# Cylinders Through Five Points: Computational Algebra and Geometry

**Daniel Lichtblau**

Wolfram Research, Inc.
100 Trade Center Dr.
Champaign, IL 61820
danl@wolfram.com

**Abstract**. We address the following question: Given five points in $\mathbb{R}^3$, determine a right circular cylinder containing those points. We obtain algebraic equations for the axial line and radius parameters and show that these give six solutions in the generic case. An even number (0, 2, 4, or 6) will be real valued and hence correspond to actual cylinders in $\mathbb{R}^3$. We will investigate computational and theoretical matters related to this problem. In particular we will show how exact and numeric Gröbner bases, equation solving, and related symbolic−numeric methods may be used to advantage. We will also discuss some applications.

2000 Mathematics Subject Classification: 14N10, 14Q10, 51N20, 52A15, 65C05, 65H10, 68U20.

Key words and phrases: Computational geometry, enumerative geometry, Gröbner bases, nonlinear systems, symbolic−numeric computation.

## 1. Outline of the Problem and Related Work

Given five points in $\mathbb{R}^3$, we are to determine all right circular cylinders containing those points. We do this by solving equations for the axial line and radius parameters. We will show that generically one obtains six solutions to these equations. Of these an even number are real valued, as the complex valued ones appear in conjugate pairs (an immediate consequence is that there is no "unique" real cylinder through five given points unless it a solution with multicity). Moreover there are open regions in the real configuration space that give each of these possibilities so we learn that none are disallowed.

The basic problem of determining cylinders from five points may be recast in a computational geometry setting: Given five points in $\mathbb{R}^3$, find the smallest positive $r$ and orientation parameters such that the cylinder of radius $2\,r$ with those parameters encloses tangentially the balls of radius $r$ centered at the points.

Here are some questions we will consider. The first three are classical; we address them here to illustrate the utility of symbolic computation in such investigations. The last ones are related to more recent work in computational and integral geometry.

(1) Given the points and corresponding cylinder parameters, how might we display them graphically?

(2) Given the cylinder parameters, how may we obtain its implicit equation as a hypersurface in $\mathbb{R}^3$?

(3) Reversing this, how can one obtain parameters from the implicit form?

(4) Given six or more points, how do we find the coordinates of a (generically unique) cylinder in $\mathbb{R}^3$ that "best" fits those points?

(5) Given five points chosen with random uniform distribution in a cube, what is the expected probability that one lies inside the convex hull of the other four (this is related to the "no real cylinder" case).

(6) How might we rigorously provide, via straightforward computation, the generic number of solutions to the algebraic equations that describe cylinders through five indeterminate points.

In the sequel we frequently use the term "real cylinders" to denote real valued solutions to the cylinder equations that arise from a given configuration of five points. Sometimes we refer to arbitrary solutions as "cylinders" even if they have complex values. The meaning should be clear from context. We refer to configurations as "generic" if they do not have multiple solutions and if all sufficiently small perturbations of the configuration give rise to the same number of solutions. This amounts to the configuration not lying on the discriminant variety [25] but we will not belabor this point. In some places we also use generic to mean that a system is in general position so that the Shape Lemma applies [2]. As we will have occasion to change our underlying set of variables we note that this last notion is dependent on the variables under consideration.

That one obtains six cylinders was previously demonstrated in [5] though by rather different means. Various proofs are also presented in [8] [14] [28] [29]. A related problem, finding cylinders of a given radius through four given points in $\mathbb{R}^3$, is discussed in [16] [20] [30] [33]. A nice survey of computational commutative algebra methods that are applicable to nonlinear problems in computational geometry can be found in [7]. Another good general treatment of theoretical and practical aspects of Gröbner bases in computational geometry is chapter 7 of [19]. A companion paper to this one [28] delves further into enumerative geometry aspects of cylinders through five points.

The remainder of this paper is structured as follows. In section 2 we present several computational sides to the problem. These include finding and counting cylinder solutions. In section 3 we handle various associated computational geometry problems, and basics of point/cylinder visualization. Section 4 delves into the frequencies of real cylinders containing random point sets from a certain distribution. These investigations are again largely computational, though we relate some to a recent result in integral geometry. In section 5 we use simple symbolic computation methods to prove the enumerative geometry result that there are six solutions to a generic set of cylinder equations. Section 6 poses some further questions regarding cylinders through five points, and outlines how one might attack them computationally using aspects of the discriminant variety [25]. Following that is a brief summary. An attempt is made to emphasize the ways in which symbolic, numeric, and hybrid computation methods are useful in these investigations. Computations in the sequel were performed with the version of *Mathematica* [38] under development at the time of this writing. All implementation code is provided in the appendix. While the code used seems generally good for the tasks at hand, the author makes no claim to the effect that other programs might not be as good (or perhaps better). Explanatory detail is provided so that readers inclined to programming, and armed with software capable of symbolic and numeric computation, might reproduce results similar to those shown herein. An incomplete list of programs capable of handling some or all of the computations in the sequel includes CoCoA, Macaulay 2, Magma, Maple, Risa/Asir, and Singular. More specialized software well suited for some of these computations (e.g. Gröbner bases and numeric system solving) would include FGb and PHCpack.

**Acknowledgements**

I am pleased to acknowledge helpful conversations with Dick Bishop, John D'Angelo, Dan Grayson, and Dave Rusin (the last conducted via email; see also [32]). I thank David Cox and Thorsten Theobald for bringing to my attention the earlier work of Bottema and Veldkamp. I thank the Math Department of the University of Illinois for extending their hospitality during the academic year 1998−9 when I was a visitor there, during which time preliminary parts of this work were done. In particular I thank Jack Wetzel for inviting me to speak twice on this topic at the UIUC weekly Geometry Potpourri seminar. I thank an anonymous and very patient referee for carefully reading the draft and making several detailed comments and suggestions that helped to clarify (and in some cases repair) many points in the exposition.

## 2. Computing Cylinders Through Five Points

### Finding cylinder parameters from a set of 5 points

We will assume unless otherwise stated that our points are generic. In particular, no three are collinear, no four are coplanar, cylinder axes do not lie in coordinate planes, and so forth. With these assumptions we avoid computational pitfalls that would arise from parametrizing axial directions using a sphere (this gives rise to two problems: we have one extra variable, and so to eliminate it we would add an equation that normalizes the direction. Moreover we would double the size of our solution set because any direction is equivalent to its negative.) Given these stipulations we proceed as follows.

With our assumptions in place, given a cylinder axis line $L$ in $\mathbb{R}^3$ we may parametrize it as

$$\{y = a\,x + b,\ z = c\,x + d\} \tag{1}$$

For any $r > 0$ there is a unique circular cylinder $C$ of radius $r$ with center axis $L$. Supposing we have five points on that cylinder the following questions now arise. How do we find $L$ and $r$? How do we use them to parametrize $C$ e.g. for purposes of plotting it?

First we discuss why this data will determine finitely many cylinders. Given a point on $C$ we will project orthogonally onto $L$ in order to get an equation involving the parameters we wish to find. We have five parameters to

determine in the setup used above. For each point we denote the length of the orthogonal projection by $perp_j$. It is computed as follows. We take $\tilde{L}$ to be the subspace obtained by translating $L$ to pass through the origin. For each point $p_j$ take $\tilde{p_j}$ to be the correspondingly translated point. We subtract from $\tilde{p_j}$ its projection onto $\tilde{L}$. This difference is the orthogonal complement of the projection and thus its magnitude is $perp_j$. This projection will give us an algebraic equation of the form

$$\left\| perp_j \right\|^2 - r^2 = 0 \tag{2}$$

A concise coordinate−free formulation of this appears in [34], and simple code to compute it is in the appendix. We show here the actual equation in terms of our point coordinates and cylinder parameters. If $p_j$ is given as $(x_j, y_j, z_j)$ then, after clearing denominators, the explicit equation in terms of cylinder parameters $(a, b, c, d, r)$ is:

$$\begin{aligned} &b^2 + b^2\,c^2 - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - r^2 - a^2\,r^2 - c^2\,r^2 + 2\,a\,b\,x_j + 2\,c\,d\,x_j + a^2\,x_j^2 + c^2\,x_j^2 - 2\,b\,y_j - 2\,b\,c^2\,y_j + \\ &\quad 2\,a\,c\,d\,y_j - 2\,a\,x_j\,y_j + y_j^2 + c^2\,y_j^2 + 2\,a\,b\,c\,z_j - 2\,d\,z_j - 2\,a^2\,d\,z_j - 2\,c\,x_j\,z_j - 2\,a\,c\,y_j\,z_j + z_j^2 + a^2\,z_j^2 = \ 0 \end{aligned} \tag{3}$$

For generic choice of points the equations should be algebraically independent, hence the dimension of the solution set would be zero. In more detail, if we take five points with indeterminate coordinates (that is, coordinates expressed as variables) then we obtain a system of five equations of the form $f_j(a, b, c, d, r) = 0$, each arising from (3) with appropriate point coordinates plugged in. From these we want to solve for the cylinder parameters in terms of those coordinates. To show there are finitely many solutions it suffices by the implicit function theorem to show that the Jacobian of the map $(f_1\ f_2,\ f_3,\ f_4,\ f_5)$ has full rank for these generic coordinates. One can do this explicitly by finding the symbolic Jacobian, plugging in random values for the coordinates, and checking that the resulting matrix has full rank. We will instead show a computation in the last section that demonstrates there are generically at most nine solutions. Simple reasoning will further reduce this to eight. We also provide computational proofs that there are in fact but six.

Let us demonstrate how to solve for the cylinder parameters with a specific example. We will take as our parameter values

$$a = \ 3,\ b = 2,\ c = 4,\ \mathrm{d} = -1,\ r = \ \sqrt{21} \tag{4}$$

The locus of points on $C$ is obtained as sums of a vector on $L$ plus a vector of length $r$ perpendicular to $L$. All vectors perpendicular to $L$ are spanned by any independent pair. We can obtain an orthonormal pair $(w_1, w_2)$ in the standard way by finding the null space to the matrix whose one row is the vector along the axial direction, that is, $vec = (1, a, c)$, and then using Gram−Schmidt to orthogonalize that pair. From this we obtain vectors $\left(-4/\sqrt{17}\ ,\ 0,\ 1/\sqrt{17}\right)$ and $\left(-3/\sqrt{442}\ ,\ \sqrt{17/26}\ ,\ -6\sqrt{2/221}\right)$.

We will then select five "random" points on $C$. We do this by selecting five values for an axial vector scale factor $x$ and five values for an angle $\theta$ such that $0 \le \theta \le 2\pi$. Our points will be of the form $v + w$ where

$$\begin{aligned} v &= \ offset + vec\,x \\ w &= r\cos(\theta)\,w_1 + r\sin(\theta)\,w_2 \end{aligned}$$

We now discuss recovery of a set of cylinder parameters from these five points. Given a point on $C$ we want to project orthogonally onto $L$, to get an equation involving the parameters we wish to find. As discussed above we first translate our point by subtracting *offset*. We then project onto the line spanned by *vec*. Subtracting this projection from the translated vector gives us our *perp*. For example, one point on the cylinder in question is approximately `{5.86419, 9.90186, 16.3218}`. The corresponding expression we set to zero is

$$\begin{aligned} &364.45 - 116.133\,\mathrm{a} + 300.791\,\mathrm{a}^2 - 19.8037\,\mathrm{b} + 11.7284\,\mathrm{a}\,\mathrm{b} + \mathrm{b}^2 - 191.429\,\mathrm{c} - \\ &\quad 323.233\,\mathrm{a}\,\mathrm{c} + 32.6437\,\mathrm{a}\,\mathrm{b}\,\mathrm{c} + 132.435\,\mathrm{c}^2 - 19.8037\,\mathrm{b}\,\mathrm{c}^2 + \mathrm{b}^2\,\mathrm{c}^2 - 32.6437\,\mathrm{d} - \\ &\quad 32.6437\,\mathrm{a}^2\,\mathrm{d} + 11.7284\,\mathrm{c}\,\mathrm{d} + 19.8037\,\mathrm{a}\,\mathrm{c}\,\mathrm{d} - 2\,\mathrm{a}\,\mathrm{b}\,\mathrm{c}\,\mathrm{d} + \mathrm{d}^2 + \mathrm{a}^2\,\mathrm{d}^2 - \mathrm{r}^2 - \mathrm{a}^2\,\mathrm{r}^2 - \mathrm{c}^2\,\mathrm{r}^2 \end{aligned}$$

We can use numerical methods to find some roots. This is very sensitive to initial conditions. For example, starting at $(a, b, c, d, r) = (3.2, 2.8, 3.7, -1.6, 3.3)$, which is quite close to the values we began with, will recover those

values. Starting instead at (2.7, 1.8, 3.2, −.7, 3) gives solution parameters $(a, b, c, d, r) =$ (1.91, 7.09, 2.49, 7.02, −4.44) to three decimal places.

Another well known method to find numeric roots is to sum the squares of the polynomials to be satisfied, and then minimize this sum. This too is sensitive to initial guesses. Using initial values of $(a, b, c, d, r) =$ (2.4, 1.8, 2.2, −1.8, 3) recovers the second solution shown above. Using instead (2.4, 1.8, 3.2, −.8, 3) gives a useless result with residual larger than $10^6$. Clearly we need a better approach.

**Solving simultaneously for all roots of the cylinder parameter equations**

An obvious drawback to the methods seen thus far is the need for good initial guesses. We may take advantage of the fact that the equations are all polynomial and instead use a global solver suitable for such systems. We demonstrate below the utility of this approach. In order to have simpler equations for visual purposes will work with a new example comprised of integer coordinates in the range (−10,10). To further simplify matters we will solve for the square of the radius (this will avoid solutions with negative values for $r$ as well as cut in half the number of complex valued solutions). An example problem with pseudorandom coordinates in the indicated range gave rise to the polynomials shown below. The points we chose to lie on the cylinder(s) are:

$$(7,9,8),(8,-4,-10),(-4,1,4),(-9,-9,-10),(-7,-10,-10) \tag{5}$$

The five corresponding polynomials we set to zero are as below.

$$\begin{aligned}
&\big(145 - 126\,a + 113\,a^2 - 18\,b + 14\,a\,b + b^2 - 112\,c - 144\,a\,c + 16\,a\,b\,c + 130\,c^2 - 18\,b\,c^2 + \\
&\quad b^2\,c^2 - 16\,d - 16\,a^2\,d + 14\,c\,d + 18\,a\,c\,d - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - rsqr - a^2\,rsqr - c^2\,rsqr, \\
&\quad 116 + 64\,a + 164\,a^2 + 8\,b + 16\,a\,b + b^2 + 160\,c - 80\,a\,c - 20\,a\,b\,c + 80\,c^2 + 8\,b\,c^2 + b^2\,c^2 + \\
&\quad 20\,d + 20\,a^2\,d + 16\,c\,d - 8\,a\,c\,d - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - rsqr - a^2\,rsqr - c^2\,rsqr, \\
&\quad 17 + 8\,a + 32\,a^2 - 2\,b - 8\,a\,b + b^2 + 32\,c - 8\,a\,c + 8\,a\,b\,c + 17\,c^2 - 2\,b\,c^2 + b^2\,c^2 - \\
&\quad 8\,d - 8\,a^2\,d - 8\,c\,d + 2\,a\,c\,d - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - rsqr - a^2\,rsqr - c^2\,rsqr, \\
&\quad 181 - 162\,a + 181\,a^2 + 18\,b - 18\,a\,b + b^2 - 180\,c - 180\,a\,c - 20\,a\,b\,c + 162\,c^2 + 18\,b\,c^2 + \\
&\quad b^2\,c^2 + 20\,d + 20\,a^2\,d - 18\,c\,d - 18\,a\,c\,d - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - rsqr - a^2\,rsqr - c^2\,rsqr, \\
&\quad 200 - 140\,a + 149\,a^2 + 20\,b - 14\,a\,b + b^2 - 140\,c - 200\,a\,c - 20\,a\,b\,c + 149\,c^2 + 20\,b\,c^2 + \\
&\quad b^2\,c^2 + 20\,d + 20\,a^2\,d - 14\,c\,d - 20\,a\,c\,d - 2\,a\,b\,c\,d + d^2 + a^2\,d^2 - rsqr - a^2\,rsqr - c^2\,rsqr\big)
\end{aligned} \tag{6}$$

We mention that this system is not substantially simpler to solve numerically than the preceding one; its main virtue for our purposes is that it is more concise to print. In contrast to local methods, which, as we saw, may fail to get a particular solution, it turns out to be computationally straightforward to obtain all solutions to this system. We do this in *Mathematica* with the `NSolve` function. It uses a hybrid symbolic−numeric technique to efficiently find all roots. Details of this technology are discussed in [10] [11] [27]. The basic idea is to compute a numeric Gröbner basis and then do an eigendecomposition of a certain matrix formed therefrom. Our solution set is as below.

$$\begin{aligned}
&\{a \to -1.03253 + 0.760393\,i,\ b \to 6.11349 - 3.37419\,i,\ c \to -0.322931 - 1.37768\,i, \\
&\quad d \to -0.295427 + 6.8709\,i,\ rsqr \to 344.25 + 23.8554\,i\},\ \{a \to -1.03253 - 0.760393\,i, \\
&\quad b \to 6.11349 + 3.37419\,i,\ c \to -0.322931 + 1.37768\,i,\ d \to -0.295427 - 6.8709\,i,\ rsqr \to 344.25 - 23.8554\,i\}, \\
&\{a \to 0.151635,\ b \to -1.25748,\ c \to 1.58897,\ d \to -6.45046,\ rsqr \to 83.0554\}, \\
&\{a \to 30.9362,\ b \to 93.172,\ c \to 37.1186,\ d \to 92.7034,\ rsqr \to 198.258\}, \\
&\{a \to 0.613253 - 0.359335\,i,\ b \to -4.49777 - 3.77132\,i,\ c \to 0.102934 + 0.159852\,i, \\
&\quad d \to -1.56979 + 2.23275\,i,\ rsqr \to 57.5606 + 13.7534\,i\},\ \{a \to 0.613253 + 0.359335\,i, \\
&\quad b \to -4.49777 + 3.77132\,i,\ c \to 0.102934 - 0.159852\,i,\ d \to -1.56979 - 2.23275\,i,\ rsqr \to 57.5606 - 13.7534\,i\}
\end{aligned}$$

Actually we can get exact solutions in the same way, albeit at greater (though still quite reasonable) computational cost. This illustrates a sort of cascading hybrid algorithm: one starts with a symbolic−numeric method to handle numeric problems, then modifies it to give exact rather than approximate results.

```
Timing[exactsolns = NSolve[exprs, {a, b, c, d, rsqr}, WorkingPrecision → Infinity];]
{0.972061 Second, Null}
```

The exact values for the solution set have a leaf count of 12139. These solutions are comprised of algebraic numbers coming from defining polynomials with integer coefficients of several hundred digits. This is far too large to

warrant printing.

**Overview of other approaches to solving the cylinder equations**

We can improve considerably on the computational efficiency of finding cylinder parameters from five points. For one, a different formulation of the problem, to be utilized later, finds directions for which all points project onto the same circle in a plane perpendicular to the direction. Using this we can reduce the computational time by a substantial factor vs. the method shown above.

In addition to changing the formulation of the problem to one that is computationally easier, one might also change the solver method. We discuss one very efficient alternative. This is the sparse homotopy method described in [26]. Here one constructs a readily solved system using information from the Newton polytope. One then forms a homotopy to move from each solution of the first system to a solution of the new system. Specifically, if we call the systems $F(x)$ and $G(x)$ respectively, where $x$ denotes a vector of variables, then one adds a new variable, $t$, and sets up the homotopy between solutions in each set as a relation $(1 - t)\,F(x) + t\,G(x) = 0$. At time $t = 0$ we have a solution to the first system, and at time $t = 1$ we have a solution to the new system. Techniques for moving along the homotopy path generally utilize a predictor−corrector method to increment $t$ by a small amount and then alter the coordinates of $x$ to maintain the relation above; a general introduction to this method is presented in [24]. For our cylinder problem there is a nice refinement that goes by the name of the "cheater's homotopy" [26] wherein we start with known solutions for one set of points and hence can skip the first step of the general approach. In order to find cylinder parameters for each subsequent set of points we simply use a homotopy appropriate for the new set of equations.

An occasional disadvantage to the general sparse homotopy technique is that in some cases one has fewer actual solutions than are given by the starting system. When this occurs, in the process of following the homotopies some must wander off to infinity. This can pose difficulties for the software in terms of deciding when a path is diverging rather than merely wandering afar prior to converging. For our problem family the sparse homotopy method will predict that there are eight solutions for cylinder parameters, two more than are actually present. Hence the cheater's homotopy is all the more appealing for this class of problems. It should be noted, however, that the general sparse method [36] is far better at approximating the correct number of solutions than any earlier approach based on homotopies. Moreover it tends to handle systems with far more solutions than can successfully be tackled by methods that require computation of matrix eigensystems such as that presented in [10].

**The size of the solution set**

The preceding example had six solutions. We now investigate further. Again taking the polynomial system (6), we form a lexicographic Gröbner basis. This is a standard tactic for computational equation manipulation [1] [3] [6] [12] [18]. The idea is that it effectively triangulates the polynomial system in a manner that will become clear below. If we order so that $a$ is the lexicographically "smallest" variable then we obtain a univariate in that variable, along with other polynomials. As the coefficients are large we will only show that first polynomial.

$$33\,369\,819\,849\,015 - 260\,250\,873\,299\,469\,a + 250\,872\,620\,195\,750\,a^2 + 127\,385\,909\,908\,067\,a^3 - 186\,344\,103\,956\,650\,a^4 - 259\,033\,149\,843\,189\,a^5 + 8\,563\,282\,997\,415\,a^6$$

It is instructive to learn the structure of the Gröbner basis. The first polynomial is of degree 6 in the variable $a$ (as we already knew), and the rest are quintic in $a$ and linear and with a constant coefficient in each of the respective other variables. So now we see what was meant by triangularizing the system. To solve it one could find the six roots in $a$ and back substitute each into the remaining equations in order to get six corresponding solutions in each of the remaining variables.

This tells us to expect six solutions in general. As noted earlier this result may be found in several references. Later we will give computational proofs. For now we offer two reasons to believe this result; each may be viewed as a Monte Carlo "proof".

1: In the theory of lexicographic Gröbner bases there is a fact known as the Shape Lemma [2], which may be stated as follows. As is well known, a generic zero dimensional polynomial ideal over an infinite field is radical and in general position with respect to the last variable in any ordering of the variables. In other words, the variety has no multiplicity and moreover its finitely many points do not share any coordinates. The lemma states that under these

circumstances any lexicographic Gröbner basis will have exactly one polynomial with leading term a pure product in each variable, all but the one in the smallest variable will be linear, and that one in the smallest variable will have degree equal to the size of the solution set. One interpretation of the Shape Lemma is almost a matter of philosophy: one proves the above fact given a radical ideal in general position, and then asserts that generic ideals satisfy these hypotheses. In addition to the Shape Lemma there is the following result: lexicographic Gröbner bases of ideals defined over rational function fields remain Gröbner bases after generic specialization of coefficients [17] [22]. In other words, there is a Zariski−open set in the parameter space for which specializations do not alter the skeleton of the basis. We use these facts as follows: if our selection of coefficients was generic, we may conclude that the generic Gröbner basis has the same shape as that of the basis we just obtained. Moreover we may believe that our selection was generic because (i) it had the correct shape of a generic basis, and (ii) we used pseudorandom data selected from a fairly large set.

2: A simulation with $2^{12}$ randomly chosen configurations always gave exactly six solutions.

One might ask why we do not simply compute a lexicographic basis for our system using indeterminates as coeffi−cients. The answer is that it does not finish in finite time. Indeed, even making one coordinate a parameter leads to tremendous computational effort (several minutes on current processors) and very large coefficients for the basis. That is to say, polynomials in that parameter are of high degree and have large integer coefficients. An alternative computational approach to finding generic cardinality of solution sets—one with the added virtue of being tractable—is presented in [29].

As Gröbner bases computations never leave their base field (that is, if we begin with real data then the polynomials in the basis will have real coefficients) we conclude that complex solutions will be in pairs. Thus we might have zero, two, four, or six (real) cylinders in $\mathbb{R}^3$. In the case of the example above we have two. We will later use the results of the large simulation tests to say a bit about percentages of examples for which one obtains given numbers of real solutions.

## 3. Computational Geometry of the Solution Cylinders

**Finding the implicit equation of a cylinder from its parametric form**

Given the parameters of a cylinder, it is natural to ask how one might obtain the implicit form. The first method we show, best described as "applied brute force", is from modern elimination theory. Some references for this tech−nique are [1] [12] [19] [23]. We begin with equations for $(x, y, z)$ in terms of the five parameters and the sine and cosine of an (unrestricted) angular parameter.

In more detail, we have a parametrization for the cylinder in terms of a scalar multiplier $t$ for the direction vector *vec* and an angle $\theta$ to determine a unit vector in the plane orthogonal to *vec*. To make this parametrization algebraic we can use the usual pair of trigonometric functions, abbreviated below as algebraic variables (*cos*, *sin*). This gives one parameter more but of course we also now have the polynomial $cos^2 + sin^2 - 1$. A standard Gröbner basis method for elimination of variables uses a term ordering that is typically efficient for partially triangularizing the polynomials. In particular it weights terms that involve any of the elimination variables higher than all other terms. We form a Gröbner basis with respect to such an ordering and remove all polynomials that contain any of the elimination variables (*t*, *cos*, *sin*). What remains, a single polynomial, is the implicit relation in the variables $(x, y, z)$.

$$b^2 + b^2 c^2 - 2 a b c d + d^2 + a^2 d^2 - r^2 - a^2 r^2 - c^2 r^2 + (2 a b + 2 c d) x + \left(a^2 + c^2\right) x^2 +$$
$$\left(-2 b - 2 b c^2 + 2 a c d\right) y - 2 a x y + \left(1 + c^2\right) y^2 + \left(2 a b c - 2 d - 2 a^2 d\right) z - 2 c x z - 2 a c y z + \left(1 + a^2\right) z^2$$

Note that, as one might expect, the implicit polynomial is a function of $r^2$.

For the example using cylinder parameters from (4) the implicit polynomial is computed to be

$$-420 + 4 x + 25 x^2 - 92 y - 6 x y + 17 y^2 + 68 z - 8 x z - 24 y z + 10 z^2$$

There is a simpler way to find the implicit form for a cylinder. Just use the formulation we described for finding the distance from a point to the axial line. This gives an equation satisfied by every point on the cylinder. Hence it will be the hypersurface expression we seek. As is so often the case, we see that brute force can be useful but it is no match for finesse. The first approach remains of interest because it is a standard technique in computational algebraic geometry, and works when geometric intuition may not be so readily available.

**Finding cylinder parameters from the implicit form**

Now we look into the reverse problem of finding parameters from the implicit form. While algebraic parametrization is in general difficult, the case of quadric surfaces in $\mathbb{R}^3$ is not terribly hard; general methods for this are presented in chapter 5 of [19]. For the case of cylinders we will show a very simple approach which we illustrate using the example above.

As we know the general implicit form, it suffices merely to equate coefficients with those of the specific implicit form and solve for the parameters. Some of the coefficients are linear in the cylinder parameters so this is computationally quite easy. For the example with cylinder parameters given in (4) the implicit form is shown in the previous subsection. One can equate coefficients using, for example, the *Mathematica* function `SolveAlways`. This yields, as we expect, $\{rsqr \to 21,\ b \to 2,\ d \to -1,\ a \to 3,\ c \to 4\}$.

Were the coefficient equations not so readily solvable we could instead do as follows. Starting with that cylinder in implicit form we generate at least five points that lie on it. To this end we might simply take values for $(x, y)$ coordinates, and solve for $z$. We then form equations for the parameters from the first five points and solve them. This gives candidate parameter values. Last we find the implicit equation corresponding to each set of parameters: the correct parameters will be the ones that recover the original implicit form (up to scalar multiple).

**Solving for overdetermined cylinders**

An important question to ask is what we might do to find a cylinder when we are given more than five given points? The typical case is where the points all lie approximately on a cylinder and we wish to find the best fitting one (perhaps to assess tolerance). We will use a local optimization method for this task. We can set up an expression to minimize as follows. First form the list of orthogonal complements to projections of our points onto the axial line. Then take a sum of squares of differences between projected lengths and radius.

We already saw that it is quite important to have good starting values. We do this by taking five points, solving for all exactly determined cylinder parameters therefrom, and then using other points to decide which of the six possibilities we should utilize. Specifically, at the set of "good" approximations we will have real values and our sum of squares will be near zero. This is referred to as a minimal subset method.

To illustrate we resurrect our original example but this time we use more points and we add random noise to all of them. The eight points below are thus perturbed slightly from the known example cylinder.

$(-2.61303,\ 4.97448,\ -3.39489),\ (-6.50929,\ -17.4652,\ -19.5735),$

$\left(9.39443,\ 18.5\quad 3.93821\times 10^{12},\ 18.2057\right),\ (12.7263,\ 29.6737,\ 32.2087),\ (5.481,\ 20.4069,\ 30.6016),$

$(7.21938,\ 33.4364,\ 34.1586),\ (10.6382,\ 20.9278,\ 29.2479),\ (-4.81338,\ -25.7862,\ -39.1488)$

We first obtain a set of candidate starting values. Using the method described above, we get six possible sets. We select the best candidate by calculating values of the six implicit equations at all points, summing absolute values for each equation over all points, and using the parameters that correspond to the implicit equation that yields the smallest such sum. For a particular set of choices we obtained the residuals below.

$(340.514,\ 340.514,\ 8047.95,\ 0.1098,\ 1973.7,\ 1973.7)$

It is clear that the fourth set of initial values is the one we should use. With this providing initial values a local minimization of the sum of squares of residuals gives the resulting parameter values below.

$\left\{4.02685\times 10^{-9},\ \{a \to 2.99996,\ b \to 1.99945,\ c \to 3.99999,\ d \to -1.00099,\ r \to 4.58255\}\right\}$

As a general remark, attempts with different optimization methods indicate that this sort of expression is quite problematic to minimize without reasonable starting points. Hence the ability to solve the exactly determined system is quite important as it provides an essential preprocessing step.

There are interesting applications to this. In the industrial realm of geometric tolerancing one wishes to measure how well an object conforms to specifications. The cylinder is of course a very common object in manufacture. A

good approach to metrology involving cylinders may be found in [15]. The technology discussed therein is espe−cially effective when the object in question is small and may be readily positioned, but one might accept a cruder approach e.g. to check an underground pipeline. For this sort of task one could probe five points, obtain from them a set of approximate cylinder parameters, then probe several others and obtain parameters for a least−squares nearest cylinder as above. One can then check whether all probed points are within specification tolerance in actual radial measure from the computed axial line. Other applications include fitting a cylinder to a point cloud [8] [9], positioning of femur pieces for surgical fracture reduction [37], and the first step of fitting peptides and other biomacromolecules to a helix [4]. We note that the method above is strictly a fitting problem. If we wish to fit points in regions with multiple objects we must preprocess via image segmentation. A robust statistics approach is presented in [31] that also relies on sampling exact fits of minimal subsets. In order to qualify as "robust" it requires a method to distinguish and discard outliers. Once the object points in the region are segmented one can then fit cylinders as above.

**Visualization of cylinders containing a set of points**

Once one has parameters for real cylinders containing a set of points one might wish to plot the configuration. For this purpose it is often useful to shrink the cylinder radius mildly so that the points are more readily visible. We also connect them by segments as this tends to make more clear how they are situated on the cylinder.

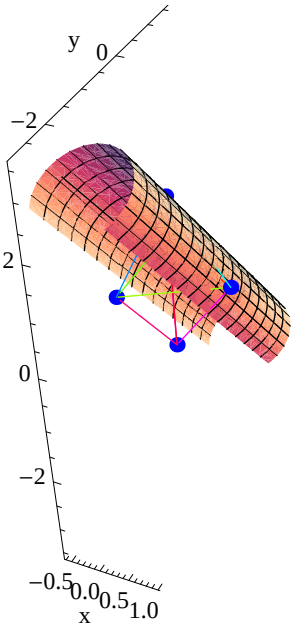
Let us look at an interesting configuation. Our points are $(1, 0, 0)$, $\left(-1/2, \sqrt{3}/2, 0\right)$, $\left(-1/2, -\sqrt{3}/2, 0\right)$, $\left(0, 0, \sqrt{2}\right)$, and $\left(0, 0, -\sqrt{2}\right)$. One notes that it is hardly generic in the sense that the points form a double regular tetrahedron (with edge length of $\sqrt{3}$). In particular gives a pair of cylinders with axes parallel to the $y\,z$ coordinate plane, and this means we cannot obtain all six cylinder parameters as solutions to the equations based on (1), (3) that we have worked with thus far.

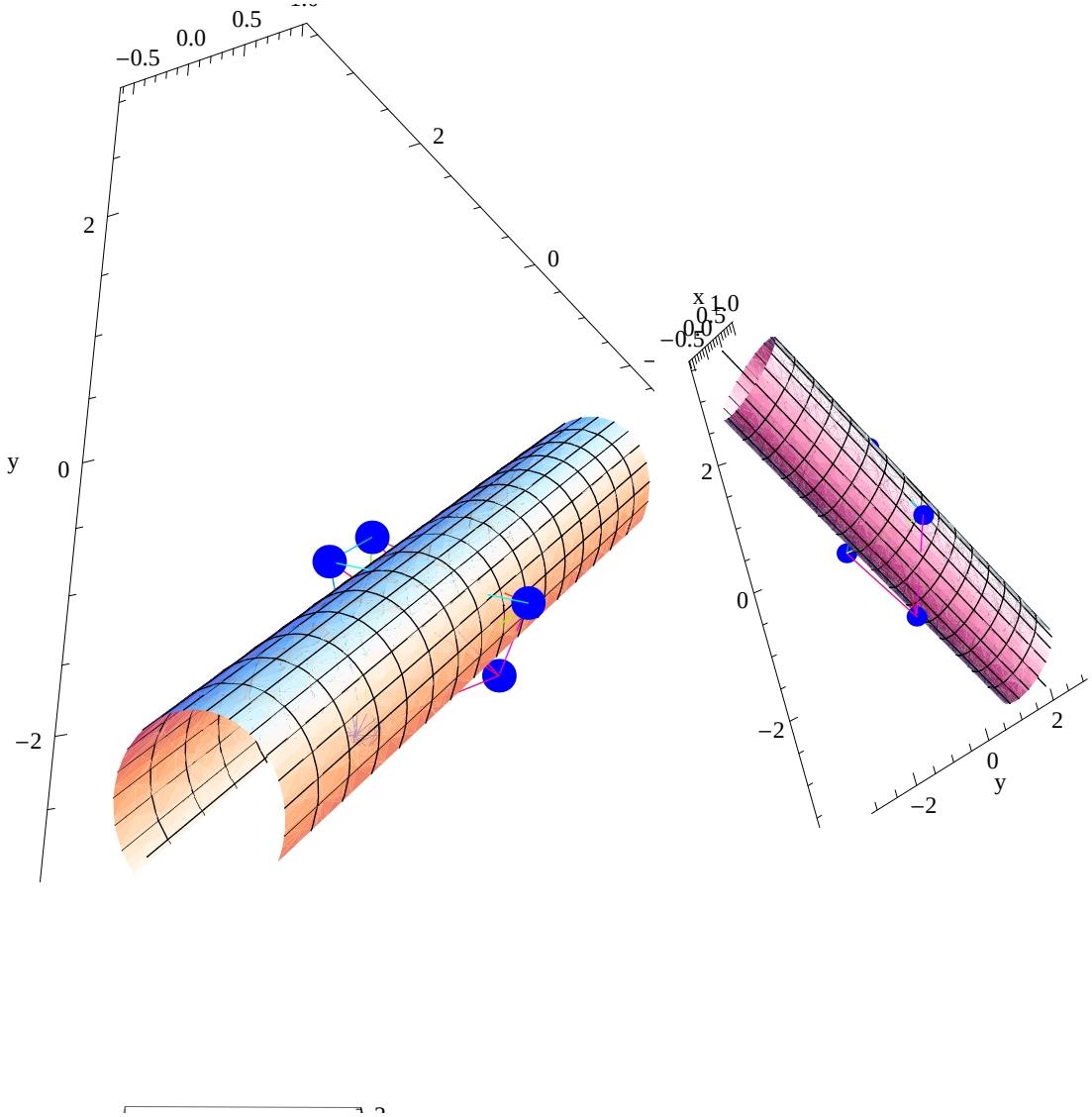
For this example we instead use the third coordinate to parametrize the cylinder axis, as $\{a\,z + b,\, c\,z + d,\, z\}$. The change this imposes on the code is quite modest and is indicated in the appendix. With respect to this setup one gets a very simple set of parameter values where the radii are all $9/10$. Two are

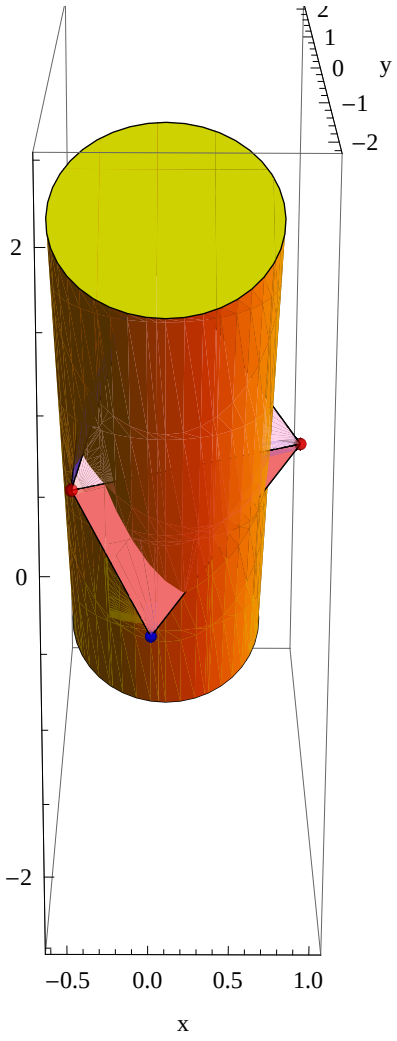
$$(a,\, b,\, c,\, d,\, rsqr) = \left(0,\, 1/10,\, \sqrt{2/3}\,,\, 0,\, 81/100\right)$$

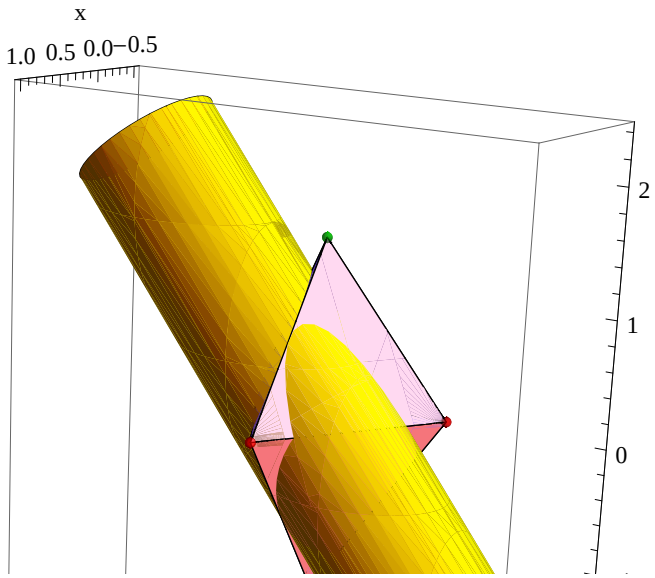
$$(a,\, b,\, c,\, d,\, rsqr) = \left(-1\Big/\sqrt{2}\,,\, -1/20,\, -1\Big/\sqrt{6}\,,\, \sqrt{3}\,\Big/20,\, 81/100\right)$$

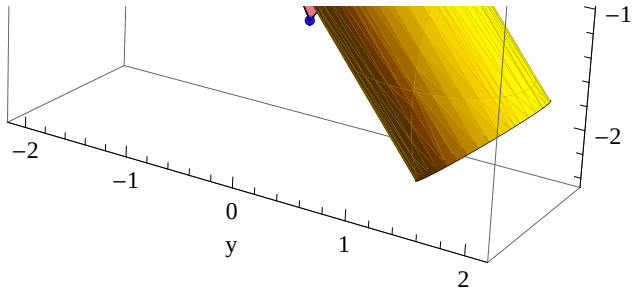
The rest are similar to these first two but with various of the first four components negated. Here are plots using various viewpoints. We show some renditions of the first solution cylinder using different graphics and viewpoints. This may help to visualize how the cylinder axes pass through pairs of faces of the doubled tetrahedra.


y
0
−2
2
0
−2
−0.5
0.0
0.5
1.0
x

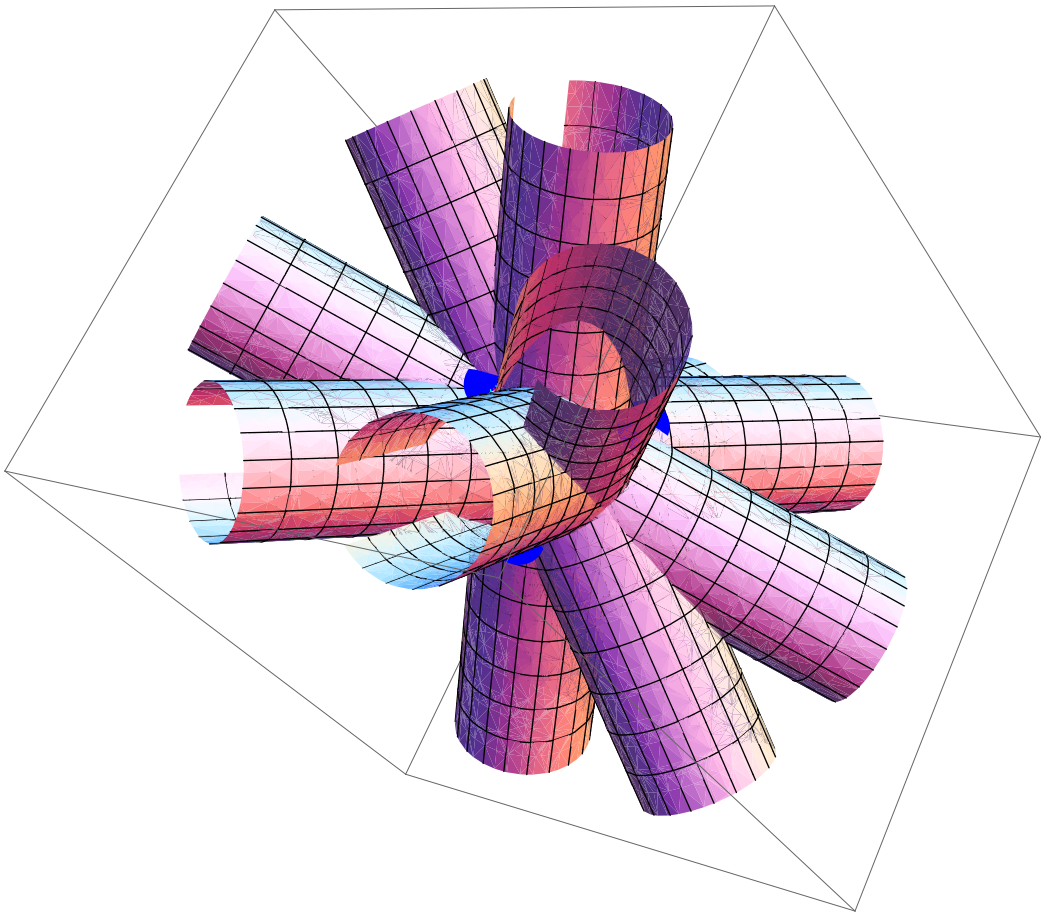

−0.5
0.0
0.5
2
0
y
0
−2
x
1.0
0.5
0.0
−0.5
2
0
−2
2
0
y
−2


2
1
0
y
−1
−2
2
0
−2
−0.5
0.0
0.5
1.0
x


x
1.0
0.5
0.0
−0.5
2
1
0




Here is a plot of all six cylinders containing these five points.



It is interesting to note that from this double regular tetrahedron construction one may obtain twelve real cylinders of a certain radius that intersect four particular points. Such an example was first presented in [30]; here we show how it arises naturally from our construction above. We begin with a regular tetrahedron and this time glue four others onto it, one on each face. The vertices of the original tetrahedron will be our four points. Clearly from each of the glued on tetrahedra we get the six cylinders as above, each intersecting those four points and all having the same radius.While this would appear to give $4 \times 6 = 24$ cylinders, they pair off so that the actual total is twelve. We remark that computational techniques essentially identical to those we have shown can be used to find the parame− ters for this problem as well. A different approach, using homotopy continuation methods as described e.g. in [26], was employed in [16]. A generalization of finding cylinders of possibly different given radii through four points has been studied in [20]. In [34] there is a construction giving 6 real cylinders that is similar to the one above, but using a perturbed configuration that avoids symmetry. Computations are similar to those in the appendix.

## 4. Real Cylinders: Probabilities and Configurations

### Enumerating real cylinders

We now investigate cases in which a configuration of five points will give rise to the various possible numbers of real cylinders containing it. First we note one obvious situation for which there can be no real cylinders: if one point is inside the convex hull of the other four then, as right circular cylinders are convex, no real cylinder can contain all five points. It would be interesting to know how frequently this arises for point sets that are random under some reasonable distribution. A simple simulation is revealing. We used $2^{12}$ examples with point coordinates chosen as independent and uniformly distributed pseudorandom integers in the range $[-100, 100]$. From these we found the frequencies of zero, two, four, or six real cylinders.

In one such simulation the frequencies obtained were (931, 2206, 865, 94). So roughly 23% give no real cylinders. It is natural to ask whether these are all configurations in which one point is enclosed by the other four. This turns out not to be so. We first discuss the frequency of such random configurations for which one point is enclosed by the hull of the other four. Presently we will see an open set in the configuration space for which no point lies inside the hull of the rest, and for which there are no real cylinders through all points.

To approximate the one−enclosed−by−four situation we generated $2^{14}$ random configurations and checked how many cases one point was within the convex hull of the other four. In a simulation we obtained $1147/2^{14}$ or about 0.070. Thus, for the no−real−cylinder examples subject to the distribution of points we used (which closely approxi−mates points uniformly distributed in a cube), we surmise that almost three out of four cases do not arise in this way. A partly proven conjecture in [28] states that the remaining configurations with no real solutions may be regarded as perturbations of the ones wherein one point is enclosed by the other four.

The frequency of one point being enclosed by the others is related to some classical problems in integral geometry. One way to pose it is as a three dimensional version of Sylvester's problem [13]: What is the probability that five points chosen at random in a unit cube all lie on the convex hull they define? Another variant is to find the expected volume of a random tetrahedron in the unit cube (several other variations are posed in the reference). We will call this expected volume *vTet*. To see how these problems are related, we order the five random points, then ask what is the probability that the first is enclosed by the others. This is exactly that expected volume. Now observe that the expected likelihood that any one point is enclosed by the other four is $5\,vTet$, as these are each pairwise exclusive events. Indeed, by taking the average of the five cases of one−point−enclosed−by−the−rest one obtains a Monte Carlo simulation of *vTet*: it is in the ballpark of $1/70$.

Taking this another step we might refine the estimate by quadrature. We utilized a quasi−Monte Carlo evaluation and obtained as our approximation 0.01364. This is clearly in accord with the approximation by simulation described above.

The problem of finding the expected volume of a tetrahedron with vertices independently and uniformly distributed inside a cube was recently solved [39] using an elaborate breakdown of the region and several exact multivariate integral computations. The actual value is $3977/216\,000 - \pi^2/2160$, or approximately .013843. This agrees with both the quadrature result and the simulation to almost three decimal places.

### Configurations that give six real cylinders

We previously obtained six real cylinders above by starting with a regular tetrahedron and gluing a copy of itself to one face to obtain five points. If the common face is in the $x\,y$ plane (so that one tetrahedron points up, the other down), then each intersects one of the three faces of the upper tetrahedron and one the faces of the lower not connected by an edge to the intersected upper face. In fact it is quite clear by symmetry that if we have one real cylinder then we must have six: we get two "conjugates" by rotating, and three more by reflecting through the $x\,y$ plane. There is another configuration, from [32], that can be seen to give six cylinders. We have four points forming vertices of a square in the $x\,y$ plane. This is the base of a pyramid with the fifth point as its apex above the centroid of this square. We obtain two horizontal cylinders each passing through a pair of opposite triangular faces of the pyramid. The remaining four each pass through a triangular face, angled upward, and an edge of the base.

It is a (vague) conjecture that all configurations giving rise to six cylinders in $\mathbb{R}^3$ are small perturbations of one of

these two configurations. This idea, admittedly difficult to quantify, is based on visual experimental evidence.

**More configurations that give no real cylinders**

As noted earlier we get no real cylinders whenever one point is in the convex hull of the other four. It is also clear from experiments (and theoretical grounds discussed in [28]) that there are other configurations that give no real cylinders. We now use symbolic computation to derive a particular family of such configurations.

We begin with a double tetrahedron glued along a common face in the horizontal plane, allowing the upper vertex to vary on a vertical line. To make results of computations more concise we now work with a lower $z$ coordinate that is $-1$ instead of $-\sqrt{2}$. Thus our points are now (1, 0, 0), $\left(-1/2, \sqrt{3}/2, 0\right)$, $\left(-1/2, -\sqrt{3}/2, 0\right)$, (0, 0, −1), and (0, 0, $z$). One may readily check that when the indeterminate coordinate is 1 we have 6 real cylinders (all radii are $5/6$). If we alter either or both of the upper and lower vertices we can jump from having six cylinders through the five points to having none. This is explained via a symmetry argument that we outline below. As noted earlier, if we have one real cylinder from such a configuration then the threefold symmetry will give us two more, for three cylinders (counting multiplicity). As this is an odd value either we must have another (and again by threefold symmetry, six altogether), or else there must be multiplicity. One can argue against multiplicity on geometric grounds, but a simple algebraic observation is that in any case we cannot have multiple solutions on more than a finite set of configurations as we move that top vertex along a vertical line (else we would have multiplicity of solutions everywhere on that variety in the configuration space). As we cannot have three real solutions counting multiplicity, we see that we either have six or none.

Below we explicitly show this phenomenon. Note that as we use a Gröbner basis approach we cannot in any straightforward way impose positivity on that mobile vertex. Were it to become negative we would have no real cylinders because either it or the other negative vertex will be in the tetrahedron hull of the remaining four vertices. Our interest is in the case where it gets larger, however, and our discussion will handle that.

We use the five equations based on (2), with the axial parametrization $\{a\,z + b,\ c\,z + d,\ z\}$, to obtain the cylinder polynomials. Let us now look at a lexicographic Gröbner basis for this polynomial set. We regard the moving vertex vertical coordinate $z$ as a parameter and do the basis computation over the rational function field in that parameter. When the variables are ordered so that $c$ is lexicographically smallest we have the basis given below.

$$\begin{aligned}
&\{-6 + 16\,z - 20\,z^2 + 8\,z^3 + c^2\,(-9 + 18\,z) + c^4\left(-12 + 48\,z - 48\,z^2\right) + c^6\left(-4 + 24\,z - 48\,z^2 + 32\,z^3\right),\\
&\quad 2 + c^2\,(5 - 10\,z) + a\left(-2 + 6\,z - 4\,z^2\right) + c^4\left(2 - 8\,z + 8\,z^2\right),\\
&\quad c^3\,(-5 + 10\,z) + c^5\left(-2 + 8\,z - 8\,z^2\right) + c\left(2 - 8\,z + 4\,z^2\right) + d\left(-4 - 4\,z + 8\,z^2\right),\\
&\quad -1 - 2\,z + b\left(2 - 8\,z^2\right) + c^4\left(-2 + 8\,z - 8\,z^2\right) + c^2\left(-4 + 6\,z + 4\,z^2\right),\ 5 + 20\,z^2 + \mathrm{rsqr}\left(-4 - 16\,z - 16\,z^2\right)\}
\end{aligned}$$

We saw from symmetry considerations that we obtain at least one real solution if and only if we obtain six real solutions. So it suffices to indicate situations where we cannot have six. For this we focus on the univariate polynomial in the last variable, $c$. First note that it is a cubic polynomial in $c^2$. For our task it suffices to find values of $z$ for which this cubic has no positive roots. Writing the cubic in a new variable $s = c^2$ we have

$$-6 + 16\,z - 20\,z^2 + 8\,z^3 + (-9 + 18\,z)\,s + \left(-12 + 48\,z - 48\,z^2\right)s^2 + \left(-4 + 24\,z - 48\,z^2 + 32\,z^3\right)s^3 \qquad (7)$$

For $z$ sufficiently large the leading coefficient is asymptotically $32\,z^3$. Dividing (7) by this leading coefficient we have a cubic with quadratic and linear coefficients, as rational function in $z$, asymptotically going to 0, and "constant" term approaching 1/4. That is, for $z$ sufficiently large, our cubic approaches $s^3 + 1/4$. As this does not have positive roots, neither does the cubic for sufficiently large $z$. Hence the sixth degree polynomial in $c$ has no real roots when $z$ is large, so the system has no real solutions in that case.

Note that for all but finitely many values of $z$ we have a lexicographic Gröbner basis with generic shape. This follows from the main theorem of the next section (because it gives 6 solutions, has first polynomial univariate and of degree 6, and remaining polynomials linear in each respective variable; in order to conclude genericity we need to know the generic solution count is 6). This holds in particular for sufficiently large $z$. In that case we just showed that there are no real cylinders containing the five points. From this it is not hard to show that small perturbations of any of the five points in any directions will also not give rise to real valued cylinder parameters (because suffi−

ciently small perturbations of the input will still give a basis conforming to the Shape Lemma). Hence we have an open set in configuration space for which there are no real solutions.

## 5. Counting Cylinders Through Five Points

The above investigations indicate computational ways in which one might approach questions involving cylinders through five points. We now show how purely computational methods can be brought to bear on some of the theory. Related results are presented in [28].

PROPOSITION 1. *Generic configurations of five points in $\mathbb{R}^3$ lie of the surface of finitely many cylinders. Moreover an upper bound on the number of these cylinders is nine.*

PROOF: We set up some linear algebra similar to that already seen, but now we reduce to two equations in two variables along with the configuration parameters. The linear algebra is as follows. Without loss of generality we have one point at the origin, another at (1, 0, 0), and a third in the *x y* coordinate plane. We project these onto the set of planes through the origin, parametrized generically by a normal vector (*a*, *b*, 1). In each such plane these three points determine a circle, and we get one equation for each of the remaining two points in order that they project onto the same circle (which is the condition that the five be cocylindrical). Our points are (0, 0, 0), (1, 0, 0), ($x_2$, $y_2$, 0), ($x_3$, $y_3$, $z_3$), and ($x_4$, $y_4$, $z_4$). From these we obtain the polynomials below.

$$
\begin{aligned}
&\big(-x_3\,y_2 - b^2\,x_3\,y_2 + x_3^2\,y_2 + b^2\,x_3^2\,y_2 + x_2\,y_3 + b^2\,x_2\,y_3 - x_2^2\,y_3 - b^2\,x_2^2\,y_3 + 2\,a\,b\,x_2\,y_2\,y_3 - \\
&\quad 2\,a\,b\,x_3\,y_2\,y_3 - y_2^2\,y_3 - a^2\,y_2^2\,y_3 + y_2\,y_3^2 + a^2\,y_2\,y_3^2 - b\,x_2\,z_3 - b^3\,x_2\,z_3 + b\,x_2^2\,z_3 + b^3\,x_2^2\,z_3 + a\,y_2\,z_3 + \\
&\quad a\,b^2\,y_2\,z_3 - 2\,a\,b^2\,x_2\,y_2\,z_3 - 2\,a\,x_3\,y_2\,z_3 + b\,y_2^2\,z_3 + a^2\,b\,y_2^2\,z_3 - 2\,b\,y_2\,y_3\,z_3 + a^2\,y_2\,z_3^2 + b^2\,y_2\,z_3^2, \\
&-x_4\,y_2 - b^2\,x_4\,y_2 + x_4^2\,y_2 + b^2\,x_4^2\,y_2 + x_2\,y_4 + b^2\,x_2\,y_4 - x_2^2\,y_4 - b^2\,x_2^2\,y_4 + 2\,a\,b\,x_2\,y_2\,y_4 - \\
&\quad 2\,a\,b\,x_4\,y_2\,y_4 - y_2^2\,y_4 - a^2\,y_2^2\,y_4 + y_2\,y_4^2 + a^2\,y_2\,y_4^2 - b\,x_2\,z_4 - b^3\,x_2\,z_4 + b\,x_2^2\,z_4 + b^3\,x_2^2\,z_4 + a\,y_2\,z_4 + \\
&\quad a\,b^2\,y_2\,z_4 - 2\,a\,b^2\,x_2\,y_2\,z_4 - 2\,a\,x_4\,y_2\,z_4 + b\,y_2^2\,z_4 + a^2\,b\,y_2^2\,z_4 - 2\,b\,y_2\,y_4\,z_4 + a^2\,y_2\,z_4^2 + b^2\,y_2\,z_4^2\big)
\end{aligned}
\tag{8}
$$

Factoring shows that they are irreducible and hence are relatively prime. So generically they have finite intersection and an upper bound is given by the Bezout theorem. In fact, as each polynomial has degree three in the variables (*a*, *b*), we see that there are at most nine solutions for the cylinder axis direction parameters, hence at most nine solutions for the set of cylinder parameters. □

In [14] it is noted that this projected circles approach is related to the Delaunay triangulation of projections of the five points on all possible planes. Specifically, directions of projection where the triangulation changes are impor−tant, as these occur exactly when four points become cocircular. This gives a direct tie between the enumerative and computational geometry of cylinders through five points.

PROPOSITION 2. *Real valued solutions always have positive values for the square of the radius.*

The significance of this proposition is that all real valued solutions do indeed give cylinders in $\mathbb{R}^3$.

PROOF. Suppose we form a lexicographic Gröbner basis for the system of five generic polynomials from (3), with the radius−square variable ordered as smallest. Then generically (Shape Lemma) we have a basis containing a univariate polynomial in that variable. For each of the other variables there will correspond a linear polynomial in the basis, and it will have real valued coefficients. Suppose a solution to that univariate polynomial is real valued. Then the remaining cylinder parameters, on back substitution, will also be real valued as they are given by linear polynomials over the reals. Now recall that our original equations were of the form equating a sum of squares to the squared radius. Here the left hand side is a polynomial function of the input data and cylinder parameters. Hence all the original equations will have positive left hand sides, so the radius squared must also be positive. □

THEOREM. *Five generic points in $\mathbb{R}^3$ determine six distinct sets of cylinder parameters, of which an even number (counting multiplicity) are real valued. That number can be zero, two, four, or six.*

Note that the number of real valued solutions being even follows from the fact that complex solutions must appear in conjugate pairs (since all the polynomial equations have real coefficients). Moreover we saw in the enumeration simulation that all four possible cases of real valued cylinder counts arise. So we need only prove that the number of solutions is six.

PROOF 1. We form a Gröbner basis with respect to a degree based term ordering for the polynomials (8) we created in proposition 1. Looking at the head terms we find that there are 6 monomials in (*a*, *b*) that are not

reducible with respect to this basis and hence 6 solutions to the system [10] [11]. □

PROOF 2. We compute the resultant of the pair of polynomials with respect to one of the two variables. We obtain a polynomial of degree 6 in the other (with large symbolic coefficients). This means there are at most 6 solutions. As we already know there are at least that many, this suffices to show that there are generically six solutions. □

REMARK 1: One might wish to use the method of mixed volume to compute the number of solutions [21]. One finds the convex hull of the Newton polytopes of the exponent vectors for each polynomial and then computes a mixed volume. This is easy to do using the computation from the proof of proposition 1. Each of the two polynomials has the same set of power products in ($a$, $b$) and specifically the hull of the exponent vectors is given by the vertex set (0, 0), (2, 0), (2, 1), (1, 2), and (0, 3).

The volume of this region is 4. The Minkowski sum of the two polytopes is just the same hull scaled to twice its size, and the mixed volume is equal to the total volume minus the sum of the volumes of each separate hull, or $16 - 8 = 8$. So the generic number of solutions for equations with these sets of exponent vectors is 8 rather than 6. Indeed, one can verify this immediately by solving a pair of random equations that use the same power products. We thus conclude that the entire family of cylinder problems is nongeneric with respect to the theory presented in [21]. A hint as to why this is so may be gleaned from the computational proof of theorem 1 presented in [28]. This sort of nongeneric example is also noted in [21]. The related problem discussed in [16] and [30] similarly fails to be generic for the polyhedral homotopy solving method.

REMARK 2: Proof 1 uses a brute force computation of a Gröbner basis for a system with generic configuration parameters. This approach is not tractable for most geometric problems, and that it worked here is indication of the relative simplicity of this formulation of the problem.

REMARK 3: Proof 2 is similar in method to an argument in [33] which implies that there are at most 12 cylinders of a given radius through four fixed points.

REMARK 4: Other proofs of varying levels of complexity may be found in [5] [8] [14] [28]. An algorithm that effectively automates finding the cardinality of generic solution sets to geometric configuration problems is given in [29]. It relies on showing that the solution count is constant in a neighborhood of a given point in configuration space.

## 6. Nongeneric solution sets (too many, too few, or too familiar)

Some further problems of interest include understanding the configurations of five distinct points that are degenerate for the problem at hand. Specifically we would like to know:

(1) When the number of solutions is infinite.
(2) When the number, counting multiplicity, is less than 6.
(3) When there are multiple solutions.

Some aspects of the first two questions are addressed in the companion paper [28]. Among other things we note that a sufficient condition to have either infinitely many cylinders through five points, or at most four, is that the points be coplanar. It is conjectured in [28] that these are also necessary conditions, and moreover that infinitely many cylinders exist exactly when either four points are collinear or three are collinear with the line determined by the remaining two parallel to the line through those first three. In this section we discuss how some ideas involving the discriminant variety [25] might be used to approach these questions. We emphasize that this is entirely tentative as we have not obtained concrete results with this approach to date, in part due to certain complexities presented by the problem formulation (and in no small way to gaps in the author's understanding of the work in question). We will describe code in the appendix that starts out along this computational path.

The basic idea, from [25], is to formulate polynomial conditions for where we do not have 6 solutions to our cylinder equations. We will start with the two equations for the two axis variables (these will be our "main" variables), given in (8). We create a block term order for all variables, including the indeterminate point coordinates, such that the heavier weighted block corresponds to the main variables. We find the basis and look at leading terms in the two main variables; these have coefficients in the indeterminate parameters. A necessary condition that the basis not be valid for a configuration is that such a leading term coefficient vanish.

This computational method has weaknesses. First, one really needs to ensure that the five points are distinct. This might be done by adding polynomial conditions (with new variables) to enforce that certain differences be nonzero (or at least that one difference from some set of possibilities be nonzero; this is not harder in principle but does use polynomials of higher degree in the new variables).

A second difficulty is that our algebraicization of the problem took a shortcut in setting a direction coordinate to 1. We may be in a situation similar to that posed by the doubled regular tetrahedra, wherein our initial choice of direction parameters could not describe all actual cylinders through the points. The code in the appendix will generate leading coefficient factors that include $y_2$, for example. Were this to vanish we would have three collinear points on the *x* axis, and this would force all cylinder directions to be in that direction and hence have a last coordinate of zero. Another such factor is shown below.

$$y_2\, y_4\, z_3 - y_4^2\, z_3 - y_2\, y_3\, z_4 + y_3^2\, z_4 + z_3^2\, z_4 - z_3\, z_4^2$$

It is less obvious how this might be a result of the limitations in our representation of the direction vector, though presumably that is the case. Other factors are far longer and correspondingly less amenable to this author's understanding.

We might try to adjust for the direction vectors which we cannot capture, e.g. by using polynomial relations to rule out configurations that would give such cylinders. An alternative would be to work with the computationally more difficult formulation wherein a direction vector has three variables and a new polynomial is used to "normalize" this direction (e.g. by making the sum of squares equal to 1). Actually this too is not going to get all possible cylinders because we consider complex solutions, and in that setting we can have a complex "direction" vector with square sum of coordinates vanishing. This method has the drawback of being computationally more intensive than the shortcut approach.

Yet another issue is that we may have a variety in the configuration parameter space for which leading terms in our basis vanish, but for which a different term ordering might behave perfectly well (these are called "representation singularities" in [35]). That is, we might have the correct solution count but a basis that is not of the same shape as generic bases.

A related matter of interest is to describe the configurations that give some given number (even, counting by multiplicity) of real solutions. In [28] there is considerable discussion of the case of no real cylinders. Again one might wish to approach this computationally using discriminant variety tools from [25]. Here the ideal of interest is the set of certain Jacobian minors (as well as the original polynomials). At multiple solutions these will vanish. Hence any characterization of these, intersected with real space, will include the boundaries in the configuration space between different numbers of real solutions. Computationally this would appear to be a daunting problem and it would be interesting to learn if any existing software can make progress with it. Also as the result is expressed in terms of algebraic relations, it would then be useful to understand from them the underlying geometric relations that describe the four cases of real solution cardinalities.

## 7. Summary

We have discussed computational methods for finding cylinders through a given set of five points in $\mathbb{R}^3$. Along the way we have covered several related problems and computational approaches thereto. We have investigated various real valued scenarios using simulation. Overall we have combined geometric reasoning with Gröbner bases and several related tools from symbolic computation in order to study a rich family of problems from enumerative and computational geometry.

## 8. References


[1] W. Adams and P. Loustaunau. *An Introduction to Gröbner Bases*. Graduate Studies in Mathematics **3**. American Mathematical Society, 1994

[2] E. Becker, M. G. Marinari, T. Mora, and C. Traverso. The shape of the Shape Lemma. In *Proceedings of the 1994 International Symposium on Symbolic and Algebraic Computation (ISSAC 94)*. 129−133. ACM Press, 1994.

[3] T. Becker, W. Weispfenning, and H. Kredel. *Gröbner Bases*: *A Computational Approach to Commutative Algebra*. Graduate Texts in Mathematics **141**. Springer−Verlag, 1993.

[4] G. Bellesia. Private communication, 2004.

[5] O. Bottema and G. Veldkamp. On the lines in space with equal distances to *n* given points. *Geometriae Dedicata* **6**:121−129. D. Reidel Publishing Company, 1977.

[6] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In *Multidimensional Systems Theory*, chap 6. N. K. Bose, ed. D. Reidel Publishing Company, 1985.

[7] B. Buchberger. Applications of Gröbner bases in non−linear computational geometry. In *Proceedings of the International*

*Symposium on Trends in Computer Algebra 87*, R. Janssen, ed. Lecture Notes in Computer Science **296**:52−80. Springer−Verlag, 1988.
[8] T. Chaperon and F. Goulette. A note on the construction of right circular cylinders through five 3D points. Technical Report, Centre de Robotique, Ecole des Mines de Paris. 2003.
[9] T. Chaperon, F. Goulette, and C. Laurgeau. Extracting cylinders in full 3D data using a random sampling method and the Gaussian image. In *Proceedings of the of the 6th International Fall Workshop Vision, Modeling, and Visualization (VMV 01)*, Stuttgart, Germany, November 2001. T. Ertl, B. Girod, G. Greiner, H. Niemann, and H−P. Seidel, eds. 35−42, 2001.
[10] R. Corless. Editor's corner: Gröbner bases and matrix eigenproblems. *ACM Sigsam Bulletin: Communications in Computer Algebra* **30**(4):26−32, 1996.
[11] D. Cox. Introduction to Gröbner bases. In *Proceedings of Symposia in Applied Mathematics* **53,** D. Cox and B. Sturmfels, eds. 1−24. ACM Press, 1998.
[12] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Computer Algebra*. Undergraduate Texts in Mathematics. Springer−Verlag, 1992.
[13] H. T. Croft, K. Falconer, and R. T. Guy. *Unsolved Problems in Geometry*. 54−55. Springer−Verlag, 1991.
[14] O. Devillers, B. Mourrain, F. Preparata, and P. Trebuchet. On circular cylinders by four or five points in space. *Discrete and Computational Geometry* **28**:83−104, 2003.
[15] O. Devillers and F. Preparata. Evaluating the cylindricity of a nominally cylindrical point set. In *Symposium on Discrete Algorithms (SODA 2000)*. 518−527, 2000.
[16] C. Durand. Symbolic and Numerical Techniques for Constraint Solving. Ph.D thesis. Purdue University, Department of Computer Science, 1998.
[17] P. Gianni. Properties of Gröbner bases under specialization. In *European Converence on Computer Algebra (Eurocal 87)*. J. Davenport, ed. Lecture Notes in Computer Science **378**:293−297. Springer−Verlag, 1987.
[18] P. Gianni and T. Mora. Algebraic Solutions of systems of polynomial equations using Gröbner bases. In *Applied Algebra, Algebraic Algorithms, and Error−Correcting Codes (AAECC5)*. L. Hugeut, A. Poli, eds. Lecture Notes in Computer Science **356**:247−257. Springer−Verlag, 1988.
[19] C. Hoffmann. *Geometric and Solid Modelling: An Introduction*. Morgan Kaufmann, 1989
[20] C. M. Hoffmann and B. Yuan. On spatial constraint solving approaches. In *Proceedings of the Third International Workshop on Automated Deduction in Geometry (ADG 2000)*. J. Richter−Gebert and D. Wang, eds. Lecture Notes in Computer Science **2061**:1−15. Springer−Verlag, 2001.
See also: There are 12 common tangents to four spheres, 2000. Website URL:
http://www.cs.purdue.edu/homes/cmh/distribution/SphereTangents.htm
[21] B. Huber and B. Sturmfels. Bernstein's theorem in affine space. *Discrete and Computational Geometry* **17**:137−141, 1997.
[22] M. Kalkbrenner. Solving systems of algebraic equations by using Gröbner bases. In *European Conference on Computer Algebra (Eurocal 87)*. J. Davenport, ed. Lecture Notes in Computer Science **378**:282−292. Springer−Verlag, 1987.
[23] M. Kalkbrenner. Implicitization of rational parametric curves and surfaces. In *Applied Algebra, Algebraic Algorithms, and Error−Correcting Codes (AAECC8)*. S. Sakata, ed. Lecture Notes in Computer Science **508**:249−259. Springer−Verlag, 1990.
[24] I. Kotsireas. Homotopies and polynomial system solving I: Basic principles. A*CM Sigsam Bulletin: Communications in Computer Algebra* **35**(1):19−32, 2001.
[25] D. Lazard and F. Rouillier. Solving parametric polynomial systems. Technical report: INRIA Rapport de recherche 5322, 2004.
[26] T. Y. Li, T. Sauer, and J. A. Yorke. The cheater's homotopy: an efficient procedure for solving systems of polynomial equations. *SIAM Journal on Numerical Analysis* **26**(5):1241−1251, 1989.
[27] D. Lichtblau. Solving finite algebraic systems using numeric Gröbner bases and eigenvalues. In *Proceedings of the World Conference on Systemics, Cybernetics, and Informatics (SCI 2000)*, Volume 10, 555−560. (Concepts and Applications of Systemics, Cybernetics, and Informatics). M. Torres, J. Molero, Y. Kurihara, and A. David, eds. International Institute of Informatics and Systemics, 2000.
[28] D. Lichtblau. Cylinders through five points: complex and real enumerative geometry. Manuscript, to be presented at the Sixth International Workshop on Automated Deduction in Geometry (ADG 2006).
[29] D. Lichtblau. Simulating perturbation to enumerate parametrized systems. Manuscript.
[30] I. G. Macdonald, J. Pach, and T. Theobald. Common tangents to four unit balls in $\mathbb{R}^3$. *Discrete and Computational Geometry* **26**(1):1−17, 2001.
[31] G. Roth and M. D. Levine. Extracting geometric primitives. *Computer Vision, Graphics and Image Processing: Image Understanding* **58(1)**:1−22, 1993.
[32] D. Rusin. Website URL:
http://www.math−atlas.org/98/5pt.cyl
[33] E. Schömer, J. Sellen, M. Teichmann, and C. Yap. Smallest enclosing cylinders. *Algorithmica* **27**:170−186, 2000.
[34] F. Sottile. From enumerative geometry to solving systems of polynomial equations with Macaulay 2. In *Computations in Algebraic Geometry with Macaulay 2*, D. Eisenbud, D. Grayson, M. Stillman, and B. Sturmfels, eds. Algorithms and Computation in Mathematics **8**:101−129. Springer−Verlag, 2001.
[35] H. Stetter. Stabilization of polynomial systems solving with Groebner bases. *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (ISSAC 97)*. 117−124. ACM Press, 1997.
[36] J. Vershelde, P. Verlinden, and R. Cools. Homotopies exploiting Newton polytopes for solving sparse polynomial systems. *SIAM Journal on Numerical. Analysis* **31**(3):915−930, 1994.
[37] S. Winkelbach, R. Westphal, and T. Goesling. Pose estimation of cylindrical fragments for semi−automatic bone fracture reduction. In *Pattern Recognition (DAGM 2003)*. B. Michaelis, G. Krell eds. Lecture Notes in Computer Science **2781**:566−573. Springer−Verlag, 2003.
[38] S. Wolfram. *The Mathematica Book* (5th edition). Wolfram Media, 2003.
[39] A. Zinani. The expected volume of a tetrahedron whose vertices are chosen at random in the interior of a cube. *Monat−shefte fü*r *Mathematik* **139:**341−348, 2003.

## 9. Appendix

We give *Mathematica* code used herein. Small amounts of text relate the code to the ways in which it is used in the body of this article. It should be of interest that in total there is not much code. This again points to the importance of symbolic−numeric computation in investigations of the sort found herein: this would be something of a hardship to code "from scratch" but is straightforward with programs that support such functionality.

Below is code used to set up our first example.

```
perp[vec1_, vec_, offset_] := vec1 - offset - Projection[vec1 - offset, vec, Dot]

{a, b, c, d, r} = {3, 2, 4, -1, Sqrt[21]};
vec = {1, a, c}; offset = {0, b, d};
pair = NullSpace[{vec}];
{w1, w2} = Orthogonalize[pair, Dot];

SeedRandom[111 111];
xvals = Table[Random[Real, {-10, 10}], {5}];
thetas = Table[Random[Real, {0, 2π}], {5}];
points = Table[xvals[[j]] vec + offset + r Cos[thetas[[j]]] w1 + r Sin[thetas[[j]]] w2, {j, 5}];
```

When we use this point set to find $r$ and the parameters describing $L$ we must of course clear those that we set above so they become symbolic indeterminates. Using these points we will obtain the needed algebraic expressions for extraction of roots.

```
Clear[vec, offset, a, b, c, d, r]
vec = {1, a, c}; offset = {0, b, d};
perps = Map[perp[#, vec, offset] &, points];
exprs = Map[Numerator[Together[#.# - r^2]] &, perps];
```

Here we attempt to get solutions to the parameter equations using local root−finding methods.

```
rt1 = FindRoot[Evaluate[Thread[exprs == 0]], {a, 3.4},
   {b, 2.8}, {c, 3.7}, {d, -1.6}, {r, 3.3}, MaxIterations → 500]
rt2 = FindRoot[Evaluate[Thread[exprs == 0]], {a, 2.7}, {b, 1.8},
   {c, 3.2}, {d, -.7}, {r, 3}, MaxIterations → 500]
```

Now we use local optimization, minimizing a sum of squares with the Levenberg−Marquardt method (which tends to be good for sums of squares).

```
esquares = Apply[Plus, Map[#^2 &, exprs]];
{m3, rt3} = FindMinimum[Evaluate[esquares], {a, 2.4}, {b, 1.8},
   {c, 3.2}, {d, -.8}, {r, 3}, MaxIterations → 500, Method → LevenbergMarquardt]
{m4, rt4} = FindMinimum[Evaluate[esquares], {a, 2.4}, {b, 1.8},
   {c, 2.2}, {d, -1.8}, {r, 3}, MaxIterations → 500, Method → LevenbergMarquardt]
```

We use NSolve as below to find all solutions.

```
Timing[solns = NSolve[exprs, {a, b, c, d, rsqr}]]
```

We can even use NSolve to find exact solutions.

```
Timing[exactsolns = NSolve[exprs, {a, b, c, d, rsqr}, WorkingPrecision → Infinity];]
```

In our second example we indicate the polynomial exponent structure of a set of polynomials. This is done with the code below.

```
gb = GroebnerBasis[exprs, Sort → True];
list1 = Map[Apply[List, #] &, gb] /. _Integer * x_ → x;
list2 = Map[Cases[#, a_ /; ! IntegerQ[a]] &, list1]
```

The code below will find the implicit form of a cylinder given a parametric formulation. It implements a standard elimination method using Gröbner bases.

```
Clear[a, b, c, d, t, r];
vec = {1, a, c}; offset = {0, b, d};
pair = NullSpace[{vec}];
{w1, w2} = Orthogonalize[pair, Dot];
polys = Append[t vec + offset + r cos w1 + r sin w2 - {x, y, z}, sin^2 + cos^2 - 1];
ee = Numerator[MapAll[Together, polys]];
ff = Numerator[Together[PowerExpand[ee]]];
implicit = First[GroebnerBasis[ff, {x, y, z}, {t, sin, cos}, Sort → True,
     MonomialOrder → EliminationOrder, CoefficientDomain→ RationalFunctions]];
```

We can check this on the particular example by showing that each given point is a root of the polynomial.

```
thisExample = implicit /. {a → 3, b → 2, c → 4, d → -1, r → Sqrt[21]}
thisExample /. Map[Thread[{x, y, z} -> #] &, points] // Chop
```

Here is another way to get the implicit form. It is quite simple, but, unlike the preceding general approach, it is specific to the geometric object in question.

```
pp = perp[{x, y, z}, vec, offset];
implicit2 = Numerator[Together[pp.pp - r^2]]
```

Here we find the parameter values from the implicit form of a cylinder.

```
generalImplicitForm = implicit /. r^2 → rsqr;
thiscase = generalImplicitForm /. {a → 3, b → 2, c → 4, d → -1, rsqr → 21};
SolveAlways[thisExample == generalImplicitForm ,{x, y, z}]
```

An alternative approach to the parametrization is given below.

```
points = Partition[
   Flatten[{x, y, z} /. Table[Solve[{thiscase == 0, x == j, y == 0}, {x, y, z}], {j, 0, 2}]], 3];
perps = Table[perp[points[[k]], vec, offset], {k, 5}];
exprs = (Numerator[Together[#1.#1 - rsqr]] &) /@ perps;
Select[candidates, NumericQ[Together[generalImplicitForm /. #1 / thiscase]] &]
```

Below we create an overdetermined example of eight points. It contains some random noise to move it from the exact parameter values.

```
SeedRandom[1 111 111];
{a, b, c, d, rsqr} = {3, 2, 4, -1, 21};
numpts = 8;
vec = {1, a, c}; offset = {0, b, d};
pair = NullSpace[{vec}];
{w1, w2} = Orthogonalize[pair, Dot];
xvals = Table[Random[Real, {-10, 10}], {numpts}];
thetas = Table[Random[Real, {0, 2 π}], {numpts}];
randomNoise3D[max_] := max Table[Random[], {3}]
points = Table[xvals[[j]] vec + offset + Sqrt[rsqr] Cos[thetas[[j]]] w1 +
    Sqrt[rsqr] Sin[thetas[[j]]] w2 + randomNoise3D[0.001], {j, numpts}];
```

We obtain a set of candidate starting values as below (again we need to clear the various parameters that we used above to recreate the example).

```
Clear[a, b, c, d, rsqr];
vec = {1, a, c}; offset = {0, b, d};
perps = Table[perp[points[[j]], vec, offset], {j, numpts}];
exprs = Map[Numerator[Together[Rationalize[#.#, 0] - rsqr]] &, Take[perps, 5]];
solns = NSolve[exprs, {a, b, c, d, rsqr}];
```

We find the correct set of starting values for a local optimization of parameters.

```
squaresums =
  Apply[Plus, Abs[(generalImplicitForm /. solns) /. Map[Thread[{x, y, z} → #] &, points]]];
candidate = solns[[Position[squaresums, Min[squaresums]][[1, 1]]]];
```

We use the starting values in a local minimization of a sum of squares in order to get a best fit cylinder to the point set.

```
sumsquarelens = Plus @@ ((Sqrt[#1.#1] - r)^2 &) /@ perps;
startvals = (List @@ #1 &) /@ candidate;
newstartvals = startvals /. {rsqr, v_} :> {r, Sqrt[v]};
{min, ee} = FindMinimum[Evaluate[sumsquarelens], Evaluate[Sequence @@ newstartvals]]
```

Below is code used for plotting a cylinder and point set.

```
showlines[points_List, rest___] := Module[{plotpoints, plotlines, len = Length[points]},
  plotpoints = Table[Graphics3D[{Blue, PointSize[0.05], Point[points[[j]]]}], {j, 1, len}];
  plotlines = Table[Graphics3D[{Hue[1/26 (j^2 + k - 4)], Line[{points[[j]], points[[k]]}]}],
    {k, 1, len - 1}, {j, k + 1, len}];
  Show[plotpoints, plotlines, rest, DisplayFunction → Identity, Boxed → False, Axes → True,
   AxesLabel → {"x", "y", None}, ViewPoint → {-1/2, 2, 1}, ImageSize → {300, 480}]]
cylinderplot[rt_, pts_List, vec_, offset_, showcyl_ : True] :=
 Module[{r, vec2, lin, x, theta, w1, w2, circ, cylplot, axis, pair},
  r = Sqrt[rsqr /. rt]; vec2 = vec /. rt; lin = x vec + offset /. rt;
  pair = NullSpace[{vec2}];
  {w1, w2} = Orthogonalize[pair, Dot];
  circ = 0.7 r Cos[theta] w1 + 0.7 r Sin[theta] w2 /. rt;
  cylplot = ParametricPlot3D[Evaluate[lin + circ], {x, -3, 3},
    {theta, -0.5 π, 0.85 π}, Shading → True, DisplayFunction → Identity];
  axis = ParametricPlot3D[lin, {x, -3.5, 3.5}, DisplayFunction → Identity];
  If[showcyl, showlines[pts, axis, cylplot, DisplayFunction → Identity],
   showlines[pts, axis, DisplayFunction → Identity]]]
```

Our example uses two regular tetrahedra glued along a face of each. The common face is in the $xy$ coordinate plane. As this cannot be handled with the setup used above, we take different direction and offset vectors. Specifically, we have solutions with axes parellel to the $yz$ coordinate plane, that is to say, $x$ coordinate of zero, and our generic axis of choice, (1, $a$, $c$), will not find these parameter sets. Hence instead we use ($a$, $c$, 1) as our direction vector.

```
dpoints = {{1, 0, 0}, {-1/2, Sqrt[3]/2, 0}, {-1/2, -Sqrt[3]/2, 0}, {0, 0, Sqrt[2]}, {0, 0, -Sqrt[2]}};
vec = {a, c, 1}; offset = {b, d, 0};
solveCylinders[pts_List, vec_, offset_, prec_ : Automatic] := Module[{exprs, k, perps},
  perps = Table[perp[pts[[k]], vec, offset], {k, 5}];
  exprs = (Numerator[Together[#1.#1 - rsqr]] &) /@ perps;
  NSolve[exprs, {a, b, c, d, rsqr}, WorkingPrecision → prec]]
```

One obtains the exact cylinder parameters as below.

```
solns = solveCylinders[dpoints, vec, offset, Infinity];
FullSimplify[{a, b, c, d, rsqr} /. solns]
```

The first two may be plotted with the following code.

```
nsols = N[solns];
Show[GraphicsArray[Table[cylinderplot[nsols[[j]], dpoints, vec, offset, True], {j, 2}]]];
```

A nice plot may be constructed more simply in the version of *Mathematica* under development at the time of this writing. We show it for the first solution cylinder.

```
{p1, p2, p3, p4, p5} = dpoints;
top1 = (vec * 2 + offset) /. solns[[1]];
bot1 = (-vec * 2 + offset) /. solns[[1]];
Graphics3D[
 {{Red, Sphere[#, 0.04] & /@ {p1, p2, p3}, Green, Sphere[p4, 0.04], Blue, Sphere[p5, 0.04]},
  {White, Opacity[0.1], Polygon[{{p1, p2, p4}, {p2, p3, p4}, {p3, p1, p4}}],
   Polygon[{{p1, p2, p5}, {p2, p3, p5}, {p3, p1, p5}}]},
  {Yellow, Opacity[0.6], Cylinder[{top1, bot1}, 0.70]}},
 Axes → True, AxesLabel → {"x", "y", None}]
```

One can instead plot the cylinders together in one picture using the code below.

```
multiplecylinderplot[rt_, pts_List, vec_, offset_] :=
 Module[{len = Length[pts], r, vec2, lin, x, theta,
   w1, w2, circ, cylplot, axis, cyls, plotpoints, plotlines},
  plotpoints = Table[Graphics3D[{Blue, PointSize[0.05], Point[pts[[j]]]}], {j, 1, len}];
  plotlines = Table[Graphics3D[{Hue[1/26 (j^2 + k - 4)], Line[{pts[[j]], pts[[k]]}]}],
    {k, 1, len - 1}, {j, k + 1, len}];
  cyls = Table[
    r = Sqrt[rsqr /. rt[[j]]];
    vec2 = vec /. rt[[j]];
    lin = x vec + offset /. rt[[j]];
    pair = NullSpace[{vec2}];
    {w1, w2} = Orthogonalize[pair, Dot];
    circ = 0.87 r Cos[theta] w1 + 0.87 r Sin[theta] w2 /. rt[[j]];
    axis = ParametricPlot3D[lin, {x, -3.5, 3.5}, DisplayFunction -> Identity];
    cylplot = ParametricPlot3D[Evaluate[lin + circ],
      {x, -3, 3}, {theta, -0.85 π, 0.85 π}, DisplayFunction -> Identity];
    cylplot
    , {j, Length[rt]}];
  Show[plotpoints, plotlines, Apply[Sequence, cyls], ViewPoint -> {1/4, 3, 0}]
 ]
```

For example we might plot the six from the doubled tetrahedron example.

```
solns = solveCylinders[dpoints, vec, offset];
multiplecylinderplot[N[Take[solns, 6]], dpoints /. z → Sqrt[2.], vec, offset]
```

Here we counted the number of real solutions for $2^{12}$ point configurations wherein coordinates were taken as pseudorandom integers uniformly and independently distributed in the range [−100, 100].

```
Clear[a, b, c, d, rsq];
vec = {1, a, c}; offset = {0, b, d};
pair = NullSpace[{vec}];
{w1, w2} = Orthogonalize[pair, Dot];
SeedRandom[1111];
len = 2^12;
intpoints = Table[Table[Random[Integer, {-100, 100}], {5}, {3}], {len}];
Timing[rvals = Table[perps = Table[perp[intpoints[[j, k]], vec, offset], {k, 5}];
    exprs = Map[Numerator[Together[#.# - rsqr]] &, perps];
    solns = NSolve[exprs, {a, b, c, d, rsqr}];
    rs = N[rsqr /. solns];
    {j, Cases[rs, _Real]}, {j, len}];]
```

We separate into numbers of real cylinders.

```
rvals2 = Sort[rvals, Length[#1[[2]]] ≤ Length[#2[[2]]] &];
lens = (Length[#1[[2]]] &) /@ rvals2;
lenlens = Length /@ Split[lens]
```

Below is a simulation of the one−enclosed−by−four configuration. The code below will generate random configurations and then check to see in how many cases one point lies within the convex hull of the rest.

```
plane[p1_, p2_, p3_] := With[{norm = Cross[p1 - p2, p1 - p3]},
  {norm, norm.p1}]
sameside[{p1_, p2_, p3_}, p4_, p5_] := Module[{norm, d},
  {norm, d} = plane[p1, p2, p3];
  (norm.p4 - d) * (norm.p5 - d) > 0]
encloses[pnts : {p1_, p2_, p3_, p4_}, p5_] := Module[{combos, j},
  combos = Table[{Drop[pnts, {j}], pnts[[j]]}, {j, Length[pnts]}];
  Apply[And, Map[sameside[#[[1]], #[[2]], p5] &, combos]]]
anyenclosed[pnts : {p1_, p2_, p3_, p4_, p5_}] := Module[{combos, j},
  combos = Table[{Drop[pnts, {j}], pnts[[j]]}, {j, Length[pnts]}];
  Apply[Or, Map[encloses[#[[1]], #[[2]]] &, combos]]]
SeedRandom[1111];
len = 2^14;
realpoints = Table[Table[Random[Real, {-100, 100}], {5}, {3}], {len}];
Timing[enclosedlist = Transpose[{Range[len], Map[anyenclosed, realpoints]}];]
```

We now can check the proportion of cases with one point enclosed by the rest.

```
hasenclosed = Cases[enclosedlist , {_, True}];
numenclosed = Length[hasenclosed]
N[numenclosed / len]
```

We estimate via quadrature the expected volume of a tetrahedron with vertices uniformly and indepently distributed in the unit cube.

```
vol[p1_, p2_, p3_, p4_] := Abs[(p2 - p1).Cross[p3 - p1, p4 - p1]] / 6

NIntegrate[Evaluate[vol[{x1, y1, z1}, {x2, y2, z2}, {x3, y3, z3}, {x4, y4, z4}]],
  {x1, 0, 1}, {y1, 0, 1}, {z1, 0, 1}, {x2, 0, 1}, {y2, 0, 1}, {z2, 0, 1},
  {x3, 0, 1}, {y3, 0, 1}, {z3, 0, 1}, {x4, 0, 1}, {y4, 0, 1}, {z4, 0, 1},
  PrecisionGoal → 2, AccuracyGoal → 6, MaxPoints → 1 000 000]
```

We generate the six real cylinders containing points that are vertices of a pyramid with square base in the horizontal plane and four upward triangular faces.

```
dpoints = {{1, 0, 0}, {-1, 0, 0}, {0, 1, 0}, {0, -1, 0}, {0, 0, 3/2}};
vec = {1, a, c}; offset = {0, b, d};
solns = solveCylinders[dpoints, vec, offset]
```

We plot two as below. They illustrate the two types of cylinder we obtain from this configuration. One is the fourfold set through each triangular face and sloped downward through the square base. The other two are horizontal and go through a pair of opposite triangular faces.

```
Show[GraphicsArray[Map[cylinderplot[nsols[[#]], dpoints, vec, offset, True] &, {3, 5}]]];
```

We form a family of point configurations based on regular tetrahedra glued along a face in the horizontal plane, but with the upper vertex allowed to move vertically.

```
dpointslong = {{1, 0, 0}, {-1/2, Sqrt[3]/2, 0}, {-1/2, -Sqrt[3]/2, 0}, {0, 0, -1}, {0, 0, z}};
vec = {1, a, c}; offset = {0, b, d};
perps = Table[perp[dpointslong[[k]], vec, offset], {k, 5}];
exprs = Map[Numerator[Together[#.# - rsqr]] &, perps]
```

We get a Gröbner basis over the field of rational functions in the coordinate of that moving vertex.

```
gb = GroebnerBasis[exprs, {rsqr, d, b, a, c}, CoefficientDomain → RationalFunctions]
```

We reformulate the cylinder problem as follows. We seek direction vectors such that projection onto a plane normal thereto gives five points on a circle. Such directions will give rise to cylinders containing the five points; from the direction vector one can solve for the remaining cylinder parameters. As our interest is in counting solutions, without loss of generality we can fix one point at the origin, another at one unit along the $x$ axis, and a third lying in the $x\,y$ coordinate plane.

```
Clear[a, b];
normal = {a, b, 1};
spanners = Orthogonalize[NullSpace[{normal}], Dot];
points = {{0, 0, 0}, {1, 0, 0}, {x2, y2, 0}, {x3, y3, z3}, {x4, y4, z4}};

projpoint[p_, span_] := Sum[p.span[[j]] span[[j]], {j, 1, Length[span]}];

projpoints = Table[Together[projpoint[points[[j]], spanners]], {j, Length[points]}];
circle[{p1_, p2_, p3_}, normal_] := Module[{rsqr, c1, c2, c3, c, cp1, cp2, cp3, gb},
  c = {c1, c2, c3};
  cp1 = c - p1; cp2 = c - p2; cp3 = c - p3;
  polys = Append[Thread[{cp1.cp1, cp2.cp2, cp3.cp3} - rsqr], c.normal];
  gb = GroebnerBasis[polys, {c1, c2, c3, rsqr}, CoefficientDomain → RationalFunctions];
  {{c1, c2, c3}, rsqr} /. Solve[gb == 0, {c1, c2, c3, rsqr}]]
{cen, radsqr} = First[circle[Take[projpoints , 3], normal]];
vec1 = projpoints[[4]] - cen;
vec2 = projpoints[[5]] - cen;
polys = Numerator[Together[{vec1.vec1 - radsqr, vec2.vec2 - radsqr}]];
```

The first proof of theorem 1 computes a Gröbner basis for *polys* and uses it to count solutions. The code for this counting is a few dozen lines and is omitted. Similar code may be found in [29].

In the second proof of theorem 1 we find the resultant of our pair of polynomials generated above, with respect to one variable. We then look at its degree in the remaining variable as this gives an upper bound on the number of solutions.

```
res = Resultant[polys[[1]], polys[[2]], b];
Exponent[res, a]
```

In another proof of theorem 1 (presented in [28]) we look at the vanishing set at infinity. To this end we homogeniz our two cubic polynomials, find the highest degree terms, and solve for when they vanish.

```
hompolys = Expand[a*polys /. {a -> a*t, b -> b*t}] /. t^(n_.) :> w^(4 - n)/a;
initials = hompolys /. w -> 0
```

We solve for the vanishing of the initials.

```
solns = Solve[initials == 0, {a, b}]
```

One can plot the intersections of the vanishing sets for the direction parameter polynomials with real space. Intersec− tions of these two curves give real cylinders. One way to do such a plot is as below. This is used to illustrate a result in [28] concerning the case of no real cylinders.

```
p1 = ContourPlot[polys[[1]], {a, -40, 40}, {b, -40, 40}, Contours -> {0}, ContourShading → False,
    PlotPoints → 200, ContourStyle → {Thickness[.0005], Dashing[{.03, .01}], Hue[.04]}];
p2 = ContourPlot[polys[[2]], {a, -40, 40}, {b, -40, 40},
    Contours -> {0}, ContourShading → False, PlotPoints → 200,
    ContourStyle → {Thickness[.001], Dashing[{.12, .04}], Hue[.6]}, DisplayFunction → Identity];
Show[{p1, p2}, DisplayFunction → $DisplayFunction]
```

We define a utility to give a matrix of weights for exponent vectors to effect the degree reverse lexicographic term order. One can put together blocks of such orders (on subsets of variables) using `blockMatrix` below. This might be useful in computing a basis over cylinder variables and point coordinate parameters e.g. for attempting to determine cases in which we lose solutions.

```
drlMatrix[n_] := Prepend[Table[-KroneckerDelta[j + k - (n + 1)], {j, n - 1}, {k, n}], Table[1, {n}]]
blockMatrix[m1_, m2_] := Module[{l1 = Length[m1], l2 = Length[m2], mat1},
  mat1 = Transpose[Join[Transpose[m1], Table[0, {l2}, {l1}]]];
  mat2 = Transpose[Join[Table[0, {l1}, {l2}], Transpose[m2]]];
  Join[mat1, mat2]]
```

We continue to work with our set of two polynomials in the axis direction variables. We extract the coordinate parameters and build a block term order where the direction variables are weighted higher.

```
vars = {a, b};
params = Complement[Variables[polys], vars];
avars = Join[vars, params];
wmat = blockMatrix[drlMatrix[Length[vars]], drlMatrix[Length[params]]];
```

We form this Gröbner basis.

```
gb = GroebnerBasis[polys, avars, MonomialOrder→ wmat];
```

The code below finds leading terms that are pure products in one of the axis direction variables. These have coeffi− cients that are polynomials in the coordinates. We find each such leading term with its coefficient polynomial and factor it.

```
leads = First[GroebnerBasis'DistributedTermsList[gb, avars, MonomialOrder→ wmat]];
leads2 = Map[First, Map[Function[{x}, Split[x, Take[#1[[1]], 2] === Take[#2[[1]], 2] &]], leads]];
leads3 = Select[leads2, #[[1, 1, 1]] === 0 || #[[1, 1, 2]] === 0 &];
```

We now find all the square free parts of the coefficient factors.

```
pc = Map[#[[2]] * Apply[Times, avars ^ #[[1]]] &, leads3, {2}];
pc2 = Apply[Plus, pc, {1}];
fax = Map[FactorList, pc2];
leadfax = DeleteCases[Map[First, Flatten[fax, 1]], _Integer] /. {a :→ Sequence[], b :→ Sequence[]};
```