A streaming algorithm for computing an approximate minimum spanning ellipse

Asish Mukhopadhyay∗† Eugene Greene∗‡

1 Introduction

Zarrabi-Zadeh and Chan [3] proposed a simple algorithm for computing an approximate minimum spanning ball of a set of \( n \) points \( S = \{ s_1, ..., s_n \} \) in the streaming model of computation. Each new ball in their algorithm is the smallest that contains the new point and the previous ball. Though the algorithm is simple, they came up with a very clever and elegant analysis to show that the approximation ratio is \( 3/2 \) (in terms of radius). The algorithm uses only \( O(1) \) space and time for each new point. Inspired by the above paper, in this note we investigate the problem of extending their algorithm to compute an approximate ellipse in the same model.

The problem of computing a minimum area spanning ellipse of a planar set of points has been around for a very long time. Dyer [1] proposed an \( O(1) \) space deterministic algorithm for this (indeed in \( O(n) \) time for any fixed dimension \( D \)). Welzl [2] proposed an \( O(n) \) expected time algorithm. In this note we show how to find an approximate minimum ellipse spanning \( S \), with an approximation factor of \( \frac{\sqrt{3}}{\pi} \), where \( \varepsilon \) is the eccentricity of the minimum spanning ellipse of \( S \), and \( k \) is a constant defined later in the text. The algorithm similarly uses only \( O(1) \) space and time for each new point.

The following symbols will be used to denote the geometry of an ellipse \( E \) (see Figure 1): \( E \) will have center \( p_0 = (x_0, y_0) \); \( r_1 \) is half the length of the axis corresponding to \( \phi \), and \( r_2 \) is half the length of the other axis (\( r_1 \) can correspond to the minor or major axis); \( \phi \) is the angle corresponding to the direction of \( r_1 \).

2 Our solution

When the algorithm receives its first point, the approximate minimum ellipse is just that point. We will only consider subsequent points that are not already in the current approximate ellipse. For the second point, the approximation becomes a segment. As long as subsequent points are on the supporting line of this segment, the approximate ellipse will be a segment. The next point not lying on the supporting line produces a non-degenerate ellipse defined by the new point and the endpoints of the current segment. The approximate ellipse is thus equal to the exact ellipse up to this moment.

Assuming that the \( i \)th ellipse \( E_i \) encloses all of \( S_i = \{ s_1, ..., s_i \} \), then at the \( (i + 1) \)th iteration, we approximate the minimum ellipse by the smallest ellipse \( E_{i+1} \) enclosing both \( E_i \) and \( s_{i+1} \). Given non-degenerate ellipse \( E_i \), we can derive a transformation \( T_i \) that maps \( E_i \) to the unit circle. \( T_i \) is simply a translation, rotation, and scaling. We then add another rotation to \( T_i \), so that

\[
\begin{align*}
\theta_{i+1} &= T_i(s_{i+1}) \quad \text{is the positive } x\text{-axis.}
\end{align*}
\]

Now let \( E_{i+1}' \) be the smallest ellipse that contains the unit circle and \( s_{i+1} \). Then

\[
\begin{align*}
E_{i+1}' &= T_i^{-1}(E_{i+1}) \quad \text{is the smallest ellipse that contains } E_i \text{ and } s_{i+1}.
\end{align*}
\]

The problem of finding the smallest ellipse \( E_{i+1}' \) containing an ellipse \( E_i \) and a point \( s_{i+1} \) is thus reduced to that of finding the smallest ellipse \( E_{i+1}' \) containing the unit circle and a point \( s_{i+1} \) on the positive \( x \)-axis. \( E_{i+1}' \) must have an axis on the \( x \)-axis.

Consider any ellipse \( C_{i+1}' \), containing and tangent to the unit circle, and passing through \( s_{i+1}' \), that is aligned with the \( x \)-axis. (See Figure 2.) Let \( \alpha \) be the angle made by the \( x \)-axis, the origin, and the point at which \( C_{i+1}' \) touches the upper half of the unit circle. It turns out that \( E_{i+1}' \) is described by

\[
\begin{align*}
\beta &= \cos \alpha = \frac{d-\sqrt{d^2-4c}}{2} \\
x_0 &= \frac{d^2-1}{2d-\beta-\beta} \\
a &= \frac{1}{(\beta-x_0)(\beta-x_0)} \\
c &= \frac{1}{1-x_0}.
\end{align*}
\]
It is then a simple matter to transform this ellipse using $T_i^{-1}$.

\[ C_i' + 1 \alpha s_i' + 1 (\cos \alpha, \sin \alpha) \]

Figure 2: An ellipse containing and tangent to the unit circle, and passing through $s_i' + 1$

\[ \frac{9k}{4\sqrt{1-e^2}} \text{MinEllipse}. \] (1)

If we set $k = 6$, we get $\frac{9k}{4\sqrt{1-e^2}} \text{MinEllipse}$.

Another potential route to finding an approximation ratio uses the following idea: Assuming that the exact minimum ellipse is the unit circle, how can we construct a corresponding sequence of points that will make the approximate ellipse as large as possible? It turns out that if such a sequence of points exists, then the ratio of the area of the corresponding approximate ellipse to the that of the unit circle will be the approximation ratio of the algorithm.

We observe that in order to construct the biggest $E_i' + 1$ from some $E_i$, the point $s_i + 1$ should be on the supporting line of $E_i$’s minor axis. We can use this observation to attempt to create a very bad approximate ellipse. One example of a bad point set is shown in Figure 3. The algorithm finds the left and right points first, then finds the points going from the origin to the top of the circle, and finally the points from the origin to the bottom of the circle. As we increase the number of points on the $y$-axis, the ratios of the areas of the approximate and exact ellipses get bigger, and they appear to converge. See Figure 4.

3 Approximation ratio

Suppose $\text{ApproxEllipse} \leq k \text{ApproxDisk}$ (in terms of area), where $\text{ApproxEllipse}$ is the ellipse generated by our algorithm, $\text{ApproxDisk}$ is the circle generated by the algorithm of [3], and $k$ is a constant. We have not been able to establish the value of $k$ exactly, but extensive experiments indicate that $k$ lies between 5 and 6. We know from [3] that $\text{ApproxDisk} \leq \frac{9}{4 \text{MinDisk}}$. Now, $\text{MinDisk} \leq \text{MinEllipse}/\sqrt{1-e^2}$, where $e$ is the eccentricity of the minimum spanning ellipse of the point set. Thus, combining the above results we obtain the following approximation ratio:

\[ \frac{\text{ApproxEllipse}}{\text{MinEllipse}} \leq \frac{9k}{4\sqrt{1-e^2}}. \]

4 Higher dimensions

We can transform a non-degenerate $D$-dimensional ellipse $E_i$ into the $D$-dimensional unit ball, and rotate so that the new point $s_{i+1}$ is on the positive $x_1$-axis. Now the new ellipse $E_{i+1}'$ will have to be symmetric with respect to the $x_1$-axis. We use the same values of $x_0$, $a$, and $c$. Each new point would require at most $O(D^3)$ space and time.

5 Conclusions

We have presented a streaming algorithm for computing an approximate minimum spanning ellipse, that can be extended to $D$ dimensions. The main open question is to obtain an approximation ratio that does not involve the eccentricity of the minimum spanning ellipse. We have implemented this algorithm. This implementation can be viewed at http://cs.uwindsor.ca/~asishm by clicking on the link software.

\begin{table}[h]
\begin{tabular}{|c|c|}
\hline
\textbf{# Points} & \textbf{Ratio} \\
\hline
$2 \times 10^2$ & 4.18103486895557 \\
$2 \times 10^3$ & 5.267371718841295 \\
$2 \times 10^7$ & 5.281897415706886 \\
$2 \times 10^9$ & 5.28204328082953 \\
\hline
\end{tabular}
\caption{Area ratios (approximate ellipse / exact ellipse)}
\end{table}

References

