Subject: Re: Compute area between curves
Posted by jameskuyper on Mon, 13 Oct 2008 13:24:27 GMT
View Forum Message <> Reply to Message

frankosuna wrote:
> On Oct 12, 3:17 pm, James Kuyper <jameskuy...@verizon.net> wrote:
>> frankosuna wrote:
>>> Dear IDLers,
>>> How can I compute the area between two curves given two functions?
>> Are the curves closed? That is, do you create the complete curve by
>> drawing a line from the final <x,y> pair to the first <x,y> pair?
>>
>> Are the x values the same for the two curves? Are they evenly spaced?
>>
>> Note: you don't need to post the same question multiple times, this is a
>> newsgroup, not a chat room. Your message will stay up indefinitely. As a
>> general rule, you might have to wait 24 hours or more before getting an
>> answer.
>
> The curves are not closed... I posted some images of the actual rings
> I am trying
> to compare. They look like parabolas.

Given the geometry involved, I think it's more likely that they are
sections of an ellipse, not parabolas.

The rings might differ in shift
> and slight rotation from each other. So because the rings might be
> shifted, they probably have different x-values. Also the x-values are
> continuous..meaning that once the ring starts.. there is an (x,y)
> value until the end of the ring. When I was extracting the (x,y)
> location of every pixel that makes up the rings I noticed that a lot
> of x values had multiple y's. To fix this I used the MIN x value for
> that group in order to be able to compute the area. I'm not sure if
> that's bad or good.

I'm not sure I follow that. You describe those groups as having the same
x value, so MIN(x) doesn't make sense. If you meant MIN(y), that would
make more sense, but taking the average makes more sense than using the
minimum.

> I am using INT_TABULATED and TSUM currently and get very close values
> for the most part. I compute the area under the curve for both rings
> and then subtract but I'm not sure if that is correct either.

That approach should produce reasonable results, if you use the average
y value rather than the minimum y value. However, this approach a
assumes that for any given value of x, a vertical line at x intersects

each curve only once. Since I suspect that these are actually sections of an ellipse, it's entirely possible that some images will violate that condition. If you had an image of the entire ellipse, any given x value would cross 0, 1, or 2 times, depending upon the value of x. If you have such an image, the simplest approach would be to break the images into two sections, breaking them at the point where the tangent to the curve is vertical. Calculate the area between the curves separately for each part.

A more general approach would would work regardless of the shapes of the two curves. Just connect the two curves to create a single combined curve that starts by listing all the points on one curve in clockwise order, then continues by listing all of the points of the other curve in counter-clockwise order. As a result, the combined curve encloses the area that lies between the two curves. Then use POLY_AREA to calculates the area enclosed by the combined curve. If you've got lots of data points and a very thin area between the two curves, you'll almost certainly need to do the calculations in double precision in order to get useful results; otherwise the algorithm used will be dominated by round-off errors. Note that you should connect the two curves in such a way that the combined curve goes around the area between them in a counter-clockwise direction. If you do it in the wrong order, you'll get the negative of the correct area.

If you are taking this approach, there's no need to remove cases where you have multiple y values for the same value of x. You'll get more accurate results by keeping all of those points, rather than by replacing them with the average value of y.
