
Subject: Re: Need Some Good Ideas

Posted by [Craig Markwardt](#) on Thu, 21 Feb 2002 07:46:35 GMT

[View Forum Message](#) <> [Reply to Message](#)

Craig Markwardt <craigmnet@cow.physics.wisc.edu> writes:

> Hey David--

>

> Interesting problem! The first thing that comes to mind is to try to
> smooth the data somewhat. For example, by fitting a spline through it
> (SPL_INIT), and then interpolating the spline onto a much finer grid
> (SPL_INTERP).

I caught the bug. Here is my submission, in the form of a procedure called BLOBTERP which follows. You pass it your array of X and Y contour points, and the number of output points you want. It produces a set of contour points which are smoothly interpolated and regularly sampled along the length of the arc.

Here is a test data set and the way that BLOBTERP is called.

```
x = randomn(seed, 20) & y = randomn(seed, 20) ;; Random points
th = atan(y,x)      & ii = sort(th)          ;; Sort in a circle
x = x(ii) & x = [x, x(0)] ;; Complete the arc
y = y(ii) & y = [y, y(0)]
```

```
blobterp, x, y, 50, xx, yy
plot, xx, yy
```

The only thing that might not be "perfect" is the seam where the outline joins itself, where there will be a derivative discontinuity. Unfortunately the IDL spline routine doesn't do cyclic constraints, so it would be hard to perfect this. You might consider dropping points near (x(0),y(0)).

Time for resting,
Craig

```
;; X & Y = outline of blob
;; N = number of desired interpolants
;; XX & YY = regularly sampled interpolants
pro blobterp, x, y, n, xx, yy
```

```
npt = n_elements(x)
nc = npt*100
t = dindgen(npt)
```

```

;; Interpolate very finely
t1 = dindgen(nc+1)/100
x1 = spl_interp(t, x, spl_init(t, x), t1)
y1 = spl_interp(t, y, spl_init(t, y), t1)

;; Compute cumulative path length
ds = sqrt((x1(1:*)-x1)^2 + (y1(1:*)-y1)^2)
ss = [0d, total(ds, /cum)]

;; Invert this curve, solve for TX, which should be evenly sampled in
;; the arc length space
sx = dindgen(n+1)*max(ss)/n
tx = spl_interp(ss, t1, spl_init(ss, t1), sx)

;; Reinterpolate the original points using the new values of TX
xx = spl_interp(t, x, spl_init(t, x), tx)
yy = spl_interp(t, y, spl_init(t, y), tx)
return

```

end

--

 Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
 Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
