Intersection of Ellipses

David Eberly
Magic Software, Inc.
http://www.magic-software.com
Created: October 10, 2000

1 Introduction

This article describes how to compute the points of intersection of two ellipses, a geometric query labeled find intersections. It also shows how to determine if two ellipses intersect without computing the points of intersection, a geometric query labeled test intersection. Specifically, the geometric queries for the ellipses $E_0$ and $E_1$ are:

- **Find Intersections.** If $E_0$ and $E_1$ intersect, find the points of intersection.
- **Test Intersection.** Determine if
  - $E_0$ and $E_1$ are separated (there exists a line for which the ellipses are on opposite sides),
  - $E_0$ properly contains $E_1$ or $E_1$ properly contains $E_0$, or
  - $E_0$ and $E_1$ intersect.

An implementation of the find query, in the event of no intersections, might not necessarily determine if one ellipse is contained in the other or if the two ellipses are separated. Let the ellipses $E_i$ be defined by the quadratic equations

$$Q_i(X) = X^T A_i X + B_i^T X + C_i$$

for $i = 0, 1$. It is assumed that the $A_i$ are positive definite. In this case, $Q_i(X) < 0$ defines the inside of the ellipse and $Q_i(X) > 0$ defines the outside.

2 Find Intersection

The two polynomials $f(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2$ and $g(x) = \beta_0 + \beta_1 x + \beta_2 x^2$ have a common root if and only if the Bézout determinant is zero, $$(\alpha_2 \beta_1 - \alpha_1 \beta_2)(\alpha_1 \beta_0 - \alpha_0 \beta_1) - (\alpha_2 \beta_0 - \alpha_0 \beta_2)^2 = 0.$$
This is constructed by the combinations

\[ 0 = \alpha_2 g(x) - \beta_2 f(x) = (\alpha_2 \beta_1 - \alpha_1 \beta_2)x + (\alpha_2 \beta_0 - \alpha_0 \beta_2) \]

and

\[ 0 = \beta_1 f(x) - \alpha_1 g(x) = (\alpha_2 \beta_1 - \alpha_1 \beta_2)x^2 + (\alpha_0 \beta_1 - \alpha_1 \beta_0), \]

solving the first equation for \( x \) and substituting it into the second equation. When the Bézout determinant is zero, the common root of \( f(x) \) and \( g(x) \) is

\[ \bar{x} = \frac{\alpha_2 \beta_0 - \alpha_0 \beta_2}{\alpha_1 \beta_2 - \alpha_2 \beta_1}. \]

The common root to \( f(x) = 0 \) and \( g(x) = 0 \) is obtained from the linear equation \( \alpha_2 g(x) - \beta_2 f(x) = 0 \) by solving for \( x \).

The ellipse equations can be written as quadratics in \( x \) whose coefficients are polynomials in \( y \),

\[ Q_i(x, y) = \left( a_{11}^{(i)} y^2 + b_1^{(i)} y + c^{(i)} \right) + \left( 2a_{01}^{(i)} y + b_0^{(i)} \right) x + \left( a_{00}^{(i)} \right) x^2. \]

Using the notation of the previous paragraph with \( f \) corresponding to \( Q_0 \) and \( g \) corresponding to \( Q_1 \),

\[ \alpha_0 = a_{11}^{(0)} y^2 + b_1^{(0)} y + c^{(0)}, \quad \alpha_1 = 2a_{01}^{(0)} y + b_0^{(0)}, \quad \alpha_2 = a_{00}^{(0)}, \]

\[ \beta_0 = a_{11}^{(1)} y^2 + b_1^{(1)} y + c^{(1)}, \quad \beta_1 = 2a_{01}^{(1)} y + b_0^{(1)}, \quad \beta_2 = a_{00}^{(1)}. \]

The Bézout determinant is a quartic polynomial \( R(y) = u_0 + u_1 y + u_2 y^2 + u_3 y^3 + u_4 y^4 \) where

\[
egin{align*}
  u_0 & = v_2 v_{10} - v_4^2 \\
  u_1 & = v_0 v_{10} + v_2 (v_7 + v_9) - 2v_3 v_4 \\
  u_2 & = v_0 (v_7 + v_9) + v_2 (v_6 - v_8) - v_3^2 - 2v_1 v_4 \\
  u_3 & = v_0 (v_6 - v_8) + v_2 v_5 - 2v_1 v_3 \\
  u_4 & = v_0 v_5 - v_7^2
\end{align*}
\]

with

\[
egin{align*}
  v_0 & = 2 \left( a_{00}^{(0)} a_{01}^{(1)} - a_{00}^{(1)} a_{01}^{(0)} \right) \\
  v_1 & = a_{00}^{(0)} a_{11}^{(0)} - a_{00}^{(1)} a_{11}^{(0)} \\
  v_2 & = a_{00}^{(0)} b_0^{(0)} - a_{00}^{(1)} b_0^{(0)} \\
  v_3 & = a_{00}^{(0)} b_1^{(0)} - a_{00}^{(1)} b_1^{(0)} \\
  v_4 & = a_{00}^{(0)} c^{(1)} - a_{00}^{(1)} c^{(0)} \\
  v_5 & = 2 \left( a_{01}^{(0)} a_{11}^{(1)} - a_{01}^{(1)} a_{11}^{(0)} \right) \\
  v_6 & = 2 \left( a_{01}^{(0)} b_1^{(1)} - a_{01}^{(1)} b_1^{(0)} \right) \\
  v_7 & = 2 \left( a_{01}^{(0)} c^{(1)} - a_{01}^{(1)} c^{(0)} \right) \\
  v_8 & = a_{11}^{(0)} b_0^{(1)} - a_{11}^{(1)} b_0^{(0)} \\
  v_9 & = b_0^{(0)} b_1^{(1)} - b_0^{(1)} b_1^{(0)} \\
  v_{10} & = b_0^{(0)} c^{(1)} - b_0^{(1)} c^{(0)}
\end{align*}
\]
For each \( \bar{y} \) solving \( R(\bar{y}) = 0 \) solve \( Q_0(x, \bar{y}) = 0 \) for up to two values \( \bar{x} \). Eliminate any false solution \((\bar{x}, \bar{y})\) by verifying that \( P_i(\bar{x}, \bar{y}) = 0 \) for \( i = 0, 1 \).

3 Test Intersection

3.1 Variation 1

All level curves defined by \( Q_0(x,y) = \lambda \) are ellipses, except for the minimum (negative) value \( \lambda \) for which the equation defines a single point, the center of every level curve ellipse. The ellipse defined by \( Q_1(x,y) = 0 \) is a curve that generally intersects many level curves of \( Q_0 \). The problem is to find the minimum level value \( \lambda_0 \) and maximum level value \( \lambda_1 \) attained by any \((x,y)\) on the ellipse \( E_1 \). If \( \lambda_1 < 0 \), then \( E_1 \) is properly contained in \( E_0 \). If \( \lambda_0 > 0 \), then \( E_0 \) and \( E_1 \) are separated. Otherwise, \( 0 \in [\lambda_0, \lambda_1] \) and the two ellipses intersect.

This can be formulated as a constrained minimization that can be solved by the method of Lagrange multipliers: Minimize \( Q_0(\bar{X}) \) subject to the constraint \( Q_1(\bar{X}) = 0 \). Define \( F(\bar{X}, t) = Q_0(\bar{X}) + tQ_1(\bar{X}) \). Differentiating yields \( \nabla F = \nabla Q_0 + t\nabla Q_1 \) where the gradient indicates the derivatives in \( \bar{X} \). Also, \( \partial F/\partial t = Q_1 \).

Setting the \( t \)-derivative equal to zero reproduces the constraint \( Q_1 = 0 \). Setting the \( \bar{X} \)-derivative equal to zero yields \( \nabla Q_0 + t\nabla Q_1 = 0 \) for some \( t \). Geometrically this means that the gradients are parallel.

Note that \( \nabla Q_i = 2A_i\bar{X} + B_i \), so

\[
0 = \nabla Q_0 + t\nabla Q_1 = 2(A_0 + tA_1)\bar{X} + (B_0 + tB_1).
\]

Formally solving for \( \bar{X} \) yields

\[
\bar{X} = -(A_0 + tA_1)^{-1}(B_0 + tB_1)/2 = \frac{1}{\delta(t)}\bar{Y}(t)
\]

where \( \delta(t) \) is the determinant of \( (A_0 + tA_1) \), a quadratic polynomial in \( t \), and \( \bar{Y}(t) \) has components quadratic in \( t \). Replacing this in \( Q_1(\bar{X}) = 0 \) yields

\[
\bar{Y}(t)^T A_1 \bar{Y}(t) + \delta(t)B_0^T \bar{Y}(t) + \delta(t)^2C_1 = 0,
\]

a quartic polynomial in \( t \). The roots can be computed, the corresponding values of \( \bar{X} \) computed, and \( Q_0(\bar{X}) \) evaluated. The minimum and maximum values are stored as \( \lambda_0 \) and \( \lambda_1 \), and the earlier comparisons with zero are applied.

This method leads to a quartic polynomial, just as the \textit{find} query did. But this query does answer questions about the relative positions of the ellipses (separated or proper containment) when the \textit{find} query indicates that there is no intersection.

3.2 Variation 2

A less expensive \textit{test} query is based on the \textit{find} query, but cannot answer the question of proper containment or separation when there is no intersection. Rather than solve the quartic equation \( R(y) = 0 \) that was derived in the section on finding intersections, it is enough to determine if \( R(y) \) has any real roots. The
ellipses intersect if and only if there are real roots. If \( u_4 = 0 \) and \( u_3 = 0 \), then there are real roots as long as \( u_1^2 - 4u_0u_2 \geq 0 \). If \( u_4 = 0 \) and \( u_3 \neq 0 \), then the cubic polynomial necessarily has a real root. If \( u_4 \neq 0 \), then multiply the equation, if necessary, by \(-1\) to make the leading coefficient positive. The polynomial has no real roots if and only if \( R(y) > 0 \) for all \( y \). It is enough to compute the local minima of \( R \) and show they are all positive. This requires finding the roots of the cubic polynomial \( R(y) = 0 \) and evaluating \( R(y) \) and testing if it is positive at those roots.

But it is even possible to avoid finding roots whatsoever. This uses the method of bounding roots by Sturm sequences. Consider a polynomial \( f(t) \) defined on interval \([a, b]\). A Sturm sequence for \( f \) is a set of polynomials \( f_i(t), 0 \leq i \leq m \) such that \( \text{Degree}(f_{i+1}) > \text{Degree}(f_i) \) and the number of distinct real roots for \( f \) in \([a, b]\) is \( N = s(a) - s(b) \) where \( s(a) \) is the number of sign changes of \( f_0(a), \ldots, f_m(a) \) and \( s(b) \) is the number of sign changes of \( f_1(b), \ldots, f_m(b) \). The total number of real–valued roots of \( f \) on \( \mathbb{R} \) is \( s(-\infty) - s(\infty) \). It is not always the case that \( m = \text{Degree}(f) \). The classic Sturm sequence is \( f_0(t) = f(t), f_1(t) = f'(t), \) and \( f_i(t) = -\text{Remainder}(f_{i-2}/f_{i-1}) \) for \( i \geq 2 \). The polynomials are generated by this method until the remainder term is a constant. This method is applied to \( R(y) \) on \((-\infty, \infty)\) to determine the number of real roots.

### 3.3 Variation 3

This test is similar to variation 2, but it requires that one of the ellipses be axis–aligned (let it be \( E_0 \) for the argument). It is possible to force this to happen by an affine change of variables, the correct transformation requiring determining the eigenvalues of \( A_0 \), an operation that involves solving a quadratic equation. If the application already knows the axes of the ellipses, then this only reduces the computation time. I believe this argument also shows that \( R(y) \) can never be cubic, only quadratic or quartic.

The quadratic equation for the axis–aligned ellipse can be written as \((y - y_0)^2 = a_0 + a_1x + a_2x^2\) where \( a_2 < 0 \). The other ellipse equation can be written as \((y - y_0)^2 + (b_{10} + b_{11}x)(y - y_0) + (b_{00} + b_{01}x + b_{02}x^2) = 0\). Substituting \((y - y_0)^2\) from the first equation into the second one, solving the second for \((y - y_0)\), replacing it in the first, and cross–multiplying leads to the polynomial \( P(x) = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 \) where

\[
\begin{align*}
c_0 &= \(b_{00} + a_0\)^2 - a_0b_{10}^2 \\
c_1 &= 2(b_{00} + a_0)(b_{01} + a_1) - 2a_0b_{10}b_{11} - a_1b_{11}^2 \\
c_2 &= 2(b_{00} + a_0)(b_{02} + a_2) + (b_{01} + a_1)^2 - a_0b_{11}^2 - a_2b_{10}^2 - 2a_1b_{10}b_{11} \\
c_3 &= 2(b_{01} + a_1)(b_{02} + a_2) - a_1b_{11}^2 - 2a_2b_{10}b_{11} \\
c_4 &= (b_{02} + a_2)^2 - a_2b_{11}^2
\end{align*}
\]

Since \( a_2 < 0 \), the only way \( c_4 = 0 \) is if \( a_2 = -b_{02} \) and \( b_{11} = 0 \). In this case, \( c_3 \) is forced. If both \( c_4 = c_3 = 0 \), then \( c_2 = (b_{01} + a_1)^2 - a_2b_{10}^2 \). The only way \( c_2 = 0 \) is if \( a_1 = -b_{01} \) and \( b_{10} = 0 \). In this case, \( c_1 = 0 \) is forced and the polynomial is \( c_0 = 0 \), finally leading to \( P(x) \) being identically zero. The two quadratic equations are for the same ellipse. So the only three cases to trap in the code are via the Boolean short circuit, \( c_4 \neq 0 \) or \( c_2 \neq 0 \) or ellipses are the same. The hard case is \( c_4 \neq 0 \), but as in variation 2, it is enough just to argue whether or not \( P(y) \) has roots. This only requires solving a cubic polynomial equation \( P'(y) = 0 \) and testing the values of \( P(y) \).

The method of Sturm sequences, as shown in variation 2, can also be applied here for the fastest possible test query.