Subject: Re: 2D Savitzky-Golay derivative filter?
Posted by David Fanning on Sun, 03 Feb 2013 19:08:36 GMT
View Forum Message <> Reply to Message

David Grier writes:

> I would like to use Savitzky-Golay filters to calculate gradients of images.
> Before I roll my own code to calculate the filter coefficients, I was wondering
> if anyone here already had working routines that they would be willing to share.
>
> IDL's built-in SAVGOL routine only computes one-dimensional filters, and I need
> two-dimensional filters. Erik Rosolowsky's SAVGOL2D computes two-dimensional
> smoothing filters, but not two-dimensional derivative filters.
>
> Alternatively, is there an equivalently good IDL way to compute image gradients?
> A useful replacement would offer the noise rejection of Savitzky-Golay without
> suppressing peaks the way that SMOOTH() does.

The Sobel and Roberts functions are both 2D gradient or derivative filters, or you can just roll your own and do some kind of Laplacian filter:

http://www.idlcoyote.com/ip_tips/sharpen.html

Cheers,

David

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

Subject: Re: 2D Savitzky-Golay derivative filter?
Posted by dg86 on Sun, 03 Feb 2013 21:37:11 GMT
View Forum Message <> Reply to Message

Thanks David for following up.

Sobel and Roberts both use nearest-neighbor estimates for derivatives, and thus do a bad job of computing gradients in noisy images. Savitzky-Golay does a much better job at rejecting noise because it uses information over a larger domain. It might seem tempting just to smooth the image to suppress noise before using a simple derivative operator. The problem is that smoothing suppresses real features along with the noise. Savitzky-Golay is much better at preserving features such as peaks and ridges.

So, unless somebody already has an implementation, I might spend a couple of days rolling my own.

All the best,

David

Subject: Re: 2D Savitzky-Golay derivative filter?
Posted by dg86 on Thu, 07 Feb 2013 02:19:01 GMT
View Forum Message <> Reply to Message

Dear Folks,

I append a routine that computes two-dimensional Savitzky-Golay filters for smoothing and taking derivatives of images. This is based substantially on Erik Rosolowsky's savgol2d() routine, with some code simplification and the addition of capabilities for computing derivatives of specified order along each direction.

The benefit of Savitzky-Golay filters for image analysis is that they suppress noise while retaining features of interest such as peaks and ridges. They therefore are particularly useful for computing gradients of images with additive noise.

Comments and suggestions are warmly solicited.

All the best,

David

```
;+
; NAME:
;    savgol2d()
;
; PURPOSE:
;    Calculate two-dimensional Savitzky-Golay filters for smoothing images or
;    computing their derivatives.
;
; CALLING SEQUENCE:
;    filter = savgol2d(dim, order)
;
; INPUTS:
;    dim: width of the filter [pixels]
;    order: The degree of the polynomial
;
; KEYWORD PARAMETERS:
;    dx: order of the derivative to compute in the x direction
;        Default: 0 (no derivative)
```

```
;    dy: order of derivative to compute in the y direction
;        Default: 0 (no derivative)
;
; OUTPUTS:
;    filter: [dim,dim] Two-dimensional Savitzky-Golay filter
;
; EXAMPLE:
; IDL> dadx = convol(a, savgol2d(11, 6, dx = 1))
;
; MODIFICATION HISTORY:
;  Algorithm based on SAVGOL2D:
;  Written and documented
;  Fri Apr 24 13:43:30 2009, Erik Rosolowsky <erosolo@A302357>
;
;  02/06/2013 Revised version by David G. Grier, New York University
;-

function savgol2d, dim, order, dx = dx, dy = dy

COMPILE_OPT IDL2

umsg = 'USAGE: filter = dgsavgol2d(dim, order)'

if n_params() ne 2 then begin
   message, umsg, /inf
   return, -1
endif
if ~isa(dim, /scalar, /number) then begin
   message, umsg, /inf
   message, 'DIM should be the integer width of the filter', /inf
   return, -1
endif

if ~isa(order, /scalar, /number) then begin
   message, umsg, /inf
   message, 'ORDER should be the integer order of the interpolaying polynomial', /inf
   return, -1
endif
if ~(order lt dim) then begin
   message, umsg, /inf
   message, 'ORDER should be less than DIM', /inf
   return, -1
endif

if ~isa(dx, /scalar, /number) then dx = 0
if ~isa(dy, /scalar, /number) then dy = 0
if dx lt 0 or dy lt 0 then begin
   message, umsg, /inf
```

```
   message, 'DX and DY should be non-negative integers', /inf
   return, -1
endif
if (dx + dy ge order) then begin
   message, umsg, /inf
   message, 'DX + DY should not be greater than ORDER', /inf
   return, -1
endif

npts = dim^2

x = rebin(findgen(dim)-dim/2, dim, dim)
y = transpose(x)
x = reform(x, npts)
y = reform(y, npts)

Q = findgen((order+1)*(order+2)/2, npts)

n = 0
for nu = 0, order do begin
   ynu =  y^nu
   for mu = 0, order-nu do begin
      Q[n++, *] = x^mu * ynu
   endfor
endfor

a = transpose(invert(Q # transpose(Q)) # Q)
filter = fltarr(npts)
b = [1., fltarr(npts-1)]
ndx = dx + (order + 1) * dy
for i = 0, npts-1 do begin
   filter[i] = (a ## b)[ndx]
   b = shift(b, 1)
endfor

return, reform(filter, dim, dim)
end
```