
Subject: Re: Simple? problem

Posted by [James Kuyper](#) on Tue, 09 Apr 2002 17:23:47 GMT

[View Forum Message](#) <> [Reply to Message](#)

Ivan Valtchanov wrote:

```
> Hi,
>
> I have a small problem concerning a good programming techniques, so here it is:
> -----
> pro test,nx,ny,image
>
> ; Make some random X and Y arrays
> x = randomu(s,1000)
> y = randomu(s,1000)
>
> ; now give arbitrary weights for each point
> w = randomu(s,1000)/100.0
> ; take the square
> w2=2.0*w*w
>
> ; I want to construct a 2-D image with a specified dimension
> image = fltarr(nx,ny)
>
> ; I want to sum up the contribution of each point as Gaussian
> ; with width=w to the image pixels
>
> for i=0, nx-1 do begin
>   xgi = i/float(nx)
>   dx = x-xgi
>   for j=0, ny-1 do begin
>     ygi = j/float(ny)
>     dy = y-ygi
>     dr2 = dx*dx+dy*dy
>     arg = -dr2/w2 > (-20.0) ; to avoid overflows
```

The only thing that statement prevents is underflows. It prevents arg from ever being smaller than -20.0. $\exp(-20.0)$ is an extremely small number, not an extremely large one. Ordinarily, underflows aren't much of a problem.

```
> image[i,j] = total(exp(arg))
```

'arg' is a scalar, so $\exp(\text{arg})$ is also a scalar. Therefore, the `total()` doesn't do anything useful. You could just as well say $\exp(\text{arg})$.

```
> endfor
> endfor
```

```

>
> return
> end
> -----
>
> This is obviously quite unoptimised - two cycles etc. Do you have any ideas, references or do
you know if it is already solved in IDL?
>
> I have looked for something similar in David Fanning pages and IDL astronomical libraries but I
couldn't find something to adapt, maybe I have missed it?

image = exp((x-(indgen(nx)#replicate(1.0/float(nx),ny))^2 - $
(y-replicate(1.0/float(ny),nx)#indgen(ny))^2))/w2)

```

Subject: Re: Simple? problem

Posted by [Jaco van Gorkom](#) on Tue, 09 Apr 2002 17:52:16 GMT

[View Forum Message](#) <> [Reply to Message](#)

"Ivan Valtchanov" <ivanv@discovery.saclay.cea.fr> wrote in message
news:20020409174917.78c62461.ivanv@discovery.saclay.cea.fr...
< skipped code of not quite so simple problem ... >
> This is obviously quite unoptimised - two cycles etc. Do you have any
ideas, references or do you know if it is already solved in IDL?
>
> I have looked for something similar in David Fanning pages and IDL
astronomical libraries but I couldn't find something to adapt, maybe I have
missed it?

Well, I would start by reading the dimensional juggling tutorial on David's
pages
a few times. Then read it again, and practice with the examples.
A nice compromise between the speed penalty for using loops and excessive
memory use for array operations is usually found by vectorizing the inner
loop
only. Thus FOR j=... until ENDFOR could be replaced by something along the
lines of:

```

ygi = findgen(ny)/ny
dy = rebin(y, 1000, ny, /sample) - rebin(reform(ygi, 1, ny), 1000, ny,
/sample)
dr = rebin(dx^2, 1000, ny, /sample) + dy^2
arg = dr^2 / rebin(w2, 1000, ny, /sample) > -20.0
image[i,0] = total(exp(arg), 1) ; the total over only the first dimension
(1000)

```

Probably I mixed up all the rebin statements here, since I have not read the
juggling tutorial in months. But thirty minutes of toying around on the
command
line with help statements usually gets the syntax right.

Jaco

P.S.: It is quite possible that I have misunderstood the meaning or purpose of your code. Maybe you could describe in words exactly what you want to accomplish?

Subject: Re: Simple? problem

Posted by [James Kuyper](#) on Tue, 09 Apr 2002 18:26:30 GMT

[View Forum Message](#) <> [Reply to Message](#)

James Kuyper wrote:

> Ivan Valtchanov wrote:

...

>> x = randomu(s,1000)

>> y = randomu(s,1000)

...

>> dx = x-xgi

...

>> dy = y-ygi

>> dr2 = dx*dx+dy*dy

>> arg = -dr2/w2 > (-20.0) ; to avoid overflows

...

>> image[i,j] = total(exp(arg))

>

>

> 'arg' is a scalar, so exp(arg) is also a scalar. Therefore, the total()

> doesn't do anything useful. You could just as well say exp(arg).

My apologies, I didn't read your code carefully enough. For some reason I was thinking that x and y were scalars.

Forget the rest of my message; I'll need to re-think it, and I'm a little too busy to do that right now. However, I'm pretty sure something a little more complex than what I wrote, but on the same lines, is feasible.

Subject: Re: Simple? problem

Posted by [Ivan Valtchanov](#) on Wed, 10 Apr 2002 08:13:53 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Tue, 9 Apr 2002 19:52:16 +0200

"Jaco van Gorkom" <j.c.van.gorkom@fz-juelich.de> wrote:

>
> P.S.: It is quite possible that I have misunderstood the meaning or purpose
> of
> your code. Maybe you could describe in words exactly what you want to
> accomplish?
>

Thanks for your points. I'll look at the juggling tutorial...

To describe what is my objective: this is an adaptive kernel smoothing, where the width of the kernel is locally variable. I would like to apply it to scattered points and not (for the moment) to images.

Hope this helps?

Ivan
