Image Reconstruction

Parallel Imaging / Coil Compression

M229 Advanced Topics in MRI

Kyung Sung, Ph.D.

2019.05.28

Class Business

- Final project abstract / presentation
- Office hours
  - Instructors: Fri 10-12 noon
  - email beforehand would be helpful

Today's Topics

- Parallel Imaging
  - SMASH review
  - Auto-SMASH
  - GRAPPA
- Coil compression
- k-t BLAST / k-t SENSE

SMASH Review

- The linear combination of coil sensitivities looks like sinusoids:

$$e^{-i2\pi(m\Delta k_y)y} = \sum_{j=0}^{L-1} a_{j,m} C_j(y)$$

- Once we have $a_{j,m}$,

$$\hat{m}(k_y + m\Delta k_y) = \int_y m(y) e^{-i2\pi k_y y} e^{-2\pi(m\Delta k_y)y} dy$$

$$\hat{m}(k_y + m\Delta k_y) = \int_y m(y) e^{-i2\pi k_y y} \sum_{j=0}^{L-1} a_{j,m} C_j(y) dy$$

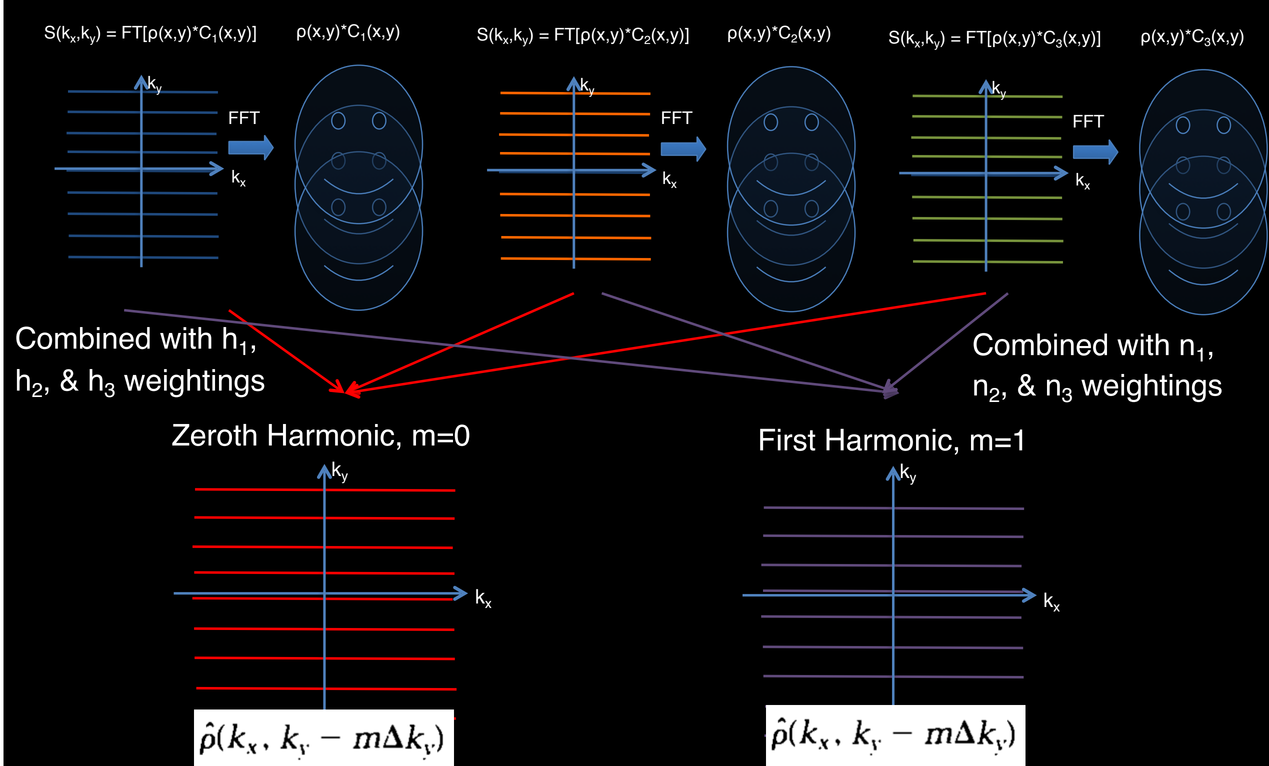
SMASH Review

$$\hat{m}(k_y + m\Delta k_y) = \int_y m(y) e^{-i2\pi k_y y} \sum_{j=0}^{L-1} a_{j,m} C_j(y) dy$$

$$\hat{m}(k_y + m\Delta k_y) = \sum_{j=0}^{L-1} a_{j,m} \int_y C_j(y) m(y) e^{-i2\pi k_y y} dy$$

$$\hat{m}(k_y + m\Delta k_y) = \sum_{j=0}^{L-1} a_{j,m} m_j(k_y)$$

SMASH Reconstruction



Auto-SMASH

- Estimate $a_{j,m}$ directly

calibration

$$\hat{m}(k_y + m\Delta k_y) = \sum_{j=0}^{L-1} a_{j,m} m_j(k_y)$$

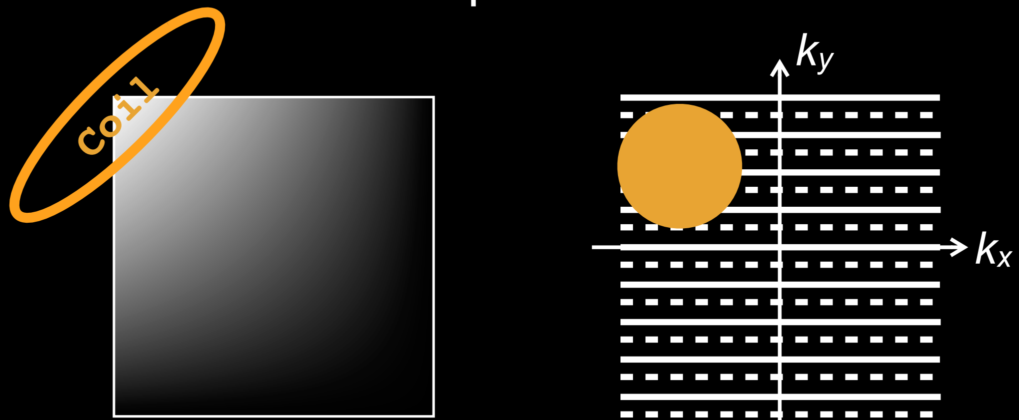
synthesis

- Solve for $a_{j,m}$ from calibration data & synthesize the missing data with $a_{j,m}$

Parallel Imaging (GRAPPA)

GRAPPA

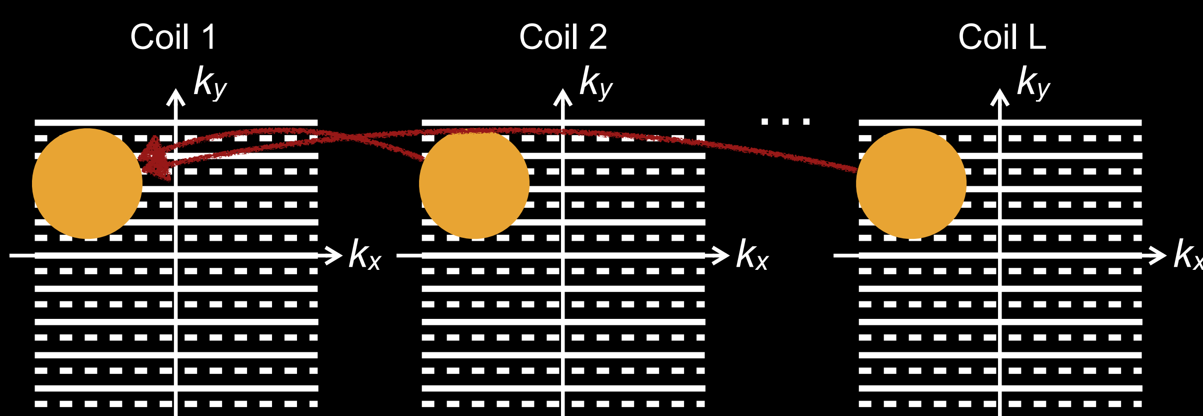
- Coil sensitivities are
  - local in image space
  - extended in k-space



$m(\vec{x})C_j(\vec{x})$ $\qquad$ $M(\vec{k}) * C_j(\vec{k})$

GRAPPA

- Missing information is implicitly contained by adjacent data



$M(\vec{k}) * C_1(\vec{k})$ $\qquad$ $M(\vec{k}) * C_2(\vec{k})$ $\qquad$ $M(\vec{k}) * C_L(\vec{k})$

GRAPPA Reconstruction

- How do we find missing data from these samples?

$$\underbrace{\hat{m}_k(k_x, k_y)}_{\text{missing data for each coil}} = \sum_{i,j,k} \underbrace{a_{i,j,k}}_{\text{weights}} \cdot \underbrace{m_k(k_x + i\Delta k_x, k_y + j\Delta k_y)}_{\text{neighborhood data for each coil}}$$

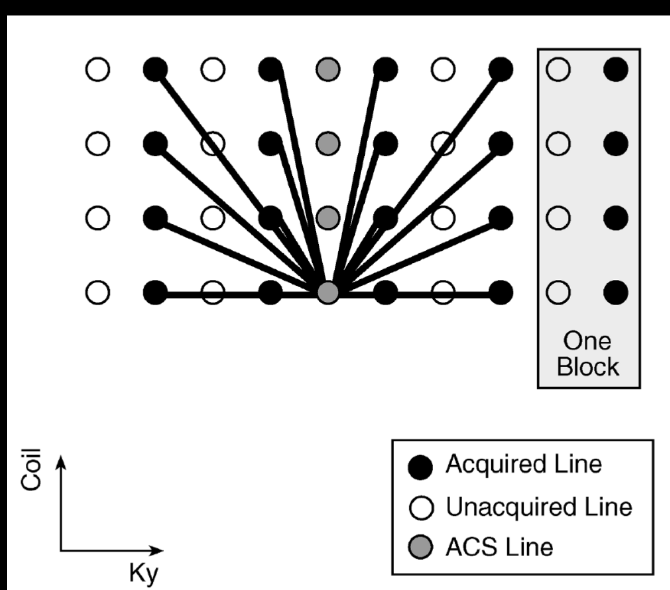
missing data for each coil

weights

neighborhood data for each coil

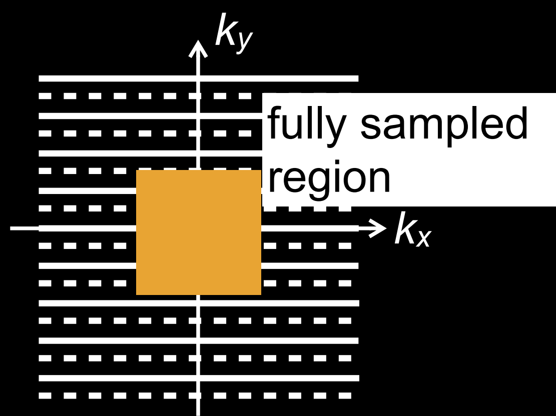
Auto-Calibration

$$\hat{m}_k(k_x, k_y) = \sum_{i,j,k} a_{i,j,k} \cdot m_k(k_x + i\Delta k_x, k_y + j\Delta k_y)$$



Auto-Calibration

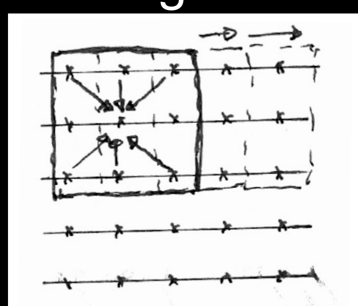
- Assume there is a fully sampled region
- We have samples of what the GRAPPA synthesis equations should produce



- Invert this to solve for GRAPPA weights

Auto-Calibration

- Calibration area has to be larger than the GRAPPA kernel
- Each shift of kernel gives another equation



- Here, 3x3 kernel, 5x5 calibration area gives 9 equations

Auto-Calibration

$$\hat{m}_k(k_x, k_y) = \sum_{i,j,k} a_{i,j,k} \cdot m_k(k_x + i\Delta k_x, k_y + j\Delta k_y)$$

- Write as a matrix equation

GRAPPA Coefficients

$$M_{k,c} = M_A \cdot a_k$$

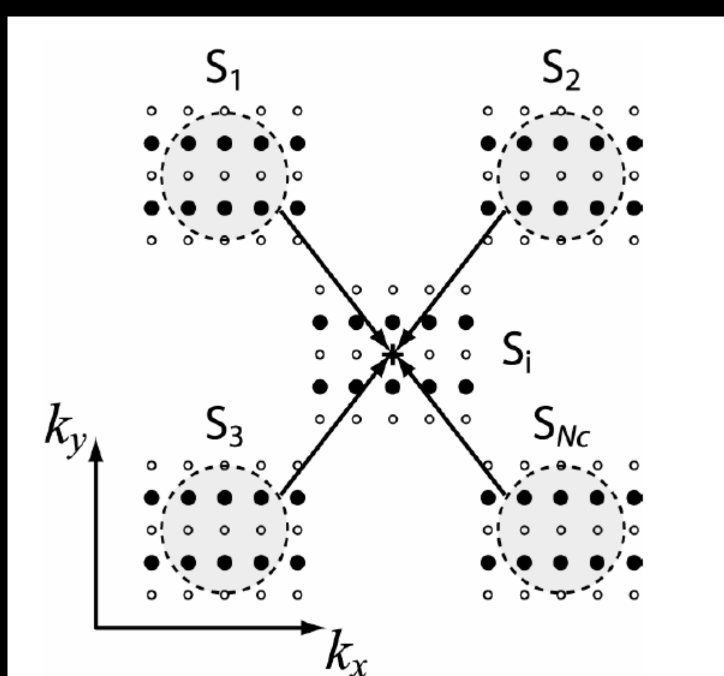
Calibration Data

Neighborhood Data

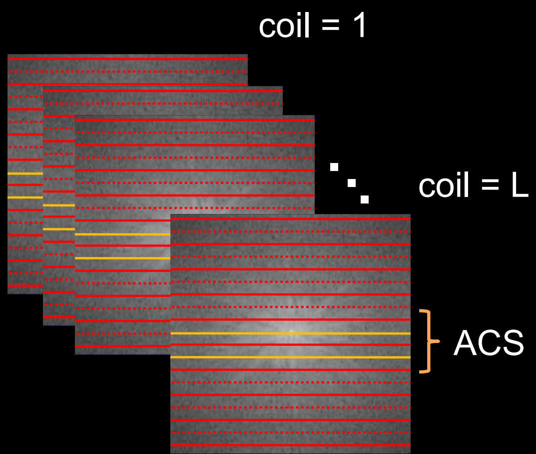
- GRAPPA weights are:

$$a_k = (M_A^* M_A + \lambda I)^{-1} M_A^* M_{k,c}$$

GRAPPA - Synthesis



Auto-Calibration Parallel Imaging



ACS (Auto-Calibration Signal) lines

$$\sum_{l=1}^{L} S_l^{ACS}(k_y - m\Delta k_y) = \sum_{l=1}^{L} n(l,\, m) S_l(k_y)$$

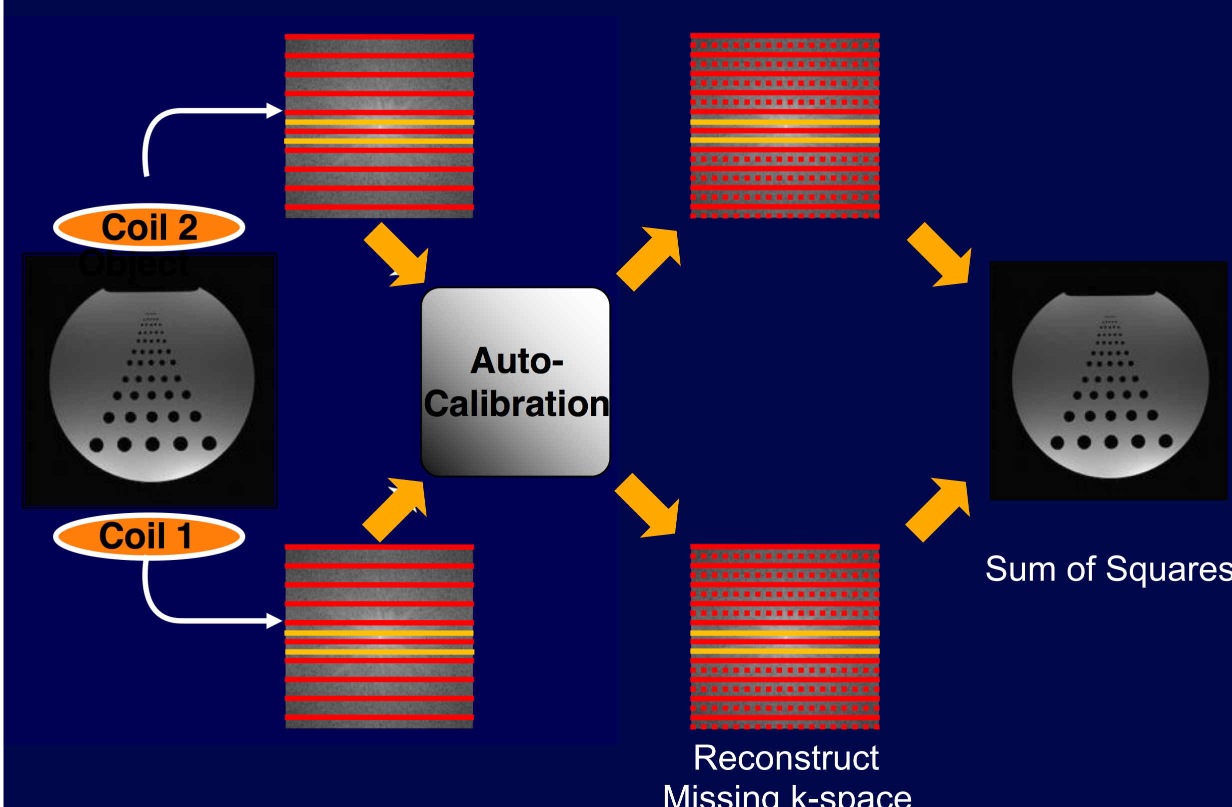
GRAPPA formula to reconstruct signal in one channel

$$S_j(k_y - m\Delta k_y) = \sum_{l=1}^{L} \sum_{b=0}^{N_b-1} n(j,\, b,\, l,\, m) S_l(k_y - bA\Delta k_y)$$

A: Acceleration factor
n(j,b,l,m): GRAPPA weights

Griswold et al. MRM, 47(6):1202-1210 (2002)

GRAPPA Reconstruction

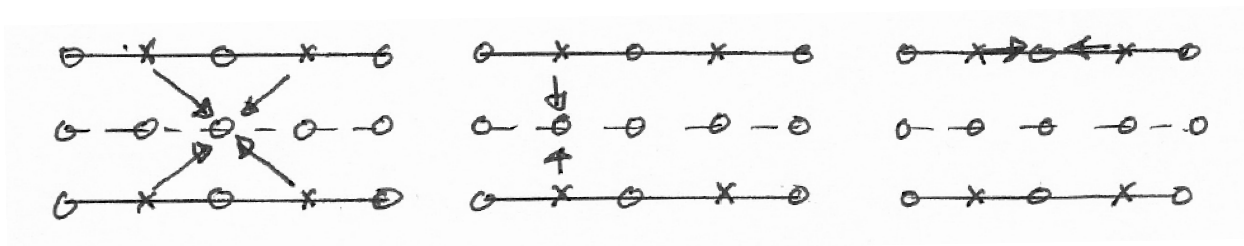


GRAPPA

- Compute GRAPPA weights from calibration region
- Compute missing k-space data using the GRAPPA weights
- Reconstruct individual coil images
- Combine coil images

Considerations of GRAPPA

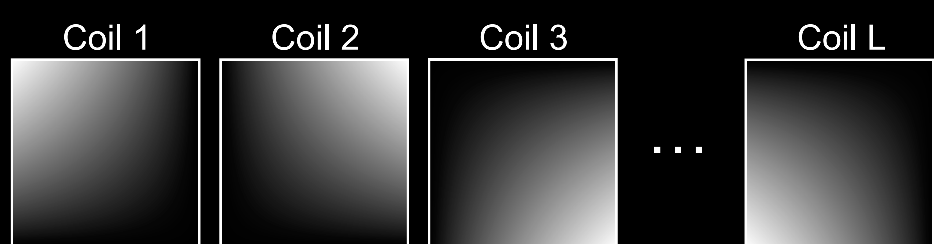
- Calibration region size
- GRAPPA kernel size
- Sample geometry dependence



Coil Compression

Coil Compression

- Array coil sensitivities



- Each coil sees a local region
- Not clear how much acceleration is possible
  - g-factor hits a wall at 3-4 in 1D, why?
  - What is the fundamental dimensionality?

Eigen Coils

- Make a matrix of vectorized sensitivity maps

$$\begin{pmatrix} \vdots & \vdots & \vdots \\ C_1(\vec{x}) & C_2(\vec{x}) & C_3(\vec{x}) \\ \vdots & \vdots & \vdots \end{pmatrix}$$

- The matrix C*C shows the correlation between channels

Eigen Coils

- Compute the eigen decomposition of C*C

C*C = BDB*

  - B is a unitarty matrix of eigenvectors

$$D = \begin{pmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ & & \ddots & \\ 0 & & & \lambda_L \end{pmatrix}$$

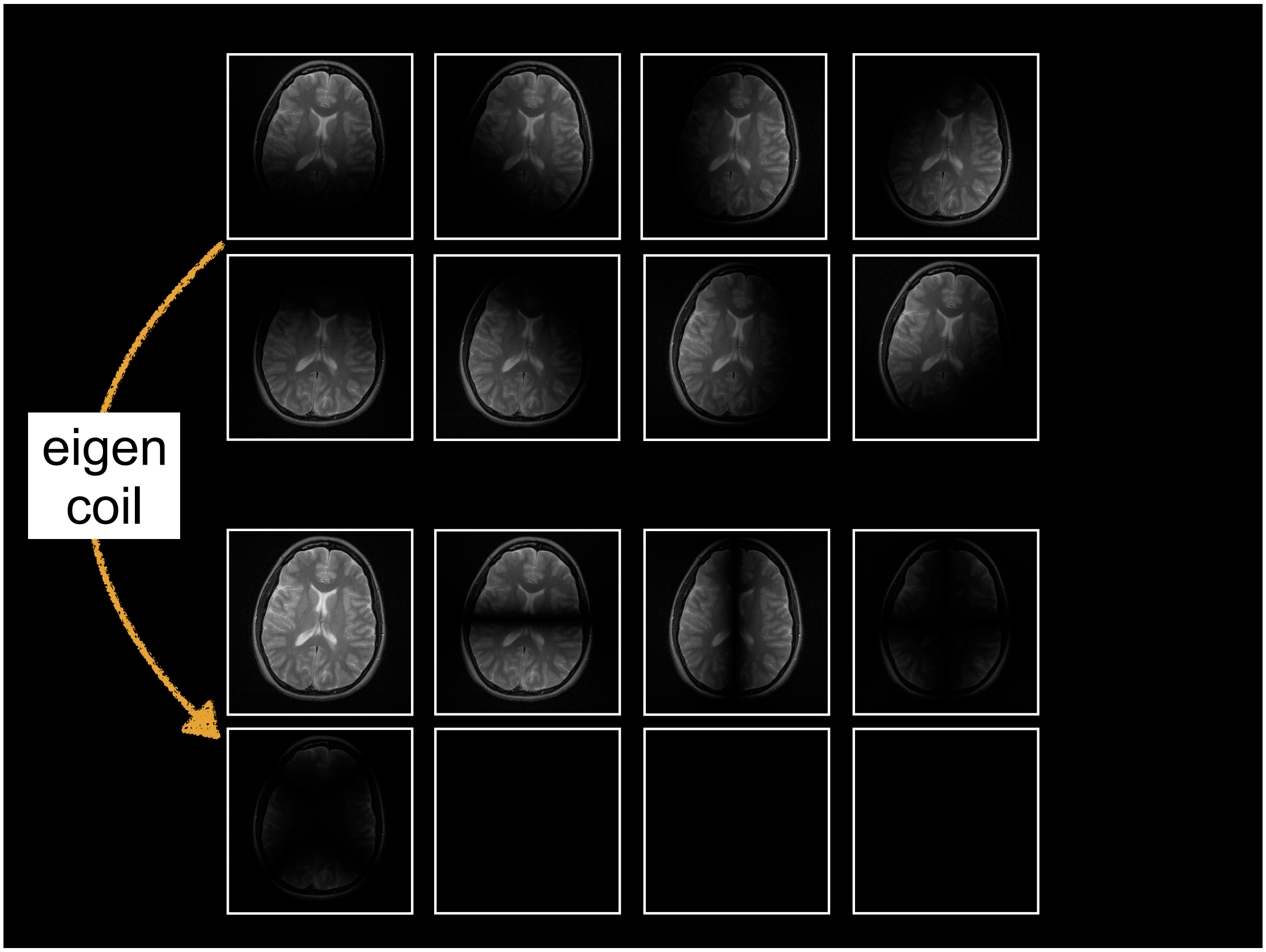
  - Diagonal matrix of eigenvalues

Eigen Coils

- B*C*CB = D
  C'
- C' = CB
  - λ_i tells you how much energy is in each eigen coil channel
  - These eigen coils drop off rapidly, telling how many independent channel you have

MATLAB Demo

```
load brain_mcoil.mat

[nx, ny, nc] = size(im);

C = reshape(im,nx*ny,nc);

[B, D] = eig(C'*C);

C_hat = C*B;

C_hat = reshape(C_hat,nx,ny,nc);
```



Coil Compression

- Use the eigen coil basis to reduce the size of your parallel imaging reconstruction
- M is a matrix of the vectorized aliased data, compute

$$M' = MB$$

  - the data rotated into the eigen coil space
  - only keep the colums of M' that have significant eigen coils
  - Reconstruct using eigen coils C'

Thanks!

- Next time
  - Compressed Sensing

Kyung Sung, PhD

ksung@mednet.ucla.edu

http://kyungs.bol.ucla.edu