Subject: Re: Resampling data with irregular time base
Posted by Richard G. French on Sat, 05 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Karl Krieger wrote:

```
>
> I have data with an irregular time base, which I would like to resample
> in a regular spaced time base. How can I average over all original data
> points in each interval of the new time vector without resorting to a
> FOR loop?
> Currently I am using this horrible kludge:
>
> deltat = newtime[1] - newtime[0]
> FOR n=0, n_elements(newtime)-1 DO BEGIN
>   index = where((oldtime GT (newtime[n]-deltat/2.)) AND $
>                 (oldtime LE (newtime[n]+deltat/2.)), $
>                 count)
>   IF count GT 0 THEN newdata[n] = total(olddata[index]) / count
> ENDFOR
>
```

It appears to me that you are not after an interpolated smooth function here, since if you have a very small deltat, you will get lots of newdata[] values that are zero in cases when no olddata[] fall within the time range of newdata[]. If you have a densely populated set of olddata and you are trying to average, this may not be a problem. There may be a way to use the histogram routine to get the information you are after. I have not checked into that, but others may have suggestions. It seems to me that the HISTOGRAM routine can give you the elements that go into each bin, and if so, then you can retrieve the points that contribute to each time bin and do the normalization yourself.

I think the REVERSE_INDICES keyword may be what you want, but you still may end up having to do a loop.

>
> Set this keyword to a named variable in which the list of reverse indices is returned. This list is returned as a longword vector whose number of elements is the sum of the number of elements in the histogram, N, and the number of array elements included in the histogram, plus one.
>
> The subscripts of the original array elements falling in the ith bin, 0 ï¿½ i < N, are given by the expression: R(R[i] : R(i+1)-1), where R is the reverse index list. If R[i] is equal to R[i+1], no elements are present in the ith bin.
>
> Example Make the histogram of array A:

```
>
> H = HISTOGRAM(A, REVERSE_INDICES = R)
>
> IF R(i) NE R(i+1) THEN A(R(R(I) : R(i+1)-1)) = 0
```

> ;Set all elements of A that are in the ith bin of H to 0.
>
> The above is usually more efficient than the following:
>
> bini = WHERE(A EQ i, count)
>
> IF count NE 0 THEN A(bini) = 0
>

If you don't mind interpolating between neigboring points, you might try
doing a linear interpolation onto a very fine time grid that is
oversampled by some
integer multiple of the deltat you want in the end - choose this
multiplier
to be approximately the value of the closest spacing of your data
points.
Then you can use REBIN to average the regularly interpolated result to
deltat.
I've used this approach quite often when I have a relatively smooth but
irregularly spaced set of points - if it is REALLY smooth, then you can
use cubic splines and bypass the interpolation step altogether. But my
hunch is that you have some data that may be quite variable over short
time scales and that where there are no data points, you really want a
zero in the newdata[] array, not an iterpolation. In this case, I would
do a histogram of the oldtime array at the newtime spacing, pick out the
zero elements in the histogram and set the corresponding elements of the
interpolated array to zero. I guess it all depends on the nature of your
data and the accuracy you are after.

Dick French
rfrench@mediaone.net