Subject: Re: Newbie question concerning summations/loops in IDL
Posted by mbweller on Wed, 30 Jul 2008 23:55:35 GMT
View Forum Message <> Reply to Message

On Jul 30, 3:57 am, Jeremy Bailin <astroco...@gmail.com> wrote:
> On Jul 30, 4:33 am, Wox <nom...@hotmail.com> wrote:
>
>
>
>> On Tue, 29 Jul 2008 23:19:19 -0700 (PDT), mbwel...@gmail.com wrote:
>>> On Jul 29, 7:27 pm, Chris <beaum...@ifa.hawaii.edu> wrote:
>>>> On Jul 29, 1:12 pm, mbwel...@gmail.com wrote:
>
>>>> > Hello,
>
>>>> > I have need of some experienced users with sort of a newbie question.
>
>>>> > I am writing a code that needs a summation in it, this is what I have
>>>> > thus far:
>
>>>> > v= ; volume of region
>>>> > a= ; area of region
>>>> > o= 60*!pi/180 ; fault dip angle
>>>> > g= ; scaling factor
>>>> > t= 150 ; elastic lithosphere thickness
>>>> > h= ; depth of faulting
>
>>>> > ind_small = where(thaext[1,*] lt t)
>>>> > ind_large = where(thaext[1,*] ge t)
>>>> > thaext_small = thaext[*,ind_small]
>>>> > thaext_large = thaext[*,ind_large]
>
>>>> > ens=(sin(o)*cos(o)/v)* ; horizonatal normal strain for small faults
>>>> > enl=(cos(o)/a)* ; horizonatal normal strain for
>>>> > large faults
>>>> > evs=(-sin(o)*cos(o)/v)* ; vertical normal strain for small faults
>>>> > evl=(-cos(o)/a)* ; vertical normal strain for
large faults
>
>>>> > The summation needs to be after * in the ens, enl, evs and evl
>>>> > fields.
>>>> > It must be of the form:
>>>> > summation N, i=0 [Di Li Hi] for small faults, where N = ind_small, Hi=
>>>> > T/sin(o) and
>>>> > summation N, i=0 [Di Li] for large faults, where N=ind_large
>
>>>> > Could anyone provide any insight/guidance?
>

>>>> > Thanks,
>>>> > ~Matt
>
>>>> I don't know what some of your variables are (Li? Di?), but you might
>>>> want to look at TOTAL() to start- you can use that to do most
>>>> summation tasks.
>
>>> L and D are data from a ascii table that is already ready in, while i
>>> is the indice of the summation. I've looked at total, but the examples
>>> were sorely lacking. I was hoping that perhaps a useful example, given
>>> my code and desire, could be supplied.
>
>>> ~Matt
>
>> I'm not sure what you mean with "summation N, i=0 [Di Li Hi] ... where
>> N=ind_small". The index i goes from 0 to what? And what are you
>> summing? D[i]*L[i]*H[i]?
>
> Okay, if I understand it correctly, then what you're saying is that in
> ind_small you multiply by an extra factor of t/sin(o) inside the sum,
> but not in ind_large?
>
> H = replicate(1., n_elements(D))
> H[ind_small] = t/sin(o)
> summation = total(D*L*H)
>
> Is that what you're looking for?
>
> (by the way, look up !RADEG).
>
> -Jeremy.

Thanks for the responses.
I think that I did not adequately explain what I needed to do, Let me
be more specific now. (this might be a little complicated)

I have a .sav file which is a FLOAT array[2,7923] but may go as high
as [2,18000] and the forms are as such: [id, Length].

ind_small and ind_large are where I select the lengths to be smaller
or larger, respectively, than t. Then place them back into the new
matrices thaext_small and thaext_large. (not completely sure if this
is necessary.)

Now comes the part that I am a little confused on how to program.

ens, enl, evs and evl fields are going to be a constant * a summation
(which will be different for all four).

The number of sums or (N) needs to be equal to the number of the faults down selected by Ind_small (or since ind_small = where(thaext[1,*] lt t), it needs to sum the number of the second column in the array). This number will be different for bot the large and small cases (eg. ind_large = where(thaext[1,*] ge t)). So, i then should be # of points in column 2 of ind_small/ind_large - 1 (I would think).

The summation is [D[i]*L[i]*H[i]] for small faults and the summation is [D[i]*L[i]] for large faults, where:
D[i]=C[i]*L[i] for small faults and
D[i]=C[i]*H[i] for large faults,
L[i]= length (from column 2 of thaext_small/thaext_large) and
H[i]=(1/2 or 1/3)*L[i] for small faults and
H[i] = t/sin(o) for large faults and
C[i] may or may not be a constant

This should now read as constant * summation[C[i]*L[i]*L[i]*L[i]] for small faults and constant * summation [C[i]*L[i]] for large faults.

I think that's everything I need to be able to do, hopefully it's a bit clearer now.

Thanks,
~Matt
