form for the remaining joint positions [Cox]. That flag form should have the property that, for a Zariski-open set of mechanisms in the family, and for every fixed point in H_Q, the number of discrete solutions (counting multiplicity) in that flag is D_Q.

The motivation for the main problem arises in the framework of computer-aided mechanism design, analysis and dynamic simulation. From the flag form it is possible to compute efficiently all allowed positions of a mechanism. By constructing the flag for a family of mechanisms, we can use the same basic algorithm for mechanisms in a general class, thereby allowing a mechanism designer to change the dimensions (e.g. lengths) of the bodies, the orientations and positions of the joints, in a interactive manner. For simple families, small changes can be made to every one of the parameters while staying within the class of simply-constrained mechanisms. Our computational procedure provides a good coordinate atlas for all mechanisms in that local family.

Once the algebraic equations are formulated, a polynomial flag of the type just described can be computed by the Groebner-Basis algorithm [Buchberger1,2]. In practice, however, the Groebner-Basis algorithm does not provide answers quickly (see [EM] for a discussion of computational issues). The subject of this paper is an approach that produces a flag of the same type by more practical computational methods.

In general, a single subset Q will not provide a good atlas for an entire configuration manifold because the projection map π generally has a non-empty branch locus. To form a good atlas, one may consider a finite collection of subsets $Q_1, \cdots, Q_K$ for which the intersection of the corresponding branch loci is empty. A good atlas is needed for working on dynamical problems such as mechanism control.

3. DEVELOPMENT OF THE MULTI-AFFINE SYSTEM

For a given mechanism, each of the rigid bodies in W has its own reference coordinate system. From a kinematic viewpoint, the role of the body coordinate system is to allow specification of the geometric relation (separation, relative orientation in space) among the joints on that body. The mathematical data describing the body in its coordinate chart is a subset of the data required by a machinist to build it. Within each coordinate chart a three-dimensional coordinate system is specified, and the coordinates of salient parts of the body are identified. In the following we will use the notation U_j to denote the coordinate chart of body j, and (x_j, y_j, z_j) to denote the coordinate functions in U_j. When discussing properties that hold for a general chart we often omit the subscript j.

3.1 Loop Closure Equations

We first discuss the notion of transition functions. Roughly speaking, a transition function is a mapping from one coordinate chart to another that identifies those points that are joined together in the assembled mechanism. In the next paragraph we clarify this notion.

Let SO(3) denote the special orthogonal group of real orthogonal matrices of determinant 1. The Euclidean group E(3) is the group of all mappings ϕ from R^3 to R^3 of the form:

$$\phi(x) = A\,x + b \tag{3.1}$$

where A is in SO(3) and b is in R^3. There is a standard representation of E(3) as a subgroup

of 4×4 matrices by the mapping ρ defined as follows:

$$\rho(\phi) = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} \tag{3.2}$$

where ϕ, A and b are as in equation 1. The value of the transformation ϕ applied to the three-vector x is realized by identifying x with the four-vector x', where

$$x' = \begin{bmatrix} x \\ 1 \end{bmatrix} \tag{3.3}$$

and computing the usual matrix product

$$[\phi(x)]' = \rho(\phi)\, x' \tag{3.4}$$

Consider two coordinate charts, U_1 and U_2, for a pair of bodies connected by a joint. Using the 4×4-matrix notation, the set of Euclidean motions ϕ_{12} satisfying the constraint equations have the following forms, depending on joint type.

First, for an R-type joint, it is easy to check that the 1-parameter transition functions are of the form

$$\rho(\phi_{21}(\theta)) = \begin{bmatrix} R(\theta) & b - R(\theta)a \\ 0 & 1 \end{bmatrix} \tag{3.5}$$

where a, b are constant vectors that represent fixed points of the relative motion between the

two bodies in the two coordinate systems. The matrix $R(\theta)$ is a rotation matrix of the form:

$$R(\theta) = \begin{bmatrix} V & V_{perp} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) \\ 0 & -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} W & W_{perp} \end{bmatrix}^T \tag{3.6}$$

for θ an arbitrary angle, V the axis of rotation in U_2, W the axis of rotation in U_1, V_{perp} an orthonormal 2×3 complement to V, and W_{perp} an orthonormal 2×3 complement to W.

For a P-type joint, the one-parameter transition functions are of the form:

$$\rho(\phi_{21}(t)) = \begin{bmatrix} I_3 & t\,a \\ 0 & 1 \end{bmatrix} \tag{3.7}$$

where t is a scalar variable and a is a constant vector in the direction of the motion. The matrix I_3 is the 3×3 identity matrix.

For an S-type joint, the most general transition function is of the form:

$$\rho(\phi_{21}(R)) = \begin{bmatrix} R & b - Ra \\ 0 & 1 \end{bmatrix} \tag{3.8}$$

where a, b are constant vectors that represent fixed points of the relative motion between the two bodies in the two coordinate systems. R is a general rotation matrix.

Now that the transition functions have been defined it is an easy matter to write down the loop-closure equations (LCEs). The loop-closure equations are obtained by taking the

composition of transition functions around a closed loop. Starting, say, in coordinate chart U_1, pick a general point x_1. Any transition function ϕ_{21} maps x_1 to its image x_2 in the coordinate chart U_2 (change of coordinates). The point x_2 is mapped in turn to x_3 in U_3 by a transition function ϕ_{32}, and so on, until finally the point x_L in U_L is mapped by the function ϕ_{1L} to y_1 in the original chart U_1. When the parameters of the transition functions are those one would measure on the mechanism in a realizable configuration, the compatibility relation $y_1 = x_1$ is satisfied. This compatibility relation holds for all x_1 in U_1 for which the composition is defined. This relation is represented mathematically by a loop closure equation.

For a realizable configuration, the compatibility relation must hold for the following geometric reason. Viewing the transition functions as changes of coordinates associated with the different bodies, we see that y_1 and x_1 are the coordinates of the same geometric point in the same coordinate chart U_1, so they are equal. The associated loop closure equation states that the composition is the identity map in E(3).

Consider the example of the 7R mechanism. Because this mechanism has only a single loop, all loop equations have the form:

$$\phi_{j+7\ j+6}(\theta_{j+6})\ \phi_{j+6\ j+5}(\theta_{j+5})\ \cdots\ \phi_{j+1\ j}(\theta_j) = \text{identity} \tag{3.9}$$

where the subscripts are evaluated mod 7. As an example, one of these equations in terms of the 4×4 matrices is (pick coordinates so that $b_j = 0$):

$$\begin{bmatrix} R_{17}(\theta_7) & -R_{17}(\theta_7)\ a_7 \\ 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} R_{32}(\theta_2) & -R_{32}(\theta_2)\ a_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_{21}(\theta_1) & -R_{21}(\theta_1)\ a_1 \\ 0 & 1 \end{bmatrix} = I \tag{3.10}$$

where I is the 4×4 identity matrix, $R_{j+1\ j}(\theta_j)$ is the subscripted version of equation 6, and a_j

is the subscripted version of equation (3.5).

The reader familiar with differential geometry will recognize the significance of the above construction in more concise but abstract terms. What we have described is a special type of flat-Euclidean manifold structure [Charlap] on a geometric space associated with the mechanism. This abstract viewpoint was the original motivation behind the transition function approach. This abstract point of view does not tell us how to proceed with the computations, but it did get us started. The quaternion pair notation described next provides the computational key.

3.2 The Quaternion Pair Notation

Quaternions are often used in kinematic analysis, so the mechanism community is familiar with their properties. Even so, it appears that the quaternion-based approach we present here has not been exploited in mechanism analysis.

The point of introducing quaternions is to transform the loop closure equations (LCEs) derived in Section 3.1 into a simpler form. The simplification is not merely cosmetic, it is a basic reduction of the system of equations into an equivalent but more tractable form.

We assume the reader is familiar with the basics of quaternions. A general reference is [Porteus]. The section below is a slight generalization, of the more detailed presentation in [ME], so some details are omitted.

3.2.1 Unit Quaternions and SO(3)

There is a well-known covering map from the unit quaternions Q(1) to the group of rotation

matrices SO(3). This brief section reviews that construction and some of its immediate corollaries.

Let p be a nonzero quaternion. Then pwp* is a pure quaternion if and only if w is a pure quaternion. By the mapping $w \rightarrow pwp^*$ each nonzero quaternion p defines an invertible linear transformation S(p) of T. We will primarily consider this transformation in the special case where p is a unit quaternion, in which case S(p) is in SO(3). Relative to the basis {i,j,k} of T (which we tacitly assume in the sequel), the matrix S(p) associated with p has the form:

$$S(p_0, p_1, p_2, p_3) = 2\begin{bmatrix} p_0^2 + p_1^2 & p_1p_2 - p_0p_3 & p_1p_3 + p_0p_2 \\ p_1p_2 + p_0p_3 & p_0^2 + p_2^2 & p_2p_3 - p_0p_1 \\ p_1p_3 - p_0p_2 & p_2p_3 + p_0p_1 & p_0^2 + p_3^2 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad (3.11)$$

The mapping $p \rightarrow S(p)$ projects the unit quaternions Q(1) onto the rotation matrices SO(3). For each matrix K in SO(3) there are two quaternions, p and -p, such that $S(p) = S(-p) = K$. Furthermore, $S(p_2p_1) = S(p_2)S(p_1)$.

The geometric link between rotation matrices and their associated unit quaternions is easily derived. Specifically, any unit quaternion q has the form:

$$q = \cos(\gamma) + \sin(\gamma)\, w \qquad (3.12)$$

where w is a unit-length, pure quaternion. The matrix S(q) is a rotation by the angle 2γ about the axis in the w-direction.

3.2.2 The Quatenion-Pair Cover of the Euclidean Group

Now we turn attention to a special group defined by quaternion pairs. Specifically, we will look at the set $Q(1) \times T$ of ordered pairs of quaternions (q,w) and define a group structure on this set. Let q_1, q_2 be unit quaternions and w_1, w_2 be pure quaternions. Define the product:

$$(q_2, w_2)(q_1, w_1) = (q_2q_1, w_2 + q_2w_1q_2^*) \tag{3.13}$$

Observe that the product is well-defined in $Q(1) \times T$. Note that (1,0) is the identity element, and

$$(q, w)^{-1} = (q^*, -q^*wq) \tag{3.14}$$

The set $Q(1) \times T$ with this product forms a group, which we call the quaternion-pair group. This group is the universal cover of the Euclidean group E(3). The full power of the covering space structure is not needed for our application, all we need is the definition of the mapping. Therefore, we construct the covering map S_* below, but we do not address its topological properties.

The covering map from the quaternion-pair group to E(3) is closely related to the map S defined in Section 3.2.1 that maps Q(1) onto SO(3). Let ϕ_1, ϕ_2 be two elements of E(3) and suppose for j = 1,2:

$$\rho(\phi_j) = \begin{bmatrix} R_j & a_j \\ 0 & 1 \end{bmatrix} \tag{3.15}$$

Pick q_j such that $S(q_j) = R_j$, and let w_j be the pure quaternion

$$w_j = a_{j,1}i + a_{j,2}j + a_{j,3}k \tag{3.16}$$

where $a_j = (a_{j,1}, a_{j,2}, a_{j,3})$. Defining the map $S_*[(q_j, w_j)] = \phi_j$, it is an easy computation to verify that

$$S_*[(q_2, w_2)(q_1, w_1)] = \phi_2\, \phi_1 \tag{3.17}$$

The mapping S_* is clearly onto, it is a two-to-one mapping like π.

3.3 Transition Functions in Quaternion-Pair Form

The primary goal of this subsection is to determine a quaternion-pair form for transition functions equivalent to the expressions in equations (3.5) through (3.8) in Section 3.1. The idea is to represent the transition function $\phi_{j+1\,j}$ (we ignore the dependence of ϕ on its parameters) by a quaternion pair:

$$\sigma(\phi_{j+1\,j}) = (q_j, w_j) \tag{3.18}$$

where q_j is a unit quaternion and w_j is a pure quaternion. We begin with the special case of R-joints.

3.3.1 Representing R-joints

Consider the expression in equation (3.5), and let ϕ denote ϕ_{21}. Fix θ and select q, one of the

two quaternions such that $S(q) = R(\theta)$. Let α , β be the pure quaternions associated with the vectors a and b. The results of the previous subsection suggest that we use

$$\sigma(\phi) = (q, \beta - q\alpha q^*) \tag{3.19}$$

for the quaternion pair representing ϕ. Furthermore, if ϕ_1 and ϕ_2 are two transition functions corresponding to $(q_1, \beta_1 - q_1\alpha_1 q_1^*)$ and $(q_2, \beta_2 - q_2\alpha_2 q_2^*)$, then the composition map is the product (see equation (3.19)):

$$\sigma(\phi_2\phi_1) = (q_2q_1, \beta_2 + q_2(\beta_1 - \alpha_2)q_2^* - q_2q_1\alpha_1 q_1^* q_2^*) \tag{3.20}$$

There is a pattern in expression on the right hand side of this last equation that will be clarified in Section 3.4.

A vital detail in the quaternion-pair representation is to include the θ-dependence, which is described in the following circle lemma.

Circle Lemma: Let $R(\theta)$ be the one-parameter family of rotation matrices defined in equation (3.6). There is a pair of unit quaternions (p,q), with p orthogonal to q, such that the great circle C_1 in Q(1) defined by

$$C_1 = \{\cos(\gamma)\, p + \sin(\gamma)\, q \mid 0 \leq \gamma \leq 2\pi\} \tag{3.21}$$

is mapped by S onto the circle C_2 in SO(3) defined by

$$C_2 = \{R(\theta) \mid 0 \leq \theta \leq 2\pi\} \tag{3.22}$$

The proof of this lemma is found in [ME].

A more geometric statement of the circle lemma is: circles in the rotation group are covered by great circles in the unit quaternion group. The significance of this result will be realized in Section 3.4.

3.3.2 Representing S-joints

Consider the expression in equation (3.8), and let ϕ denote ϕ_{21}. Select q, one of the two quaternions such that S(q) = R and let α , β be the pure quaternions associated with the vectors a and b. Then as above we choose:

$$\sigma(\phi) = (q, \beta - q\alpha q^*) \tag{3.23}$$

for the quaternion pair representing ϕ. The formulas for products of transition functions associated with both S-joints and R-joints are the same as equation (3.20) above. The only difference is that the variable q is now an arbitrary unit quaternion, and not of the special form stated in the circle lemma.

3.3.3 Representing P-joints

This is the easiest case. For $\phi(t)$ as in Equation (3.7) the answer is:

$$\sigma(\phi(t)) = (q_0, t\,\alpha) \tag{3.24}$$

where q_0 is a constant unit quaternion and α is the pure quaternion associated with the vector a.

3.4 The Multi-Affine Equations

Examining the quaternion-pair representations described in the last section we find that the LCEs have the following form:

$$(q_1, w_1) \cdots (q_L, w_L) = (1,0) \tag{3.25}$$

We now observe a particular feature of the expression on the LHS of this last equation:

1) the unit-quaternion part of a product is the product of the unit quaternions
2) each summand of the pure quaternion part is a sequential conjugate expression

By sequential conjugate expression, we mean an expression of the form:

$$F(w) = q_1 \cdots q_m \, w \, q_m^* \cdots q_1^* \tag{3.25}$$

where w is a pure quaternion and m varies from 0 to L. The point is that, for all m, when a sequential conjugate expression is right-multiplied by the unit-quaternion part of the product, the result is multilinear in all the unit quaternions q_i.

In addition, the equation:

$$Qu(q_1 \cdots q_L) = 0 \tag{3.27}$$

where the function Qu means "pure-quaternion part," provides another set of multi-linear

equations in the variables q_i.

These multi-linear equations in q_i can be reduced to multi-affine equations in fewer variables by the following devices:

1) For each R-joint in the product, use the Circle Lemma and divide by cos (γ)

2) For each S-joint in the product, divide by $q_0 = Re(q)$

Let z_i denote the variable $\tan(\frac{\theta_i}{2})$, and define the variables e_j, f_j, g_j by:

$$\begin{bmatrix} e_j \\ f_j \\ g_j \end{bmatrix} = \frac{1}{q_{j,0}} \begin{bmatrix} q_{j,1} \\ q_{j,2} \\ q_{j,3} \end{bmatrix} \tag{3.28}$$

Then the LCEs become multi-affine in the variables z_i, e_j, f_j, g_j.

Observe that the LCEs are already multi-affine in the variables t_k associated with P-joints. The discussion just concluded brings us to the

Main Result: The loop closure equations (LCEs) generate a host of multi-affine equations, forming a modular system of a very special type. The number of independent variables z_i, e_j, f_j, g_j, t_k appearing in these equations is equal to the sum over all joints of the number of joint degrees of freedom. In general, there are far more equations than unknowns.

The primary benefit of this result lies in providing a special structure to the set of equations whose solutions describe the configuration spaces of mechanisms of this type. An additional feature is the systematic approach it provides for generating these equations.

An approach for numerical solution of multi-affine equations of this type is illustrated in the solution to the 7R problem presented in [ME].

4. SUMMARY

The discussion below is a summary of our six-step, general approach for mechanism kinematic analysis. The details for a particular application are presented in [ME].

We have seen that the quaternion-pair approach described below can be applied to spherical, prismatic and cylindrical joints as well as the revolute joints illustrated in [ME]. Systems of multi-affine equations are obtained in the same way. Multi-loop mechanisms can also be handled. One of the primary goals of our research effort to use the theory on problems of moderate size to determine the practical (computational) limitations of this general approach on more complex mechanisms.

4.1 Reference Coordinate Systems

Each rigid piece of the mechanism has its own reference coordinate system. The concept is simple -- each piece can be thought of as a separate entity standing by itself in Euclidean three-space.

4.2 Transition Functions

The transition functions represent the translation and rotation (Euclidean motion) associated with the change of reference coordinate systems from one body to the next. The transition function contains the fixed parameters of the body as well as the free variables associated with the joint.

4.3 Quaternion-Pair Notation

The quaternion-pair notation consists of a pair of quaternions that together represent a single Euclidean motion: the first is a unit quaternion that represents the rotation and the second is a pure quaternion that represents translation. The advantage of this notation is that it leads to loop equations that are multi-affine in the joint variables. These multi-affine equations are of lower total degree and therefore are much easier to solve than the multi quadratic equations that arise from using the standard 4×4 matrices containing rotation matrices and a translation vector.

The quaternion-pairs form a mathematical group under a product defined in Section 3.2.2. The composition of transition functions is obtained by multiplying the corresponding quaternion pairs.

4.4 Loop Equations

The loop closure equations are obtained by setting the product of all the quaternion-pair transition functions around a loop equal to the quaternion-pair identity.

4.5 Multi-Affine Equations

After multiplication by the appropriate factors, the resulting loop equations are multi-affine in the joint variables. Because the quaternion-pair multiplication is not commutative, additional independent equations can be formed by cyclic permutations of the transition functions (starting at a different body when forming the loop). Some of the equations obtained in this way involve fewer variables, so more multi-affine equations of the same degree can be formed by multiplying those equations by each of the missing variables. In this way, a large number of affine equations can be formed.

4.6 Elimination Techniques

Standard polynomial elimination techniques (such as Groebner basis) work on general systems of polynomials, but their run times are doubly exponential in the number of noninteger parameters. This makes them unusable for mechanisms of even moderate complexity. Consequently we developed specific elimination techniques based on matrix polynomial methods [Wedderburn] that are much faster for the specific (multi-affine) type of systems of polynomials arising from mechanism kinematics. These specific techniques consist of generalized eigenvalue problems, explicit polynomial representation of the null space of affine matrices, and numerical methods from linear algebra.

BIBLIOGRAPHY

[Buchberger1] Buchberger, B., "Grobner Bases: An Algorithmic Method in Polynomial Ideal Theory," Chapter 6 of Multidimensional Systems Theory, Ed. N. K. Bose, Reidel Publishing Company, 1985.

[Buchberger2] B. Buchberger, "Applications of Groebner Basis in Non-linear Computational Geometry," in "MATHEMATICAL ASPECTS OF SCIENTIFIC SOFTWARE", edited by J. R. Rice, Springer-Verlag, 1988.

[Charlap] Charlap, L., Bieberbach Groups and Flat Manifolds, Springer Verlag, 1986.

[Cox] Cox, D., J. Little and D. O'Shea, Ideals, Varieties, and Algorithms, Springer-Verlag, 1992.

[Duffy] Duffy, J., Analysis of Mechanisms and Robot Manipulators, John Wiley and Sons, 1980.

[Duffy2] Duffy, J., and Crane, C., 1980, "A Displacement Analysis of the General 7-Link 7R Mechanism," Mech. Mach. Th. 15(3), pp. 153-169.

[EM] Elgersma, M. and B. Morton, "A Polynomial-Runtime Algorithm for Multi-Affine Equations," unpublished manuscript, work in progress, March 1998.

[Freudenstein] Freudenstein, F., "Kinematics: Past, Present and Future," Mechanism and Machine Theory, Vol. 8, No. 2, pp. 151-161, 1973.

[Gunning] Gunning, R., Introduction to Holomorphic Functions of Several Variables, Wadsworth & Brooks/Cole, 1990.

[Hunt] Hunt, K. H., Kinematic Geometry of Mechanisms, Oxford University Press, 1990.

[Innocenti1] Innocenti, C. and V. Parenti-Castelli, "Forward kinematics of the general 6-6 fully parallel mechanism: an exhaustive numerical approach via a mono-dimensional-search algorithm," 22nd ASME Biennial Mechanisms Conference, Scottsdale, AZ, Sep 13-16, 1992.

[Innocenti2] Innocenti, C. and V. Parenti-Castelli, "Direct position analysis of the Stewart platform mechanism," Mechanism & Machine Theory vol. 25, no. 6, pp. 611-621, 1990.

[Lee1] Lee (Li), H.-Y., and Liang, C.-G., "A New Vector Theory for the Analysis of Spatial Mechanisms," Mech. Mach. Th. 23(3), pp. 209-217, 1988.

[Lee2] Lee (Li), H.-Y., and Liang, C.-G., "Displacement Analysis of the General Spatial 7-Link 7R Mechanism," Mech. Mach. Th. 23(3), pp. 219-226, 1988. Also presented at the 4th National Conference on Mechanisms in Yantai, Shandong Province, China (1986).

[Macaulay] Macaulay, F. S., The Algebraic Theory of Modular Systems, Cambridge University Press, 1916.

[ME] Morton, B. and M. Elgersma, "A Computational Algorithm for 7R Spatial Mechanisms," Submitted to Journal of Robotic Systems 21 June 1993.

[Porteous] Porteous, I., Topological Geometry, Cambridge University press, 1981.

[Ragovan] Raghavan, M., and Roth, B., "Kinematic Analysis of the 6R Manipulator of

General Geometry," Proceedings of the 5th International Symposium on Robotics Research, edited by H. Miura and S. Arimoto, MIT Press, Cambridge, 1990.

[Roberson] Roberson, R. and R. Schwertassek, Dynamics of Multibody Systems, Springer Verlag, 1988.

[Roth] Roth, B., J. Rastegar and V. Scheinman, "On the Design of Computer Controlled Manipulators," First CISM-IFToMM Symposium on Theory and Practice of Robots and Manipulators, Vol. 1, pp. 93-113, Udine, 1974.

[Thurston] Thurston, W. P. and J. R. Weeks, "The Mathematics of Three-Dimensional Manifolds," article in Scientific American, pp. 108-120, July 1984.

[Walker] Walker, K., "Configuration Spaces of Linkages," Undergraduate Thesis Submitted to Princeton University Matematics Department, 23 April 1985.

[Wedderburn] J. H. M. Wedderburn, Lectures on Matrices, Dover, 1964.

[Wittenberg] Wittenberg, J., Dynamics of Systems of Rigid Bodies, Teubner, 1977.

A Polynomial-Runtime Algorithm for Multi-Affine Equations

Michael R. Elgersma and Blaise G. Morton
MS MN65-2810
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418

Manuscript Note: the method presented here should be compared with other algorithms (recently published) to construct an optimal size square sparse resultant matrix for multi-affine (and multi-graded) systems of polynomials. See references:

I. M. Gelfand, M. M. Kapronov, and A. V. Zelevinski, "Hyperdeterminants," Advances in Mathematics, 96, 1992.

Bernd Sturmfels,

Rojas

I. Z. Emiris

Abstract

This paper deals with $n = n_1+n_2+...+n_k$ equations, multi-affine in k sets of variables with n_i variables in the i^{th} set. In [St] a formula is given for, d, the number of solutions, and a degree d polynomial in one variable is formed from a hyper-determinant function of the coefficients and one variable. The d roots of this polynomial are the value of that variable at the d solution points.

In the current paper, we give an alternative proof for the number of solutions, and we form a $d \times d$ eigenproblem whose eigenvalues are the d values of one variable, and the other variables are ratios of entries of the eigenvectors. This technique of forming the eigenproblem directly, then solving it numerically gives an $O(d^3)$ algorithm (where $d >> n$). The previous technique [St],[Can], [GKZ] of forming the degree d polynomial is $O(d^n)$ or $O((nk)^n)$?

The three steps in our algorithm are:

1) Use dilation to form a $[min(n_i)d] \times [min(n_i)d]$, rank-d, eigenproblem.

2) If $min(n_i) > 1$, use deflation to reduce the size of the eigenproblem

   a) Compute svd of a matrix of size $[min(n_i)d] \times d$.

   b) Compute svd of a matrix of size $[(min(n_i) - 1)d] \times [min(n_i)d]$.

3) Solve a $d \times d$ generalized eigenproblem.

The matrices involved in each of the above three steps are either linear in the coefficients of the original equations, or are the product of orthogonal matrices times matrices that are linear in the coefficients of the original equations. The d eigenvalues give the d values of one of the variables, while the corresponding values of each of the other variables are given by a ratio of two linear combinations of the eigenvector. The runtime of the algorithm is dominated by the runtime of the svd's and the eigensolver. The total number of floating point operations is $10[min(n_i)d]^3$.

This paper also deals with the case where there are more equations than unknowns. The extra equations may eliminate some or all of the solutions.

Chapter 1

Same Number of Equations as Unknowns

1.1 Introduction

Systems of multi-affine equations arise from applications where some specified input vector $\underline{u}_1$, is multiplied by a sequence of k subsystems to give a specified output vector $\underline{y}_k \in R^{(n_1+n_2+...+n_k)}$. For $i = 1$ to k, subsystem i is linear in its own subset of n_i variables, and is of the form:

$$\underline{y}_i = H_i(\underline{x}_i)\underline{u}_i \tag{1}$$

where

$$H_i(\underline{x}_i) = [H_{i,0} + H_{i,1}x_{i,1} + H_{i,2}x_{i,2} + ... + H_{i,n_i}x_{i,n_i}] \tag{2}$$

Let

$$m_{i+1} \times m_i = size(H_i(\underline{x}_i)) \tag{3}$$

To ensure as many equations as unknowns at each step:

$$m_i \geq \sum_{j=1}^{i} n_j \qquad for \;\; i \leq k \tag{4}$$

and

$$m_{k+1} = \sum_{j=1}^{k} n_j \tag{5}$$

The product of these subsystems then gives:

$$\underline{u}_{i+1} = \underline{y}_i \tag{6}$$

$$\underline{y}_k = \left(\prod_{i=1}^{k} [H_{i,0} + H_{i,1}x_{i,1} + H_{i,2}x_{i,2} + ... + H_{i,n_i}x_{i,n_i}] \right) \underline{u}_1 \tag{7}$$

Given the value of the output vector $\underline{y_k} \in R^{(n_1+n_2+...+n_k)}$ and the value of the input vector, $\underline{u_1}$, we have $n_1 + n_2 + ... + n_k$ multi-affine equations in $n_1 + n_2 + ... + n_k$ unknown $x_{i,j}$.

Applications that involve systems of multi-affine equations include:

Generalized $n \times n$ eigenvalue problem, where $k = 2$, $n_1 = n - 1$, $n_2 = 1$, $\underline{y_2} = \underline{0}$, $H_{1,j}u_1$ = column j+1 of the n-dimensional identity matrix. This gives:

$$0 = [H_{2,0} + x_{2,1}H_{2,1}] \begin{bmatrix} 1 \\ x_{1,1} \\ x_{1,2} \\ . \\ . \\ . \\ x_{1,n-1} \end{bmatrix} \tag{8}$$

Robotics [ME96], where each subsystem is a link in a robot arm, the variables are associated with the joint degrees of freedom, the inputs are the base position and attitude, and the outputs are the end-effector positions and attitude.

Electronic circuits, where multi-input multi-output subsystem i contains n_i op-amps, the variables are the impedances ratios for each op-amp, the input is a vector of voltages input to the first stage, and the output is a vector of voltages output from the last stage.

In systems like these, we can solve for all values of each parameter that give the specified output for the given input. In each case, the problem can be converted into a generalized eigenproblem.

A 1000×1000 eigenproblem requires approximately $8 * (1000)^3$ floating point operations, which can be done in approximately two minutes on a 100 MFlop computer. Several sets of exponents that result in $d \leq 1000$ solutions are given in the table below. The number of floating point operations required for the entire solution is $10[min(n_i)d]^3$ which is given in the last column. A 100 MFlop machine does 10^8 floating point operations per second, so the numbers in the last column are all with that exponent.

$n = \sum_{i=1}^{k} n_i$	n_1	n_2	n_3	n_4	n_5	n_6	$d = \frac{(n_1+n_2+...+n_k)!}{n_1!n_2!...n_k!}$	$10[min(n_i)d]^3$
6	1	1	1	1	1	1	720	37e+08
7	2	2	2	1			630	25e+08
7	3	2	2				210	7e+08
7	3	3	1				140	.3e+08
8	4	4					70	2e+08
32	30	1	1				992	98e+08
1000	999	1					1000	100e+08

1.2 Segre Embeddings

Multi-affine equations on $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$ can be rewritten as linear equations on $P^{(n_1+1)(n_2+1)...(n_k+1)-1}$ using the Segre embedding.

For i = 1 to k, let

$$X_i = \begin{bmatrix} 1 \\ x_{i,1} \\ x_{i,2} \\ . \\ . \\ . \\ x_{i,n_i} \end{bmatrix} \in P^{n_i} \qquad (9)$$

Let $v_0 = 1$. For j = 1 to k, let

$$v_j(X_1, X_2, ..., X_j) = \begin{bmatrix} v_{j-1} \\ x_{j,1}v_{j-1} \\ x_{j,2}v_{j-1} \\ . \\ . \\ . \\ x_{j,n_j}v_{j-1} \end{bmatrix} \in P^{(n_1+1)(n_2+1)...(n_j+1)-1} \tag{10}$$

So v_k is a mapping from $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$ to $P^{(n_1+1)(n_2+1)...(n_k+1)-1}$.

1.3 A Formula for the Number of Solutions

Theorem

The degree of the Segre embedding of $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$ in $P^{(n_1+1)(n_2+1)...(n_k+1)-1}$ is:

$$d = \frac{(n_1 + n_2 + ... + n_k)!}{n_1!n_2!...n_k!} \tag{11}$$

So intersecting $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$ with a codimension $n_1 + n_2 + ... + n_k$ hyper-plane ($n_1 + n_2 + ... + n_k$ linear equations) in $P^{(n_1+1)(n_2+1)...(n_k+1)-1}$ gives $d = \frac{(n_1+n_2+...+n_k)!}{n_1!n_2!...n_k!}$ point solutions.

The proof below uses techniques from intersection theory that can be found in [Fulton,84].

Proof

Let ϕ be the Segre embedding:

$$\phi : P^{n_1} \times P^{n_2} \times ... \times P^{n_k} \rightarrow P^{(n_1+1)(n_2+1)...(n_k+1)-1} \tag{12}$$

The hyper-plane class u in $P^{(n_1+1)(n_2+1)...(n_k+1)-1}$ pulls back to:

$$a_1 + a_2 + ... + a_k \in H^2(P^{n_1} \times P^{n_2} \times ... \times P^{n_k}) \tag{13}$$

where H^2 is the 2^{nd} Cohomology, and a_j is the hyper-plane class in P^{n_j}.

Look at the coefficients in the expansion of:

$$(a_1+a_2+...+a_k)^{(n_1+n_2+...+n_k)} \in H^{2(n_1+n_2+...+n_k)}(P^{n_1} \times P^{n_2} \times ... \times P^{n_k}) \quad (14)$$

This expression can be expanded by repeatedly applying the binomial expansion. Many of the resulting coefficients are zero because for each i from 1 to k we get:

$$a_i^{n_i+1} = 0 \quad (15)$$

We only get one nonzero coefficient in the expansion:

$$\begin{aligned}(a_1 + a_2 + ... + a_k)^{(n_1+n_2+...+n_k)} = \\ (a_1)^{(n_1+n_2+...+n_k)} + \\ (n_1 + n_2 + ... + n_k)(a_1)^{(n_1+n_2+...+n_k-1)}(a_2 + a_3 + ... + a_k) + ... \\ d(a_1^{n_1}a_1^{n_k}...a_k^{n_k}) + ...\end{aligned}$$

The only term in the expansion where each a_i has exponent less than $n_i + 1$ is the one where each a_i is taken n_i times.

$$(a_1 + a_2 + ... + a_k)^{(n_1+n_2+...+n_k)} = 0 + d(a_1^{n_1}a_2^{n_2}...a_k^{n_k}) + 0 + ... + 0 \quad (16)$$

The coefficient on this term is:

$$d = \prod_{j=1}^{k} \begin{pmatrix} \sum_{i=1}^{j} n_i \\ n_j \end{pmatrix} = \prod_{j=1}^{k} \begin{pmatrix} \sum_{i=j}^{k} n_i \\ n_j \end{pmatrix} = \frac{(n_1 + n_2 + ... + n_k)!}{n_1!n_1!...n_k!} \quad (17)$$

Therefore d is the degree of the Segre embedding, and there are d solutions to a set of $n_1 + n_2 + ... + n_k$ affine equations on this space.

■

1.4 Dilation Procedure for Forming an Eigenproblem

In this section, we use the root count d and its factorization to construct an algorithm for computing the solutions.

We will work with $n_1+n_2+...+n_k$ multi-affine equations on $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$. For i = 1 to k, let

$$X_i = \begin{bmatrix} 1 \\ x_{i,1} \\ x_{i,2} \\ . \\ . \\ . \\ x_{i,n_i} \end{bmatrix} \in P^{n_i} \tag{18}$$

The $n_1 + n_2 + ... + n_k$ multi-affine equations in the $x_{i,j}$ variables can be written as:

$$Av_k(X_1, X_2, ..., X_k) = 0 \in R^{n_1+n_2+...+n_k} \tag{19}$$

where the matrix A has $n_1+n_2+...+n_k$ rows and $(n_1+1)(n_2+1)...(n_k+1)$ columns. The vector $v_k(X_1, X_2, ..., X_k)$ lies in the Segre embedding of $P^{n_1} \times P^{n_2} \times ... \times P^{n_k}$ in $P^{[(n_1+1)(n_2+1)...(n_k+1)-1]}$.

The expression $Av_k(X_1, X_2, ..., X_k) = 0$ has fewer equations than monomials. We need to append more equations to get as many equations as monomials. We can multiply the equations by enough monomials to get as many equations as monomials in the new matrix equation. Let

$$sumX_i = 1 + x_{i,1} + ... + x_{i,n_i} \tag{20}$$

The vector $v_k(X_1, X_2, ..., X_k)$ contains the $(n_1+1)(n_2+1)...(n_k+1)$ monomials that appear in the expression:

$$(sumX_1)(sumX_2)...(sumX_k) \tag{21}$$

In order to get an eigenproblem with d solutions, where

$$d = \prod_{j=1}^{k} \begin{pmatrix} \sum_{i=j}^{k} n_i \\ n_j \end{pmatrix} = \frac{(n_1 + n_2 + ... + n_k)!}{n_1! n_1! ... n_k!} \tag{22}$$

we can use an equation of the form:

$$0 = M(X_k)w(X_1, X_2, ...X_{k-1}) \tag{23}$$

where $w(X_1, X_2, ...X_{k-1})$ contains d monomials. To get these d monomials, we proceed as follows.

For j = 1 to k, let

$$r_j = \sum_{i=j}^{k} n_i \tag{24}$$

Then

$$d = \prod_{j=1}^{k-1} \begin{pmatrix} r_{j+1} + n_j \\ n_j \end{pmatrix} \tag{25}$$

The expression:

$$(sumX_j)^{r_{j+1}} \quad contains \quad \begin{pmatrix} r_{j+1} + n_j \\ n_j \end{pmatrix} \quad monomials \tag{26}$$

Let the vector $w(X) \in R^d$ contain the d monomials in the expression

$$(sumX_1)^{r_2}(sumX_2)^{r_3}...(sumX_{k-1})^{r_k} \tag{27}$$

To get these d monomials, we can multiply the original $(n_1 + 1)(n_2 + 1)...(n_k + 1)$ monomials contained in $v_k(X)$ by D other monomials where

$$D = \prod_{j=1}^{k-1} \begin{pmatrix} r_{j+1} + n_j - 1 \\ n_j \end{pmatrix} = n_k d/(n_1 + n_2 + ... + n_k) \tag{28}$$

Let $g(X) \in R^D$ contain the D monomials in the following expression:

$$(sumX_1)^{(r_2-1)}(sumX_2)^{(r_3-1)}...(sumX_{k-1})^{(r_k-1)} \tag{29}$$

Let $G(X_1, X_2, ..., X_{k-1})$ be a matrix containing each of the D monomials in $g(X_1, X_2, ..., X_{k-1})$ times an identity matrix with $n_1 + n_2 + ... + n_k$ rows.

$$G(X_1, X_2, ..., X_{k-1}) = \begin{bmatrix} I_{n_1+n_2+...+n_k} g_1(X) \\ I_{n_1+n_2+...+n_k} g_2(X) \\ \cdot \\ \cdot \\ \cdot \\ I_{n_1+n_2+...+n_k} g_D(X) \end{bmatrix} \tag{30}$$

The matrix $G(X_1, X_2, ..., X_{k-1})$ has $(n_1 + n_2 + ... + n_k)D = n_k d$ rows and $(n_1 + n_2 + ... + n_k)$ columns.

The following then are $n_k d$ equations, in $(n_k + 1)d$ monomials.

$$G(X_1, X_2, ..., X_{k-1}) Av_k(X_1, X_2, ..., X_k) = 0 \in R^{(n_k d)} \tag{31}$$

The above expression contains the $(n_k + 1)d$ monomials found in:

$$(sumX_1)^{r_2}(sumX_2)^{r_3}...(sumX_{k-1})^{r_k}(sumX_k) \tag{32}$$

Separate all d monomials in $(X_1, X_2, ..., X_{k-1})$ into the vector $w(X)$, and all $n_k + 1$ monomials in X_k into a $n_k d \times d$ matrix $M(X_k)$.

$$0 = M(X_k) w(X_1, X_2, ..., X_{k-1}) \tag{33}$$

The $n_k d \times d$ matrix $M(X_k)$ is affine in X_k

$$M(X_k) = M_{k,0} + M_{k,1} x_{k,1} + ... + M_{k,n_k} x_{k,n_k} \tag{34}$$

The $M_{k,i}$ matrices are linear in the original data matrix A.

To convert this system into an eigenproblem with one of the variables as the eigenvalue, define the following two $n_k d \times n_k d$ matrices:

$$M_0 = [M_{k,0}, M_{k,1}, ..., M_{k,n_k-1}] \tag{35}$$

$$M_1 = [M_{k,n_k}, 0, ..., 0] \tag{36}$$

and the following size $n_k d$ vector:

$$\tilde{w} = \begin{bmatrix} w \\ x_{k,1} w \\ x_{k,2} w \\ . \\ . \\ . \\ x_{k,n_k-1} w \end{bmatrix} \tag{37}$$

Then the following equation is a size $n_k d \times n_k d$ generalized eigenproblem with at most d finite eigenvalues.

$$0 = M(X_k) w = (M_0 + x_{k,n_k} M_1) \tilde{w} \tag{38}$$

1.5 Deflation Removes Extraneous Roots

Generalized eigenproblems can be converted to regular eigenproblems if either M_0 or M_1 is full rank. Some robotics problems have roots at $x_{k,n_k} = \pm i$ which are extraneous. The roots at $x_{k,n_k} = \pm i, 0$, and ∞ can be removed by deflation.

If M_0 is low rank, then there are roots at $x_{k,n_k} = 0$.
If M_1 is low rank, then there are roots at $x_{k,n_k} = \infty$.
If $M_0 + iM_1$ is low rank, then there are roots at $x_{k,n_k} = +i$.
If $M_0 - iM_1$ is low rank, then there are roots at $x_{k,n_k} = -i$.

If $M_0 + xM_1$ is low rank for generic x, then the solution set contains sets of positive dimension, rather than just points.

The rest of this section uses deflation to eliminate roots at infinity. By interchanging the roles of M_0 and M_1, the roots at 0 can also be eliminated. To remove roots at $\pm i$, first do a linear fractional transformation that maps $+i$ to 0 and maps $-i$ to ∞.

If the system has repeated Jordan blocks with roots at the point being eliminated, the deflation procedure reduces the size of each of those Jordan blocks by one, each pass through the loop. So the maximum number of passes is equal to the maximum size of any Jordan block associated with the eigenvalue being eliminated.

The matrices M_0 and M_1 in the last section are square, $n_k d \times n_k d$, but the M_1 matrix multiplying x_{k,n_k} is low rank. (For matrices with more rows than columns, see the section on "Incomplete Intersection".) To get rid of the roots at infinity, we can get a smaller $d \times d$ generalized eigenproblem using deflation.

THEOREM (Eigen-Problem Deflation)

The finite pair λ, e is a solution to $0 = [M_0 + \lambda M_1]e$ iff the finite pair λ, f is a solution to $0 = [L_0 + \lambda L_1]f$ where L_0 and L_1 are defined in terms of the svd of the M_i as follows:

$$M_1 = [U_1, U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} V^* \tag{39}$$

Then taking the svd of $U_2{}^* M_0$:

$$U_2{}^* M_0 = [\mathcal{U}_1, \mathcal{U}_2] \begin{bmatrix} \mathcal{S}_1 & 0 \\ 0 & 0 \end{bmatrix} [\mathcal{V}_1, \mathcal{V}_2]^* \tag{40}$$

Using these svd's, L_0 and L_1 are defined by:

$$[L_0 + \lambda L_1] = U_1{}^* [M_0 + \lambda M_1] \mathcal{V}_2 \tag{41}$$

A solution pair λ, f of $0 = [L_0 + \lambda L_1]f$ is then mapped back to a solution pair λ, e of $0 = [M_0 + \lambda M_1]e$ by:

$$e = \mathcal{V}_2 f \tag{42}$$

PROOF

case 1) $U_1 = \emptyset$ and $\mathcal{V}_2 \neq \emptyset$ (infinite number of finite λ, e solutions)

This can only happen if $M_1 = 0$. In this case, U_1 is the empty set, so the equation $[L_0 + \lambda L_1]f = 0$ is satisfied by any λ and any f. Since $M_1 = 0$, U_2 is square, so $\mathcal{V}_2 = kernel(M_0) = kernel(M_0 + \lambda M_1)$, so $e = \mathcal{V}_2 f$ is a solution to $[M_0 + \lambda M_1]e$ for any λ and any f.

case 2) $\mathcal{V}_2 = \emptyset$ (no finite λ, e solutions)

This can only happen if $U_2{}^* M_0$ is full column rank. In this case, $U_2{}^*[M_0 + \lambda M_1] = U_2{}^* M_0$ which has no right kernel, so $[M_0 + \lambda M_1]e$ cannot be zero for any nonzero value of e. In this case, there are no nonzero solutions for e. Also, since $\mathcal{V}_2 = \emptyset$, $f = \emptyset$. So in this case, neither system has a solution.

case 3) $U_1 \neq \emptyset$ and $\mathcal{V}_2 \neq \emptyset$ (finite number of finite λ, e solutions)

Let

$$\begin{bmatrix} \tilde{f} \\ f \end{bmatrix} = [\mathcal{V}_1, \mathcal{V}_2]^* e \tag{43}$$

Multiplying the eigenproblem by the square orthogonal matrix $[U_1, U_2]^*$ gives:

$$0 = [U_1, U_2]^*[M_0 + \lambda M_1]e = [U_1, U_2]^*[M_0 + \lambda M_1][\mathcal{V}_1, \mathcal{V}_2] \begin{bmatrix} \tilde{f} \\ f \end{bmatrix} \tag{44}$$

Let

$$L_{1,1} = U_1^*[M_0 + \lambda M_1]\mathcal{V}_1 \tag{45}$$

$$L_{1,2} = U_1^*[M_0 + \lambda M_1]\mathcal{V}_2 \tag{46}$$

$$L_{2,1} = U_2^*[M_0 + \lambda M_1]\mathcal{V}_1 \tag{47}$$

$$L_{2,2} = U_2^*[M_0 + \lambda M_1]\mathcal{V}_2 \tag{48}$$

Then:

$$[U_1, U_2]^* [M_0 + \lambda M_1] [\mathcal{V}_1, \mathcal{V}_2] = \begin{bmatrix} L_{1,1} & L_{1,2} \\ L_{2,1} & L_{2,2} \end{bmatrix} \tag{49}$$

Since $U_2^* M_1 = 0$ and $U_2^* M_0 \mathcal{V}_2 = 0$, the $L_{2,2}$ block is identically zero.
With $L_{2,2}$ identically zero, the bottom part of the matrix equation gives:

$$0 = L_{2,1}\tilde{f} + L_{2,2} f = L_{2,1}\tilde{f} \tag{50}$$

Plugging in the expression for $L_{2,1}$ then gives

$$0 = U_2^*[M_0 + \lambda M_1]\mathcal{V}_1\tilde{f} = (U_2^* M_0)\mathcal{V}_1\tilde{f} = \mathcal{U}_1\mathcal{S}_1\tilde{f} \tag{51}$$

Since the matrix $\mathcal{U}_1\mathcal{S}_1$ has full column rank, this implies that

$$\tilde{f} = 0 \tag{52}$$

so

$$L_{1,1}\tilde{f} = 0 \tag{53}$$

This reduces the eigenproblem to:

$$0 = L_{1,2} f \tag{54}$$

which can also be written as:

$$0 = [U_1^*[M_0 + \lambda M_1]\mathcal{V}_2]\, f = [L_0 + \lambda L_1] f \tag{55}$$

■

After solving this new problem, the vector e can then be obtained from:

$$e = \mathcal{V}_2 f \tag{56}$$

The above procedure will have to be repeated if the resulting L_1 matrix is still low rank (replace M_0, M_1, and e with L_0, L_1, and f then repeat). The eigenvectors get mapped back by the product of the $\mathcal{V}_2$ matrices from each step of the iteration.

After the final L_1 is full rank, the following $\hat{d} \times \hat{d}$ eigenproblem gives the $\hat{d}$ values of $\lambda = x_{k,n_k}$ and the corresponding $\hat{d}$ values of the eigenvector f.

$$[L_0 + x_{k,n_k} L_1] f = 0 \tag{57}$$

The vector $w(X_1, X_2, ..., X_{k-1})$ is the first d elements of the vector $\tilde{w}$. The other $x_{i,j}$ can be extracted from the vector $w(X_1, X_2, ..., X_{k-1})$.

1.6 Pseudo Code for the Dilation Procedure

Let

$$\hat{M} = [M_0, M_1, ..., M_{n_1}] \tag{58}$$

Let

$$\hat{w}(X) = \begin{bmatrix} w(X) \\ x_{k,1} w(X) \\ x_{k,2} w(X) \\ . \\ . \\ . \\ x_{k,n_k} w(X) \end{bmatrix} \tag{59}$$

Then

$$\hat{M}\hat{w}(X) = G(X)Av_k(X) \tag{60}$$

When multiplying monomials in $G(X)Av_k(X)$ to get $\hat{M}\hat{w}(X)$, the exponents on the monomials in $G(X)$ and $v_k(X)$ add. Let $base_G(:,j)$ be a column of exponents on the $x_{i,m}$ in the j^{th} monomial in $G(X)$. Let the function *index* be defined such that $index_G(base_G(:,j)) = j$.

The j^{th} monomial in $G(X)$ is given by:

$$\prod_{l=1}^{k}\prod_{m=1}^{n_l} x_{l,m}^{base_G[(\sum_{i=1}^{l-1} n_i)+m,j]} \tag{61}$$

The r^{th} monomial in $v_k(X)$ is given by:

$$\prod_{l=1}^{k}\prod_{m=1}^{n_l} x_{l,m}^{base_v[(\sum_{i=1}^{l-1} n_i)+m,r]} \tag{62}$$

The product of the j^{th} monomial in $G(X)$ and the r^{th} monomial in $v_k(X)$ gives monomial number $s = index_{\hat{w}}(base_G(:,j) + base_v(:,m))$ in $\hat{w}(X)$. The s^{th} monomial in $\hat{w}(X)$ is given by:

$$\prod_{l=1}^{k}\prod_{m=1}^{n_l} x_{l,m}^{base_{\hat{w}}[(\sum_{i=1}^{l-1} n_i)+m,s]} \tag{63}$$

These *base* and *index* ideas can be used in the following pseudo code for filling in the M matrix.

for i = 1 to D Put $N = n_1 + n_2 + ... + n_k$ rows of A into M
  for j = 1 to $(n_1+1)(n_2+1)...(n_k+1)$ Put a column of A into M
    Compute exponents $base_G(:,i)$ on the i^{th} monomial in $G(X)$
    Compute exponents $base_v(:,j)$ on the j^{th} monomial in $v_k(X)$
    $row_{\hat{w}} = index_{\hat{w}}(base_G(:,i) + base_v(:,j))$
    $rows_{\hat{M}} = (i-1) * N + 1 : i * N$
    $\hat{M}(rows_{\hat{M}}, row_{\hat{w}}) = A(:,j)$
  end
end

1.7 Examples

$P^1 \times ... \times P^1 \times P^m \times P^1$ $\quad$ $k-1$ copies of P^1, 1 copy of P^m.

Get $d \times d$ matrix $M_0 + x_k M_1$.

$(sumX_1)^{r_2}(sumX_2)^{r_3}...(sumX_{k-1})^{r_k}(sumX_k)$

$= (1+x_1)^{m+k-2}(1+x_2)^{m+k-3}...(1+x_{k-1,1}+...+x_{k-1,m})^1(1+x_k)$

So $M(x_k)$ has two monomials, 1 and x_k, while
w has $d = (m+(k-1))(m+(k-2))...(m+1) = (m+(k-1))!/m!$ monomials.

In particular, $P^1 \times P^m \times P^1$

$(sumX_1)^{r_2}(sumX_{k-1})^{r_3}(sumX_3)$

$= (1+x_1)^{m+1}(1+x_{2,1}+...+x_{2,m})^1(1+x_3)$

Let $[1, x_1] \in P^1$,

$$\underline{v} = \begin{bmatrix} 1 \\ x_{2,1} \\ x_{2,2} \\ . \\ . \\ . \\ x_{2,m} \end{bmatrix} \in P^m \tag{64}$$

and $[1, x_3] \in P^1$.

The $2*(m+1)*2$ monomials form the vector $\underline{w}$:

$$\underline{w} = \begin{bmatrix} \underline{v} \\ x_1\underline{v} \\ x_3\underline{v} \\ x_1x_3\underline{v} \end{bmatrix} \tag{65}$$

The $2+m$ equations in $2*(m+1)*2$ monomials are in the form:

$$0 = [A_{00}, A_{01}, A_{10}, A_{11}]\underline{w} \tag{66}$$

Dilation is done by multiplying the above equations by all monomials in $(1+x_1)^{m+1}$, giving:

$$0 = \begin{bmatrix} A_{00} & A_{01} & 0 & . & . & 0 & A_{10} & A_{11} & 0 & . & . & 0 \\ 0 & A_{00} & A_{01} & 0 & . & . & 0 & A_{10} & A_{11} & 0 & . & 0 \\ . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & 0 & A_{00} & A_{01} & 0 & . & . & 0 & A_{10} & A_{11} & 0 \\ 0 & . & . & 0 & A_{00} & A_{01} & 0 & . & . & 0 & A_{10} & A_{11} \end{bmatrix} \begin{bmatrix} \underline{v} \\ x_1\underline{v} \\ {x_1}^2\underline{v} \\ . \\ . \\ . \\ {x_1}^{m+1}\underline{v} \\ x_3\underline{v} \\ x_3x_1\underline{v} \\ x_3{x_1}^2\underline{v} \\ . \\ . \\ . \\ x_3{x_1}^{m+1}\underline{v} \end{bmatrix} \tag{67}$$

This equation can be written in the form

$$0 = [M_0 + x_3 M_1]vec \tag{68}$$

where M_0 and M_1 are each $(m+2)(m+1) \times (m+2)(m+1)$. This gives a square eigenproblem with $(m+2)(m+1)$ eigenvalues.

Chapter 2

More Equations than Unknowns

2.1 Curves on Segre Spaces

Let σ be the Segre map from $(P^1)^n \times P^m$ to P^N.

$$\sigma : (P^1)^n \times P^m \longrightarrow \Sigma \subset P^N \tag{1}$$

Let $Y \subset P^N$ be a linear subspace of P^N.
Let C be a degree d curve given by

$$C = Y \cap \Sigma \tag{2}$$

where the intersection would be empty if Y were in general position, so the intersection of Y and Σ is not transverse.

Let σ' be the Segre map from $(P^1)^{n'} \times P^m$ to $P^{N'}$. Let $Y' \subset P^{N'}$ be a linear subspace of $P^{N'}$.

QUESTION: What is the smallest n' such that

$$C \cup \{points\} = Y' \cap \Sigma' \tag{3}$$

where some extraneouse points have been introduced in the elimination of some variables.

ANSWER(?): $n' =$ smallest l s.t. $d \leq d' = (l+m)!/m!$

SOLVE: Use dilation to form a $d' \times d'$ parameterized eigenproblem. Use deflation to eliminate the extraneous points, leaving a $d \times d$ parameterized eigenproblem.

2.2 Example: $7R$ mechanism

For each fixed value of z_7, we get $30 + m$ equations on $P^1 \times P^1 \times P^1 \times P^1 \times P^1 \times P^1$, with 16 solutions. If only 6 equations are kept, get $6! = 720$ solutions.

If 30 or more equations are kept, a linear algebra procedure can be used to eliminate some variables and get a smaller problem with only the 16 legitimate solutions. However, if this elimination procedure is taken too far, it ends up as 4 equations on $P^1 \times P^1 \times P^1 \times P^1$ with $4! = 24$ solutions. Eight of these 24 solutions are at $\pm i$ which are extraneous. The remaining 16 solutions are legitimate.

To see how the extra equations eliminate the extraneous solutions, we can look at the Smith form of the affine matrices. The $30 + m$ equations are of the form:

$$0 = [M_{7,0} + z_7 M_{7,1}] \begin{bmatrix} I_{32} \\ z_6 I_{32} \end{bmatrix} \begin{bmatrix} I_{16} \\ z_5 I_{16} \end{bmatrix} \begin{bmatrix} I_8 \\ z_4 I_8 \end{bmatrix} v_8(z_3, z_2, z_1) \tag{4}$$

For each fixed value of z_7, let

$$M_6(z_6) = [M_{6,0} + z_6 M_{6,1}] = [M_{7,0} + z_7 M_{7,1}] \begin{bmatrix} I_{32} \\ z_6 I_{32} \end{bmatrix} \tag{5}$$

The matrix $M_6(z_6)$ is size $(30+m) \times 32$. It is rank 30 for all z_6 (unless $m = 0$, then rank drops for discrete values of z_6). Its Smith form decomposition is:

$$M_6(z_6) = [G_{6_L}(z_6), G_{6_R}] \begin{bmatrix} K_{6_{top}} \\ K_{6_{bot}}(z_6) \end{bmatrix} \tag{6}$$

where the size $(30 + m) \times 30$ matrix $G_6(z_6) = [G_{6_L}(z_6), G_{6_R}]$ is rank 30 for all z_6 (unless m=0, then rank drops for discrete values of z_6). The matrix $K_6(z_6)$ is size 30×32. The top 16 rows of $K_6(z_6)$ are constant, denoted by the 16×32 matrix $K_{6_{top}}$.

Since $G_6(z_6)$ is full rank for all z_6,

$$kernel[M_6(z_6)] = kernel[K_6(z_6)] \tag{7}$$

In this case we also get (WHY?):

$$kernel[K_6(z_6)] = kernel[K_{6_{top}}] \tag{8}$$

so we get a new problem, with z_6 eliminated, and only 16 equations:

$$0 = K_{6_{top}} \begin{bmatrix} I_{16} \\ z_5 I_{16} \end{bmatrix} \begin{bmatrix} I_8 \\ z_4 I_8 \end{bmatrix} v_8(z_3, z_2, z_1) \tag{9}$$

If we repeat the above procedure again, eliminating z_5, we introduce some extraneous roots because this time the kernel of the new $K(z)$ matrix is not the same as the kernel of the constant part of $K(z)$.

Let

$$M_5(z_5) = K_{6_{top}} \begin{bmatrix} I_{16} \\ z_5 I_{16} \end{bmatrix} \tag{10}$$

The 16×16 matrix $M_5(z_5)$ is rank 12 for all (generic?) values of z_5, and can be factored as:

$$M_5(z_5) = [G_{5_L}(z_5), G_{5_R}] \begin{bmatrix} K_{5_{top}} \\ K_{5_{bot}}(z_5) \end{bmatrix} \tag{11}$$

where the size 16×12 matrix $G_5(z_5) = [G_{5_L}(z_5), G_{5_R}]$ is rank 12 for all (generic?) values of z_5. The matrix $K_5(z_5)$ is size 12×16. The top 4 rows of $K_5(z_5)$ are constant, denoted by the 4×16 matrix $K_{5_{top}}$.

The commutative diagram below shows what happens if we keep only the constant part of $K(z)$ at each step. The polynomial kernels, $X(z)$, of $M(z)$ are represented as coefficient matrices, $X(b_j) = M(b_j)^{\perp}$, times polynomials $p_j(z)$ using Lagrange interpolation. Lagrange interpolation results in $X(b_j)$ being the same rank for each j, while the standard polynomial basis, $X(z) = \sum_{i=0}^{i_{max}} X_i z^i$, gives coefficient matrices that can have lower rank for X_0 and/or $X_{i_{max}}$. Lagrange interpolation also allows us to compute only numerical kernels of $M(b_j)$ rather than using Wedderburn theory to compute $X(z) = M(z)^{\perp}$.

After the following diagram stabilizes, the final $M(z)$ has as many equations as unknowns. All original roots survive, but some "extraneous" roots may be obtained. When the final $M(z)$ has more columns than rows, we can use dilation to produce a square eigenproblem. The "extranous" roots are either at $\pm i$ when $z = tan(\theta/2)$ or at $0, \inf$ when $z = e^(i\theta)$. The possible reason for the "extraneous" roots is:

The discarded non-constant part of $K(z)$ would have removed those roots. Recall that the real part of the quaternion equations were dropped, because they were not multi-affine. Since we know where the extraneouse roots are, we can remove them using deflation, before an eigenproblem is done. When Wedderburn gave $X(z)$ with low degree, it was in effect romoving roots at infinity, similar to the way deflation works.

The second commutative diagram uses Wedderburn theory to explicitly construct $X(z) = M(z)^{\perp}$. The degree of the $X(z)$ polynomial can be less than its generic value if the equations $0 = M(z)v$ has "extraneous" roots at infinity or zero. Roots at infinity or zero can cause $X_i = 0$ for the first few or last few values of i. In either case, replaceing $X(z)$ with the lower degree polynomial obtained by dropping the zero valued coefficients, gives a smaller system of equations which no longer contain the "extraneous" roots at infinity or zero.

The advantage of using the Lagrange interpolation instead of the Wedderburn theory, is that the degree of the $X(z)$ can be "fractonal", ie the leading coefficient matrix could be low rank. This causes problems when computing $M_{n-1}(z_{n-1})$ using Wedderburn. However, Lagrange interpolation has no such problems.

When the final (or original) $M(z)$ is square (eg 3P-3R structure), then only some roots are legitimate, while many roots are "extranous". Some of these are the $\pm i$ or $0, \inf$ roots known to satisfy the multi-affine equations, while others do not have eigenvectors that satisfy the required Segre structure. For the 3P-3R structure, this can be avoided by using a smaller system of equations associated with only the rotation equations (no translation equations).

First commutative diagram for the 6R structure.

$$
\begin{array}{ccccccccccc}
 & & & & [X_4(b_1), X_4(b_0)] & & & & & & \\
0 & \rightarrow & & R^8 & \longrightarrow & R^8 & \rightarrow & 0 & & & \\
 & & \uparrow \begin{bmatrix} p_1(z_4)I_4 \\ p_0(z_4)I_4 \end{bmatrix} & & & \uparrow I_8 & & & & & \\
 & & & & X_4(z_4) & & M_4(z_4) & & & & \\
0 & \rightarrow & & R^4 & \longrightarrow & R^8 & \longrightarrow & R^4 & \rightarrow & 0 & \\
 & & & & & \downarrow \begin{bmatrix} I_8 \\ z_4 I_8 \end{bmatrix} & & \downarrow I_4 & & & \\
 & & & & [X_5(b_2), ..., X_5(b_0)] & & [M_{40}, M_{41}] & & & & \\
0 & \rightarrow & & R^{12} & \longrightarrow & R^{16} & \longrightarrow & R^4 & \rightarrow & 0 & \\
 & & \uparrow \begin{bmatrix} p_2(z_5)I_4 \\ p_1(z_5)I_4 \\ p_0(z_5)I_4 \end{bmatrix} & & & \uparrow I_{16} & & & & & \\
 & & & & \begin{bmatrix} X_{5_{top}}(z_5) \\ X_{5_{bot}}(z_5) \end{bmatrix} & & M_5(z_5) & & & & \\
0 & \rightarrow & & R^4 & \longrightarrow & R^{16} & \longrightarrow & R^{16} & \rightarrow & R^4 & \rightarrow 0 \\
 & & & & & \downarrow \begin{bmatrix} I_{16} \\ z_5 I_{16} \end{bmatrix} & & \downarrow I_{16} & & & \\
 & & & & [X_6(b_7), ..., X_6(b_0)] & & [M_{50}, M_{51}] & & & & \\
0 & \rightarrow & & R^{16} & \longrightarrow & R^{32} & \longrightarrow & R^{16} & \rightarrow & 0 & \\
 & & \uparrow \begin{bmatrix} p_7(z_6)I_2 \\ \vdots \\ p_0(z_6)I_2 \end{bmatrix} & & & \uparrow I_{32} & & & & & \\
 & & & & X_6(z_6) & & M_6(z_6) & & & & \\
0 & \rightarrow & & R^2 & \longrightarrow & R^{32} & \longrightarrow & R^{30+m} & \rightarrow & R^m & \rightarrow 0 \\
 & & & & & & & (12) & & &
\end{array}
$$

Alternatively, the 8×8 quadratic matrix $M_4(z_5) = [-X_{5_{top}}(z_5), X_{5_{bot}}(z_5)]$ gives an eigenproblem with 16 roots.

Second commutative diagram for the 6R structure.

$$
\begin{array}{ccccccccccc}
 & & & & \begin{bmatrix} X_{5_{top}}(z_5) \\ X_{5_{bot}}(z_5) \end{bmatrix} & & & M_5(z_5) & & & \\
0 & \rightarrow & & R^4 & \longrightarrow & & R^{16} & \longrightarrow & R^{16} & \rightarrow R^4 & \rightarrow 0 \\
 & & \downarrow & \begin{bmatrix} z_4 I_4 \\ I_4 \end{bmatrix} & & & \downarrow [-z_4 I_8, I_8] & & & & \\
 & & & & M_4(z_5) & & & & & & \\
0 & \rightarrow & & R^8 & \longrightarrow & & R^8 & \rightarrow & 0 & &
\end{array}
\tag{13}
$$

For other problems, such as the 4R-2P structure, we can first compute $M_4(z_4)$, then apply the above procedure twice to obtain a 2×2 degree 4 matrix with 8 good eigenvalues. Generic data would result in a 2×2 degree 6 matrix with 12 eigenvalues. If we kept the zero coefficients in the 2×2 degree 6 matrix, we get 8 "good" eigenvalues, and 4 at zero or infinity.

2.3 Incomplete Intersections (for eigenproblems)

After a generic linear change of coordinates, any 0-dimensional ideal in n variables, containing d non-repeated points, can be represented by n polynomial equations (a complete intersection). The roots of this set of n polynomials will be the original d points (plus additional points at infinity in the homogeneous case). However, if some structure is imposed on the type of polynomial equations used to represent the ideal, e.g. multi-affine polynomials, then it may be necessary to use more equations than unknowns (an incomplete intersection).

Given any system of polynomial equations of any degree, it can be converted to a system of multi-affine equations by replacing powers of variables with new variables. This generates new variables as well as new relationships amongst the variables. If the ideal is 0-dimensional with d non-repeated point solutions, then after a generic linear change of coordinates, the reduced Gröbner basis, $G = \{g_1, g_1, ..., g_n\}$, of the ideal is of the form [Gianni and Mora 1989]:

$$\begin{bmatrix} g_1 \\ g_2 \\ \cdot \\ \cdot \\ \cdot \\ g_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-1} \\ x_n^d \end{bmatrix} - B \begin{bmatrix} x_n^0 \\ x_n^1 \\ \cdot \\ \cdot \\ \cdot \\ x_n^{d-1} \end{bmatrix} \tag{14}$$

where the matrix B is unique.

Since d can be any integer, it need not be of the form obtained for generic multi-affine systems, in particular it need not even be factorable. When a system of polynomials of any degree is put into multi-affine form, coefficients on some monomials may be missing. This may result in some roots being at infinity or more equations than unknowns (incomplete intersection). Some systems of polynomials for 0-dimensional ideals may also start out as incomplete intersections. When there are more equations than unknowns, then the procedure described in previous sections multiplies them all by new monomials, resulting in an "eigenproblem" with more equations than unknowns. These extra equations eliminate some of the eigenvalues and eigenvectors.

An $m \times n$ eigenproblem with d solutions has d eigenvalues and d eigenvectors (assume all Jordon blocks are size 1?). Let C and D be the $d \times d$ diagonal matrices whose ratio give the d generalized eigenvalues, and let V_0 be the $n \times d$ matrix whose i^{th} column is the i^{th} eigenvector. The eigen problem can be written as:

$$0 = [A_0, B_0] \begin{bmatrix} V_0 C \\ -V_0 D \end{bmatrix} \tag{15}$$

where A_0 and B_0 are each $m \times n$ matrices with $m > n > d$.

We want to compute V_0, C, and D.

If we numerically compute the kernel of the $[A_0, B_0]$ matrix, we get

$$\begin{bmatrix} B_1 \\ -A_1 \end{bmatrix} = kernel[A_0, B_0] \tag{16}$$

then

$$\begin{bmatrix} V_0 C \\ -V_0 D \end{bmatrix} = \begin{bmatrix} B_1 \\ -A_1 \end{bmatrix} V_1 \tag{17}$$

where V_1 is an unknown $d_1 \times d_1$ matrix,

Since the diagonal C and D matrices commute, right multiplication of the top half of the above equation by D gives the negative of right multiplication of the bottom half of the above equation by C. This gives a new $n \times d_1$ eigenproblem:

$$0 = [A_1, B_1] \begin{bmatrix} V_1 C \\ -V_1 D \end{bmatrix} \tag{18}$$

Repeating this process gives:

$$\begin{bmatrix} B_2 \\ -A_2 \end{bmatrix} = kernel[A_1, B_1] \tag{19}$$

then

$$\begin{bmatrix} V_1 C \\ -V_1 D \end{bmatrix} = \begin{bmatrix} B_2 \\ -A_2 \end{bmatrix} V_2 \tag{20}$$

where V_2 is an unknown $d_2 \times d_2$ matrix,

¿From this, we get a new $d_1 \times d_2$ eigenproblem:

$$0 = [A_2, B_2] \begin{bmatrix} V_2 C \\ -V_2 D \end{bmatrix} \tag{21}$$

This procedure is repeated until the integer sequence d_i converges to d. If the $[A_0, B_0]$ matrix starts out square, then numerical experiments indicate that the integer sequence converges for $i \geq i_J$ where i_J is the size of the largest Jordan block (when $0 < d < \infty$).

The eigenvectors of the original system can be recovered using either:

$$V_0 D^i = A_1 V_1 D^{i-1} = ... = A_1 A_2 ... A_i V_i \tag{22}$$

or

$$V_0 C^i = B_1 V_1 C^{i-1} = ... = B_1 B_2 ... B_i V_i \tag{23}$$

The diagonal entries of the C and D matrices are never both zero at the same location on the diagonal. Therefore any column of the V_0 matrix can be solved for using one or the other of the above two equations. If $D_{j,j} \neq 0$, then

$$col_j[V_0] = [A_1 A_2 ... A_i] col_j[V_i] / {D_{j,j}}^i \tag{24}$$

If $C_{j,j} \neq 0$, then

$$col_j[V_0] = [B_1 B_2 ... B_i] col_j[V_i] / {C_{j,j}}^i \tag{25}$$

Actually, the division by the diagonal entries of C or D can be eliminated, since eigenvectors are typically scaled to unit length anyway.

Note that if we multiply the equation:

$$V_0 D^i = A_1 A_2 ... A_i V_i \tag{26}$$

on the right by C^i we get the same thing as multiplying the equation

$$V_0 C^i = B_1 B_2 ... B_i V_i \tag{27}$$

on the right by D^i. Therefore we get:

$$0 = [A_1 A_2 ... A_i, \;\; B_1 B_2 ... B_i] \begin{bmatrix} V_i C^i \\ -V_i D^i \end{bmatrix} \tag{28}$$

CONVERGENCE OF THE ITERATION

THEOREM: If the $m \times n$ eigenproblem $[A_0 + \lambda B_0]v = 0$ has $0 < d < \infty$ solutions, then the iteration defined by:

$$0 = [A_i, B_i] \begin{bmatrix} B_{i+1} \\ -A_{i+1} \end{bmatrix} \tag{29}$$

converges in $i_{max} + 1$ iterations to a square $d \times d$ eigenproblem, where i_{max} is bounded by:

$$ceiling \left[\frac{n-d}{m-n} \right] \leq i_{max} \leq n - d \tag{30}$$

See Gantmacher Normal Form to get an exact equation for i_{max} [Thomps,1970].

PROOF

Let $d_{-1} = m$, $d_0 = n$, and for $i \geq 0$, let d_{i+1} be the dimension of the kernel of M_i. The matrix M_i has $2d_i$ columns, so

$$d_{i+1} = 2d_i - rank(M_i) \geq 0 \tag{31}$$

Note that

$$rank([A_0 + \lambda B_0]) = rank\left([A_0, B_0]\begin{bmatrix} I \\ \lambda I \end{bmatrix}\right) \leq rank([A_0, B_0]) \tag{32}$$

Assume $rank(M_0) \geq d_0$ so we get at most a finite number of solutions to the original eigenproblem. We must have $rank(M_i) \geq d_i$, else we would get an infinite number of solutions to the i^{th} eigenproblem, $0 = (A_i + \lambda B_i)v(\lambda)$ for all values of λ. This would map down to an infinite number of solutions to the previous eigenproblem, $0 = (A_{i-1} + \lambda B_{i-1})B_i v(\lambda)$ for all values of λ. Eventually, this would map down to an infinite number of solutions to the original eigenproblem, $0 = (A_0 + \lambda B_0)B_1 B_2 ... B_i v(\lambda)$ for all values of λ.

Because $rank(M_i) \geq d_i$, we get:

$$d_{i+1} = 2d_i - rank(M_i) \leq d_i \tag{33}$$

so the sequence of nonnegative integers, d_i, is monotonically decreasing.

LEMMA:

The integer sequence d_i converges to d, the number of solutions to the original eigenproblem.

PROOF OF LEMMA:

If $d_{i_{max}+1} = d_{i_{max}}$, then we get a square eigenproblem at that step. If the eigenvalues are not repeated, then there is a full set of $d_{i_{max}}$ eigenvectors, so the kernel of the $2d_{i_{max}} \times d_{i_{max}}$ matrix $[A_{i_{max}+1}, B_{i_{max}+1}]$ will be $d_{i_{max}}$ dimensional, so $d_i = d_{i_{max}}$ for all $i \geq i_{max}$.

end of PROOF OF LEMMA:

If d_i decreases at the slowest possible rate, i.e. $d_{i+1} = d_i - 1$ until $i = i_{max}$, then $i_{max} = n - d$. In general

$$i_{max} \leq n - d \tag{34}$$

Since $rank(M_i) \leq NumRows(M_i) = d_{i-1}$, the equation

$$d_{i+1} = 2d_i - rank(M_i) \tag{35}$$

gives:

$$d_{i+1} \geq 2d_i - d_{i-1} \tag{36}$$

Combining this with the initial conditions, $d_{-1} = m$, $d_0 = n$, gives:

$$d_i \geq n - i(m-n) \tag{37}$$

This decreasing integer sequence converges for $i \geq i_{max}$ where i_{max} is obtained by setting $d_i = d$ in the above formula.

$$i_{max} \geq ceiling\left[\frac{d_0 - d}{m - d_0}\right] = ceiling\left[\frac{n-d}{m-n}\right] \tag{38}$$

Combining the upper and lower bounds gives:

$$ceiling\left[\frac{n-d}{m-n}\right] \leq i_{max} \leq n - d \tag{39}$$

To obtain a square eigenproblem, one more iteration is needed, since $A_i + \lambda B_i$ is size $d_{i-1} \times d_i$.

$$[A_{i_{max}+1} + \lambda B_{i_{max}+1}] \in R^{d \times d} \tag{40}$$

When the sequence converges to some integer d, that must be the number of solutions to the original eigenproblem because we get square $d \times d$ matrices A_i and B_i, so the following eigenproblem $0 = [A_i - \lambda B_i]v$ has d eigenvalues. The eigenvector matrix gets mapped back into the original large space by recursively applying $Cv_{i-1} = B_i v_i$ or $Dv_{i-1} = A_i v_i$.

Ok even if there are repeated eigenvalues, and not a full set of d eigenvectors.

■

COMMUTING DIAGRAM:

Let $M_i = [A_i, B_i]$. Let $L_{i+1} = kernel(M_i)$ so that $M_i L_{i+1} = 0$. Define $-A_{i+1}$ and B_{i+1} as the bottom and top halves of the L_{i+1} matrix.

$$\begin{bmatrix} B_{i+1} \\ -A_{i+1} \end{bmatrix} = L_{i+1} \tag{41}$$

so $A_i B_{i+1} = B_i A_{i+1}$.

Let

$$\hat{A}_i = \begin{bmatrix} A_i & 0 \\ 0 & A_i \end{bmatrix} \qquad and \qquad \hat{B}_i = \begin{bmatrix} B_i & 0 \\ 0 & B_i \end{bmatrix} \tag{42}$$

The i^{th} row in the following diagram gives the i^{th} eigenproblem. The vertical maps show how the eigenvectors, kernels, and ranges are mapped (assuming that $C = I$). The following diagram commutes since $A_i B_{i+1} = B_i A_{i+1}$.

$$\begin{array}{ccccccccc}
0 & \longrightarrow & R^{d_{imax+2}} & \longrightarrow & R^{2d_{imax+1}} & \longrightarrow & R^{d_{imax}} & \longrightarrow & 0 \\
 & & \vdots & & \vdots & & \vdots & & \\
 & & & L_4 & & M_3 & & & \\
0 & \longrightarrow & R^{d_4} & \longrightarrow & R^{2d_3} & \longrightarrow & R^{d_2} & \longrightarrow & 0 \\
 & & \downarrow B_4 & & \downarrow \hat{B}_3 & & \downarrow B_2 & & \\
 & & & L_3 & & M_2 & & & \\
0 & \longrightarrow & R^{d_3} & \longrightarrow & R^{2d_2} & \longrightarrow & R^{d_1} & \longrightarrow & 0 \\
 & & \downarrow B_3 & & \downarrow \hat{B}_2 & & \downarrow B_1 & & \\
 & & & L_2 & & M_1 & & & \\
0 & \longrightarrow & R^{d_2} & \longrightarrow & R^{2d_1} & \longrightarrow & R^{d_0} & \longrightarrow & 0 \\
 & & \downarrow B_2 & & \downarrow \hat{B}_1 & & \downarrow B_0 & & \\
 & & & L_1 & & M_0 & & & \\
0 & \longrightarrow & R^{d_1} & \longrightarrow & R^{2d_0} & \longrightarrow & R^{m} & \longrightarrow & 0
\end{array} \tag{43}$$

Note that if the assumption $C = I$ is replaced with the assumption $D = I$, then a similar diagram can be made by replacing the vertical B_i and $\hat{B}_i$ maps with vertical A_i and $\hat{A}_i$ maps.

If $d_i = d$ for $i \geq i_J$, then we get the following $d \times d$ eigenproblem with

eigenvalues = diag(D).

$$0 = [A_{i_J+1}, B_{i_J+1}] \begin{bmatrix} V \\ -VD \end{bmatrix} \tag{44}$$

After the previous diagram converges, we get a pull-back from the following commuting diagram:

$$\begin{array}{ccc} & [A_{i_J+1}, B_{i_J+1}] & \\ R^{2d_{i_J+1}} & \longrightarrow & R^{d_{i_J}} \\ \downarrow \hat{B_{i_J+1}} & & \downarrow B_{i_J} \\ & [A_{i_J}, B_{i_J}] & \\ R^{2d_{i_J}} & \longrightarrow & R^{d_{i_J-1}} \end{array} \tag{45}$$

$$PullBack = ker\left[(B_{i_J}, -[A_{i_J}, B_{i_J}]) : R^{d_{i_J}} \oplus R^{2d_{i_J}} \to R^{i_J-1}\right] \tag{46}$$

$$= \begin{bmatrix} I_{d_{i_J}} & 0 \\ 0 & B_{i_J+1} \\ I_{d_{i_J}} & -A_{i_J+1} \end{bmatrix} \tag{47}$$

$$dim(PullBack) = (d_{i_J} + 2d_{i_J}) - rank([B_{i_J}, -(A_{i_J}, B_{i_J})]) \tag{48}$$

$$= (d_{i_J} + 2d_{i_J}) - rank([A_{i_J}, B_{i_J}]) = (d_{i_J} + 2d_{i_J}) - (2d_{i_J} - d_{i_J+1}) = 2d \tag{49}$$

2.4 Incomplete Intersections (general polynomials)

Any set of real polynomial equations in $x \in C^n$ can be written in matrix form as:

$$0 = \begin{bmatrix} M_0 \\ Q_0(x) \end{bmatrix} v_0(x) \tag{50}$$

Where $M_0 \in R^{m \times p}$ is a matrix of coefficients, $v_0(x) \in Z^p[x]$ is a vector of monomials, and $Q_0(x) \in Z^{q,p}[x]$ represents the relations among the monomials $v_0(x)$.

$$Q_0(x) = \sum_{i=1}^{k} Q_{0,i} w_i(x) \tag{51}$$

where the $Q_{0,i}$ matrices are filled with integers, and the $w_i(x)$ are monomials. For example, in the generalized eigenproblem, $0 = [A - \lambda B]eigvec$, we get $M_0 = [A, B]$, $v_0(x) = \begin{bmatrix} -eigvec \\ \lambda eigvec \end{bmatrix}$, and $Q_0(x) = [\lambda I, I]$.

In the general case, $v_0(x)$ can be expressed as:

$$v_0(x) = M_0{}^{\perp} c_0(x) \tag{52}$$

for any polynomial vector $c_0(x)$ that satisfies

$$Q_0(x) M_0{}^{\perp} c_0(x) = 0 \tag{53}$$

This can be expanded as:

$$0 = Q_0(x) M_0{}^{\perp} c_0(x) = \sum_{i=1}^{k} Q_{0,i} w_i(x) M_0{}^{\perp} c_0(x) \tag{54}$$

$$= [Q_{0,1} M_0{}^{\perp}, Q_{0,2} M_0{}^{\perp}, ..., Q_{0,k} M_0{}^{\perp}] \begin{bmatrix} w_1(x) c_0(x) \\ w_2(x) c_0(x) \\ \vdots \\ w_k(x) c_0(x) \end{bmatrix} \tag{55}$$

Let

$$M_1 = [Q_{0,1} M_0{}^{\perp}, Q_{0,2} M_0{}^{\perp}, ..., Q_{0,k} M_0{}^{\perp}] \tag{56}$$

and

$$v_1(x) = \begin{bmatrix} w_1(x)c_0(x) \\ w_2(x)c_0(x) \\ \vdots \\ w_k(x)c_0(x) \end{bmatrix} \tag{57}$$

We then have a new matrix equation

$$0 = M_1 v_1(x) \tag{58}$$

The new monomial vector, $v_1(x)$ satisfies the relations:

$$0 = Q_1(x)v_1(x) \tag{59}$$

where

$$Q_1(x) = \begin{bmatrix} w_2(x)I & -w_1(x)I & 0 & 0 & ... & 0 \\ w_3(x)I & 0 & -w_1(x)I & 0 & ... & 0 \\ \vdots & 0 & 0 & \ddots & & 0 \\ w_k(x)I & 0 & 0 & 0 & ... & -w_1(x)I \end{bmatrix} \tag{60}$$

This gives the new system:

$$0 = \begin{bmatrix} M_1 \\ Q_1(x) \end{bmatrix} v_1(x) \tag{61}$$

If we can solve this new system for $v_1(x)$, then we can reconstruct $v_0(x)$ as follows:

$$v_0(x) = M_0{}^{\perp} c_0(x) = M_0{}^{\perp} P_i v_1(x)/w_i(x) \tag{62}$$

where

$$P_i = [0, 0, ..., 0, I, 0, ..., 0] \tag{63}$$

such that $P_i v_0(x) = w_i(x)c_0(x)$.

If we cannot solve the new system for $v_1(x)$, then repeat the above process until $Q_j(x)M_j$ converges to a fixed dimension and degree.

Given the set of d solutions: $X = [x(1), x(2), ..., x(d)]$

Let $V_0 = [v_0(x(1)), v_0(x(2)), ..., v_0(x(d))]$. Then the above matrix equation can be written as:

$$0 = M_0 V_0 \tag{64}$$

We can express V_0 in terms of the numerical kernel, $M_0{}^{\perp}$, of the M_0 matrix.

$$V_0 = M_0{}^{\perp} C_0 \tag{65}$$

where

$$C_0 = [c_0(x(1)), c_0(x(2)), ..., c_0(x(d))] \tag{66}$$

Special Case

For the eigen-problem,

$$v_0(x) = \begin{bmatrix} I_n \\ x_n I_n \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \tag{67}$$

This gives $Q_0(x) = [x_n I_n, -I_n]$.

In a slightly more general case

$$v_0(x) = \begin{bmatrix} I_n \\ G_0(x) \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \tag{68}$$

where $G_0(x)$ is an $m \times n$ matrix. In this case, $Q_0(x) = [G_0(x), -I_m]$. This leads to the following commutative diagram:

$$
\begin{array}{ccccccccc}
 & & & M_0{}^{\perp} & & M_0 & & & \\
0 & \longrightarrow & R^m & \longrightarrow & R^{m+n} & \longrightarrow & R^n & \longrightarrow & 0 \\
 & & \downarrow \begin{bmatrix} 0 \\ I_m \end{bmatrix} & & \updownarrow I_{m+n} & & \uparrow [I_n, 0] & & \\
 & & & F(x) & & H(x) & & & \\
 & & R^{m+n} & \longrightarrow & R^{m+n} & \longrightarrow & R^{m+n} & & \\
 & & \uparrow \begin{bmatrix} I_n \\ 0 \end{bmatrix} & & \updownarrow I_{m+n} & & \downarrow [0, I_m] & & \\
 & & & \begin{bmatrix} I_n \\ G_0(x) \end{bmatrix} & & [G_0(x), -I_m] & & & \\
0 & \longrightarrow & R^n & \longrightarrow & R^{m+n} & \longrightarrow & R^m & \longrightarrow & 0
\end{array}
\tag{69}
$$

where

$$
F(x) = \left[\begin{bmatrix} I_n \\ G_0(x) \end{bmatrix} \ , \ M_0{}^{\perp} \right] \tag{70}
$$

$$
H(x) = \begin{bmatrix} M_0 \\ G_0(x) \quad -I_m \end{bmatrix} \tag{71}
$$

and

$$
H(x)F(x) = \begin{bmatrix} M_0 \begin{bmatrix} I_n \\ G_0(x) \end{bmatrix} & 0 \\ 0 & [G_0(x), I_m] M_0{}^{\perp} \end{bmatrix} \tag{72}
$$

The upper left block is the original problem. The lower right block is the new problem. This applies to the eigenproblem, where $F(x) = F(x_n)$ $H(x) = H(x_n)$ are both square matrices that are full rank except when x_n is an eigenvalue.

Let $Q_0(x) = [\lambda I_n, -\mu I_n]$ and show that $det[F(x_n)] = 0$ iff $det[H(x_n)] = 0$. This can then be used to prove that the Incomplete Intersection algorithm ends up with a new eigenproblem whose roots are identical with those of the original system.

2.5 References

[Wedderburn]
William Fulton, "Intersection Theory," Springer, 1984.

Blaise Morton and Michael Elgersma, "A New Computational Algorithm for 7R Spatial Mechanisms," Mechanism and Machine Theory, vol 31 no 1, pp 23-43, 1996.

Daniel R. Grayson and Michael E. Stillman "Macaulay2," 1996.

Groebner Basis complexity: $[poly(n)]d^n < flops < [poly(n)]d^{n^2}$
Lazard

Groebner Bases
Buchberger
P. Gianni and Teo Mora (1989), "Algebraic Solutions of Systems of Polynomial Equations Using Groebner Bases," Proceedings of AAECC 5, LNCS 356, p 247-257.

Groebner Basis: $d^n < flops < d^{n^2}$
Multivariable Resultants
Bhubabeswar Mishra, "Algorithmic Algebra," Springer-Verlag, 1993.

Deepak Kapur and Yagati N. Lakshman, "Elimination Methods: an Introduction," Chapter 2 in: B. R. Donald, D. Kapur, and J. L. Mundy, "Symbolic and Numerical Computation for Artificial Intelligence," Academic Press, 1992.

Dilation, Macaulay Resultant details and recent extensions.
Lazard(1981), Canny(1988).

U-Resultants (doesn't work if common zeros, even at infinity) (extension to Macaulay Resultant) Canny(1988), Lakshman(1990a), Lakshman and Lazard (1991), Manocha and Canny (1991)

F. S. Macaulay, "The Algebraic Theory of Modular Systems," Cambridge Tracts in Math. and Math. Physics, vol 19, 1916.

Deflation
Dongarra??

Multi-linear Equations (Volume of the Newton Polytope, Chow Variety):
[St] Bernd Sturmfels, "Sparse Elimination Theory," Cortona conference 1991, published in "Computational Algebraic Geometry and Commutative Algebra" ed. David Eisenbud and Lorenzo Robbiano, 1994.

[Can] Leandro Caniglia, "How to Compute the Chow Form of an Unmixed Polynomial Ideal in Single Exponential Time," Applicable Algebra in Engineering, Communication, and Computing. Springer-Verlag 1990.

[GKZ] I.M. Gelfand, M.M. Kapranov, A.V. Zelevinski, "Discriminants, Resultants, and Multidimensional Determinants (Mathematics: Theory and Applications)," Birkhauser, 1994.

Homotopy, Multi-homogeneous:
A. Morgan and A. Sommese, "A homotopy for solving general polynomial system that respects m-homogeneous structures," Applied Mathematics and Computation 24 (1987) 101-113.

Incomplete Intersection
[Thomps,1970] Gerald L. Thompson and Roman L. Weil, "Reducing The Rank of $(A - \lambda B)$," Proc. AMS 26, 4 (Dec. 1970), p 548-554.

[Dell 1971] Alice M. Dell, Roman L. Weil, and Gerald L. Thompson, "Algorithm 405, Roots of Matrix Pencils: The Generalized Eigenvalue Problem [F2]," Communications of the ACM, Feb. 1971, Vol 14, Number 2.

[Gant, 1959] F. R. Gantmacher, "The Theory of Matrices, II," Chelsea Pub. Co., New York, 1959, pp. 35-40.

Acknowledgments

The authors would like to thank Steven G. Pratt for writing efficient "index" and "base" functions used to implement the dilation algorithm. The authors would also like to thank Kathryn Lenz for several helpful discussions on dilation and deflation.