Nuts & Bolts of Advanced Imaging

Image Reconstruction – Parallel Imaging

Michael S. Hansen, PhD

Magnetic Resonance Technology Program
National Institutes of Health, NHLBI

NIH National Heart, Lung, and Blood Institute

Declaration of Financial Interests or Relationships

Speaker Name: Michael S. Hansen

I have no financial interests or relationships to disclose with regard to the subject matter of this presentation.

Outline

- Noise correlation
- SNR scaled reconstruction
  - Obtaining images in SNR units
- Pseudo Replica Method
  - Determining the SNR (and g-map) for any parallel imaging reconstruction
- Iterative methods
  - Non-Cartesian Parallel Imaging
- Regularization in Iterative Methods

Noise in Parallel Imaging

Idealized Experiment:

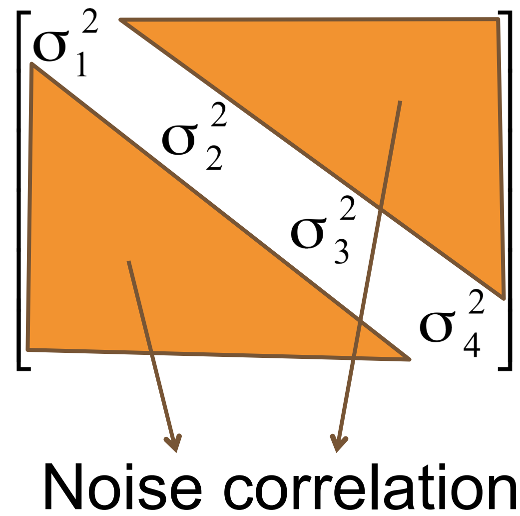
$$\mathbf{s} = \mathbf{E}\boldsymbol{\rho}$$

Noise covariance matrix

$$\Psi_{\Upsilon,\Upsilon'} = \langle \eta_{\Upsilon}, \eta_{\Upsilon'} \rangle$$

In practice, we are affected by noise

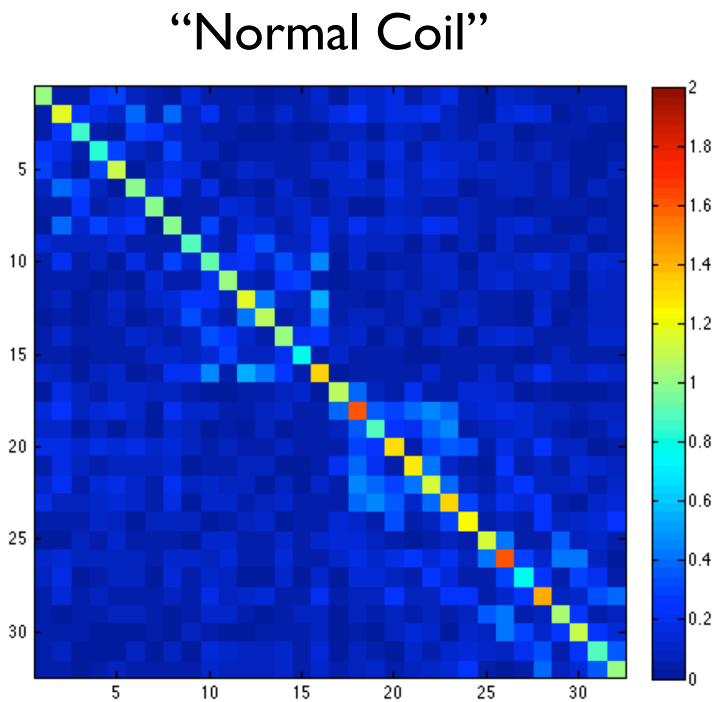
$$\mathbf{s} = \mathbf{E}\boldsymbol{\rho} + \boldsymbol{\eta}$$

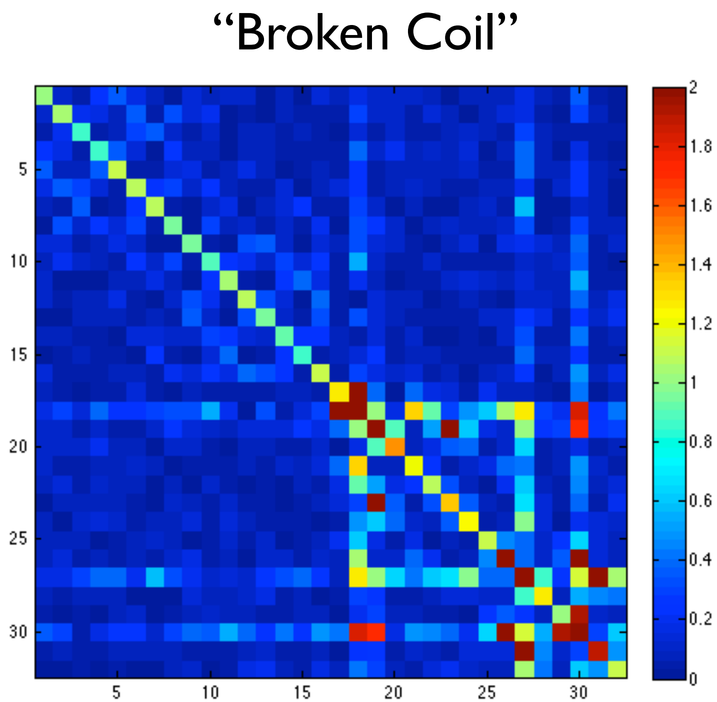


We can measure this noise covariance:

```
% Matlab
% eta:[Ncoils, Nsamples]
Psi = (1/(Nsamples-1))*(eta * eta');
```

Psi Examples – 32 Channel Coil





Examination of the noise covariance matrix is an important QA tool. Reveals broken elements, faulty pre-amps, etc.

Noise Pre-Whitening

Solving Linear Equations:

$$\mathbf{Ax} + \boldsymbol{\eta} = \mathbf{b} = \begin{bmatrix} c_1 & c_2 \\ c_3 & c_4 \\ c_5 & c_6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$X_i \ : \ \text{Random value with zero mean } (\mu = 0) \text{ and variance } \sigma_i^2$$

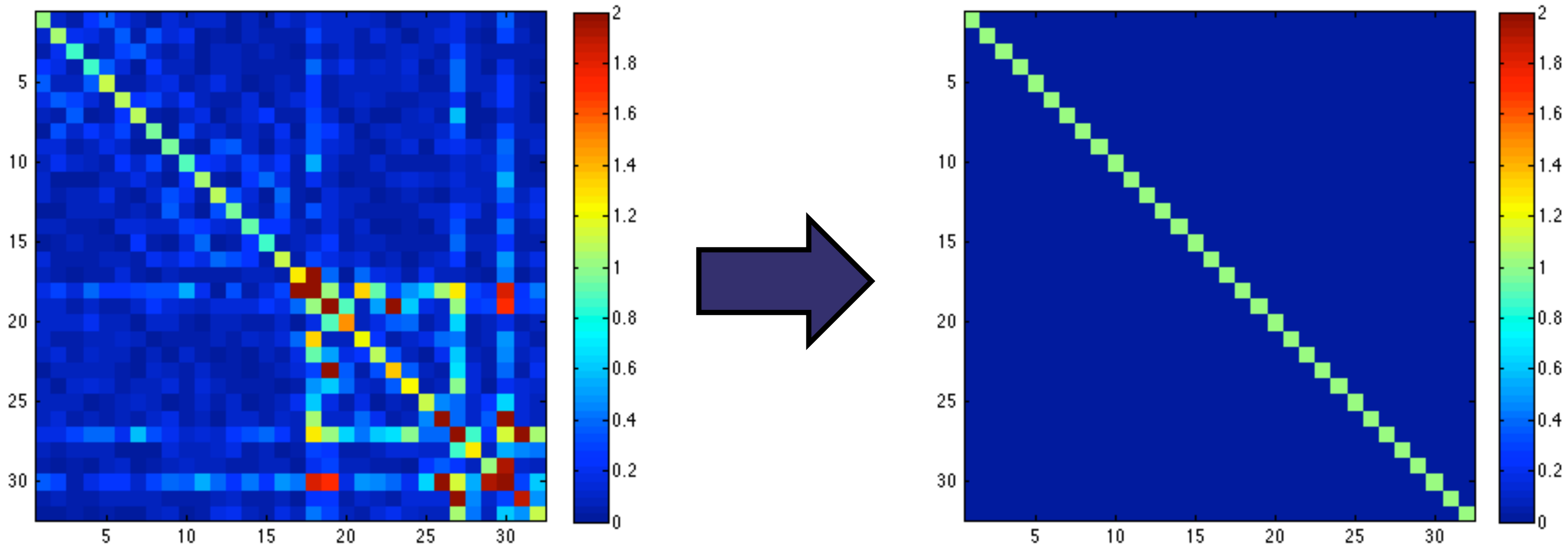
Suppose you know that:

$$\sigma_3^2 = 5\sigma_1^2 = 5\sigma_2^2$$

Put less weight on this equation

Noise Pre-Whitening

We would like to apply an operation such that we have unit variance in all channels:



Noise Pre-Whitening

More generally, we want to weight the equations with the "inverse square root" of the noise covariance, if

$$\mathbf{\Psi} = \mathbf{L}\mathbf{L}^{\mathrm{H}}$$

We will solve:

$$\mathbf{L}^{-1}\mathbf{A}\mathbf{x} = \mathbf{L}^{-1}\mathbf{b}$$

Or:

$$\mathbf{x} = \left(\mathbf{A}^{\mathrm{H}}\mathbf{\Psi}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{H}}\mathbf{\Psi}^{-1}\mathbf{b}$$

In practice, we simply generate "pre-whitened" input data before recon

Noise Pre-Whitening

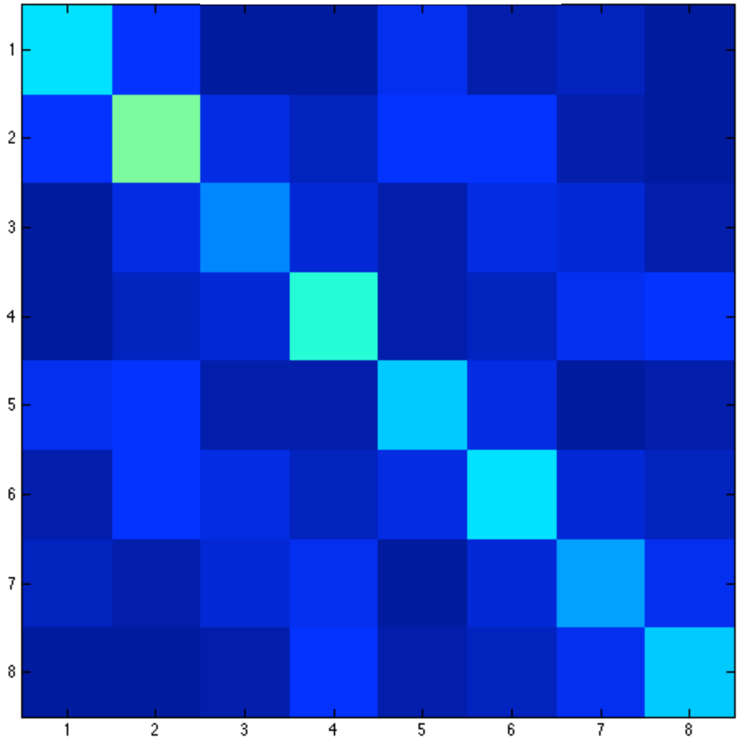
Matlab:

```
%eta [Ncoils,Nsamples]
%psi [Ncoils,Ncoils]
%data [Ncoils,Nsamples]
%csm : Coil sensitivity map

psi = (1/(Nsamples-1))*(eta * eta');

L = chol(psi,'lower');
L_inv = inv(L);

data = L_inv * data;
csm = L_inv * csm;

%Now noise is “white”
%Reshape data and do recon
```

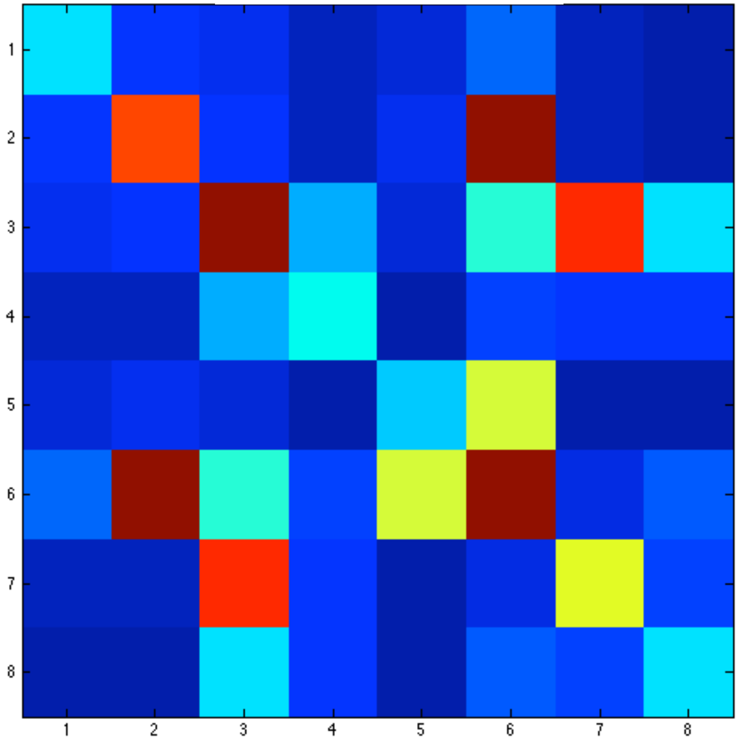
Noise covariance matrix

Example with test dataset

“Normal Coil”

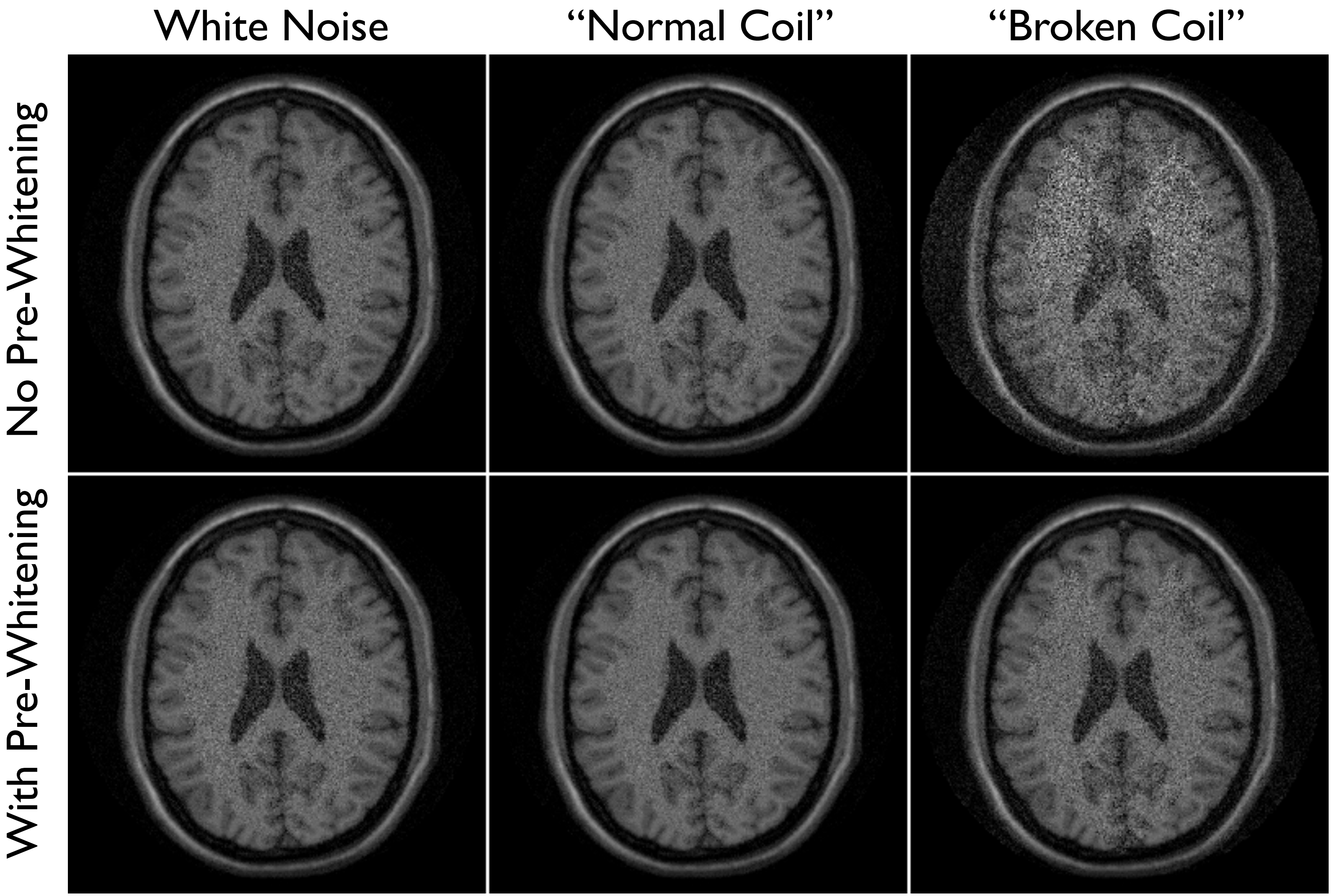


“Broken Coil”



At least two broken pre-amps

Noise Pre-Whitening – SENSE Example

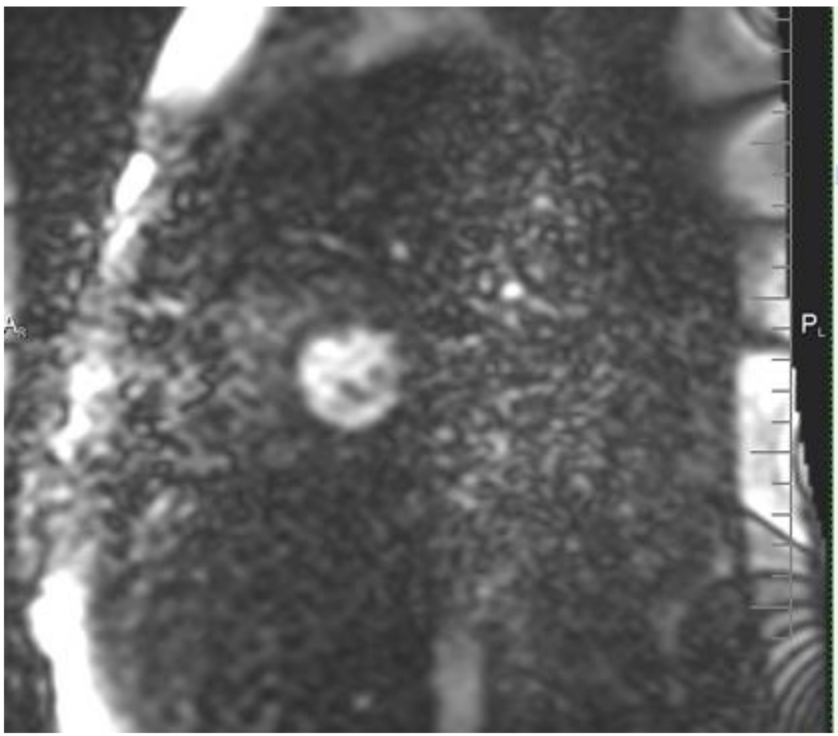


`ismrm_demo_noise_decorrelation.m`

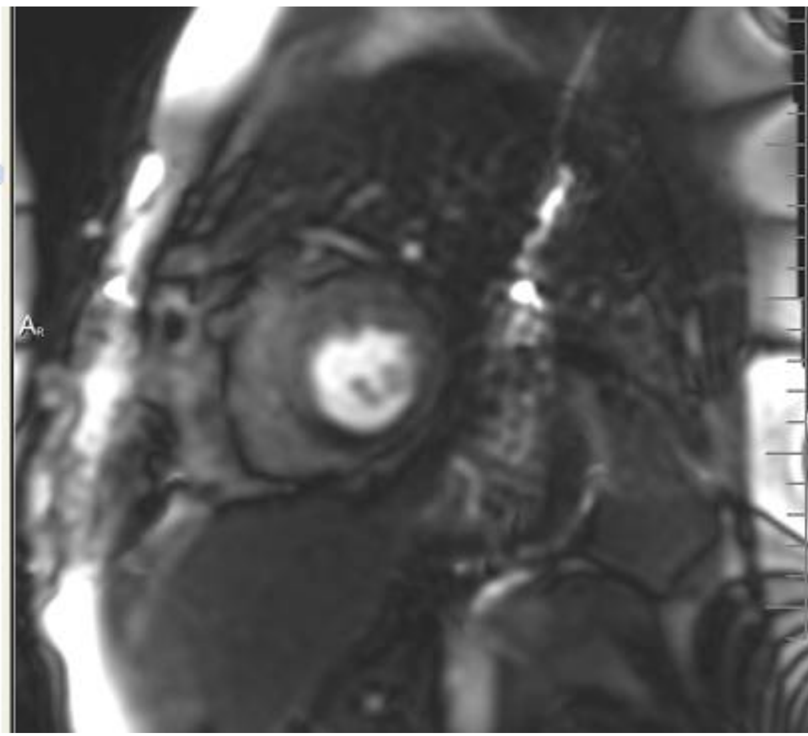
Noise Pre-Whitening – In vivo example

In vivo stress perfusion case where broken coil element resulted in non-diagnostic images.

Without pre-whitening

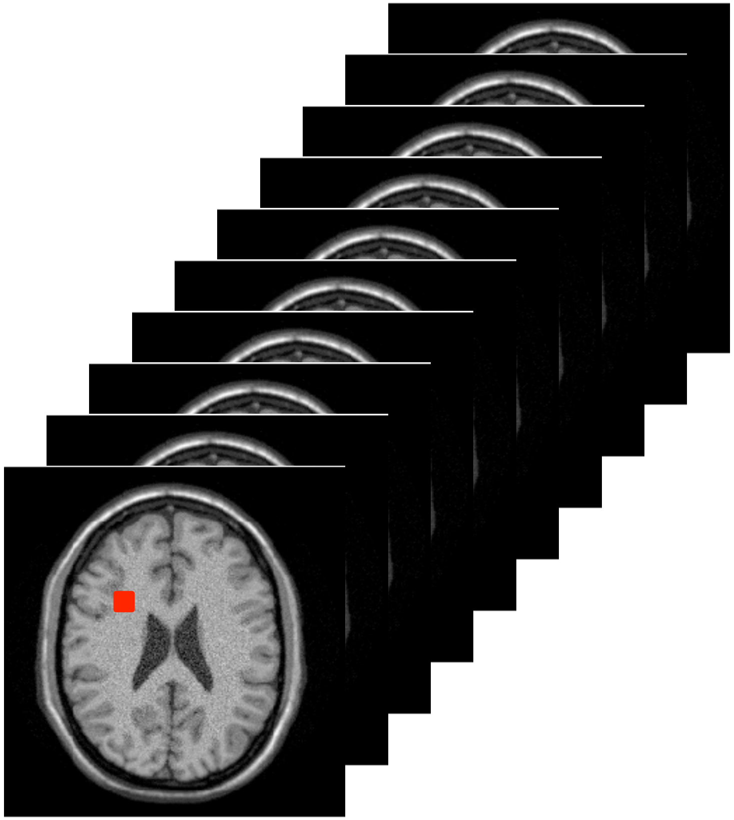


With pre-whitening



Example provided by Peter Kellman, NIH

Signal to Noise Ratio (Definitions)



$$SNR(x,y) = \frac{S(x,y)}{\sigma(x,y)}$$

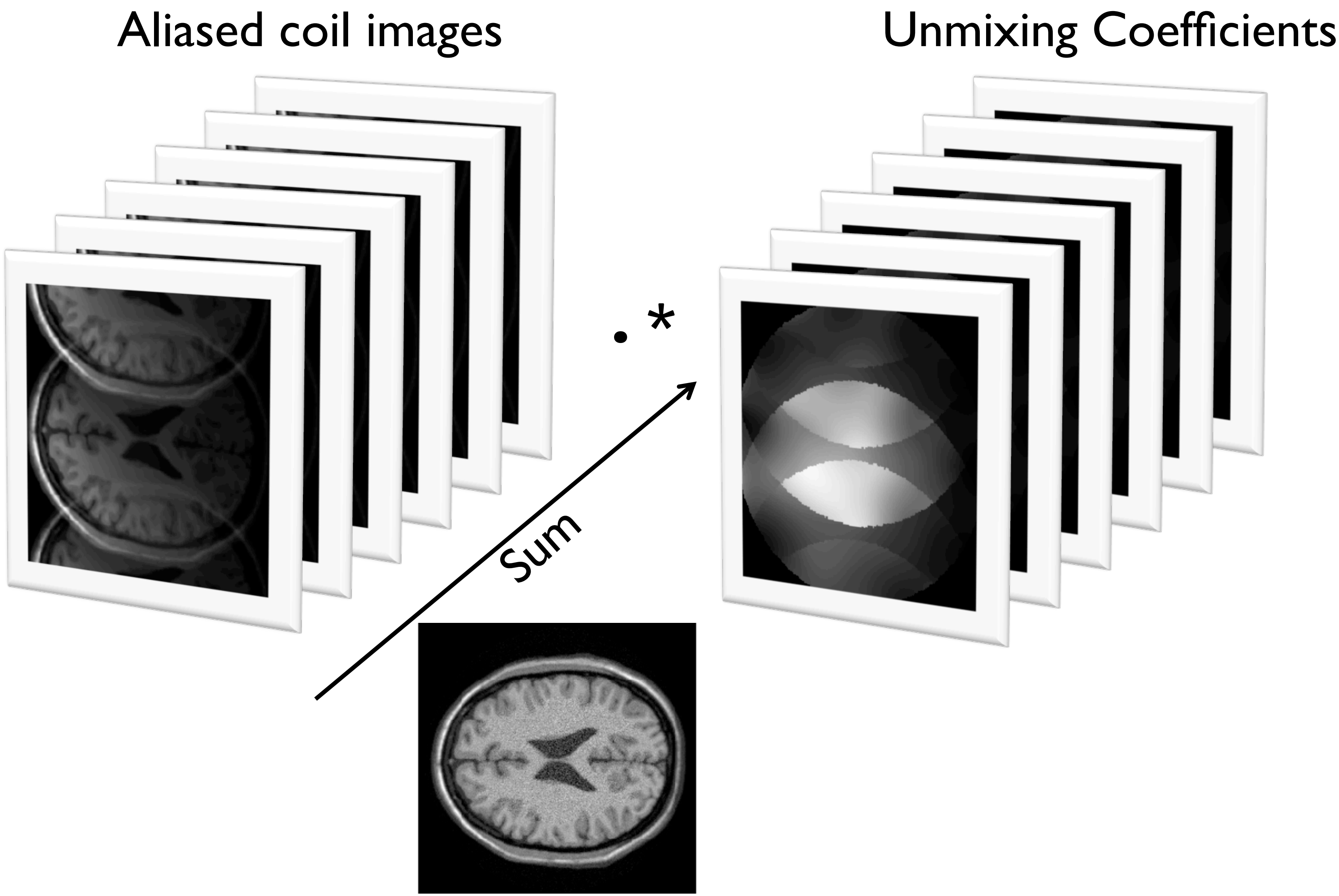
Intuitively, SNR is measured by repeating the experiment.

Signal level is the mean signal over multiple experiments.

Noise level is the standard deviation over multiple experiments

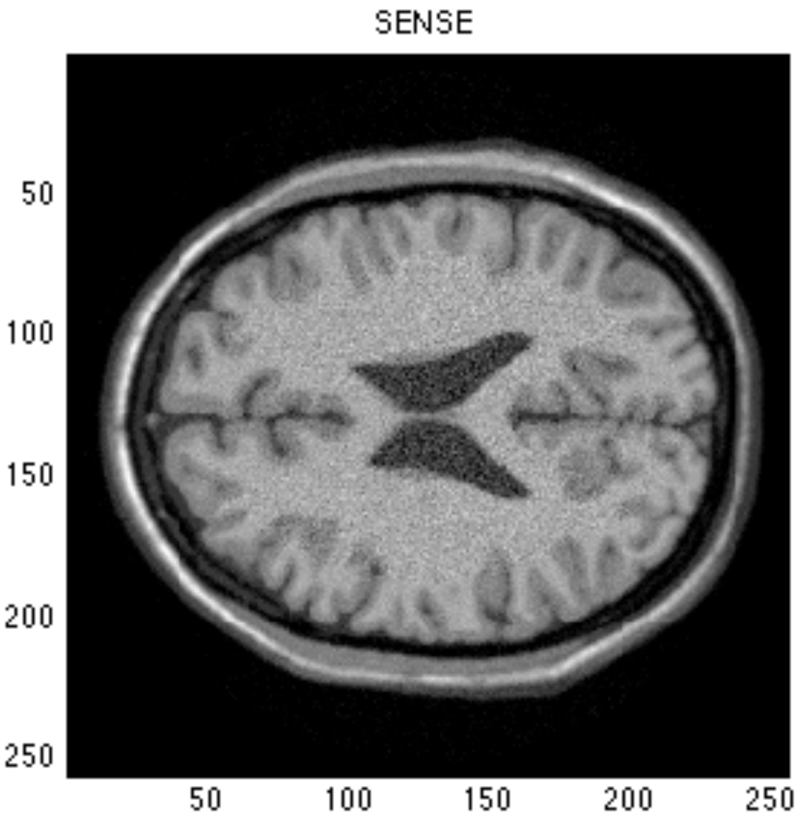
Such experiments are hard to perform in practice.

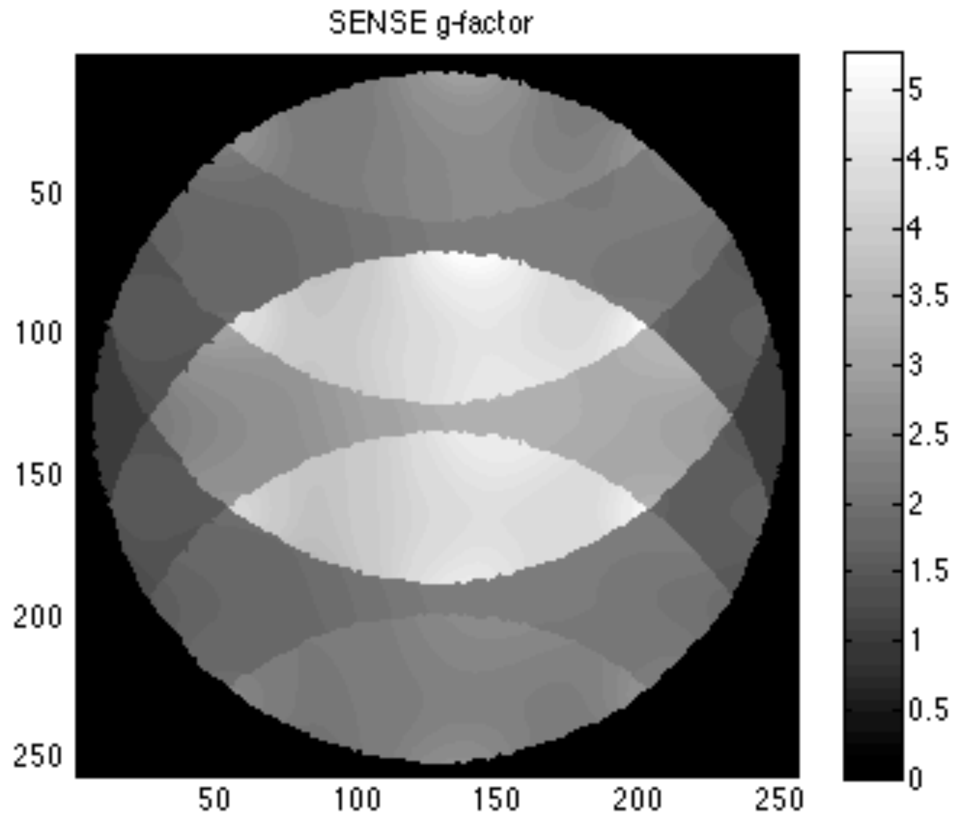
SENSE – Image Synthesis with Unmixing Coefficients



SENSE – Simple Rate 4 Example

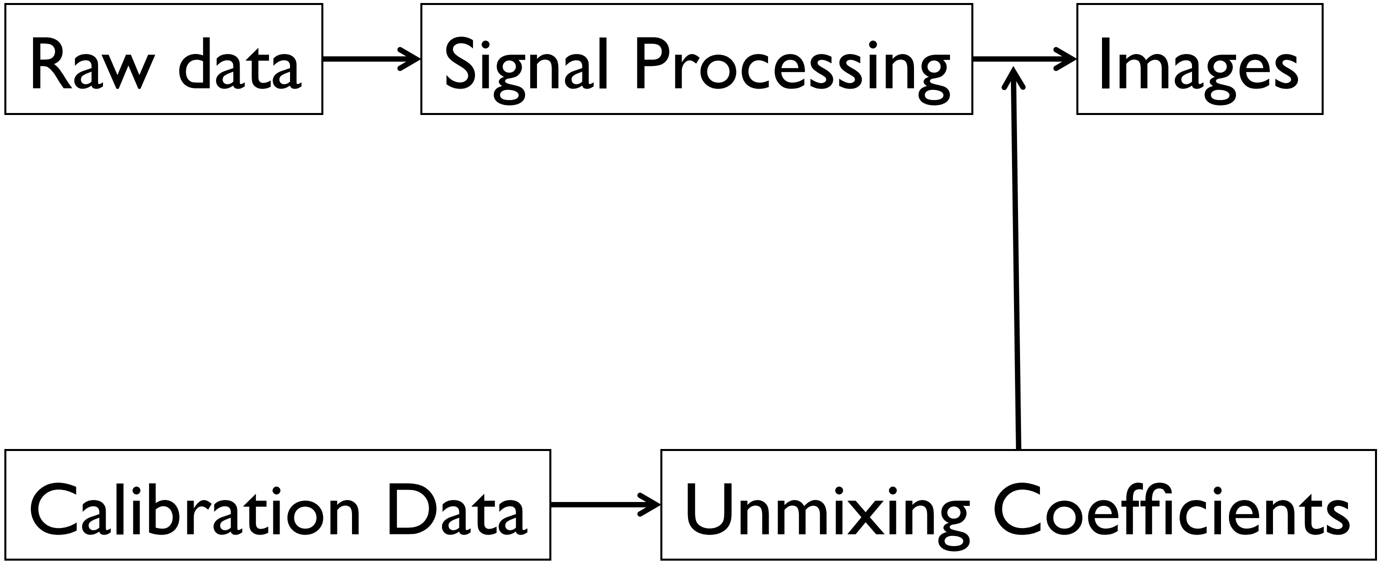
$$\tilde{\rho}(x_1) = \sum_{i=0}^{N_c} u_i a_i \qquad g(x_1) = \sqrt{\sum_{i=0}^{N_c} |u_i|^2} \sqrt{\sum_{i=0}^{N_c} |S_i|^2}$$





Reconstruction in SNR Units

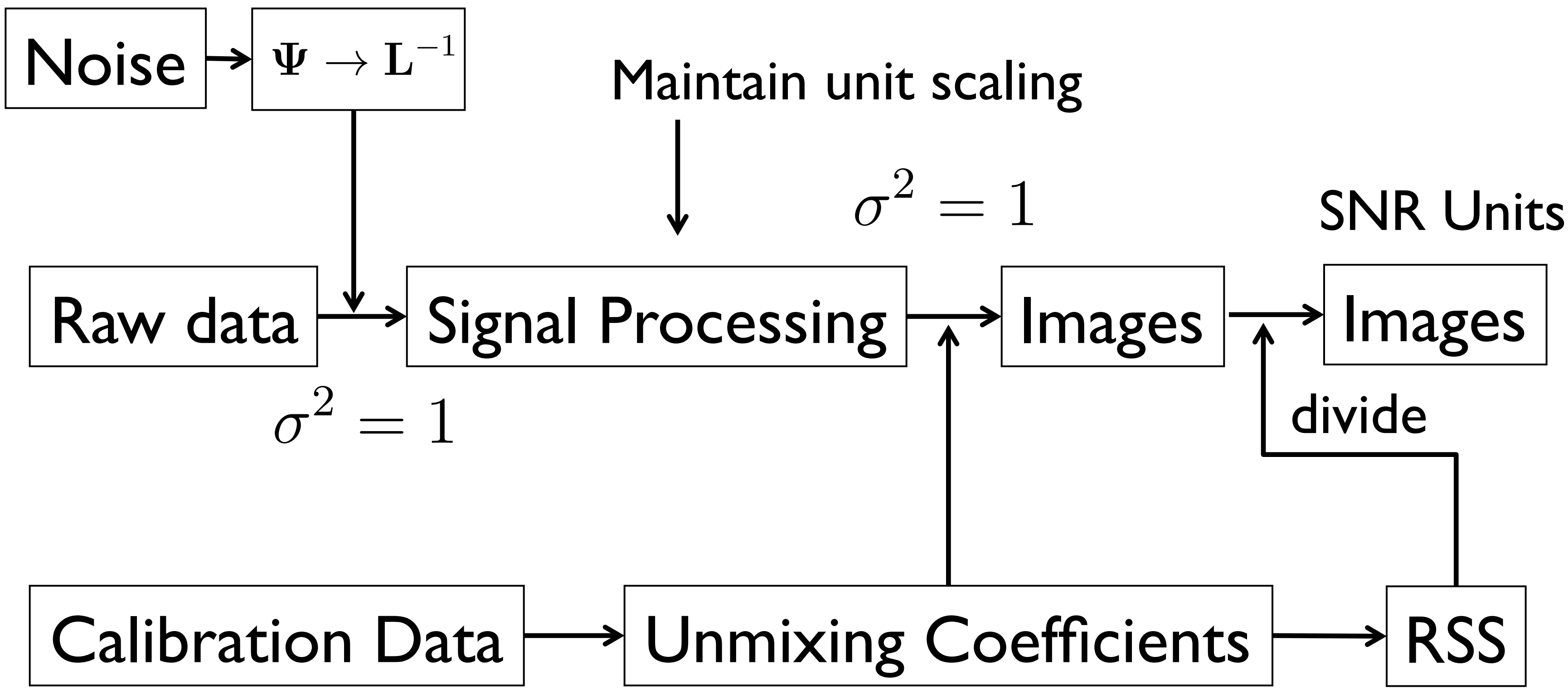
Reconstruction Pipeline



Kellman et al., Magnetic Resonance in Medicine 54:1439 –1447 (2005)

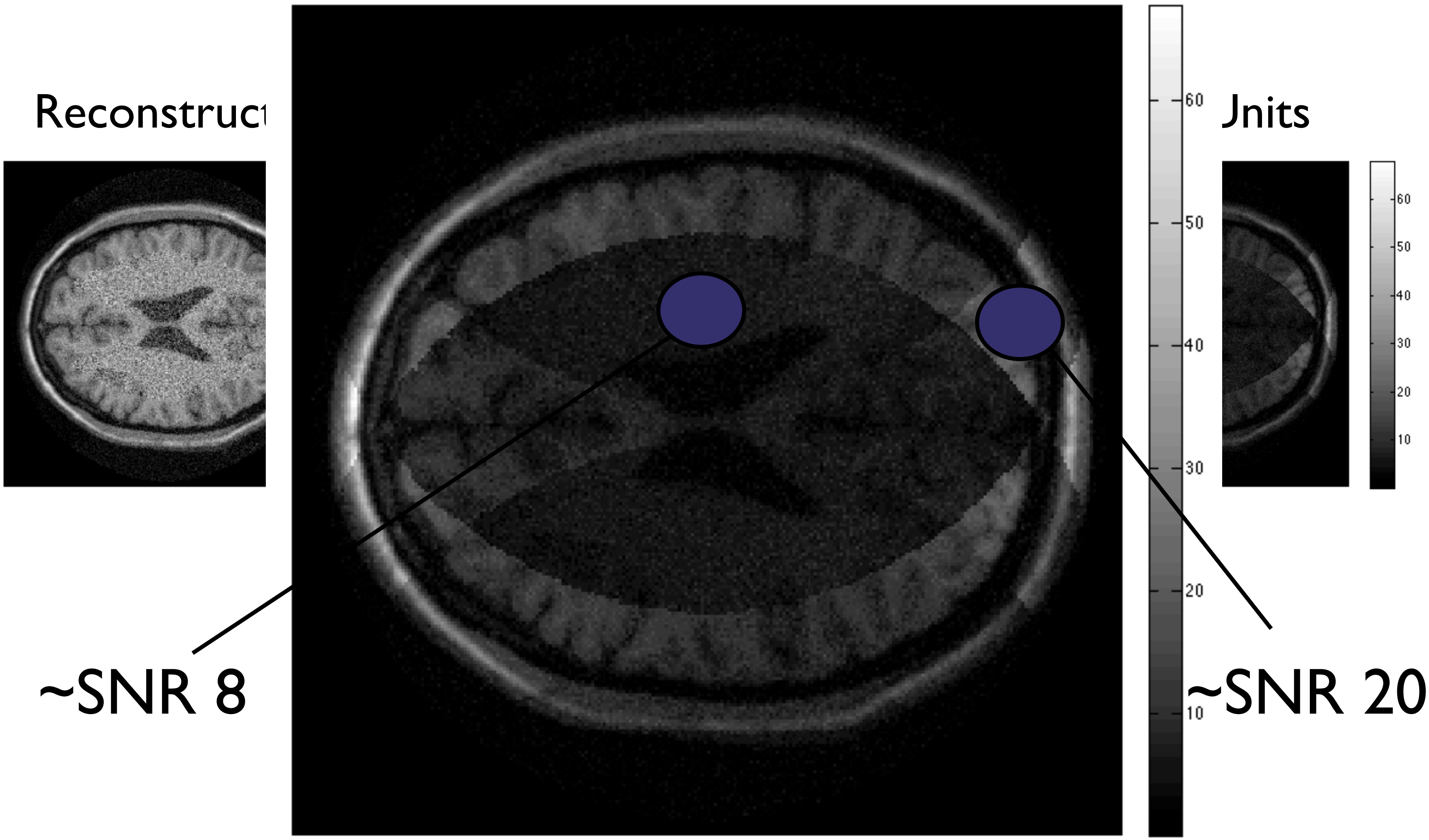
Reconstruction in SNR Units

Reconstruction Pipeline



Kellman et al., Magnetic Resonance in Medicine 54:1439 –1447 (2005)

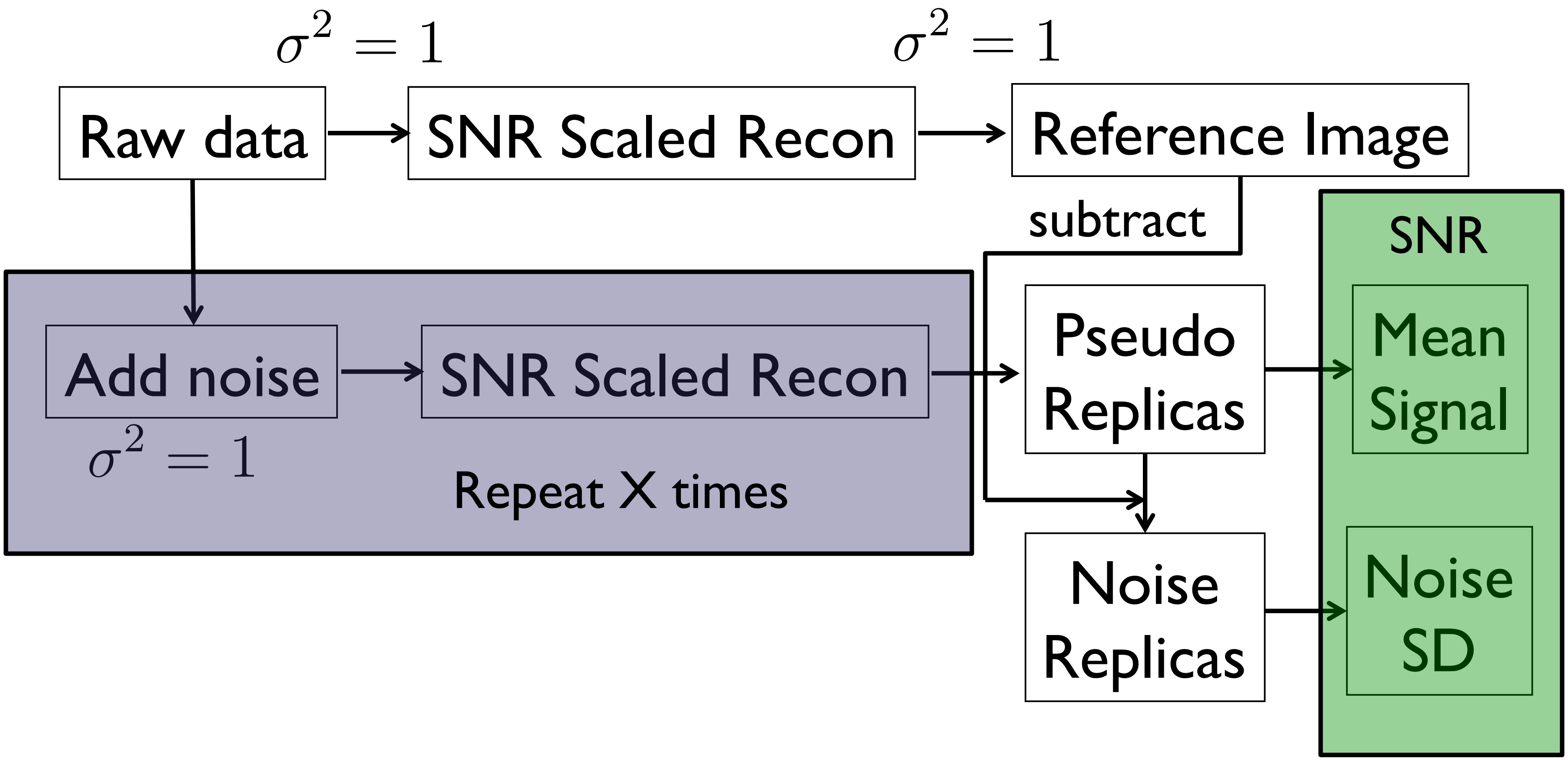
Reconstruction in SNR Units



–1447 (2005)

Pseudo-Replica Method

What if unmixing coefficients are never explicitly formed:

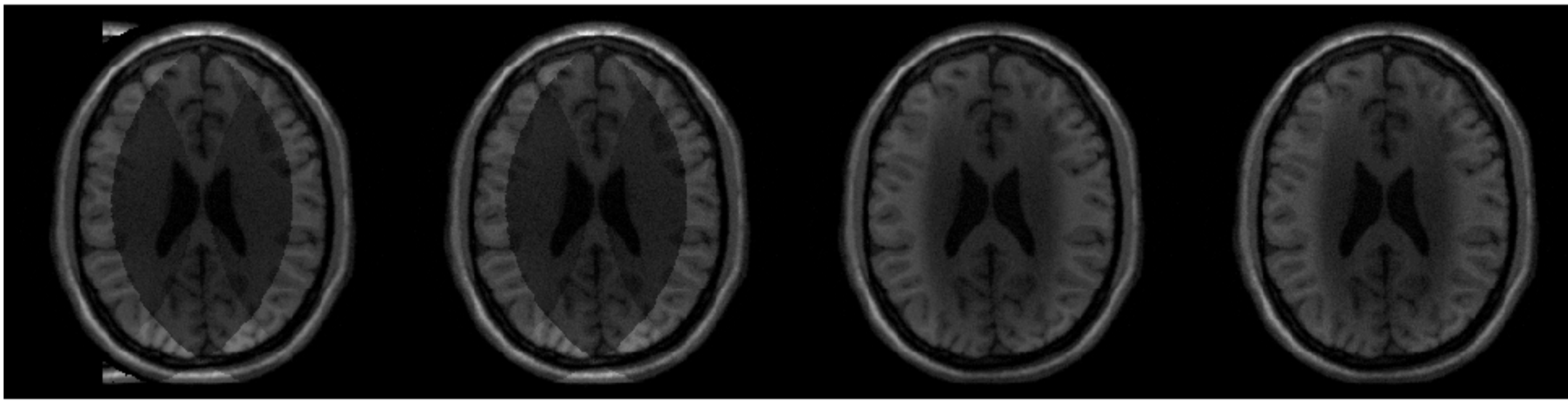


`ismrm_pseudo_replica.m`

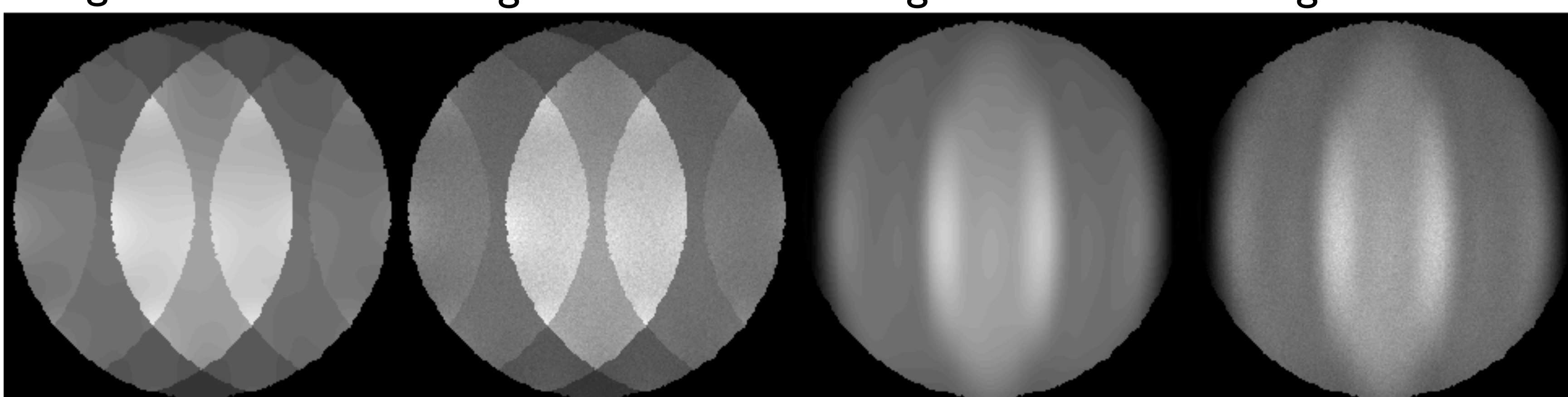
Pseudo-Replica Method – Example 256 trials

SENSE R4 GRAPPA R4

SNR UNMIX SNR PSEUDO SNR UNMIX SNR PSEUDO

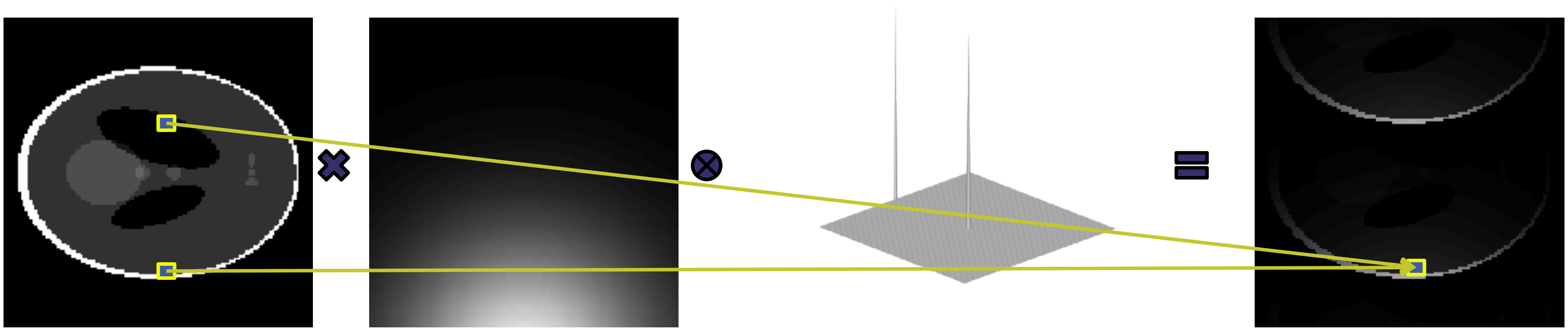


g UNMIX g PSEUDO g UNMIX g PSEUDO

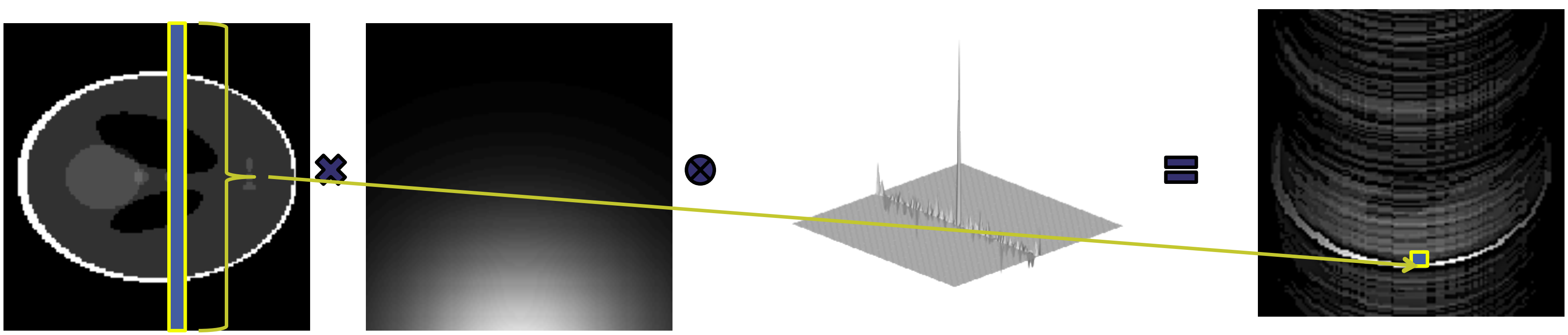


Advantage of Cartesian Undersampling

Cartesian Undersampling



“Random” Undersampling



Non-Cartesian Parallel MRI

To solve the general non-Cartesian case, we return to the original problem:

$$\mathbf{s} = \mathbf{E}\boldsymbol{\rho} \qquad \tilde{\boldsymbol{\rho}} = \arg\min_{\boldsymbol{\rho}} \left\{ \|\mathbf{E}\boldsymbol{\rho} - \mathbf{s}\|_2 \right\}$$

It is not practical to solve with direct inversion in general.

But we can use a number of different iterative solvers to arrive at the solution

- Conjugate Gradients
- LSQR (Matlab)

```
>> help lsqr
 lsqr   lsqr Method.
    X = lsqr(A,B) attempts to solve the system of linear equations A*X=B
    for X if A is consistent, otherwise it attempts to solve the least
    squares solution X that minimizes norm(B-A*X)...

    X = lsqr(AFUN,B) accepts a function handle AFUN instead of the matrix A.
    AFUN(X,'notransp') accepts a vector input X and returns the
    matrix-vector product A*X while AFUN(X,'transp') returns A'*X. In all
    of the following syntaxes, you can replace A by AFUN...
```

Iterative SENSE – First Cartesian

To use LSQR (or Conjugate Gradients), we "just" need to be able to write a function that does the multiplication with E and E^H:

Let's first look at a simple Cartesian case

Multiplication with E^H

```
rho = zeros(size(csm)); %csm: coil sensitivities
%sampling_mask: 1 where sampled, zero where not
rho(repmat(sampling_mask,[1 1 size(csm,3)]) == 1) = s(:);
rho = ismrm_transform_kspace_to_image(rho,[1,2]);
rho = sum(conj(csm) .* rho,3);
```

Multiplication with E

```
s = repmat(reshape(rho,size(csm,1),size(csm,2)),[1 1 size(csm,3)]) .* csm;
s = ismrm_transform_image_to_kspace(s, [1,2]);
s= s(repmat(sampling_mask,[1 1 size(csm,3)]) == 1);
```

Iterative SENSE

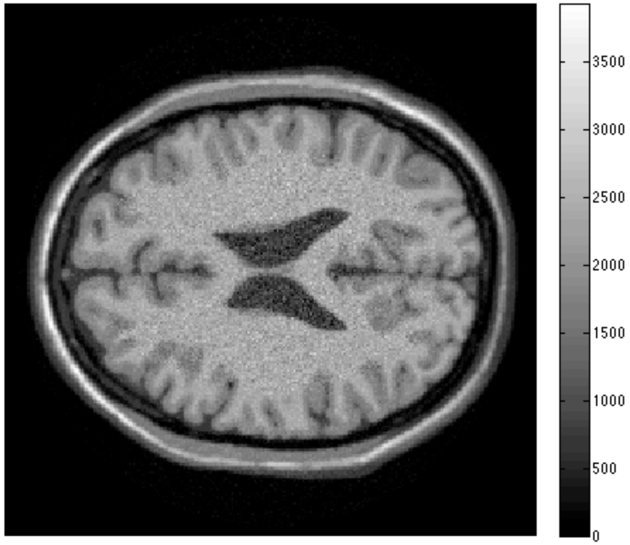
If we have the multiplication with E and E^H implemented as a Matlab function:

```
function o =  e_cartesian_SENSE(inp, csm, sp, transpose_indicator)
% sp: sampling pattern
% csm: coil sensitivities
```

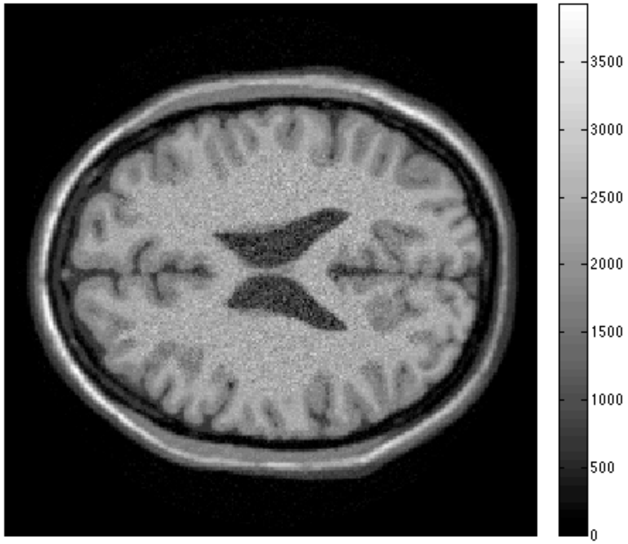
Iterative SENSE could be implemented as:

```
% s: vector of acquired data
E = @(x,tr) e_cartesian_SENSE(x,csm,(sp > 0),tr);
img = lsqr(E, s, 1e-5,50);
img = reshape(img,size(csm,1),size(csm,2));
```

Cartesian SENSE



Iterative SENSE



Quick note on the non-uniform FFT

To implement multiplication with E and E^H in the non-Cartesian case, we need to do the non-uniform Fourier transform[1,2].

In this course, we will use Jeff Fesslers "nufft" package. We recommend you download the latest version from:

http://web.eecs.umich.edu/~fessler/irt/fessler.tgz

```
%k: k-space coordinates [Nsamples, 2], range –pi:pi
%w: Density compensation weights
%s: Data

%Prepare NUFFT
N = [256 256]; %Matrix size
J = [5 5];     %Kernel size
K = N*2;       %Oversampled Matrix size
nufft_st = nufft_init(k,N,J,K,N/2,'minmax:kb');

recon = nufft_adj(s .* repmat(w,[1 size(s,2)]),nufft_st);
```

[1]Keiner, J., Kunis, S., and Potts, D. Using NFFT 3 - a software library for various nonequispaced fast Fourier transforms. ACM Trans. Math. Software, 2009
[2]Fessler J and Sutton B. Nonuniform fast Fourier transforms using min-max interpolation. IEEE TSP 2003

Iterative SENSE – non-Cartesian

To use LSQR (or Conjugate Gradients), we "just" need to be able to write a function that does the multiplication with E and E^H:

Now we have the tools for the non-Cartesian case:

Multiplication with E^H

```
samples = size(nufft_st.om,1); coils = numel(s)/samples;
s = reshape(s,samples,coils);
rho = nufft_adj(s .* repmat(sqrt(w),[1 coils]),nufft_st)./sqrt(prod(nufft_st.Kd));
rho = sum(conj(csm) .* rho,3);
rho = rho(:);
```

Ensure operators are adjoint

From `nufft_init`

Multiplication with E

```
s = repmat(reshape(rho,size(csm,1),size(csm,2)),[1 1 size(csm,3)]) .* csm;
s = nufft(s,nufft_st)./sqrt(prod(nufft_st.Kd));
s = s .*repmat(sqrt(w),[1 size(s,2)]);
s = s(:);
```

Iterative SENSE – non-Cartesian

If we have the multiplication with E and E^H implemented as a Matlab function:

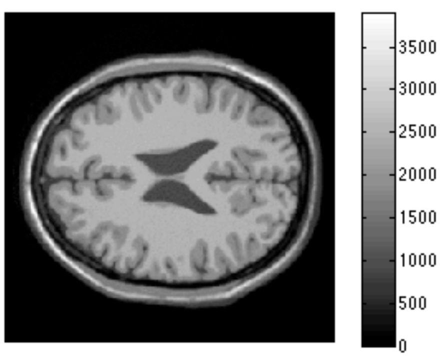
```
function o = e_non_cartesian_SENSE(inp, csm, nufft_st, w, transpose_indicator)
% nufft_st: From nufft_init
% csm: coil sensitivities, w: density compensation
```

Non-Cartesian SENSE could be implemented as:

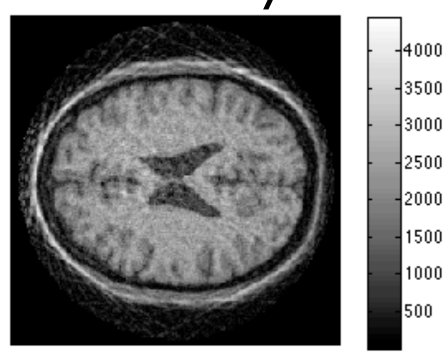
```
% s: vector of acquired data
E = @(x,tr) e_non_cartesian_SENSE(x, csm, nufft_st, w, tr);
img = lsqr(E, s .* repmat(sqrt(w),[size(csm,3),1]), 1e-3,30);
img = reshape(img,size(csm,1),size(csm,2));
```

Due to definition of E

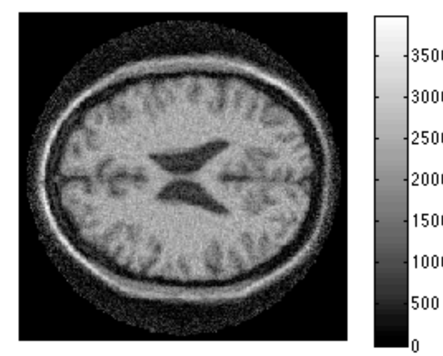
Fully sampled



24 projections
nufft only



24 projections
SENSE



Regularization – Iterative Methods

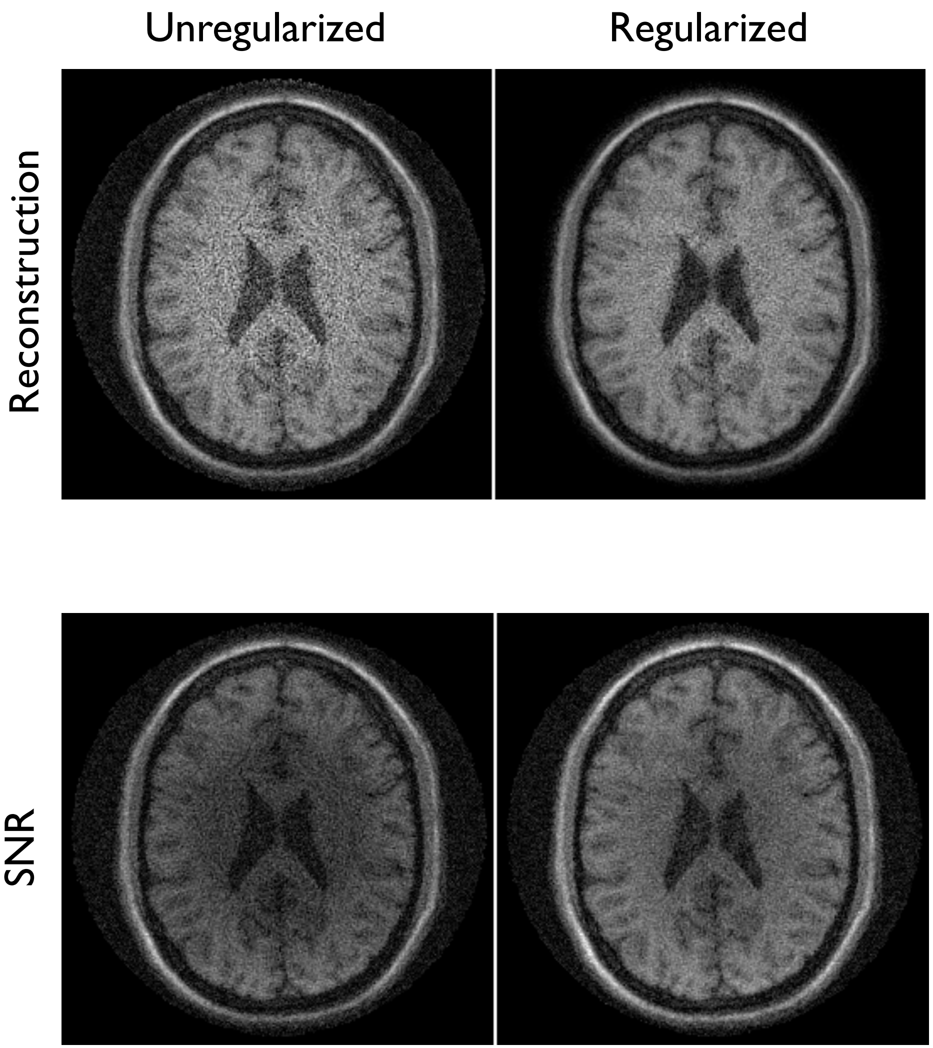
$$\tilde{\boldsymbol{\rho}} = \arg\min_{\boldsymbol{\rho}} \left\{ \left\| \mathbf{E}\boldsymbol{\rho} - \mathbf{s} \right\|_2 + \lambda \left\| \mathbf{L}\boldsymbol{\rho} \right\|_2 \right\}$$

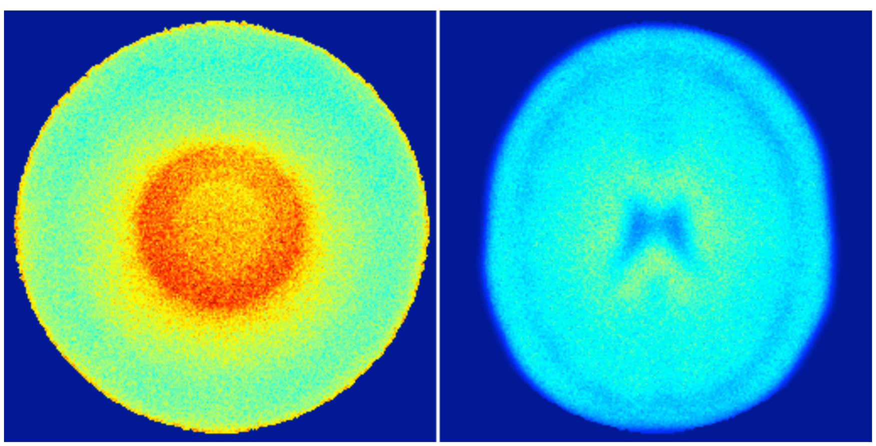
Equivalent to solving:

$$\begin{array}{r} \text{Measured data} \longrightarrow \\ \text{Vector of zeros} \longrightarrow \end{array} \begin{bmatrix} \mathbf{s} \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{E} \\ \mathbf{L} \end{bmatrix} \boldsymbol{\rho}$$

`ismrm_demo_regularization_iterative_sense.m`

Regularization – Iterative Methods

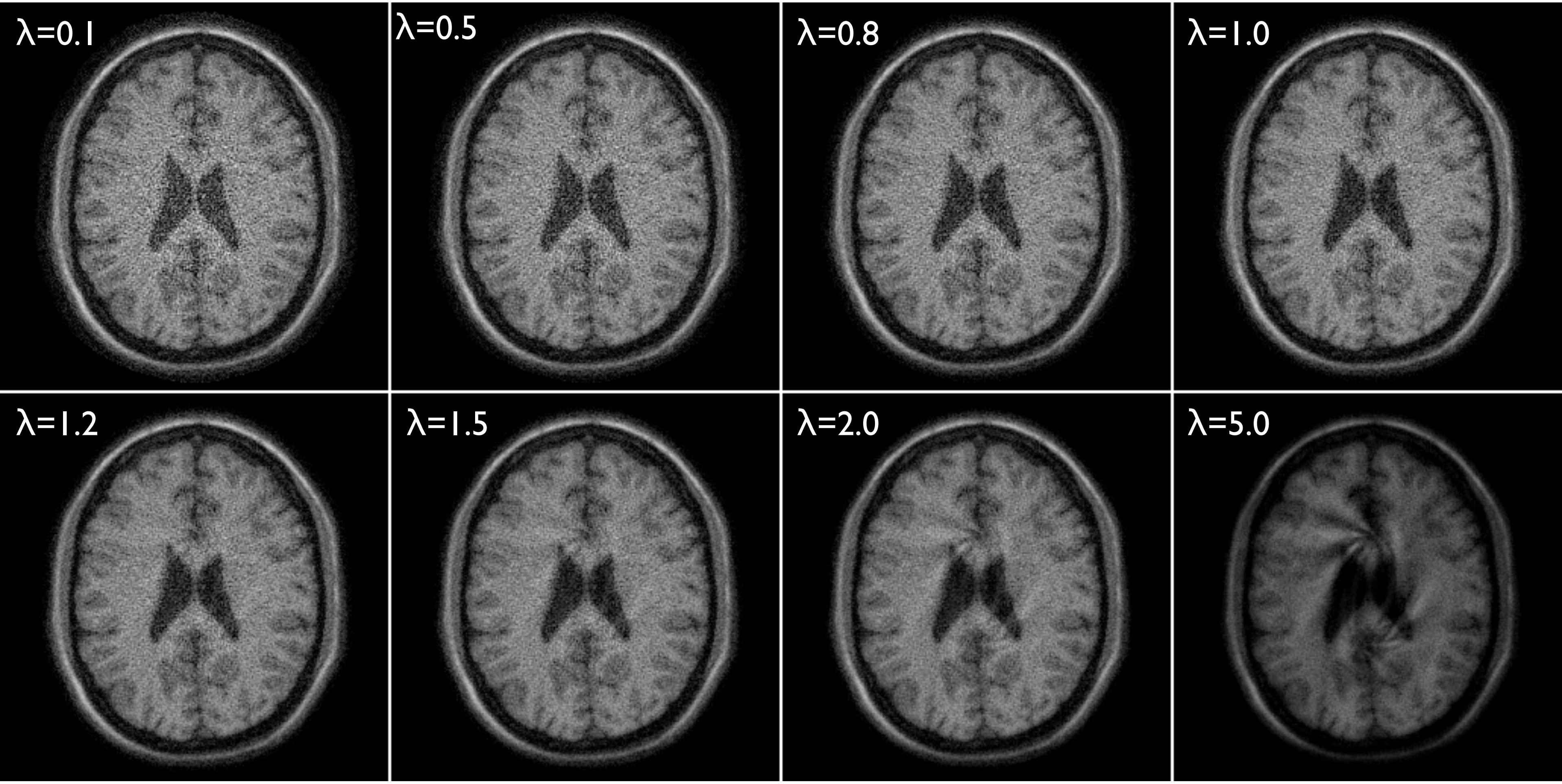




ismrm_demo_regularization_iterative_sense.m

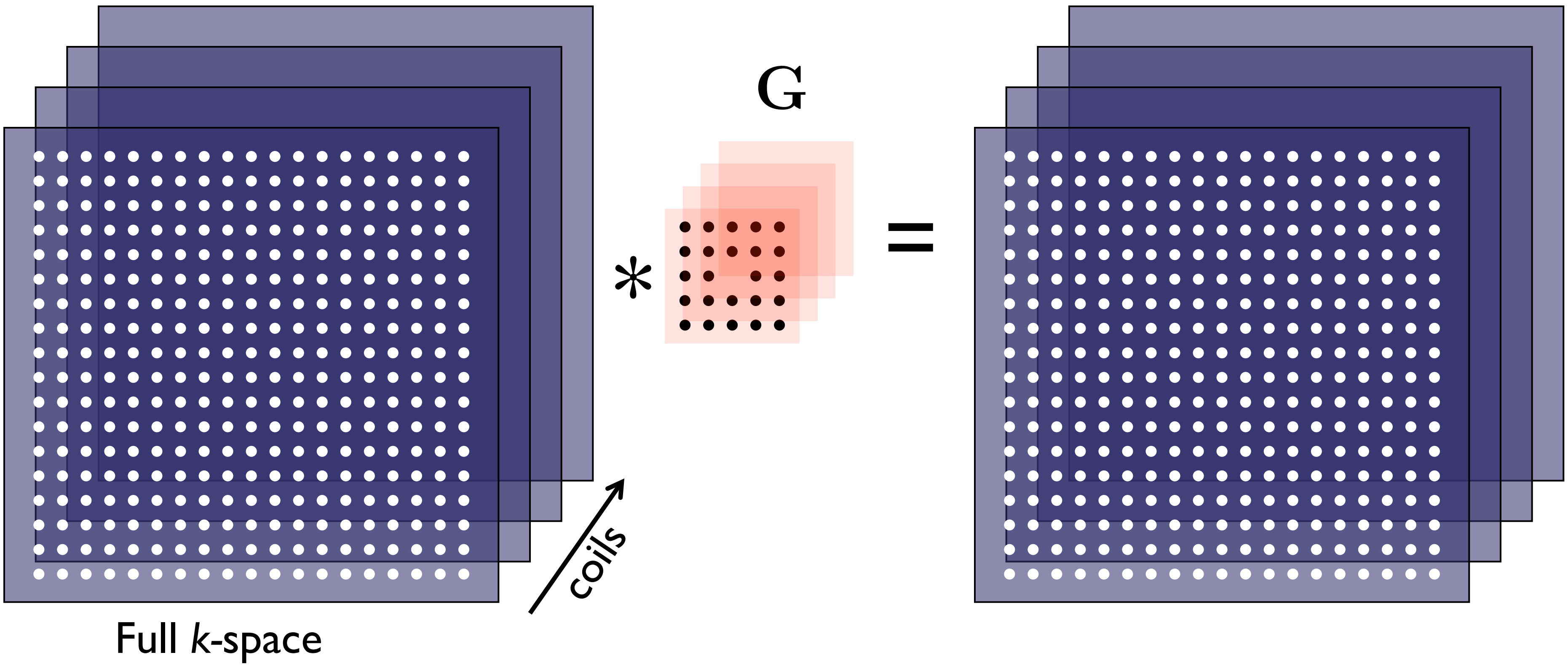
Regularization – Iterative Methods

$$\tilde{\boldsymbol{\rho}} = \arg\min_{\boldsymbol{\rho}} \{\|\mathbf{E}\boldsymbol{\rho} - \mathbf{s}\|_2 + \lambda\|\mathbf{L}\boldsymbol{\rho}\|_2\}$$



SPIRiT Approach

k-space points can be synthesized from neighbors



$$\mathbf{G}\mathbf{d} = \mathbf{d}$$

Lustig and Pauly. Magn Reson Med. 2010

SPIRiT Approach

We can formulate the reconstruction problem in *k*-space as:

$$\tilde{\mathbf{x}} = \arg\min_{\mathbf{x}} \{\|\mathbf{D}\mathbf{x} - \mathbf{y}\|_2 + \lambda \|\mathbf{G}\mathbf{x} - \mathbf{x}\|_2\}$$

$\mathbf{x}$	:	Cartesian k-space solution.
$\mathbf{D}$	:	Sampling operator (e.g. onto non-Cartesian k-space)
$\mathbf{y}$	:	Sampled data
$\mathbf{G}$	:	SPIRiT convolution operator

Can be applied as multiplication in image space

Could also be sampling operator from image to *k*-space

Lustig and Pauly. Magn Reson Med. 2010

Spiral Imaging Example

Gridding SENSE SPIRiT

`ismrm_demo_non_cartesian.m`

Summary

- Noise decorrelation is used to reduce the impact of varying noise levels in receive channels.
- SNR scaled reconstruction are a way to evaluate reconstructions directly on the images.
- Pseudo Replica Method allows the formation of SNR scaled images in methods where unmixing coefficients are not explicitly obtained
- Iterative methods can be used for both Cartesian and non-Cartesian methods
- Regularization can be added to iterative methods in a straightforward fashion

Acknowledgements

- Jeff Fessler
  - http://web.eecs.umich.edu/~fessler/code/
- Brian Hargreaves
  - http://mrsrl.stanford.edu/~brian/mritools.html
- Miki Lustig
  - http://www.eecs.berkeley.edu/~mlustig/Software.html

Download code, examples:

http://gadgetron.sf.net/sunrise

http://gadgetron.sourceforge.net/sunrise/

ISMRM Sunrise Course on

Teachers

Michael S. Hansen - michael.hansen@nih.gov
Philip Beatty - philip.beatty@sri.utoronto.ca

Slides

- Hansen Slides

Source Code and Examples

Download ismrm_sunrise_parallel.zip with source code and

Exercises

- Beatty Practical Session
- Hansen Practical Session

ISMRM Sunrise Practical Session

This document contains the second set of practical exercises for the ISMRM course on parallel imaging.

Contents

Excercise Data

All the data used in this set of exercises can be found in the file hansen_exercises.mat. We will start by clearing the workspace and loac

```
close all; clear all;
load hansen_exercises.mat
whos
```

```
  Name                 Size                Bytes  Class     Attributes

  data               256x256x8           8388608  double    complex
  data_spiral      18176x8               2326528  double    complex
  k_spiral         18176x2                290816  double
  noise_color        256x256x8           8388608  double    complex
  noise_spiral     18176x8               2326528  double    complex
  reg_img            256x256              524288  double
  smaps              256x256x8           8388608  double    complex
  sp                 256x256              524288  double
  w_spiral         18176x1                145408  double
```

Noise Pre-Whitening

Hands-on Cheat Sheet