Subject: Re: Questions on the subject of Interpolation
Posted by Kenneth P. Bowman on Thu, 20 Mar 2008 13:14:44 GMT
View Forum Message <> Reply to Message

In article
<94a536e7-1c7f-4df5-b699-c2483254ceee@d4g2000prg.googlegroups.com>,
 eyuchen@gmail.com wrote:

> Hi all,
>
> I was having problems on making 2D interpolations on irregular data
> points. I read the past posts on this news group, I read the articles
> in David's website, and I read Ken's sample chapter on this subject.
>
> However, there are still some aspects that I would like to ask about.
>
> 1. Is it possible to construct an irregular data interpolation scheme
> that reduced to bilinear interpolation at the limit of regular grid?
>
> Let me elaborate on this a little bit: The interpolation schemes for
> regular grids all take the neighboring 4-pts to do the job, whereas
> that for irregular grids takes only 3. Sometimes, we do have cases
> where the grid is just slightly distorted and a reasonable 4-pt
> interpolation *seems* possible.(namely, it is possible to define the
> "neighboring four points," and one starts to wonder if a 4-pt scheme
> is superior than the 3-pt scheme.

The mathematical form for the interpolating function is different in
the the triangular- and rectangular-grid (bilinear) cases. The triangular
case uses a plane through the three points defining the triangle. The
bilinear interpolator is actually a curved surface in most cases.

Here is a demonstration.

```
x = [0.0, 1.0]
y = [0.0, 1.0]
z = [[0.0, 1.0], [1.0, 0.0]]
xx = REBIN(FINDGEN(11)/10.0, 11, 11)
yy = REBIN(TRANSPOSE(FINDGEN(11)/10.0), 11, 11)
zz = INTERPOLATE(z, xx, yy)
iSurface, zz, xx, yy
```

To make an irregular (triangular) and regular (rectangular) grid agree,
you need to triangulate the regular grid and use planar interpolation,
rather than doing bilinear interpolation. (That is, divide each rectangular
element into two triangles.)

> If 3-pt scheme is superior, then we shouldn't be using 4-pt scheme

> even in the case of regular grids. If 4-pt scheme is better, there
> should be a way to do 4-pt irregular interpolation, at least when some
> conditions are met, right?

Interpolation is an approximation procedure and necessarily involves compromises. You should be aware of what your chosen interpolator is doing to your data and make your choice based on your criteria (e.g., accuracy, smoothness, speed, ease of use). Do you need an interpolator with continuous derivatives, etc.? Do you require that all interpolated values be within the range of tabulated (input) values? (Higher order interpolators do not guarantee that.)

> 2. There is a very good property about 1D linear interpolation, which
> can be shown as follows:
>
> IDL> x=dindgen(5)
> IDL> y=[1.2,3.2,4.5,6.1,6.2] ; just some arbitrary irregular spaced
> data.
> IDL> plot, x, y
> IDL> print, interpol(y,x,2.1)
> 4.6599998
> IDL> print, interpol(x,y,4.6599998)
> 2.0999999
>
> that is, INTERPOL(x,y,INTERPOL(y,x,?))=? (? is just any number or
> vector.)
> I believe this really lies in the fact that the inverse of linear
> transformation is still a linear transformation. Therefore, a linear
> mapping from x to y is the inverse of the linear mapping from y to x.
> (you can add a /spline keyword and INTERPOL(x,y,INTERPOL(y,x,?))=?
> won't be true anymore)

Linear interpolation is easy to invert, if you need that property.

> 3. What is the real difference between INTERPOLATE and BILINEAR? It
> seems to me that INTERPOLATE can do everything BILINEAR does, with
> more accuracy.

Beats me.

Ken Bowman