QUADPLOT:
A programme to plot quadric surfaces

Keath R Borg
and
Alex F Bielajew †
August 14, 1995

PIRS-0491

Ionizing Radiation Standards
Institute for National Measurement Standards
National Research Council of Canada
Ottawa, Canada
K1A 0R6
Tel: 613-993-2197
Fax: 613-952-9865
email: alex@irs.phy.nrc.ca

† Corresponding author.

Abstract

A two part computer programme called `QUADPLOT` is described. The first part of the programme generates plotting files in MTV plot data format for quadric surfaces. A arbitrary second order equation was taken as the input. This equation represents the surface to be plotted, and includes surfaces that are rotated and translated from one of the canonical forms. The second part of the programme plots the `HOWFAR` and `HOWNEAR` trajectories. `HOWNEAR` plots the shortest trajectory from the starting point to the surface. In the process, other trajectories which cross perpendicular to the tangent plane are found, and these trajectories are displayed also. `HOWFAR` requires one more input, a direction of travel, and plots the forwards and backwards trajectories which would intersect the quadric surface. The techniques used in the programme will be described in this report.

1 QUADPLOT: Plotting Quadric Surfaces

1.1 Overview

The main part of the programme is concerned with taking input from the user, and recognising the surface which has to be plotted. It would be easy to classify the shape if the surface ware always given in canonical form, but this condition could not be counted on. A general overview of how the surface is converted to canonical form will be given before discussing the details.

The surface to plot was expressed as an equation of degree two, as shown in equation 1:

$$f(\vec{x}) = \sum_{i,j=0}^{3} a_{ij} x_i x_j = 0. \tag{1}$$

The a_{ij}'s are arbitrary constants, and the four dimensional vectors x_i and x_j have components $(1, x, y, z)$. For more details, the reader is referred elsewhere [1]. Before describing how the input is processed, some basic terminology should be reviewed. A cross term is a term $x_i x_j$, where i is not equal to j, and neither i nor j is equal to zero. Cross terms are a result of a rotation. A linear term is a term where i or j equals zero, but both i and j do not equal zero. If a linear term and a quadratic term appear in x_j exclusively, a translation has occurred. For example, if one has $x_j^2 + 2x_j$, and j is equal to one, then there is a translation along the x axis. If j were two or three, then the translation would be in the y or z directions respectively. Finally, a_{00} is the coefficient of a constant term.

The task of plotting the surface is started by rotating and translating the given equation into an appropriate co-ordinate system. The proper rotation and translation gives the surface in canonical form. Having the equation in canonical form makes it much easier to determine what type of quadric surface is being described. (Table 1 lists the names of several quadric surfaces, and some of them are displayed in figure 1.)

After classifying the object, points are calculated which lay on the surface. These points are in the new co-ordinate system, and therefore must be translated and rotated back to the original co-ordinate system. Figure 2 gives more of the details on this process.

The output of `QUADPLOT` is an ascii file that may be read and manipulated with the MTV 3D plotting programme [2].

1.2 Preparing for Classification

`Subroutine data` is used to obtain the input. This data is then placed into a three by three matrix, named `coef`, which is similar to the one shown in figure 3. `Subroutines tred2` and `tqli` are then used to determine the eigenvectors and eigenvalues of `coef`. Both of these

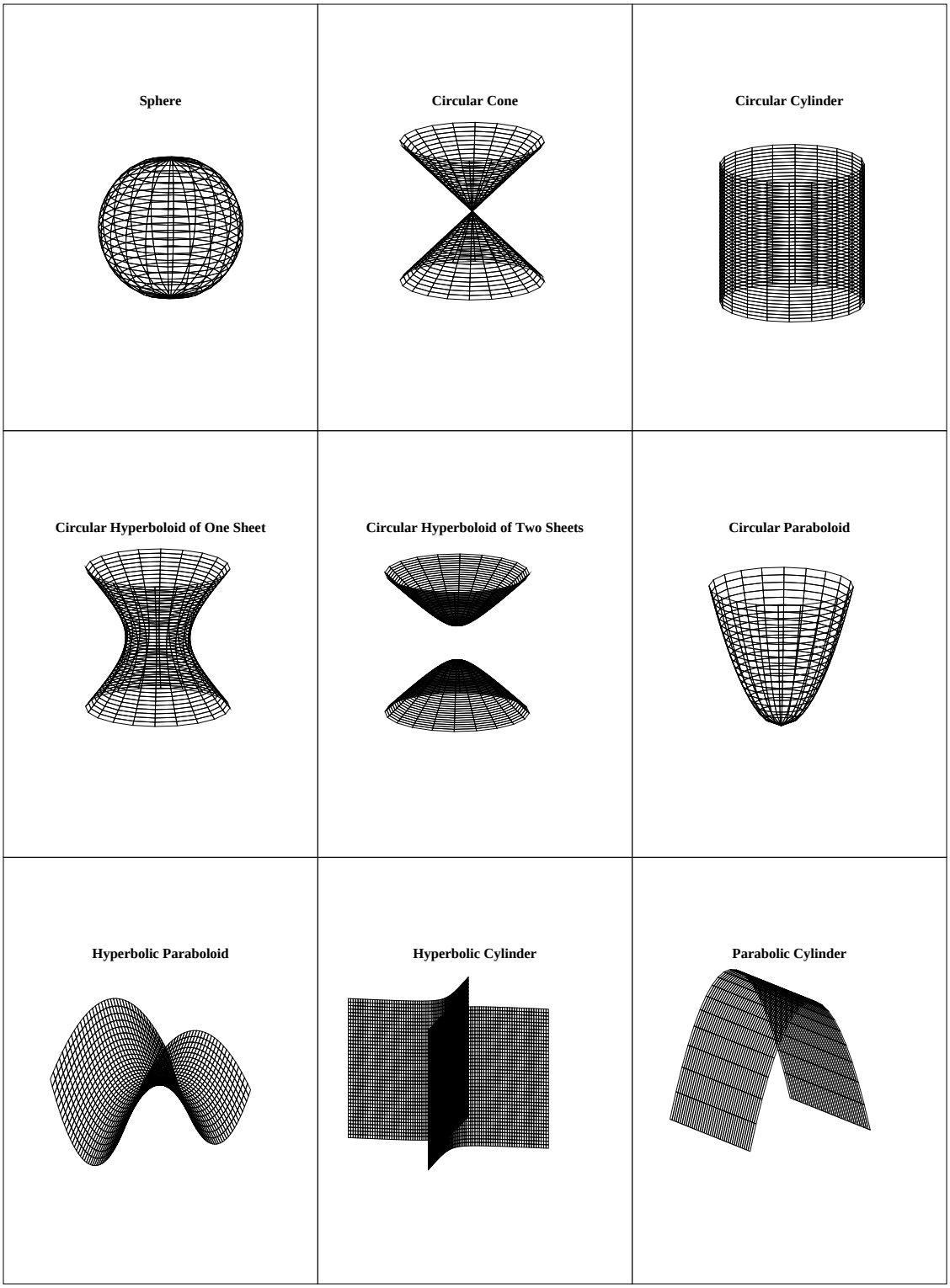

Figure 1: An illustration of some of the surfaces which were plotted.

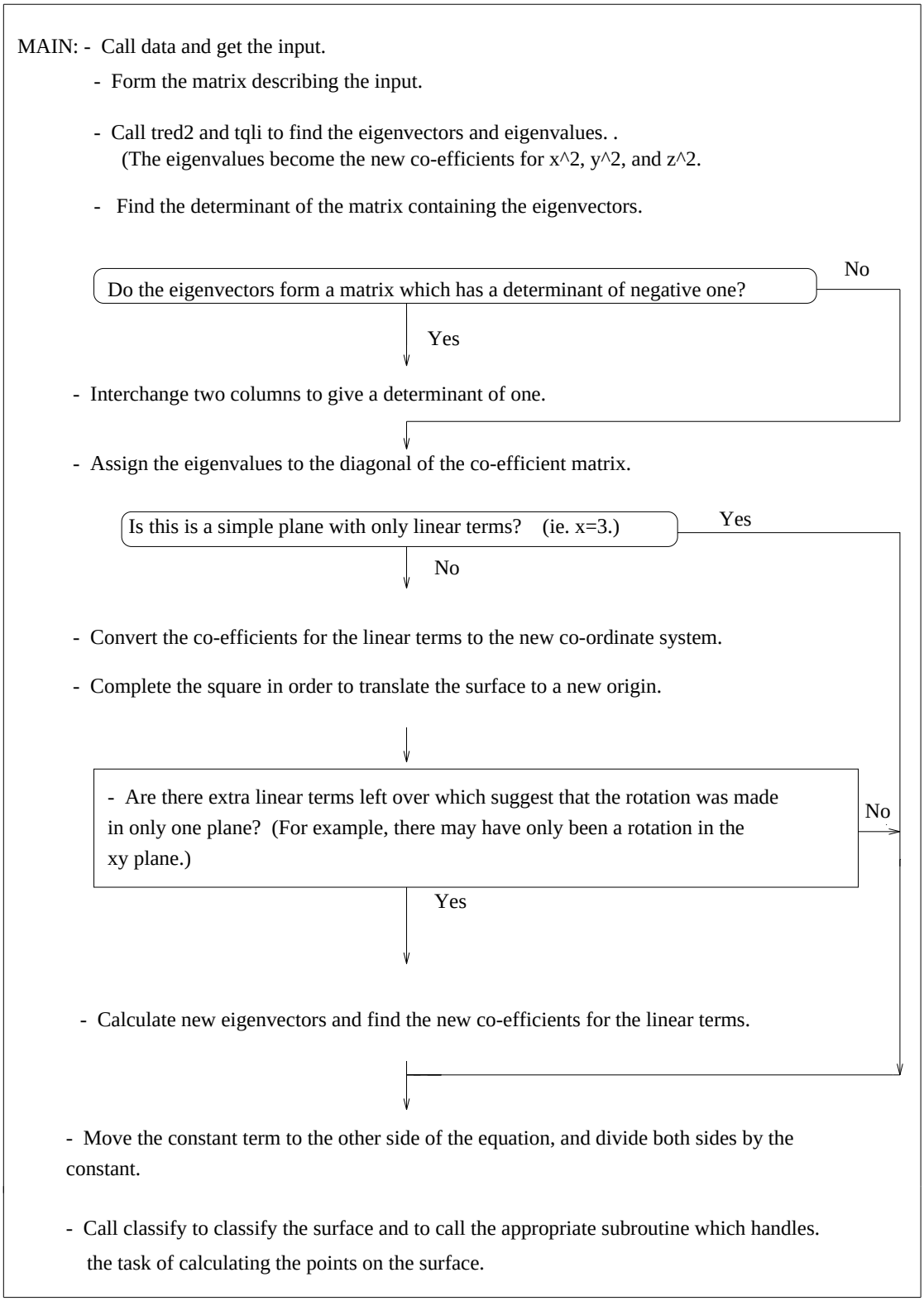

Figure 2: The algorithm used by the main programme, `QUADPLOT`.

	x	y	z
x	a	h	g
y	h	b	f
z	g	f	c

Figure 3: The format of the matrix which holds the co-efficients of the surface.

subroutines were taken from *Numerical Recipes* [3]. Since the eigenvectors are of unit length, the determinant of a matrix containing the eigenvectors could be either positive or negative. As will be explained, a positive determinant is desired. If the determinant is not positive, interchanging two of the columns gives the desired sign.

Before explaining why the positive determinant is required, some linear algebra will be explained, and some observations will be made. The matrix, `coef`, is a symmetric matrix ($\texttt{coef}^t = \texttt{coef}$). Like all symmetric matrices, `coef` could be diagonalised orthogonally. (Recall that if the matrix W is orthogonal, and W^{-1} `coef` W gives a diagonal matrix, then `coef` is diagonalisable orthogonally.) The orthogonal matrix needed to orthogonally diagonalise `coef` is a normalised basis set for the matrix `coef`. This fact means that the matrix containing the eigenvectors for `coef` is a suitable choice, since the eigenvectors are normalised already (each column in an orthogonal matrix is like a unit vector). If the determinant of this basis set is positive one, the matrix represents a rotation. Keeping in mind that a rotation is needed, it should now be apparent why the determinant of the eigenvector matrix was checked. The eigenvectors found by `subroutines tred2` and `tqli` could also be used as an axis to plot the surface. After diagonalising `coef`, the eigenvalues are left on the main diagonal, which means that the eigenvalues are the new co-efficients for quadratic terms of the form x_j^2. All other entries in the 3×3 matrix are zero, which indicates that the cross terms (such as xy) are successfully eliminated.

The case of a simple plane ($\sum_{k=0}^{3} a_k x_k = 0$) does not lend itself well to the next part of the algorithm. After diverting this special case, the linear terms have to be rotated into the new co-ordinate system. To explain the conversion of the linear terms, it may be easiest to start by explaining in more detail how the quadratic terms are converted.

The equation given by the user can be represented as:

$$x^t(\texttt{coef})x + Kx + c = 0, \tag{2}$$

where K contains the linear co-efficients, `coef` contains the quadratic co-efficients, x is a three-dimensional vector (x, y, z), and c is a constant. The next step is to let $x = Rx^p$, where x^p is a counterpart to x in the new co-ordinate system, and R is a rotation matrix. R is not any rotation matrix, though. R is actually the new co-ordinates expressed in terms of the old co-ordinate system (the eigenvectors of `coef`). Making the substitution into

equation 2, the following is obtained:

$$(Rx^p)^t(\texttt{coef})(Rx^p) + K(Rx^p) = 0. \tag{3}$$

Recalling that taking the transpose of a product reverses the order of multiplication, equation 3 can be reduced to the following form:

$$(x^p)^t(R^t)(\texttt{coef})(R)(x^p) + (KR)x^p = 0. \tag{4}$$

As explained previously, R diagonalises orthogonally `coef`. Therefore, equation 4 can be expressed in the form:

$$(x^p)^t(D)x^p + (KR)x^p = 0, \tag{5}$$

where D is a diagonal matrix. Equation 5 shows that KR gives the new linear terms. That means the new linear terms can be found by multiplying the original linear terms by the rotation matrix returned by `subroutine tqli`.

With this detail taken care of, the surface is translated to a new origin by completing the square. For example, an input of $x^2 + 4x$ gives a translation of -2 in the x direction. After completing the square in x, y, and z, the new origin is stored as $(c1, c2, c3)$.

The next step is to check for another special case. In some instances, there can be a rotation which produces no cross terms. For example, if one were dealing with a parabolic cylinder ($x = z^2$), a rotation in the xy plane produces no cross terms. Instead, there would be x and y linear terms to deal with. Since this would make identification of the surface difficult, new eigenvectors are found in the (xy) plane, as explained in appendix I. The new eigenvectors are chosen so that there is only one linear term remaining.

1.3 Classification and Plotting

Once the surface is rotated and translated into canonical form, the surface type must be recognized. The first step in recognizing the surface is to check for cases where the standard form is multiplied by negative one. In these cases, the equation is multiplied by negative one to return the expected form of the equation. With this accounted for, the number of zero and negative quadratic terms are used to help identify the surface. In addition, the linear and constant terms are used in identifying the surface. Figure 4 summarises these steps.

After identifying the surface, a closer look is taken at the equation to determine if there is symmetry about one of the axes. If, for example, one has a hyperboloid of one sheet extending along the y-axis, the equation needed to find points on this surface would be different than the equation needed to find points on the surface of a hyperboloid extending down the z-axis. To solve this problem, a single subroutine was made to find points on each type of surface. These points are passed to a matrix which rotates the calculated points into the correct orientation. For example, the subroutine which finds points on a hyperboloid of one sheet finds points for a hyperboloid with symmetry in the z-direction. If the user asks

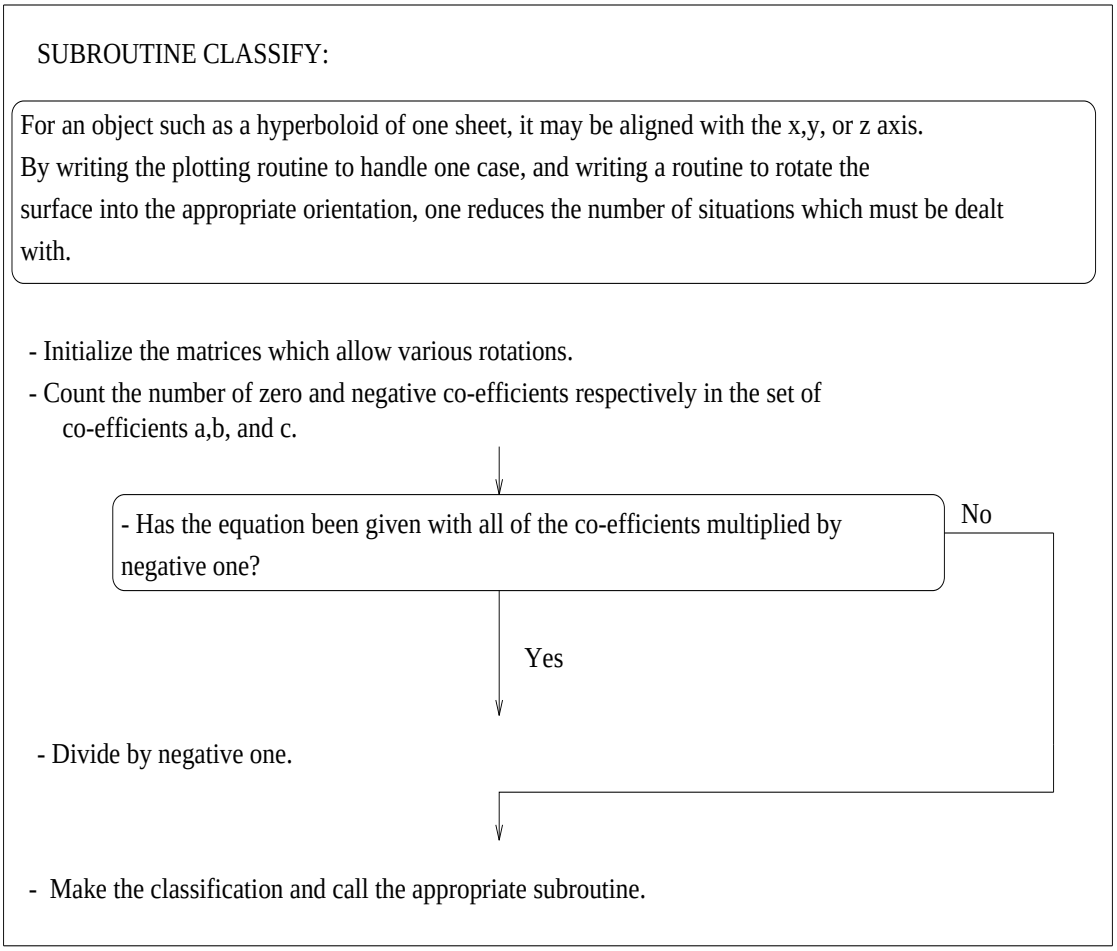

Figure 4: The method used by `subroutine classify`.

for a hyperboloid with symmetry in the y-direction, `subroutine classify` passes the next subroutine a matrix to rotate the points into the desired direction.

There is other information which is passed to the subroutine which calculates the points. This information includes the new origin $(c1, c2, c3)$, as well as a matrix which undoes the rotations used to create the new co-ordinate system. When rotating back to the basis set of eigenvectors, the rotation matrix contains the rotated eigenvectors expressed in terms of the basis set. This fact meant that the matrix R is used to rotate back into the original co-ordinates.

Before the points are actually written to a data file, another subroutine is called to query the user for more information on how to do the graph. The increment size and axis names are then determined. All of the subroutines which take care of the plotting for each type of surface use the general techniques described above. Table 1 summarises the names of the subroutines which handle each of the surfaces, and table 2 summarises the subroutines which are called. Table 3 describes some of these common subroutines.

Subroutine name	Plots quadric surface...	Example equation
`hyper`	hyperboloid of 1 sheet	$ax^2 + by^2 - cz^2 - d = 0$
`hyper`	hyperboloid of 2 sheets	$ax^2 - by^2 - cz^2 - d = 0$
`cone`	circular/elliptical cones	$ax^2 + by^2 - cz^2 = 0$
`ellipsoid`	ellipsoids/spheroids/spheres	$ax^2 + by^2 + cz^2 - d = 0$
`ecyl`	circular/elliptical cylinders	$ax^2 + by^2 - d = 0$
`hyparab`	hyperbolic paraboloid	$ax^2 - by^2 - rz = 0$
`eparab`	elliptic paraboloids	$ax^2 + by^2 - rz = 0$
`pcyl`	parabolic cylinders	$ax^2 + by = 0$
`hcyl`	hyperbolic cylinders	$ax^2 - by^2 - d = 0$
`intpl`	intersecting planes	$ax^2 - by^2 = 0$
`ppl`	parallel planes	$ax^2 - d = 0$
`ppl`	coincident planes	$ax^2 = 0$
`plane`	simple planes	$px + qy + rz - d = 0$

Table 1: The surfaces plotted by each subroutine is given. As well, an example equation is shown for each type of shape. For surfaces such as the cone, they were actually further divided into circular and elliptical objects. An elliptical cone would have $a \neq b$.

2 HOWFAR

Given an arbitrary starting position for a particle, and given a direction of travel, it can be determined if and where the particle would intersect the surface. For justification of the

Subroutine	Subroutine to get parameters	Writing subroutine
`hyparab`	`height`	`negate`
`eparab`	`height /section`	`rotflip`
`cone`	`height /section`	`rotflip`
`hyper`	`height /section`	`rotflip`
`ellipsoid`	`increment/section`	`rotflip`
`ecyl`	`maxmin /section`	`inf`
`pcyl`	`square`	`inf`
`ppl`	`maxminII`	`inf`
`hcyl`	`maxmin`	`inf`
`intpl`	`maxminII`	`inf`
`plane`	`maxminII`	N.A.

Table 2: The first column shows the various subroutines are called by `subroutine classify` to calculate points which were on various surfaces. The second column shows subroutines which are called to get information on how to do the plot, and the final column shows which subroutine is called to actually write the data to the datafile.

Subroutine	Description
`negate`	Uses symmetry to find points in the remaining three quadrants
`flip`	Flips the points across the xy plane to the negative z-side
`inf`	Will extend planes and cylinders to the desired length
`plane`	Calculates its own eigenvectors and does its own plotting

Table 3: The subroutines which are described could be accessed by any of the subroutines which calculated points on the surfaces. Each subroutine above uses a different type of symmetry to reduce the number of points which had to be calculated, but all of the subroutines perform a rotation between co-ordinate systems.

technique used, the reader is referred elsewhere [1]. It was shown that the distance to the surface can be found by solving the following equation for s, where s is the distance to the surface, p_i is a co-ordinate of the starting point, u_j is a component of the direction vector, and $a_{i,j}$ is one of the arbitrary co-efficients from equation 1.

$$s^2 \left(\sum_{i,j=0}^{3} a_{ij} u_i u_j \right) + 2s \left(\sum_{i,j=0}^{3} a_{ij} p_i u_j \right) + \left(\sum_{i,j=0}^{3} a_{ij} p_i p_j \right) = 0 \tag{6}$$

Since the distance s can be calculated, and since the direction $\vec{u}$ is known, the final point of the trajectory, $\vec{x}$, can be found using:

$$\vec{x} = s\vec{u} + \vec{p} \tag{7}$$

3 HOWNEAR

3.1 The General Method

As shown elsewhere [1], one may minimize the distance from an arbitrary point to a quadric surface using Lagrange multipliers. Depending on the surface, one may find an expression involving λ matching either the form of equation 8 or 9:

$$\sum_{i=1}^{3} a_i \left(\frac{p_i}{1 + \lambda a_i} \right)^2 + c = 0, \tag{8}$$

$$\sum_{i=1}^{2} a_i \left(\frac{p_i}{1 + \lambda a_i} \right)^2 + b \left(p_3 - \frac{\lambda b}{2} \right) = 0. \tag{9}$$

After solving for λ, one may substitute into an expression which will give the values of x_i. Equation 10 and equation 11 correspond to equations 8 and 9 respectively:

$$x_i = \frac{p_i}{1 + \lambda a_i} : \quad (i = 1, 2, 3), \tag{10}$$

$$x_i = \frac{p_i}{1 + \lambda a_i} : \quad (i = 1, 2), \qquad x_3 = \left(p_3 - \frac{\lambda b}{2} \right). \tag{11}$$

The amount of effort required to solve for λ depends on the type of surface being plotted. Table 4 summarises the order of the equation which has to be solved for each of the surfaces. In some cases, the geometry is simple enough that geometrical methods are used instead of Lagrange multipliers. In the case of fifth and sixth order equations, the `HOWNEAR` option is not calculated. It has been shown that there are no analytical methods to solve equations of order five and higher [3]. However, there are numerical methods which may be used to find the roots.

ellipsoids	6
elliptic hyperboloids of one and two sheets	6
elliptic paraboloid	5
hyperbolic paraboloid	5
spheroids	4
elliptic cones	4
elliptic cylinder	4
hyperbolic cylinder	4
circular paraboloid	3
parabolic cylinder	3
sphere	2
circular cylinder	2
parallel planes	2
circular cone	2
intersecting planes	2
simple plane	1

Table 4: A summary of the order of the equation which must be solved in order to determine λ for each of the surfaces which may be plotted. If the order is four or less, the value for λ is be determined analytically.

Before discussing how to find values for λ, a few more general ideas should be mentioned. The starting point given by the user has to be rotated and translated into the co-ordinate system describing the surface in canonical form. As described earlier, the equations are developed for surfaces in canonical form. Once a solution was found, it is translated and rotated back into the user's co-ordinate system for plotting.

3.2 `Subroutine Secondorder`: Order Two

Surfaces such as the sphere are dealt with by this subroutine. After expanding and clearing the denominators of equation 8, one can collect the quadratic (a), linear (b), and constant (c) terms. Substituting these values into the quadratic formula, the values of λ can then be determined. If one were to use the quadratic equation in the form of equation 12, there could be errors.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{12}$$

These errors arise when a or c is very small. One of the roots will then require the subtraction of a number nearly equal to b, from b. To remove this problem, the algorithm suggested in *Numerical Recipes* [3] is used. Q is evaluated, as in equation 13, and this value is then

substituted into equation 14 to obtain the roots.

$$\begin{aligned} &\text{IF } b > 0 \quad q = -\tfrac{1}{2}\left[b + \sqrt{b^2 - 4ac}\right] \\ &\text{ELSE} \qquad q = -\tfrac{1}{2}\left[b - \sqrt{b^2 - 4ac}\right] \end{aligned} \tag{13}$$

$$\text{root}_1 = \frac{q}{a} \qquad\qquad \text{root}_2 = \frac{c}{q} \tag{14}$$

3.3 `Subroutine Findcube:` Third Order

The solution to a third order equation may be three real roots, or one real root and a complex and its conjugate. Complex conjugates do not need to be dealt with in the third order case, but the reader is forewarned that the subroutine is designed to return complex conjugates. This ability is needed when solving the quartic equations, as described in the next section.

The algorithm which is used is taken from *Numerical Recipes* [3]. The intermediate values Q and R are calculated using the substitutions:

$$Q = \frac{a^2 - 3b}{9} \qquad\qquad R = \frac{2a^3 - 9ab + 27c}{54} \tag{15}$$

If R^2 is found to be less than Q^3, then the intermediate value θ can be calculated as:

$$\theta = \arccos\left(\frac{R}{\sqrt{Q^3}}\right) \tag{16}$$

Using θ and Q, the three real roots are then calculated using equation:

$$\begin{aligned} x_1 &= -2\sqrt{Q}\cos\left[\frac{\theta}{3}\right] - \frac{a}{3} \\ x_2 &= -2\sqrt{Q}\cos\left[\frac{\theta + 2\pi}{3}\right] - \frac{a}{3} \\ x_3 &= -2\sqrt{Q}\cos\left[\frac{\theta - 2\pi}{3}\right] - \frac{a}{3} \end{aligned} \tag{17}$$

If R^2 is not less than Q^3, the intermediate values a_2 and b_2 are calculated using:

$$\begin{aligned} a_2 &= \pm\left[|R| + \sqrt{R^2 - Q^3}\right]^{\frac{1}{3}} \\ b_2 &= Q/A \quad (a_2 \neq 0) \\ b_2 &= 0 \qquad (a_2 = 0) \end{aligned} \tag{18}$$

In the above equation, the positive option is chosen if R is greater than zero, and the negative option is chosen if R is less than zero. a_2 and b_2 are then used to find the cube roots:

$$\begin{aligned}
\texttt{root}_1 &= (a_2 + b_2) - \frac{a}{3} \\
\texttt{root}_2 &= -\frac{a_2 + b_2}{2} - \frac{a_2}{3} + i\frac{\sqrt{3}}{2}(a_2 - b_2) \\
\texttt{root}_3 &= -\frac{a_2 + b_2}{2} - \frac{a_2}{3} - i\frac{\sqrt{3}}{2}(a_2 - b_2)
\end{aligned} \tag{19}$$

As stated earlier, the complex numbers are used during the solution of the quartic equation. When there are complex solutions, the real part is stored as root two, and the imaginary part was stored as root three. The number of real roots is returned so that the complex roots can not be mistaken for real roots.

3.4 `Subroutine Quartic`: Fourth Order

The technique used to solve the quartic equation comes from Korn and Korn's text [4]. To find the solution of a quartic equation with the form $x^4 + ax^3 + bx^2 + cx + d = 0$, the substitution $x = y + a/4$ is made to eliminate the x^3 term. The new co-efficients p, q, and r are calculated to give a quartic of the form:

$$y^4 + py^2 + qy + r = 0. \tag{20}$$

The variables p, q, and r are then used to calculate co-efficients for the cubic resolvent. The cubic resolvent is:

$$z^3 + \frac{p}{2}z^2 + \frac{p^2 - 4r}{16}z - \frac{q^2}{64} = 0, \tag{21}$$

and it can be thought of as the discriminant's third order counterpart.

The roots to the cubic resolvent are found as described in the previous section. If there are three positive real roots to the cubic resolvent, then four real roots can be calculated for the reduced quartic equation. One real root and a complex conjugate gives two real roots for the quartic. The roots are calculated to be:

$$y_i = \pm\sqrt{z_1} \pm \sqrt{z_2} \pm \sqrt{z_3}, \tag{22}$$

where the z_i represent the roots of the cubic resolvent. Since the complex numbers which could potentially arise from the cubic resolvent would be complex conjugates, the imaginary parts would cancel out during addition. The signs of the square roots are chosen so that the product of the three square roots would be the opposite sign of the q term. This restriction on the sign leaves either two or four combinations for adding and subtracting the terms, depending on whether a complex root is found for the cubic resolvent. To find the roots for the quartic equation with the cubic term, the original quartic equation, the substitution $x_i = y_i + \frac{a}{4}$ is made. As before, the y_i represent the roots of the simplified quartic.

3.5 What to Do with the Roots?

Regardless of whether the equation is second, third, or fourth order, the same technique is used once the roots are found. The potential point of intersection is calculated by substituting the value of λ into either equation 10 or 11. Since Lagrange multipliers give the extrema, in this case either the farthest point or the closest point, all the points are tested to find which one is at the minimum distance. This test is implemented by calculating the distance from the starting point to each potential point of intersection. As each new point is determined, it is stored in an array. After all of the points are calculated, the closest point is pulled out of the array, and is rotated and translated back into the co-ordinates of the user. The other points of intersection are also displayed; however, dotted lines are used to show the trajectory is not a minimum. Figure 5 gives an example of a typical `HOWNEAR` solution.

3.6 Special Cases

If the starting point is on an axis of symmetry or a plane of symmetry, there is no unique solution for λ. Two examples would be starting the particle on the z axis, or on the xy plane, of a cone that has its axis of symmetry in the z-direction. In either case the third term in equation 8 could be the only negative term, and the constant c would be zero. Starting on the xy plane would remove the negative term, and a solution would not be possible. If the particle were to start on the z axis, then only the third term would be non-zero, and once again there would be no solution for λ.

Starting a particle on a plane or axis of symmetry means that there can be more than one solution to `HOWNEAR`. In order to get a single solution, the particle is displaced by a small amount. This allowed one of the solutions to be found, even though there are multiple solutions.

4 Using Geometrical Methods

4.1 One or more Planes

A geometrical approach was used when plotting figures involving planes. In the case of coincident planes, intersecting planes, and parallel planes, the same basic technique is used. The method is summarised below:

$$\vec{P} = \vec{P}_0 - [\vec{n} \cdot \vec{p}]\, \vec{n} \tag{23}$$

where $\vec{P}$ is the final point, $\vec{P}_0$ is the starting point, $\vec{n}$ is a unit normal vector to the plane, and $\vec{p}$ is the position of the starting point with respect to a point on the plane. The magnitude of the dot product is the perpendicular distance to the surface. Subtracting a vector in the

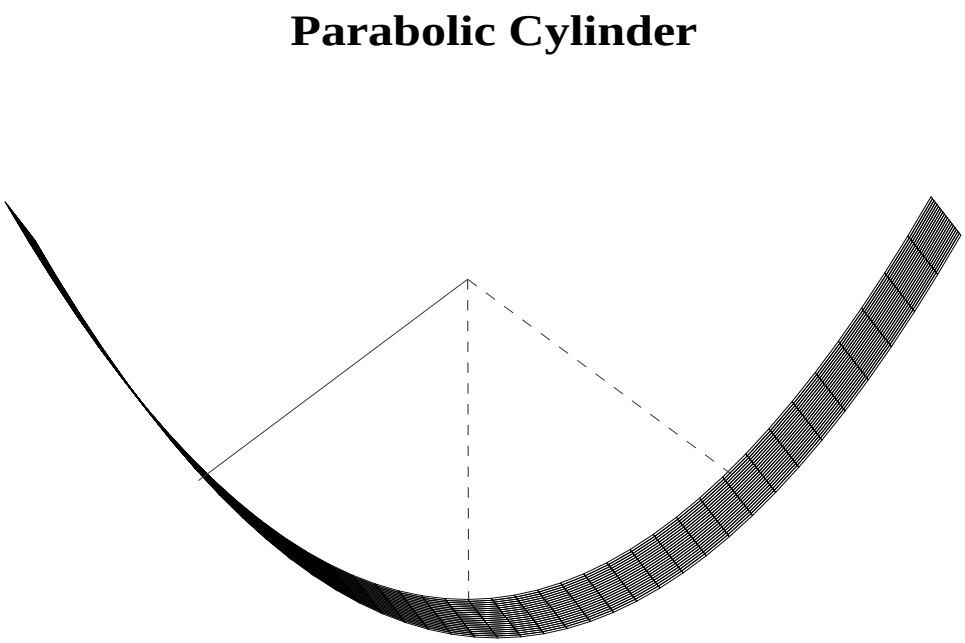

Figure 5: This figure shows an example of a `HOWNEAR` solution which uses the general technique of Lagrange multipliers. The solid line represents the smallest trajectory. All three trajectories are perpendicular to the surface. The starting point is located on an axis of symmetry, which introduces some problems as described in the next section. The method used to fix the problem prevents the two side trajectories from being displayed as solid lines.

direction of the normal with this magnitude gives the point of intersection. Figure 6 gives a typical example of the `HOWNEAR` solution found using geometrical methods.

4.2 `Plane`: The Simple Plane

This is the subroutine which plots planes of the form $(ax + by + cz - d = 0)$. Since this situation is a little bit different than the others, it is handled separately. The starting point is first moved into the new co-ordinate system. Because of the way the new axes are oriented, the x-component is the component perpendicular to the plane. The `HOWNEAR` solution gives trajectories perpendicular to the surface, so solving for the point of intersection is simple. The x-component of the starting point is set to zero, and this new point is translated and rotated back into the user's co-ordinate system. Figure 7 gives an example of a solution to `HOWNEAR` for a simple plane.

5 Conclusions

All of the real quadric surfaces described by Olmsted [5] may be plotted. Since there is also an option to rotate and translate surfaces, the user has some choice for the geometry. When Monte Carlo simulations are being developed to model complex surfaces, however, combinations of simple surfaces are used. To be more compatible with the Monte Carlo codes, the programme developed during the work term would ideally allow the user to combine various shapes. Despite the large number of surfaces which may be modeled by combining shapes, four dimensional surfaces, such as the torus, cannot be modeled easily.

Improvements could be made to the existing code, even without going to four dimensions. For surfaces which have `HOWNEAR` equations of degree five or more, numerical methods could be used to solve for λ. Once λ has been found, each potential point would be tested to find the point at the minimum distance. Adding the numerical methods would allow `HOWNEAR`to be calculated for all of the surfaces.

Acknowledgement

One of us (AFB) would like to thank Kenny Toh of the Intel Corporation for his rapid modifications to the MTV package that were required in the drafting of this report.

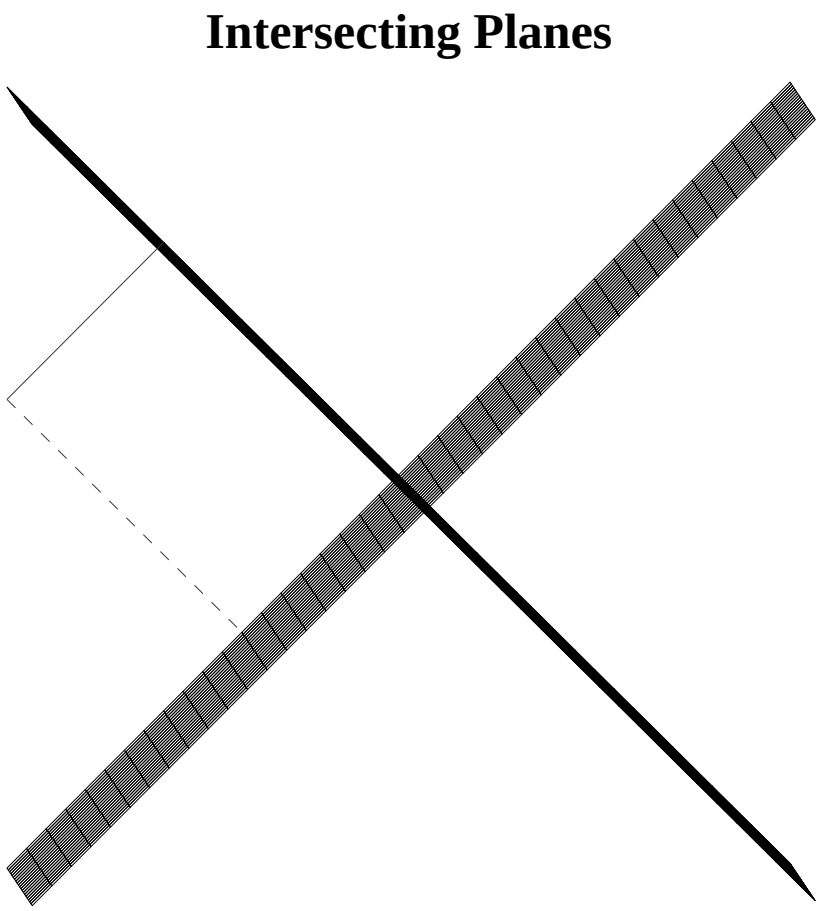

Figure 6: This figure shows an example solution to `HOWNEAR` in the case where one or more planes are involved. In these cases, a geometrical solution as used.

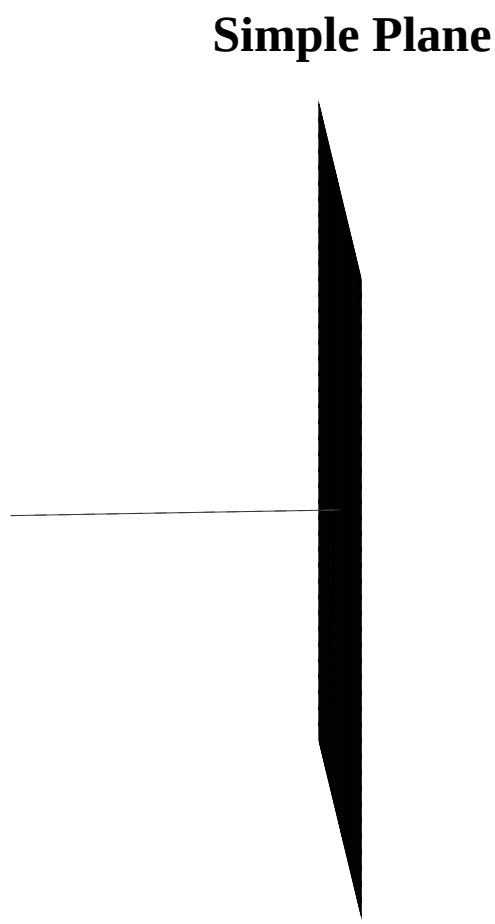

Figure 7: An example of a solution to `HOWNEAR` in the case of a simple plane ($ax+by+cz-d=0$). The simple plane is treated differently than the other planes.

References

[1] A.F. Bielajew, "HOWFAR and HOWNEAR: Geometry Modeling for Monte Carlo Particle Transport," National Research Council of Canada Report PIRS-0341 (1995).

[2] K. Toh, "MTV Plot Data Format Version 1.4.0," Technology CAD, Intel Corporation, Santa Clara, California (unpublished report) (1994).

[3] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, "Numerical Recipes—The art of scientific computing," (Cambridge University Press, Cambridge) (1986).

[4] G.A. Korn and T.M. Korn, "Mathematical Handbook for Scientists and Engineers," (McGraw-Hill, Toronto) (1961).

[5] J.M.H. Olmsted, "Solid Analytic Geometry," (Appleton-Century-Crofts Inc, New York) (1947).

Appendix: Selecting eigenvectors

In cases where there is a rotation but linear terms are created rather than cross terms, new eigenvectors have to be chosen. The explanation will be based on a rotation in the xy plane, but the technique could be modified to suite a rotation in any plane.

If there is a rotation in the xy-plane, then one would have px and qy as linear terms. The new eigenvectors can be calculated using:

$$\begin{aligned} \vec{l} &= \frac{p\vec{l} + q\vec{k}}{\sqrt{p^2+q^2}} \\ \vec{k_1} &= -\vec{l_2} \\ \vec{k_2} &= \vec{l_1} \\ \vec{k_3} &= \vec{k_3} \end{aligned} \tag{24}$$

where $\vec{l}$ is the x-eigenvector, and $\vec{k}$ is the y-eigenvector. The new value for $\vec{l}$ is the resultant of adding the original eigenvectors and accounting for the magnitude in each direction. $\sqrt{p^2+q^2}$ is used to ensure that $\vec{l}$ remains a unit vector. The adjustments to the components of $\vec{k}$ are made to ensure that it is perpendicular to the new x-eigenvector.