Subject: Re: interpolation in 5 dimensional space (how and speed)
Posted by Chris Lee on Thu, 05 May 2005 08:46:06 GMT
View Forum Message <> Reply to Message

In article <1115247557.742714.266820@o13g2000cwo.googlegroups.com>,
"pdeyoung" <deyoung@hope.edu> wrote:

> We have a project where we track a large number of test particles
> through a magnetic field. Then using the ending variables (five) we
> want to infer the starting parameters for real particles by comparing
> them to the nearest test particles. (Please don't laugh at my efforts
> below - I really don't know the correct way to do this.) In some sense
> the one dimensional analogy would be to have a y value and find the x
> values given that you could calculate the y's from a grid of x values
> ahead of time. Of course in the one-d example there is no assurance
> that the y's will be equally spaced. Similarly, in the code below the
> results of the tracks are not equally spaced in output space. (In the
> code below for simplicity while testing, I just use random arrays.) For
> background, I found the closest point in each "quadrant" and then found
> a weighted average based on the distance from the test point and the
> closest points. (This could be totally bogus.) Anyway, is there a
> better (and faster) way to do this or is this approach reasonable. If
> so, is there a way to do it faster. Ultimately we will have to do this
> 10^6 times for the real data set. I am using IDL6.1 Thanks in advance.
> Paul DeYoung
> deyoung@hope.edu

Are you really trying to do a 5D 'bi-cubic' interpolation here?
Does it actually work? Save yourself a headache and put all
of the points into an array, the sooner you do this, the
better the code will scale/vectorize. e.g.

```
min_array=[min_dist_1, min_dist_2...., min_dist_32]
quad_array=[quad1(index1), quad2(index2)....quad32(index32)]

indices=array_indices(ranarray_1, quad_array)
sqrt_min_array=sqrt(min_array)
weight=total(1/sqrt_min_array)

xinterp=total(indices[0,*]/sqrt_min_array)/weight
;
```

it might not speed it up, but it's shorter and easier to read. And when
you vectorize the input then it should be easier/possible to vectorize
interpolate. It's this vectorizing (in particle number) which will get
you the biggest boost.

Where did the equation for the interpolation come from? Have a look at what IDL does in INTERPOLATE, it might be different.

Chris.