

THE MAXPOL USERS' GUIDE

Version 1.0

by

Mahdi S. Hosseini
Konstantinos N. Plataniotis

The Edward S. Rogers Sr.
Department of Electrical & Computer Engineering (ECE)
University of Toronto (UofT)

© Copyright 2017 by Mahdi S. Hosseini

Contents

1	Introduction	1
1.1	What is MaxPol Package?	1
1.2	Which disciplinary fields can use this package?	2
1.3	Licensing	2
2	Installation	3
2.1	Supported Platforms	3
2.2	Third-party codes included in MaxPol Package	3
3	The Basics	4
3.1	Parameter Description	4
3.2	FIR Derivative kernels	5
3.3	Derivative matrix	9
3.4	2D Directional (Steerable) Derivatives	10
3.5	How MaxPol generalizes the literature methods?	12
4	Demo Examples: A Quick Start	13
4.1	FIR frequency responses	13
4.2	Spectral analysis of derivative matrix	14
4.3	Directional (steerable) 2D derivatives	14
4.4	Forward signal and imaging problems	14
4.4.1	One dimensional (1D) signal	14
4.4.2	Two dimensional (2D) image/surface	15
4.4.3	Canny edge detection	15
4.4.4	Image Sharpening	15
4.4.5	Three dimensional (3D) Volume	16
4.5	Inverse imaging problems	16
5	Citing MaxPol	17

Chapter 1

Introduction

1.1 What is MaxPol Package?

MaxPol is an open source library codes written in MATLAB offers a comprehensive tool for numerical differentiation. The method is based on undetermined coefficients and maxflat design technique to render variety of FIR kernels in closed form that can be used to approximate fullband/lowpass derivatives of a given discrete signals and images.

The term “*fullband*” is the same terminology used in numerical difference methods for high-order-of-accuracy differentiation, meaning that the filter response (aka transform function) is obedient to the ideal derivative response. The term “*lowpass*” is used when a cutoff frequency is set to truncate the filter response beyond a certain level, that is usually assumed to contain noise/aliasing artifacts. While you can set a cutoff threshold to suppress the noise on the stop-band with free-residues, the differentiator is highly accurate on the pass-band.

MaxPol V1.0 supports a number of utilities including: (a) FIR kernels up to any order of differentiation in both centralized and staggered scheme, (b) arbitrary side-shift nodes for discrete boundary formulation, (b) derivative matrix for global estimation of derivatives of vector-valued (discrete) signals, (c) tensor mode decomposition in any order of dimension e.g. 2D images and 3D volumes, and (d) two dimensional (2D) directional (steerable) derivative kernels for image decomposition.

To use the MaxPol package effectively and how it compares with other existing methods in the literature, please refer to [1,2] for more information. The package contains intuitive examples from diverse applications in signal and image processing.

1.2 Which disciplinary fields can use this package?

MaxPol package provides comprehensive numerical solutions for uniform discrete differentiation that can be of interest to broad audiences in engineering and science.

In the context of image processing, the package can be used in diverse imaging applications. Basically, whenever you need to compute discrete derivatives of an image in the form of gradient, Hessian, or even high order tensor, this package can be easily used for accurate approximations. In forward imaging problems **MaxPol** can be used to estimate high order feature moments applied in edge detection, curvature estimation, image enhancement, and image sharpening. **MaxPol** can be also used in inverse problems for image restoration purposes such as in gradient surface reconstruction, image stitching, image diffusion, variational regularization problems applied in image in-painting, enhancement/de-noising, deconvolution problems, and many more.

The package can be also utilized as a numerical solver to the PDE problems applied in fluid mechanics, acoustics, and wave equations. It can be of interest in dynamic control systems to directly estimate state variables to avoid noise robust numerical integrations.

1.3 Licensing

MaxPol is an open source MATLAB library codes. It is intended to be free for academic research and teaching purposes. **MaxPol** is covered under GNU General Public License.

Copyright © 2017 Mahdi S. Hosseini

MaxPol is a free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Please contact mahdi.hosseini@mail.utoronto.ca for more information.

Chapter 2

Installation

2.1 Supported Platforms

MaxPol version 1.0 is a package contains several MATLAB m-file function codes. It can be run in many platforms such as Windows, Mac, and Linux. You will be needing to install MATLAB software in your PC and run `maxpol_startup.m` file before you use the library.

2.2 Third-party codes included in MaxPol Package

The software package includes third party codes namely, Savtizky-Golay differentiators available from ¹ introduced in [3–7] and Lagrangian differentiators available from ² introduced in [8,9] . These functions are used in several demo examples to compare their performance with MaxPol differentiators.

¹<https://www.mathworks.com/matlabcentral/fileexchange/4038-savitzky-golay-smoothing-and-differentiation-filter>

²<http://faculty.washington.edu/rjl/fdmbook/matlab/fdcoeffF.m>

Chapter 3

The Basics

3.1 Parameter Description

The parameters used within the **MaxPol** MATLAB functions are described as follows:

l: order which defines the tap-length of FIR derivative filter that is **2l+1** for centralized and **2l** for staggered scheme.

n: order of differentiation.

P: degree of maximally polynomial at zero frequency of FIR derivative filter. Another interpretation of this parameter is it controls the cutoff frequency level of the filter from the lowest possible value $n \leq P$ all the way to its maximum level i.e. $P \leq 2l$ for centralized and $P \leq 2l-1$ for staggered scheme. The maximum level in fact corresponds to the fullband response (relaxed cutoff).

s: amount of side-shift of the filter. When **s=0** is selected zero then there is no side shift and it implies that the FIR kernel approximates the derivative at the center of the filter. With positive and negative non-zero side-shifts, the filter approximates the derivative towards the left and right boundaries, respectively.

nod: accepts either '**centralized**' or '**staggered**' to determine the difference scheme.

sym_flag: accepts either '**true**' or '**false**' which determines the method of FIR coefficient calculation. When the flag is turned on then symbolic calculation is performed to compute equations (11) and (12) from [1]. When the flag is turned off then numerical computations are performed to compute the coefficients from the same equations. In case of fullband filter design, i.e. $P=2l$ for centralized and $P=2l-1$ for staggered, the closed form solutions from Theorems 3 and 4 from [1] are recalled. Note that the symbolic calculation requires high computational effort in computer.

sparse_flag: accepts either **true** or **false** for sparse restoration of the derivative matrix.

theta: direction/angle in radians for directional image differentiation

3.2 FIR Derivative kernels

Two main MATLAB functions are explained here to produce FIR derivative coefficients in centralized and staggered schemes. They read as follows

```
[c] = derivcent(l, P, s, n, sym_flag)
```

This function generates centralized derivative coefficients with tap length of $2l+1$. With different combinations of l and P you can generate fullband or lowpass coefficients. Our recommendation is to turn on the symbolic flag for lowpass differentiation $P < 2l+1$ to avoid numerical perturbations of coefficients. If you are using this for fullband differentiation $P = 2l+1$ then you can turnoff this flag to avoid delay time for symbolic processing and maintain high numerical precision.

Few examples of the centralized FIR kernels are presented in Tables 3.1, 3.2, and 3.4 for zero, first, and second order derivatives.

```
[c] = derivstag(l, P, s, n, sym_flag)
```

This function generates staggered derivative coefficients with tap length of $2l$. With different combinations of l and P you can generate fullband or lowpass coefficients. Our recommendation is to turn on the symbolic flag for lowpass differentiation $P < 2l-1$ to avoid numerical perturbations of coefficients. If you are using this for fullband differentiation $P = 2l-1$ then you can turnoff this flag to avoid delay time for symbolic processing and maintain high numerical precision.

Few examples of the staggered FIR kernels are presented in Tables 3.1, 3.3, and 3.5 for zero, first, and second order derivatives.

Without loss of generality, these two MATLAB functions can be easily called to produce arbitrary derivative orders such as first, second, third, fourth, etc. Arbitrary polynomial degrees can be set to generate filters. With ordinary desktop computers, you can achieve filter tap-lengths of < 100 with symbolic calculations which is much higher than what we usually select for signal/image processing tasks. Moreover, the filter can be generated in arbitrary side-shift nodes for all fullband and lowpass differentiation. For more information on the filter frequency response please refer to Figures 3 and 4 in [1].

Table 3.1: List of examples for zero order centralized FIR derivative coefficients, here $s=0$. The term zero order differentiation is equivalent to lowpass/interpolating filter.

n	l	P	c (centralized)
0	1	0	$[1, 2, 1]/4$
0	1	2	$[0, 1, 0]$
0	2	0	$[1, 4, 6, 4, 1]/16$
0	2	2	$[-1, 4, 10, 4, -1]/16$
0	2	4	$[0, 0, 1, 0, 0]$
0	3	0	$[1, 6, 15, 20, 15, 6, 1]/64$
0	3	2	$[-1, 0, 9, 16, 9, 0, -1]/32$
0	3	4	$[1, -6, 15, 44, 15, -6, 1]/64$
0	3	6	$[0, 0, 0, 1, 0, 0, 0]$
0	4	0	$[1, 8, 28, 56, 70, 56, 28, 8, 1]/256$
0	4	2	$[-3, -8, 12, 72, 110, 72, 12, -8, -3]/256$
0	4	4	$[3, -8, -12, 72, 146, 72, -12, -8, 3]/256$
0	4	6	$[-1, 8, -28, 56, 186, 56, -28, 8, -1]/256$
0	4	8	$[0, 0, 0, 0, 1, 0, 0, 0, 0]$
0	5	0	$[1, 10, 45, 120, 210, 252, 210, 120, 45, 10, 1]/1024$
0	5	2	$[-1, -5, -5, 20, 70, 98, 70, 20, -5, -5, -1]/256$
0	5	4	$[3, 0, -25, 0, 150, 256, 150, 0, -25, 0, 3]/512$
0	5	6	$[-1, 5, -5, -20, 70, 158, 70, -20, -5, 5, -1]/256$
0	5	8	$[1, -10, 45, -120, 210, 772, 210, -120, 45, -10, 1]/1024$
0	5	10	$[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]$

n	l	P	c (staggered)
0	1	1	$[1, 1]/2$
0	2	1	$[1, 3, 3, 1]/8$
0	2	3	$[-1, 9, 9, -1]/16$
0	3	1	$[1, 5, 10, 10, 5, 1]/32$
0	3	3	$[-3, 5, 30, 30, 5, -3]/64$
0	3	5	$[3, -25, 150, 150, -25, 3]/256$
0	4	1	$[1, 7, 21, 35, 35, 21, 7, 1]/128$
0	4	3	$[-5, -7, 35, 105, 105, 35, -7, -5]/256$
0	4	5	$[15, -63, 35, 525, 525, 35, -63, 15]/1024$
0	4	7	$[-5, 49, -245, 1225, 1225, -245, 49, -5]/2048$
0	5	1	$[1, 9, 36, 84, 126, 126, 84, 36, 9, 1]/512$
0	5	3	$[-7, -27, 0, 168, 378, 378, 168, 0, -27, -7]/1024$
0	5	5	$[35, -45, -252, 420, 1890, 1890, 420, -252, -45, 35]/4096$
0	5	7	$[-35, 225, -504, 0, 4410, 4410, 0, -504, 225, -35]/8192$
0	5	9	$[35, -405, 2268, -8820, 39690, 39690, -8820, 2268, -405, 35]/65536$

Table 3.2: List of examples for first order centralized FIR derivative coefficients, here $s=0$.

n	l	P	c (centralized)
1	1	2	$[-1, 0, 1]/2$
1	2	2	$[-1, -2, 0, 2, 1]/8$
1	2	4	$[1, -8, 0, 8, -1]/12$
1	3	2	$[-1, -4, -5, 0, 5, 4, 1]/32$
1	3	4	$[5, -12, -39, 0, 39, 12, -5]/96$
1	3	6	$[-1, 9, -45, 0, 45, -9, 1]/60$
1	4	2	$[-1, -6, -14, -14, 0, 14, 14, 6, 1]/128$
1	4	4	$[2, 1, -16, -27, 0, 27, 16, -1, -2]/96$
1	4	6	$[-33, 166, -174, -978, 0, 978, 174, -166, 33]/1920$
1	4	8	$[3, -32, 168, -672, 0, 672, -168, 32, -3]/840$
1	5	2	$[-1, -8, -27, -48, -42, 0, 42, 48, 27, 8, 1]/512$
1	5	4	$[11, 32, -39, -256, -322, 0, 322, 256, 39, -32, -11]/1536$
1	5	6	$[-73, 160, 445, -1280, -2890, 0, 2890, 1280, -445, -160, 73]/7680$
1	5	8	$[279, -2040, 5485, -2640, -31290, 0, 31290, 2640, -5485, 2040, -279]/53760$
1	5	10	$[-2, 25, -150, 600, -2100, 0, 2100, -600, 150, -25, 2]/2520$

Table 3.3: List of examples for first order staggered FIR derivative coefficients, here $s=0$.

n	l	P	c (staggered)
1	1	1	$[-1, 1]$
1	2	1	$[-1, -1, 1, 1]/4$
1	2	3	$[1, -27, 27, -1]/24$
1	3	1	$[-1, -3, -2, 2, 3, 1]/16$
1	3	3	$[7, -31, -38, 38, 31, -7]/96$
1	3	5	$[-9, 125, -2250, 2250, -125, 9]/1920$
1	4	1	$[-1, -5, -9, -5, 5, 9, 5, 1]/64$
1	4	3	$[13, -11, -111, -87, 87, 111, 11, -13]/384$
1	4	5	$[-149, 1007, -2629, -3785, 3785, 2629, -1007, 149]/7680$
1	4	7	$[75, -1029, 8575, -128625, 128625, -8575, 1029, -75]/107520$
1	5	1	$[-1, -7, -20, -28, -14, 14, 28, 20, 7, 1]/256$
1	5	3	$[19, 33, -120, -368, -234, 234, 368, 120, -33, -19]/1536$
1	5	5	$[-409, 1449, 756, -10516, -9414, 9414, 10516, -756, -1449, 409]/30720$
1	5	7	$[2161, -19149, 73680, -147224, -242214, 242214, 147224, -73680, 19149, -2161]/430080$
1	5	9	$[-1225, 18225, -142884, 926100, -12502350, 12502350, -926100, 142884, -18225, 1225]/10321920$

Table 3.4: List of examples for second order centralized FIR derivative coefficients, here $s=0$.

n	l	P	c (centralized)
2	1	2	[1,-2,1]
2	2	2	[1,0,-2,0,1]/4
2	2	4	[-1,16,-30,16,-1]/12
2	3	2	[1,2,-1,-4,-1,2,1]/16
2	3	4	[-1,5,1,-10,1,5,-1]/12
2	3	6	[2,-27,270,-490,270,-27,2]/180
2	4	2	[1,4,4,-4,-10,-4,4,1]/64
2	4	4	[-7,12,52,-12,-90,-12,52,12,-7]/192
2	4	6	[17,-128,368,128,-770,128,368,-128,17]/720
2	4	8	[-9,128,-1008,8064,-14350,8064,-1008,128,-9]/5040
2	5	2	[1,6,13,8,-14,-28,-14,8,13,6,1]/256
2	5	4	[-5,-4,39,64,-34,-120,-34,64,39,-4,-5]/384
2	5	6	[173,-766,201,4504,-374,-7476,-374,4504,201,-766,173]/11520
2	5	8	[-32,311,-1312,2832,1344,-6286,1344,2832,-1312,311,-32]/5040
2	5	10	[8,-125,1000,-6000,42000,-73766,42000,-6000,1000,-125,8]/25200

Table 3.5: List of examples for second order staggered FIR derivative coefficients, here $s=0$.

n	l	P	c (staggered)
2	2	3	[1,-1,-1,1]/2
2	3	3	[1,1,-2,-2,1,1]/8
2	3	5	[-5,39,-34,-34,39,-5]/48
2	4	3	[1,3,1,-5,-5,1,3,1]/32
2	4	5	[-11,35,57,-81,-81,57,35,-11]/192
2	4	7	[259,-2495,11691,-9455,-9455,11691,-2495,259]/11520
2	5	3	[1,5,8,0,-14,-14,0,8,5,1]/128
2	5	5	[-17,7,140,92,-222,-222,92,140,7,-17]/768
2	5	7	[919,-5397,8400,21032,-24954,-24954,21032,8400,-5397,919]/46080
2	5	9	[-3229,37107,-204300,745108,-574686,-574686,745108,-204300,37107,-3229]/645120

3.3 Derivative matrix

The MATLAB function to produce derivative matrix (square size) is explained for all four (4) possible cases introduced in [1]. The function reads as follows.

```
[D, D_forward] = derivmtx(1, P, n, N, nod, sparse_flag, sym_flag)
```

This function generates derivative matrix of square size N. With different combinations of 1 and P you can derive fullband or lowpass coefficients embedded in the matrix rows. Our recommendation is to turn on the symbolic flag for lowpass differentiation to avoid numerical perturbations of coefficients. If you are using this for fullband differentiation then you can turnoff this flag to avoid delay time for symbolic processing and maintain high numerical precision. Two examples are provides to generate fullband derivative matrices where 1=3, n=1, N=8.

First order derivative matrix with **nod='staggered'** nodes and P=5,

$$\overleftarrow{D}_{1,8} = \frac{1}{1920} \begin{bmatrix} -9129 & 26765 & -34890 & 25770 & -10205 & 1689 & 0 & 0 \\ -1689 & 1005 & 1430 & -1110 & 435 & -71 & 0 & 0 \\ 71 & -2115 & 2070 & 10 & -45 & 9 & 0 & 0 \\ -9 & 125 & -2250 & 2250 & -125 & 9 & 0 & 0 \\ 0 & -9 & 125 & -2250 & 2250 & -125 & 9 & 0 \\ 0 & 0 & -9 & 125 & -2250 & 2250 & -125 & 9 \\ 0 & 0 & -9 & 45 & -10 & -2070 & 2115 & -71 \\ 0 & 0 & 71 & -435 & 1110 & -1430 & -1005 & 1689 \end{bmatrix}$$

First order derivative matrix with **nod='centralized'** nodes and P=6,

$$D_{1,8} = \frac{1}{60} \begin{bmatrix} -121 & 204 & -60 & -120 & 165 & -84 & 16 & 0 \\ -16 & -41 & 60 & 20 & -40 & 21 & -4 & 0 \\ 4 & -36 & -5 & 40 & 0 & -4 & 1 & 0 \\ -1 & 9 & -45 & 0 & 45 & -9 & 1 & 0 \\ 0 & -1 & 9 & -45 & 0 & 45 & -9 & 1 \\ 0 & -1 & 4 & 0 & -40 & 5 & 36 & -4 \\ 0 & 4 & -21 & 40 & -20 & -60 & 41 & 16 \\ 0 & -16 & 84 & -165 & 120 & 60 & -204 & 121 \end{bmatrix}$$

For more information on the generation of these matrices and their spectral response please refer to [1].

3.4 2D Directional (Steerable) Derivatives

The MATLAB function to produce 2D directional derivative kernel is explained for all possible selections introduced in [2]. The function reads as follows.

```
[K, bases] = derivdirec(1, P_x, P_y, n, theta, sym_flag)
```

















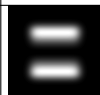






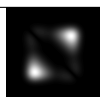










This function generates steerable derivative kernel of square size $(2l+1) \times (2l+1)$. The decomposing bases are based on MaxPol derivative kernels. The function accepts the tap-length polynomial of the filter l and maximally polynomial degrees P_x , P_y on both x- and y- axes. These degrees corresponds to the cutoff level of each axes. The steering angle of the kernel is determined by `theta` in radians.

Tables 3.6 and 3.7 display few filter responses of the directional derivative kernels generated from this MATLAB function for their corresponding selected parameters. As per the maxpol degree P increases, the cutoff threshold increases as a correspondence transform wider range of frequency band for differentiation.

Table 3.6: 2D-convolution kernel $K_{2D}(n, \theta, P_x, P_y)$ for arbitrary order of differentiation n , steering angle θ , and cutoff frequency controlled by flatness degree P_x and P_y . Selected parameters here are $l=13$, $P_x=P_y=4$ for all filters.

	Impulse Response				Frequency Response			
	$\theta = 0$	$\theta = \frac{\pi}{4}$	$\theta = \frac{\pi}{2}$	$\theta = \frac{3\pi}{4}$	$\theta = 0$	$\theta = \frac{\pi}{4}$	$\theta = \frac{\pi}{2}$	$\theta = \frac{3\pi}{4}$
$n=0$								
$n=1$								
$n=2$								
$n=3$								
$n=4$								

Table 3.7: 2D-convolution kernel $K_{2D}(n, \theta, P_x, P_y)$ for arbitrary order of differentiation n , steering angle θ , and cutoff frequency controlled by flatness degree P_x and P_y . Selected parameters here are $l=13$, $P_x=P_y=12$ for all filters.

Impulse Response					Frequency Response			
n=0								
	$\theta = 0$	$\theta = \frac{\pi}{4}$	$\theta = \frac{\pi}{2}$	$\theta = \frac{3\pi}{4}$	$\theta = 0$	$\theta = \frac{\pi}{4}$	$\theta = \frac{\pi}{2}$	$\theta = \frac{3\pi}{4}$
n=1								
n=2								
n=3								
n=4								

2D directional kernels generated by this MATLAB function is favorably comparable with their Gaussian counterparts introduced in [10, 11] for image feature extraction tasks. In particular, MaxPol kernels are much accurate on frequency transformation of the pass-band, also contain sharp roll-off on the cutoff band. We refer the reader to image sharpening application in [2] for efficiency of these kernels.

For instance, Figure 3.1 shows the contour level of the frequency magnitudes drop from pass-band to stop-band two dimension.

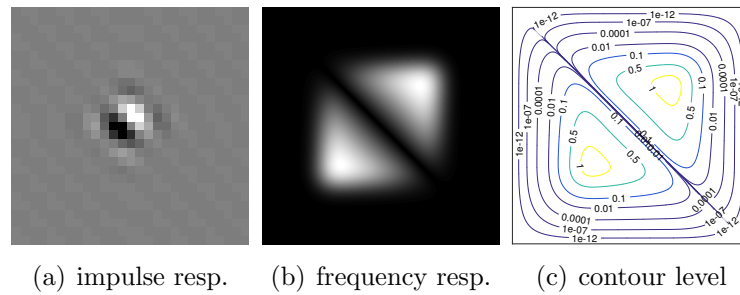


Figure 3.1: Contour level plot of 2D direction derivative kernel. Setup parameters are $l=13$, $P_x=P_y=12$, and $\theta=\pi/4$. The equivalent cutoff level is $\omega_{c_x} = \omega_{c_y} \approx 0.5\pi$

3.5 How MaxPol generalizes the literature methods?

In this section we provide a summary on how **MaxPol** generalizes many methods in the literature. For thorough study on the **MaxPol** we refer the reader to [1, 2]. In particular we refer the reader to the Table 1, page 7 in [1] which provides an overview of existing numerical difference methods in the literature, including **MaxPol**, and how they compare to each other in terms of feature utilities. The following Table 3.8 demonstrates the cross parameter that can be tweaked for **MaxPol** and becomes equivalent to other methods.

Table 3.8: List of numerical methods as variants of **MaxPol** method

	Selected parameters in MaxPol
Fornberg [8, 9]	nod='centralized', $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l$
	nod='staggered' , $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l-1$
Savitzky-Golay (Fullband) [3–7]	nod='centralized', $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l$
	nod='staggered' , $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l-1$
Khan [12–16]	nod='centralized', $s=0$, $n \geq 0$, $l \geq 0$, $P=2l$
	nod='staggered' , $s=0$, $n \geq 0$, $l \geq 0$, $P=2l-1$
Li [17]	nod='centralized', $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l$, D_n
O'Leary [18, 19]	nod='centralized', $-1 \leq s \leq 1$, $n=1$, $l \geq 0$, $P=2l$, D_n
Hassan [20]	nod='centralized', $-1 \leq s \leq 1$, $n \geq 0$, $l \geq 0$, $P=2l$, D_n
Carlsson [21]	nod='centralized', $s=0$, $n=1$, $l \geq 0$, $P=2l$
Kumar [22–24]	nod='centralized', $s=0$, $n=1$, $l \geq 0$, $P=2l$
	nod='staggered' , $s=0$, $n=1$, $l \geq 0$, $P=2l-1$
Selesnick [25–27]	nod='centralized', $s=0$, $0 \leq n \leq 1$, $l \geq 0$, $n \leq P \leq 2l$
	nod='staggered' , $s=0$, $0 \leq n \leq 1$, $l \geq 0$, $n \leq P \leq 2l-1$

Chapter 4

Demo Examples: A Quick Start

In this chapter we provide intuitive examples of signal and image processing from both forward and inverse problems. In particular, our aim here is to show how the **MaxPol** package can be utilized in different applications for derivative approximation. The future release of **MaxPol** package will contain more advanced and diverse imaging applications.

4.1 FIR frequency responses

From the root directory please navigate to `\demo examples\FIR frequency responses\`. There are six demo examples provided in this directory:

```
demo_MaxPol_transfer_function_centralized_scheme.m
demo_MaxPol_transfer_function_staggered_scheme.m
demo_SavtizkyGolay_transfer_function_centralized_scheme.m
demo_SavtizkyGolay_transfer_function_staggered_scheme.m
demo_Fornberg_transfer_function_centralized_scheme.m
demo_Fornberg_transfer_function_staggered_scheme.m
demo_Gaussian_transfer_function.m
demo_lowpass_methods_comparison.m
demo_lowpass_methods_cutoff_analysis.m
```

The purpose of these MATLAB functions is to demonstrate the filter responses (aka transfer function) of associated derivative kernels of lowpass/fullband utilized by different filter methods.

4.2 Spectral analysis of derivative matrix

From the root directory please navigate to `\demo examples\spectral analysis of derivative matrix\`. There are four demo examples provided in this directory:

`demo_distribution_of_eigenvalues_lowpass_vs_fullband.m`

`demo_distribution_of_eigenvalues_staggered_vs centralized.m`

`demo_distribution_of_eigenvalues_study_on_polynomial_order_1.m`

`demo_distribution_of_eigenvalues_study_on_size_N.m`

The above MATLAB m-files demonstrate the spectral response of MaxPol derivative matrices in terms of eigenvalue distribution. The evolution of eigenvalues are studied over different parameter setup such as staggered/centralized, matrix size, polynomial degree, and fullband/lowpass.

4.3 Directional (steerable) 2D derivatives

From the root directory please navigate to `\demo examples\directional (steerable) 2D derivatives\`. There is only one demo example provided in this directory:

`demo_2D_directional_derivatives_MaxPol.m`

`demo_2D_directional_derivatives_comparison.m`

demonstrates two dimensional (2D) derivative kernel images and their filter response using different filter methods. You can set different parameters of derivative orders, steering angle, and cutoff levels to visualize variety of kernels. Variants of MaxPol 2D kernels are shown in Table 3.6 and Table 3.7.

You can apply these kernel in an image convolution problem to approximate directional derivative of an image with the corresponding parameters using the MATLAB build-in function `imfilter`.

4.4 Forward signal and imaging problems

4.4.1 One dimensional (1D) signal

From the root directory please navigate to `\demo examples\forward signal and imaging problems\one dimensional decomposition\`. There are two demo examples provided in this directory:

`demo_sinusoid_global_accuracy_analysis.m`

`demo_sinusoid_staggered_vs centralized_analysis.m`

The above MATLAB m-files provides a thorough analysis on estimating first order derivative of sinusoid signals. The efficiency of different polynomial degrees for differentiation is studied through variety of harmonic frequencies.

4.4.2 Two dimensional (2D) image/surface

From the root directory please navigate to `\demo examples\forward signal and imaging problems\two dimensional decomposition\`. There are two demo examples provided in this directory:

`demo_2D_zone_plate_MaxPol_accuracy_analysis.m`

`demo_2D_zone_plate_accuracy_analysis_comparisons.m`

These two MATLAB m-files analyze the two dimensional differentiation of zone plate sinusoid grating image. Thorough comparison is made with other counterpart methodologies to study the accuracy of numerical differentiation on boundary and interior domains of the zone plate images.

4.4.3 Canny edge detection

From the root directory please navigate to `\demo examples\forward signal and imaging problems\Canny edge detection\`. The main demo example is:

`demo_image_edge_detection.m`

This MATLAB m-file performs Canny edge detection on an image utilized by four different lowpass filters of MaxPol, Savitzky-Golay, Farid-Simoncelli, and Gaussian. The Gaussian is the conventional method proposed in [28]. You can set your own parameters of cutoff level and polynomial degree for all four kernels and see how they affect the outcome results.

4.4.4 Image Sharpening

From the root directory please navigate to `\demo examples\forward signal and imaging problems\image sharpening (unsharp masking)\`. The main demo example is:

`demo_unsharp_masking.m`

This MATLAB m-file performs unsharp masking method introduced in [2] on flower image utilized by four different lowpass filters of MaxPol, Savitzky-Golay, Farid-Simoncelli, and Gaussian. You can set your own cutoff frequency parameters to visualize different sharpening levels. The initial parameters are optimized for flower image based on the analysis described in [2].

4.4.5 Three dimensional (3D) Volume

From the root directory please navigate to `\demo examples\forward signal and imaging problems\three dimensional decomposition\`. The main demo example is:

`demo_3D_MRI_decomposition.m`

This MATLAB demo example in particular loads a 3D MRI volumetric data from human brain, available from¹, and differentiates the discrete data in its third dimension. We cut from y-axis and demonstrate the result of decomposition in xz plane. The data is originally contaminated with noise and we use both MaxPol and Savitzky-Golay methods for lowpass differentiation.

You can initialize your own parameters to visualize different analysis of the data.

4.5 Inverse imaging problems

From the root directory please navigate to `\demo examples\inverse imaging problems\`. There are two demo examples provided in this directory:

`demo_zone_plate_noisy_gradient_surface_recovery.m`

`demo_image_stitching.m`

Both demo MATLAB m-files are about gradient surface reconstruction problems. The zone plate example reconstructs surface image from synthetic gradient samples using different derivative matrix cases introduced in [1]. The second example is about real application of stitching preview images in digital pathology images. For more information on the experimental details please refer to [1].

¹<http://brainweb.bic.mni.mcgill.ca/brainweb/>

Chapter 5

Citing MaxPol

It is all the time a pleasure to hear from you to collect your thoughts and valuable feedback. If you face with a bug in the codes please report at `mahdi.hosseini@mail.utoronto.ca`. If you are using the maxpol package in your teaching and research area, we would like to definitely hear from you to learn the technical diversity of our users. If you are using MaxPol in your research publications, we appreciate if you use a similar sentence in your paper as follows:

“For numerical solution of the problem (perhaps a PDE equation) we used **MaxPol**¹, a package to solve numerical differentiation [1, 2].”

The corresponding BiBTeX citations are:

```
@article{HosseiniPlataniotisSIAM2017,  
title={Finite Differences in Forward and Inverse Imaging Problems--MaxPol Design},  
author={Mahdi S. Hosseini and Konstantinos N. Plataniotis},  
journal={submitted to SIAM Journal on Imaging Sciences (SIIMS)},  
year={2017},  
publisher={SIAM}}
```

```
@article{HosseiniPlataniotisTIP2017,  
title={Derivative Kernels: Numerics and Applications},  
author={Mahdi S. Hosseini and Konstantinos N. Plataniotis},  
journal={Accepted in IEEE Transactions on Image Processing (TIP)},  
year={2017},  
publisher={IEEE}}
```

¹MATLAB codes are available online at ...

Bibliography

- [1] M. S. Hosseini and K. N. Plataniotis, “Finite differences in forward and inverse imaging problems–maxpol design,” *submitted to SIAM Journal on Imaging Sciences (SIIMS)*, 2017.
- [2] M. S. Hosseini and K. N. Plataniotis, “Derivative kernels: Numerics and applications,” *Accepted in IEEE Transactions on Image Processing (TIP)*, 2017.
- [3] A. Savitzky and M. J. E. Golay, “Smoothing and differentiation of data by simplified least squares procedures,” *Analytical Chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.
- [4] P. A. Gorry, “General least-squares smoothing and differentiation by the convolution (savitzky-golay) method,” *Analytical Chemistry*, vol. 62, no. 6, pp. 570–573, 1990.
- [5] P. Meer and I. Weiss, “Smoothed differentiation filters for images,” in *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, vol. 2, pp. 121–126, IEEE, 1990.
- [6] J. Luo, K. Ying, P. He, and J. Bai, “Properties of savitzky–golay digital differentiators,” *Digital Signal Processing*, vol. 15, no. 2, pp. 122–136, 2005.
- [7] R. W. Schafer, “What is a savitzky-golay filter?[lecture notes],” *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 111–117, 2011.
- [8] B. Fornberg, “Generation of finite difference formulas on arbitrarily spaced grids,” *Mathematics of computation*, vol. 51, no. 184, pp. 699–706, 1988.
- [9] B. Fornberg, “Classroom note:calculation of weights in finite difference formulas,” *SIAM Review*, vol. 40, no. 3, pp. 685–691, 1998.
- [10] W. T. Freeman and E. H. Adelson, “The design and use of steerable filters,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 9, pp. 891–906, 1991.

- [11] M. Jacob and M. Unser, “Design of steerable filters for feature detection using canny-like criteria,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 8, pp. 1007–1019, 2004.
- [12] I. R. Khan and R. Ohba, “Closed-form expressions for the finite difference approximations of first and higher derivatives based on taylor series,” *Journal of Computational and Applied Mathematics*, vol. 107, no. 2, pp. 179 – 193, 1999.
- [13] I. Khan and R. Ohba, “New design of full band differentiators based on taylor series,” *Vision, Image and Signal Processing, IEE Proceedings -*, vol. 146, pp. 185–189, Aug 1999.
- [14] I. R. Khan and R. Ohba, “New finite difference formulas for numerical differentiation,” *Journal of Computational and Applied Mathematics*, vol. 126, no. 12, pp. 269 – 276, 2000.
- [15] I. R. Khan, R. Ohba, and N. Hozumi, “Mathematical proof of closed form expressions for finite difference approximations based on taylor series,” *Journal of Computational and Applied Mathematics*, vol. 150, no. 2, pp. 303 – 309, 2003.
- [16] I. R. Khan and R. Ohba, “Taylor series based finite difference approximations of higher-degree derivatives,” *Journal of Computational and Applied Mathematics*, vol. 154, no. 1, pp. 115 – 124, 2003.
- [17] J. Li, “General explicit difference formulas for numerical differentiation,” *Journal of Computational and Applied Mathematics*, vol. 183, no. 1, pp. 29 – 52, 2005.
- [18] P. O’Leary and M. Harker, “An algebraic framework for discrete basis functions in computer vision,” in *Computer Vision, Graphics & Image Processing, 2008. ICVGIP’08. Sixth Indian Conference on*, pp. 150–157, IEEE, 2008.
- [19] P. O’Leary and M. Harker, “A framework for the evaluation of inclinometer data in the measurement of structures,” *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 5, pp. 1237–1251, 2012.
- [20] H. Hassan, A. Mohamad, and G. Atteia, “An algorithm for the finite difference approximation of derivatives with arbitrary degree and order of accuracy,” *Journal of Computational and Applied Mathematics*, vol. 236, no. 10, pp. 2622 – 2631, 2012.
- [21] B. Carlsson, “Maximum flat digital differentiator,” *Electronics Letters*, vol. 27, pp. 675–677, April 1991.

- [22] B. Kumar and S. Dutta Roy, "Design of digital differentiators for low frequencies," *Proceedings of the IEEE*, vol. 76, pp. 287–289, Mar 1988.
- [23] B. Kumar and S. Dutta Roy, "Maximally linear fir digital differentiators for high frequencies," *Circuits and Systems, IEEE Transactions on*, vol. 36, pp. 890–893, Jun 1989.
- [24] B. Kumar, S. Dutta Roy, and H. Shah, "On the design of fir digital differentiators which are maximally linear at the frequency π/p , p isin; positive integers," *Signal Processing, IEEE Transactions on*, vol. 40, pp. 2334–2338, Sep 1992.
- [25] I. Selesnick and C. Burrus, "Maximally flat low-pass fir filters with reduced delay," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 45, pp. 53–68, Jan 1998.
- [26] I. W. Selesnick and C. S. Burrus, "Generalized digital butterworth filter design," *IEEE Transactions on Signal Processing*, vol. 46, pp. 1688–1694, Jun 1998.
- [27] I. Selesnick, "Maximally flat low-pass digital differentiator," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 49, pp. 219–223, Mar 2002.
- [28] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, pp. 679–698, Nov 1986.