Subject: Re: faster then where possible?
Posted by Jean H. on Thu, 07 May 2009 16:42:28 GMT
View Forum Message <> Reply to Message

rogass@googlemail.com wrote:
> Hi,
> i'm searching for some alternative approaches to compute the following
> "much" faster:
>
> -> matrix1 has m columns and n rows, matrix2 has 2 columns and n rows
> -> the values in matrix2 are NOT in matrix1, but within the min-max-
> range of matrix1
>
> szm1=size(matrix1,/dimensions)
> szm2=size(matrix2,/dimensions)
> index={ind:ptr_new()}
> indices=replicate(index,szm2[1])
>
> for j=0ull,szm1[1] do begin
> helpindex= where(matrix1[*,j] ge matrix2[0,j] and matrix1[*,j] le
> matrix2[1,j],c)
> if c gt 0 then begin
> indices[j] = ptr_new(uintarr(c))
> (*indices)[j]=helpindex
> endif else continue
> endfor
>
> It seems to be a typical Nearest-Neighbor-Problem, but all alternative
> approaches I tried were always slower. Maybe someone here has a good
> idea?
>
> Thank you and best regards
>
> Christian
>

Hi,

if you have enough memory, you could do something like this (not tested):
minArray = rebin(matrix2[0,*], n_elements(matrix1[*,0],
n_elements(matrix1[0,*]))
maxArray = rebin(matrix2[1,*], n_elements(matrix1[*,0],
n_elements(matrix1[0,*]))

goodIdx = where (matrix2 lt minArray and matrix2 gt minarray)

then you just have to transform the index so they match each row

Jean