Subject: Re: data inside a circle
Posted by Michael Wallace on Tue, 14 Jun 2005 00:10:17 GMT
View Forum Message <> Reply to Message

I've been thinking about this a little more as well and wanted to throw out this suggestion for how to think about things. For this example, I'm going to ignore the elliptical case, but you should see how to extend this idea to cover ellipses as well.

Here's the approach in brief. Create a sorted X -> index mapping and a sorted Y -> index mapping. Calculate bounding box for a given circle. Search sorted X and Y lists for appropriate indexes. Intersect the indexes returned -- this gives all points within the bounding box. Check distance of point from the center of the circle to determine inside or outside.

Now, here's the longer explanation. What you can do is create mappings into your unordered list of data points. You can create a mapping of X value -> index in unordered list and a mapping of Y value -> index in unordered list. These mappings can be sorted by X value and Y value respectively. You have to do this because you can't sort the list of data points in place such that you can search on X equally as fast as you can search on Y. You can create both the X mapping and the Y mapping at the same time, so when it's all said and done, you should only have a running time of O(n log n) so far.

No matter what, you must iterate over the list of circles, O(n). For each circle, you first calculate the X and Y ranges of the bounding box, O(1). Now you have to search the mappings for the range just calculated, O(log n). Calculate the intersection of the set of indexes that satisfy the X range and the set of indexes that satisfy the Y range, O(n). Finally, calculate the distance from the center of the circle to the resultant set of data points to determine if the particular data point is inside or outside the circle, O(n).

Finally, if multiple circles have the same center, you can sort that list, O(n log n), such that circles with the same center are grouped together and are then sub-ordered from the largest radius to the smallest. Get the bounding box for the largest circle and find the data points within the circle. Instead of searching the data point list again, just use the data points you already have and do the inside/outside check for each smaller circle, culling out more data points each time.

HTH,
Mike