Subject: Re: Interpolation to conserve integrated flux
Posted by penteado on Sun, 30 May 2010 05:47:23 GMT
View Forum Message <> Reply to Message

These may help

http://www.ppenteado.net/idl/pp_integral.html
http://www.ppenteado.net/idl/pp_resample.html
http://www.ppenteado.net/idl/pp_convol.html

http://www.ppenteado.net/idl/pp_integral.pro
http://www.ppenteado.net/idl/pp_resample.pro
http://www.ppenteado.net/idl/pp_convol.pro

Subject: Re: Interpolation to conserve integrated flux
Posted by jkj on Sun, 30 May 2010 12:03:42 GMT
View Forum Message <> Reply to Message

On May 29, 8:13 pm, John Shaw <jds...@udel.edu> wrote:
> Hi all,
>
> I am hoping someone out there can help me with this.
>
> I am looking to combine spectra such that the integrated flux between
> any arbitrary set of points is conserved.
>
> Consider a set of two equal length float arrays, one containing
> wavelength values (call this x1, x2, x3, ...) and one containing flux
> values (call these y1, y2, y3,...). So that y1 goes with x1 and y2
> goes with x2, and so on.
>
> Unfortunately, the wavelength arrays (x1, x2, x3, ...) have similar
> values but not the same, e.g.:
>
> x1 = [3212.7, 3215.1, 3217.5, 3219.9, ...]
> x2 = [3213.1, 3215.4, 3217.7, 3220.0, ...]
>
> What I want to do is combine x1 and x2 to x_comb and y1 and y2 to
> y_comb such that if one were to sum between two wavelengths the in the
> new arrays (x_comb and y_comb) you would get the same as doing them in
> the original (x1,y1 and x2,y2) and adding the results. That is, if I
> need to sum up values in y when x is between 3214.0 and 3219.0. For
> x1, I would linearly interpolate the values in y1 for the two
> wavelengths and sum under the curve. I then do the same for x2 and
> y2. I can already do this - that part is simple. What I want to do
> is have new arrays x_comb and y_comb such that I can do the same sum
> across such that the same (summed region in y1) + (summed region in

> y2) = (summed region in y_comb).
>
> Is there an algorithm that anyone knows that will do this?
>
> Or, as an intermediate step, is there a way interpolate the values in
> x2 to x1, and y2 to an new set of values (y2_new) that keeps the sums
> of y2 and y2_new the same between any two arbitray points (as
> described above)?
>
> Thanks - John.

You should use the spline function to translate one spectrum to the same x-values as the other. We use the spline function with electron spectra quite often. The simplest thing to do is use the INTERPOL function with SPLINE set.

y_comb = interpol(y_orig, x_orig, x_other, /spline)

where "x_orig" and "y_orig" are the values of one spectrum or the other and "x_other" are the wavelengths of the other spectrum, the wavelengths that you wish the original spectrum was defined in so that you could directly compare their integrated values. Simply overplot the "orig" and "comb" [x_comb, of course, is the same as x_other] sets of spectrum to convince yourself that the spline function did a good job of translating from one to the other.

-Kevin

Subject: Re: Interpolation to conserve integrated flux
Posted by Craig Markwardt on Sun, 30 May 2010 19:32:15 GMT
View Forum Message <> Reply to Message

On May 29, 9:13 pm, John Shaw <jds...@udel.edu> wrote:
> Hi all,
>
> I am hoping someone out there can help me with this.
>
> I am looking to combine spectra such that the integrated flux between
> any arbitrary set of points is conserved.
>
> Consider a set of two equal length float arrays, one containing
> wavelength values (call this x1, x2, x3, ...) and one containing flux
> values (call these y1, y2, y3,...). So that y1 goes with x1 and y2
> goes with x2, and so on.
>
> Unfortunately, the wavelength arrays (x1, x2, x3, ...) have similar
> values but not the same, e.g.:

>
> x1 = [3212.7, 3215.1, 3217.5, 3219.9, ...]
> x2 = [3213.1, 3215.4, 3217.7, 3220.0, ...]
>
> What I want to do is combine x1 and x2 to x_comb and y1 and y2 to
> y_comb such that if one were to sum between two wavelengths the in the
> new arrays (x_comb and y_comb) you would get the same as doing them in
> the original (x1,y1 and x2,y2) and adding the results. That is, if I
> need to sum up values in y when x is between 3214.0 and 3219.0. For
> x1, I would linearly interpolate the values in y1 for the two
> wavelengths and sum under the curve. I then do the same for x2 and
> y2. I can already do this - that part is simple. What I want to do
> is have new arrays x_comb and y_comb such that I can do the same sum
> across such that the same (summed region in y1) + (summed region in
> y2) = (summed region in y_comb).
>
> Is there an algorithm that anyone knows that will do this?
>
> Or, as an intermediate step, is there a way interpolate the values in
> x2 to x1, and y2 to an new set of values (y2_new) that keeps the sums
> of y2 and y2_new the same between any two arbitray points (as
> described above)?

First, I think you need to be a little careful if your measurements are at exact bin centers, or you are giving bin edges, or what. Since you want to do your operation carefully, you need to be careful about definitions.

There are an infinite number of ways to interpolate a series. And I believe there are an infinite number of ways to do it while conserving area. So you have a lot of choices!

It looks like your wavelengths are quantized at 0.1 units. The most obvious thing to do is to subdivide your original flux into smaller 0.1 unit bins, assuming equal flux density throughout the original bin. You can do that pretty easily, and can do it exactly. Once you have all the spectra on 0.1 unit binning, then they should all be on a matching X scale, and you can establish new bin edges and sum appropriately.

Craig

Subject: Re: Interpolation to conserve integrated flux
Posted by jdshaw on Sun, 30 May 2010 19:48:58 GMT
View Forum Message <> Reply to Message

Thanks for the pointers, the points I gave are examples and not exact.

For display purposes, I think INTERPOL(/spline) will do. I did not want to reinvent the wheel when doing this but I think I'll write a routine that numerically integrates and keeps a running tally of the total at each pixel point for each spectra and combine as needed for the flux values.

I think I can keep track of the "raw" counts this way, too, such that I beat down the error bars.

- John
