Subject: simple deconvolution
Posted by rogass on Tue, 22 Feb 2011 15:00:57 GMT
View Forum Message <> Reply to Message

Hi folks,
I want to implement an image deconvolution into a larger package. The following code performs either the Iterative Wiener (by A.W. Stevenson) or the Richardson-Lucy deconvolution, but both go wrong for the recovery of both smoothed images and smoothed images plus noise . I'm a little bit confused about that. Maybe somebody could help me?
The implemented CONVOLVE comes from the Astrolib. I'm using IDL 8 and the code is not optimised as you can see :)

```
function cr_deconv,im,psf,method,small=small
        sz1 = size(im,/dimensions)
        sz2 = size(psf,/dimensions)
        small=~n_elements(small)?1e-5:small
if total(sz1 eq sz2) ne 0 then begin
        p=fltarr(sz1)
        p[(sz1[0]/2)-(sz2[0]/2) ,(sz1[1]/2)-(sz2[1]/2)]=psf
endif
        p/=total(psf)
        p[where(p lt small)]=small
if method eq 'iwiener' then begin
        psf_fft=fft(p)
        psf_fft[where(abs(psf_fft) lt small)]=small
        snr=mean(median(im,3))/stddev(im-median(im,3)) : snr
        pc=psf_fft*conj(p)
        pc[where(abs(pc) lt small)]=small
        filter=pc
        filter/=(filter+1./snr)
        filter[where(abs(filter) lt small)]=small
        res=abs(fft(filter*fft(im)/psf_fft,/inverse))
for i=0l,iter-1l do begin
        res+=abs(fft((fft(convolve(i eq 0?im:res,p)-im)/psf_fft)*$
        (pc/(pc+(1./snr))),/inverse))
        snr=mean(median(res,3))/stddev(res-median(res,3))
endfor
else begin
         corr_kernel=rot(p,180)
         for i=0l,iter-1l do $
         res=(i eq 0?im:res)*convolve(im/convolve(i eq 0?
im:res,p),corr_kernel)
endelse
return,res
end
```

Thanks in advance

CR

Subject: Re: simple deconvolution
Posted by penteado on Tue, 22 Feb 2011 16:35:30 GMT
View Forum Message <> Reply to Message

On Feb 22, 12:00 pm, chris <rog...@googlemail.com> wrote:
> but both go wrong for
> the recovery of both smoothed images and smoothed images plus noise .
> I'm a little bit confused about that. Maybe somebody could help me?

This is a little vague. "go wrong" can mean almost anything.

Subject: Re: simple deconvolution
Posted by rogass on Tue, 22 Feb 2011 16:49:20 GMT
View Forum Message <> Reply to Message

On 22 Feb., 17:35, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On Feb 22, 12:00 pm, chris <rog...@googlemail.com> wrote:
>
>> but both go wrong for
>> the recovery of both smoothed images and smoothed images plus noise .
>> I'm a little bit confused about that. Maybe somebody could help me?
>
> This is a little vague. "go wrong" can mean almost anything.

Hm, you are right :) The image is not deconvolved but blurred once
more. Thats the problem.

Cheers
CR

Subject: Re: simple deconvolution
Posted by David Fanning on Tue, 22 Feb 2011 17:10:22 GMT
View Forum Message <> Reply to Message

chris writes:

> Hm, you are right :) The image is not deconvolved but blurred once
> more. Thats the problem.

Sometimes the solution is no more complicated than

writing the question. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: simple deconvolution
Posted by wlandsman on Tue, 22 Feb 2011 23:03:02 GMT
View Forum Message <> Reply to Message

On Tuesday, February 22, 2011 11:49:20 AM UTC-5, chris wrote:

> Hm, you are right :) The image is not deconvolved but blurred once
> more. Thats the problem.
>

If I remember correctly, the most difficult part of iterative deconvolution algorithms is knowing when to stop the iterations. If you iterate for too long, the noise gets amplified, and the image actually begins to look worse. I would certainly look at the image after each iteration.

The MaximDL website (a great image processing toolkit BTW) has a nice introduction to iterative deconvolution methods. --Wayne

http://www.cyanogen.com/help/maximdl/Introduction_to_Deconvo lution.htm

---

Subject: Re: simple deconvolution
Posted by James[2] on Tue, 22 Feb 2011 23:15:26 GMT
View Forum Message <> Reply to Message

On Feb 22, 7:00 am, chris <rog...@googlemail.com> wrote:
> Hi folks,
> I want to implement an image deconvolution into a larger package. The
> following code performs either the Iterative Wiener (by A.W.
> Stevenson) or the Richardson-Lucy deconvolution, but both go wrong for
> the recovery of both smoothed images and smoothed images plus noise .
> I'm a little bit confused about that. Maybe somebody could help me?
> The implemented CONVOLVE comes from the Astrolib. I'm using IDL 8 and
> the code is not optimised as you can see :)

It would help a lot if you commented your code. Then we could get an idea of what you're trying to do, and have a better chance of identifying why it doesn't work.

I'll intersperse my comments before the lines of code they refer to.

> function cr_deconv,im,psf,method,small=small
> sz1 = size(im,/dimensions)
> sz2 = size(psf,/dimensions)

This line would be more clear as: if ~keyword_set(small) then small = 1e-5

> small=~n_elements(small)?1e-5:small

I'm not sure why you test for equality in the dimensions here. This code would produce an error or unexpected results if the PSF is bigger than the image: you'd get negative indices. It looks like you're trying to say 'if the PSF and the image aren't the same size, make a new array the size of the image with the psf centered in it.' Is this correct?

> if total(sz1 eq sz2) ne 0 then begin
> p=fltarr(sz1)
> p[(sz1[0]/2)-(sz2[0]/2) ,(sz1[1]/2)-(sz2[1]/2)]=psf
> endif

This next line would produce an error if the previous if..then block didn't run. P would be an undefined variable.

> p/=total(psf)
> p[where(p lt small)]=small
> if method eq 'iwiener' then begin
> psf_fft=fft(p)

The next line throws away the sign: small negative values are changed to a small positive value. I don't know if this matters, but it seems like it might.

> psf_fft[where(abs(psf_fft) lt small)]=small

Is the colon at the end of this line supposed to be a semicolon, for a comment?

> snr=mean(median(im,3))/stddev(im-median(im,3)) : snr
> pc=psf_fft*conj(p)

Same thing with the small negative values here...

```
>           pc[where(abs(pc) lt small)]=small
>           filter=pc
>           filter/=(filter+1./snr)
```

... and here.

```
>           filter[where(abs(filter) lt small)]=small
>           res=abs(fft(filter*fft(im)/psf_fft,/inverse))
>  for i=0l,iter-1l do begin
>           res+=abs(fft((fft(convolve(i eq 0?im:res,p)-im)/psf_fft)*$
>           (pc/(pc+(1./snr))),/inverse))
>           snr=mean(median(res,3))/stddev(res-median(res,3))
>  endfor
>  else begin
>            corr_kernel=rot(p,180)
>            for i=0l,iter-1l do $
>            res=(i eq 0?im:res)*convolve(im/convolve(i eq 0?
>  im:res,p),corr_kernel)
>  endelse
>  return,res
>  end
```

I don't know enough about deconvolution algorithms to help with the parts inside the for loops. But it seems possible that a programming error is causing problems, rather than an incorrect approach mathematically.

Also, I think you should ease up on the ternary (? :) operator a little bit. It's useful for making concise expressions now and then, but in general the if..then..else block makes more understandable code.

Subject: Re: simple deconvolution
Posted by David Fanning on Tue, 22 Feb 2011 23:24:07 GMT
View Forum Message <> Reply to Message

James writes:

```
>  Also, I think you should ease up on the ternary (? :) operator a
>  little bit.  It's useful for making concise expressions now and then,
>  but in general the if..then..else block makes more understandable code.
```

I braved a += operator in an e-mail the other day where I thought the context would make its use totally transparent. Total chaos. I probably won't

do that again for another couple of years. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: simple deconvolution
Posted by James[2] on Tue, 22 Feb 2011 23:49:03 GMT
View Forum Message <> Reply to Message

On Feb 22, 3:24 pm, David Fanning <n...@idlcoyote.com> wrote:
> James writes:
>> Also, I think you should ease up on the ternary (? :) operator a
>> little bit. It's useful for making concise expressions now and then,
>> but in general the if..then..else block makes more understandable code.
>
> I braved a += operator in an e-mail the other day
> where I thought the context would make its use
> totally transparent. Total chaos. I probably won't
> do that again for another couple of years. :-)
>
> Cheers,
>
> David

Oh man, if += is wrong then I don't wanna be right.

Subject: Re: simple deconvolution
Posted by rogass on Wed, 23 Feb 2011 07:40:51 GMT
View Forum Message <> Reply to Message

Dear James, David and Wayne,
thank you for your comments. Nevertheless, there is still a
methodological bug in the code which I can't find. As I stated before,
this code snippet is in early stage, so don't wonder if it is not so
easy readable or optimised due to error catching and computational
speed. I tried several routines freely available like the deconv_tool
from F. Varosi and from you Wayne :), but they all run into problems

if the SNR is low. Unfortunately, the MaximDL package seems to be only commercially available. However, I'm looking for a routine like the Richardson-Lucy algorithm which might be appropiate for a multiscale deconvolution to suppress ringing effects.

Anyway, thank you.

Maybe someone is able to catch the error in the code? ;)

Cheers

CR

Subject: Re: simple deconvolution
Posted by Jeremy Bailin on Wed, 23 Feb 2011 14:53:54 GMT
View Forum Message <> Reply to Message

> I braved a += operator in an e-mail the other day
> where I thought the context would make its use
> totally transparent. Total chaos. I probably won't
> do that again for another couple of years. :-)

Seriously?? I think += is much much clearer than the alternative.

-Jeremy.

Subject: Re: simple deconvolution
Posted by David Fanning on Wed, 23 Feb 2011 15:30:12 GMT
View Forum Message <> Reply to Message

Jeremy Bailin writes:

> Seriously?? I think += is much much clearer than the alternative.

I think programmers of a certain age are probably not
used to it. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.

Subject: Re: simple deconvolution
Posted by penteado on Wed, 23 Feb 2011 15:45:16 GMT
View Forum Message <> Reply to Message

On Feb 23, 11:53 am, Jeremy Bailin <astroco...@gmail.com> wrote:
>> I braved a += operator in an e-mail the other day
>> where I thought the context would make its use
>> totally transparent. Total chaos. I probably won't
>> do that again for another couple of years. :-)
>
> Seriously?? I think += is much much clearer than the alternative.
>
> -Jeremy.

Same here. Also for ? : and ++. Since their meaning is more specific, just reading them is easier than the alternatives, which require more interpreting. The same way a foreach is easier to interpret because it is not as open in possibilities as a for.

On a side note, in IDL ++ and -- are curious because they are the only cases of an expression that can return a value.

Subject: Re: simple deconvolution
Posted by pgrigis on Wed, 23 Feb 2011 16:14:53 GMT
View Forum Message <> Reply to Message

On Feb 22, 10:00 am, chris <rog...@googlemail.com> wrote:
> Hi folks,
> I want to implement an image deconvolution into a larger package. The
> following code performs either the Iterative Wiener (by A.W.
> Stevenson) or the Richardson-Lucy deconvolution, but both go wrong for
> the recovery of both smoothed images and smoothed images plus noise .
> I'm a little bit confused about that. Maybe somebody could help me?
> The implemented CONVOLVE comes from the Astrolib. I'm using IDL 8 and
> the code is not optimised as you can see :)
>
> function cr_deconv,im,psf,method,small=small
> sz1 = size(im,/dimensions)
> sz2 = size(psf,/dimensions)
> small=~n_elements(small)?1e-5:small
> if total(sz1 eq sz2) ne 0 then begin
> p=fltarr(sz1)

```
>           p[(sz1[0]/2)-(sz2[0]/2) ,(sz1[1]/2)-(sz2[1]/2)]=psf
>  endif
>           p/=total(psf)
>           p[where(p lt small)]=small
>  if method eq 'iwiener' then begin
>           psf_fft=fft(p)
>           psf_fft[where(abs(psf_fft) lt small)]=small
>           snr=mean(median(im,3))/stddev(im-median(im,3)) : snr
>           pc=psf_fft*conj(p)
>           pc[where(abs(pc) lt small)]=small
>           filter=pc
>           filter/=(filter+1./snr)
>           filter[where(abs(filter) lt small)]=small
>           res=abs(fft(filter*fft(im)/psf_fft,/inverse))
>  for i=0l,iter-1l do begin
>           res+=abs(fft((fft(convolve(i eq 0?im:res,p)-im)/psf_fft)*$
>           (pc/(pc+(1./snr))),/inverse))
>           snr=mean(median(res,3))/stddev(res-median(res,3))
>  endfor
>  else begin
>            corr_kernel=rot(p,180)
>            for i=0l,iter-1l do $
>            res=(i eq 0?im:res)*convolve(im/convolve(i eq 0?
>  im:res,p),corr_kernel)
>  endelse
>  return,res
>  end
>
>  Thanks in advance
>
>  CR
```

My understanding is that the Richardson-Lucy algorithm
works as follows.

Given an Image IM and a point-spread function PSF.

Initialization:
O=IM

Loop:
IHAT=CONV(PSF,O)
O=O*CORR(IM/IHAT,PSF)

After somewhere between 10 to 50 iterations, O is going to
be an approximation to the the deconvolved version of IM.

Here CONV and CORR are the usual convolution and correlation functions. Some care need to be taken with normalization, but this is the skeleton of the algorithm.

I do not see that your algorithm is performing this operation, or is it? Also you may want to implement the convolutions and correlations manually yourself using FFT - this way you have more control over what is happening.

Ciao,
Paolo

Subject: Re: simple deconvolution
Posted by rogass on Wed, 23 Feb 2011 20:32:13 GMT
View Forum Message <> Reply to Message

On 23 Feb., 17:14, Paolo <pgri...@gmail.com> wrote:
> On Feb 22, 10:00 am, chris <rog...@googlemail.com> wrote:
>
>
>
>
>
>
>
>
>
>
>> Hi folks,
>> I want to implement an image deconvolution into a larger package. The
>> following code performs either the Iterative Wiener (by A.W.
>> Stevenson) or the Richardson-Lucy deconvolution, but both go wrong for
>> the recovery of both smoothed images and smoothed images plus noise .
>> I'm a little bit confused about that. Maybe somebody could help me?
>> The implemented CONVOLVE comes from the Astrolib. I'm using IDL 8 and
>> the code is not optimised as you can see :)
>
>> function cr_deconv,im,psf,method,small=small
>> sz1 = size(im,/dimensions)
>> sz2 = size(psf,/dimensions)
>> small=~n_elements(small)?1e-5:small
>> if total(sz1 eq sz2) ne 0 then begin
>> p=fltarr(sz1)
>> p[(sz1[0]/2)-(sz2[0]/2) ,(sz1[1]/2)-(sz2[1]/2)]=psf
>> endif
>> p/=total(psf)
>> p[where(p lt small)]=small
>> if method eq 'iwiener' then begin

```
>>          psf_fft=fft(p)
>>          psf_fft[where(abs(psf_fft) lt small)]=small
>>          snr=mean(median(im,3))/stddev(im-median(im,3)) : snr
>>          pc=psf_fft*conj(p)
>>          pc[where(abs(pc) lt small)]=small
>>          filter=pc
>>          filter/=(filter+1./snr)
>>          filter[where(abs(filter) lt small)]=small
>>          res=abs(fft(filter*fft(im)/psf_fft,/inverse))
>>  for i=0l,iter-1l do begin
>>          res+=abs(fft((fft(convolve(i eq 0?im:res,p)-im)/psf_fft)*$
>>          (pc/(pc+(1./snr))),/inverse))
>>          snr=mean(median(res,3))/stddev(res-median(res,3))
>>  endfor
>>  else begin
>>           corr_kernel=rot(p,180)
>>           for i=0l,iter-1l do $
>>           res=(i eq 0?im:res)*convolve(im/convolve(i eq 0?
>>  im:res,p),corr_kernel)
>>  endelse
>>  return,res
>>  end
>
>>  Thanks in advance
>
>>  CR
>
>  My understanding is that the Richardson-Lucy algorithm
>  works as follows.
>
>  Given an Image IM and a point-spread function PSF.
>
>  Initialization:
>  O=IM
>
>  Loop:
>  IHAT=CONV(PSF,O)
>  O=O*CORR(IM/IHAT,PSF)
>
>  After somewhere between 10 to 50 iterations, O is going to
>  be an approximation to the the deconvolved version of IM.
>
>  Here CONV and CORR are the usual convolution and correlation
>  functions. Some care need to be taken with normalization, but
>  this is the skeleton of the algorithm.
>
>  I do not see that your algorithm is performing this operation,
>  or is it? Also you may want to implement the convolutions and
```

> correlations manually yourself using FFT - this way you have
> more control over what is happening.
>
> Ciao,
> Paolo

Dear Paolo,
you enlightened me :). A related code snippet which works for the RL
is:

```
o=im & conp=conj(psf) & psf2=fft(psf)
for i=0l,iter-1l do $
 o=o*convolve(im/convolve(o,psf,ft_psf=psf2),psf,ft_psf=conp)
```

The correlation is performed by convolving with the conjugate PSF.

THANK YOU

CR
