Subject: Re: rebinning data on new time samples without loops?
Posted by Chris Lee on Fri, 12 Dec 2003 09:18:59 GMT
View Forum Message <> Reply to Message

In article <d916d48a.0312111543.8425684@posting.google.com>, "James"
<jbattat@cfa.harvard.edu> wrote:

```
>  Hi there,
>  I have a time series of data, y, sampled on an irregular grid, t1. I
>  would like to rebin y onto an existing time series (also irregular), t2.
>  Right now, I am using a for loop and a where statement to get it done,
>  but it is rather slow, and I'm guessing that there's a way to do it
>  without a for loop (probably even using Histogram...). The t1 series is
>  ~100,000 points and the t2 series is ~3,000 points. On my system, my
>  code takes 10-15 mins to run. I'd like to make a faster solution.
>  There are basically only 2 requirements. 1. If t1[i] is more than some
>  user defined dTMax away from any t2 sample then omit the data point
>  2. I'd like to know how many t1 points went into each t2 bin.  Right now
>  i do it like this (roughly):
>    t1 = original time samples
>    y1 = data gridded on t1
>    t2 = desired time sampling
>    y2 = data gridded on t2
>
>    code to ensure overlap between t2 and t1 code to eliminate all points
>    that are not part of the overlap
>
>    for loop over t1
>   dT = t2 - replicate(t1[i],nt2)
>   ID = where(dT EQ min(dT))
>   dump y1[i] data into y2[ID] if min(dT) < dTMax increment counter[ID]
>    endfor loop
>
>    then divide y2/counter to get the rebinned value.
>  I'm sure there's a better way.  I'm curious if there's an efficient way
>  in which no loops are used.
>  Thanks very much in advance,
>  James
```

Hi,

On your code, do you need the replicate line,
[1,2,3,4] - 3 = [-2,-1,0,1]
[1,2,3,4] - [3,3,3,3] = [-2,-1,0,1]

anyway..

What's wrong with INTERPOLATE or INTERPOL?

y2=interpol(y1, t1, t2)

TRIANGULATE if the data is two dimensional.

If you really want to use your method, I would suggest making a two dimensional grid from y1 (t1, t2), and using some matrix tricks on it. Possibly based on the "Vector Comparison" thread a while back

http://groups.google.com/groups?hl=en&lr=&ie=ISO-885 9-1&q=
vector+comparison+idl&btnG=Google+Search

should be one line obviously.

Chris.

Subject: Re: rebinning data on new time samples without loops?
Posted by K. Bowman on Fri, 12 Dec 2003 14:31:23 GMT
View Forum Message <> Reply to Message

In article <d916d48a.0312111543.8425684@posting.google.com>,
jbattat@cfa.harvard.edu (James) wrote:

```
>  Hi there,
>
>  I have a time series of data, y, sampled on an irregular grid, t1.
>  I would like to rebin y onto an existing time series (also irregular),
>  t2.
>
>    for loop over t1
>   dT = t2 - replicate(t1[i],nt2)
>   ID = where(dT EQ min(dT))
>   dump y1[i] data into y2[ID] if min(dT) < dTMax
>   increment counter[ID]
>    endfor loop
>
>    then divide y2/counter to get the rebinned value.
```

Since your data is already sorted by time, you can use a binary search (VALUE_LOCATE). It is *much* faster than WHERE on long vectors.

Loop over t2 (not t1), find the points in t1 closest to t2-dt and t2+dt using VALUE_LOCATE. Then average as you are doing.

Ken bowman

Subject: Re: rebinning data on new time samples without loops?
Posted by jbattat on Fri, 12 Dec 2003 21:11:16 GMT
View Forum Message <> Reply to Message

I don't know why i wrote the replicate command in my original post...
I did not use replicate in my code!
Anyway, I would prefer to bin rather than interpolate, because I would
like to reduce my noise (t1 is much more finely sampled than t2 so
averaging many t1 points rather than interpolating should help)

But for quick and dirty analysis, I will use INTERPOL as a fast fix.
Thanks for the suggestion.

I'm still on the lookout for a clever array solution...

take care,
james

> Hi,
>
> On your code, do you need the replicate line,
> [1,2,3,4] - 3 = [-2,-1,0,1]
> [1,2,3,4] - [3,3,3,3] = [-2,-1,0,1]
>
> anyway..
>
> What's wrong with INTERPOLATE or INTERPOL?
>
> y2=interpol(y1, t1, t2)
>
> TRIANGULATE if the data is two dimensional.
>
> If you really want to use your method, I would suggest making a two
> dimensional grid from y1 (t1, t2), and using some matrix tricks on it.
> Possibly based on the "Vector Comparison" thread a while back
>
> http://groups.google.com/groups?hl=en&lr=&ie=ISO-885 9-1&q=
> vector+comparison+idl&btnG=Google+Search
>
> should be one line obviously.
>
> Chris.

Subject: Re: rebinning data on new time samples without loops?
Posted by Chris Lee on Sat, 13 Dec 2003 17:41:48 GMT
View Forum Message <> Reply to Message

James wrote:

> I don't know why i wrote the replicate command in my original post...
> I did not use replicate in my code!
> Anyway, I would prefer to bin rather than interpolate, because I would
> like to reduce my noise (t1 is much more finely sampled than t2 so
> averaging many t1 points rather than interpolating should help)
>
> But for quick and dirty analysis, I will use INTERPOL as a fast fix.
> Thanks for the suggestion.
>
> I'm still on the lookout for a clever array solution...
>
> take care,
> james
>
<snip interpolate...>

ok, binning it shall be.

This is a variation of the vector comparison referred to in the last post..

The non-loop version would be to expand the in_x and out_x into
matrices, subtract them, abs the result , then compare that to the
dtmax, so...

```
 res=abs(in_x#replicate(1,n_elements(out_x))-out_x##replicate (1,n_elements(in_x)))

gt dtmax
;all on one line..
out_y = total((in_y#replicate(1, n_elements(in_x)))*res,1)/total(res,1)
```

Of course, your memory usage goes up from O(N+M) (where N is the input
and M the output array sizes) to O(N*M), not a trivial increase. If you
can't afford that extra stick of RAM (gigabyte sticks in your case)
don't use all of the output array at once, and do half the loops (say),
don't reallocate the arrays etc. Alternatively, loop over the smaller array.

...brain thinking...Histogram with reverse indices keyword...if it were
only a regular output array.

The reverse indices should give you all of the bin numbers in the big
array that fall into each output array element, but this only works if
the output array is regular (I'm possibly wrong, but I can't see
anything in the help file). Then it's trivial to sum them and divide
them, as above.

Chris.