Subject: Improving a piece of code with arrays and for-loops
Posted by Halfdan on Thu, 08 Feb 2007 13:23:53 GMT
View Forum Message <> Reply to Message

Hello

I have been looking at my same piece of IDL-code for quite a while now and I have yet not found any good method to improve it. I want to improve the speed and rid the code of the nested for-loops. Maybe there is someone here who has good ideas and is willing to point me in the right direction?

The problematic code is below. It is a part of a method to estimate wind gusts in output from an atmospheric model. The i and j dimensions are the x- and y-locations of the model-points in the horizontal and s are the model level heights in the vertical (starting from the model top and growing towards the surface).

The code works in the vertical, starting from the surface (largest value of s) and works upwards (towards smaller s) to where the value of the variable tke is less than tkelvl or tke_diff is less than a very small number. The code has to do the following three things:

1. Choose the greatest value of wsp (windspeed) where the value of int_diff exceeds 0.
2. Choose the greatest value of wsp where the value of int_diffver exceeds 0.
3. Choose the greatest value of wsp.

This has to be repeated for every grid-point in the horizontal (I have to assume that I have very little a priori knowledge of the behaviour of any of my variables at any gridpoint and model height).

Any ideas on improving this?

Thanks in advance,
Halfdan

ps. The problematic code:

```
    for i=1,ni-2 do begin
       for j=1, nj-2 do begin
          s = ns-1
          REPEAT BEGIN
             if int_diff(i,j,s) GE 0. AND wsp(i,j,s) GT fgtmp(i,j,
0) then $
               fgtmp(i,j,0) = wsp(i,j,s)
             if wsp(i,j,s) GT fgtmp(i,j,1) then $
```

```
                fgtmp(i,j,1) = wsp(i,j,s)
              if int_diffver(i,j,s) GE 0. AND wsp(i,j,s) GT
fgtmp(i,j,2) then $
                fgtmp(i,j,2) = wsp(i,j,s)
              s=s-1
            ENDREP UNTIL tke(i,j,s) LT tkelvl(i,j) OR
tke_diff(i,j,s) LT eps
        endfor
      endfor
```

Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Wox on Fri, 09 Feb 2007 12:55:38 GMT
View Forum Message <> Reply to Message

How about this:

```
; indsetmin (s never goes there)
indsetmin=where((tke lt rebin(tkelvl,ni,nj,ns)) or (tke_diff lt eps))

; Set windspeeds to the minimum value (you can also take 0,
; if these speeds never go below zero, which seems logical ;-))
wspmin=min(wsp)
wsp[indsetmin]=wspmin

; Maximum wsp
fgtmp[*,*,2]=max(wsp, DIMENSION=3)

; Maximum wsp where (int_diff ge 0)
wsptmp=wsp
wsptmp[where(int_diff lt 0)]=wspmin
fgtmp[*,*,0]=max(wsptmp, DIMENSION=3)

; Maximum wsp where (int_diffver ge 0)
wsp[where(int_diffver lt 0)]=wspmin
fgtmp[*,*,2]=max(wsp, DIMENSION=3)
```

On 8 Feb 2007 05:23:53 -0800, "Halfdan" <halfdana@gmail.com> wrote:

```
>      for i=1,ni-2 do begin
>         for j=1, nj-2 do begin
>            s = ns-1
>            REPEAT BEGIN
>               if int_diff(i,j,s) GE 0. AND wsp(i,j,s) GT fgtmp(i,j,
> 0) then $
>                  fgtmp(i,j,0) = wsp(i,j,s)
```

```
>                 if wsp(i,j,s) GT fgtmp(i,j,1) then $
>                   fgtmp(i,j,1) = wsp(i,j,s)
>                 if int_diffver(i,j,s) GE 0. AND wsp(i,j,s) GT
> fgtmp(i,j,2) then $
>                   fgtmp(i,j,2) = wsp(i,j,s)
>                 s=s-1
>              ENDREP UNTIL tke(i,j,s) LT tkelvl(i,j) OR
> tke_diff(i,j,s) LT eps
>           endfor
>       endfor
```

Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Wox on Fri, 09 Feb 2007 12:57:26 GMT
View Forum Message <> Reply to Message

On Fri, 09 Feb 2007 13:55:38 +0100, Wox <nomail@hotmail.com> wrote:

<snip>

```
> ; Maximum wsp where (int_diffver ge 0)
> wsp[where(int_diffver lt 0)]=wspmin
> fgtmp[*,*,2]=max(wsp, DIMENSION=3)
```

must be fgtmp[*,*,1]

Subject: Re: Improving a piece of code with arrays and for-loops
Posted by Halfdan on Thu, 01 Mar 2007 11:30:55 GMT
View Forum Message <> Reply to Message

I got the method to work correctly with slight modifications. It might be a bit late but her goes.

I realized that my possible s-levels were not continous, i.e. there could be "valid" s-levels away from the surface with "illegal" levels in between. I therefore had to put a more stringent conditions on the possible s-values as I only want the continous stretch of "valid" levels near the surface.

I also choose some sensible values for wsp when the method would otherwise give 0.

há

```
    ; What s-values are not possible?
```

```
    idx = intarr(ni,nj,ns)
    idx[where( tke LT rebin(tkelvl,ni,nj,ns) or tke_diff LT eps)] =
1
    idx = reverse ( total( reverse(idx, 3), 3, /cumulative), 3)

    ; What s-values are not possible?
    indsetmin=where( idx NE 0 )

    ; Zero value for the wind
    wsptmp=wsp
    wspmin=0.
    wsptmp[indsetmin]=wspmin

    ; Max wsp
    fgtmp[*,*,1]=max(wsptmp, DIMENSION=3) > wsp[*,*,ns-1]

    ; Max wsp where(int_diff lt 0.)
    wsptmp[where(int_diff lt 0.)]=wspmin
    fgtmp[*,*,0]=max(wsptmp, DIMENSION=3) > wsp[*,*,ns-1]

    ; max wsp where (int_diffver lt 0)
    wsptmp[where(int_diffver lt 0.)]=wspmin
    fgtmp[*,*,2]=max(wsptmp, DIMENSION=3) > wsp[*,*,ns-1]
```