Subject: Re: Please help me avoid loops and conditionals
Posted by pford on Sat, 13 Sep 2003 20:58:54 GMT
View Forum Message <> Reply to Message

"Chris Lee" <cl@127.0.0.1> wrote in message
news:<20030910.151522.638422090.14392@buckley.atm.ox.ac.uk>...
> In article <c857619b.0309090844.540303e@posting.google.com>, "Patrick
> Ford" <pford@bcm.tmc.edu> wrote:
>
>
>> Greetings,
--Deleted--
>
> Hi,
> I did answer this earlier, I think it may have been sent as an email
> instead though. This is the abbreviated version.
>
> The function elp4, below, speeds up the elp2 function you give, I tested
> it on a few numbers, which were in the email (which I'm not sure got past
> my mail server, given my email address).
>
> Roughly, I tested elp2 and elp4 for 100 iterations of a
> box_dim=[10,100,250,1000] case, the elp2 took about as long doing the
> box_dim=100 case as the elp4 did on the 1000 case (25 seconds).
>
> There is another method where you can recycle the w1 and w2 arrays
> between each call to elp4, but for this to be valid, the box_dim, a, b,
> e_a and e_b variables must be constant between calls (so the masks remain
> valid). In this case, 100 iterations of box_dim=1000 took 1.7 seconds
> (compared to 25 seconds for elp4 and >60 (?) for elp2).
>
> Reusing the box_array (saving on mallocs) didn't have that much effect,
> only 0.3 seconds on any case (which is admittedly 20% for the fastest
> version but 1% for the slower).
>
> I hope Patrick got the email with the other functions in it.
>
> Chris.
> ----
>
> function elp4, a,b, box_dim,vval,e_a,e_b, I_ratio
> x_box=box_dim/2
> o_val=fix(vval/I_ratio)
> v=fix(vval)
>
> box=make_array(dimension=[box_dim,box_dim],value=0)
>
> x=(findgen(box_dim)-x_box) # replicate(1,box_dim)

```
>  y=transpose(x)
>
>  w1=where((x/(a+e_a))^2 + (y/(b+e_b))^2 le 1.0,c1)
>  if(c1 gt 0) then w2=where(((x/a)^2 + (y/b)^2)[w1] le 1.0,c2)
>
>  if(c1 gt 0) then box[w1]=v
>  if(c2 gt 0) then box[w1[w2]]=o_val
>
>  return, box
>
>  end
```

First, thank you to all who responded.

Chris, I did not get your email.

I fiddled around with this trying to do this and I came up with the following:

```
pro test

a_d = 6
b_d = 4
a = 15
b = 21
box_dim = 64
box = intarr(box_dim,box_dim)
bxd2 = box_dim*2
view = intarr(bxd2,bxd2)

  x = float((indgen(box_dim)  - box_dim/2) # replicate(1, box_dim))
  x1 = x
  y = (transpose(x) / b)^2
  x = (x / a)^2

  y1 = (transpose(x1) / (b+b_d))^2
  x1 = (x1 / (a_d+a))^2

  r0 = x + y
  r1 = x1 + y1

  mask0 = where(r0 LE 1.0)
  mask1 = where(r1  LE 1.0)
  mask2 = where( (r1 LE 1.0) AND NOT(r0 LE 1.0))

  box[mask1] = 255
  box[mask0] = 128
  view[0:box_dim-1, 0:box_dim-1] = box
```

```
box[mask2] = 255
box[mask0] = 128
view[box_dim:bxd2-1, box_dim:bxd2-1] = box
tv, view

end
```

Notice no loops or if statements.(yea!) Now the next question is it more efficient to use the mask1 method where I double assign or mask2 where I expand a logical expression?
Any significant time improvement in changing NOT(r0 LE 1.0) to (r0 GT 1.0)?

I was trying to eliminate the values in mask1 that corresponded to mask0, but at this point I don't see how to do that efficiently. My attempts using the where command were a disaster.
