Image Reconstruction

Partial k-space Reconstruction

M229 Advanced Topics in MRI

Kyung Sung, Ph.D.

5/14/2024

Today's Topics

- Fourier transform symmetries
  - Odd and Even functions
- Motivation for partial k-space recon
- Partial k-space recon methods
  - Direct method (Homodyne)
  - Iterative method (POCS)
- MATLAB code demo

Even and Odd Functions

- function f is even (or symmetric) when

$$f(x) = f(-x)$$

- function f is odd (or antisymmetric) when

$$f(x) = -f(-x)$$

Even and Odd Functions

- Any function can be written as a sum of even and odd functions

$$f(x) = \frac{1}{2}[f(x) + f(-x) - f(-x) + f(x)]$$

$$= \underbrace{\frac{1}{2}[f(x) + f(-x)]}_{f_e(x)} + \underbrace{\frac{1}{2}[f(x) - f(-x)]}_{f_o(x)}$$

Even and Odd Functions

- The integral of the product of odd and even functions is zero

$$\int_{-\infty}^{\infty} f_e(x) f_o(x) dx$$

$$= \int_{-\infty}^{0} f_e(x) f_o(x) dx + \int_{0}^{\infty} f_e(x) f_o(x) dx$$

$$= \int_{0}^{\infty} [f_e(-x) f_o(-x) dx + f_e(x) f_o(x)] dx$$

Fourier Transform Symmetry

$$F(f) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi x f} dx$$

$$F(f) = \int_{-\infty}^{\infty} f(x)\cos(2\pi x f) dx - j\int_{-\infty}^{\infty} f(x)\sin(2\pi x f) dx$$

$$F(f) = \int_{-\infty}^{\infty} f_e(x)\cos(2\pi x f) dx + \cancelto{0}{\int_{-\infty}^{\infty} f_o(x)\cos(2\pi x f) dx}$$

$$-j\,\cancelto{0}{\int_{-\infty}^{\infty} f_e(x)\sin(2\pi x f) dx} - j\int_{-\infty}^{\infty} f_o(x)\sin(2\pi x f) dx$$

Fourier Transform Symmetry

$$F(f) = \int_{-\infty}^{\infty} f_e(x) \cos(2\pi x f) dx - j \int_{-\infty}^{\infty} f_o(x) \sin(2\pi x f) dx$$

$$F(f) = F_e(f) + F_o(f)$$

real & even function?

real & odd function?

even function?

odd function?

Fourier Transform Symmetry

- Fourier transform of even part (of a real function) is real

$$FT\{f_e(x)\} = Re\{F_e(f)\}$$

- Fourier transform of even part is even

$$FT\{f_e(x)\} = F_e(f) = F_e(-f)$$

Fourier Transform Symmetry

- Fourier transform of odd part (of a real function) is imaginary

$$FT\{f_o(x)\} = Im\{F_o(f)\}$$

- Fourier transform of odd part is odd

$$FT\{f_o(x)\} = F_o(f) = -F_o(-f)$$

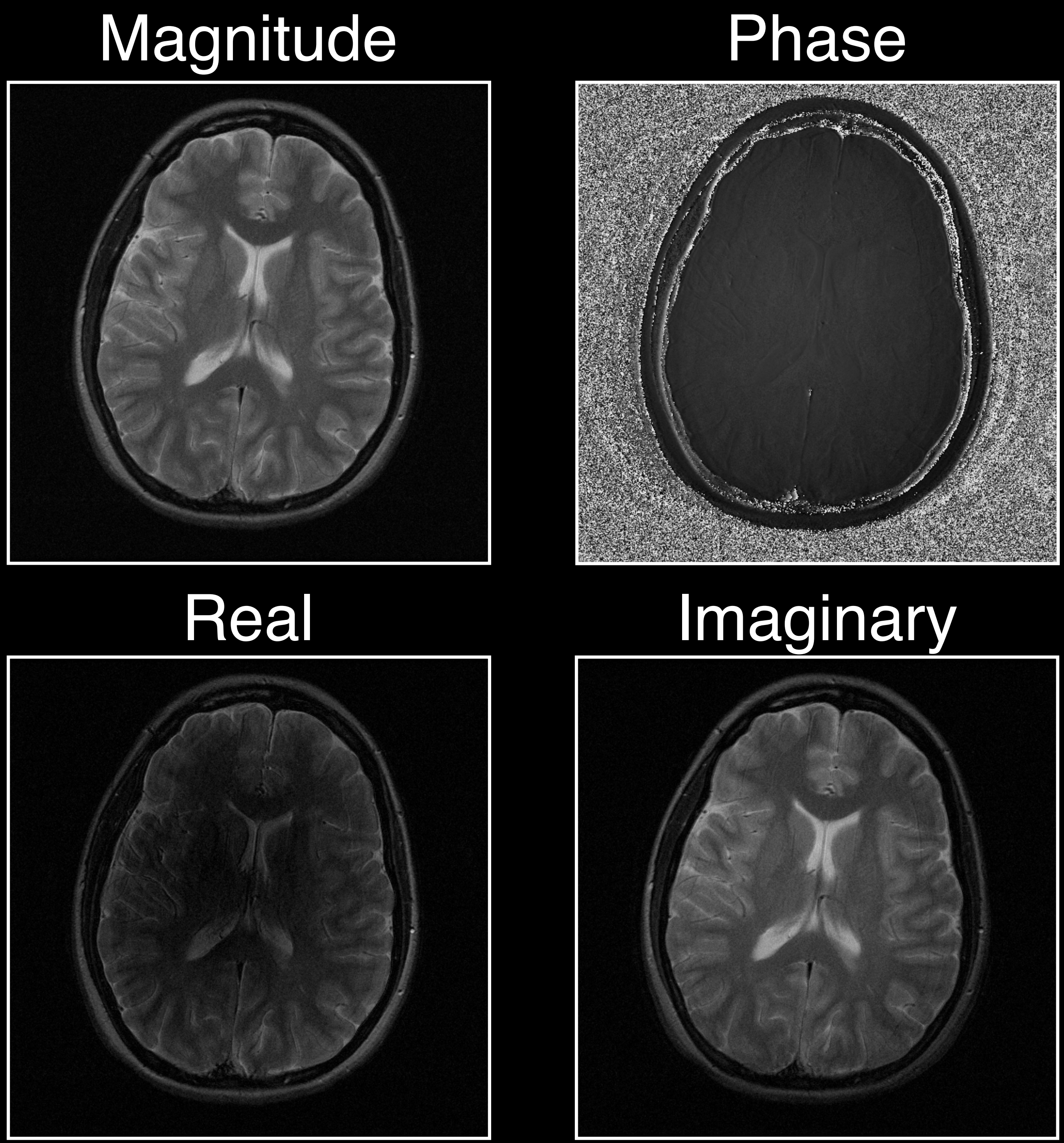
Hermitian Symmetry

- We can summarize all four symmetries possessed by Fourier transform of a real function

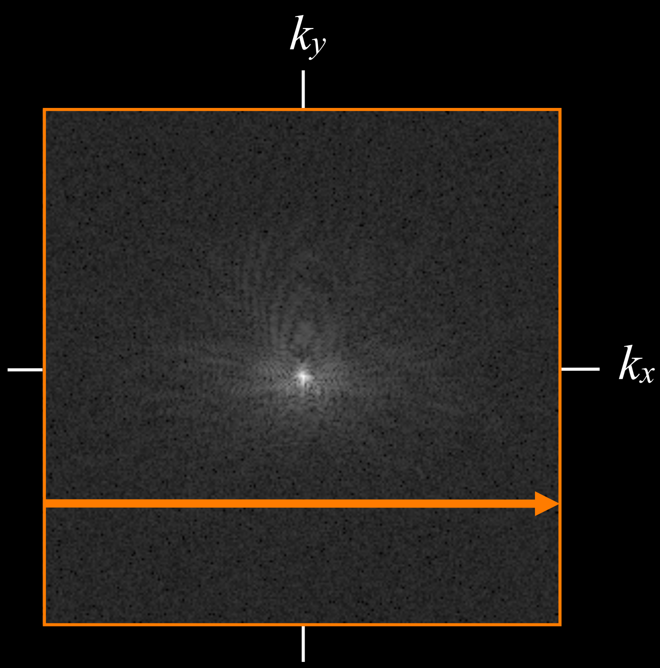
To the board ...

Motivation

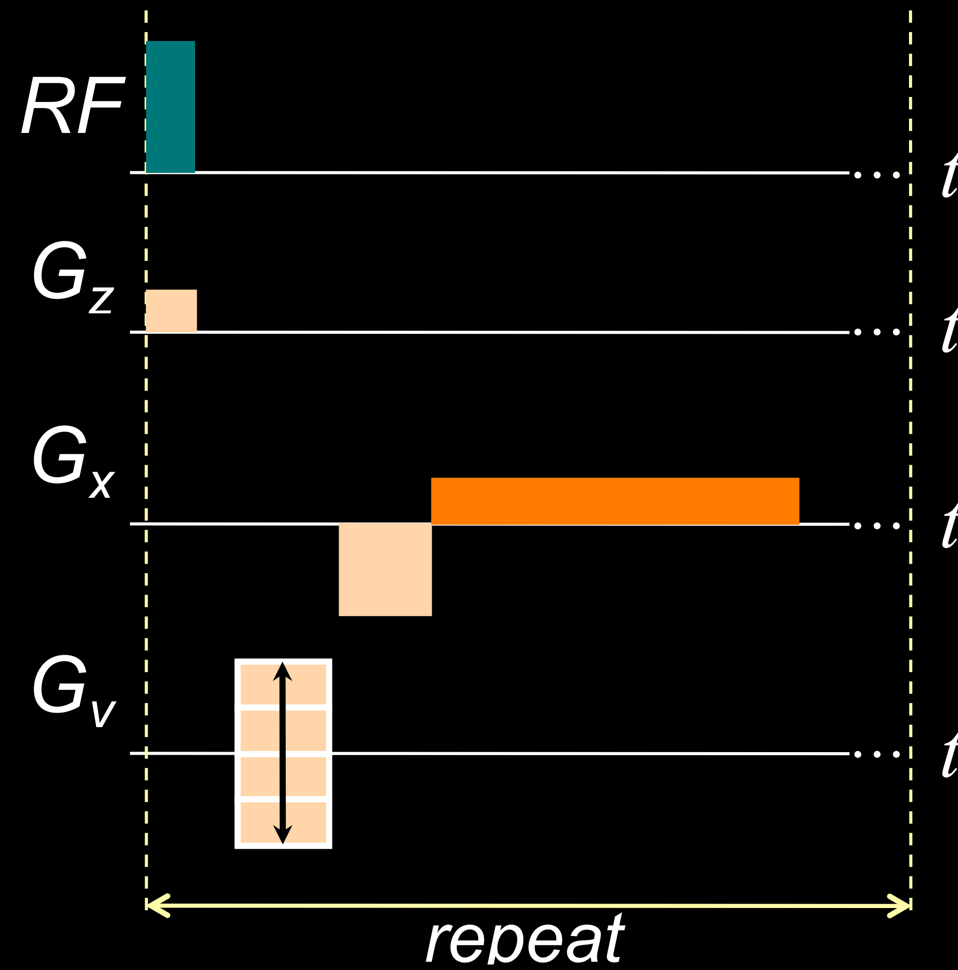
- MR images depict the spin density as a function of position
  - If this is true, only half of k-space data will need to be collected
  - Uncollected data could be synthesized by conjugate symmetry
- However, MR images are not real-valued!
  - Partial k-space reconstruction requires some type of phase correction


Magnitude
Phase
Real
Imaginary

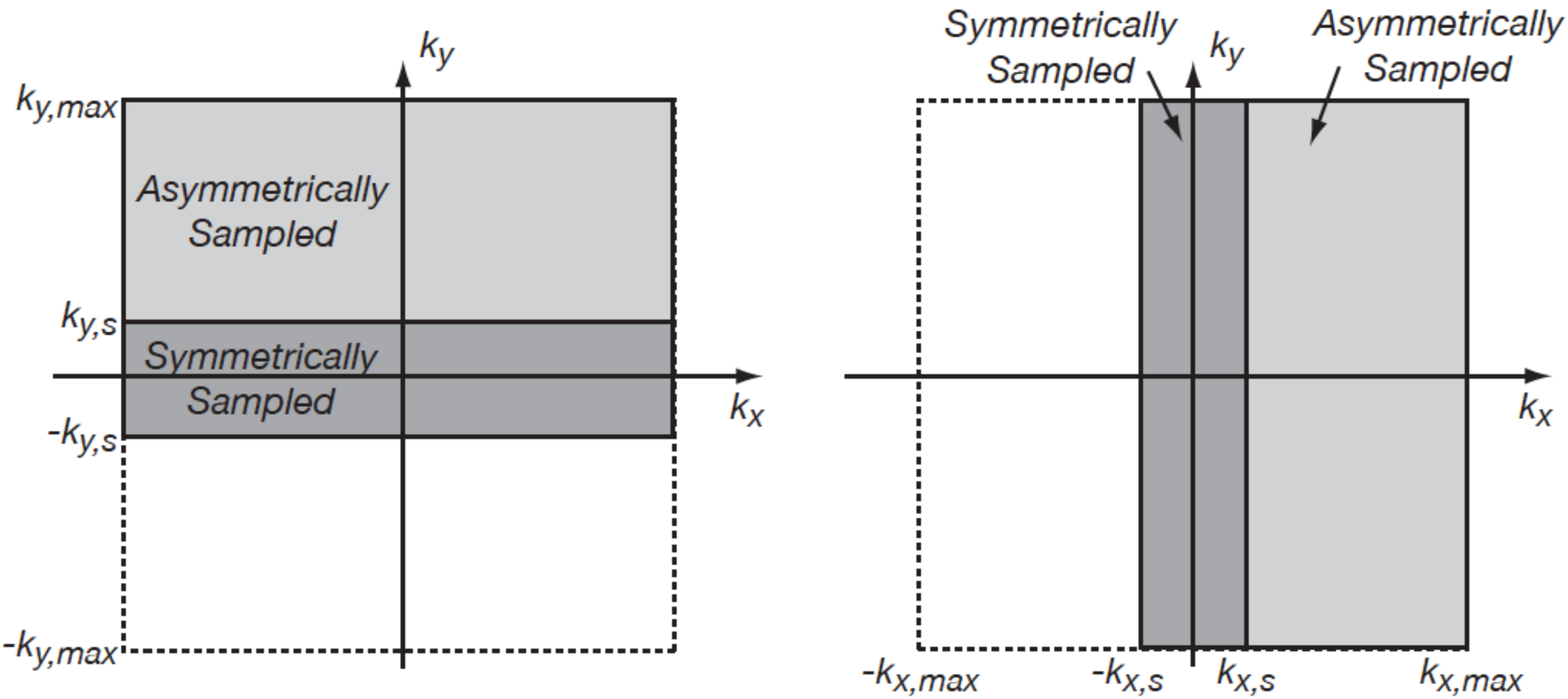
Cartesian 2D Imaging



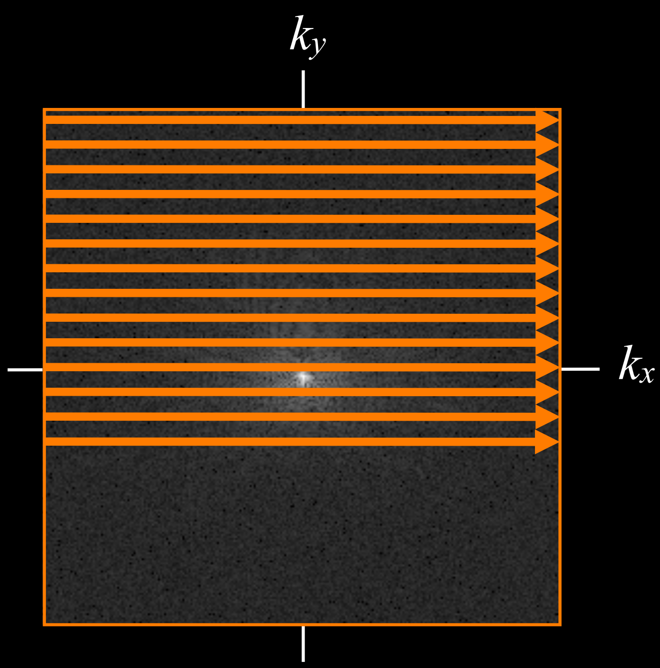
Pulse Sequence Diagram



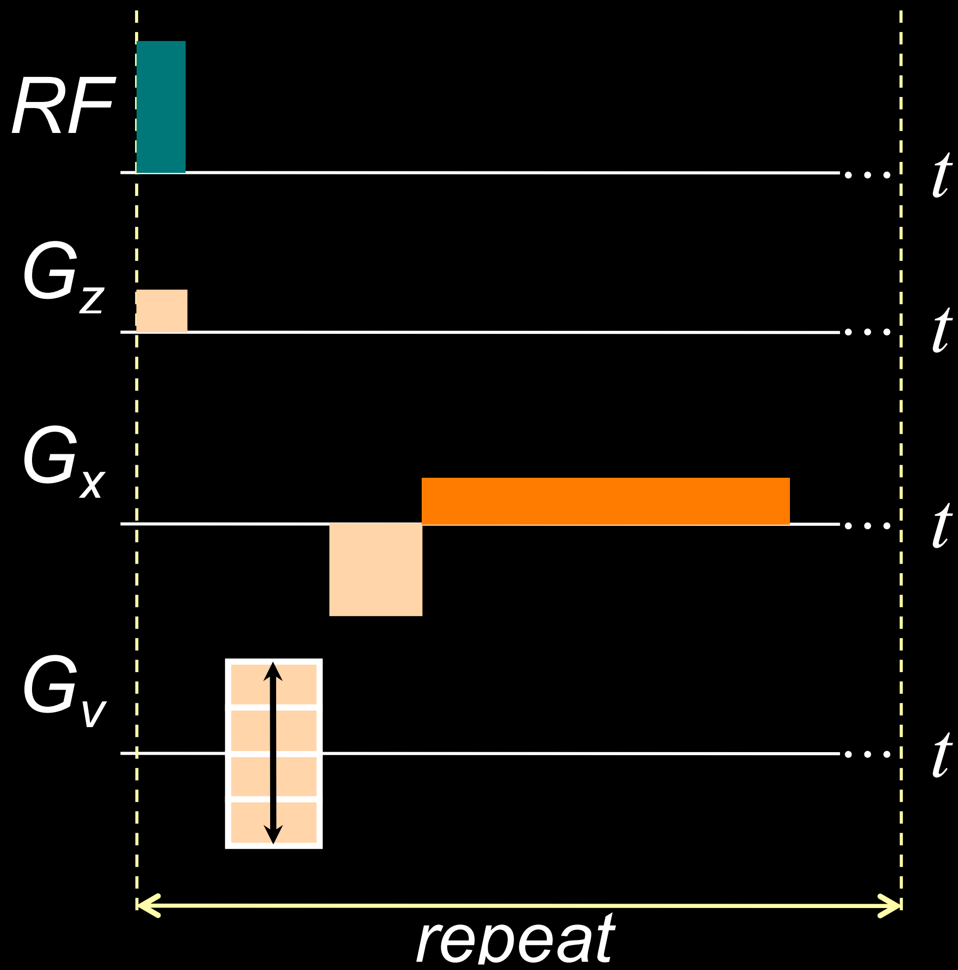
Cartesian Sampling Application



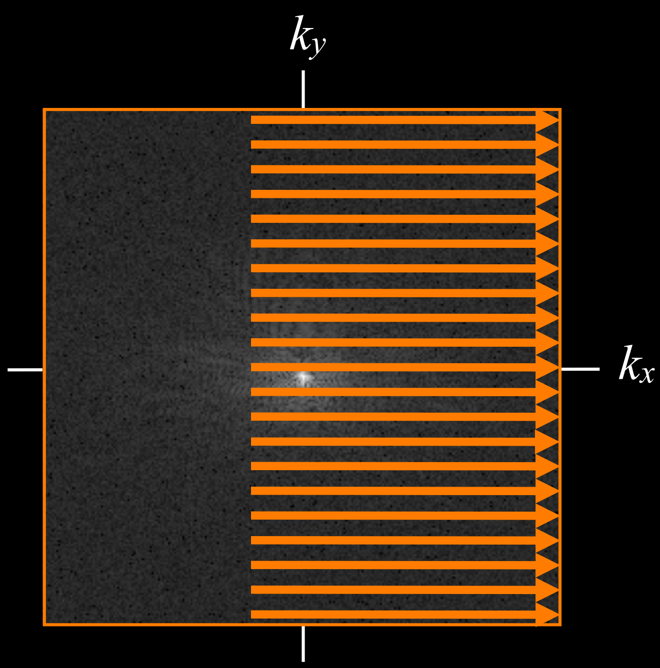
Cartesian 2D Imaging



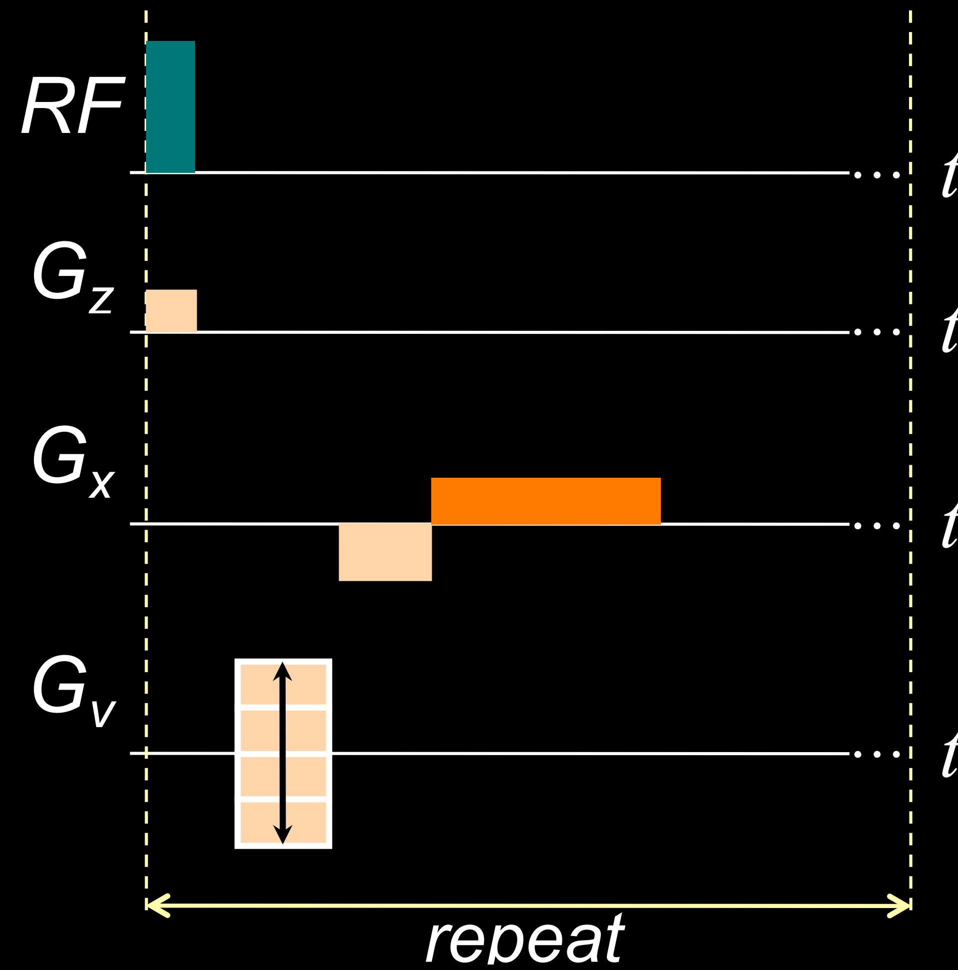
Pulse Sequence Diagram



Cartesian 2D Imaging



Pulse Sequence Diagram



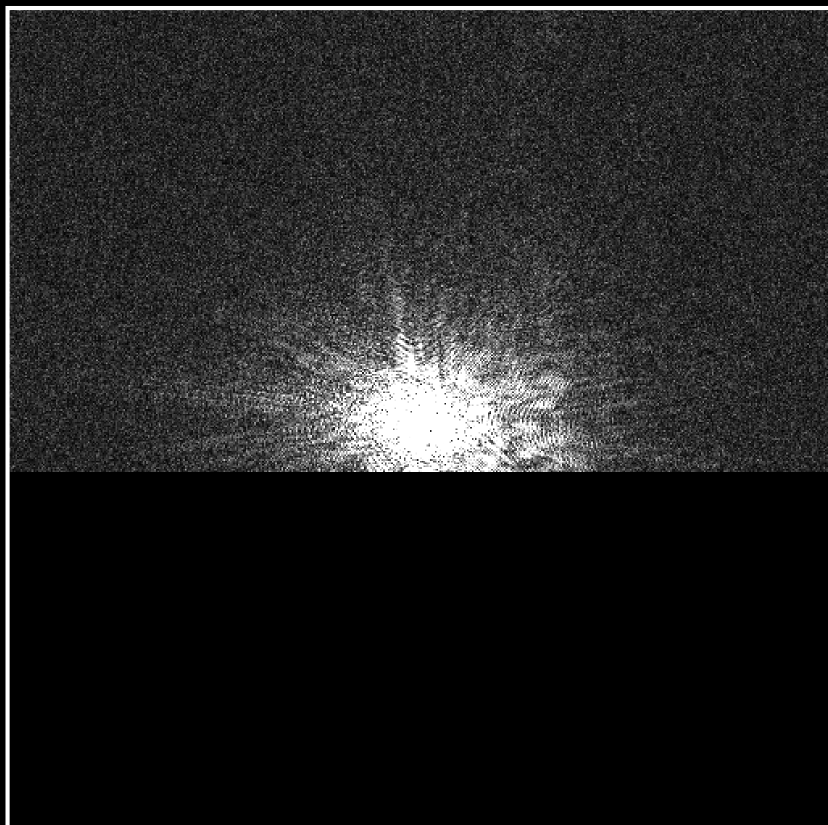
Direct Reconstruction

- Zero padding
- Phase correction and conjugate synthesis
- Homodyne reconstruction

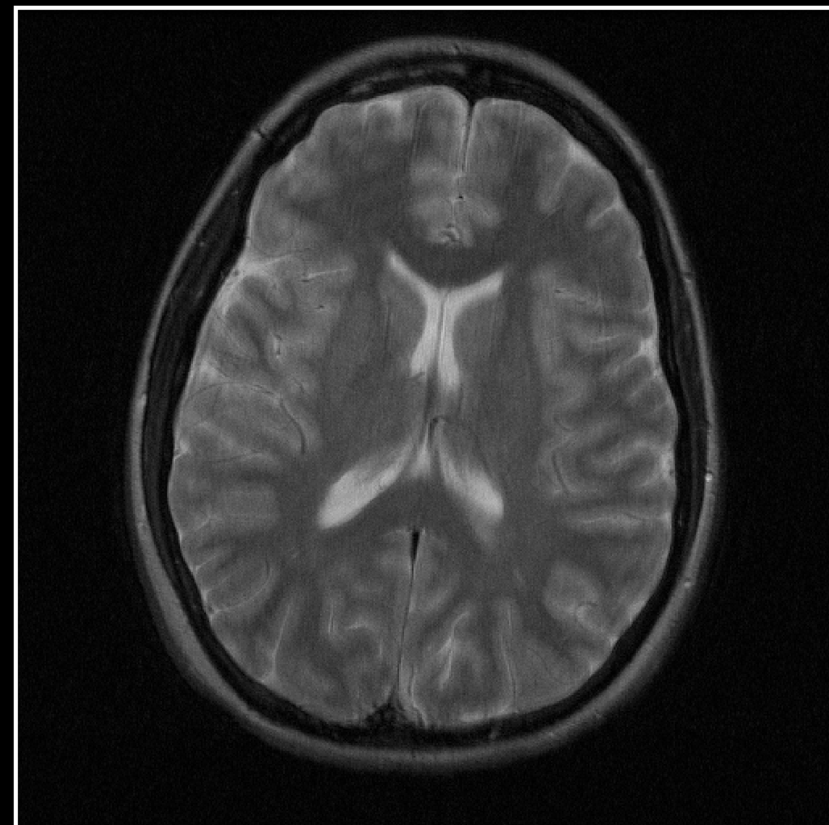
Trivial Recon by Zero-Padding

k-space

$\frac{9}{16}$



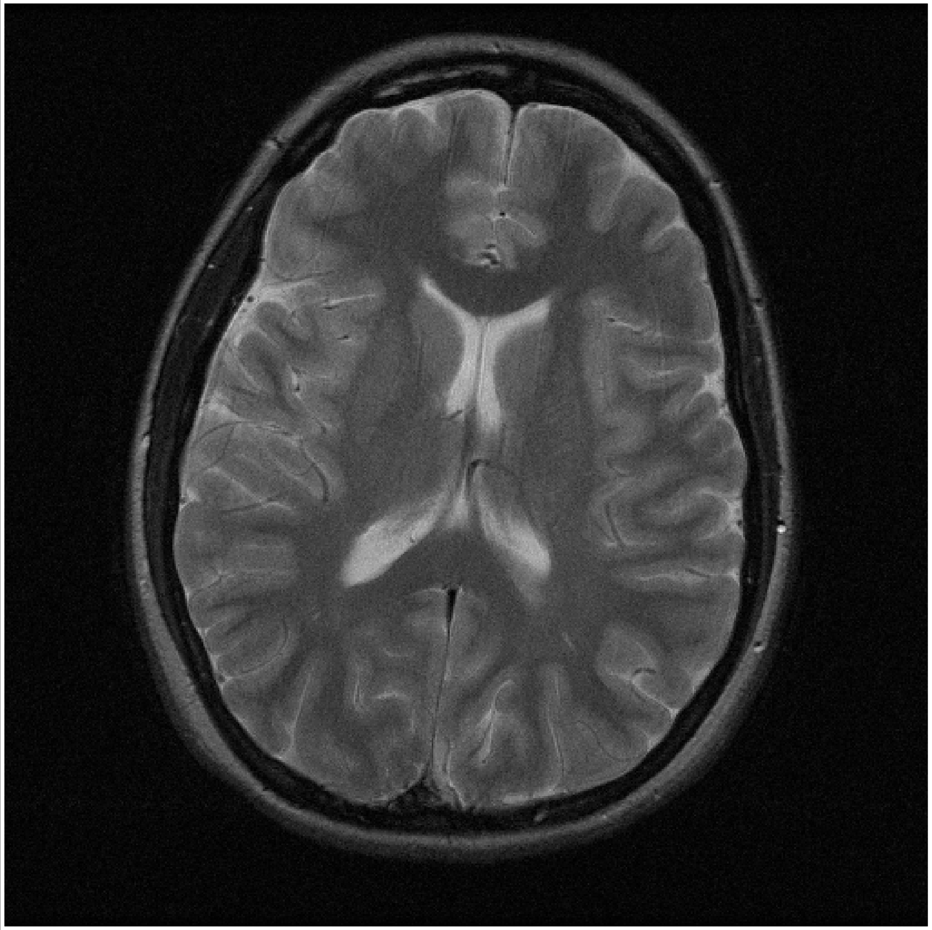
Zero Padding



Fourier Transform

Zero Padding

Original



Zero Padding

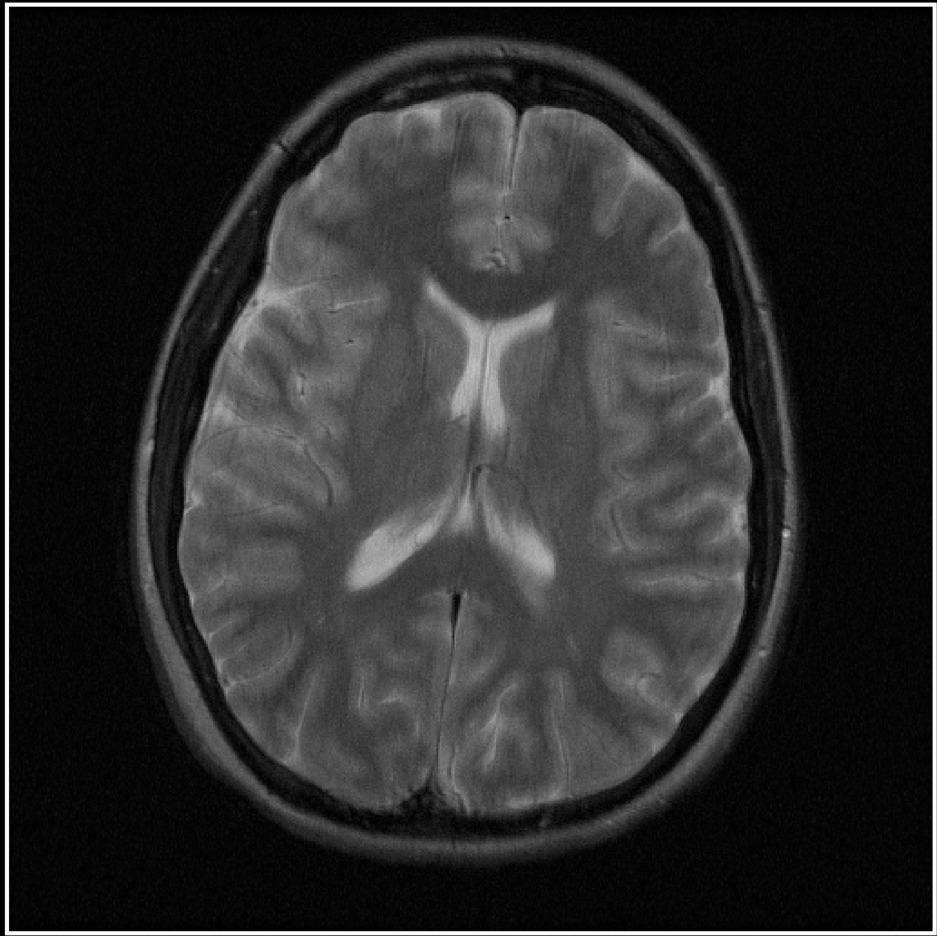
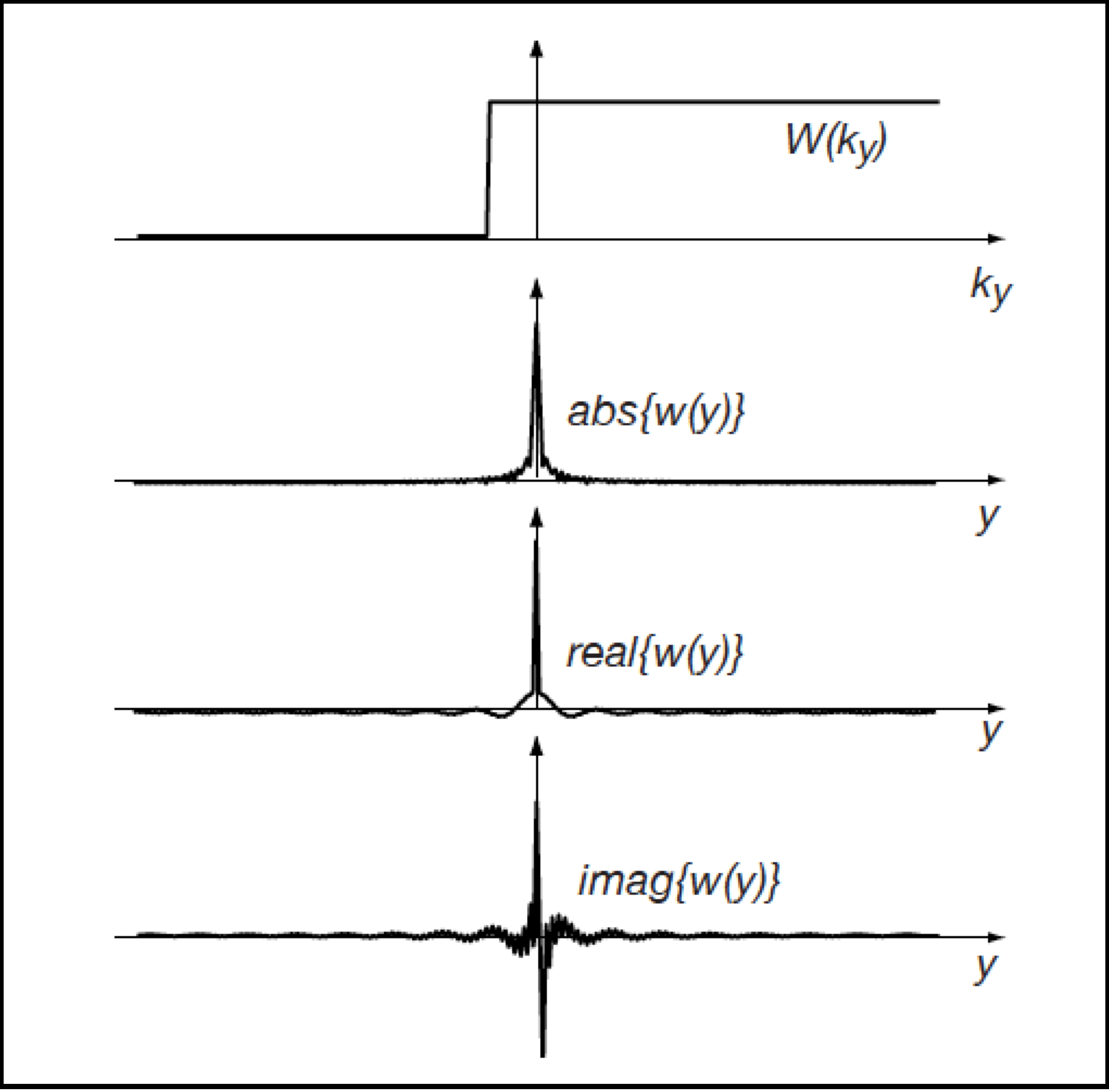


Image Artifacts by Zero Padding

- Blurring can be identified by the product of a full k-space data set multiplied by a weighting function
- The inverse Fourier transform of this weighting function is the impulse response that produces the blurring

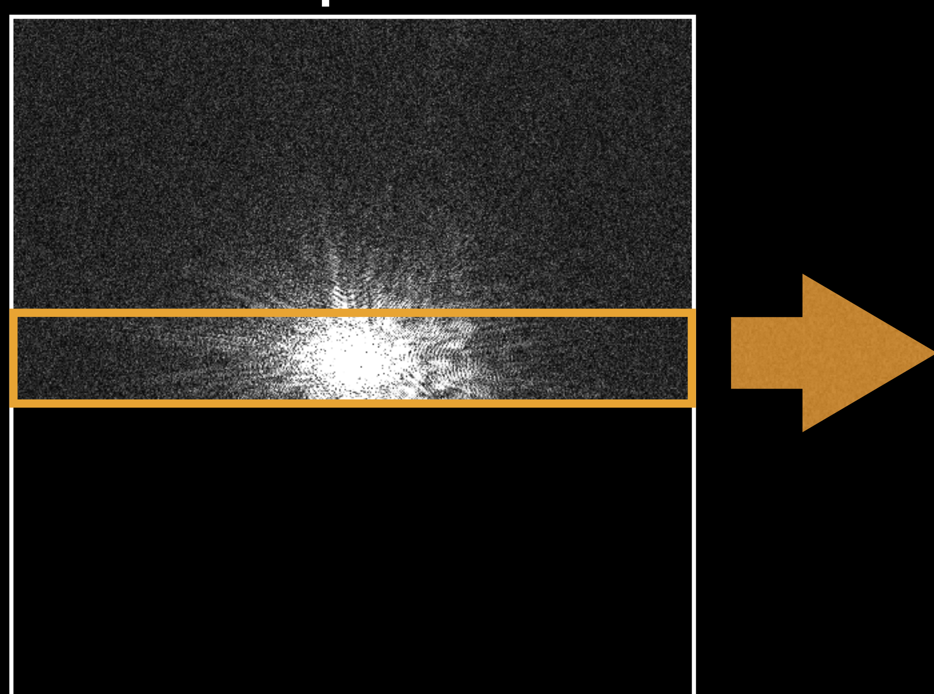
Image Artifacts by Zero Padding



Phase Correction and Conjugate Synthesis

- Phase correction must be applied
  - Use the narrow strip of data for which we have symmetric coverage

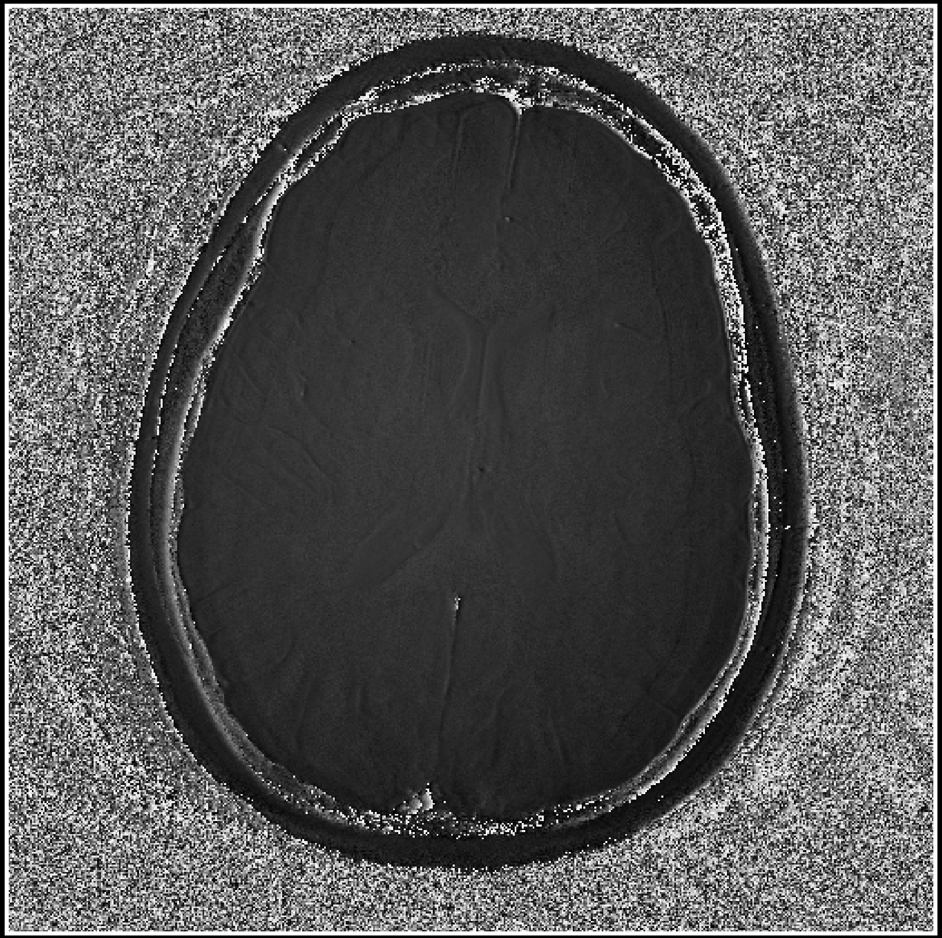
k-space



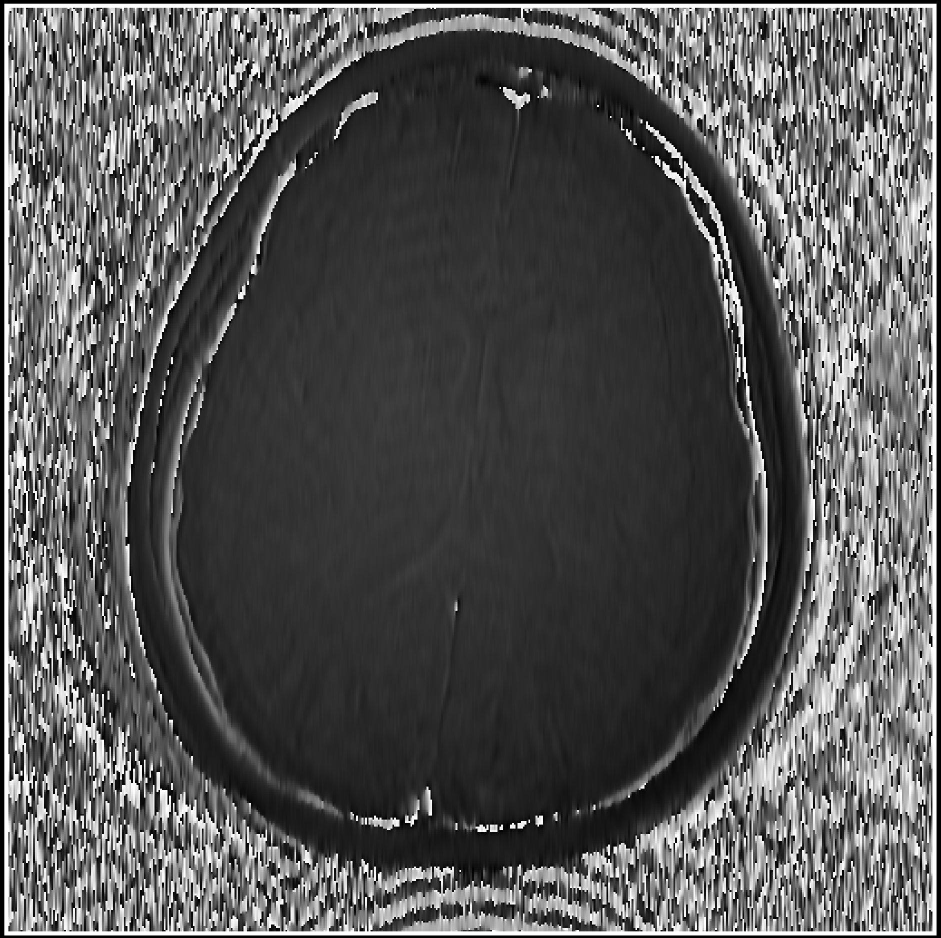
Phase Estimation $p(x,y)$

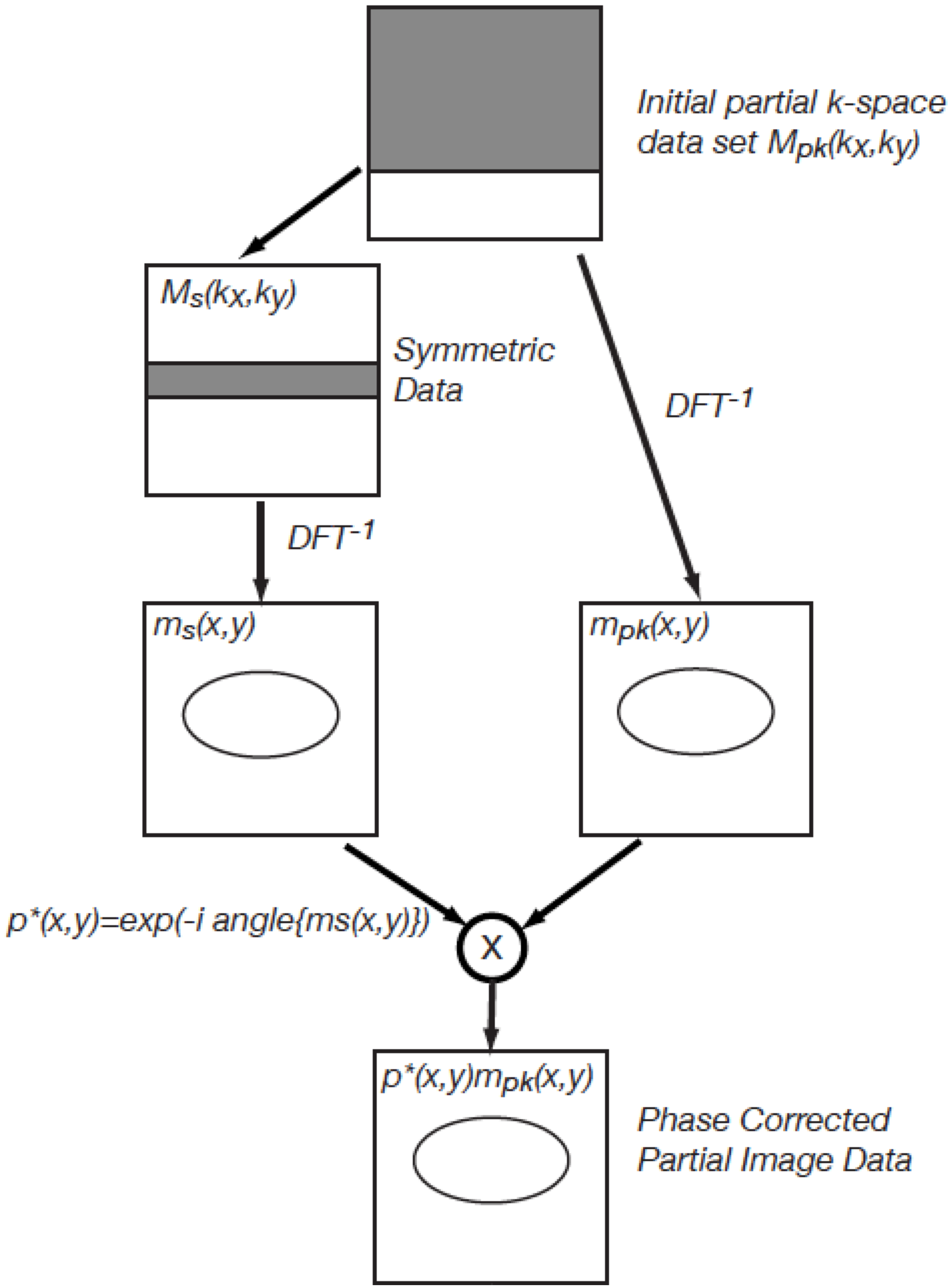
Phase Estimation

Original Phase

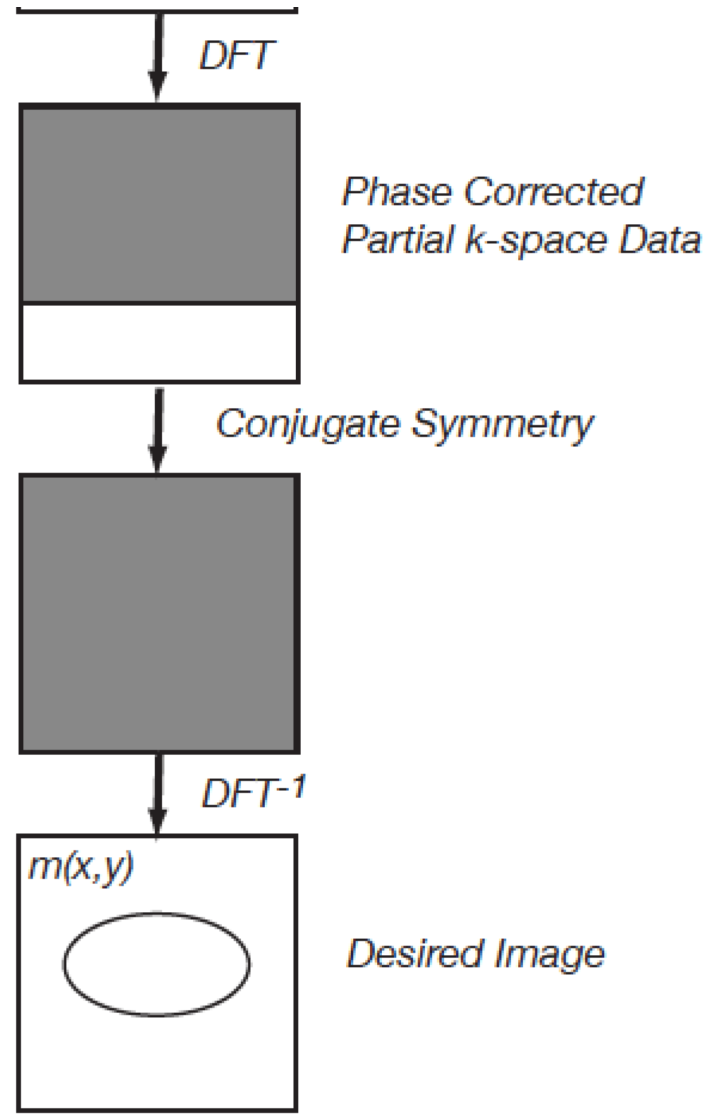


Estimated Phase





Initial partial k-space
data set Mpk(kx,ky)
Ms(kx,ky)
Symmetric
Data
DFT-1
DFT-1
ms(x,y)
mpk(x,y)
p*(x,y)=exp(-i angle{ms(x,y)})
X
p*(x,y)mpk(x,y)
Phase Corrected
Partial Image Data

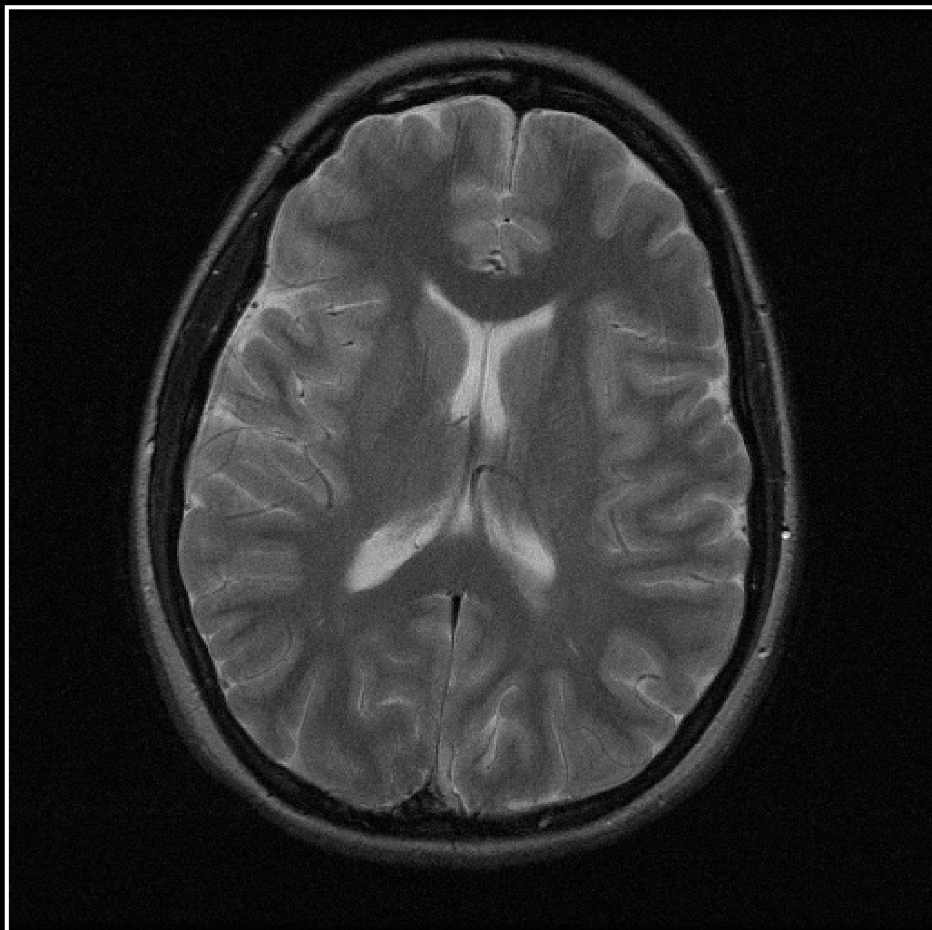

DFT
Phase Corrected
Partial k-space Data
Conjugate Symmetry
DFT-1
m(x,y)
Desired Image

MATLAB Code

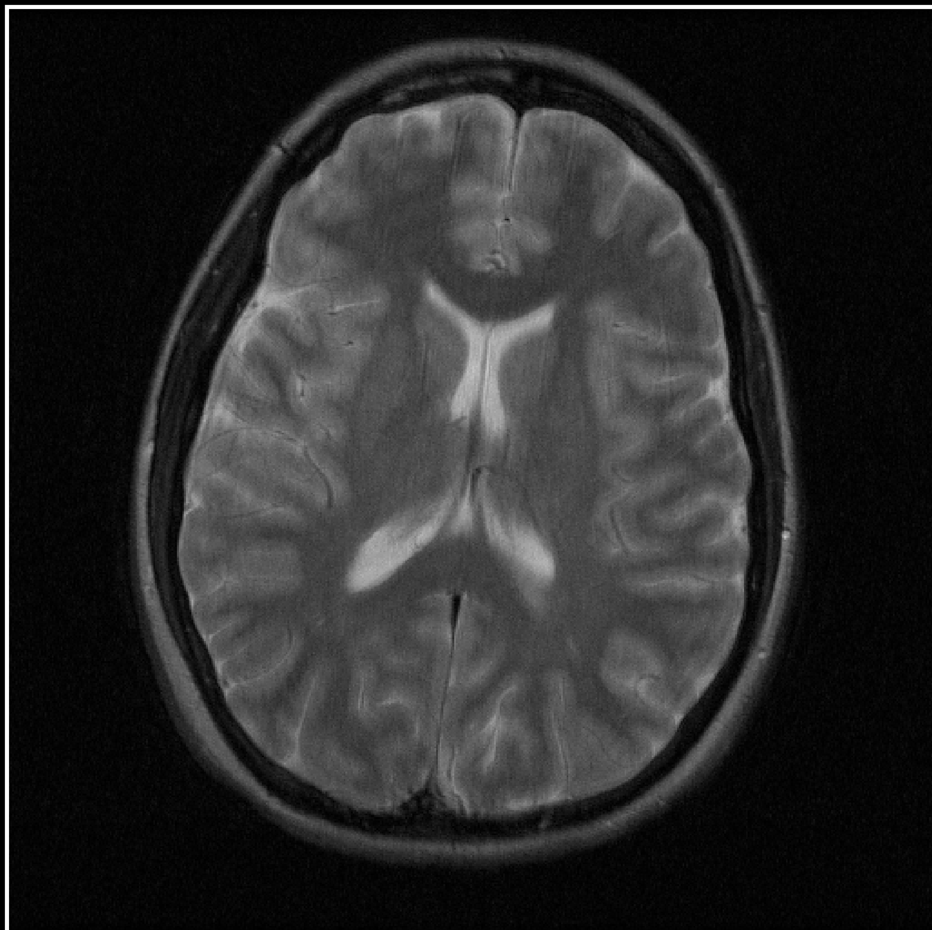
```
hnover = 224; % 7/16 sets to be zeros

data_pk = data;
data_pk(1+nx-hnover:end,:) = 0;

im_zp = fftshift(ifftn(fftshift(data_pk)));

data_center = data_pk;
data_center(1:hnover,:) = 0;

im_ph = fftshift(ifftn(fftshift(data_center)));

im_pc = im_zp.*exp(-1i*angle(im_ph));

data_pc = fftshift(fftn(fftshift(im_pc)));
data_pc(1+nx-hnover:end,:) = 0;

data_pc(1+nx-hnover:end,:) = rot90(data_pc(1:hnover,:),2);

im_pc = fftshift(ifftn(fftshift(data_pc)));
```

Phase Correction and Conjugate Synthesis

Original

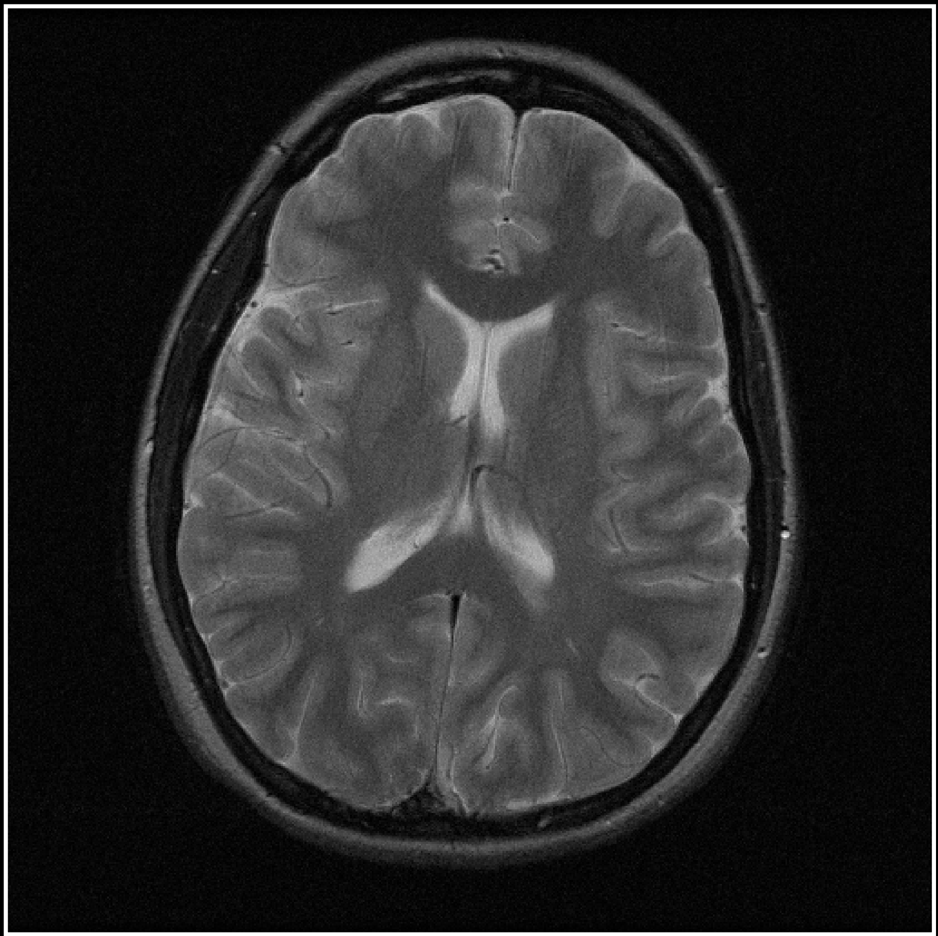


Zero Padding

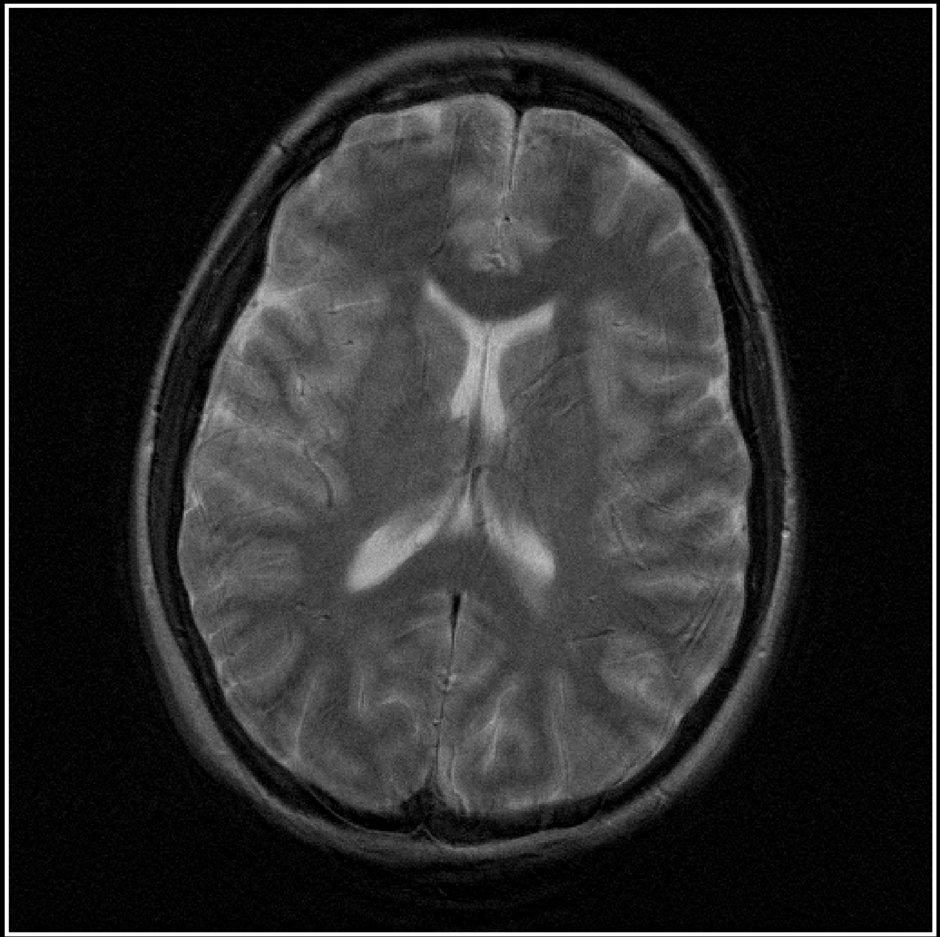


Phase Correction and Conjugate Synthesis

Original

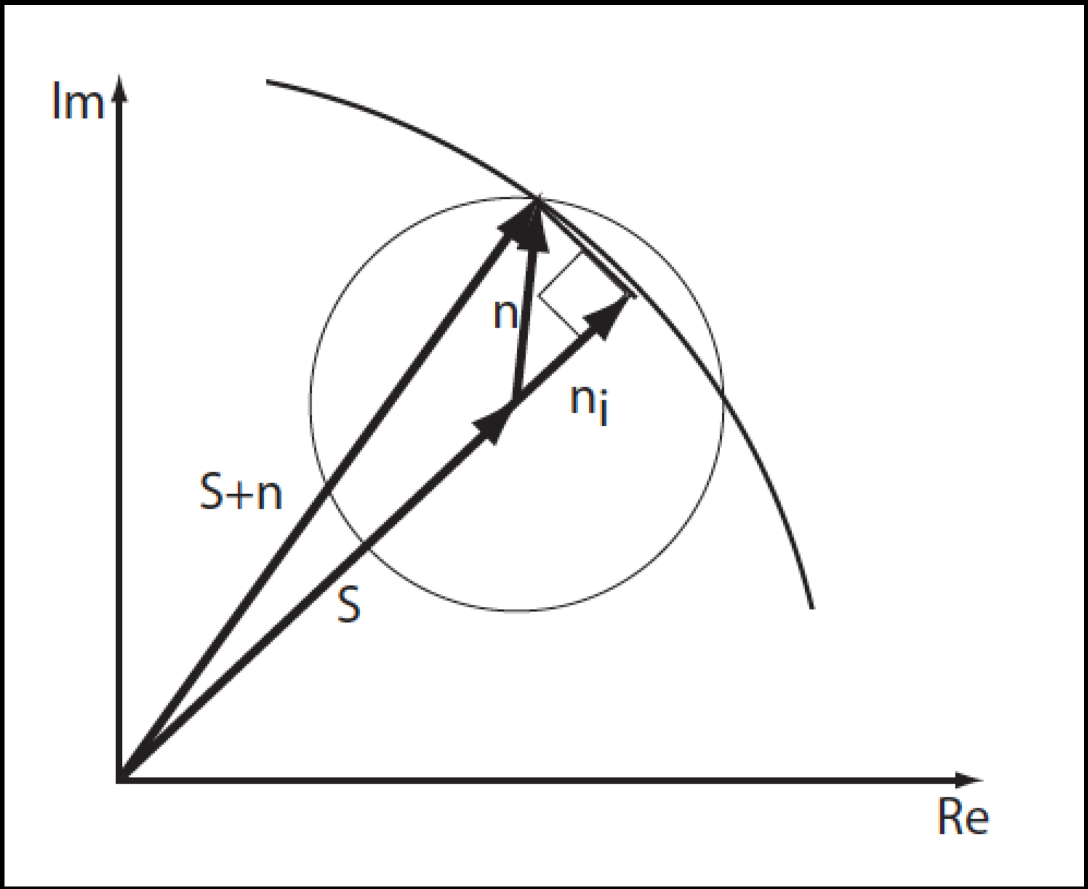


Phase Correction



Noise Consideration

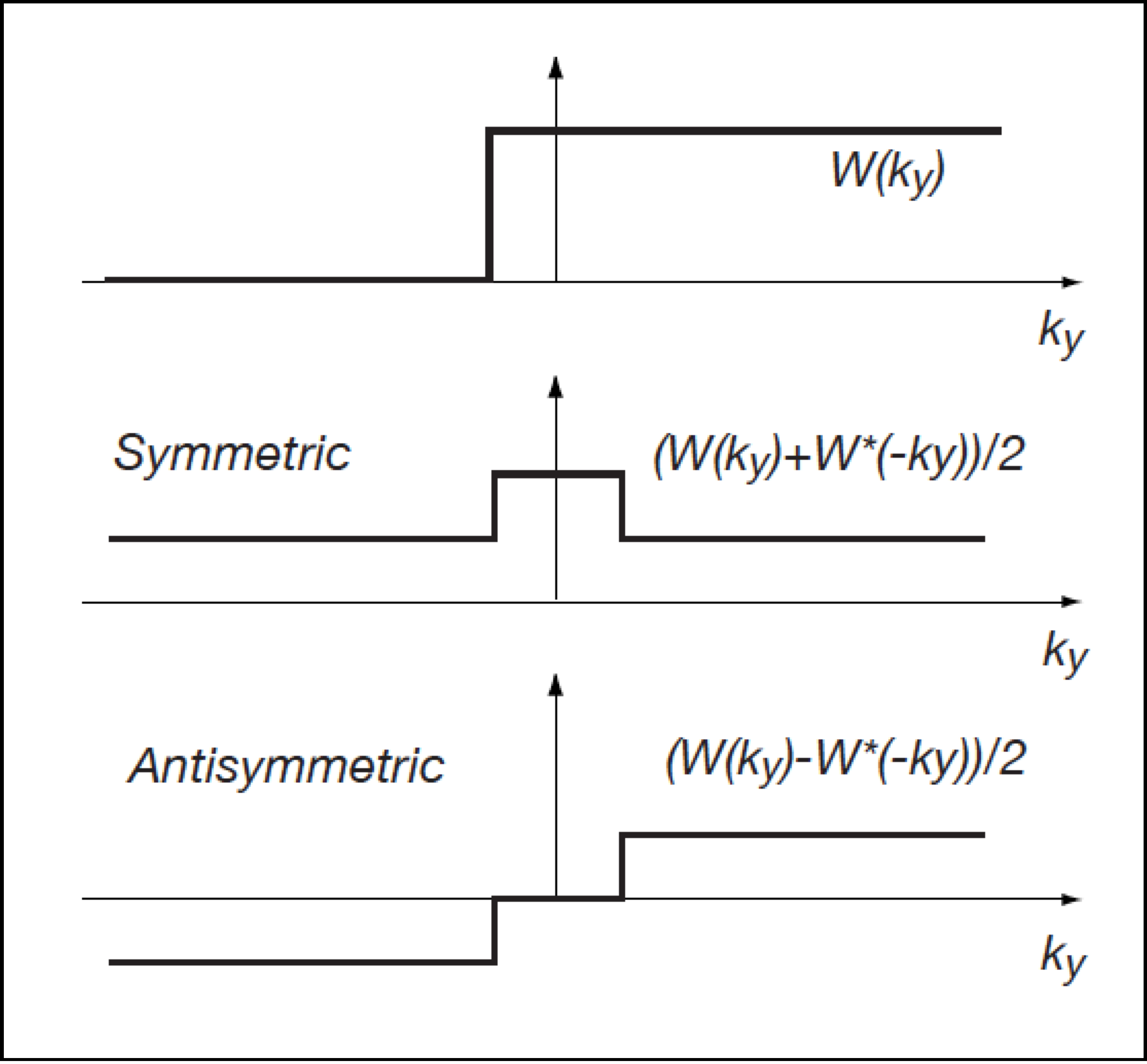
- Background in the phase corrected image has lower noise because one component of the complex noise has been suppressed



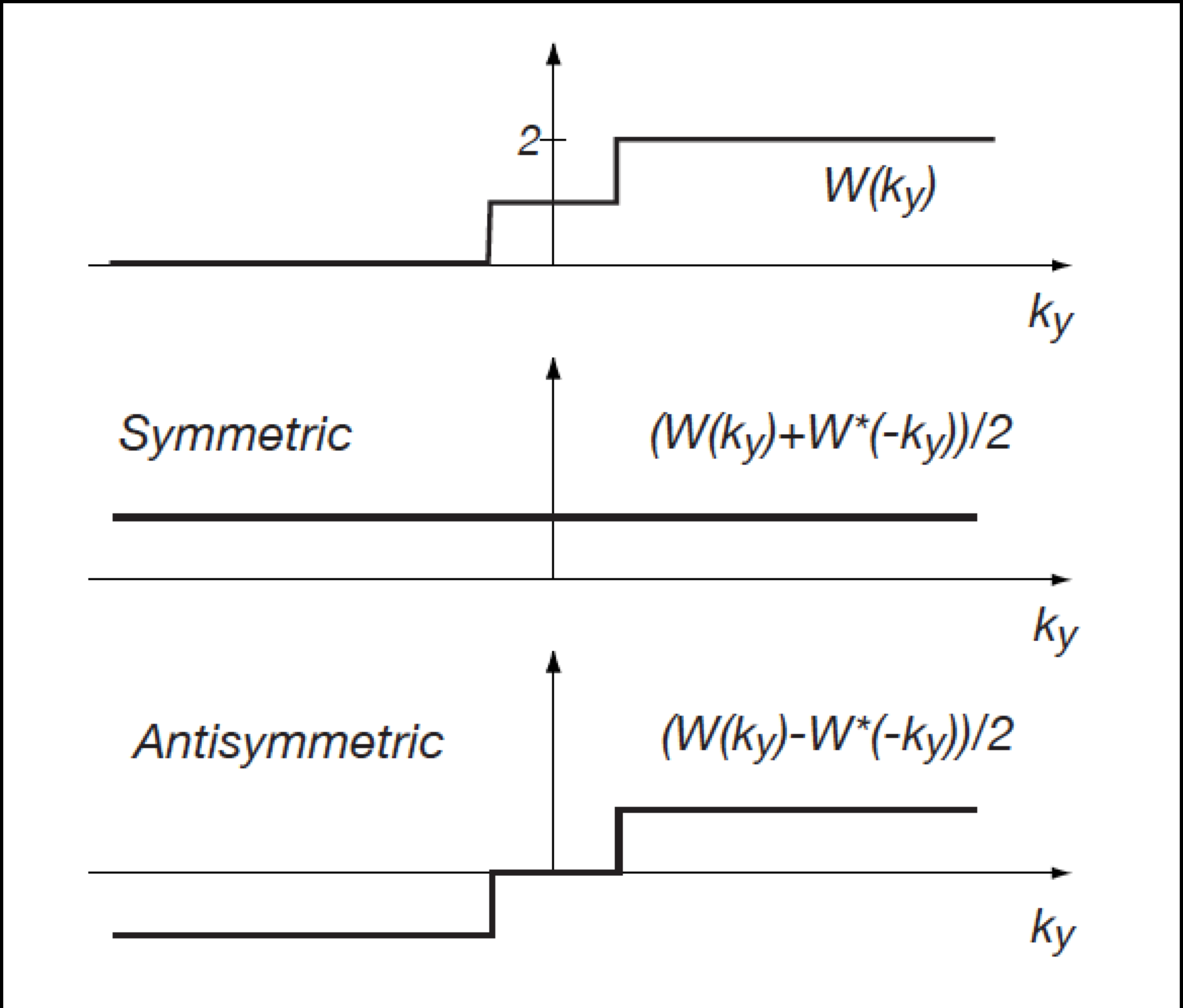
Homodyne Reconstruction

- *Real part* of an image corresponds to the conjugate symmetric component of the transform
- *Imaginary part* of an image corresponds to the conjugate anti-symmetric component of the transform

Symmetric and Antisymmetric Components



Symmetric and Antisymmetric Components

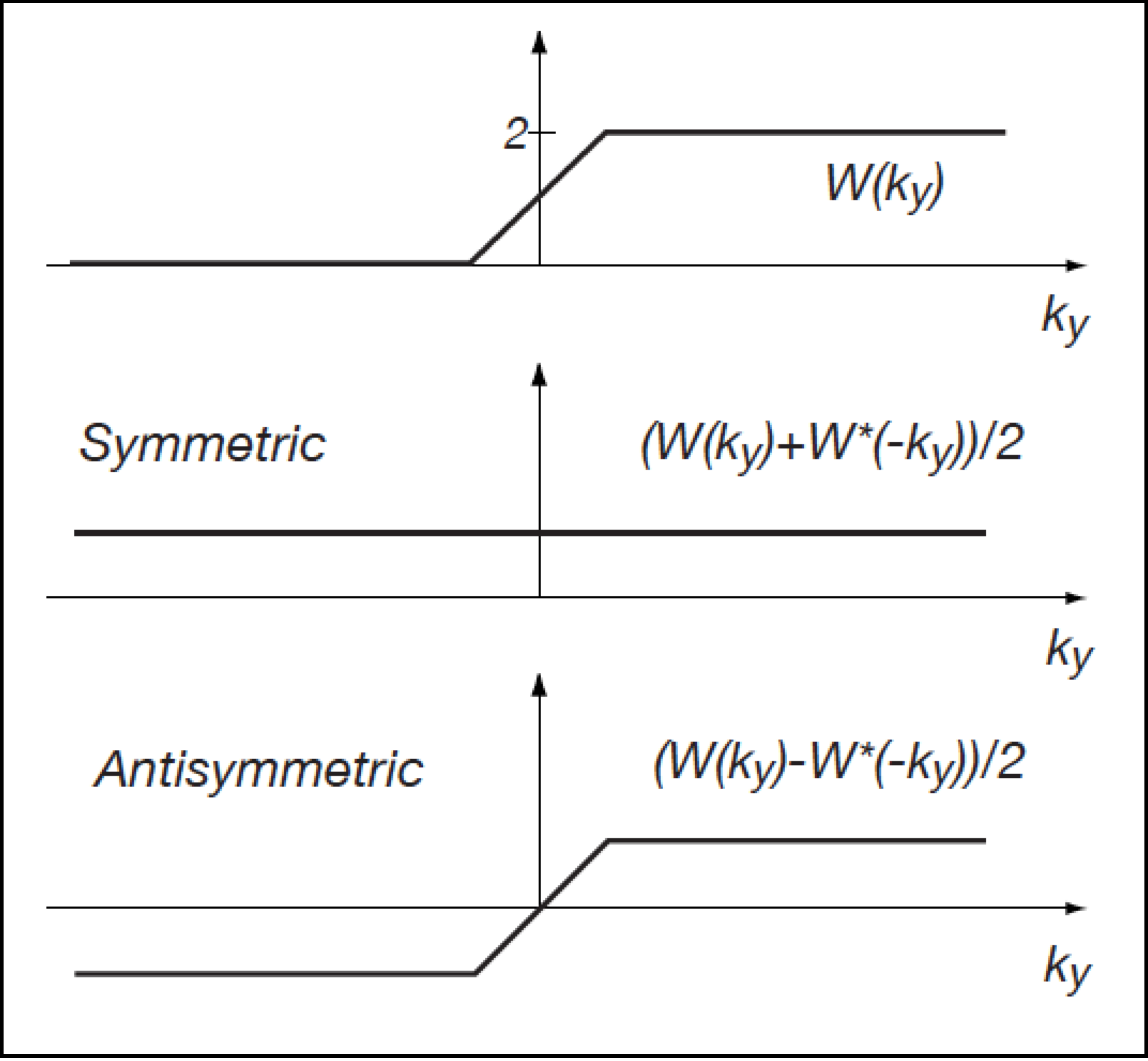


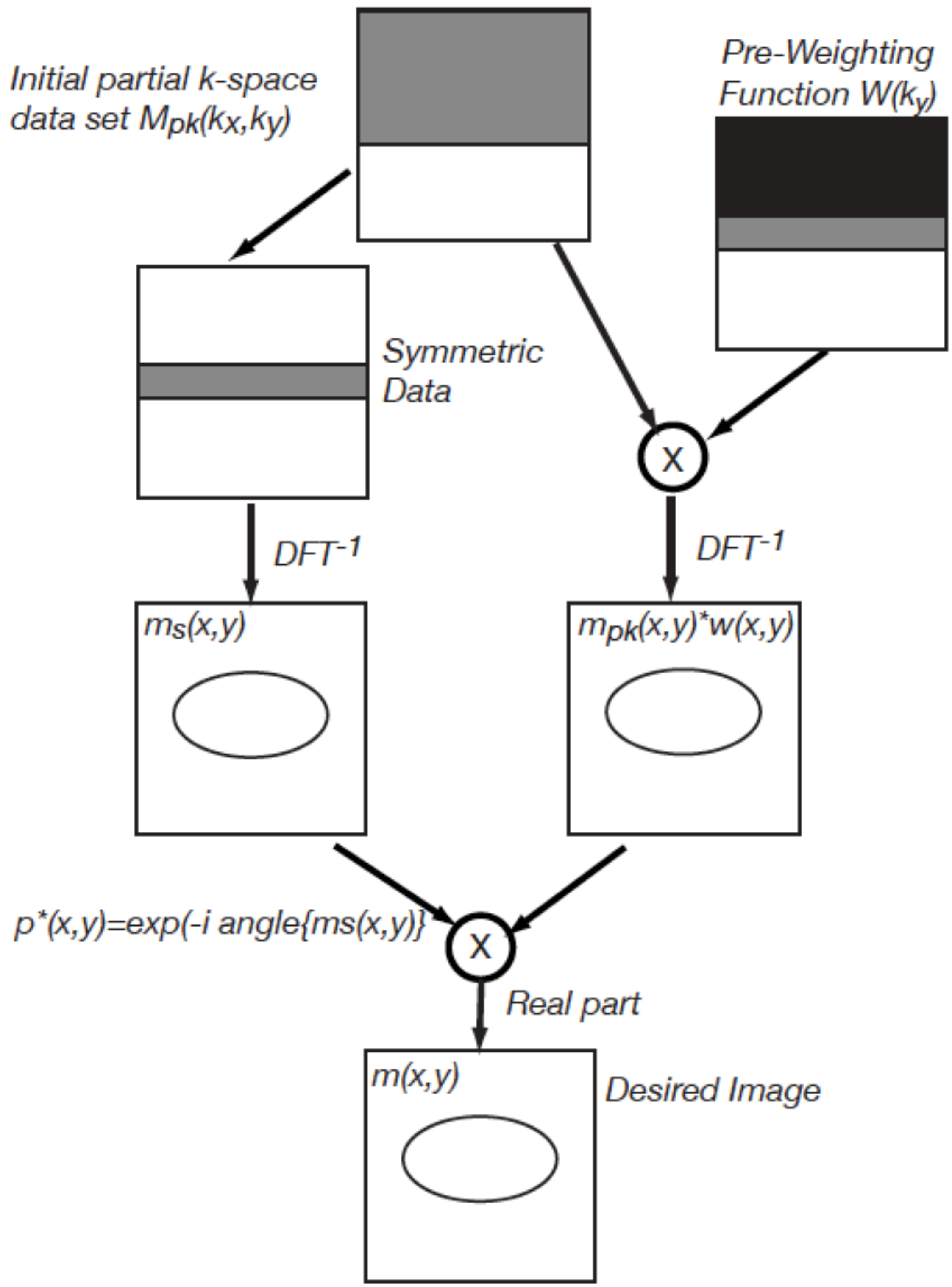
Weighting Function

$$m(x,y) = Re\{p^*(x,y)(m(x,y) * w(x,y))\}$$

- The phase correction in image space corresponds to a convolution in k-space
- The weighting sharp discontinuities of the weighting function can produce image artifacts

Preferred Weighting Function





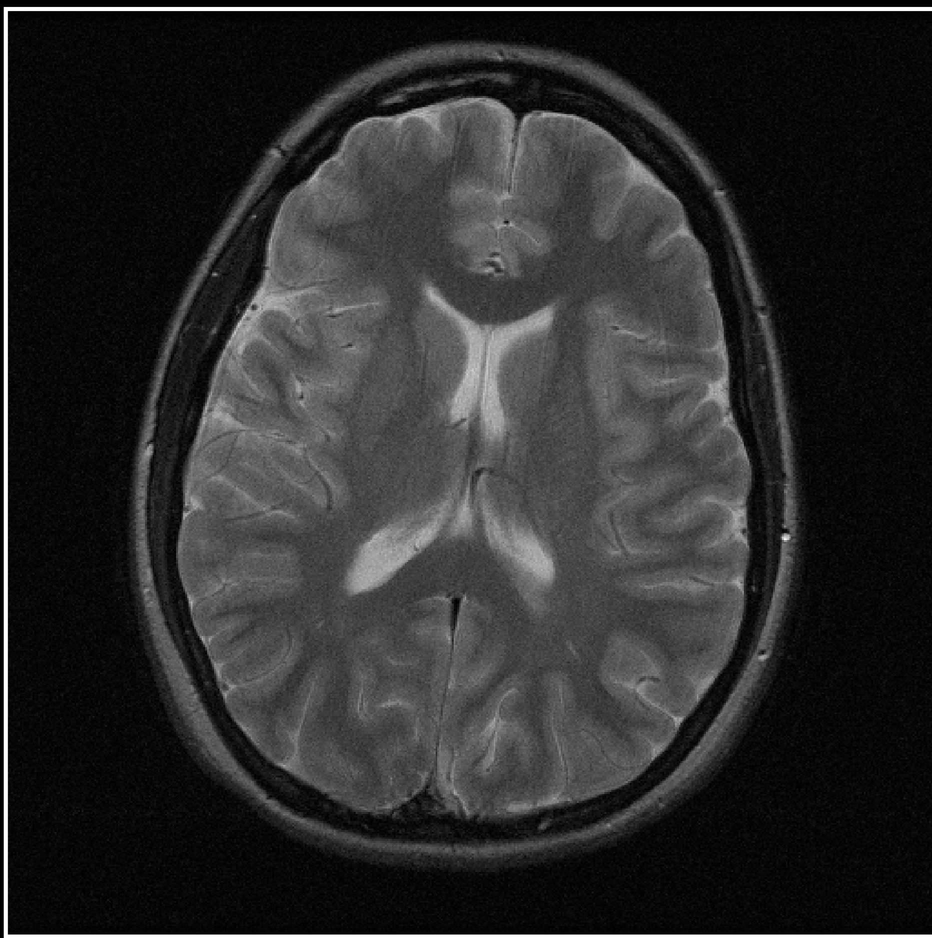
Initial partial k-space data set $M_{pk}(k_x,k_y)$
Pre-Weighting Function $W(k_y)$
Symmetric Data
X
DFT^{-1}
DFT^{-1}
$m_s(x,y)$
$m_{pk}(x,y)*w(x,y)$
$p*(x,y)=exp(-i\ angle\{ms(x,y)\}$
X
Real part
$m(x,y)$
Desired Image

MATLAB Code

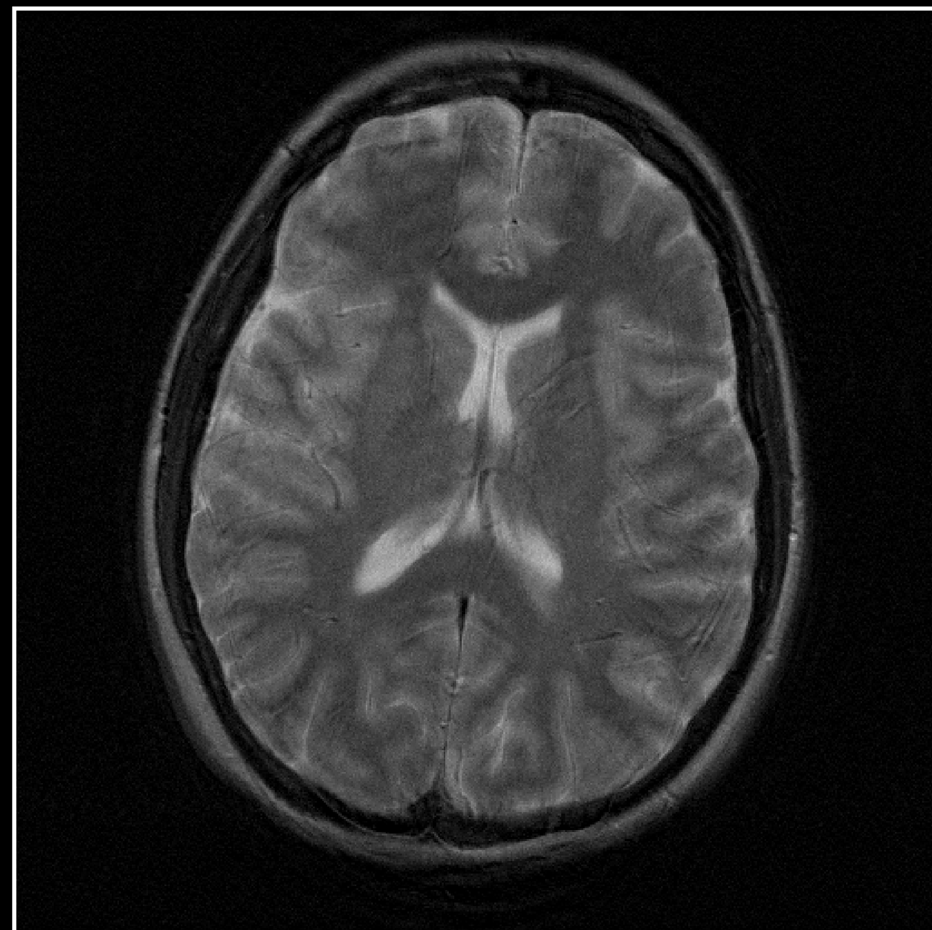
```
% Generate pre-weighting function W(ky)
W1d = zeros(nx,1);
W1d(1:hnover) = 2;
W1d(hnover+1:nx-hnover) = 2*(nx-2*hnover-1:-1:0)/(nx-2*hnover);
Wky = repmat(W1d,[1 nx]);

data_pw = data_pk.*Wky;
im_pw = fftshift(ifftn(fftshift(data_pw)));

im_homodyne = im_pw.*exp(-1i*angle(im_ph));
```

Homodyne Reconstruction

Original

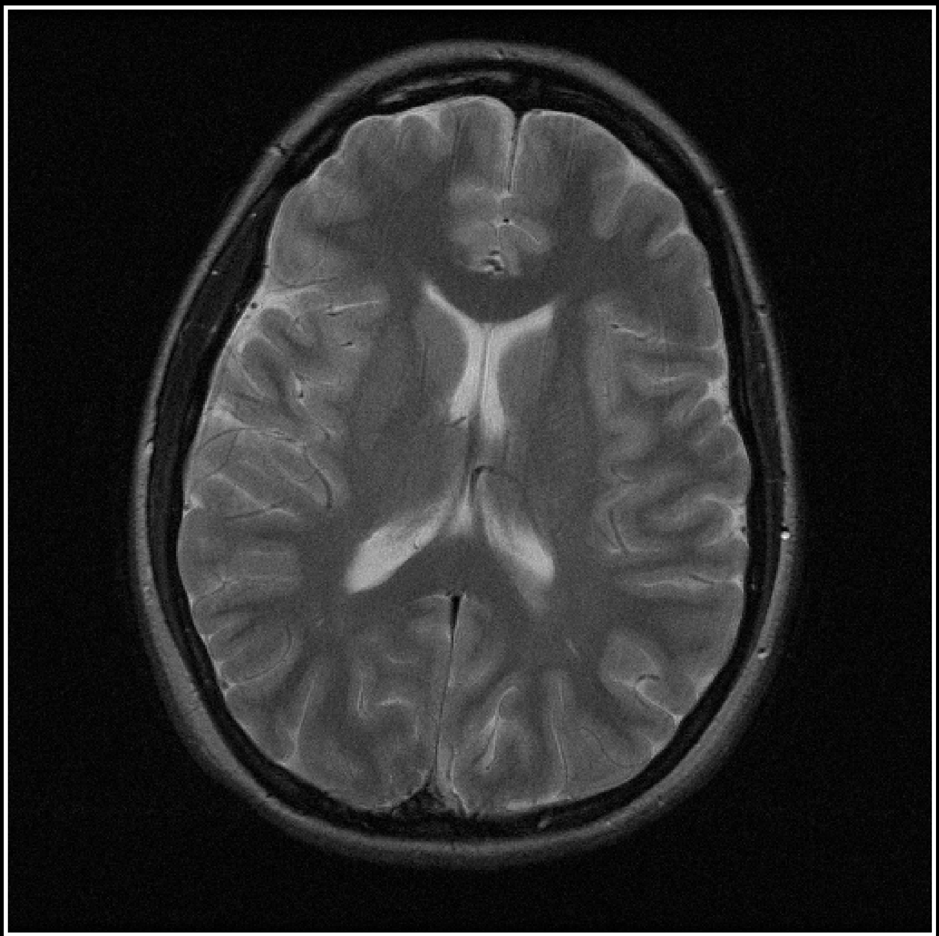


Phase Correction

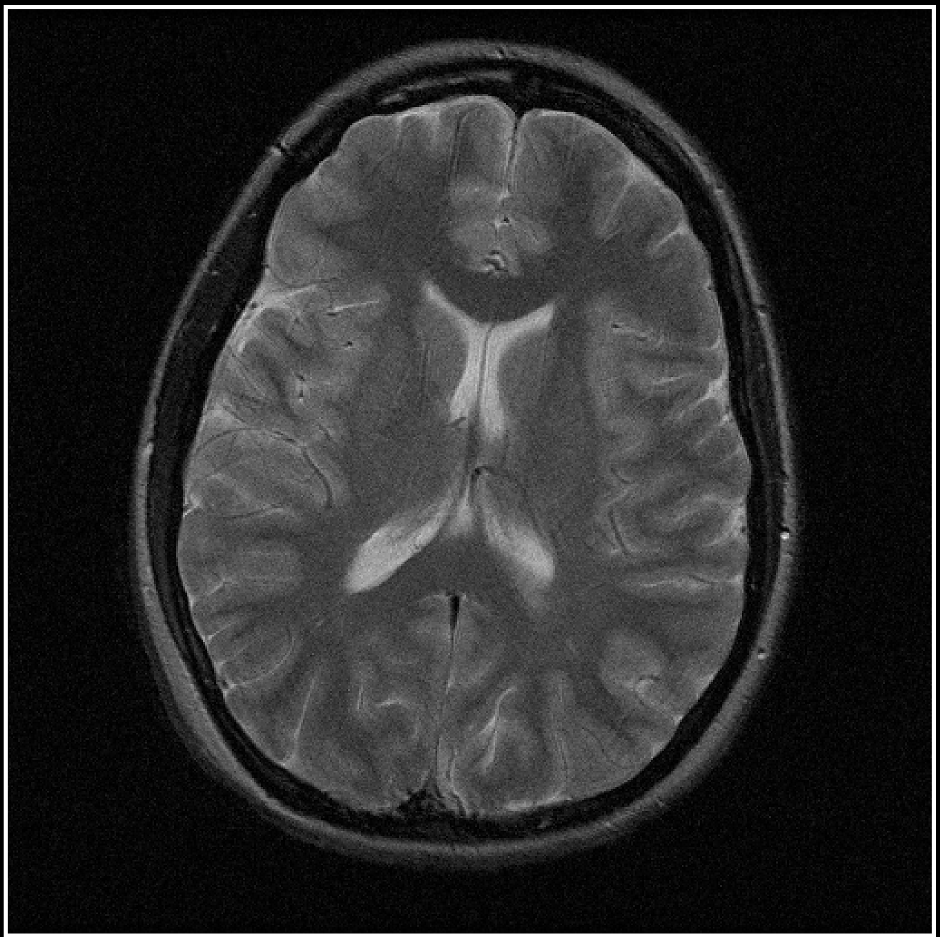


Homodyne Reconstruction

Original



Homodyne Recon



Summary of Direct Methods

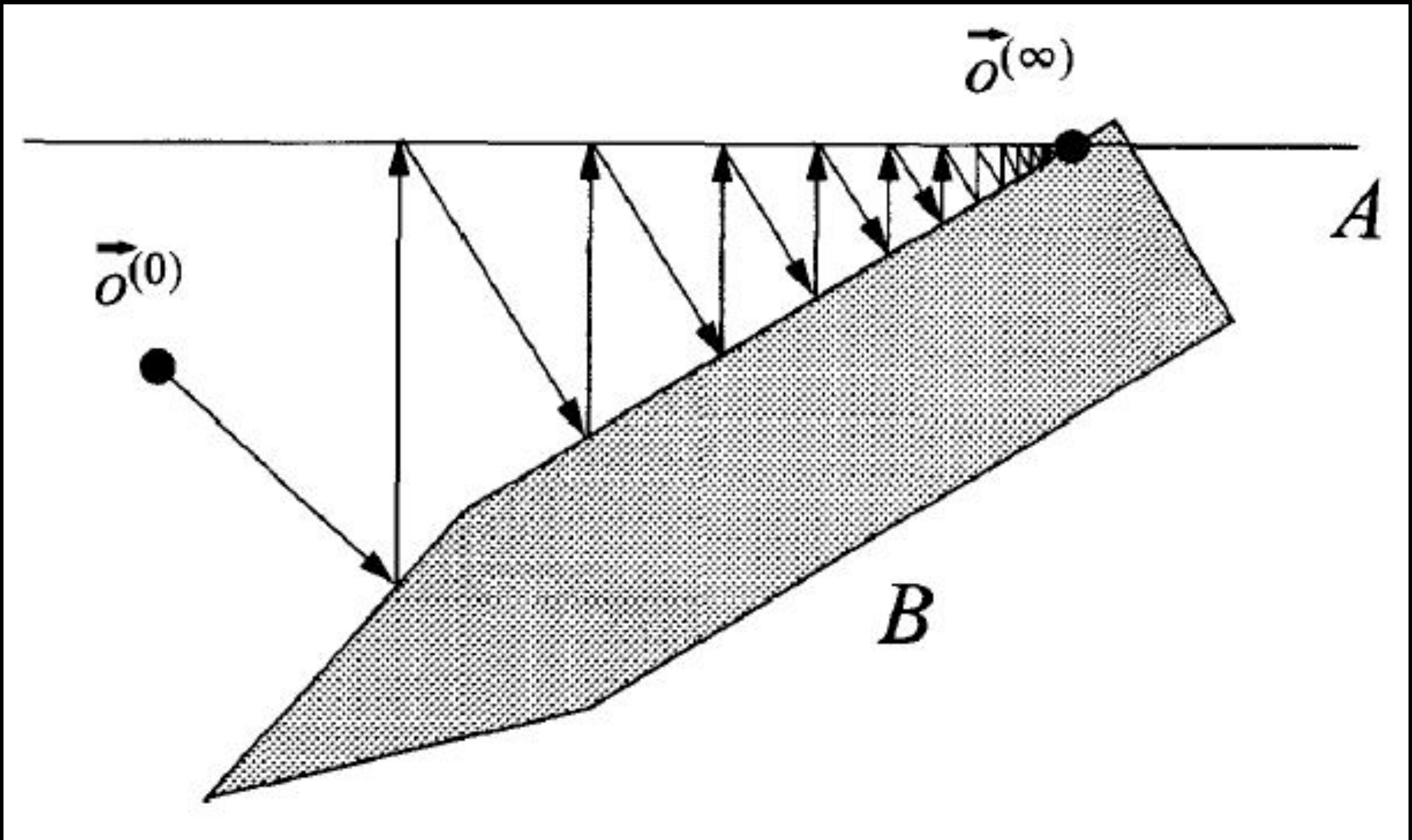
- Both homodyne and phase corrected conjugate synthesis approaches work well if image phase does not vary rapidly
- Problems with phase corrected conjugate synthesis approach are due to performing the conjugate synthesis after the phase correction

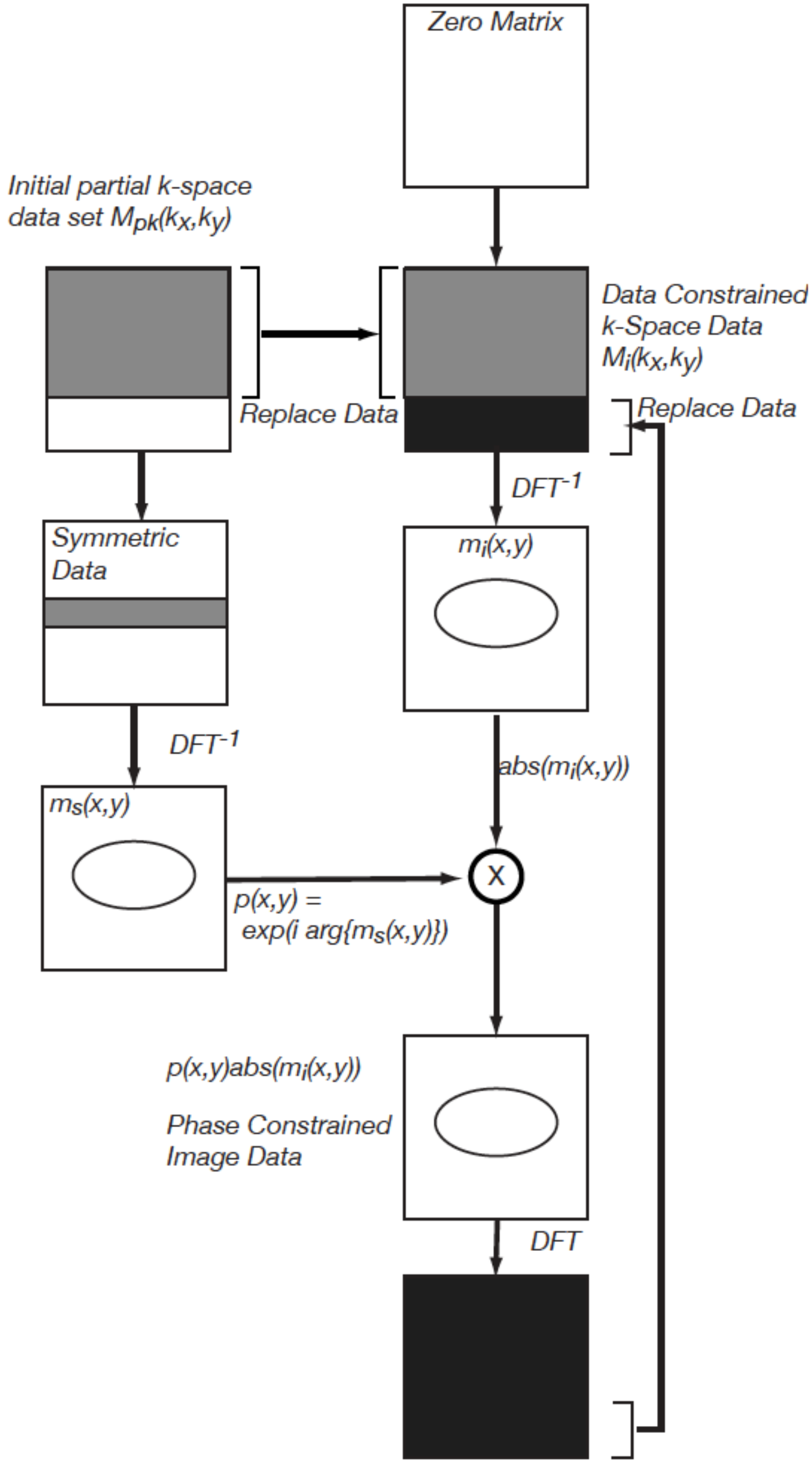
Iterative Reconstruction

$$m_i(x, y) = |m_i(x, y)|p(x, y)$$

- Estimate the missing k-space data by iteratively applying phase correction and conjugate synthesis
- In the image domain, the image phase is constrained to be that of the low resolution estimate
- In the frequency domain, the k-space data is constrained to match the acquired data when available

Projection Onto Convex Set (POCS)



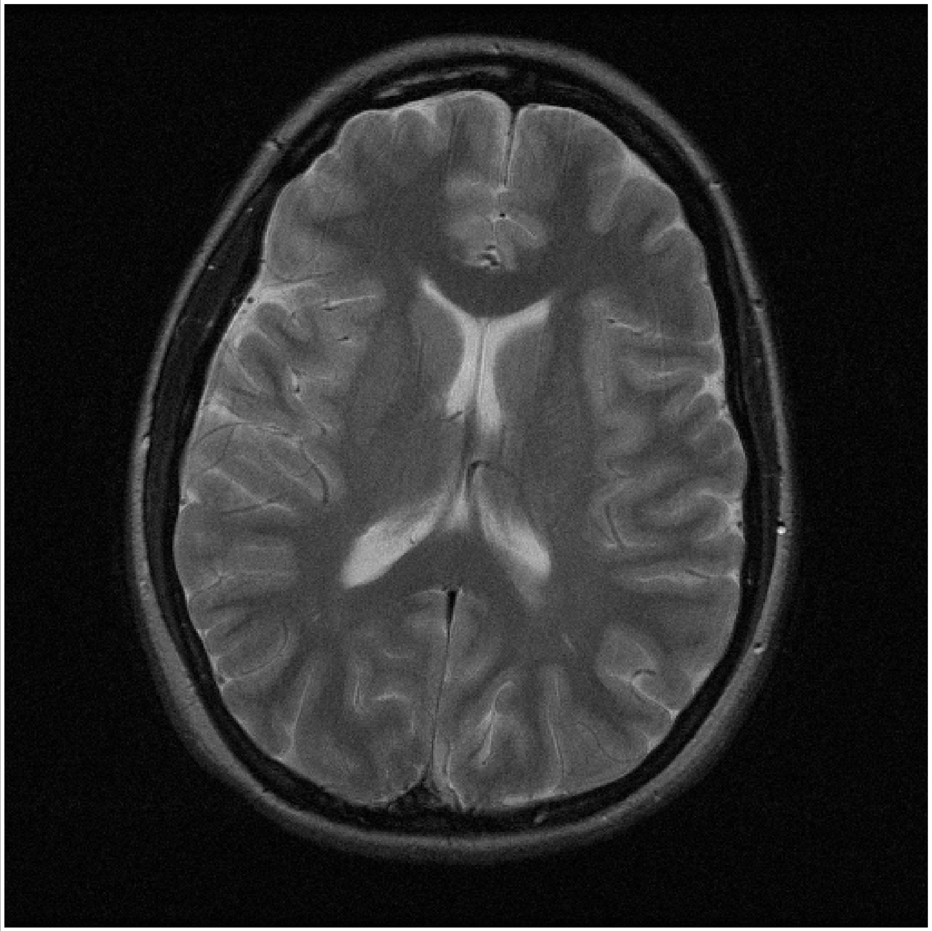

Zero Matrix
Initial partial k-space
data set $M_{pk}(k_x,k_y)$
Data Constrained
k-Space Data
$M_i(k_x,k_y)$
Replace Data
Replace Data
DFT^{-1}
Symmetric
Data
$m_i(x,y)$
DFT^{-1}
$abs(m_i(x,y))$
$m_s(x,y)$
X
$p(x,y) =$
$exp(i\ arg\{m_s(x,y)\})$
$p(x,y)abs(m_i(x,y))$
Phase Constrained
Image Data
DFT

MATLAB Code

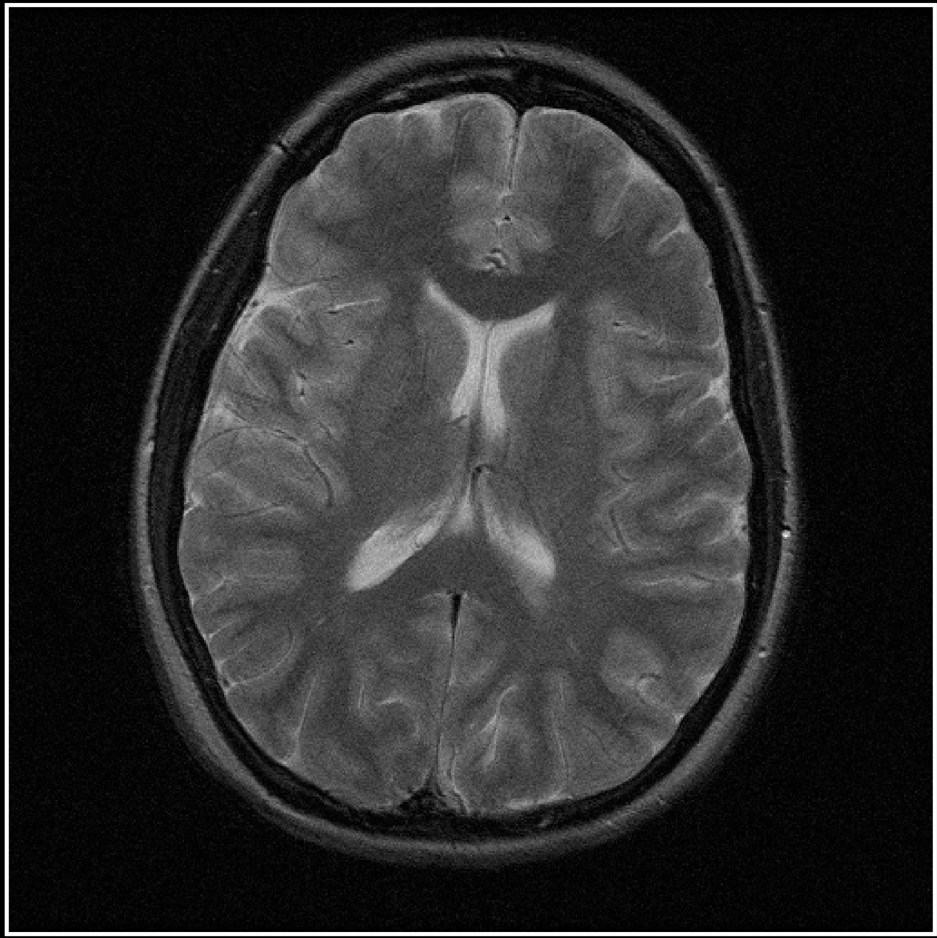
```
threshold_pocs = 0.001;

% Zero padding for initial guess
im_init = fftshift(ifftn(fftshift(data_pk)));   % Inverse DFT
% Take only magnitude term & Apply phase term
im_init = abs(im_init).*exp(1i*angle(im_ph));

% FFT
tmp_k   = fftshift(fftn(fftshift(im_init)));
diff_im = threshold_pocs + 1;

while (abs(diff_im) > threshold_pocs)
    tmp_k(1:nx-hnover,:) = data_pk(1:nx-hnover,:);
    tmp_im = fftshift(ifftn(fftshift(tmp_k)));     % Inverse DFT

    % Take only magnitude term & Apply phase term
    tmp_im = abs(tmp_im).*exp(1i*angle(im_ph));
    tmp_k = fftshift(fftn(fftshift(tmp_im)));

    % Compare the reconstructed image
    diff_im = abs(tmp_im - im_init);
    diff_im = sum(diff_im(:).^2);
    fprintf('Difference is %f\n',diff_im);

    im_init = tmp_im;
end

im_pocs = tmp_im;
```

POCS Reconstruction

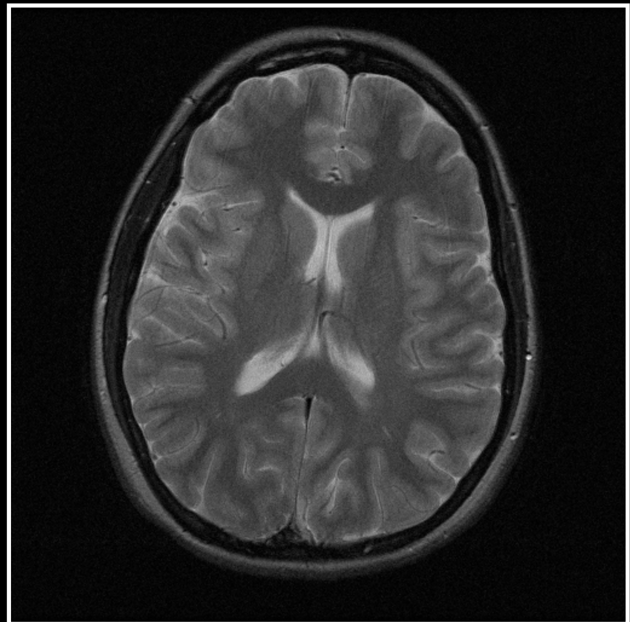
Original



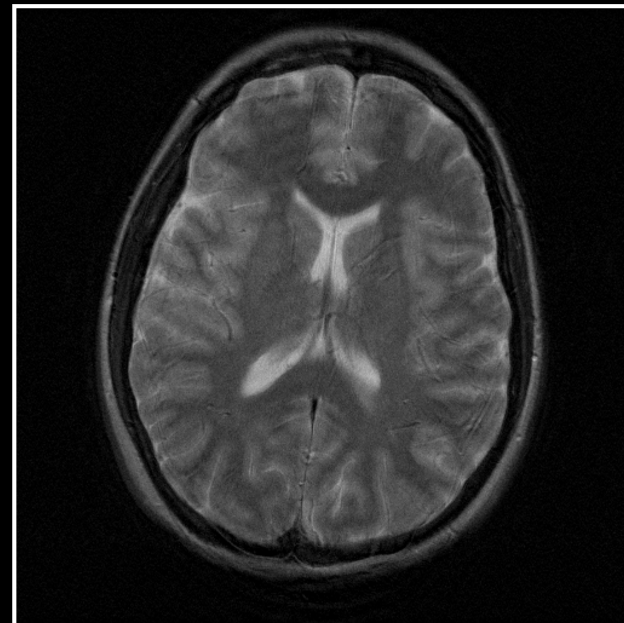
POCS Recon



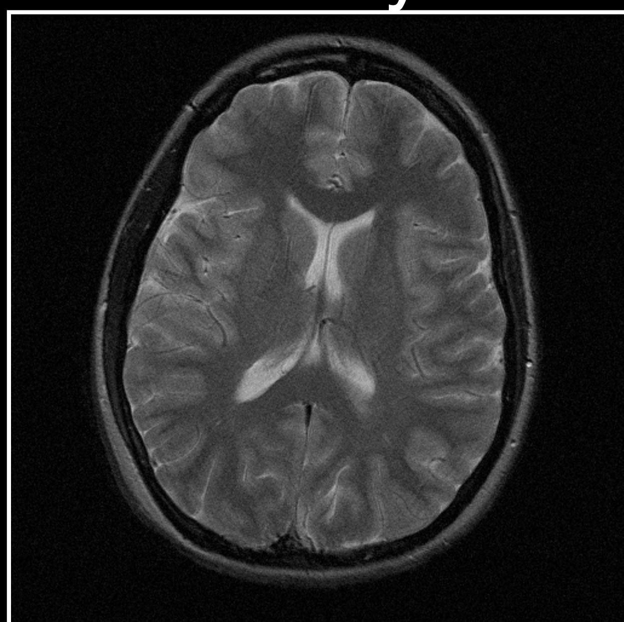
Original



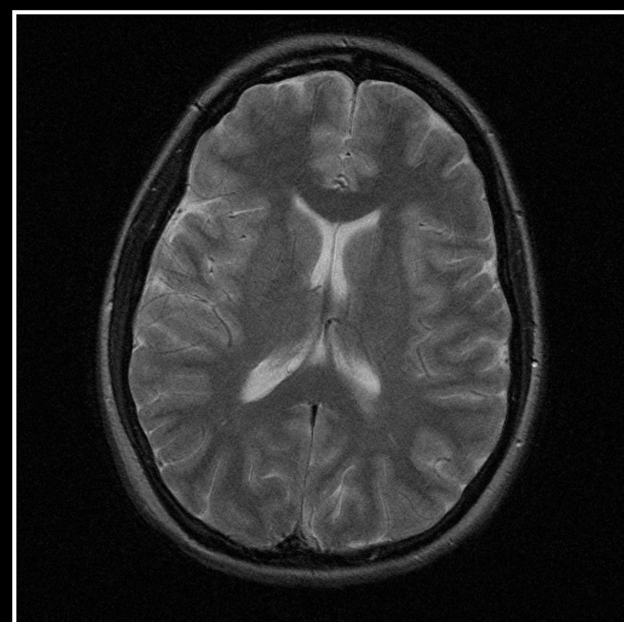
Phase Correction



Homodyne



POCS



Conclusions

- All of these algorithms work well when the image phase variations are smooth
- When the image phase changes rapidly, the homodyne algorithm produces ghosting
- POCS algorithm performs somewhat better as the k-space fraction decreases

Thanks!

Kyung Sung, PhD

ksung@mednet.ucla.edu

https://mrrl.ucla.edu/sunglab/