Subject: FIT across images
Posted by bas.basetta on Fri, 27 Mar 2009 15:28:22 GMT
View Forum Message <> Reply to Message

Hi to everybody.
I used this group so many times as a place where to find solutions to IDL problems, and I think it is maybe the most useful and accurate resource to do that, you are great. I never posted, because most of the questions have already been asked and answered. I'm doing this now because I have one that can be of general interest, and it is about fitting.
Chloe' is asking:

http://groups.google.co.uk/group/comp.lang.idl-pvwave/browse_thread/thread/a4b97b57a2f194f7/

I don't know if mine is the same, but I'm trying to explain it the best I can.

I'm using Craig Markwardt's legendary MPCURVEFIT, and a have a set of n images, each being a sizeXsize matrix. I want to fit an exponential (or in general a function) through those images, i.e. each pixel is fitted across n images to produce an exponential pixel by pixel map. Now - disgusted by myself :) not finding a proper solution - I'm repeating the fit process size*size times:

```
For i=0 to size, For j=0 to size
 fit=mpcurvefit(x,S,W,A,sigma,function_name='expf',chisq=chis q,/noderivative)
```

x is the variable (a parameter which is the same for the whole image) changing across images and S is the luminance of each pixel. x[n], S [n] and W[n] are vectors and have dimension n and A is a vector with dimension [3].

'expf' is defined

```
bx = EXP(-x*A[1])
F = A[0] * bx + A[2]
```

and then I store A for each pixel

```
map=[A[1],i,j]
```

It works fine, but it is very time consuming, and I suspect it is not very efficient in a array-oriented language. I would really like the whole image to be processed at each iteration.

This should be done using:

fit=mpcurvefit(x,S,W,A,sigma,function_name='expmap',chisq=ch isq,/
noderivative)

where x[n] is the same, but now S=[n,size,size], W=[n,size,size] and A=
[3,size,size]

and 'expmap' is defined

bx = EXP(-x*A[1,*,*])
F = A[0,*,*] * bx + A[2,*,*]

In principle X and Y should be of any size, provided that a correct
function is given to map X to Y. 'expmap' is producing F=
[1,size,size].
The procedure "mpcurvefit" does not complain but also does no
iteration, giving me back A unchanged, i.e. initial conditions.
First I though it could be a problem of coherent array's dimension,
and I tried to replicate x[n]-->x[n,size,size], with similar results.
I played with dimensions, and if something is set wrong, it complains.
But I still think that I'm not passing the parameters in the correct
way, because the resulting variable is fit=[n]=S[*,0,0], i.e. the
first point on n images.
Or maybe I'm missing some evident principle because the routine is not
meant to work like that, but I'm realistically thinking that I'm not
able to use it properly. Obviously it would be great also a solution
where I can use S[n,size*size], i.e. a very long array built up with
the image's rows, if the problem is the 3rd dimension.

I'm also trying to figure out if the general principle is not working:
if it is trying to minimize a parameter (chisq) how could it find one
representative of the whole image at each iteration, and meet a
convergence criteria? Maybe that's the point.

I need some help in case I'm using Craig's routine incorrectly, or
some suggestion, because I'm struggling, and maybe I'm missing a BIG
thing, evident for experts like you all are.

Many thanks in advance,
Bas

Subject: Re: FIT across images
Posted by bas.basetta on Wed, 08 Apr 2009 12:23:59 GMT
View Forum Message <> Reply to Message

Hi all,

just to let you know, I now figured it out.

I messed it up with

a. wrong dimensions of “F”, which has to be the same as “fit”
b. bad use of the * “simple” matrix product.

To simplify matrix products I reformed the S matrix as [n,size,size]
to [n,m], where m=size*size, and so A to [3,m]. I then used the
function

```
iv=replicate(1,(size(x,/dimensions))[0])
bx = EXP(-x#A[1,*])
F = (iv#A[0,*]) * bx + iv#A[2,*]
```

where x is now [n] and F is [n,m]. The procedure

```
fit=mpcurvefit(x,dummyS,W,A,sigma,function_name='expmap',chi sq=chisq,/
noderivative,/quiet)
```

now works, with fit=[n,m], and produces the same results as the double
forloop procedure does. The funny thing is: it is one hundred times
slower. Maybe there is a lesson to learn here.

Have fun,
Bas

Subject: Re: FIT across images
Posted by Craig Markwardt on Sun, 12 Apr 2009 23:38:47 GMT
View Forum Message <> Reply to Message

On Apr 8, 8:23 am, bas.base...@virgilio.it wrote:
> Hi all,
> just to let you know, I now figured it out.
>
> I messed it up with
>
> a. wrong dimensions of “F”, which has to be the same as “fit”
> b. bad use of the * “simple” matrix product.
>
> To simplify matrix products I reformed the S matrix as [n,size,size]
> to [n,m], where m=size*size, and so A to [3,m]. I then used the
> function
>
> iv=replicate(1,(size(x,/dimensions))[0])
> bx = EXP(-x#A[1,*])
> F = (iv#A[0,*]) * bx + iv#A[2,*]

>
> where x is now [n] and F is [n,m]. The procedure
>
> fit=mpcurvefit(x,dummyS,W,A,sigma,function_name='expmap',chi sq=chisq,/
> noderivative,/quiet)
>
> now works, with fit=[n,m], and produces the same results as the double
> forloop procedure does. The funny thing is: it is one hundred times
> slower. Maybe there is a lesson to learn here.

Yep, I would have done the fit with a FOR loop. :-)
