


STEM
Stuttgart Center for Electron Microscopy



Lectures in the theory of TEM

III. Advanced Data Analysis in TEM:

How to do what cannot be done using Image processing software

Christoph T. Koch
*Stuttgart Center for Electron Microscopy
Max Planck Institute for Intelligent Systems
Heisenbergstr. 3, 70569 Stuttgart, Germany
Email: koch@is.mpg.de*

MPI for Metals Research

STEM
Stuttgart Center for Electron Microscopy

Make a Choice !

You want to win the race? ...



94 km/h = 51 knots




... Spend some time considering what equipment you want to use.




2

MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy

Some Options for TEM Data Analysis




- DigitalMicrograph
- Matlab
- IDL
- Mathematica
- MathCad
- Semper
- Python
- ImageJ
- ...

Questions to ask (yourself):


- What are my needs ?
- Availability ?
- Price ?
- Flexibility ?
- What am I used to ?
- Who else is using it ?
- How easily can I get help ?
- ...


3
MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy


A few Possibilities






DigitalMicrograph

- Installed on most TEMs
- Can acquire CCD images and control the TEM
- Scripting language similar to C++ but specialized for image processing



Matlab


- Available at most universities, or cheap student/teaching licenses
- Contains a lot of general math functions not available within DM
- With some effort you can make it acquire images through DigitalMicrograph (see, e.g. TOM Toolbox for tomography [http://www.biochem.mpg.de/baumeister/tom_e/index.html])



Python


- Platform independent (Unix, Linux, Windows, Max OSX, Java VM, ...)
- Large user base
- Slim standalone applications


4
MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy


Resources





DigitalMicrograph


- Introduction to DM scripting: http://www.felmi-zfe.tugraz.at/dm_scripts
- A scripting handbook: <http://dm-scripting.tavernmaker.de/>
- The scripting database: http://www.felmi-zfe.tugraz.at/dm_scripts/
- Free Plug-Ins: <http://www.hremresearch.com/>
(for commercial ones – see next slide)
- Dave Mitchell's scripting website: <http://www.dmscripting.com/>
- Scripting lectures: http://www.christophtkoch.com/Vorlesung/index_SS07.html



Matlab


- Large database of Matlab programs: www.matlabcentral.org
- Matlab routine for reading DM3 files:
<http://www.mathworks.com/matlabcentral/fileexchange/27021-imagick-mrc-and-dm3-file-io>


5
MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy

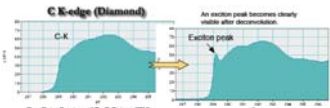
Commercial 3rd Party add-ons to DM





Electron Energy Loss Spectrum Deconvolution

Software Monochromator for EELS




DeConVEELS rectifies an Electron Energy Loss Spectrum (EELS) by deconvoluting with a low-loss or zero-loss spectrum, and thus is a software monochromator.

Advanced Deconvolution Algorithms DeConVEELS uses a Maximum Entropy Method or the Richardson-Lucy Algorithm.

Easy-to-use User Interface DeConVEELS is a plug-in for use in DigitalMicrograph (Gatan).

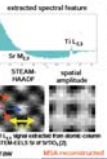
SI Module (Optional) EELS SI (Spectrum Image) data can be handled easily using this module.

www.hremresearch.com / support@hremresearch.com / +41(40)325 3919 **HREM Research Inc.**



Multivariate Statistical Analysis

Advanced Manipulation Tools for Spectrum Images (SIs)




MSA for Gatan DigitalMicrograph finds statistically significant features from 2D and 3D spectrum images gathered by spectrometric techniques such as XEDS, EELS, E-TEM and cathodoluminescence. This plug-in was originally developed by Masashi Watanabe [1].

Key Features

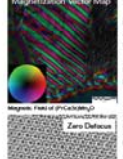
- Automatically extracts statistically significant spectral features and corresponding spatial amplitudes as principal components (see left top figure)
- Performs efficient noise reduction on SIs
- Enhances weak signals hidden under noise in SIs (see left bottom figure)
- Includes utilities to import line profiles and SIs obtained by other acquisition systems than Gatan DigitalMicrograph.

References:
[1] M. Watanabe, T.S. Wilson and M.C. Beale, Atom. and Solid State Electron Microscopy of Materials, Proc. 10th. Conf. on Solid State Phase Transformations in Progress, 1998, pp. 1-10. See also: M. Watanabe, T.S. Wilson, and M.C. Beale, Proceedings of Electron Spectroscopy Symposium and Energy Bands in an Atomically-Correlated JSM 2001, ICPAC, 10th International Conference on Spectroscopy, 2001, pp. 1-10. See also: M. Watanabe, T.S. Wilson, and M.C. Beale, Spectroscopic Investigation of Composites, in: Trends in the Application of FEI/STEM Analytical Instrumentation, J. Phys.: Conf. Ser. 147, 2007.

www.hremresearch.com / support@hremresearch.com / +41(40)325 3919 **HREM Research Inc.**



Quantitative Phase Technology



QPt Derives a phase image at the middle plane (in-focus) only from three ordinary bright-field images, and thus eliminates artifacts generally apparent in a defocused image.


QPt is based on Iata's Quantitative Phase Imaging (QPI) technology, and provides a digital solution to phase contrast electron microscopy.

Key Features:

- QPI (Basic) Derives a phase image at the middle plane (in-focus).
- QPI (Lift) Generates a microscopic magnetic field from the phase image (the top example).
- QPI (Full) Emulates optical phase modulations such as Differential Interference Contrast, Zernike Phase Contrast, Hoffman Modulation Contrast and Dark-field images.
- HREM (modal) Reconstructs a spherical aberration-corrected atomic-resolution, real-space (the bottom example).

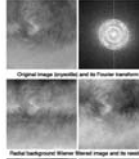
References: [1] K. Iata and S. Akita, J. of Electron Microsc. 14 (2005) 191-193. [2] K. Iata and S. Akita, Microscopy Today 11 (2006) 61-64. Credits: P/QPI/DM, courtesy Masaya Uchida, NIMS, Tsukuba. QPI/Basic, courtesy of Christian Klotter, NCR, Bremen.

www.hremresearch.com / support@hremresearch.com / +41(40)325 3919 **HREM Research Inc.**



HREM-Filters Pro/Lite

Optimal Noise Filters for High-Resolution Electron Microscopy



HREM-Filters are sophisticated Wiener and Difference filters that work even for non-ideal crystals, such as a nano-crystal or cylindrical crystal.

HREM-Filters Lite can be downloaded free of charge from our web site: www.hremresearch.com

Key Features

- Users smoothed two-dimensional background [1].
- Users locally estimated backgrounds [2].
- Trend subtraction.
- Optimal periodic Wiener filter using accurate base vectors.

(middle row) Wiener filter based on a radial background [1] does not work for a cryogenic image, and substantial features are left behind.


(bottom row) Wiener filter based on local two-dimensional backgrounds extracts all the structure information, and a difference between the filtered and original images is reconstructed.

References:
[1] T. Wilson, J. Microscopy 162 (2001) 83-81.
[2] J.C. Liou, et al., Computational Science and Data Analysis 10 (2006) 21-24.
[3] P.C. Eble and K. Mizutani, MRS, Boston, (2006) 984.


Credits: Crystal image courtesy of Tomoko Nagayama.

www.hremresearch.com / support@hremresearch.com / +41(40)325 3919 **HREM Research Inc.**

6




What these programs can't do ...




- Does not record macros (i.e. it does not automatically generate a script from command that you enter)
 - But: all the menu items can be accessed from scripts

- DM-scripts become slow, when excessive use of loops is being made (interpreted language: code is being compiled “on the fly”)
 - But: compiled C/C++-code may be included as PLUG-IN




the same is true for Matlab!

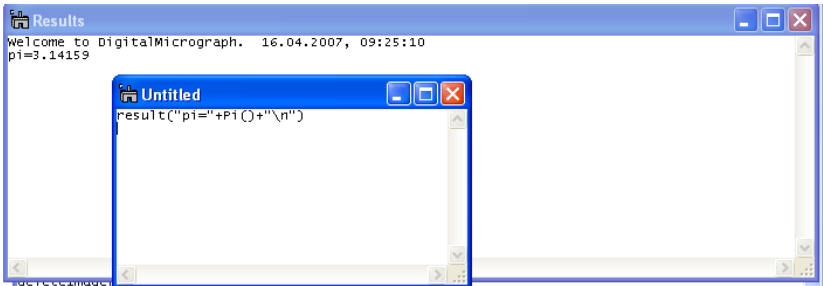
MPI for Intelligent Systems




Displaying Text



```
result("pi="+Pi()+"\n")
```





```
Command Window
To get started, select MATLAB Help or Demos from the Help menu.

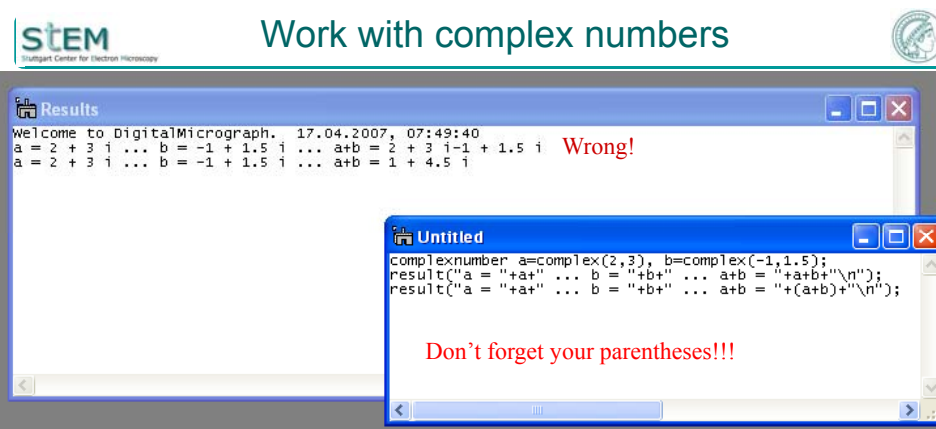
>> pi
ans =
    3.1416

>> fprintf('Pi=%f\n',pi)
Pi=3.141593
>> |
```

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy

Work with complex numbers



Results
 Welcome to DigitalMicrograph. 17.04.2007, 07:49:40
 a = 2 + 3 i ... b = -1 + 1.5 i ... a+b = 2 + 3 i -1 + 1.5 i **Wrong!**
 a = 2 + 3 i ... b = -1 + 1.5 i ... a+b = 1 + 4.5 i

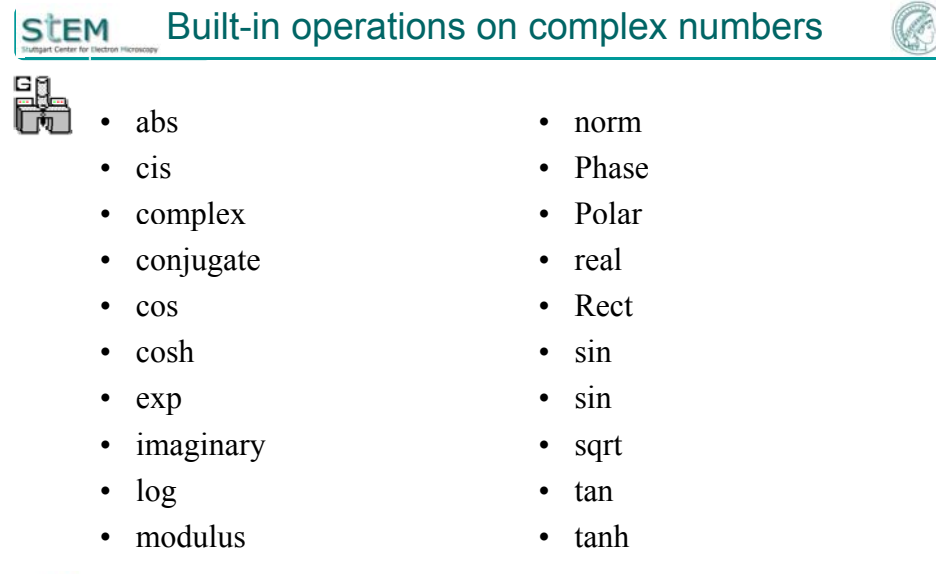
Untitled
 complexnumber a=complex(2,3), b=complex(-1,1.5);
 result("a = "+a+" ... b = "+b+" ... a+b = "+(a+b)+"\n");
 result("a = "+a+" ... b = "+b+" ... a+b = "+(a+b)+"\n");
Don't forget your parentheses!!!

Command Window
 >> a=2+3*i;
 >> b=-1+1.5*i;
 >> a+b
 ans =
 1.0000 + 4.5000i
 >>

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy


Built-in operations on complex numbers



- abs
- cis
- complex
- conjugate
- cos
- cosh
- exp
- imaginary
- log
- modulus
- norm
- Phase
- Polar
- real
- Rect
- sin
- sin
- sqrt
- tan
- tanh


There are similar commands in Matlab (and many more)!


MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy


A selection of real-number functions






- sin, asin, sinh
- cos, acos, cosh
- tan, tanh, atan, atan2, atanh
- exp, exp2, exp10
- log, log2, log10
- $\exp1(x) = \exp(x) - 1$
- $\log1(x) = \log(x + 1)$
- AiryAi, AiryBi
- BesselI, (also J, K, Y)
- SphericalBesselJ (also Y)

- Beta
- erf, erfc
- Factorial
- Gamma, GammaP, GammaQ
- LegendrePolynomial
- PoissonRandom (also Binomial, Gaussian, Gamma, Uniform)
- BinomialCoefficient
- ...




There are similar commands in Matlab (and many more)!

MPI for Intelligent Systems

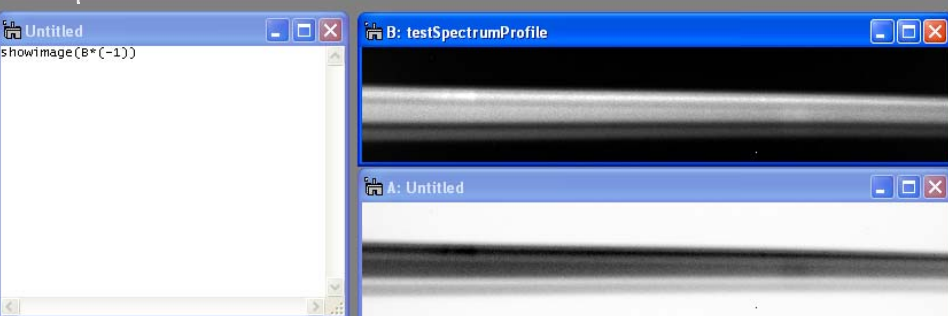


Stuttgart Center for Electron Microscopy

On the fly data manipulation




Makes sense if script is only used once !



- Images may be addressed in a script by the letter assigned to the window they are shown in.
- If a new image is generated (e.g. by performing some operation on the image), this image will receive a new letter (image variable)

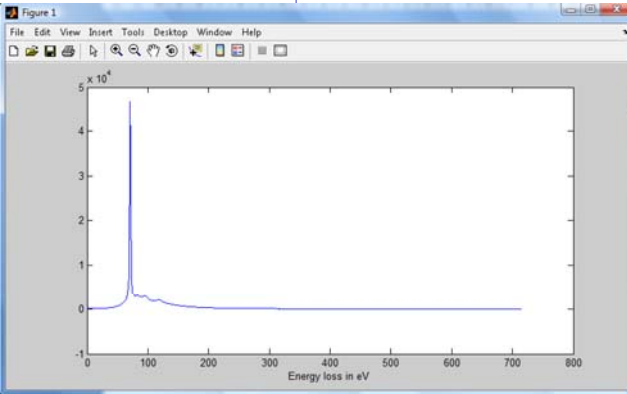
MPI for Intelligent Systems

STEM Matlab: Reading and Displaying DM3 Data 


Stuttgart Center for Electron Microscopy

```

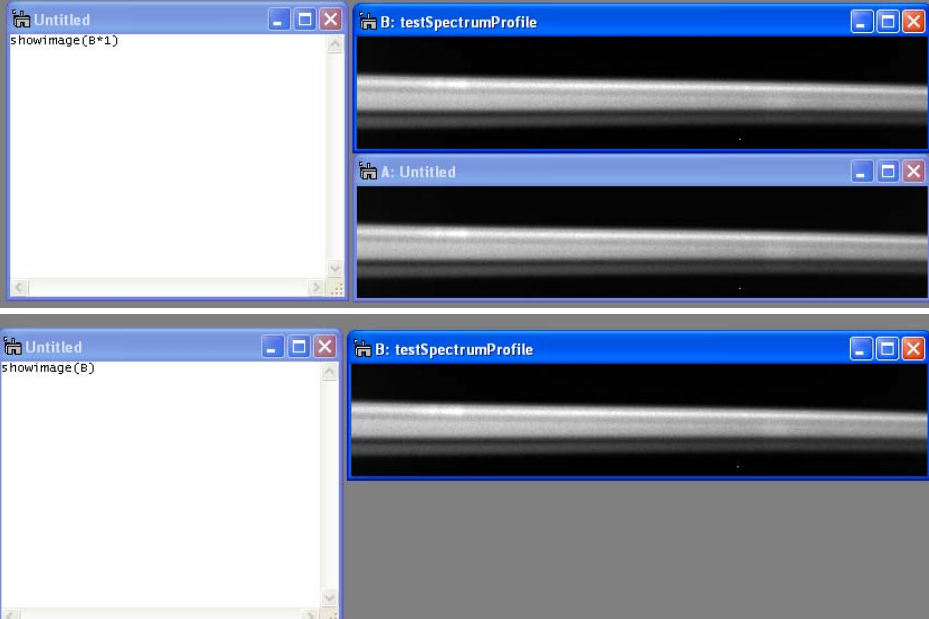
Command Window
>> [mSize,dx,dy] = ReadDM3_size('Spectra\01 ZL.dm3')
mSize =
    1022     1     1
dx =
    0.7000
dy =
     1
>> spec=ReadDM3_slice('Spectra\01 ZL.dm3',mSize,1)';
>> plot(dx*1:length(spec),spec)
>> xlabel('Energy loss in eV')
>>
    
```



13 MPI for Intelligent Systems

STEM Automatic generation of new images 

Stuttgart Center for Electron Microscopy




Untitled
showimage(B*1)

B: testSpectrumProfile


A: Untitled

Untitled
showimage(B)

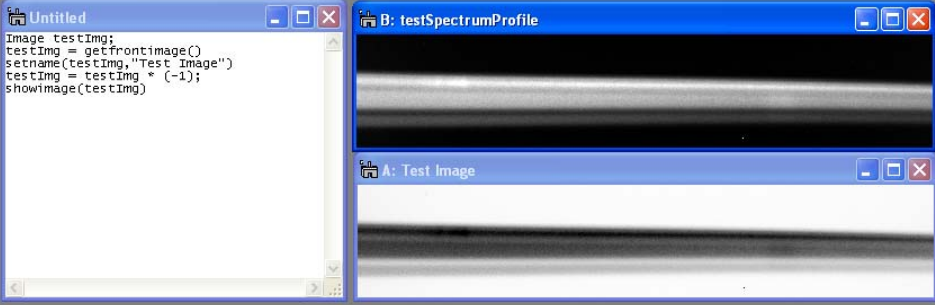
B: testSpectrumProfile



Make script independent of image name




This makes scripts more general !




Img = GetFrontImage(): obtain the image which is currently active.

Note: if no name is assigned to an image variable, the name of the image remains "Untitled". It helps to assign names to image windows using SetName

MPI for Intelligent Systems



DM Built-in commands



Script Reference 2.5

See list of script functions in [alphabetical order](#).

Images

- [Image Processing](#)
- [Image Data Types](#)
- [Real Images](#)
- [Complex Images](#)
- [RGB Images](#)
- [Image Management](#)
- [Image Display](#)
- [Image Scrap](#)

Numbers and Strings

- [Real Number Functions](#)
- [Complex Number Functions](#)
- [RGB Number Functions](#)
- [Number Conversion](#)
- [Strings](#)

Annotations, Selections, Tags, I/O

- [Annotations](#)
- [Selections](#)
- [Tags \(aka Notes\)](#)
- [Dialogs](#)
- [Input/Output](#)


Other

- [Miscellaneous](#)

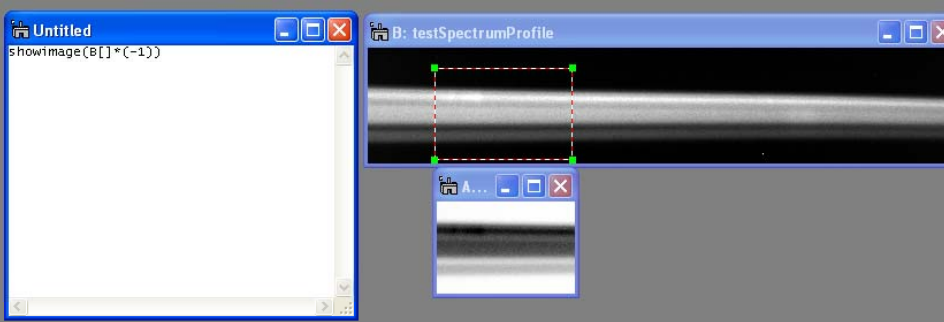
This list of commands used to be available online but has been discontinued by Gatan.

A copy of this is available at http://www.felmi-zfe.tugraz.at/dm_scripts

MPI for Intelligent Systems


STEM Addressing regions of interest (ROI) 

Stuttgart Center for Electron Microscopy




On the fly: simply use square brackets behind the image letter


MPI for Intelligent Systems

FFTW Fastest Fourier Transform in the West 


- FFTW is a software library that also allows every 3rd, 4th, fifth, etc. element to be used. It can therefore handle any size of array.
- Implemented for DM in plugin Transforms.dll
(<http://hrem.mpi-stuttgart.mpg.de/koch/DM-Plugin/index.html>)
- Implemented functions (do not allocate memory for new image):
 - `T_fft_c2c(compl_ImgIn,compl_ImgOut)`
 - `T_ifft_c2c(compl_ImgIn,compl_ImgOut)`
 - `compl_Img = T_shiftImageCenterComplex(compl_Img)`
(shifts $k=(0,0)$ of FFT to center of image for complex images)
 - DM function does not work)
 - Additional functions for computing different correlations

 Matlab comes with FFTW, but DM can only do radix 2 FFTs

MPI for Intelligent Systems



Usage of FFTW within DM



```

image img
image DoFFTW(image img)
{
  complexImage inImg, outImg, outImgs
  number top, left, bottom, right, width, height
  string name

  GetSelection(img, top, left, bottom, right)  obtain ROI
  GetName(img, name)

  width = right - left
  height = bottom - top

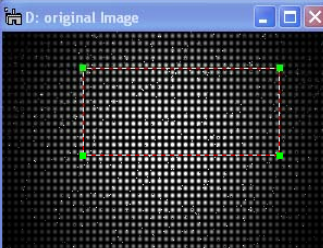
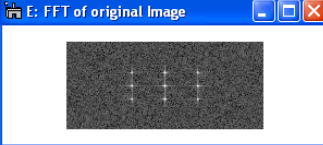
  inImg := ComplexImage("real space Image", 8, width, height)
  outImg := ComplexImage("reciprocal space Image", 8, width, height)

  inImg = img[]
  T_fft_C2C(inImg, outImg)  real -> complex image
  deleteImage(inImg)       compute FFT
  outImgs := T_shiftImageCenterComplex(outImg)
  deleteImage(outImg)

  setOrigin(outImgs, width/2, height/2)
  setName(outImgs, "FFT of " + name)
}
return outImgs


if(!GetFrontImage(img)) {
  okDialog("There is no Image.")
  exit(0)
}
showImage(DoFFTW(img))

```






Such scripts (written by Bernd Kraus) are available for FFT and IFFT, for both real- and complex input.

MPI for Intelligent Systems



Difference between '==', '=', and ':='






A==B comparison (1, if left=right, 0 otherwise)

A = B copies variable content

A:= B assigns variable A to the image given by B
(should always be used when creating images,
although it also works with '=')

(see more examples in B. Schaffer's tutorial, slides 33 & 34)



In Matlab there is no ':='

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy

Spectra: 1-dimensional images

```

// subsection: top,left,bottom,right
Image B_1d1 = B[1,1,200,2]
setname(B_1d1,'vertical profile')
showimage(B_1d1)
Image B_1d2 = B[1,1,2,1024]
setname(B_1d2,'horizontal profile')
showimage(B_1d2)
    
```

The interface shows a code editor window titled 'Untitled' with the above code. To the right, a window titled 'B: testSpectrumProfile' displays a dark image. Below it, two smaller windows are visible: 'A...' showing a vertical line and 'C: Untitled' showing a horizontal spectrum plot with intensity on the y-axis (0 to 100) and position on the x-axis (0 to 1000).

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy

Vertical 1D-image -> spectrum

```

// subsection: top,left,bottom,right
Image B1 = B[1,1,200,2]
setname(B1,'vertical section')
Image B1r = B1
RotateRight(B1r)
setname(B1r,'rotated vertical section')
showimage(B1r)

Image B2 = B[1,1,2,1024]
setname(B2,'horizontal profile')
showimage(B2)
    
```


The interface shows a code editor window titled 'Untitled' with the above code. To the right, a window titled 'B: testSpectrumProfile' displays a dark image. Below it, two smaller windows are visible: 'D: rotated vertical section' showing a spectrum plot with intensity on the y-axis (0 to 4000) and position on the x-axis (0 to 200), and 'C: horizontal profile' showing a spectrum plot with intensity on the y-axis (0 to 150) and position on the x-axis (0 to 1000).

Allowed values for selection of sub-region:


top: 0	...(bottom-1)	bottom: (top+1)	... height
left: 0	...(right-1)	right: (left+1)	... width

In Matlab, indices start at 1 (Fortran), in DM they start at 0 (C/C++)

MPI for Intelligent Systems



icol, irow, iradius, ...



- There are several *intrinsic* variables which can be used in calculations of images. Their value depends on the position within the image.
(e.g.: `icol` becomes 5 for all points in an image, which have `x=5` as coordinate. It becomes 6 for `x=6` and so on..)
- The following script creates some examples:
(The function `Pi ()` returns the value of `Pi`.)

```

image TestImage
TestImage := RealImage("Test", 4, 100, 100)
ShowImage(TestImage)

TestImage = sin(2*Pi()/iwidth*icol)
TestImage = cos(2*Pi()/iheight*irow)
TestImage = exp(-iradius**2/(iheight/10)**2)
TestImage = tan(itheta)

```

Name	Description
<code>icol</code>	column of the image
<code>iheight</code>	height of the image
<code>ipoints</code>	number of points in the image
<code>iradius</code>	distance from the center of the image
<code>irow</code>	row of the image
<code>itheta</code>	angle with respect to the center of the image
<code>iwidth</code>	width of the image
<code>iplane</code>	plane of the image (3D images)

- Often, the intrinsic variables are used in the `tert ()` command:
(The function `mod(a, b)` returns the modulo, e.g. `mod(14, 3)=2` as $14 = 4*3 + 2$)

```

TestImage = tert( mod(icol,10)==0 || mod(irow,10)==0, 1, 0)
TestImage = tert( iradius<iwidth/4 , icol , irow)


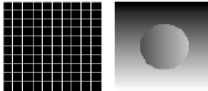
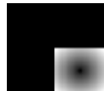
```


- Note that the variables check the actual image expression, not the image itself. If an area of an image is used, the top-left pixel of this area is (0/0):

```

TestImage = 0
TestImage[50,50,100,100] = iradius // the center is now at 75/75!


```




Not such built-in variables in Matlab

Slide: Bernhard Schaffer, TU Graz




Applications of Convolution in TEM




- Smoothing of data
- Differentiating of data (e.g. Laplacian, ...)
- Simulate the effect of microscope instabilities (e.g. sample vibrations for images, energy fluctuations in spectra)
- Simulate the effect of detector point spread functions (PSF)
- Simulate the effect of microscope aberrations in HAADF-STEM (based on an oversimplifying approximation)

MPI for Intelligent Systems




Stuttgart Center for Electron Microscopy

Applications of Deconvolution in TEM




- Inversion of gradient and Laplacian operations
- Removal of microscope instabilities (e.g. sample vibrations in images, energy instabilities in spectra)
- Removal of microscope aberrations in HAADF-STEM (assuming that the image is the convolution of probe and object function)
- Removal of plural scattering in EELS
- Removal of source energy spread in EELS

MPI for Intelligent Systems



Stuttgart Center for Electron Microscopy

Definition of Convolution



1D
$$f \otimes g = \int_{-\infty}^{\infty} f(r') \cdot g(r - r') dr'$$

2D
$$f \otimes g = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') \cdot g(x - x', y - y') dx' dy'$$

In general, the value of $F(\mathbf{r})=f(\mathbf{r}) \otimes g(\mathbf{r})$ depends on the values of $f(\mathbf{r})$ and $g(\mathbf{r})$ for all \mathbf{r} , i.e. across the whole image

Convolution Theorem:
$$f \otimes g = FT^{-1}[FT(f)FT(g)]$$

MPI for Intelligent Systems

STEM Computing the gradient by convolution

Real space:

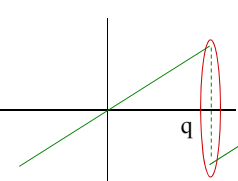
$$\frac{df}{dx} = \frac{f_{i+1} - f_i}{\Delta x} = f \otimes D_x^1 \quad D_x^1 = \begin{pmatrix} 1 & -1 \end{pmatrix}$$

or

$$\frac{df}{dx} = \frac{f_{i+1} - f_{i-1}}{2 \cdot \Delta x} = f \otimes D_x^1 \quad D_x^1 = \begin{pmatrix} 1 & 0 & -1 \end{pmatrix}$$

Reciprocal space:

$$\frac{df}{dx} = \frac{d}{dx} \sum F_q \cdot e^{2\pi i q x}$$

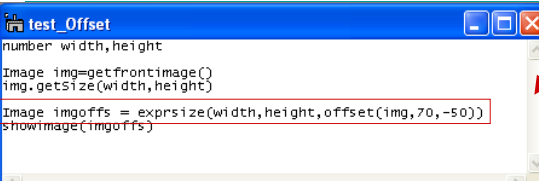
$$= \sum F_q \cdot 2\pi i q \cdot e^{2\pi i q x}$$


Discontinuous edges! $\Rightarrow \frac{df}{dx} = FT^{-1} \{ 2\pi i \cdot FT[f] \cdot q \}$

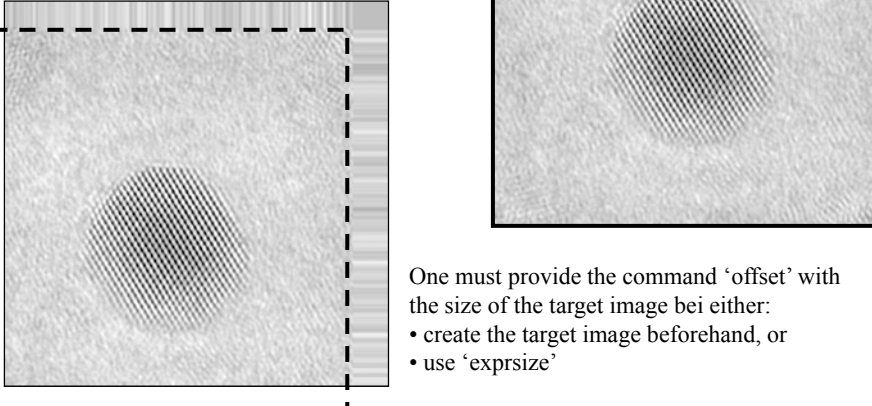
[F_q = Fourier components of f(x)]

MPI for Intelligent Systems

STEM The DM function 'offset(img,dx,dy)'



offset vector



One must provide the command 'offset' with the size of the target image bei either:

- create the target image beforehand, or
- use 'exprsize'

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy

Expand images using 'offset'

```

expandImage
number width,height
number edge = 100;

getnumber("Enter width of edge to create around image:",edge,edge);
Image img=getfrontimage()
img.getSize(width,height)

Image imgoffsets = exprsize(width+2*edge,height+2*edge,offset(img,-edge,-edge))
showimage(imgoffsets)
    
```

MPI for Intelligent Systems

STEM Stuttgart Center for Electron Microscopy


One dimensional derivative: Edge detection

```

horzLineDetect
number width,height
Image img=getfrontimage()
img.getSize(width,height)

Image imgoffsets1 = exprsize(width,height,offset(img,0,1))
Image imgoffsets2 = exprsize(width,height,offset(img,0,-1))
showimage((img-imgoffsets1)*(imgoffsets2-img))
    
```

MPI for Intelligent Systems

STEM Computing the Laplacian by convolution 

Real space:

$$\Delta f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

$$\Delta f(x, y) = f \otimes D_{xy}^2 \quad \mathbf{D}_{xy}^2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$


Reciprocal space:

$$\Delta f = \left(\frac{d^2}{dx^2} + \frac{d^2}{dy^2} \right) \sum F_{q_x, q_y} \cdot e^{2\pi i(q_x x + q_y y)}$$

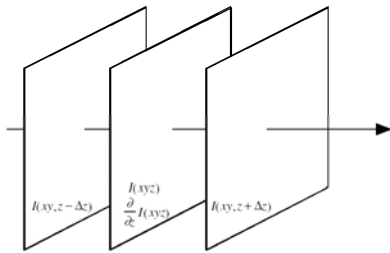
$$= -4\pi^2 \sum (q_x^2 + q_y^2) \cdot F_{q_x, q_y} \cdot e^{2\pi i(q_x x + q_y y)}$$

$$\Rightarrow \Delta f = FT^{-1} \left\{ -4\pi^2 \cdot FT[f] \cdot (q_x^2 + q_y^2) \right\}$$

MPI for Intelligent Systems

STEM Transport of Intensity Equation (TIE) 

The wave function satisfies the Schrödinger eqn in free space (Fresnel propagation):

$$\left(2ik \frac{\partial}{\partial z} + \nabla_{xy}^2 + 2k^2 \right) \psi(xyz) = 0$$


The phase of the electron wave is then given by:

$$\phi(xyz) = -\frac{2\pi}{\lambda} \nabla_{xy}^{-2} \nabla_{xy} \bullet \left(\frac{1}{I(xyz)} \nabla_{xy} \nabla_{xy}^{-2} \frac{\partial}{\partial z} I(xyz) \right)$$

where

$$\psi(xyz) \equiv \sqrt{I(xyz)} \exp\{i\phi(xyz)\} \exp\{i\mathbf{k}\mathbf{r}\}$$

Inverse Laplace operator

MPI for Intelligent Systems

STEM The Modulation Transfer Function (MTF)

Stuttgart Center for Electron Microscopy



A sharp image produced by the electron wave on the detector will be smeared by “cross-talk” between the pixels of the detector.

If an electron hits the scintillator (or phosphor screen) above a certain CCD pixel, then the neighboring pixels may also receive a few photons (this is also true for film and imaging plates).

The resulting image is the convolution of the original image with the MTF:

$$I_{\text{exp}}(\vec{r}) = I_{\text{ideal}}(\vec{r}) \otimes FT^{-1}[MTF(\vec{q})]$$

$$= FT^{-1}\{FT[I_{\text{ideal}}(\vec{r})] \cdot MTF(\vec{q})\}$$

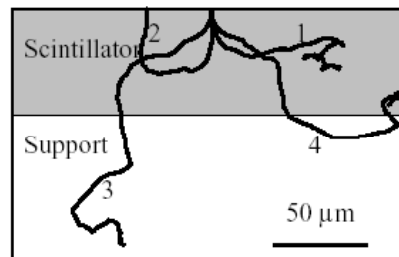
MPI for Intelligent Systems

STEM Detector Point Spread Function

Stuttgart Center for Electron Microscopy



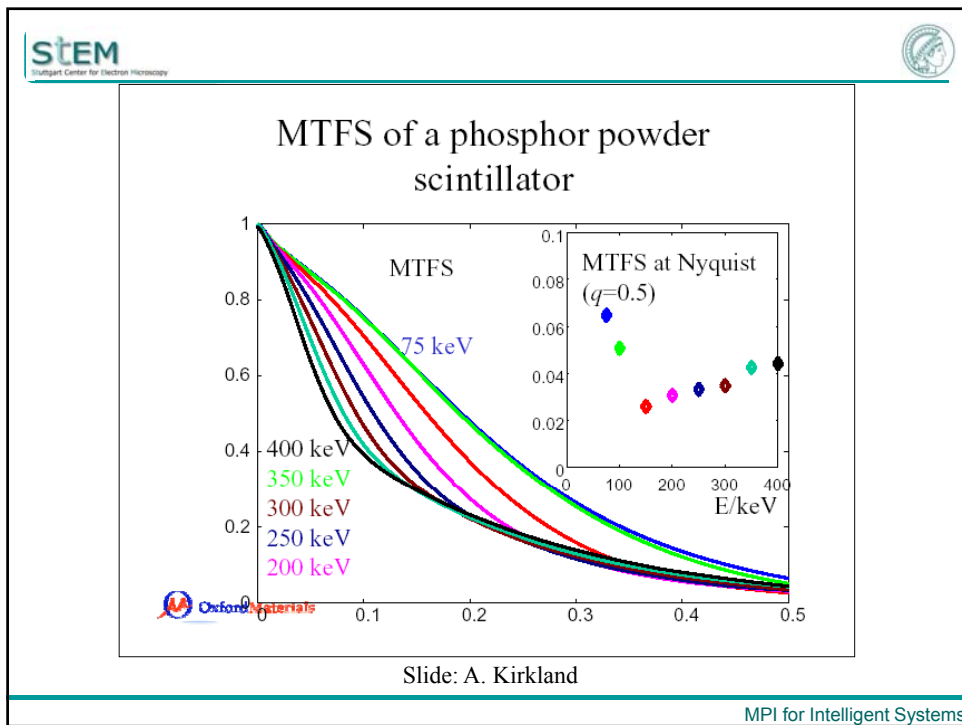
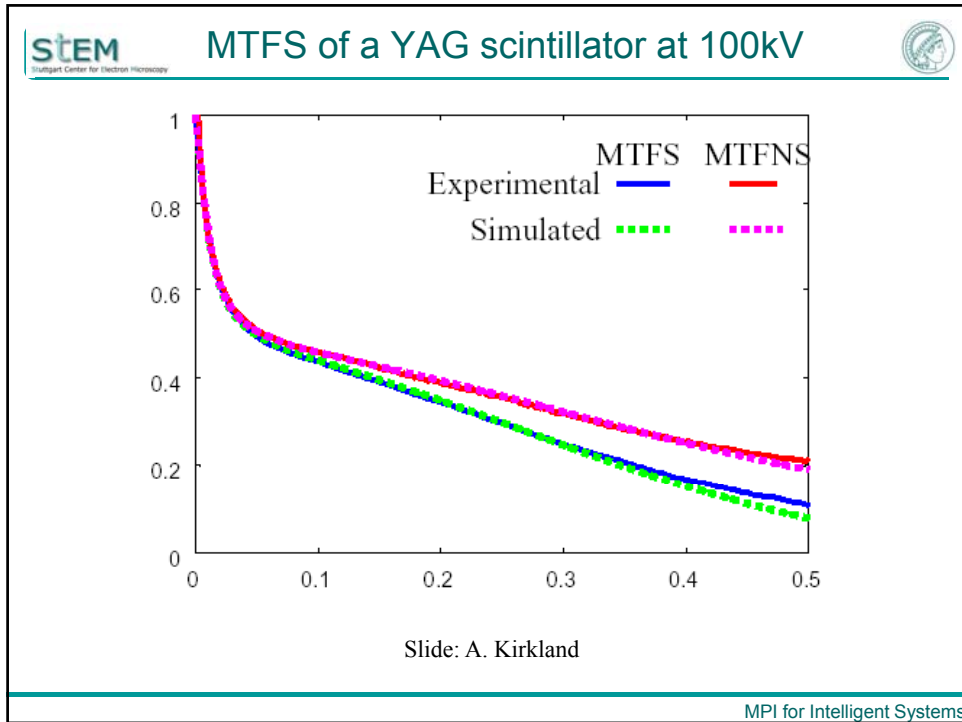
- 1.) Stopped in the scintillator.
- 2.) Back-scattered from the scintillator.
- 3.) Stopped in the support.
- 4.) Back-scattered from the support.

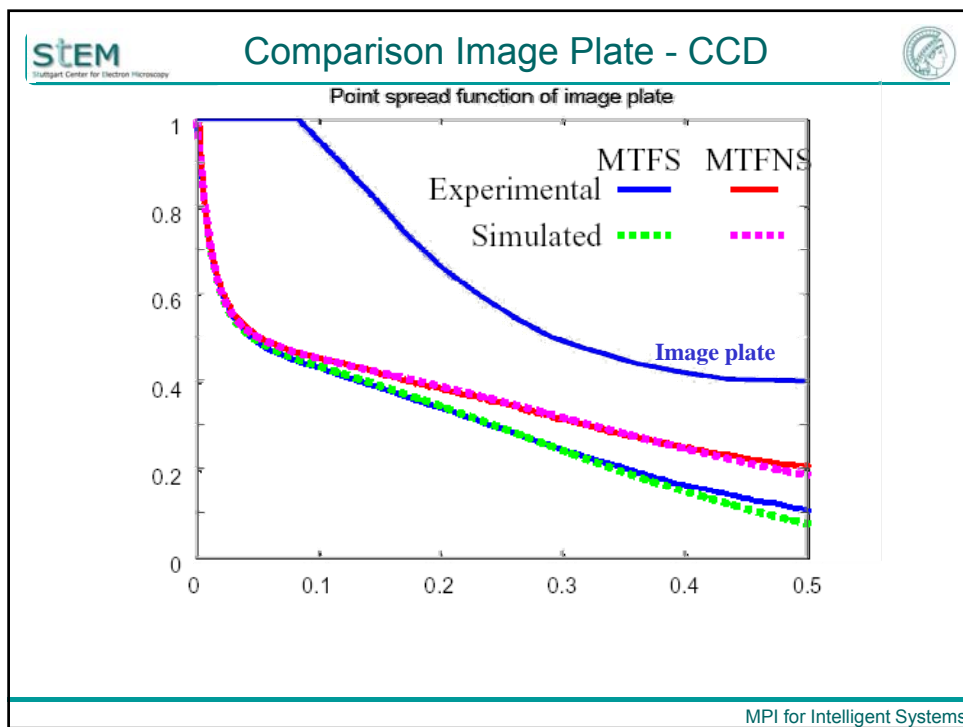
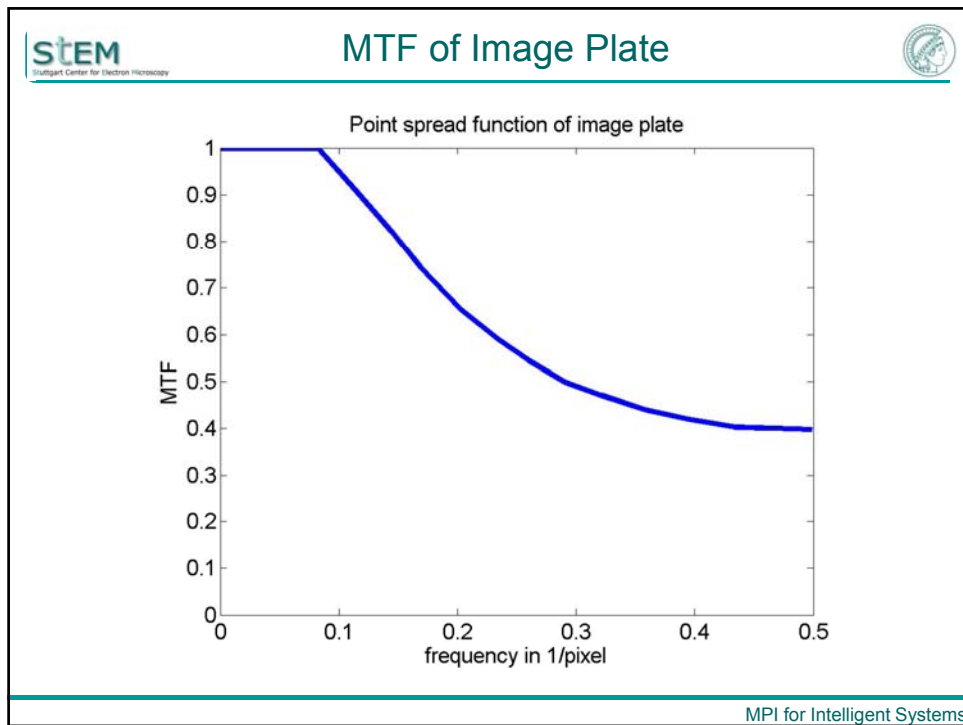



Voltage	100 kV	200 kV	300 kV	400 kV
Type 1	82 %	16 %	0.6 %	0 %
Type 2	18 %	20 %	10 %	5 %
Type 3	0 %	48 %	66 %	70 %
Type 4	0 %	16 %	22 %	25 %

Slide: A. Kirkland

MPI for Intelligent Systems








Stuttgart Center for Electron Microscopy

Convolution -> Deconvolution



Convolution of an image with the detector MTF (also called point spread function [PSF]):

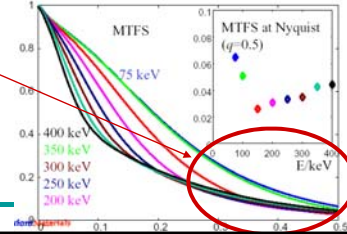
$$I_{\text{exp}}(\vec{r}) = I_{\text{ideal}}(\vec{r}) \otimes FT^{-1}[MTF(\vec{q})]$$


$$= FT^{-1}\{FT[I_{\text{ideal}}(\vec{r})] \cdot MTF(\vec{q})\}$$

De-Convolution of an image with the detector MTF:

$$I_{\text{ideal}}(\vec{r}) = FT^{-1}\{FT[I_{\text{exp}}(\vec{r})] / MTF(\vec{q})\}$$


Problem: At high frequencies the MTF(q) is very small (division by small numbers!) and $I_{\text{exp}}(\vec{r})$ may be dominated by noise.
=> **Noise Amplification!**





Stuttgart Center for Electron Microscopy

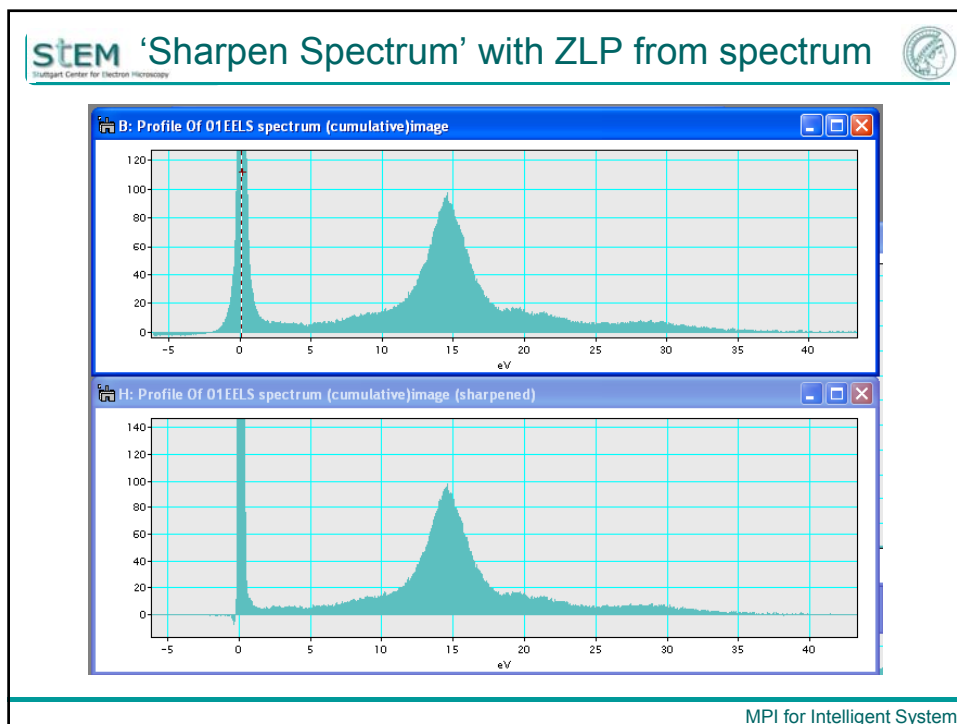
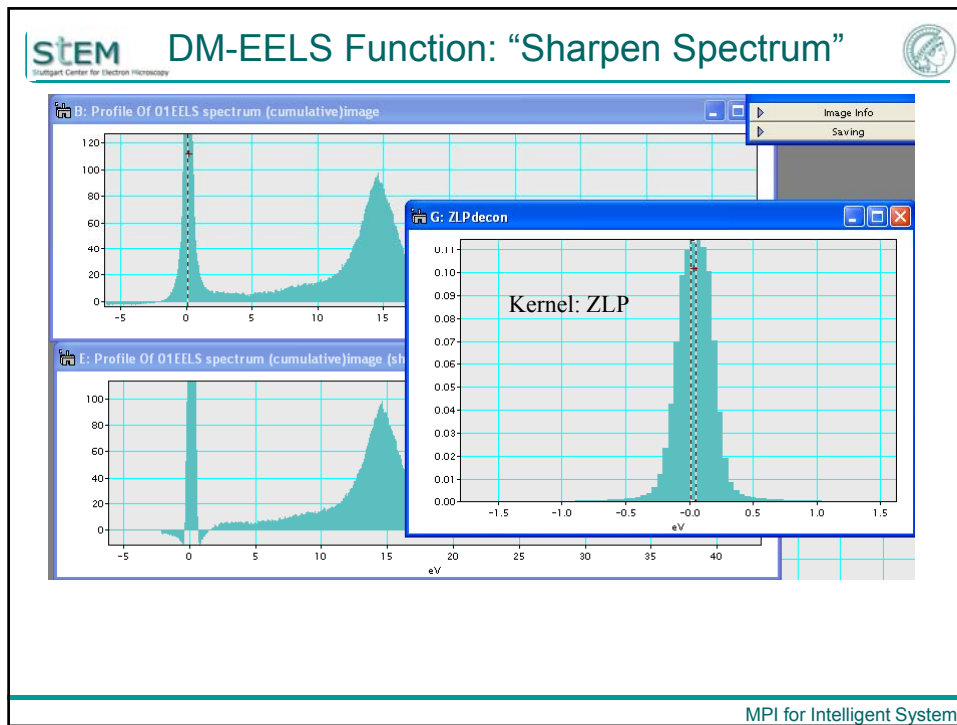
Avoiding Noise Amplification




Possible solutions to avoid noise amplification are:

1. Impose an upper limit on $1/MTF(q)$
2. Lower the upper limit on $1/MTF(q)$ with increasing q (S/N ratio usually decreases)
3. Let $1/MTF(q)$ go to zero above a certain resolution q (ideally q should match the resolution present in the image data)
4. Make sure that the deconvolution kernel (e.g. $MTF(q)$) is smooth. Otherwise this makes errors even worse.
5. Use Richardson-Lucy deconvolution
6. Use Maximum Likelihood deconvolution


MPI for Intelligent Systems






Stuttgart Center for Electron Microscopy

How does 'Sharpen Spectrum' work?




1. Fit a Gaussian to the narrow central portion (within 90% of the maximum, minimum of 3 channels) of the zero-loss peak [ZLP] (=Convolution kernel)
2. Replace that portion of the ZLP with a δ -function of equal area.
3. Subtract the fitted Gaussian from the original data and set negative pixels of the resulting "Gaussian-subtracted ZLP" to zero.
4. Place the total Intensity difference between the ZLP and the "Gaussian-subtracted ZLP" in a single pixel at the position of the ZLP maximum (\Rightarrow delta-function).
5. Normalize this modified ZLP to 1 and make sure the delta-function is in pixel(0,0).

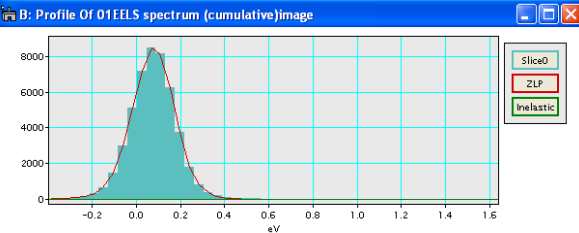
MPI for Intelligent Systems



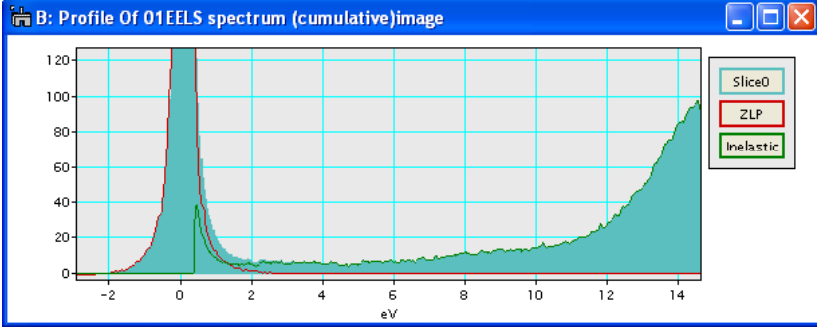
Stuttgart Center for Electron Microscopy

How does the ZLP-extraction work





1. A symmetric ZLP is assumed. The right hand side of the ZLP below $\frac{1}{4}$ of its height is replaced by its left hand side.
2. Negative counts in the "inelastic spectrum" (original - ZLP) are set to zero.



MPI for Intelligent Systems

STEM EELS Multiple Scattering Deconvolution

Stuttgart Center for Electron Microscopy



Assuming independent scattering events the intensity in an experimental EELS spectrum can be simulated by the expression

$$\begin{aligned}
 I_{\text{exp}}(E) &= ZLP(E) \otimes \left[\frac{t}{\lambda} I_{\text{theor}}(E) + \frac{1}{2!} \left(\frac{t}{\lambda} I_{\text{theor}}(E) \right)^2 \otimes \left(\frac{t}{\lambda} I_{\text{theor}}(E) \right) + \dots \right] \\
 &= ZLP(E) \otimes FT^{-1} \left[\exp \left\{ \frac{t}{\lambda} FT[I_{\text{theor}}(E)] \right\} \right] \\
 &= FT^{-1} \left\{ FT[ZLP(E)] \cdot \exp \left(\frac{t}{\lambda} FT[I_{\text{theor}}(E)] \right) \right\}
 \end{aligned}$$

This means, in order to extract the true spectrum $I_{\text{theor}}(E)$ from an experimental spectrum one must first deconvolute by the ZLP as precisely as possible.

MPI for Intelligent Systems

STEM EELS Multiple Scattering Deconvolution (2)

Stuttgart Center for Electron Microscopy



Inverting the previous expression, one can extract the single scattered spectrum from the experimental data according to

$$\frac{t}{\lambda} I_{\text{ss}}(E) = FT^{-1} \left\{ \ln \left(\frac{FT[I_{\text{theor}}(E)]}{FT[ZLP(E)]} \right) \right\}$$

How does an incomplete deconvolution of the ZLP
(or deconvolution by a smoothed ZLP) affect $I_{\text{ss}}(E)$?

It mainly affects the resolution of $I_{\text{ss}}(E)$. Peak positions and –heights will hardly be affected.

MPI for Intelligent Systems

