Multi-Channel Signal Acquisition and Image Reconstruction

MBP1400H

Mark Chiew (mark.chiew@utoronto.ca) **March 12, 2025**

Outline

- **Part 0 – Math You Should Know**
- **Part I – Measurement Model**
    - Imaging as a discrete linear system
    - Multi-channel hardware & measurement model
- **Part II – Sensitivities & Pre-processing**
    - Sensitivity maps and estimation
    - Channel combination
    - Noise pre-whitening
- **Part III – Parallel Imaging**
    - Linear encoding model of PI
    - SENSE (image domain PI)
    - GRAPPA (k-space domain PI)
- **Part IV – Noise Propagation**
    - $\sqrt{R}$ SNR penalty
    - g-factor noise amplification
    - Noise covariance

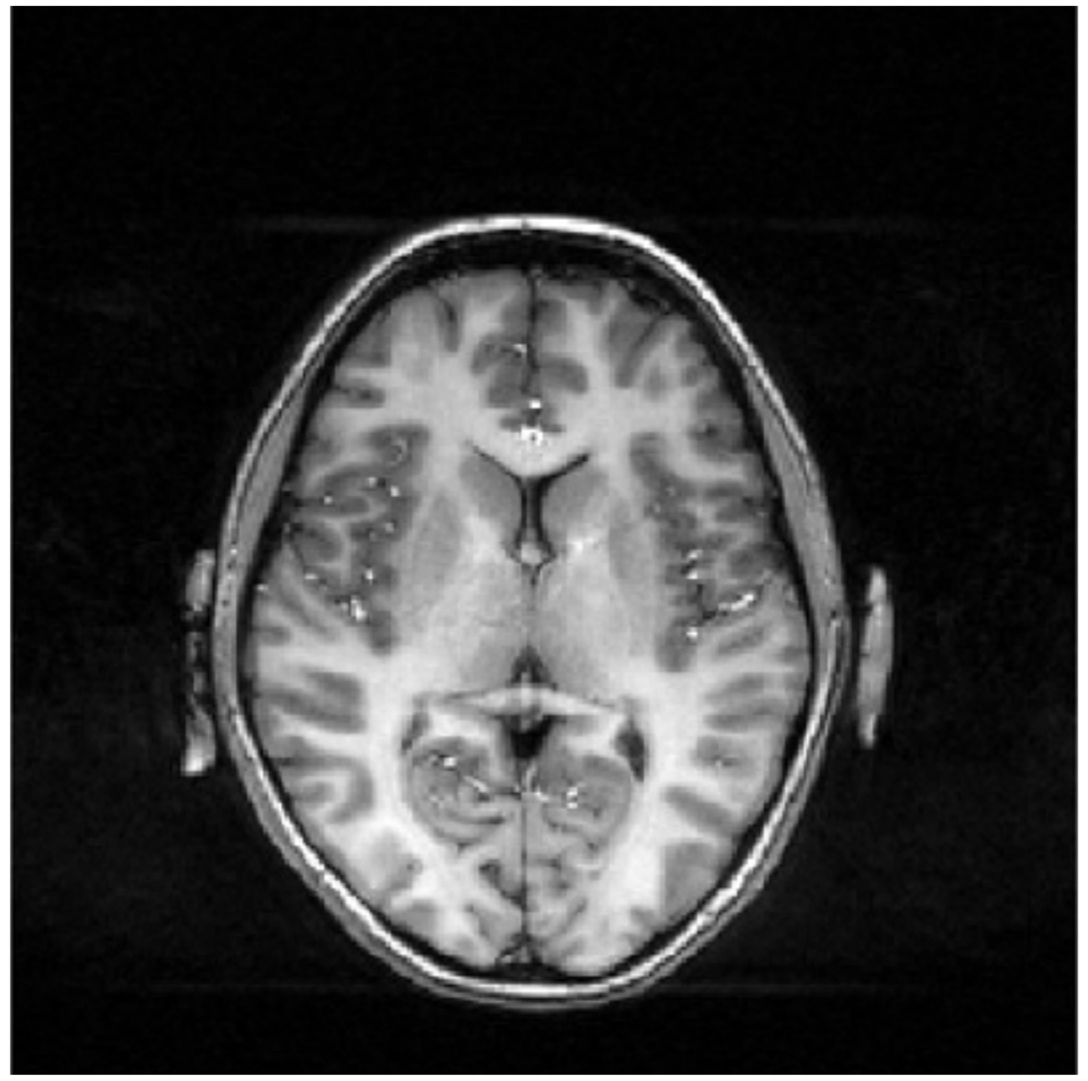
Part 0 – Math You Should Know

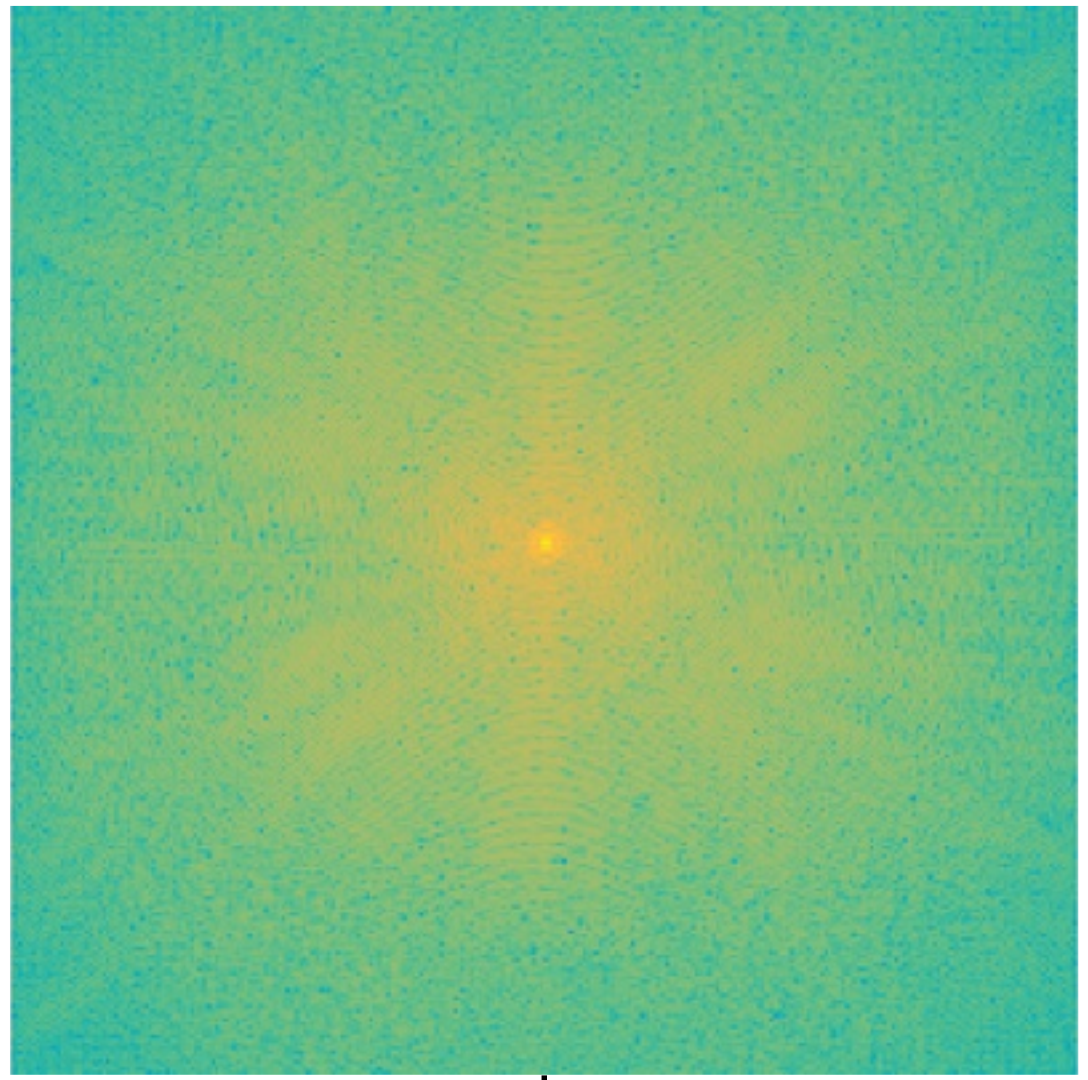
- Linear Systems (i.e. matrix equations)
    - Existence and uniqueness of solutions
    - Least squares solutions, pseudo-inverse
    - Subspaces and Null-spaces
    - Maxtrix Factorizations and Decompositions
    - Eigenvalues and Eigenvectors
- Fourier Transforms
    - Linearity
    - Convolution Theorem
- Optimization
    - Finding optima by taking derivatives/gradients of functions
- Statistics
    - Variance of linear estimators

Part I – Measurement Model

Continuous vs. Discrete

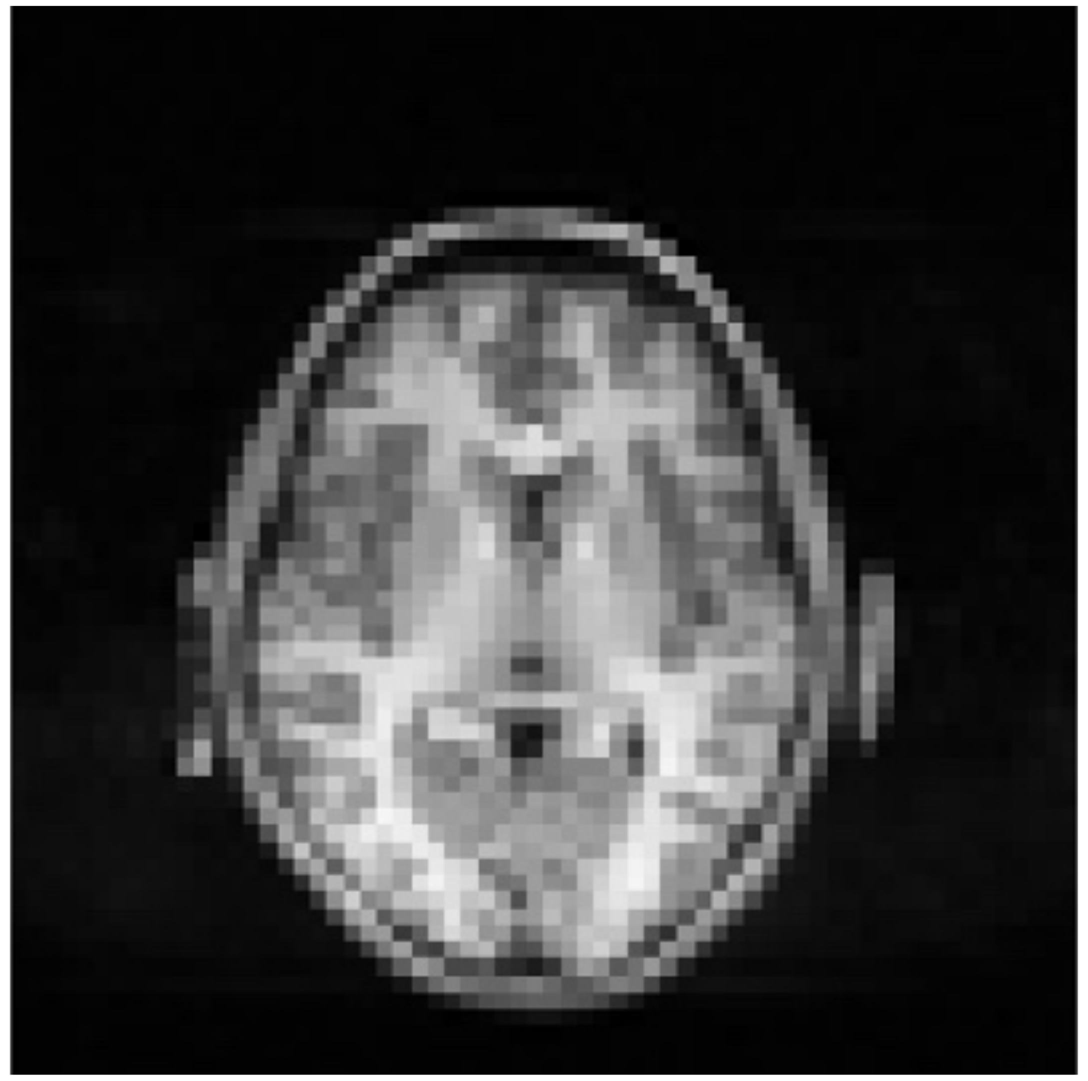
Continuous
source space

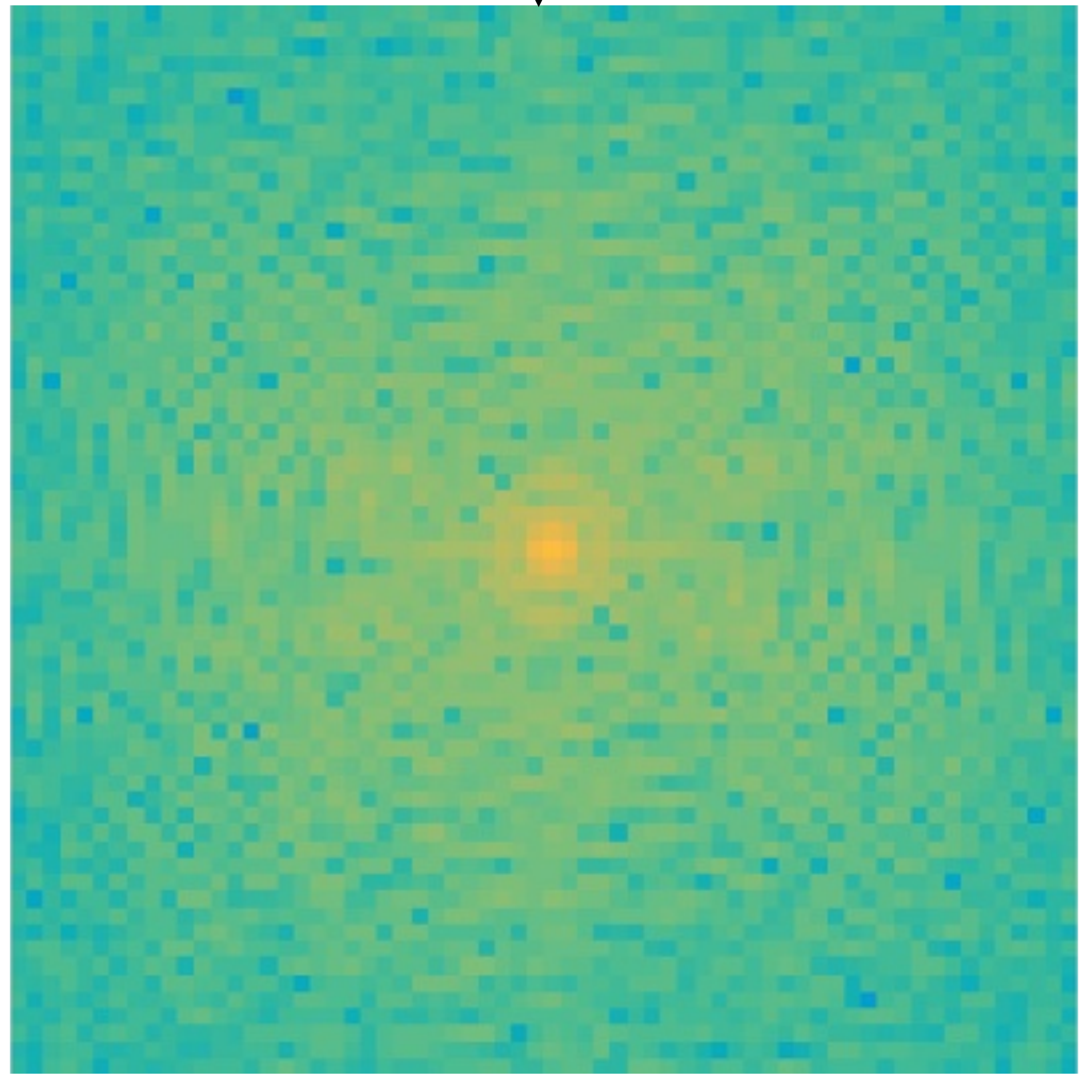




Continuous
k-space

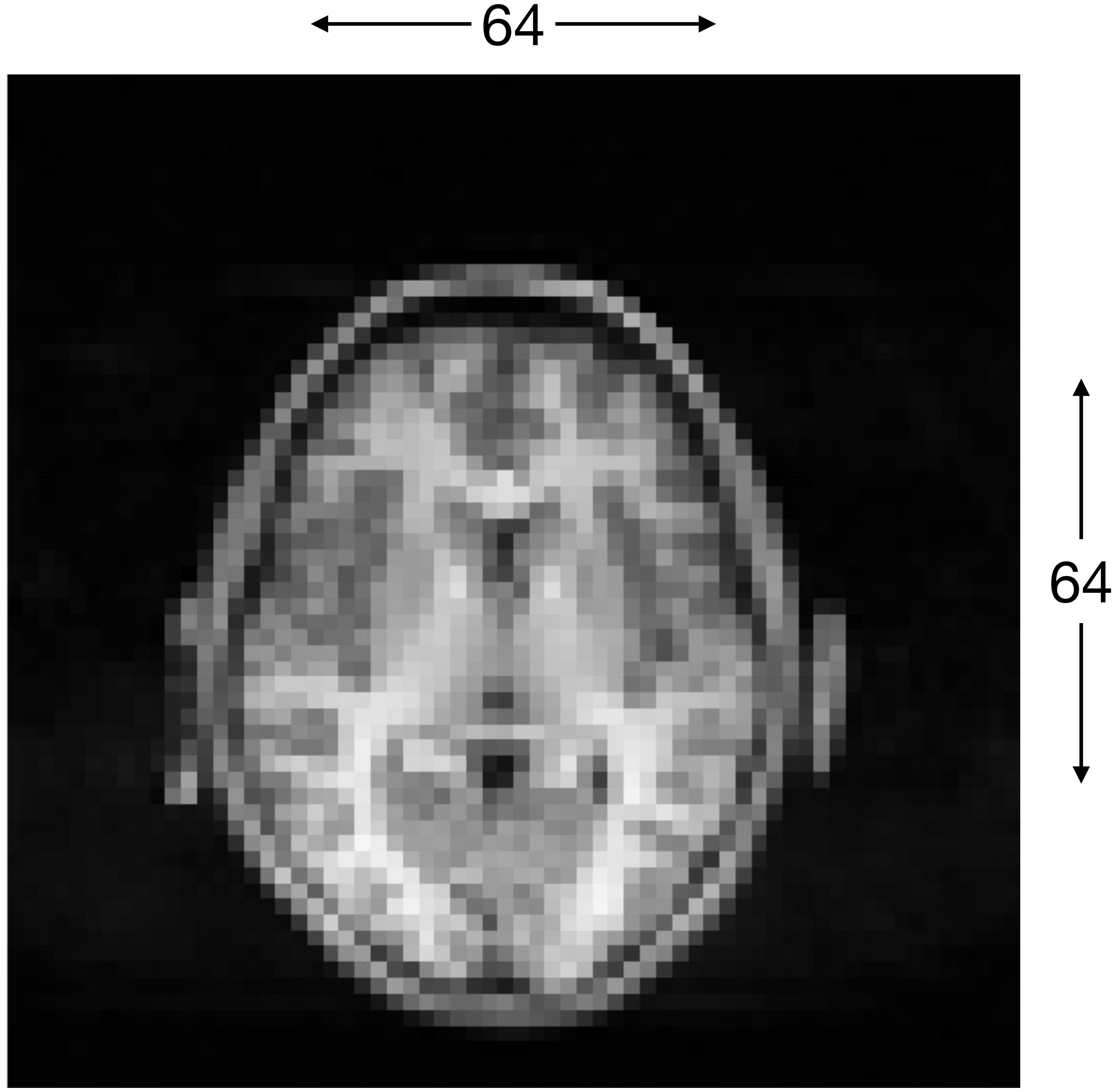
Discrete
image space





Discrete
k-space

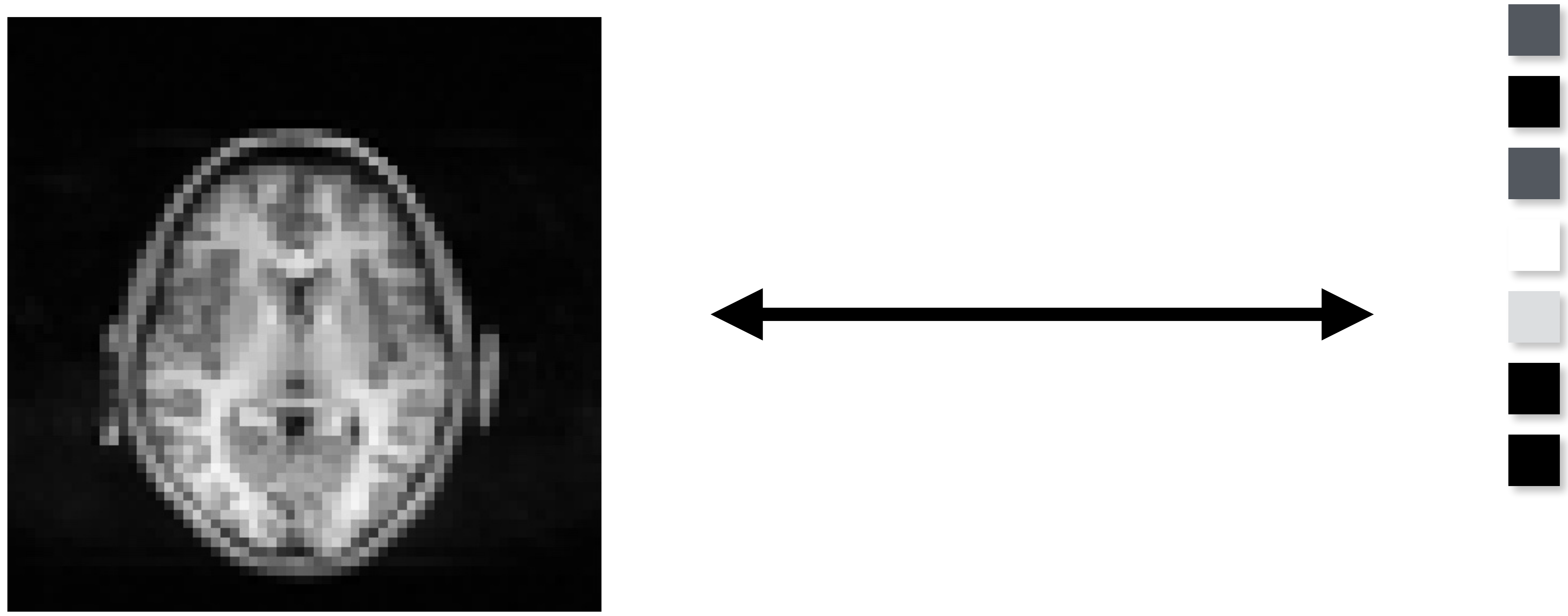
The Image as an Array of Values



- An $m \times n$ voxel image contains mn voxels
- With no knowledge or assumptions, each voxel's value is independent of the others
- Therefore the image data itself lies in $\mathbb{C}^{mn}$ (e.g. 4096 dimensional complex vector space)

The Image as a Vector

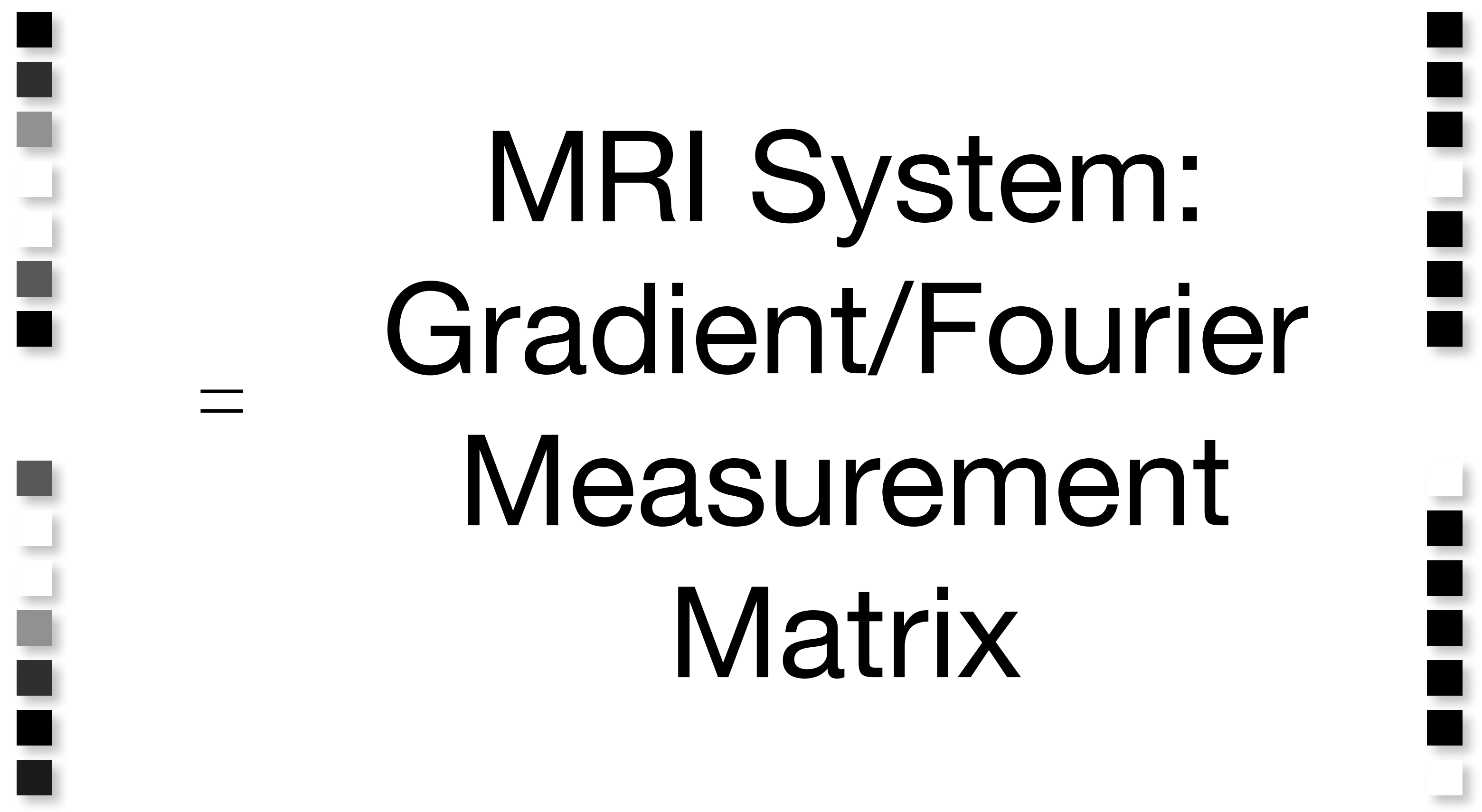
- Image data are vectors in an mn-dimensional vector space
- Consider the data in conventional [mn×1] vector form
- We know exactly how to map the data vector to an “image”



- To find the discrete image, we need to solve for this vector

Image Encoding (Measurement)

- MRI measurements are encoded via magnetic field gradients
- Gradients impose sinusoidal phase variation prior to integration
- This process is linear, and physically encodes a Fourier transform



$$k = E \quad x$$

- Our measurement can be described as a linear system

Imaging as a Linear System

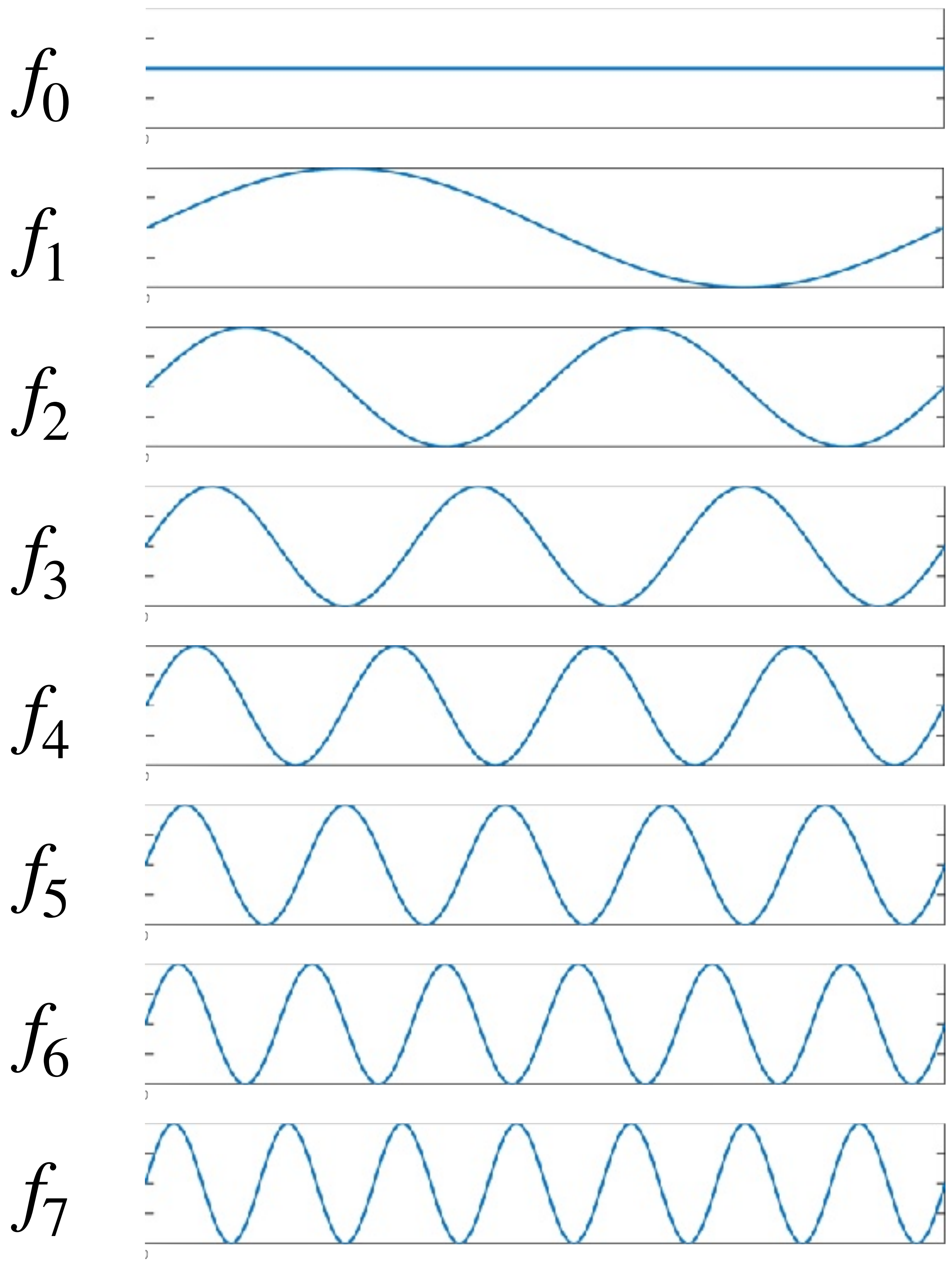
- Imaging becomes a simple matter of solving:

$$k = Ex$$

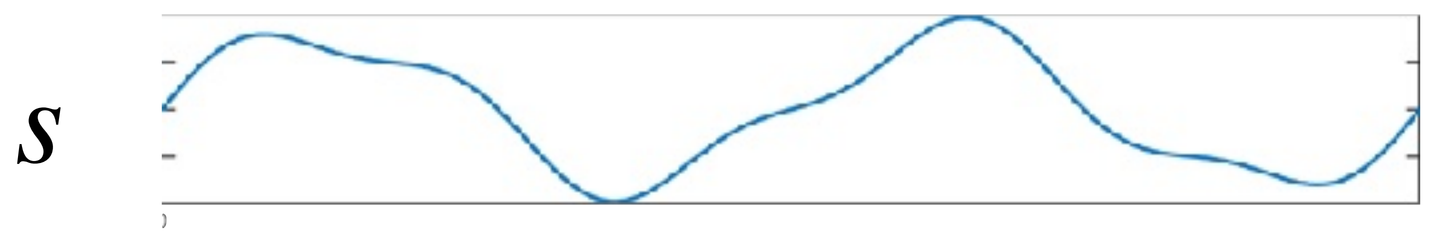
- where k is some vector of measured k-space values, E is a Fourier encoding matrix (or measurement transform) modelling the action of the MRI hardware, and x is the unknown image
- Abstraction of the imaging problem offers flexibility and utility afforded by existing mathematics
- Much of the discussion in the rest of this lecture regards the design of the linear measurement transform (matrix) E, and/or how to solve for x when E is not easily invertible
- No way to directly probe the values of each voxel (E is constrained by what is physically realizable by the MRI system)

The Fourier Basis

Fourier Basis Set



Signal



The Fourier Basis

Basis Coefficients

$$\sum f_0 \cdot s$$

$$\sum f_1 \cdot s$$

$$\sum f_2 \cdot s$$

$$\sum f_3 \cdot s$$

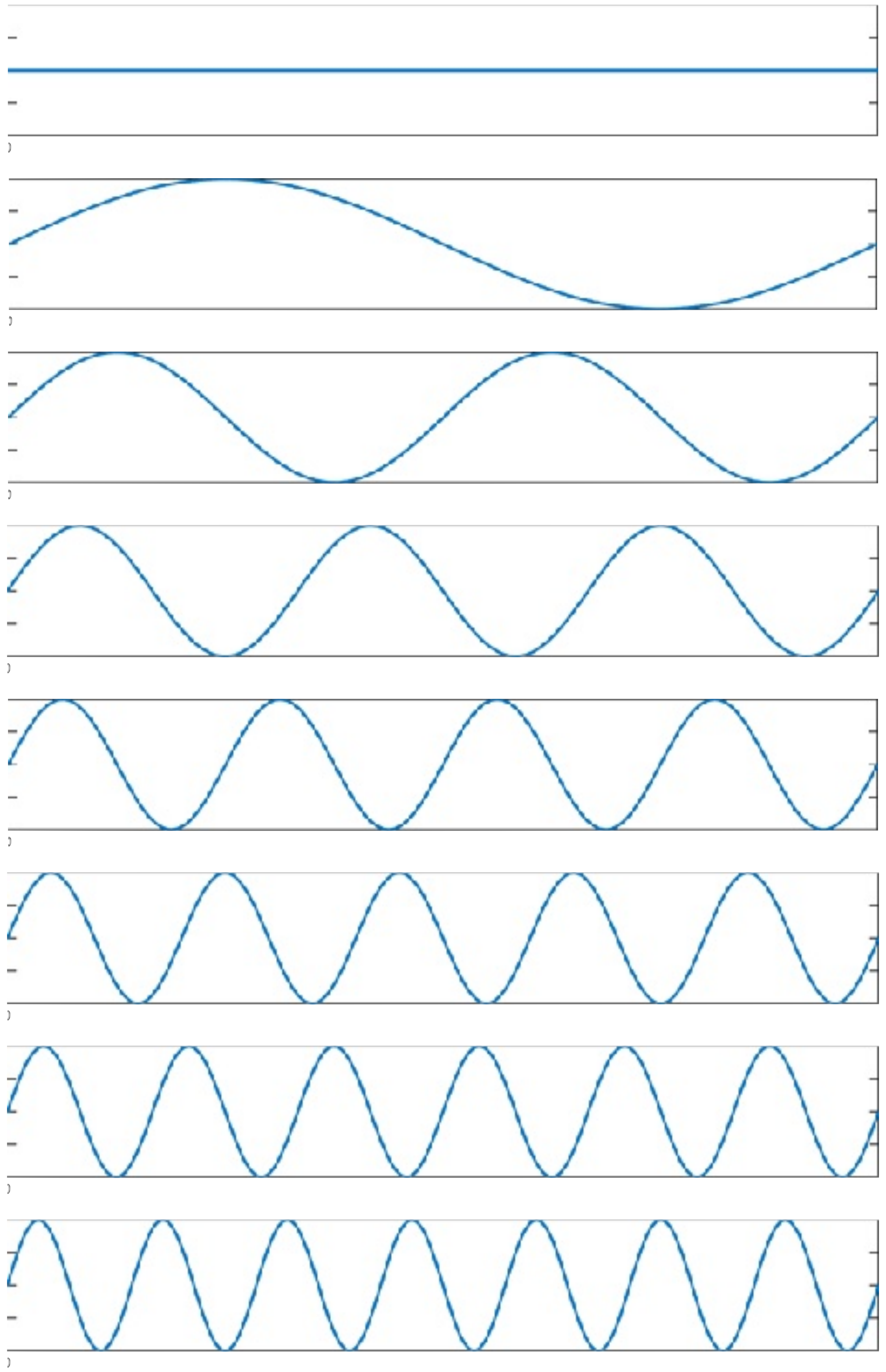
$$\sum f_4 \cdot s$$

$$\sum f_5 \cdot s$$

$$\sum f_6 \cdot s$$

$$\sum f_7 \cdot s$$

Fourier Basis Set



Signal

s

The Fourier Basis

Basis Coefficients

$0.31 - 1.20i$

$2.19 + 0.98i$

$1.06 + 2.24i$

$-0.73 + 1.22i$

$-2.94 - 1.65i$

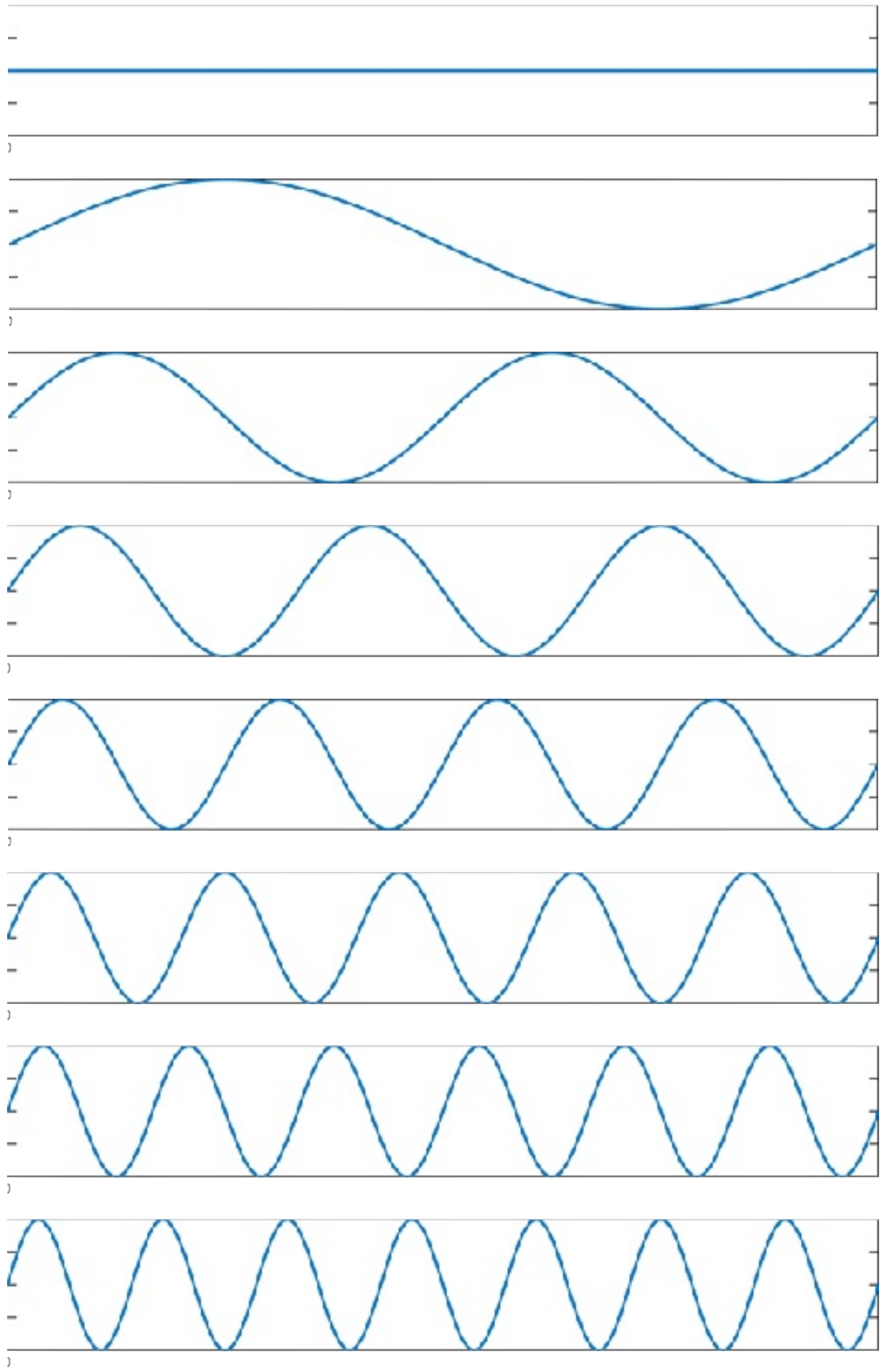
$1.49 - 2.71i$

$0.88 + 0.14i$

$-1.27 + 1.73i$

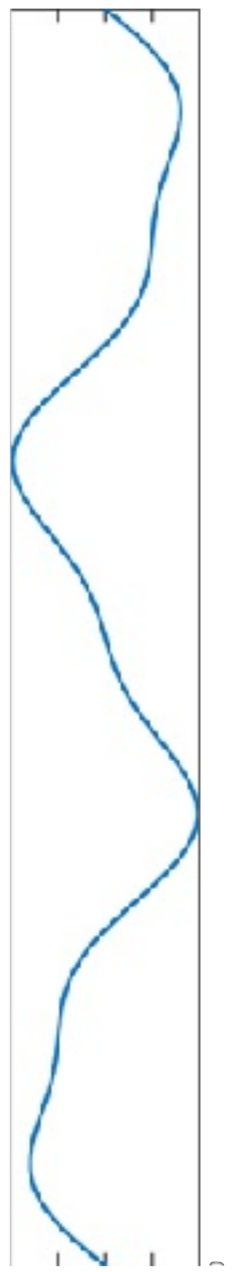
$=$

Fourier Basis Set



$*$

Signal

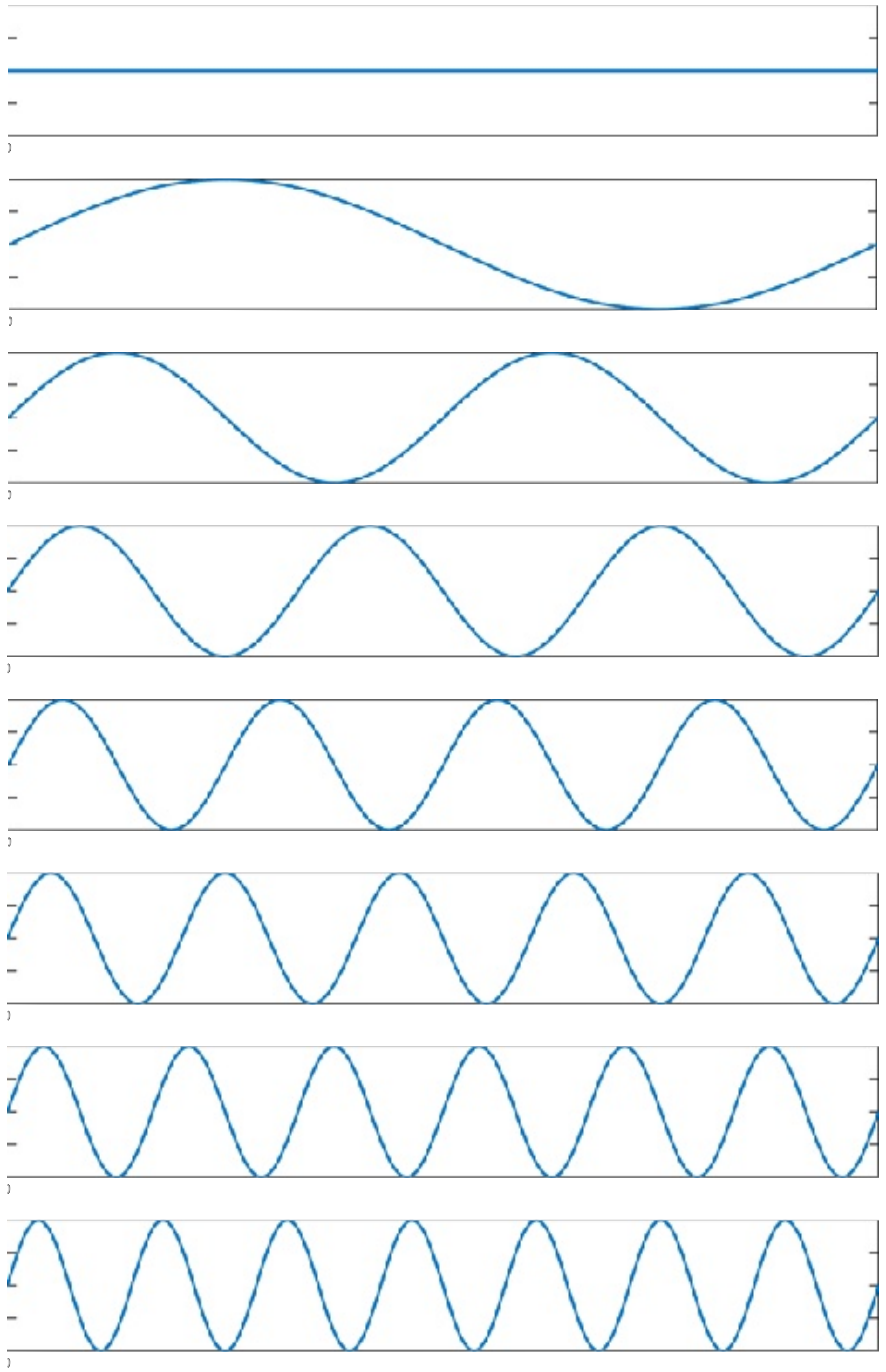


The Fourier Basis

Basis Coefficients

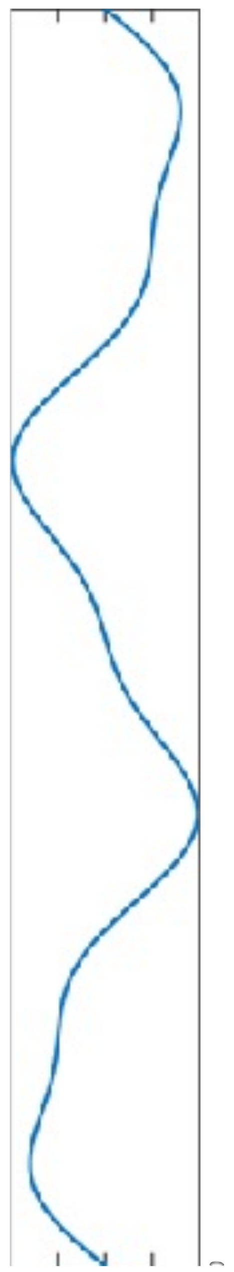
$$\begin{bmatrix} 0.31 - 1.20i \\ 2.19 + 0.98i \\ 1.06 + 2.24i \\ -0.73 + 1.22i \\ -2.94 - 1.65i \\ 1.49 - 2.71i \\ 0.88 + 0.14i \\ -1.27 + 1.73i \end{bmatrix} =$$

Fourier Basis Set



$*$

Signal

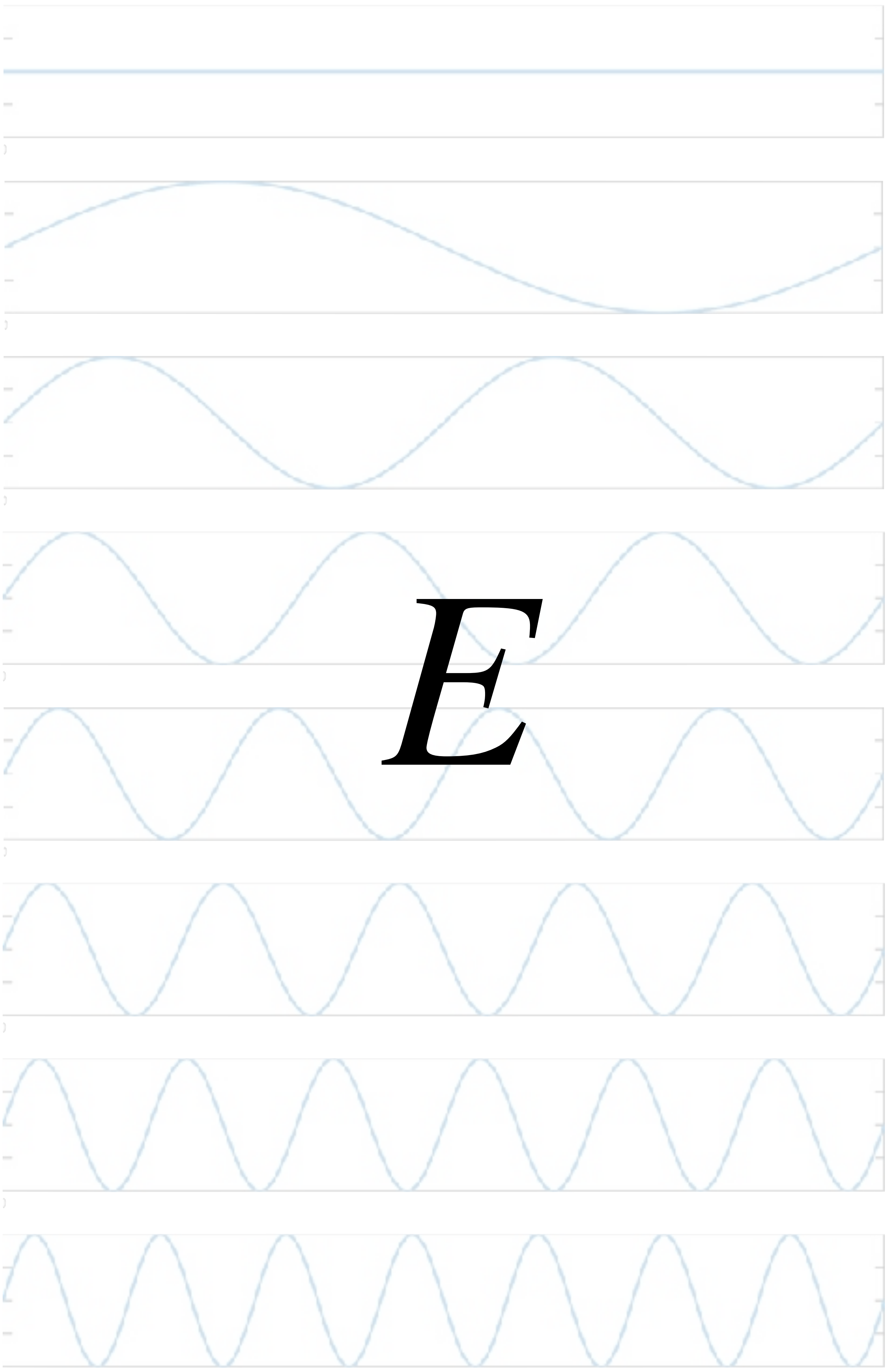


The Fourier Basis

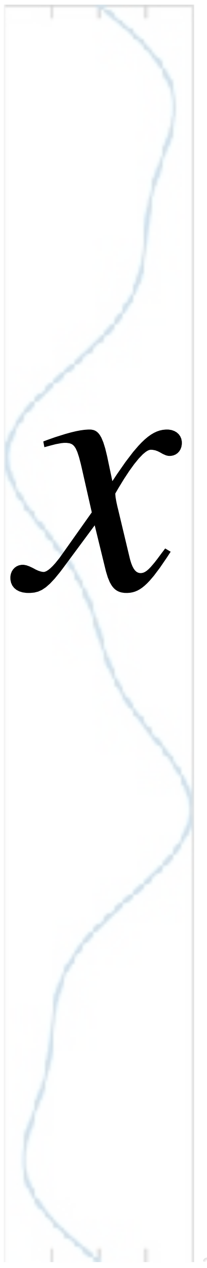
Basis Coefficients

$$\begin{bmatrix} 0.31 - 1.20i \\ 2.19 + 0.98i \\ 1.06 + 2.24i \\ -0.73 + 1.22i \\ -2.94 - 1.65i \\ 1.49 - 2.71i \\ 0.88 + 0.14i \\ -1.27 + 1.73i \end{bmatrix}$$

Fourier Basis Set



Signal



$$k = E^* * x$$

The Inverse Problem

$$k = Ex$$

- Forward problem: "Given E and x, generate k"
- Inverse problem: "Given E and k, what should x be?"

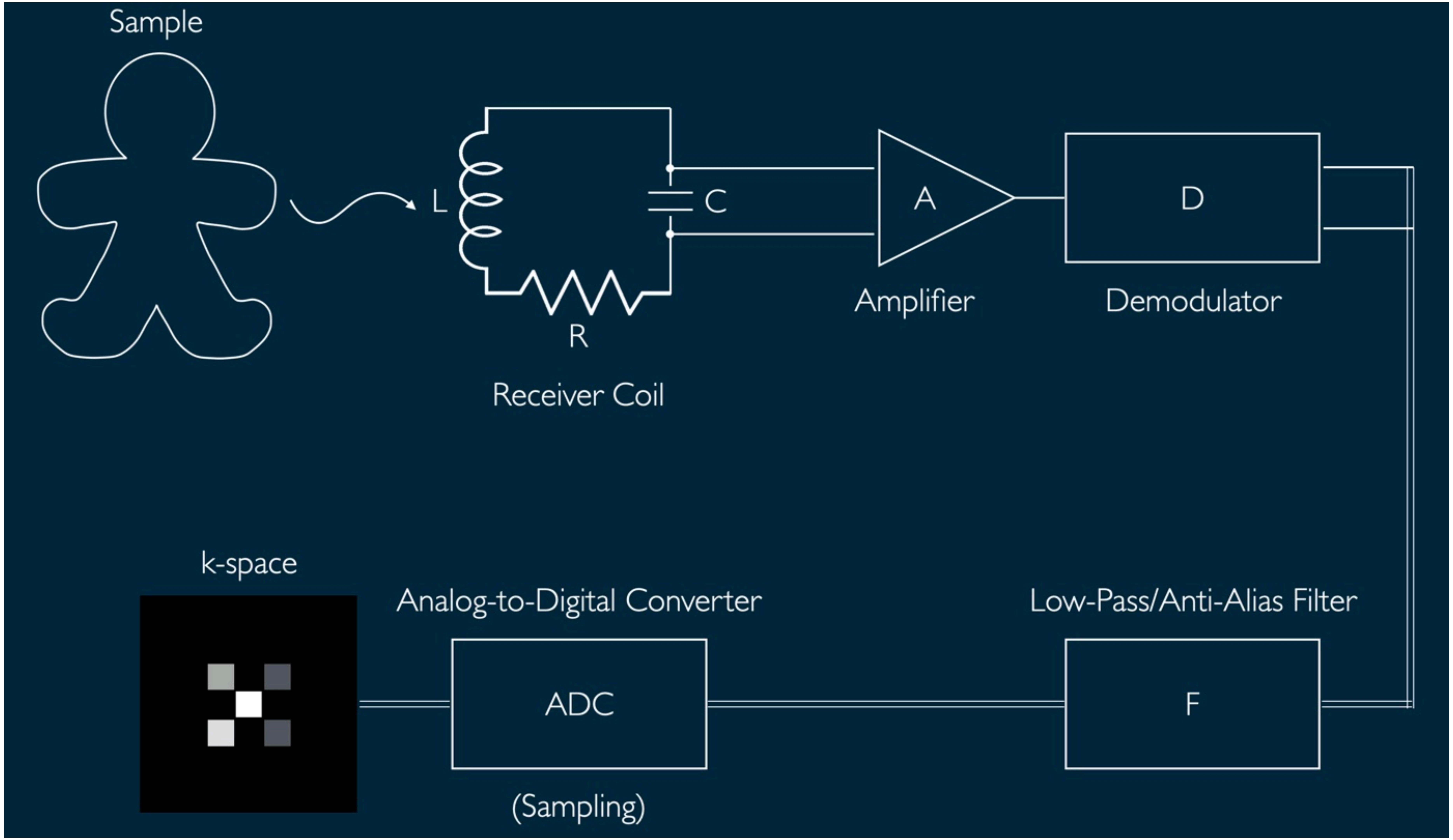
Solving the Inverse Problem

Image Reconstruction

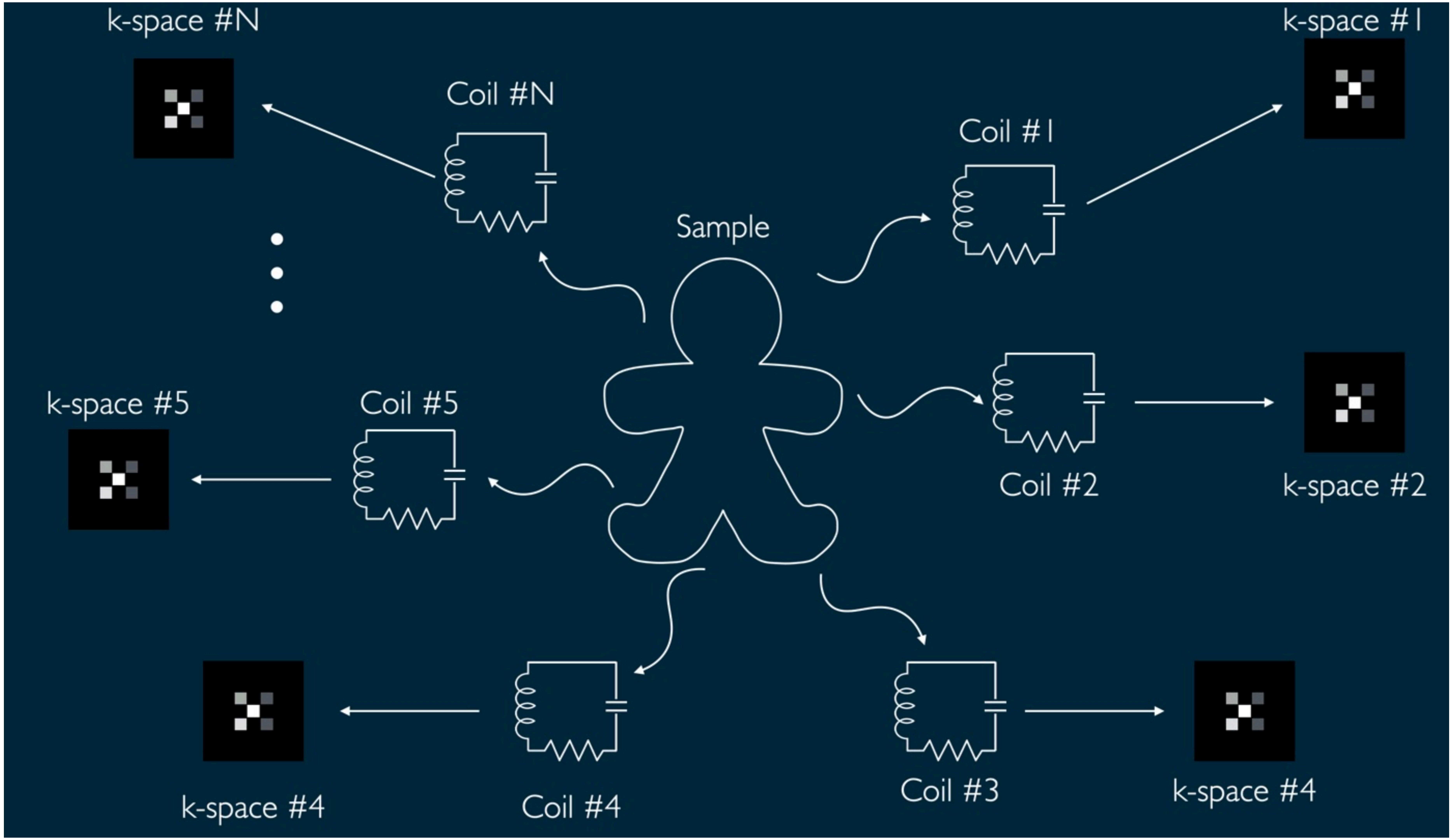
$$\hat{x} = E^{-1}k$$

- “Apply the inverse Fourier transform to the k-space data”
- We know that the inverse Fourier transform means the inverse or conjugate transpose of the (unitary) Fourier transform matrix
- A square matrix is necessary (but not sufficient) for invertibility
  - Square means equal numbers of measurements and unknowns
- Need as many k-space samples as there are image points
  - If this condition is not satisfied, how do we reconstruct the image x?

MRI Receiver Chain



Parallel Receivers



Measurement Model

- Single channel measurement model

$$s(k) = \int x(r)e^{-i2\pi k\cdot r}dr$$

- Multi channel measurement model

$$s_j(k) = \int c_j(x)x(r)e^{-i2\pi k\cdot r}dr$$

- Discrete multi channel measurement model

$$s_j(k) = \sum_l c_j(r_l)x(r_l)e^{-i2\pi k\cdot r_l}$$

$$s_j = EC_jx$$

Measurement Model

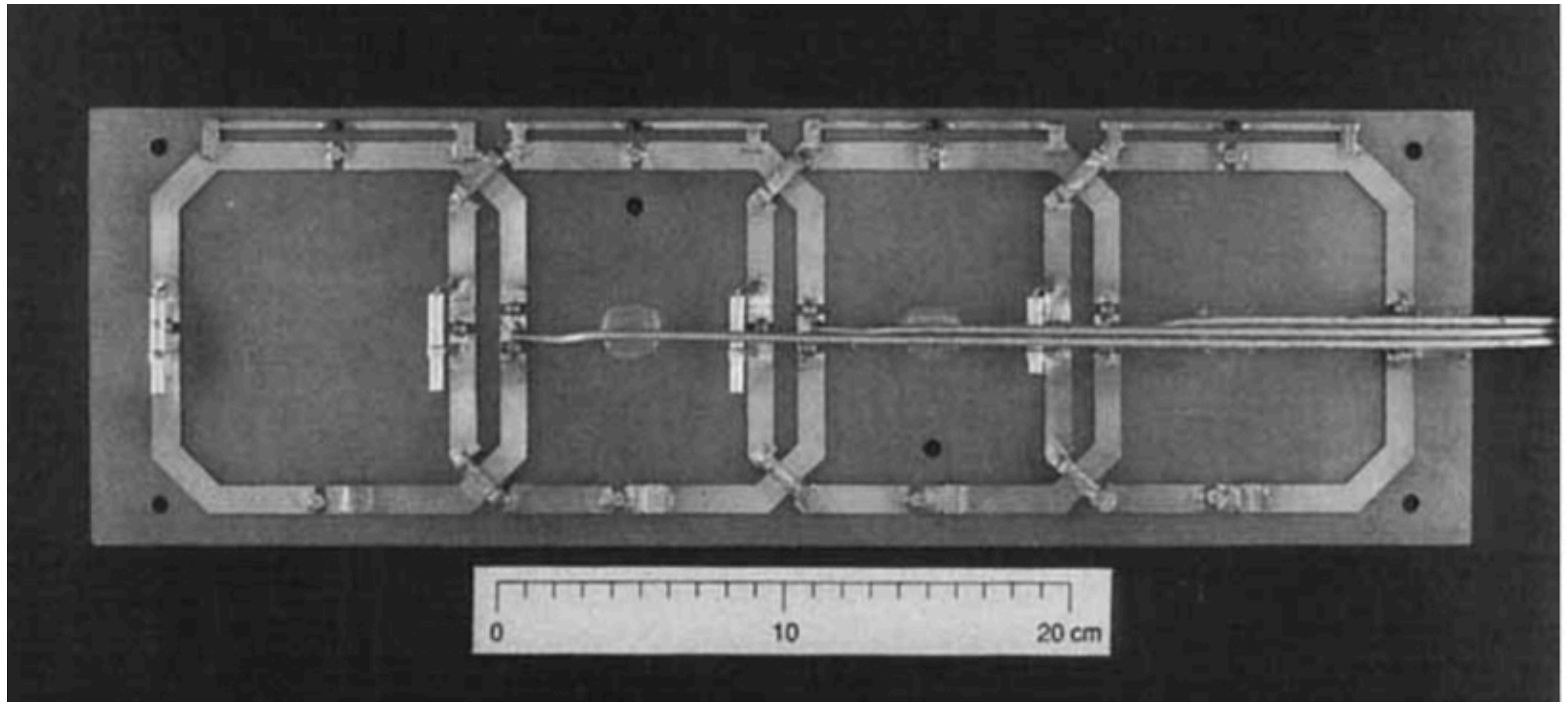
- Each “channel” measures a weighted version of the underlying image

$$\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix} = \begin{bmatrix} E & & & \\ & E & & \\ & & \ddots & \\ & & & E \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix} [x]$$

Part II – Sensitivities & Pre-processing

Example Coil Sensitivities

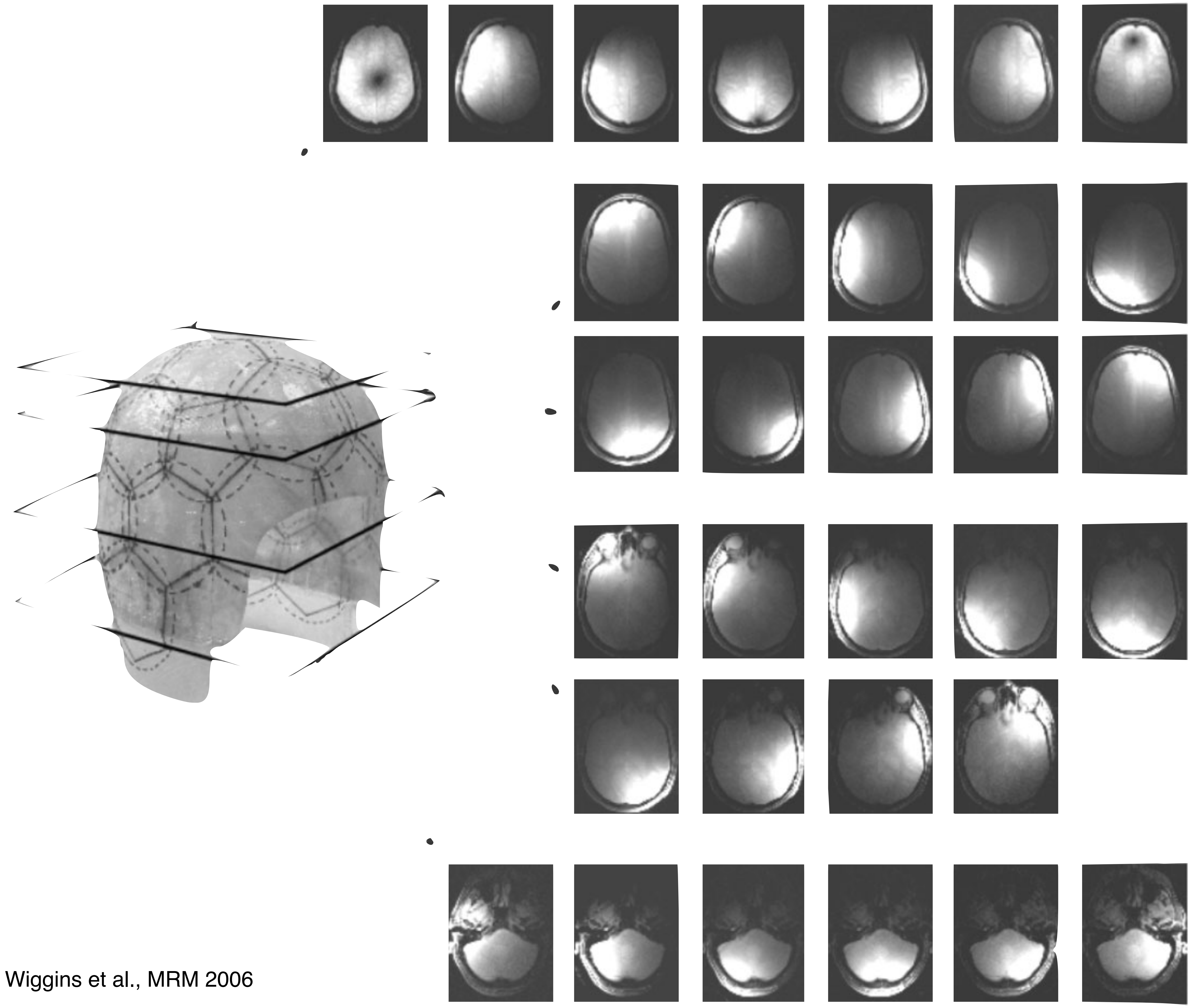
4-channel spine array



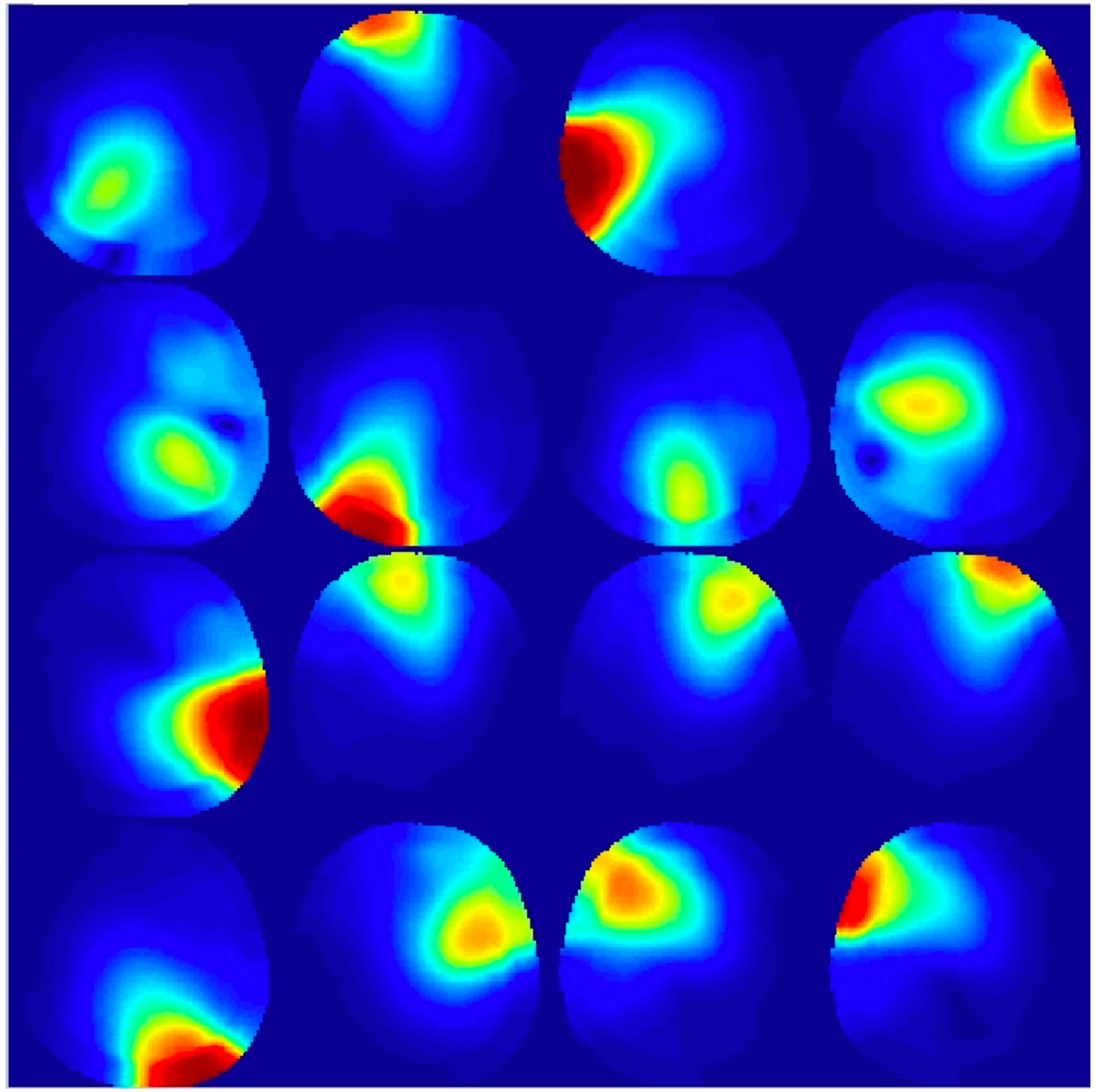
Roemer et al., MRM 1990

Example Coil Sensitivities

32-channel head array coil



Measured sensitivities



Wiggins et al., MRM 2006

What is a coil sensitivity?

- We interact with receive coils via Faraday induction
- EMF in the coil due to changing magnetic flux
- Coil is fixed, so flux change is due to magnetic field change
- Magnetic field comes from the net magnetization at each voxel
- Now consider each voxel’s magnetization to be a unit magnetic dipole
- Compute the flux contribution from that voxel source only
- That is a reflection of the sensitivity of the coil at that voxel’s location in space

Sensitivity Map Estimation

Direct method (not recommended)

- If the images associated with each coil are just weighted versions of the underlying magnetization, then we can try to simply divide each coil image by some reference image

$$x_j = C_j x$$

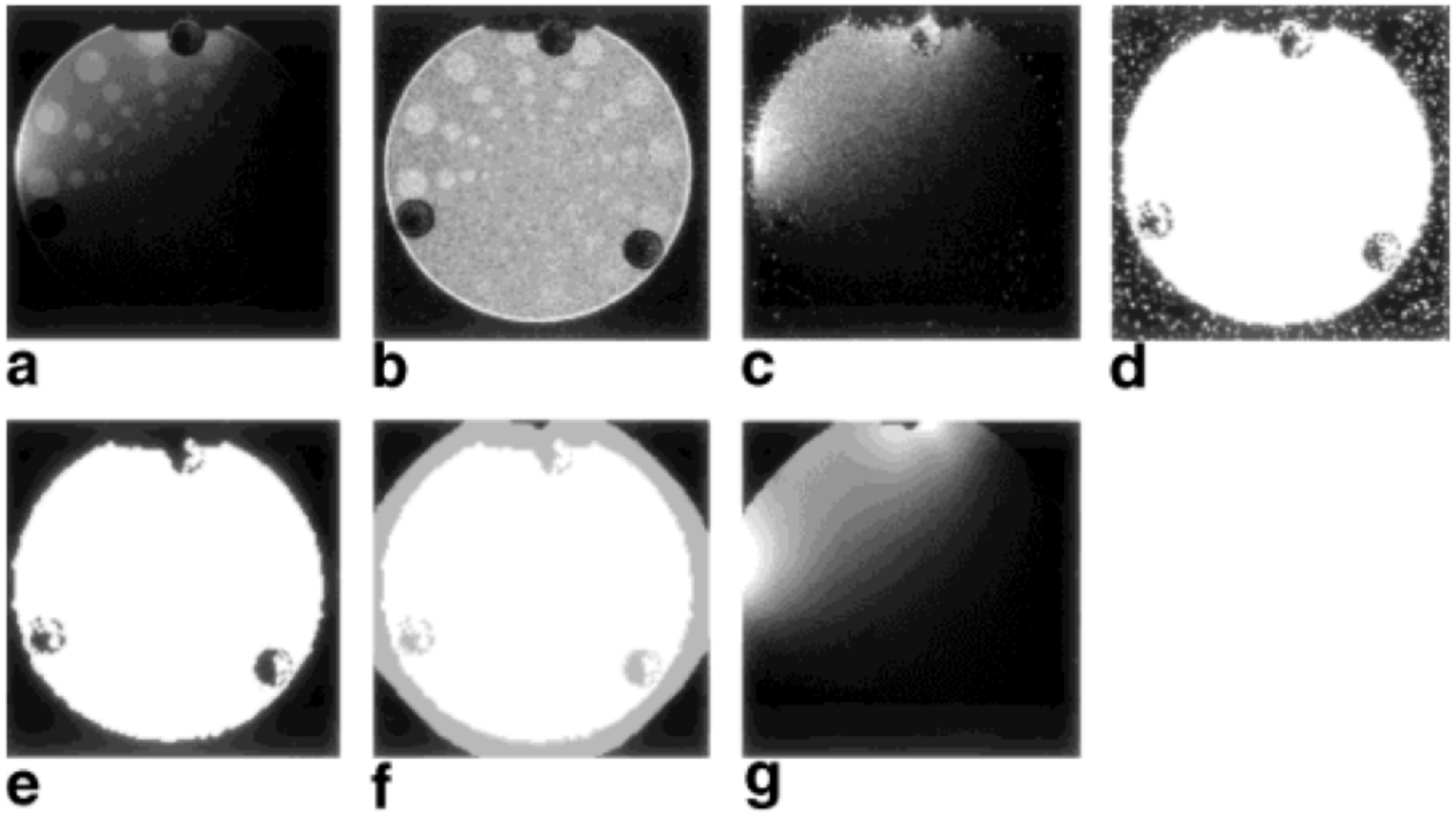
$$\hat{C}_j = x_j / x_{ref}$$

- Reference can be acquired with a body-coil (flat sensitivity) image
- Or a relative reference can be generated (e.g. sum-of-squares $x_{ref} = \sqrt{\sum_j x_j^2}$

Sensitivity Map Estimation

Direct method (not recommended)

- However, this is fairly un-reliable, prone to bias due to noise, etc.
- Also typically requires fitting some spatial model (e.g. low-order polynomial)



Pruessmann et al., MRM 1999

Sensitivity Map Estimation

Adaptive Combine Method (Walsh, MRM 2000)

- Uses the idea of sensitivities as optimal “matched filter” coefficients
- The idea is that the sensitivities are also the optimal coil combination factors
- To find these coefficients, compute the first eigenvector of:

$$\Sigma_n^{-1}\Sigma_s$$

  where Σ_n is the noise covariance matrix, and Σ_s is the signal covariance

- Estimate Σ_s from a small region-of-interest, assuming the sensitivities don’t vary rapidly over space

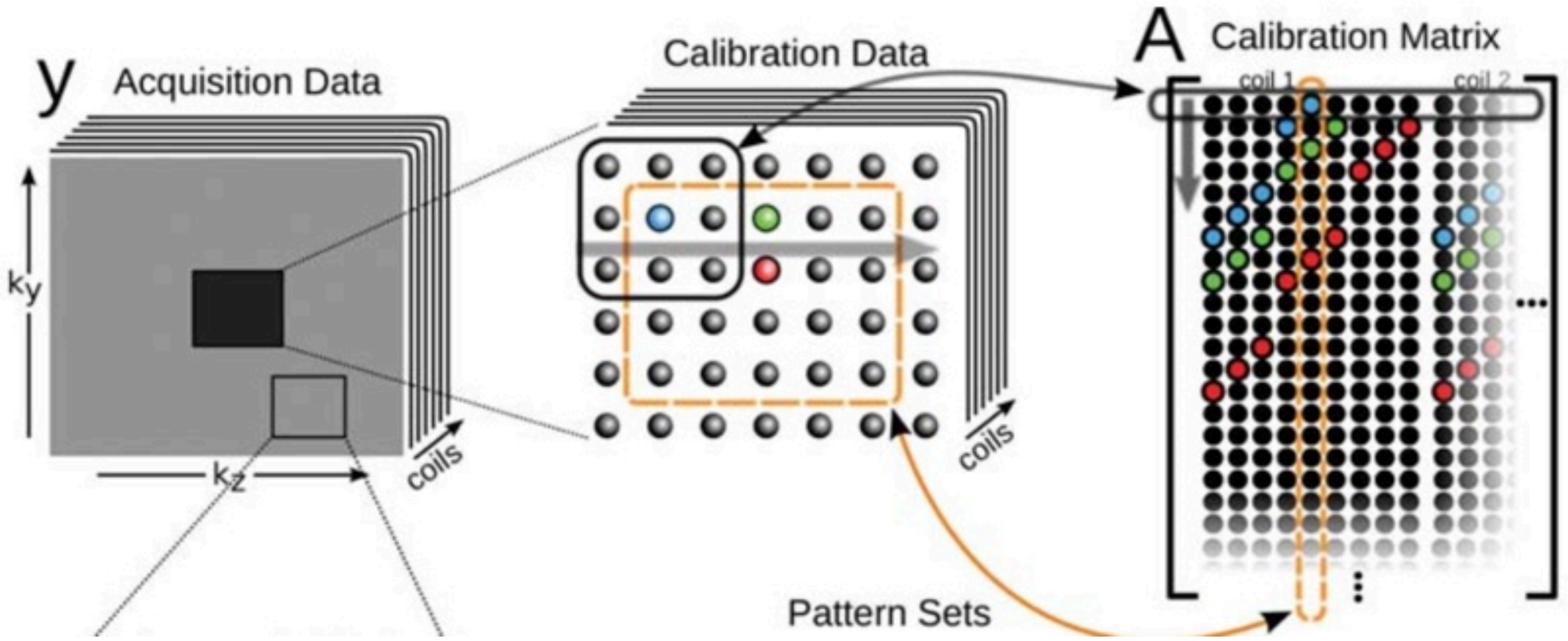
Sensitivity Map Estimation
Adaptive Combine Method (Walsh, MRM 2000)

- Take a small image patch, e.g. 5×5 voxels, extract the signal from each channel to get an $N_C \times 25$ matrix M
- Compute $\Sigma_S = MM^H$
- Compute the eigendecomposition $\Sigma_S = VDV^H$
- Find the $N_C \times 1$ column of V associated with the maximum eigenvalue
- This is the vector of sensitivities for the location associated with that patch
  - Can either assign to whole patch, or the centre of the patch
- Repeat for all image patches

Sensitivity Map Estimation

ESPIRiT (Uecker et al., MRM 2014)

- Also involves identifying eigenvectors, but coming from a different problem
- The underlying idea is that the k-space calibration matrix (i.e. a matrix constructed by concatenating vectorized k-space patches across coils) is redundant (low rank, with linear dependencies, non-trivial null-space)

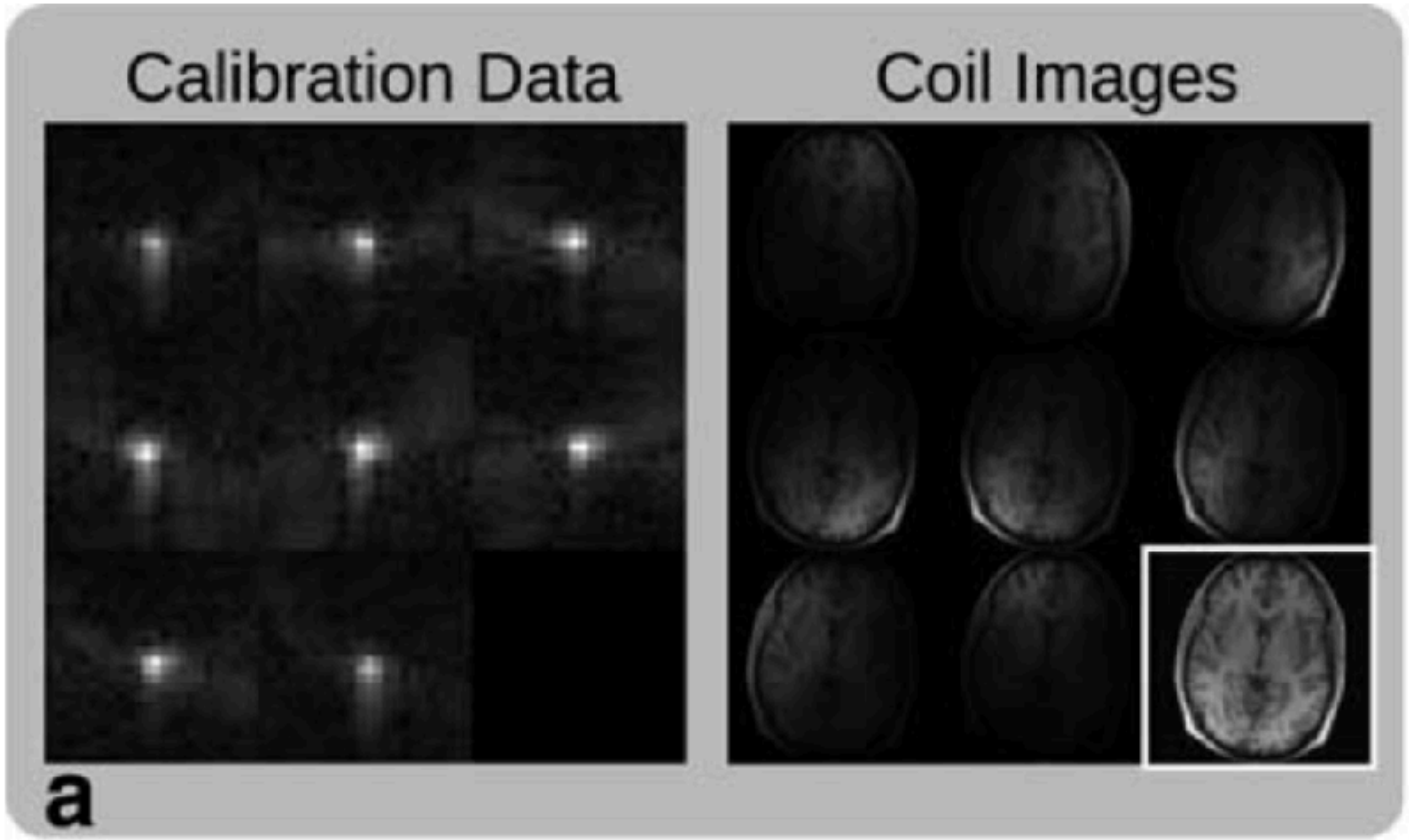


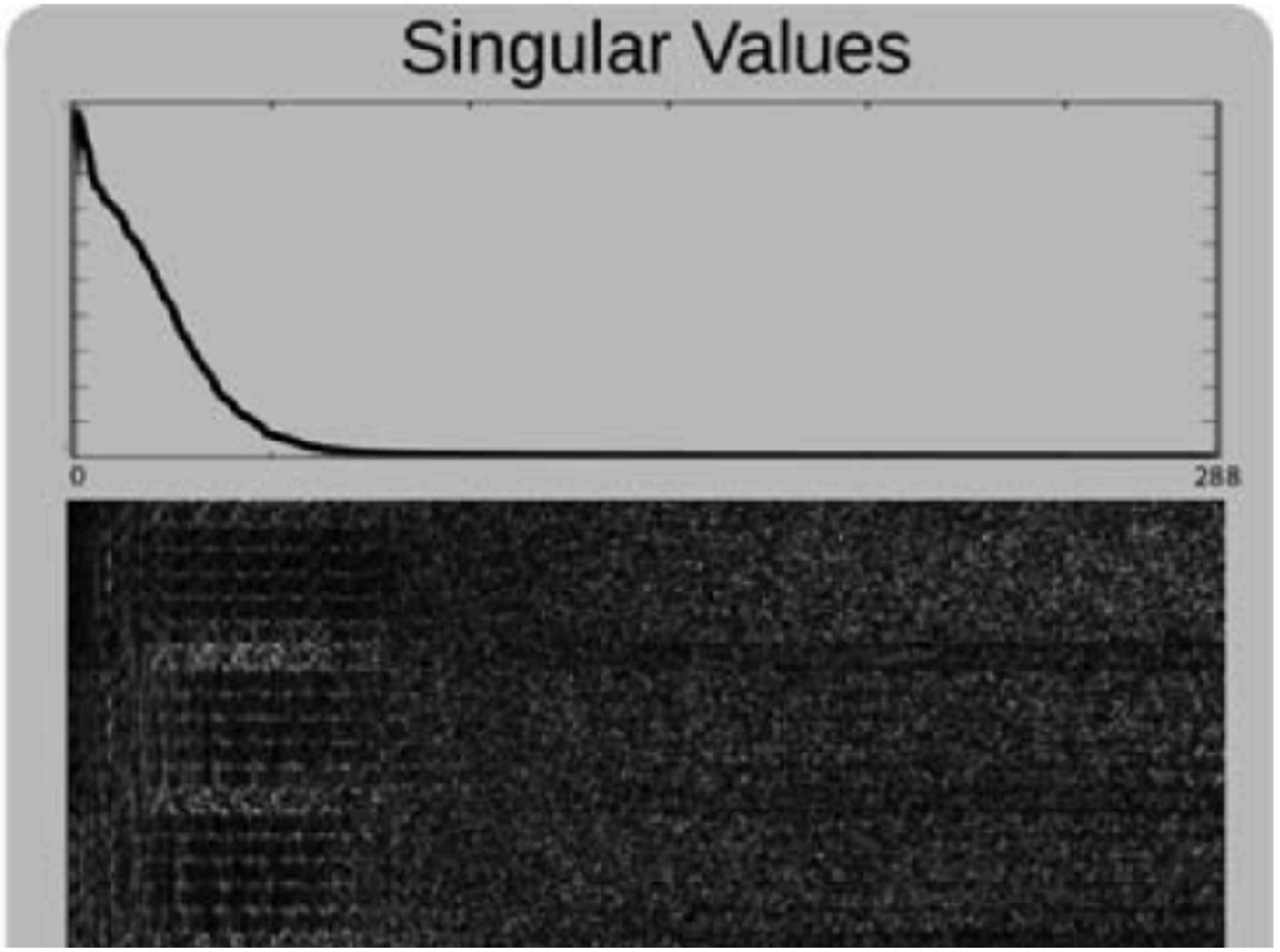
Sensitivity Map Estimation

ESPIRiT (Uecker et al., MRM 2014)

- We can therefore define a non-trivial projection operator $W = VV^H$, where V is the multi-channel subspace

- If the signal k lives in the subspace defined by V, then it must satisfy $Wk = k$





Uecker et al., MRM 2014

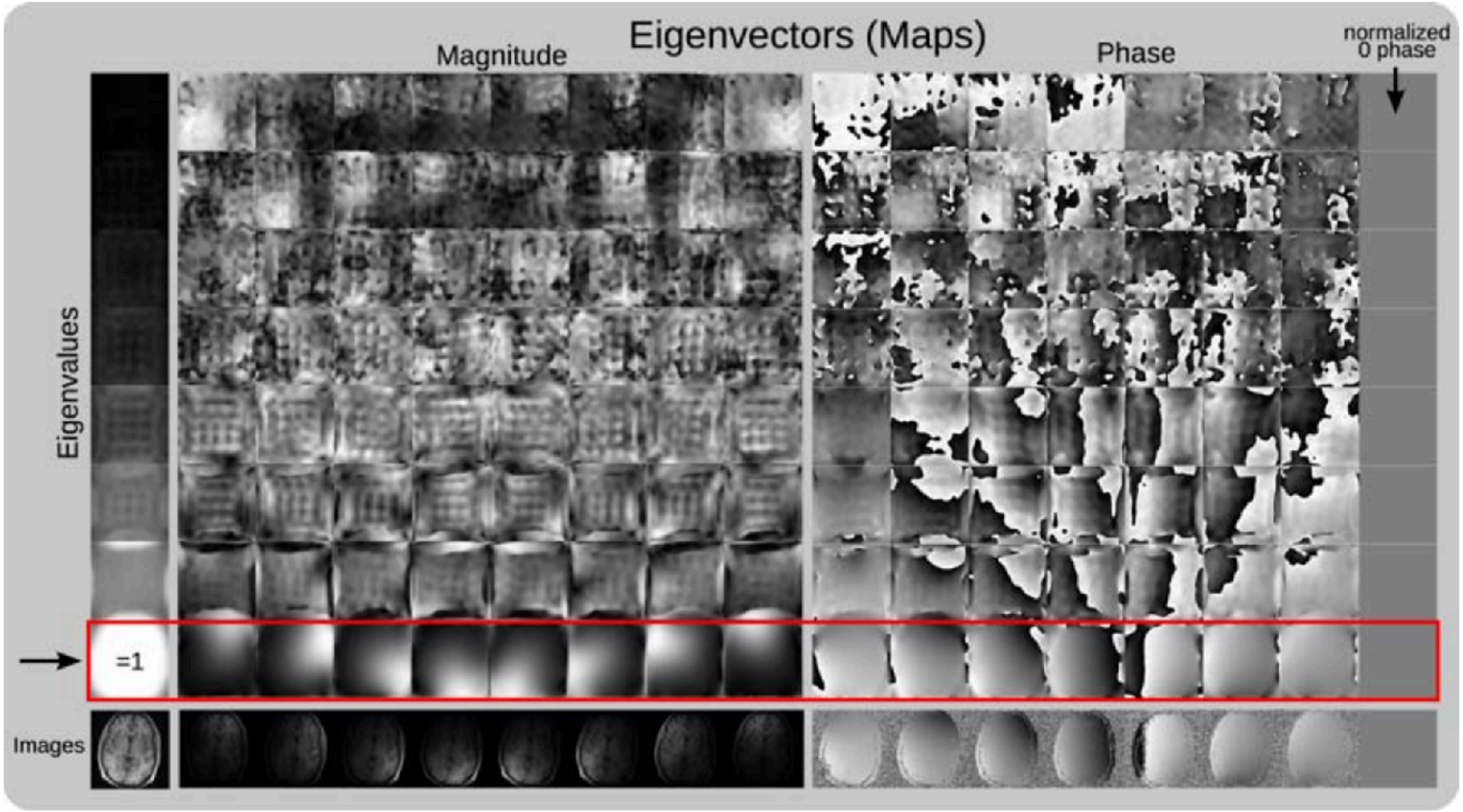
Sensitivity Map Estimation

ESPIRiT (Uecker et al., MRM 2014)

- Then, if we let $x = \mathscr{F} \cdot s \cdot m$, where $\mathscr{F}$ is the Fourier Transform, s are the sensitivities, and m is the magnetization, we have $W\mathscr{F}sm = \mathscr{F}sm$
- By taking the inverse Fourier transform on both sides, we get $\mathscr{F}^{-1}W\mathscr{F}sm = sm$
- Everywhere where m is non-zero, we have $(\mathscr{F}^{-1}W\mathscr{F})s = s$
- Therefore, the coil sensitivities are encoded here as the eigenvectors of $\mathscr{F}^{-1}W\mathscr{F}$ corresponding to eigenvalue 1

Sensitivity Map Estimation

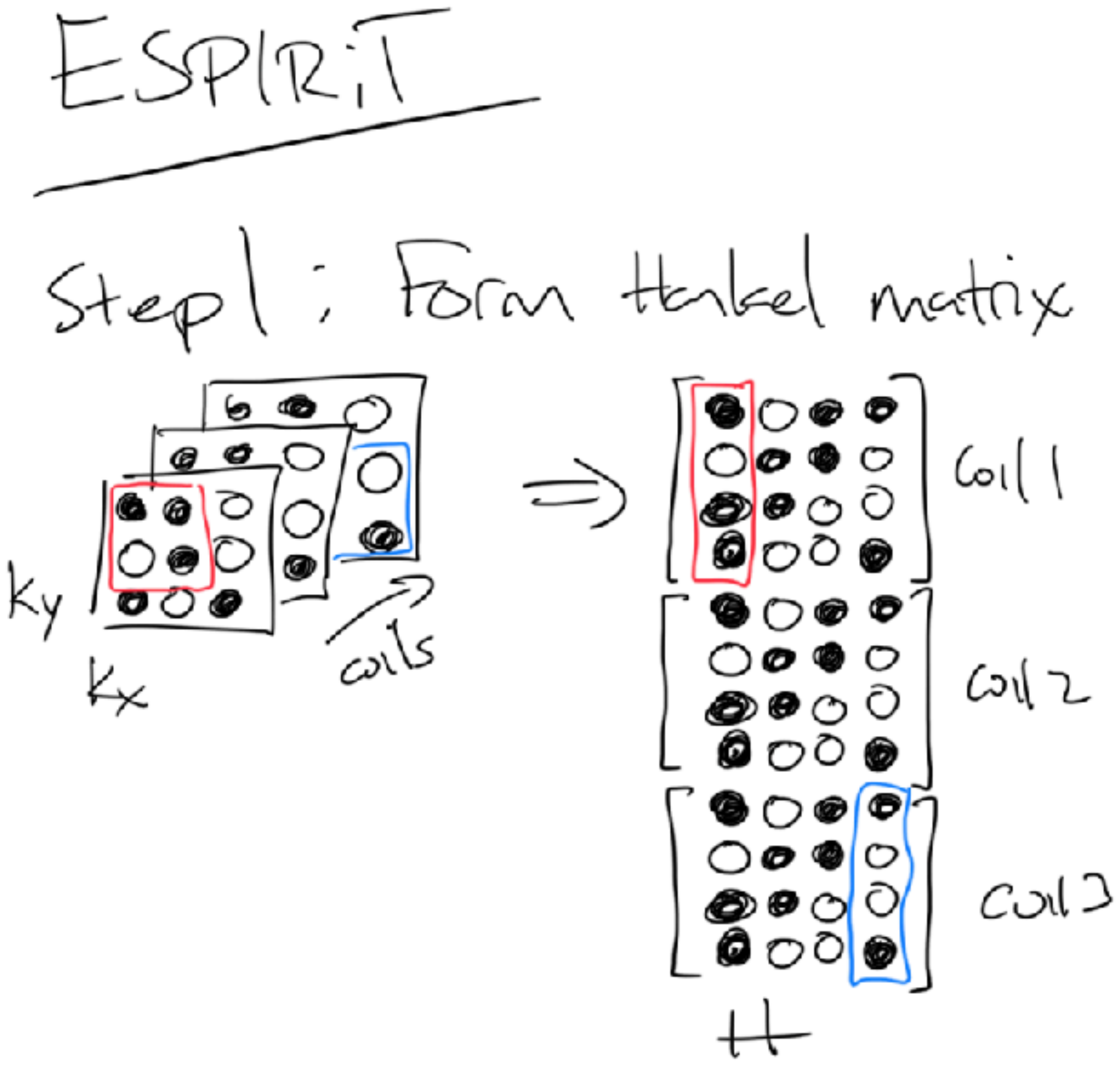
ESPIRiT (Uecker et al., MRM 2014)

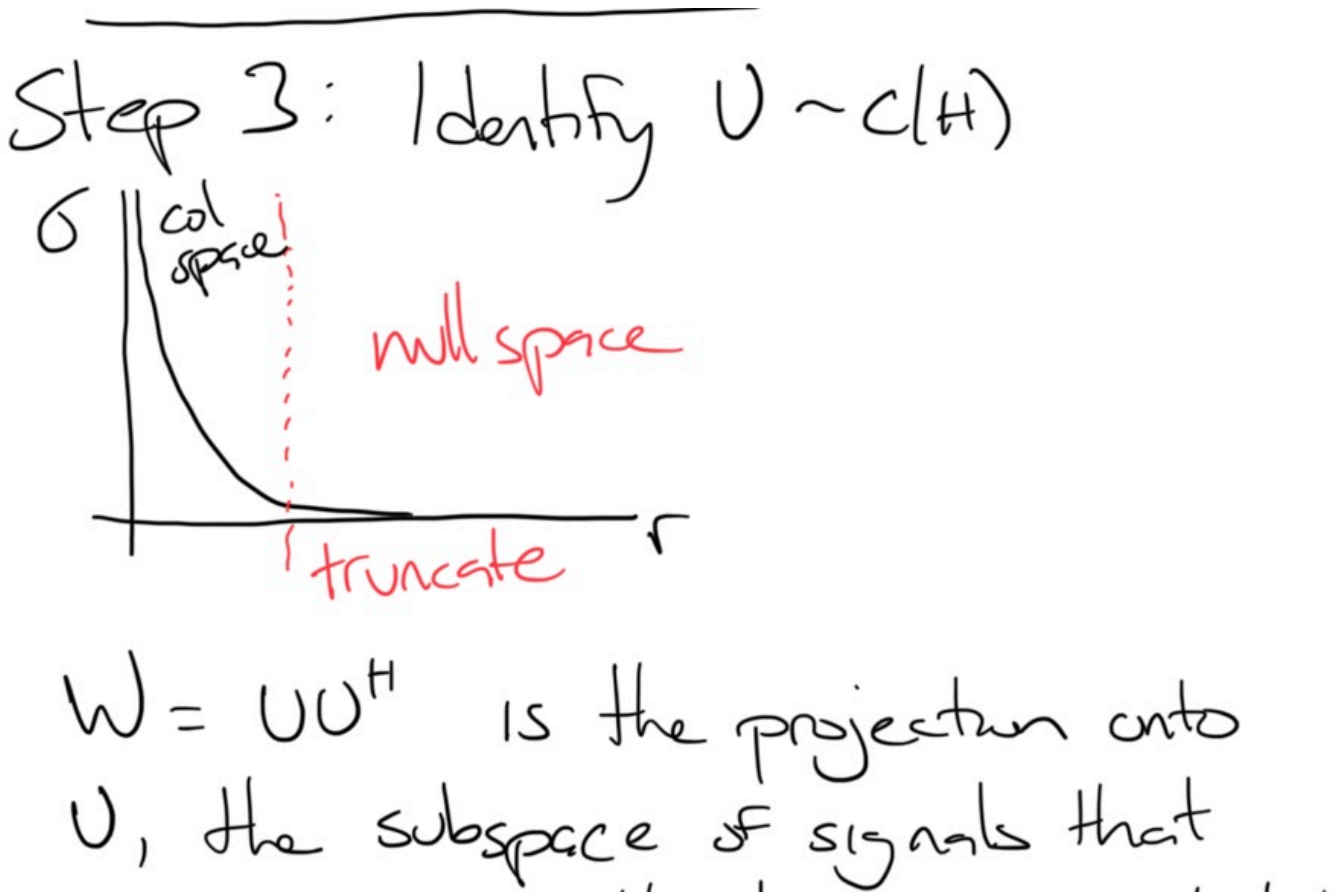


Uecker et al., MRM 2014

Sensitivity Map Estimation

https://github.com/mchiew/teaching-lectures/blob/main/ESPIRiT.pdf

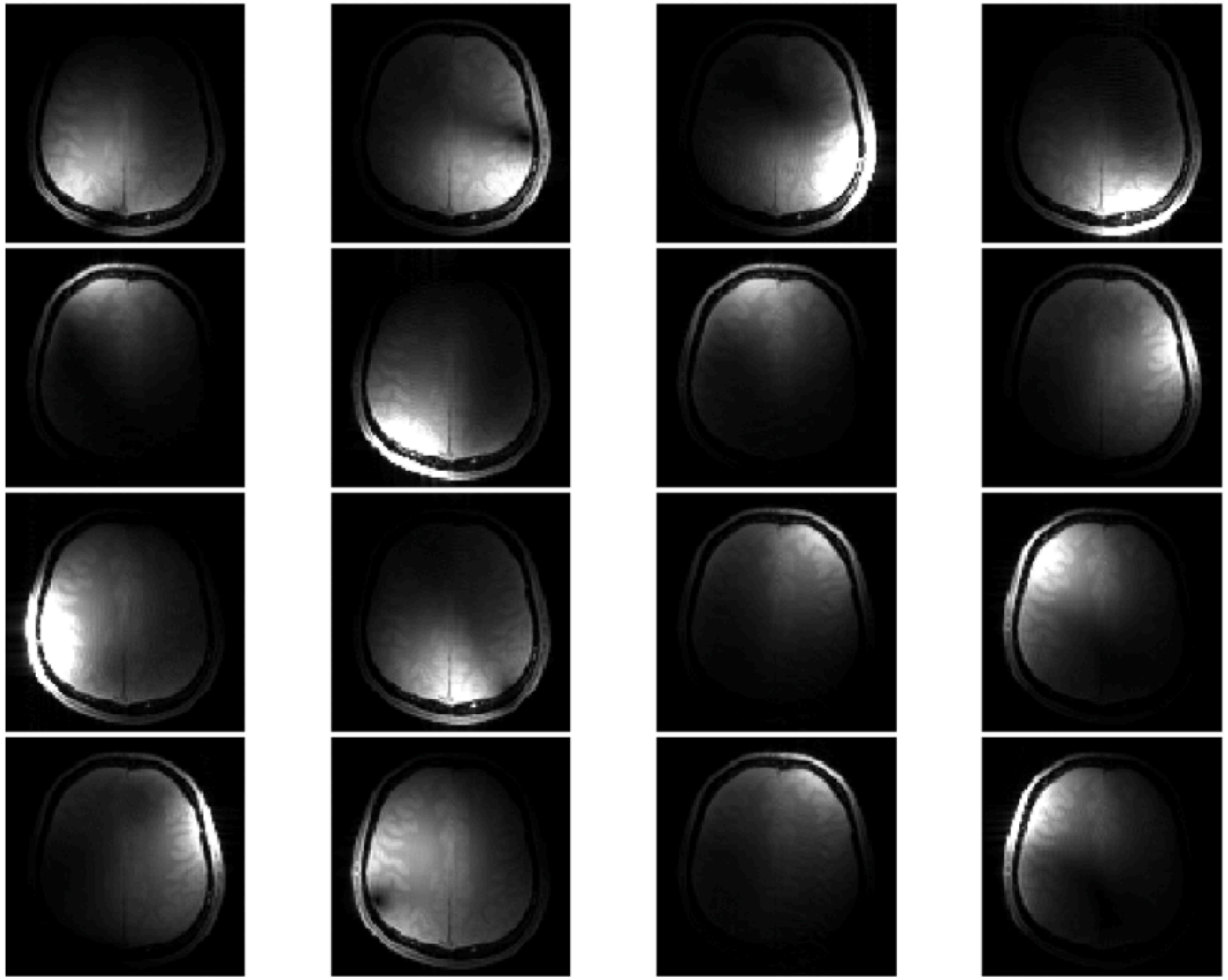




Uecker et al., MRM 2014

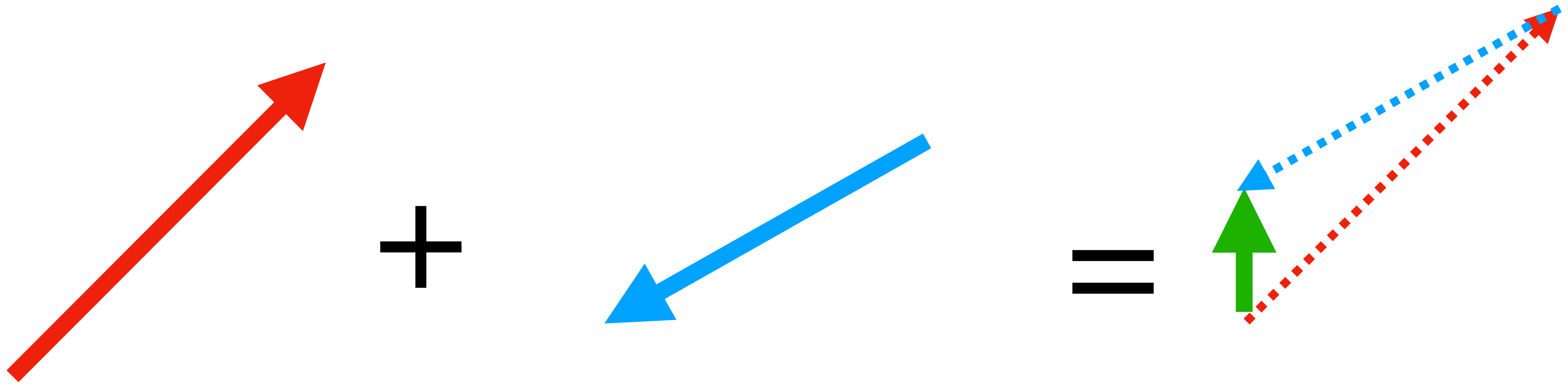
Channel Combination

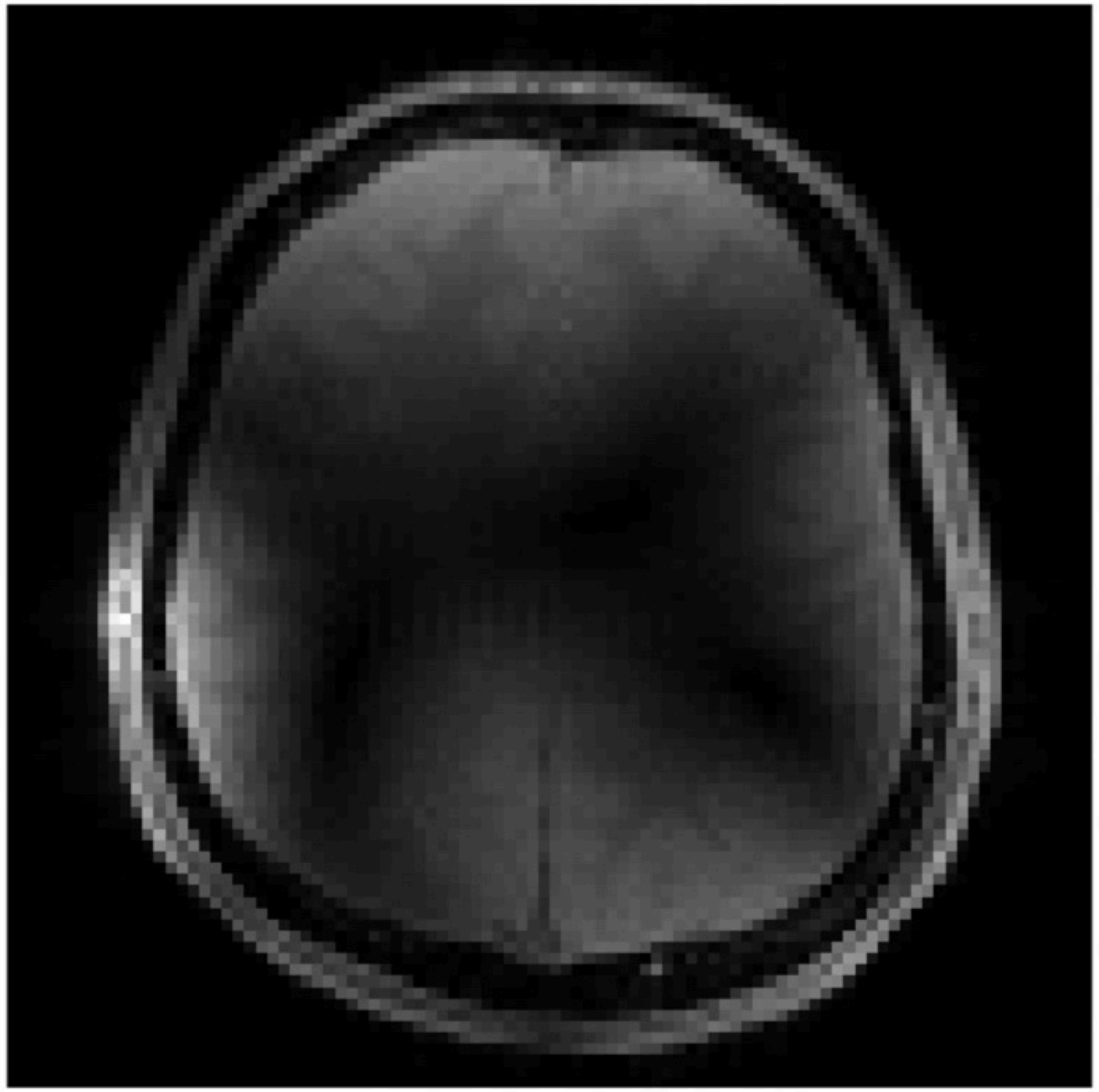
- We acquire multi-channel images with different magnitude and phase weighting across space
- How do be turn this into a single composite image?
- What if we just add/average them together?



Channel Combination

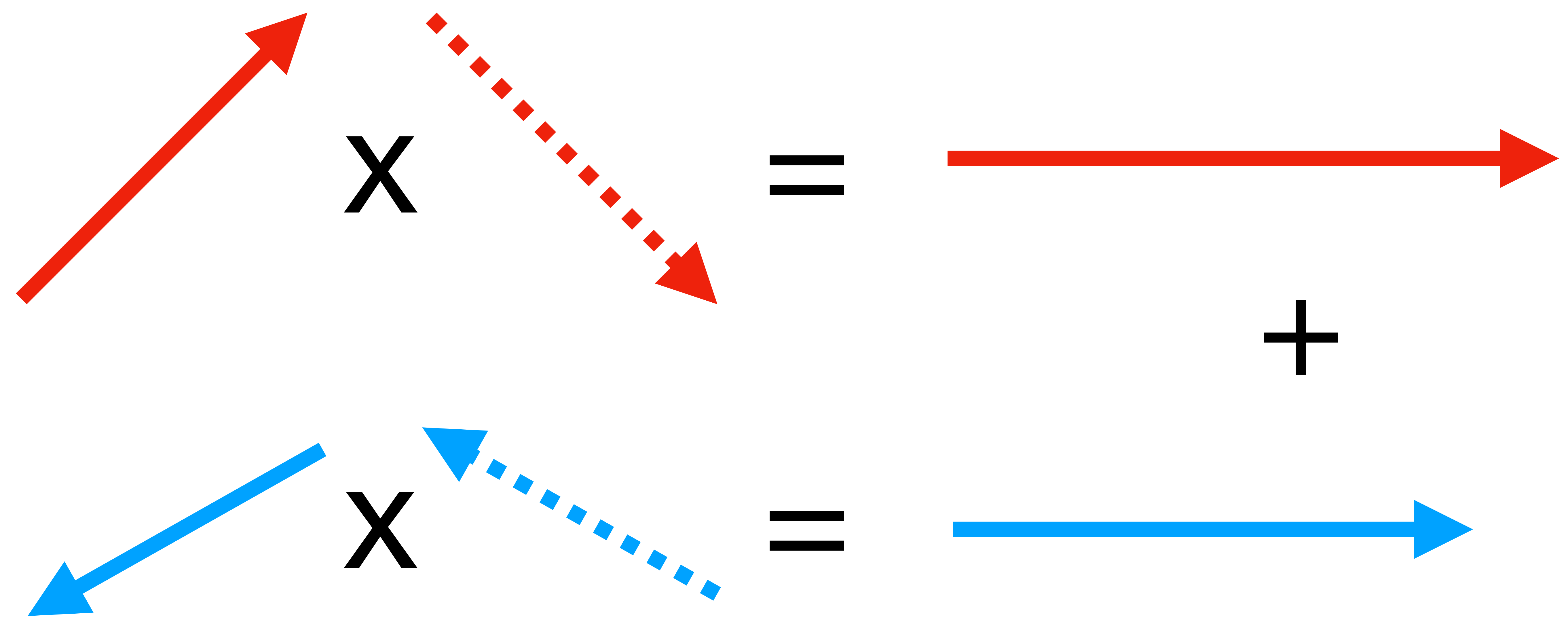
- If you just try to average the multi-channel data, you get a terrible image
- The problem is that sensitivity weighting is complex, so simple adding/averaging leads to phase cancellation





Channel Combination

- Incidentally, this is why multi-channel receive coils are sometimes referred to as “phased arrays”
- Each channel needs to have different phase modulation to get a coherent output signal (image)
- Optimal weight is the conjugate of the sensitivity



Channel Combination

Complex combination

- Formally, we need to find some weights w_j to perform the weighted combination ($w_j = 1$ for all j corresponds to sum/averaging)
- If $x_j(r) = C_j(r)x(r)$, then $x_{comb} = \sum_j w_j(r)x_j(r)$
- Ideally, we want $x_{comb} = x$, so the equation for the w_j becomes:

$$\sum_j w_j C_j x = x \implies \sum_j w_j C_j = 1$$

Channel Combination

Complex combination

- $\sum_{j}^{N_c} w_j C_j = 1$ has infinitely many possible solutions
- A natural definition is $w_j = \dfrac{1}{N_c C_j}$
- Another possible definition is $w_1 = \dfrac{1}{C_1}$ and $w_j = 0$ for $j = 2 \ldots N_c$

Channel Combination

Complex combination

- However, only one solution is SNR optimal:

$$w_j = \frac{C_j^*}{\sum_j C_j^* C_j}$$

- Try to prove this to yourself (it’s fairly straightforward)
- Intuition: noise is independent of coil weighting, so voxels with larger C_j will have higher SNR (longer signal vector relative to noise vector)

Channel Combination

Complex combination

- Compare the following two options:

$$w_j = \frac{1}{N_c C_j} \qquad\qquad w_j = \frac{C_j^*}{\sum_j C_j^* C_j}$$

- Both “undo” the phase, but the left one reduces the magnitude of large signals, whereas the one on the right squares the magnitude
- That is, on the left channels with high SNR are *downweighted,* whereas on the right channels with high SNR are *upweighted*

Channel Combination

Magnitude Combination (sum-of-squares)

- What if you don't know, or can't be bothered to estimated the sensitivities C_j
- The sum-of-squares combination requires no additional information, and weights the image *by itself*
- If $x_j = C_j x$, then define $w_j = x_j^*$
- Then $\sum_j x_j^* x_j = \sum_j C_j * C_j |x|^2$, which is close to what we want
- Taking the square root, we get: $x_{comb} = \sqrt{\sum_j C_j * C_j |x|^2} = \sqrt{\sum_j C_j * C_j} |x|$
- This is very close to SNR-optimal (~90%), is very robust, and requires no sensitivity mapping!

Extra

Derive SNR optimal weights

- Assume we have a 2-channel system, with noisy measurements such that:
- $x_1 = C_1 x + \sigma_1$ and $x_2 = C_2 x + \sigma_2$
- We can write this as $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x + \begin{bmatrix} \sigma_1 \\ \sigma_2 \end{bmatrix}$
- Let's assume $\sigma_{1,2} \sim N(0,\sigma)$
- We want to find $[w_1 \quad w_2]$ such that $[w_1 \quad w_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = w_1 x_1 + w_2 x_2$ is SNR optimal
- We want the combined image to be *unbiased*, so we need $w_1 C_1 + w_2 C_2 = 1$
- Then maximizing SNR is equivalent to minimizing the variance of $w_1 \sigma_1 + w_2 \sigma_2 \sim (w_1^2 + w_2^2)\sigma^2$

Extra

Derive SNR optimal weights

- So we want to minimize $w_1^2 + w_2^2$, subject to $w_1 C_1 + w_2 C_2 = 1$
- Solve for w_2: $w_2 = \frac{1 - w_1 C_1}{C_2}$, to get $f(w_1) = w_1^2 + \left(\frac{1 - w_1 C_1}{C_2}\right)^2$
- Take the derivative of f and set it to zero to find the optimum point:
- $\frac{df}{dw_1} = 2w_1 + 2\left(\frac{1 - w_1 C_1}{C_2}\right)\left(\frac{-C_1}{C_2}\right) = 2w_1 - 2\frac{C_1}{C_2^2} + 2\frac{w_1 C_1^2}{C_2^2}$

Extra

Derive SNR optimal weights

- $\frac{df}{dw_1} = 0 \implies 2w_1 - 2\frac{C_1}{C_2^2} + 2\frac{w_1 C_1^2}{C_2^2} = 0$
- $2w_1\left(1 + \frac{C_1^2}{C_2^2}\right) = 2\frac{C_1}{C_2^2}$
- $w_1 = \frac{C_1}{C_1^2 + C_2^2}$
- Note this is almost, but not quite the right answer, since we need the numerator to be C_1^*, not C_1
- This is because we derived everything assuming real-valued variables, not complex
- But it gives you a sense of how this functional form comes about

SNR Optimal Combination

Roemer et al., MRM 1990

- Optimized coil combination, taking into account channel noise covariances:
- Complex (SNR optimal)

$$\hat{x}_{comb} = \frac{\sum_j C_j^* \Sigma_n^{-1} x_j}{\sum_j C_j^* \Sigma_n^{-1} C_j}$$

- Sum-of-Squares

$$\hat{x}_{sos} = \sqrt{\sum_j x_j^* \Sigma_n^{-1} x_j}$$

Optimal Combination

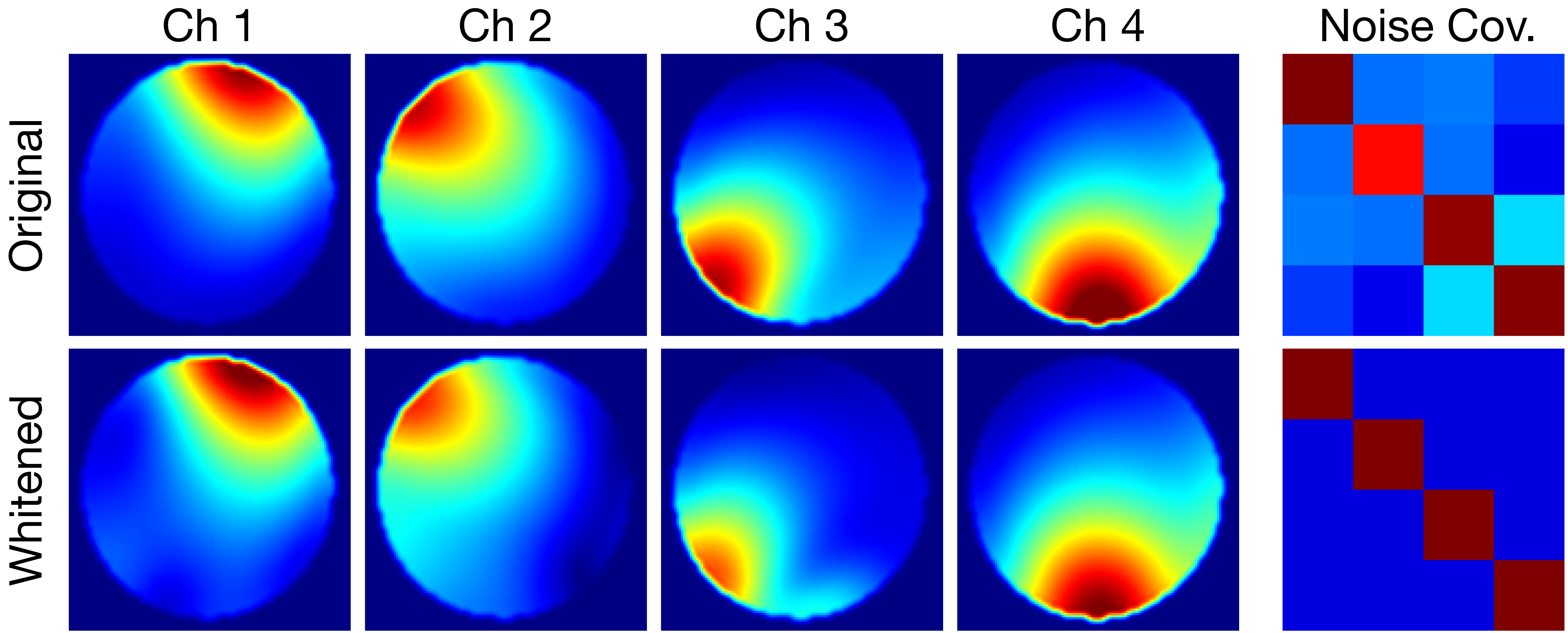
- We need to know the actual noise covariance across channels to actually arrive at SNR-optimal solutions
- Well-behaved coils often have close to identity noise coupling (i.e. minimal cross-channel noise correlations), and its a common approximation to *assume* the noise covariance is the identity
- However, we can actually measure the noise covariance matrix Σ_n to ensure SNR optimality
- By transforming the multi-channel data with Σ_n^{-1}, we can *whiten* the channels, and proceed assuming *i.i.d* noise
  - Identical: Normalizes the variance across channels (down-weights high-noise channels relative to low noise channels)
  - Independent: De-correlates the noise coupling between channels

Noise pre-whitening

- We can also “pre-whiten” the data ahead of time, to avoid needing to keep track of the noise covariance for subsequent operations
- Consider noise vectors $n = [n_1, n_2, \ldots]^T$ from each channels, covariance $\Sigma \sim nn^H$
- Compute a factorization of Σ^{-1} such that $LL^H = \Sigma^{-1}$ (e.g. Cholesky decomposition)
- Transform the multi-channel data x with L^H, then the noise from each channel gets transformed the same way, resulting in $L^H n$
- Computing a new covariance using $L^H n$ results in $\Sigma^{whitened} \sim L^H n (L^H n)^H$
- $\Sigma^{whitened} \sim L^H n n^H L = L^H \Sigma L = I$, the channels are transformed to have identity covariance

Noise pre-whitening

4-channel example



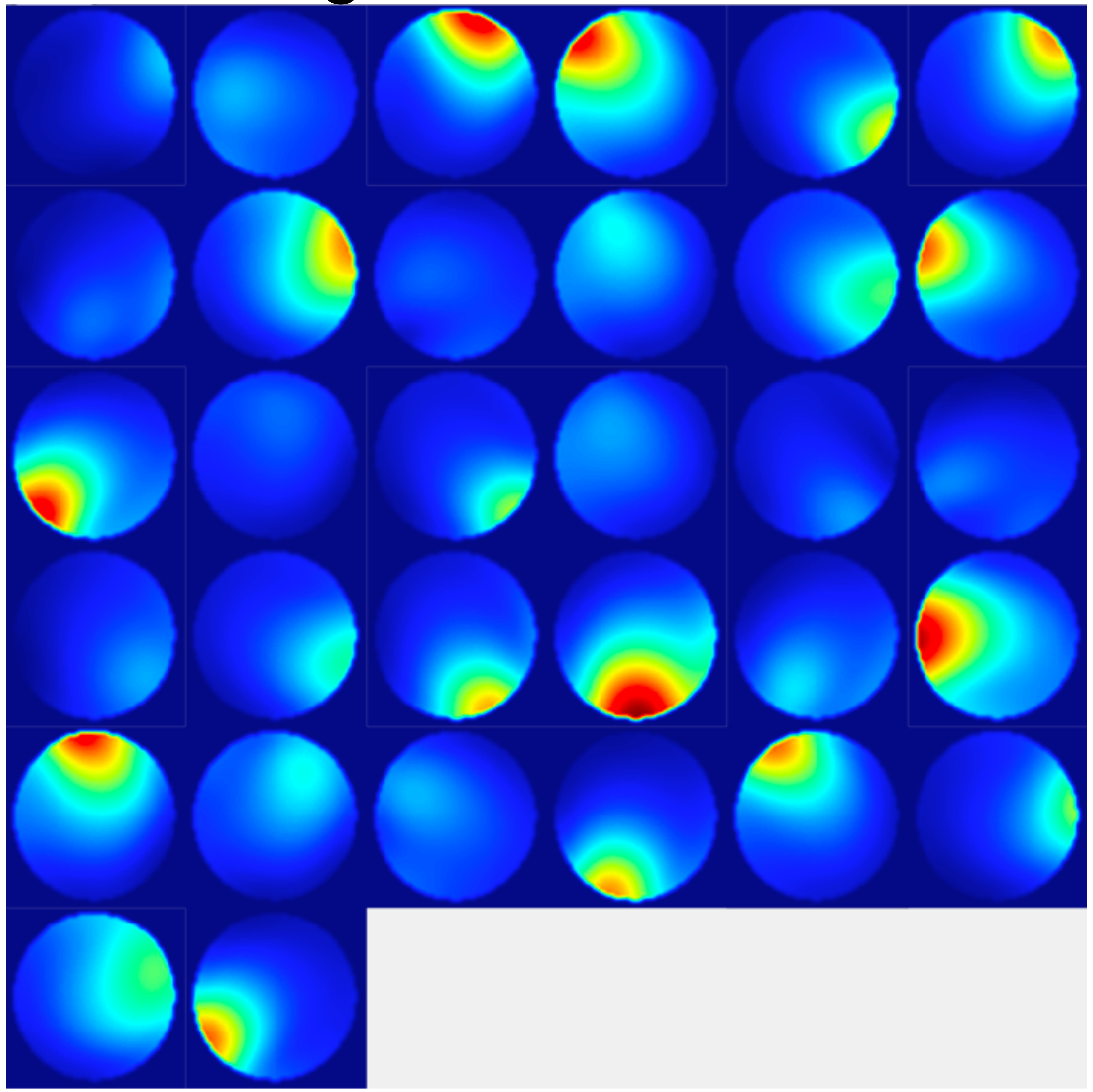
Coil Compression

Buehrer et al., MRM 2007

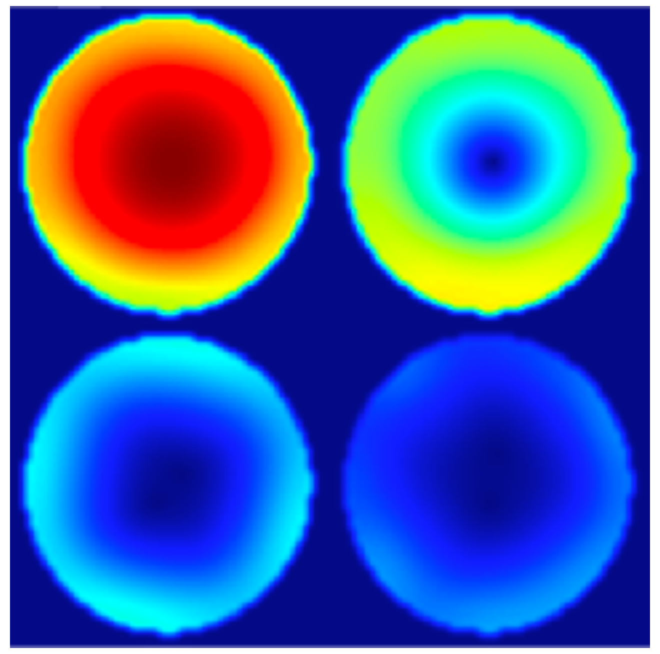
- We can define linear transformations of the multi-channel data for other reasons as well
- For example, we often want to reduce the size of the data for ease of storage or to reduce computation time and memory
- Coil compression defines a transformation to compress the channel dimension without losing significant information (although the compression is almost always lossy)
- Typically, this is done using a PCA or SVD, defining the transform by maximizing the variance of the coil sensitivities projected onto some lower-dimensional subspace

Coil Compression

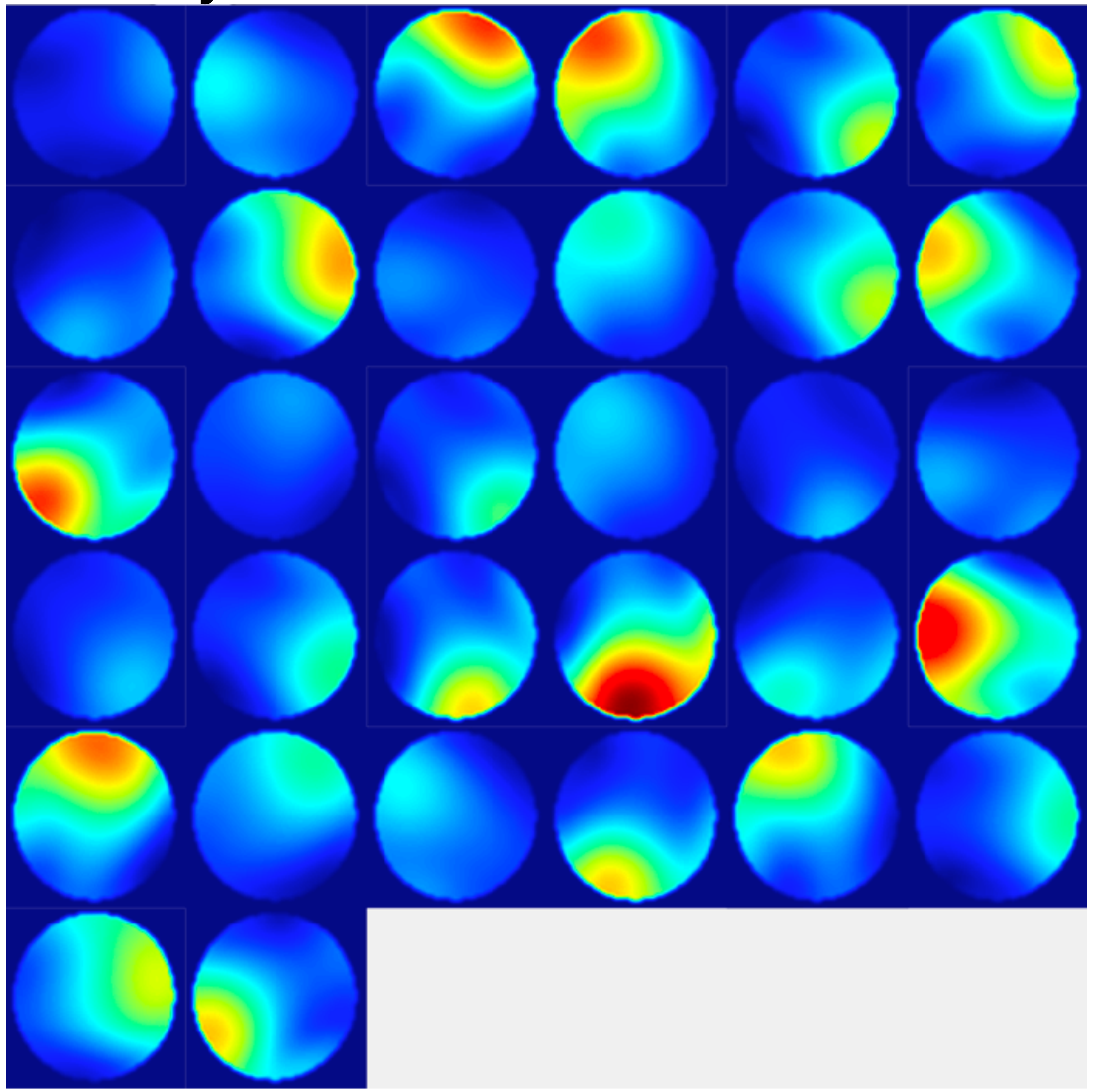
Original 32-channel



Compressed 4-channel



Synthesized 32-channel



Part III – Parallel Imaging

Parallel Receivers

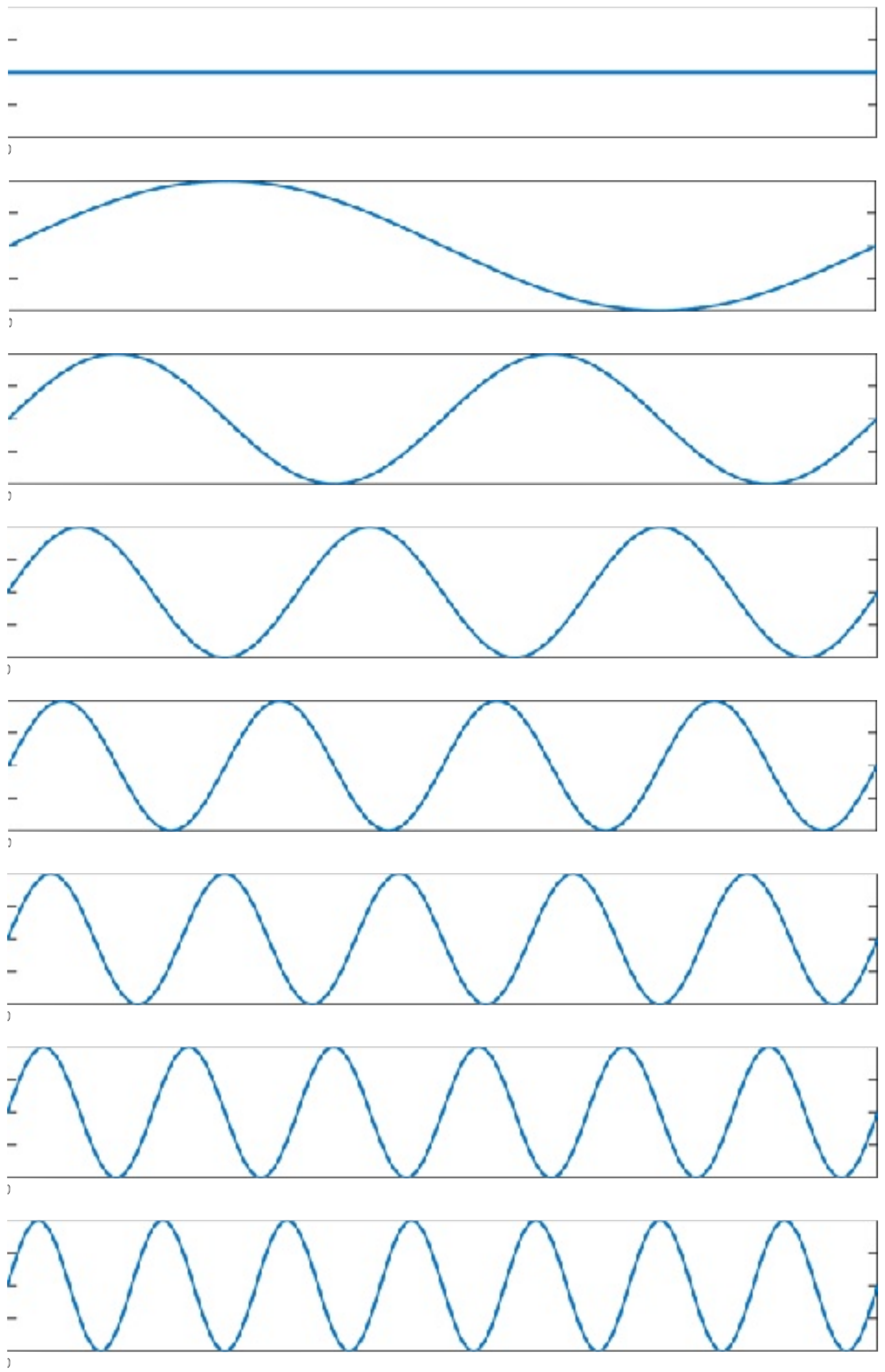
- Each coil provides a different “view” of the image
- Different coils have different “sensitivities”, which weights or filters the measurement
- Each coil or receiver element represents a separate, parallel measurement
- We use these properties to effectively increase the number of measurements we observe with no time penalty

Single-Channel Linear Measurement Model

Basis Coefficients

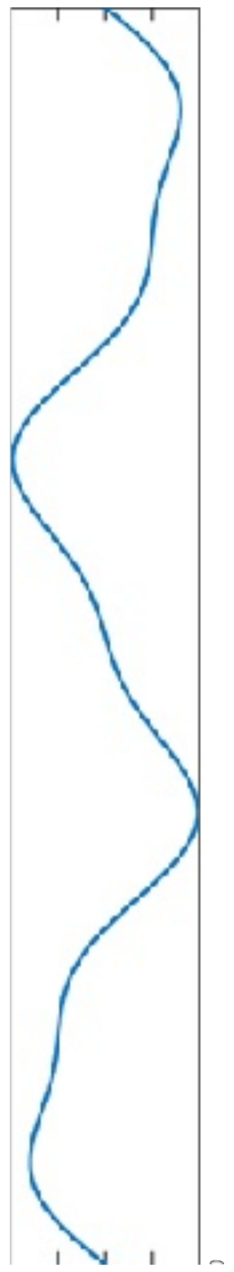
$$\begin{bmatrix} 0.31 - 1.20i \\ 2.19 + 0.98i \\ 1.06 + 2.24i \\ -0.73 + 1.22i \\ -2.94 - 1.65i \\ 1.49 - 2.71i \\ 0.88 + 0.14i \\ -1.27 + 1.73i \end{bmatrix} =$$

Fourier Basis Set

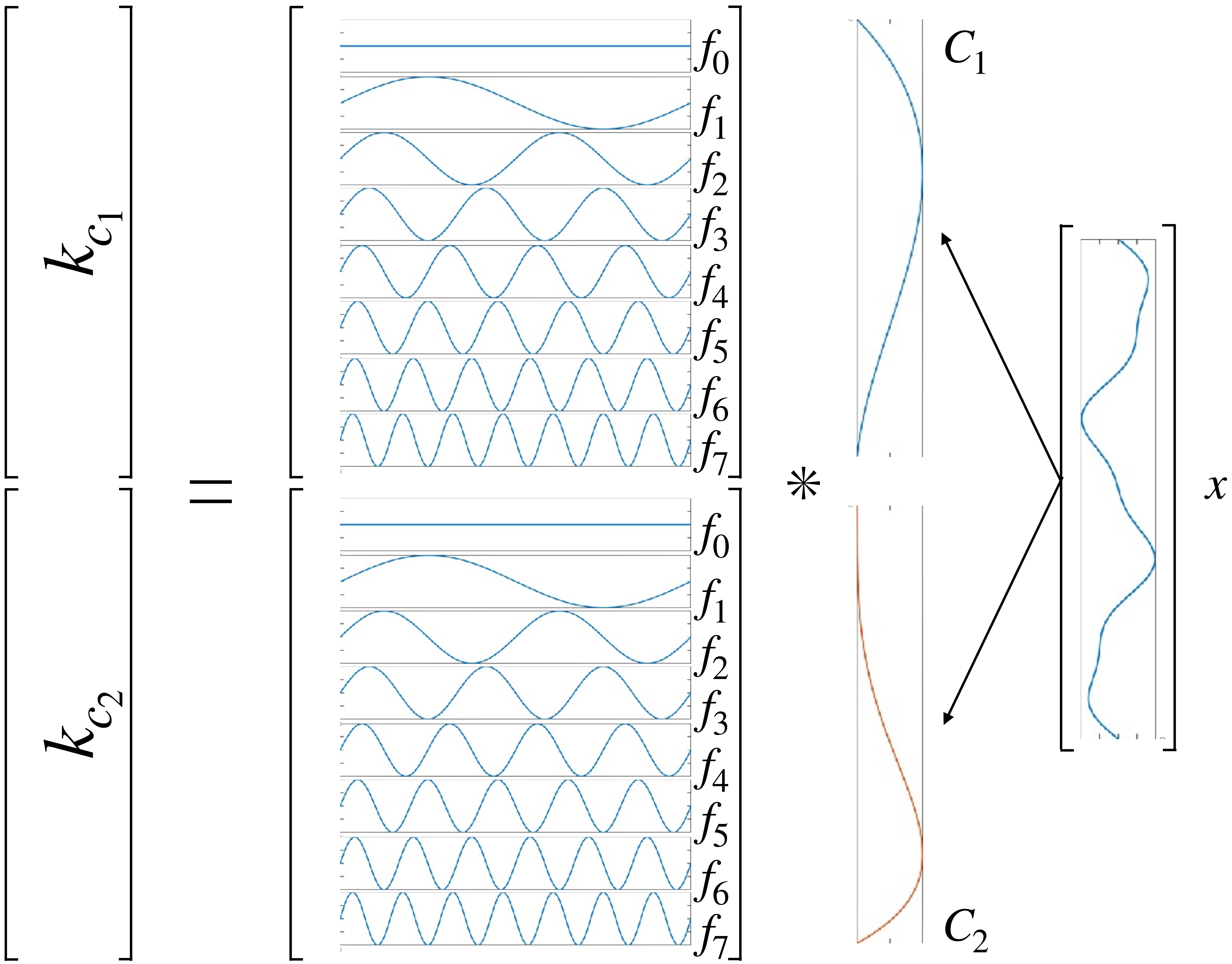


$*$

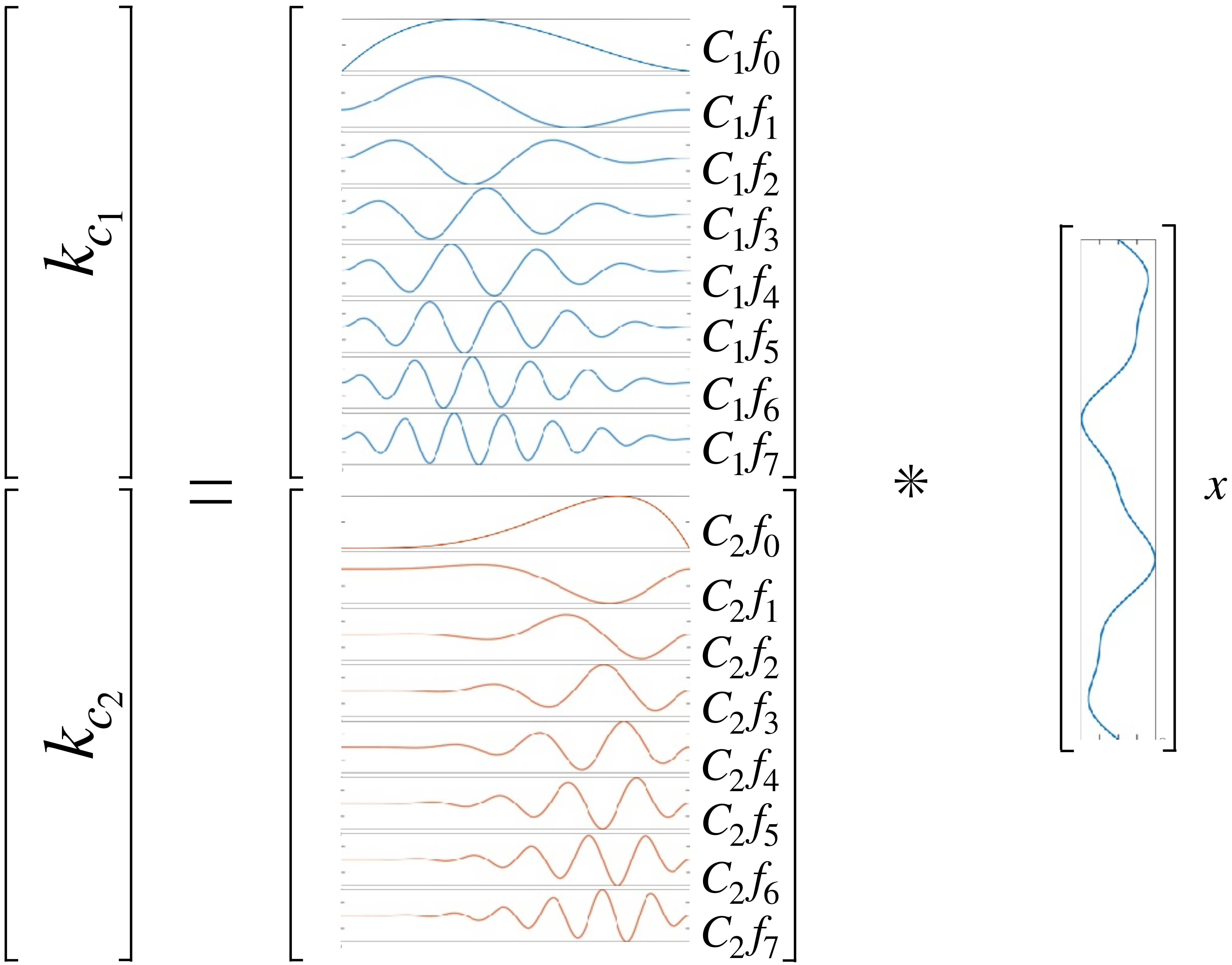
Signal



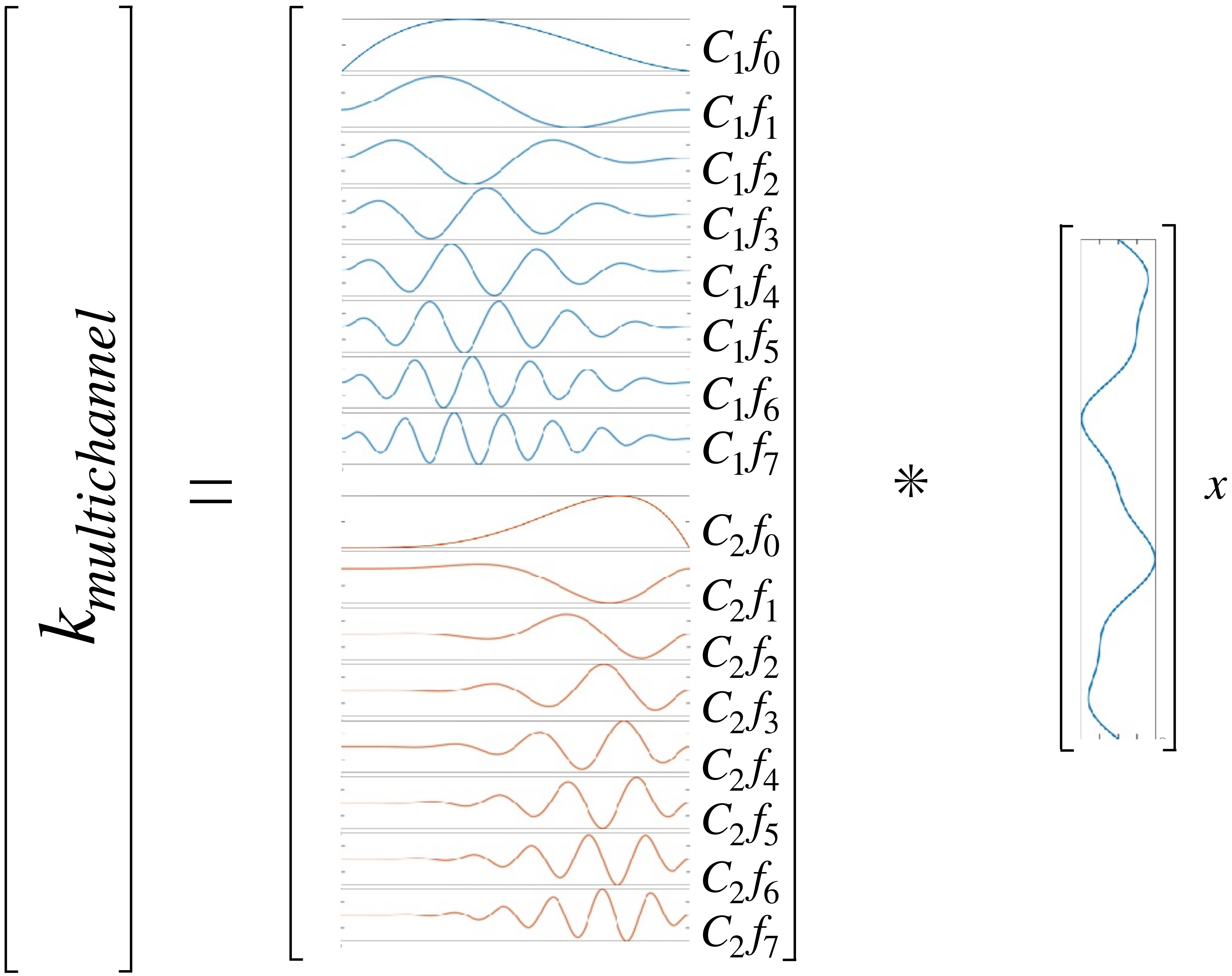
Multi-Channel Linear Measurement Model



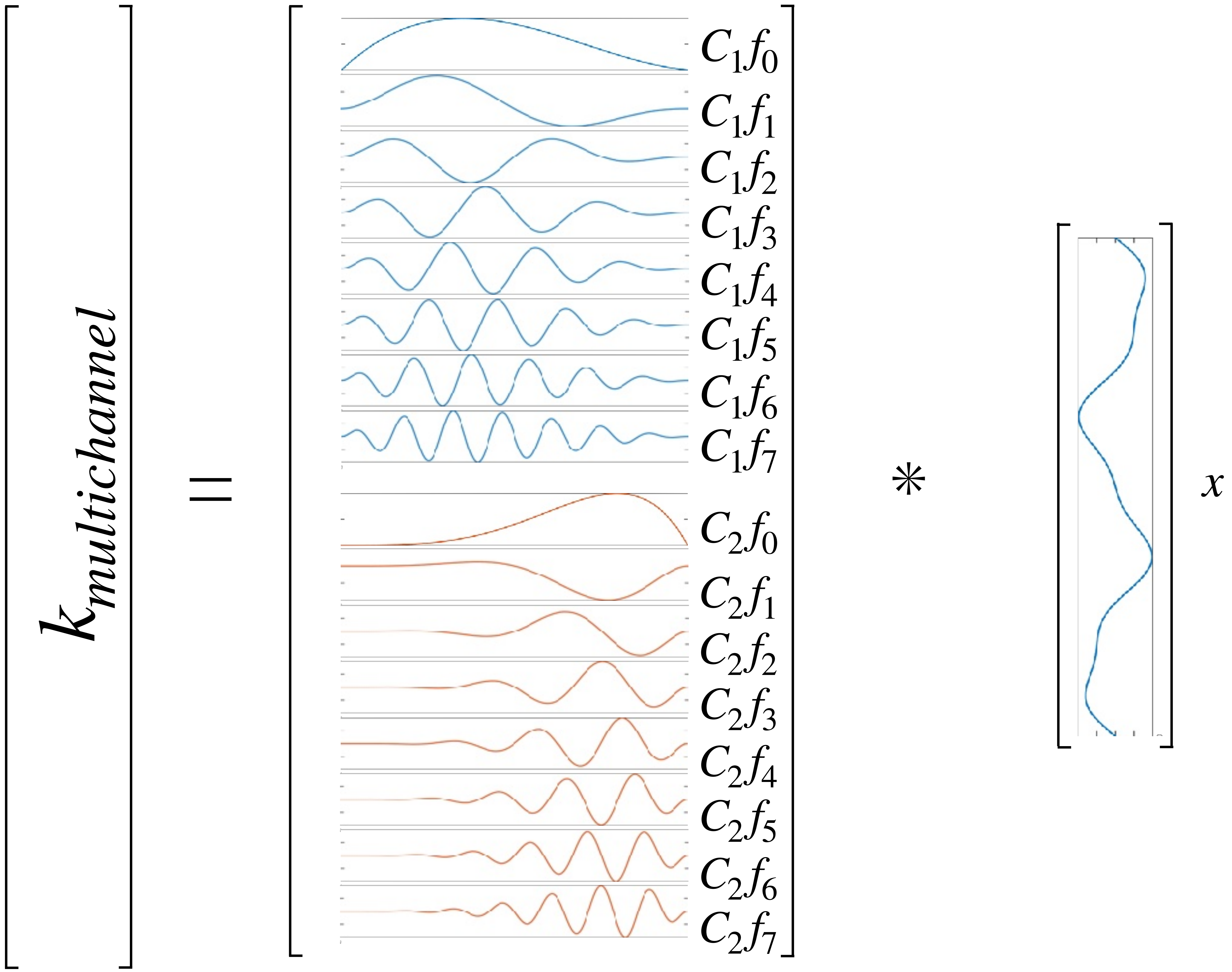
Multi-Channel Linear Measurement Model



Extended Measurement Model

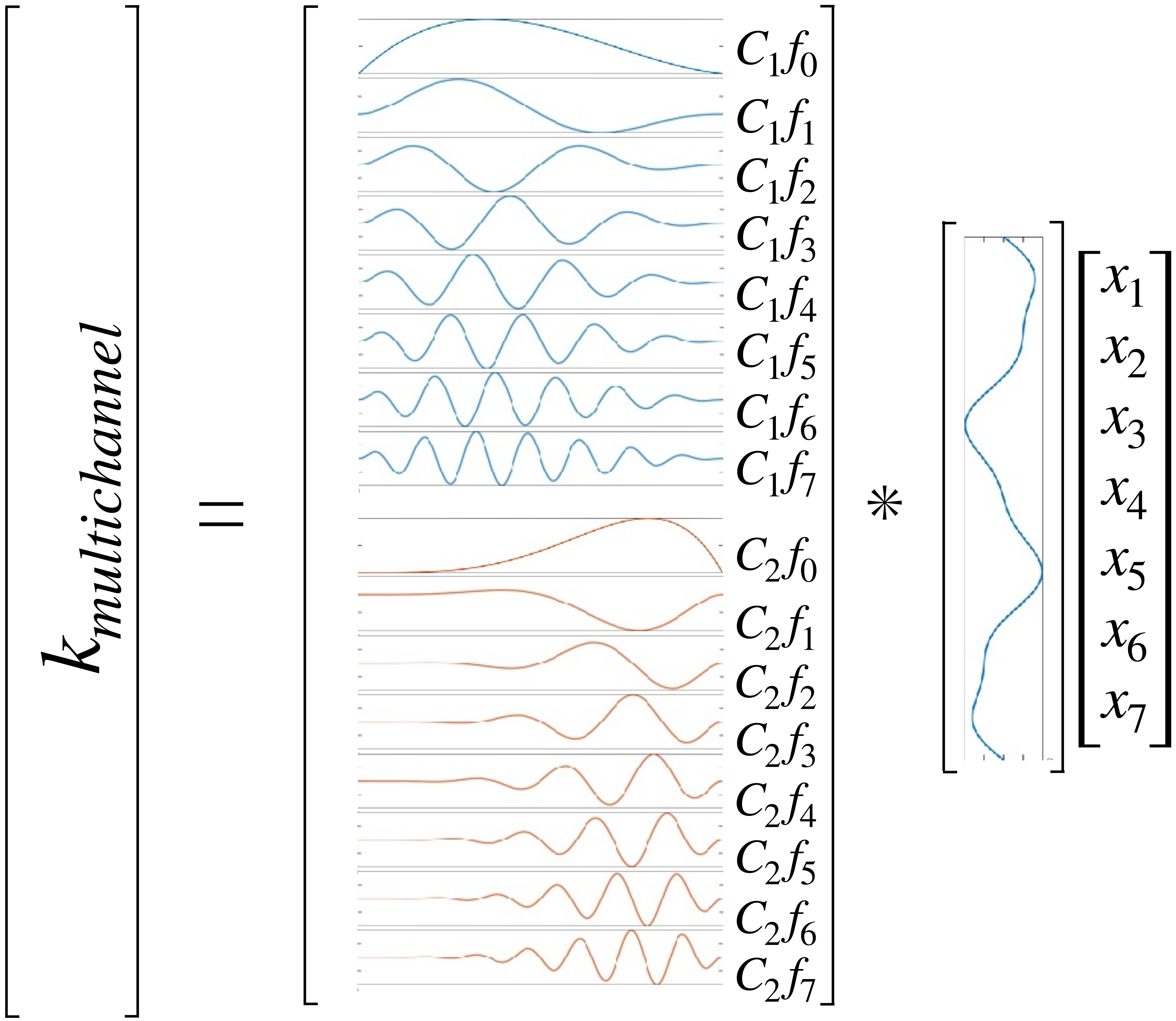


Extended Measurement Model



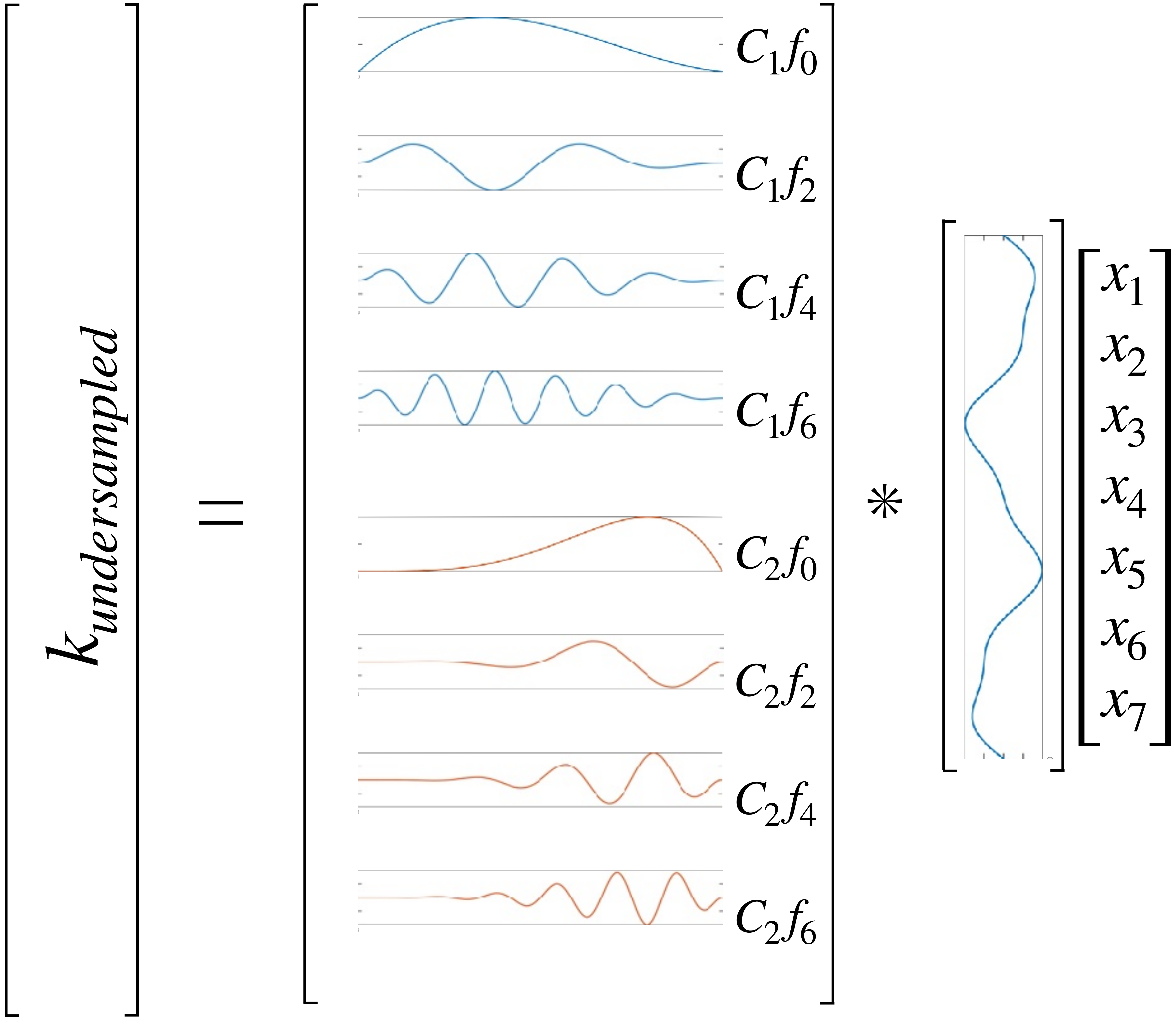
- Sampling durations in MRI are proportional to the amount of encoding needed
- Each k-space sample must be acquired sequentially, so more samples means more time
- Time can be limiting factor in high-resolution acquisitions, and in clinical protocols

Extended Measurement Model



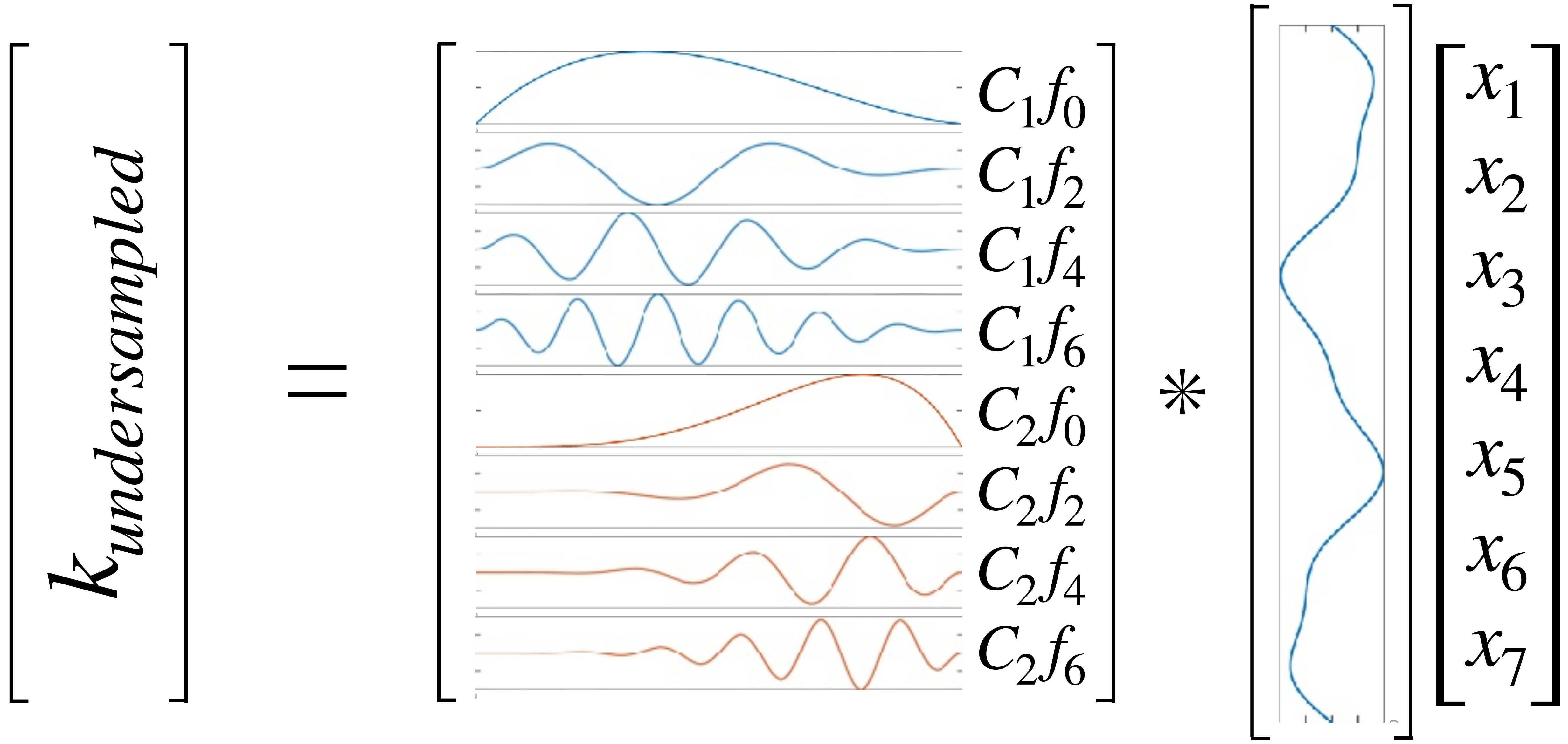
- Assume our signal has 8 unknown values
- In the original single-channel model, we had 8 measurements
- Enough to uniquely reconstruct the image
- Now, in the multi-channel measurement model, we have more than enough measurements
- What happens if we skip some?

Extended Measurement Model



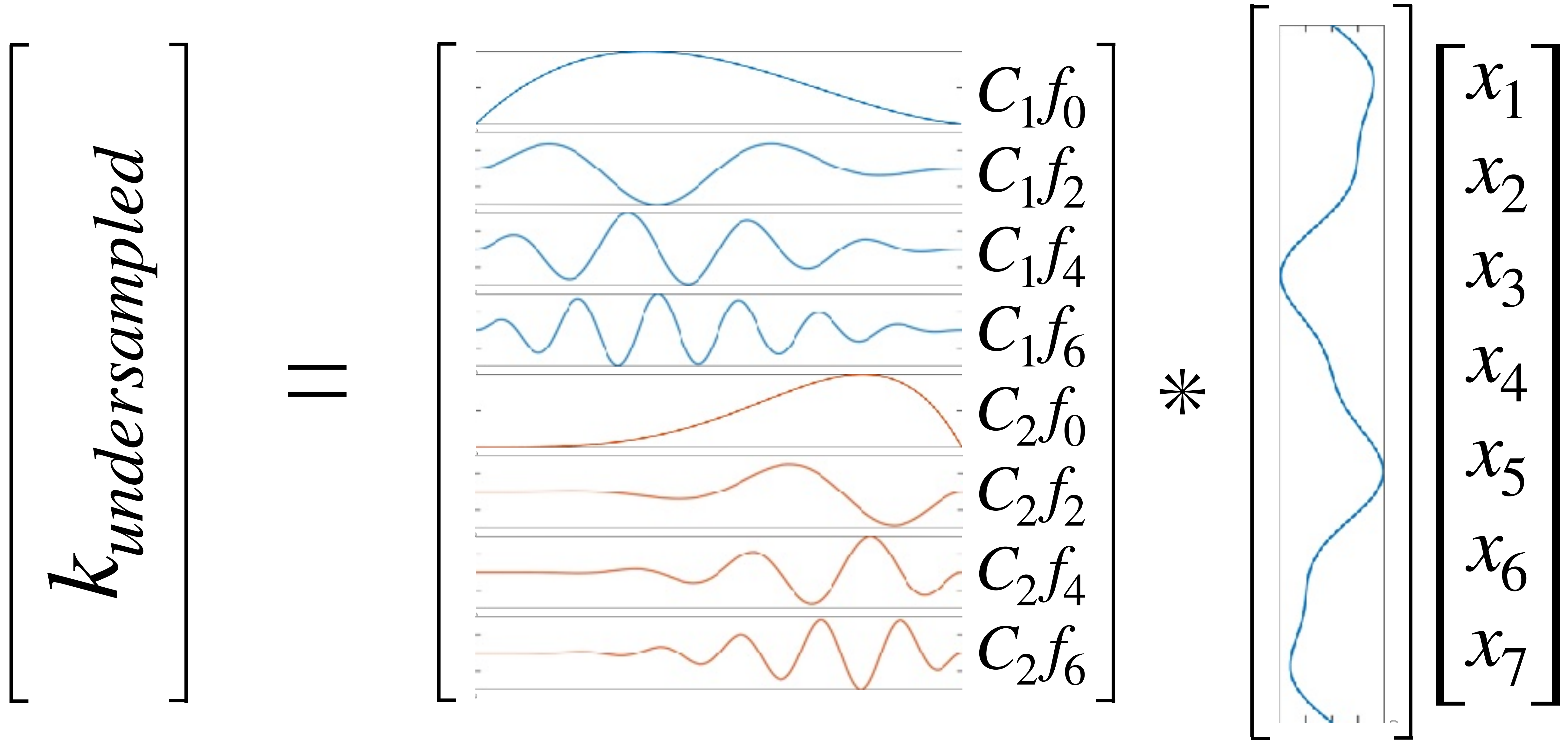
- Assume our signal has 8 unknown values
- In the original single-channel model, we had 8 measurements
- Enough to uniquely reconstruct the image
- Now, in the multi-channel measurement model, we have more than enough measurements
- What happens if we skip some?

Parallel Imaging



- Now, we have just as many measurements (8) as in the fully-sampled single-channel case
- But in half the time, since we acquire only 4 Fourier encodings
- The measurements from different channels are acquired *in parallel*
- The core idea is that missing k-space encodings are compensated by the multi-channel measurements
- This ensures the image can still be reconstructed, i.e. the imaging linear system can still be solved

Parallel Imaging



- However, we need to ensure that the the rows of the encoding matrix are still linearly independent
- Linear dependency reduces the rank and effective encoding power of the measurement operator
- e.g. there's no point measuring $[1 \quad 1]$ if we've already measured $[1 \quad 0]$ and $[0 \quad 1]$
- Ideally, we want the rows to be maximally orthogonal
- Acceleration factor (proportion of under-sampling) is limited by:
  - The number of channels
  - The linear independence of these channels
- Typically, we want to be well-overdetermined, otherwise the reconstruction is prone to significant noise amplification (more on this later)

Parallel Imaging

- If there are insufficient numbers of coils, or not enough coil sensitivity differentiation (linear independence) for the amount of undersampling
  - The encoding matrix is still underdetermined (or not full rank)
  - In other words, our encoding basis does not span the full signal space (not enough information)
  - Without enough information, there are too many degrees of freedom to identify a unique solution to the inverse problem
- If there are more coils than necessary, the problem is overdetermined
  - The encoding matrix however, is full rank and our encoding basis is an over-complete representation of the signal vector space
  - We commonly use least squares criterion to select an optimal solution

Practical Parallel Imaging

- Normally, for any non-square linear system $k = Ex$, we can solve it in the least squares sense from the normal equation:

$$E^H E x = E^H k$$

- Either directly using the pseudo-inverse:

$$\hat{x} = (E^H E)^{-1} E^H k$$

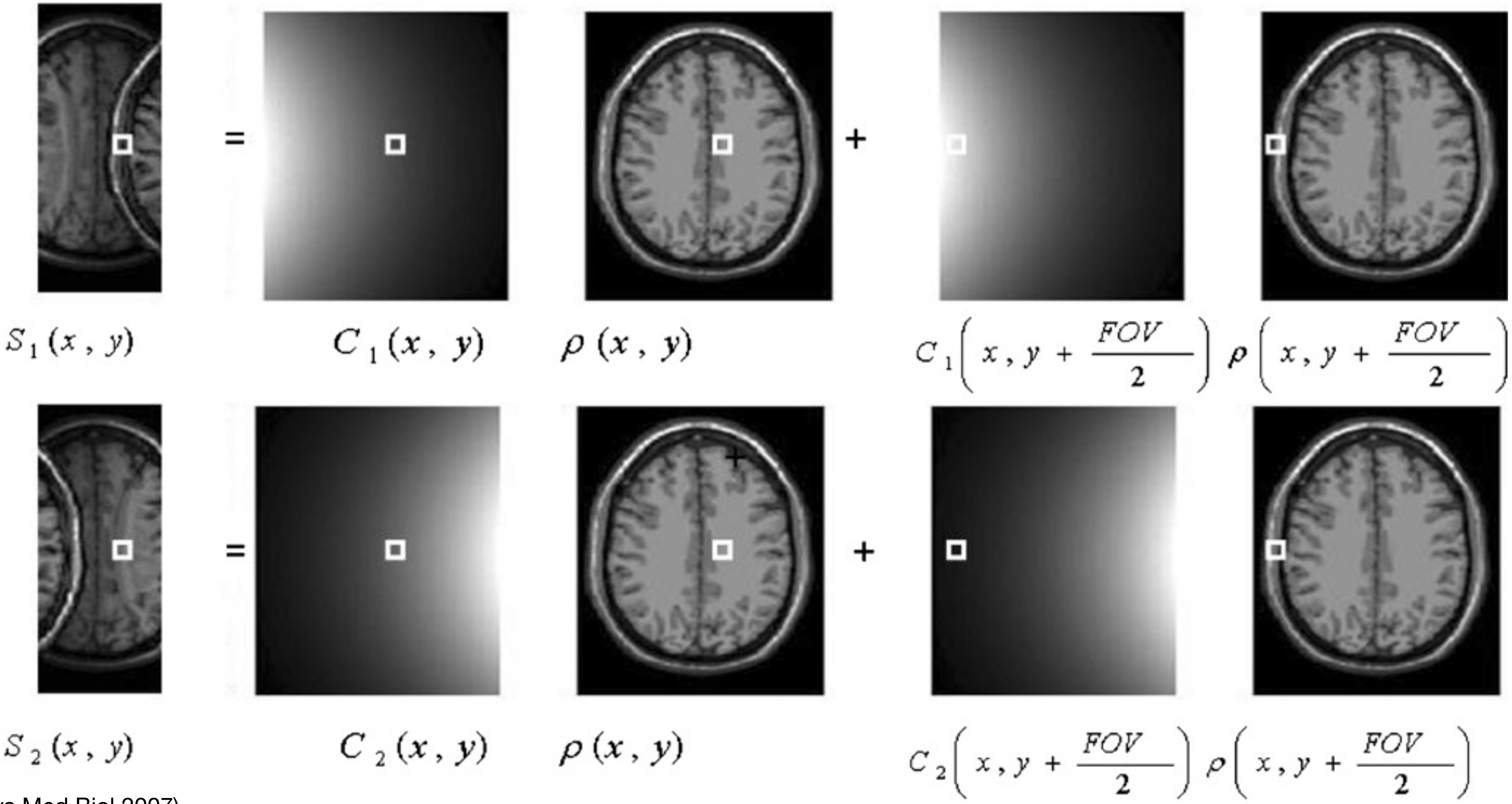
- Or iteratively (e.g. conjugate gradient)
- However, this is computationally inefficient, and practically speaking different algorithms exist to perform parallel imaging reconstruction

Practical Parallel Imaging

- SENSE (Pruessmann et al., MRM 1999)
  - Image domain parallel imaging reconstruction
- GRAPPA (Griswold et al., MRM 2002)
  - K-space domain parallel imaging reconstruction
- Fundamentally both rely on additional information from multi-channel coil sensitivities to compensate for missing k-space samples

SENSE

Pruessmann et al., MRM 1999

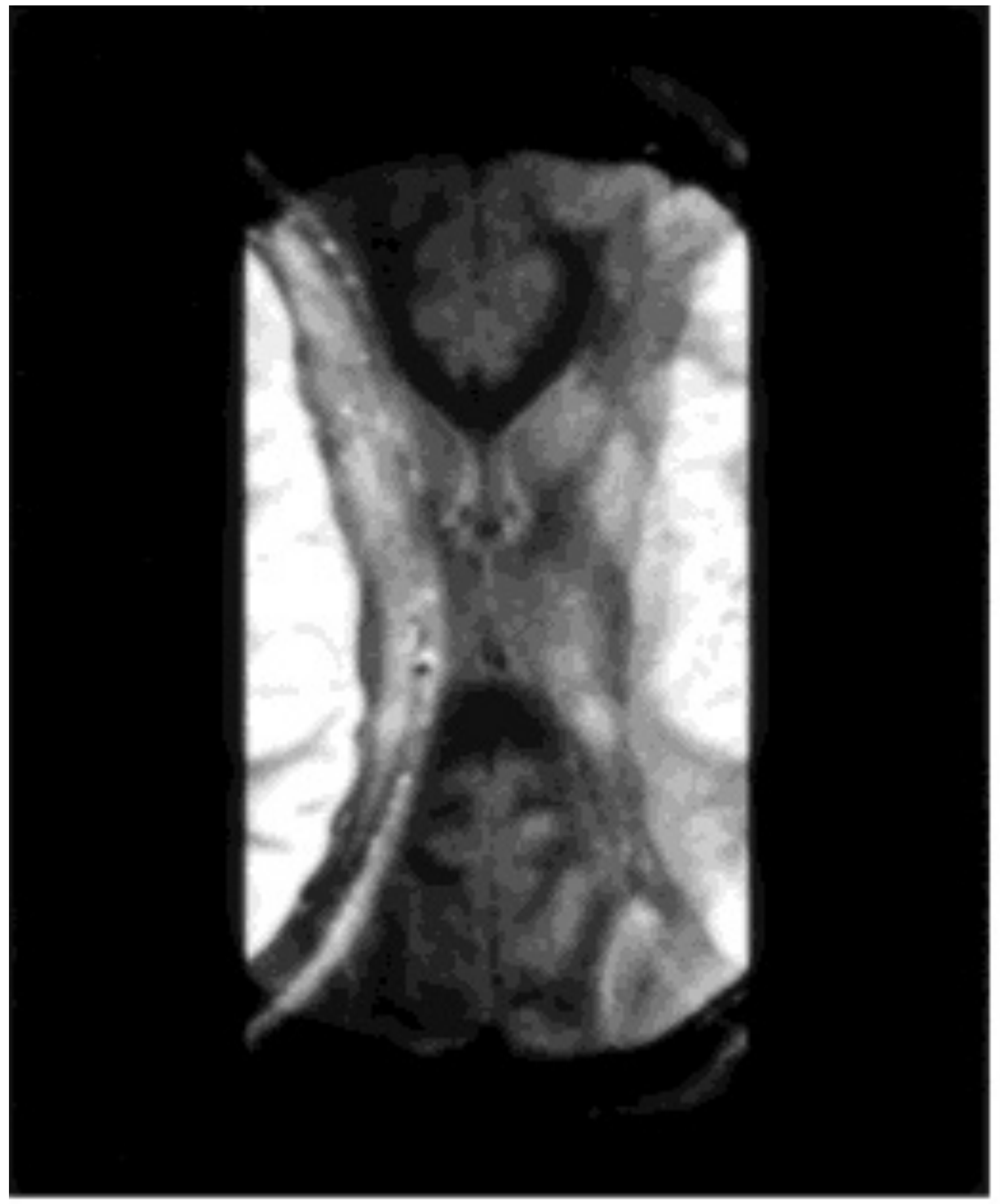


(Larkman et al., Phys Med Biol 2007)

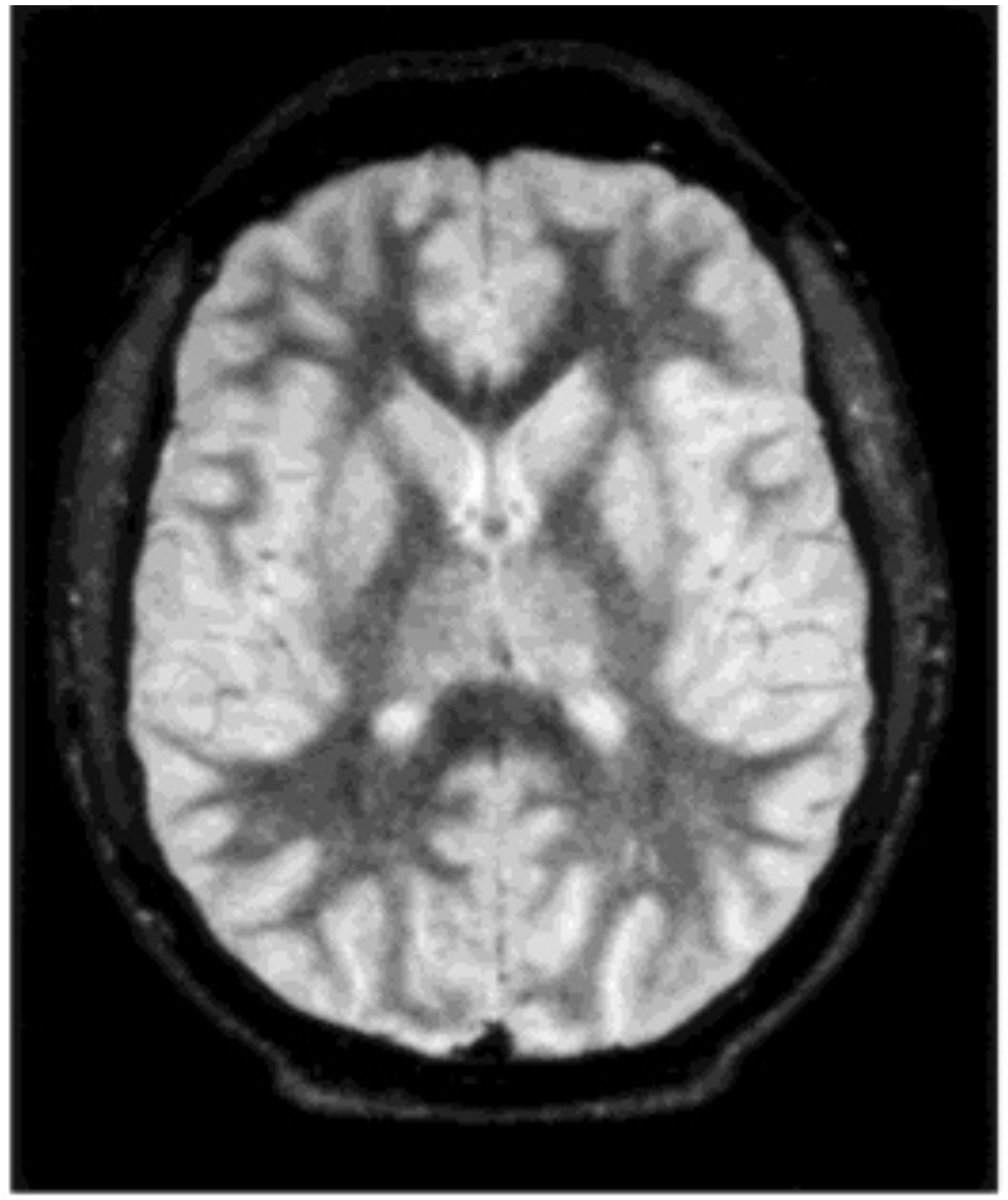
SENSE

Pruessmann et al., MRM 1999

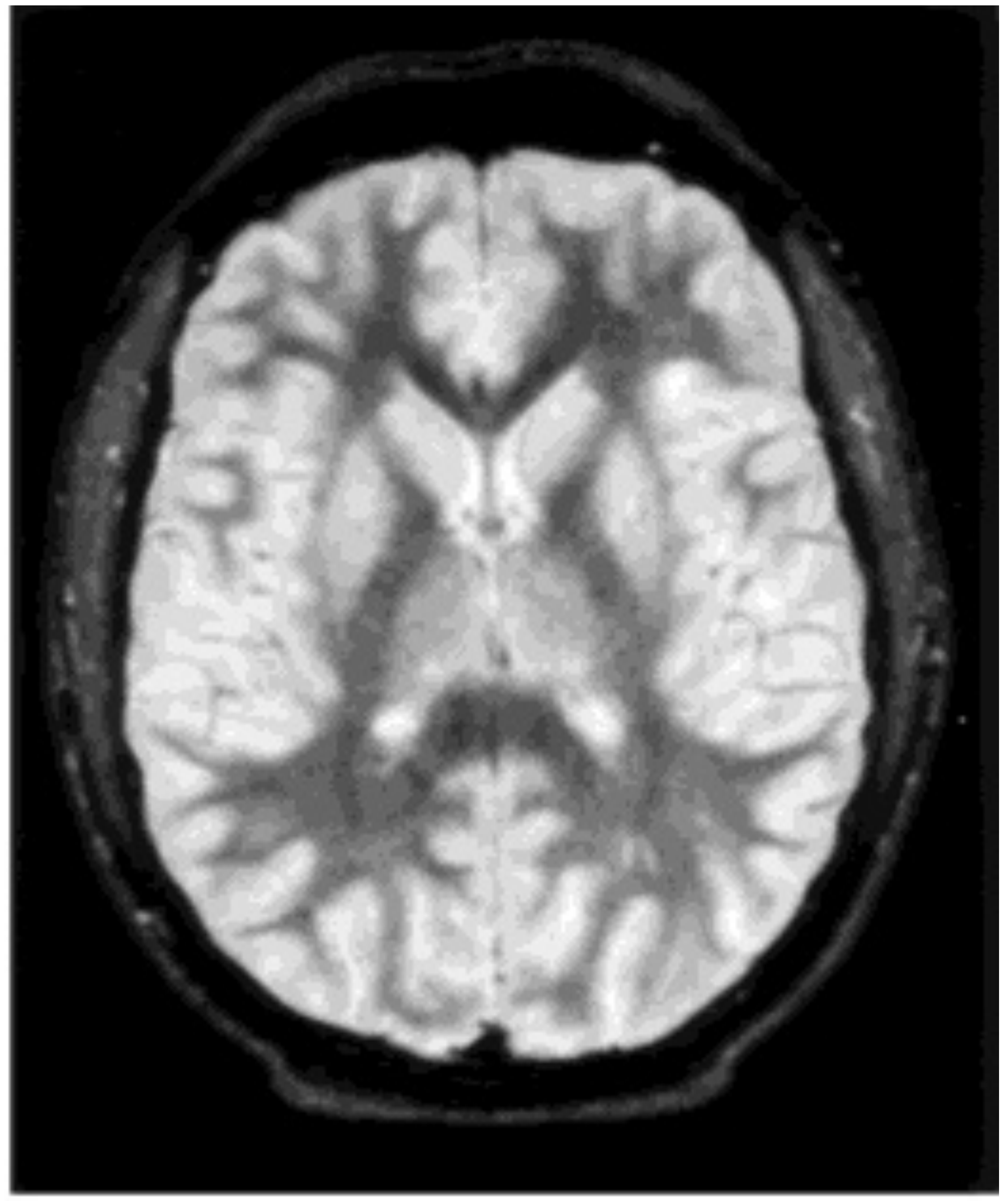
R=2
Pre-SENSE

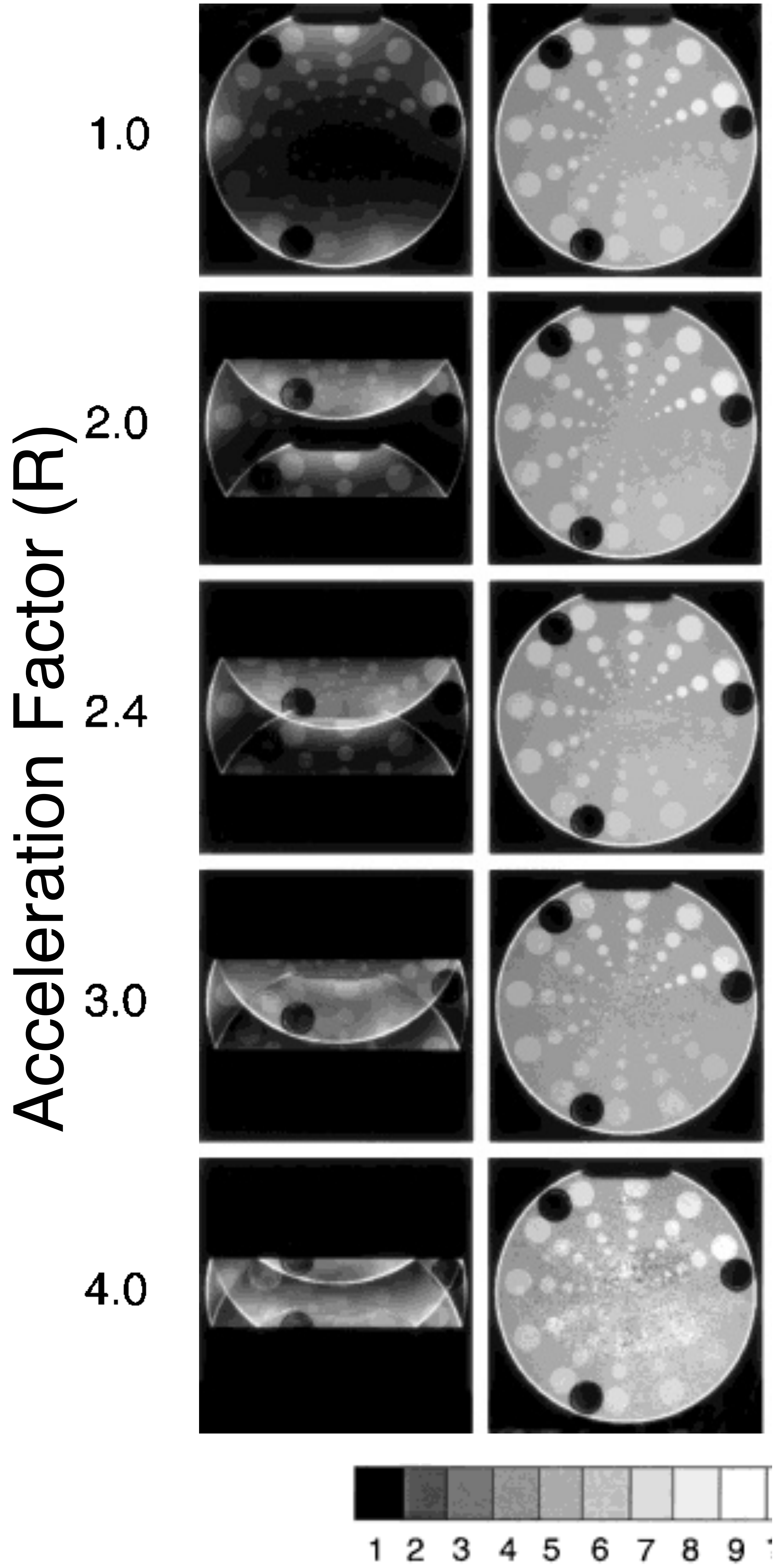


R=2
Post-SENSE



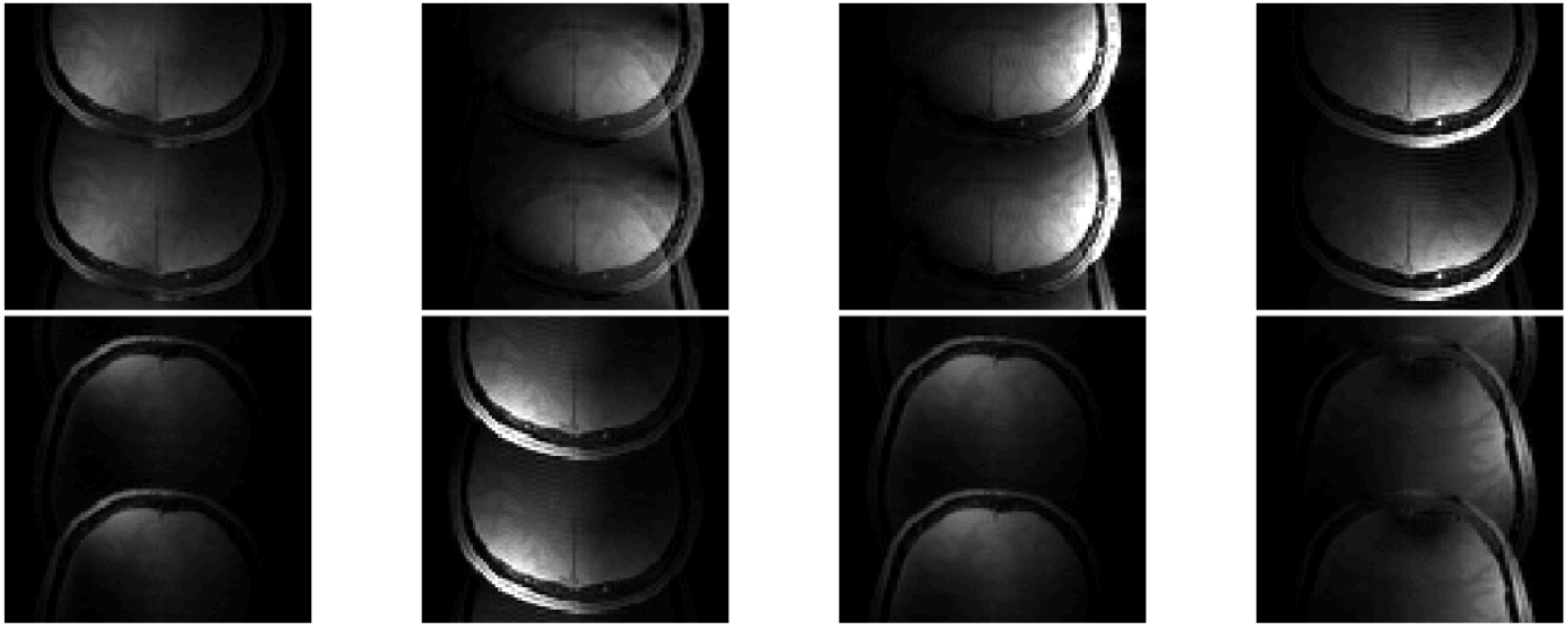
R=1
2x Slower





SENSE
Basic Idea

- If we sub-sample k-space with regularly spaced lines, the individual coil images have coherent “aliasing” (i.e. the image overlaps itself in a simple way)



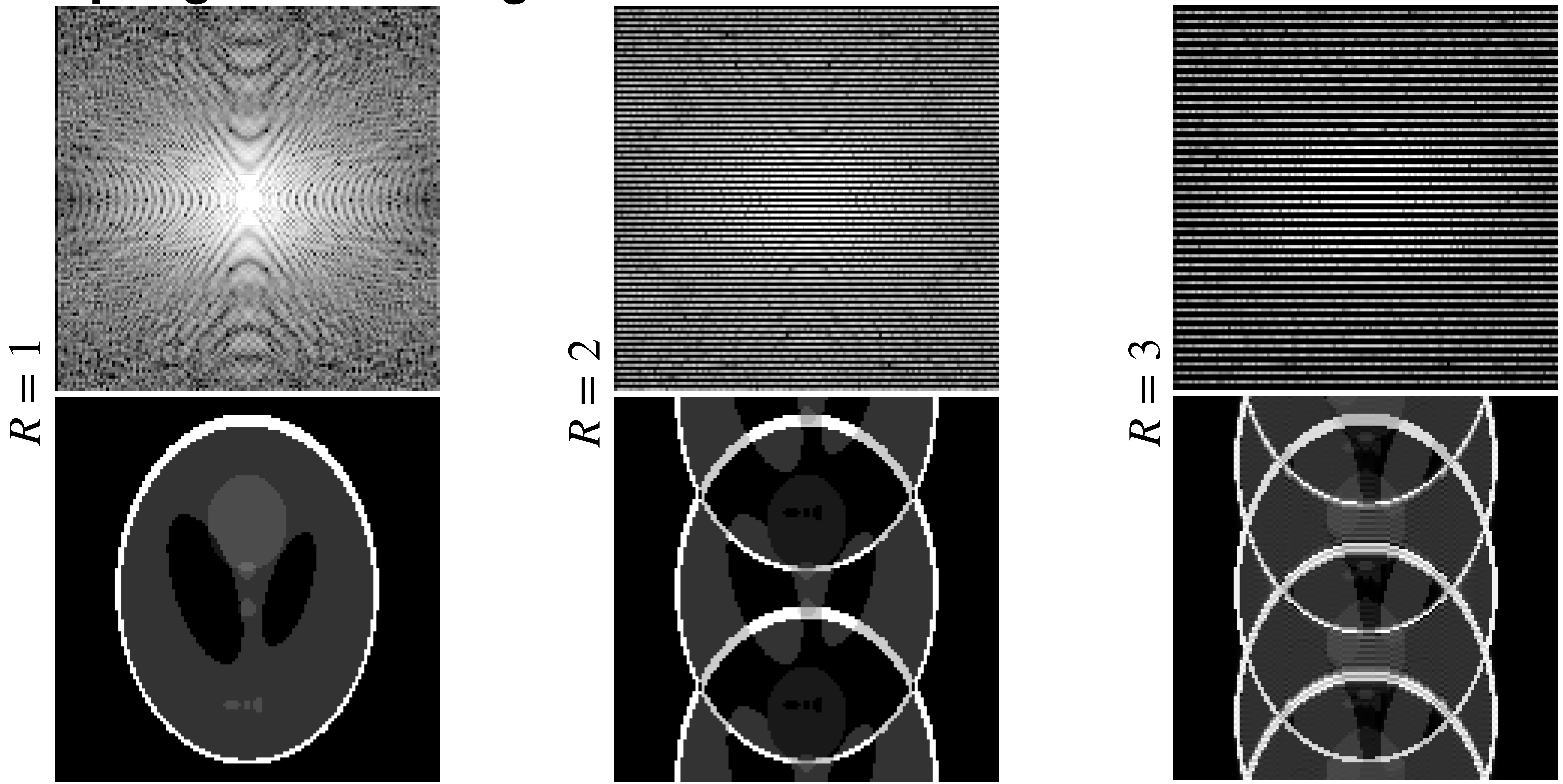
SENSE

Sampling and aliasing

- Sampling every R^{th} line in k-space is equivalent to multiplying the k-space with a Dirac comb function with width $R\Delta k$
- The impact of this sub-sampling on the image, is then convolution of the image with another Dirac comb function with width $\dfrac{1}{R\Delta k} = \dfrac{FOV}{R}$
- These comb functions are Fourier transform pairs
- This convolution results in the aliasing/overlap pattern you see, where the number of overlaps is governed by the parameter R

SENSE

Sampling and aliasing



SENSE

- Inverse Fourier Transform before using sensitivity encoding information
- This is advantageous because it greatly simplifies the inverse problem
- It becomes spatially separable, meaning that the reconstruction problem is reduced to sets of R voxels, where R is the acceleration factor, and also the aliasing factor
- Solve many trivially small problems, instead of one giant one
- SENSE requires explicit knowledge and/or mapping of the coil sensitivities

SENSE

Reconstruction

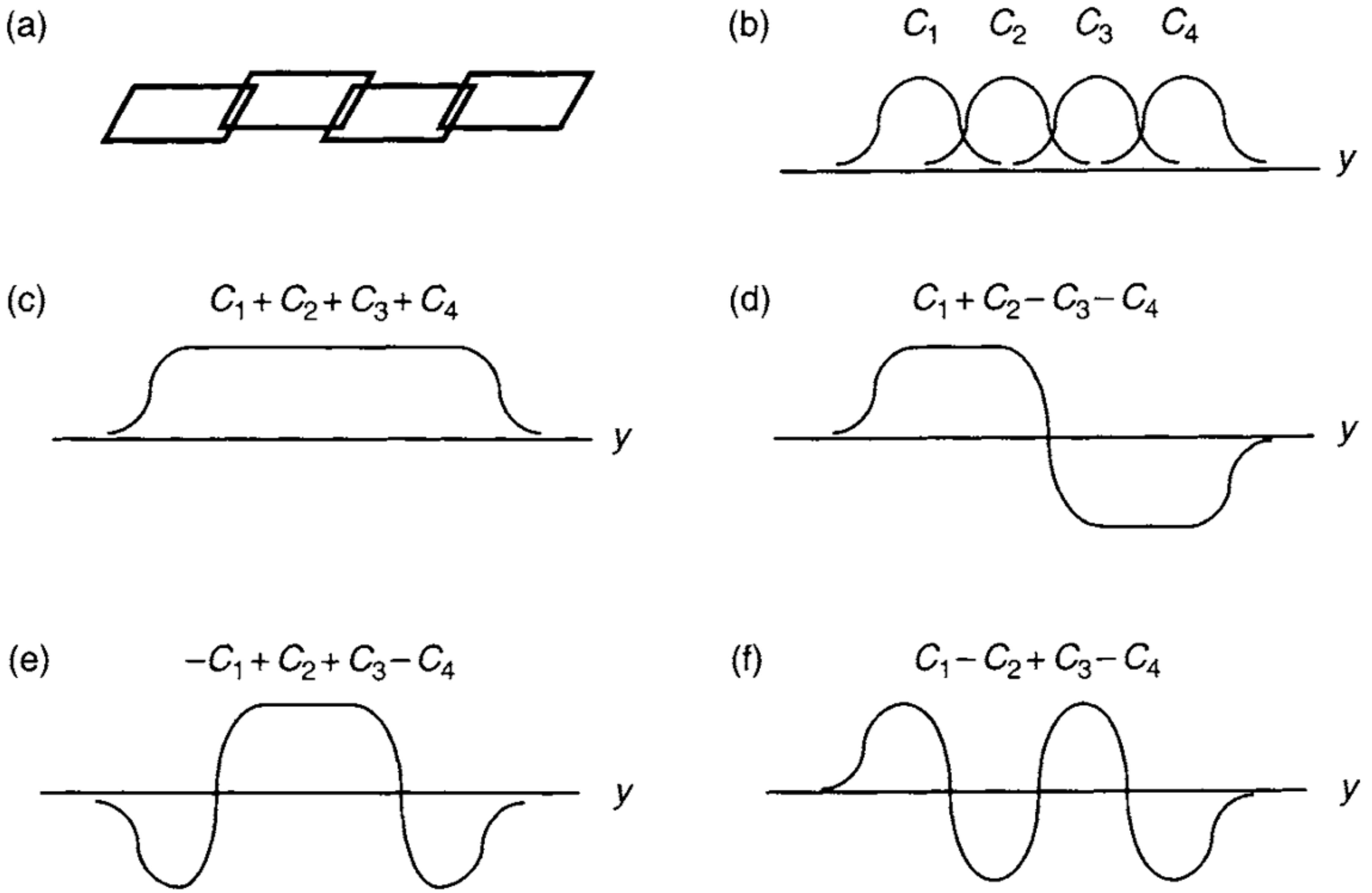
- For all sets of aliased points in the image, solve the $R-$dimensional linear system:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N_c} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1R} \\ c_{21} & c_{22} & \cdots & c_{2R} \\ \vdots & \vdots & \vdots & \vdots \\ c_{N_c1} & c_{N_c2} & \cdots & c_{N_cR} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_R \end{bmatrix}$$

- The aliasing problems are not coupled, and can be solved independently
- This is very efficient, as N_c $(10^2 \sim 10^3)$ and $R \leq 4$ (typically) are small
- Typically we solve these problems in a least-squares sense, so if $y = Cx$, $\hat{x}_r = \left(C_r^H C_r\right)^{-1} C_r^H y_r$ where r indexes all the different aliasing sub-problems

SMASH

Sodickson et al., MRM 1997

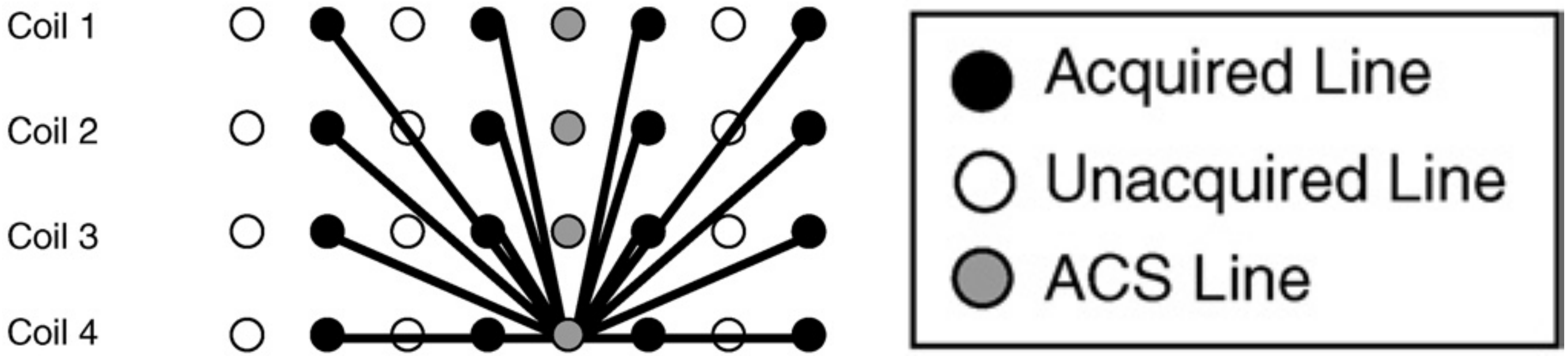


(Bernstein et al., 2004)

The key idea is to explicitly replace Fourier encoding steps with linear combinations of coil sensitivities

GRAPPA

Griswold et al., MRM 2002



Synthesize missing k-space points from linear combinations of nearby acquired points

GRAPPA

Basic Theory

- Here is the signal equation for channel j k-space: $s_j(k) = \int C_j(r)x(r)e^{-i2\pi k\cdot r}dr$
- Now, imagine we can find some weights w_j^l to approximate a Fourier encoding step for a particular channel l: $\sum_j w_j^l C_j(r) = C_l(r)e^{-i2\pi \Delta k\cdot r}$
- Then, if we apply those weights to the signal we get:

$$\sum_j w_j^l S_j = \sum_j \int w_j^l C_j x e^{-i2\pi k\cdot r}dr$$

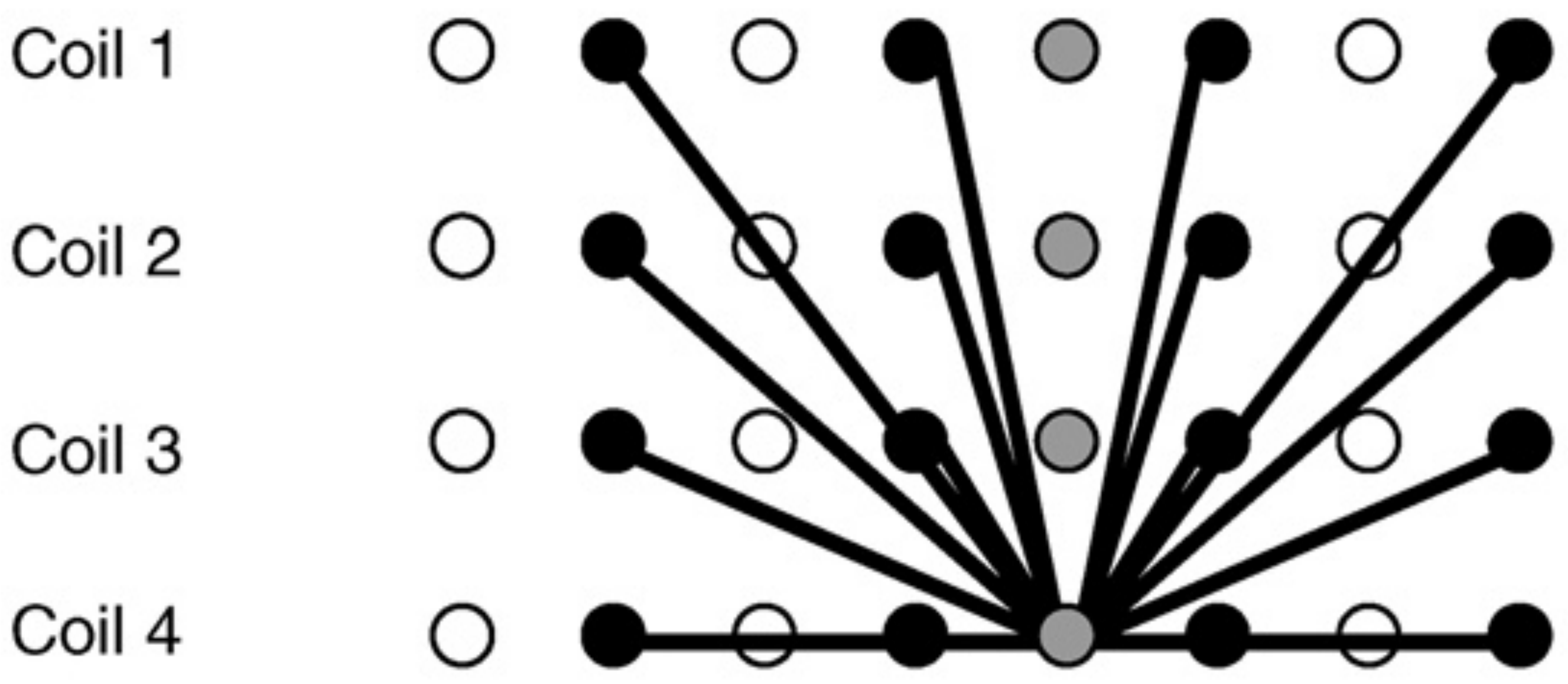
GRAPPA

Basic Theory

- This becomes: $\int \left(\sum_j w_j^l C_j \right) x e^{-i2\pi k \cdot r} dr = \int C_l(r) e^{-i2\pi \Delta k \cdot r} x e^{-i2\pi k \cdot r}$
- $\int C_l(r) x e^{-i2\pi(k+\Delta k)\cdot r} = s_l(k + \Delta k)$
- So, the weights w_j^l transform the k-space information from one location k across all channels, to a different location $k + \Delta k$ in a particular channel l

GRAPPA
Basic Theory

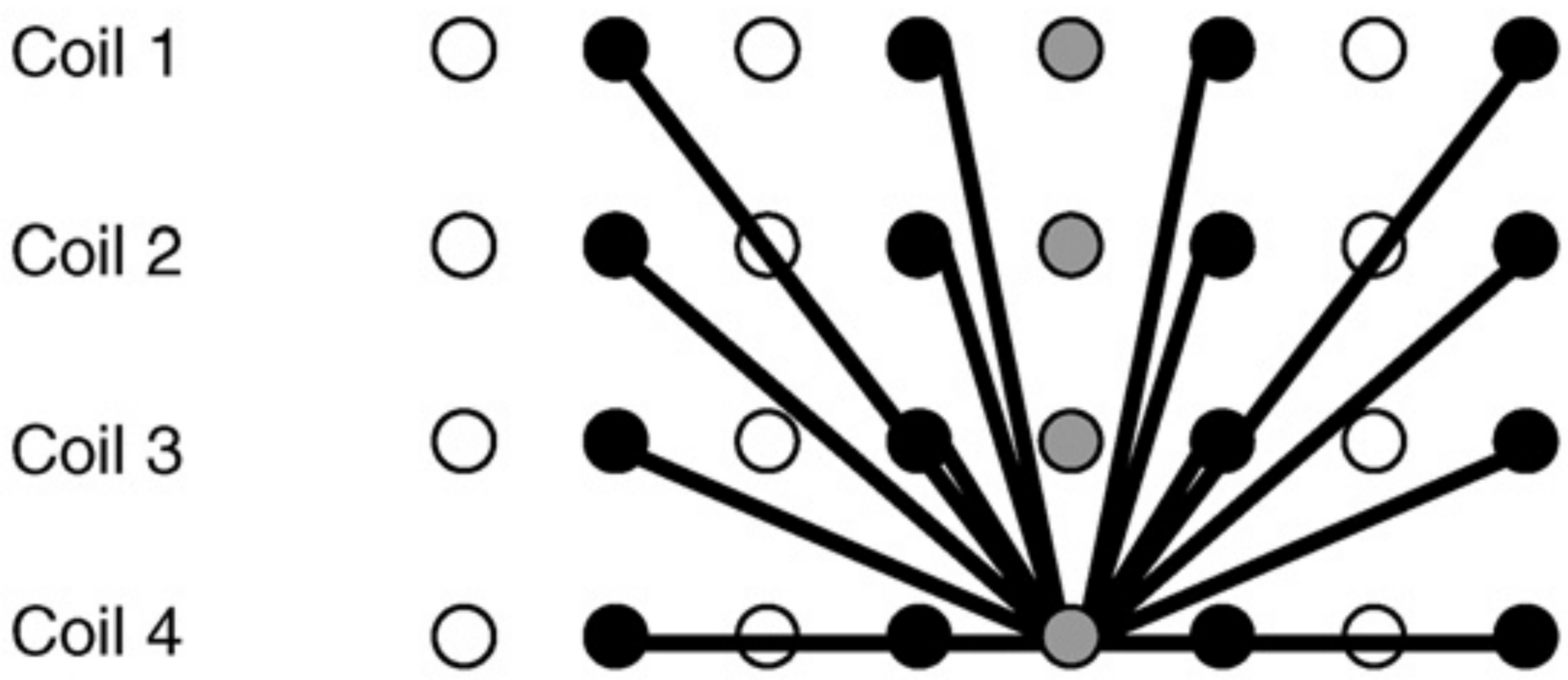
- In practice, many such *source* locations are used to synthesize a given missing *target* location



- So, there is one set of weights per output channel per source point, and taken together these weights are often referred to as the GRAPPA kernel

GRAPPA
Basic Theory

- In this simple example below, there are 4-channels, and 16 source points



- Therefore the GRAPPA kernel has $4 \times 4 \times 16 = 256$ coefficients
- The GRAPPA kernel geometry dictates the geometry of source-to-target points

GRAPPA

Shift-invariance and contrast-independence

- The GRAPPA weights synthesize a k-space step Δk, and do not depend on the actual location k
    - Therefore, the kernel is *shift-invariant*
- Also, the weights depend only on the coil sensitivity information C_j, not the underlying image contrast x
    - Therefore, the kernel is *contrast-independent*

GRAPPA

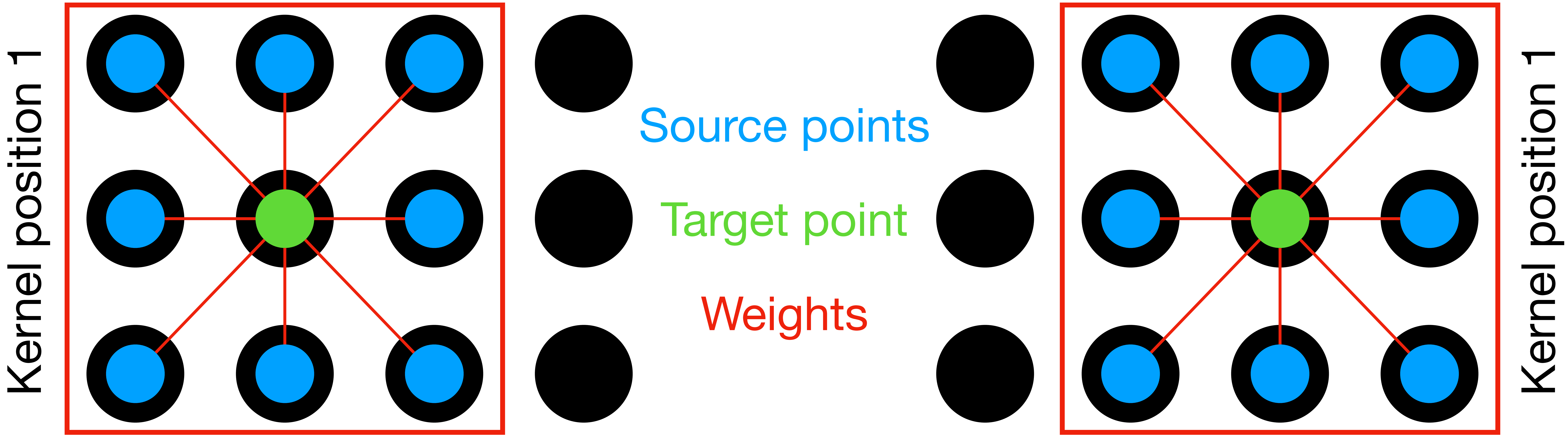
Training the kernel

- So, to train the kernel (find the weight coefficients), we can use any portion of k-space, from any contrast (with matched spatial parameters)
  - Typically (but not always), a fully-sampled central portion of k-space is acquired using the same sequence to *train* the kernel
- Also, once trained, the GRAPPA kernel can be used to fill in missing points anywhere in k-space, for any image contrast (assuming the same sensitivities and sampling geometry)
  - In practice, GRAPPA kernels are often determined on an image-specific basis due to many small differences in image properties and sequence parameters from sequence-to-sequence

GRAPPA

Training the kernel

- In the fully-sampled training k-space, often called the auto-calibration signal (ACS) or simply the calibration data, both the *target* and *source* points for a given kernel geometry are available



- Weights are typically estimated using a least-squares criterion

GRAPPA

Training the kernel

- Like SENSE, GRAPPA boils down to solving a linear problem:
  - $k_{trg} = Wk_{src}$
  - $\hat{W} = k_{trg} * [(k_{src}^H k_{src})^{-1} k_{src}^H]$, where the term inside the $[\,\cdot\,]$ is the pseudoinverse of k_{src}
- Collect all the target points together into k_{trg}, along with their corresponding source points into k_{src}, then solve for the least-squares optimal weights $\hat{W}$
- The hardest part of GRAPPA is just getting all the corresponding points together correctly across all channels with the right geometry
  - Most of the code in my GRAPPA implementation is keeping track of indices
  - Solving for the weights is then done in literally one line
- These kernel weights $\hat{W}$ can be pre-calculated, and at reconstruction time, simply applied to the under-sampled k-space source points in a sliding window type way

GRAPPA
Intuition

- Sliding a shift-invariant kernel around in k-space to fill in missing points is can be thought of as a convolution-type operation
- Recall that the Fourier transform of multiplication in the image-domain is a convolution in k-space
- Therefore, multiplication of coil sensitivities in image space is represented as a convolution in k-space with a coil sensitivity kernel
- This makes sense, since we know the kernel weights are based solely on sensitivity information, so in some sense they are a k-space representation of the coil sensitivity information

GRAPPA

Intuition

- We know convolutions with sensitivity profiles will induce local correlations – that is, neighbouring points in k-space will have some linear dependence arising from the convolution
    - These correlations are local because coil sensitivities are smooth (very limited spatial frequency support)
- Therefore, intuitively we can understand that some information for a given k-space location is already embedded in neighbouring points
- GRAPPA is simply a way for us to extract information from surrounding points in order to reconstruct a missing point

GRAPPA

Connection to null-space/low-rank methods

- Can extend the GRAPPA concept to null-space methods (e.g. PRUNO, ESPIRiT, SAKE, P-LORAKS)
- Briefly, we can rewrite the GRAPPA equation $k_{trg} = Wk_{src}$ as $0 = Wk_{src} - k_{trg}$
- We can reshape this into $0 = [W \quad -I] \begin{bmatrix} k_{src} \\ k_{trg} \end{bmatrix} = \tilde{W}k$
- Therefore, since $\tilde{W}$ and k are non-trivial, then k must have a nullspace (and therefore the data in k_{src} and k_{trg} must have some redundancy)
- We can exploit that property to facilitate reconstruction in many different ways

SENSE vs GRAPPA

SENSE	GRAPPA
Image domain	K-space domain
Requires explicit knowledge of coil sensitivity maps	Needs fully-sampled k-space, but no explicit sensitivity maps
Reconstructs a single image	Reconstructs one k-space per channel
Linear reconstruction	Linear reconstruction

Tutorials

MATLAB tutorials for SENSE and GRAPPA

- SENSE tutorial
- https://github.com/mchiew/SENSE-tutorial
- GRAPPA tutorial
- https://github.com/mchiew/grappa-tutorial

Part IV – Noise Propagation

SNR depends on acquisition time

Macovski et al., MRM 1996

- Parallel imaging often comes at a cost – there is no free lunch
- One of these costs is a $\sqrt{R}$ SNR penalty compared to a fully-sampled acquisition, where R is the acceleration factor
- So a $R = 3$ parallel imaging acquisition will result in an image with at least $\sqrt{3}$ worse SNR compared to a fully-sampled acquisition
- This is because fundamentally, SNR in MRI depends on the square root of acquisition time $\sqrt{T}$, so accelerated by factor R results in an acquisition with SNR $\sqrt{\frac{T}{R}}$

SNR depends on acquisition time

Intuition

- Consider that every time we make a measurement (i.e. a row of our encoding operator), we are sampling some aspect of the signal, and some noise
- The more time we spend, the more signal and noise samples we can acquire
- Think of the inverse Fourier transform reconstruction process like some sort of complex averaging operation
- Then acquiring for more time (or acquiring more measurements) means more noise averaging, and we know SNR scales with $\sqrt{N_{avg}}$ due to variance adding in quadrature

SNR also depends on coil sensitivities

g-Factor

- The SNR after parallel imaging also depends on the coil sensitivities
- The more orthogonal (i.e. not linearly dependent) the sensitivities are, the better SNR performance you get
- In other words, the more linearly dependent the coil sensitivities are, the more noise amplification you get in your reconstructed image
  - Intuitively, this is related to the fact that co-linear measurements are really hard to tease apart
- This factor is called the “g-factor”, where the “g” stands for (coil) geometry
- It also depends on the acceleration factor, and very quickly blows up for high R
  - Higher acceleration means more aliasing, which is a harder reconstruction problem

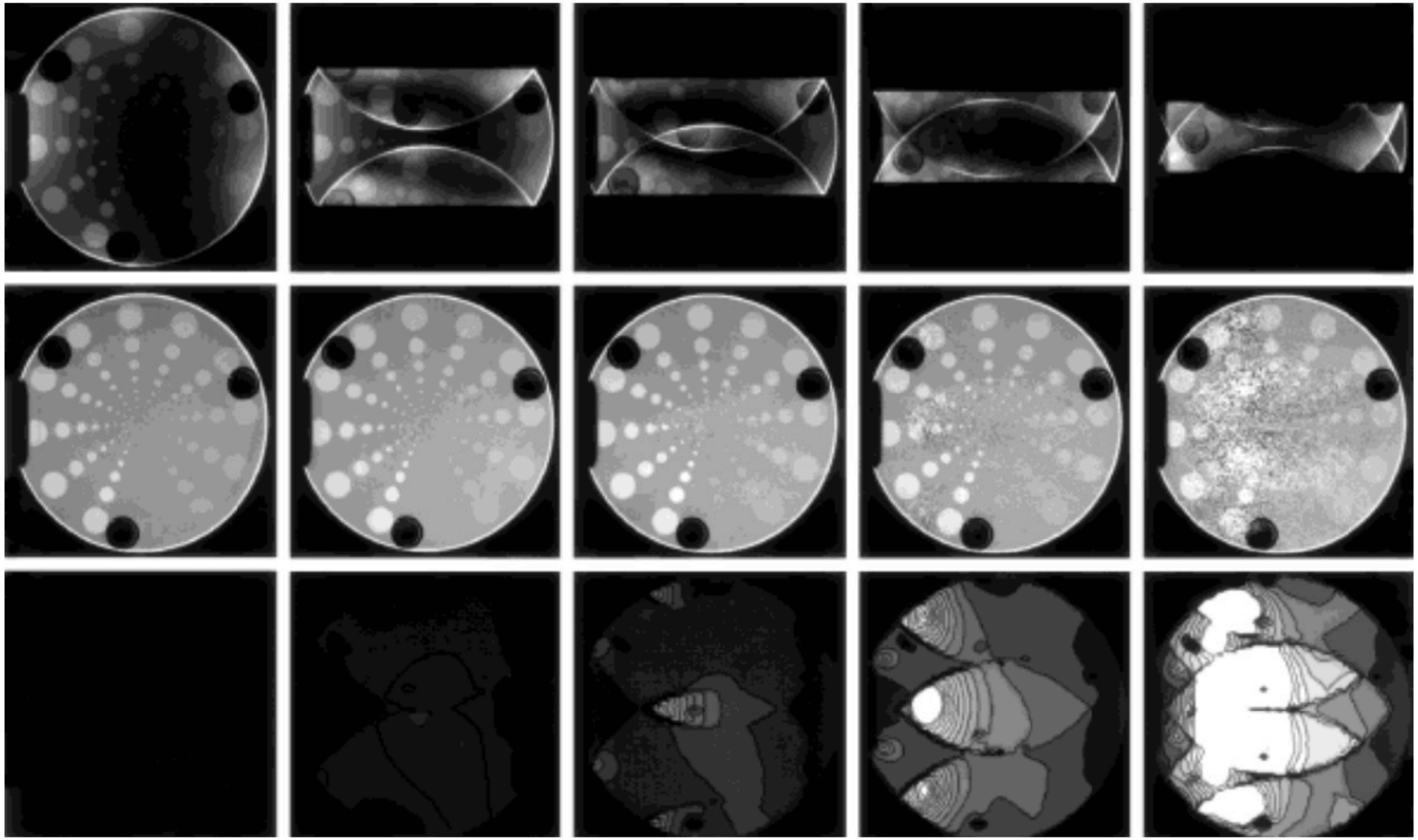
SNR also depends on coil sensitivities

g-Factor

Aliased Image

Reconstructed Image

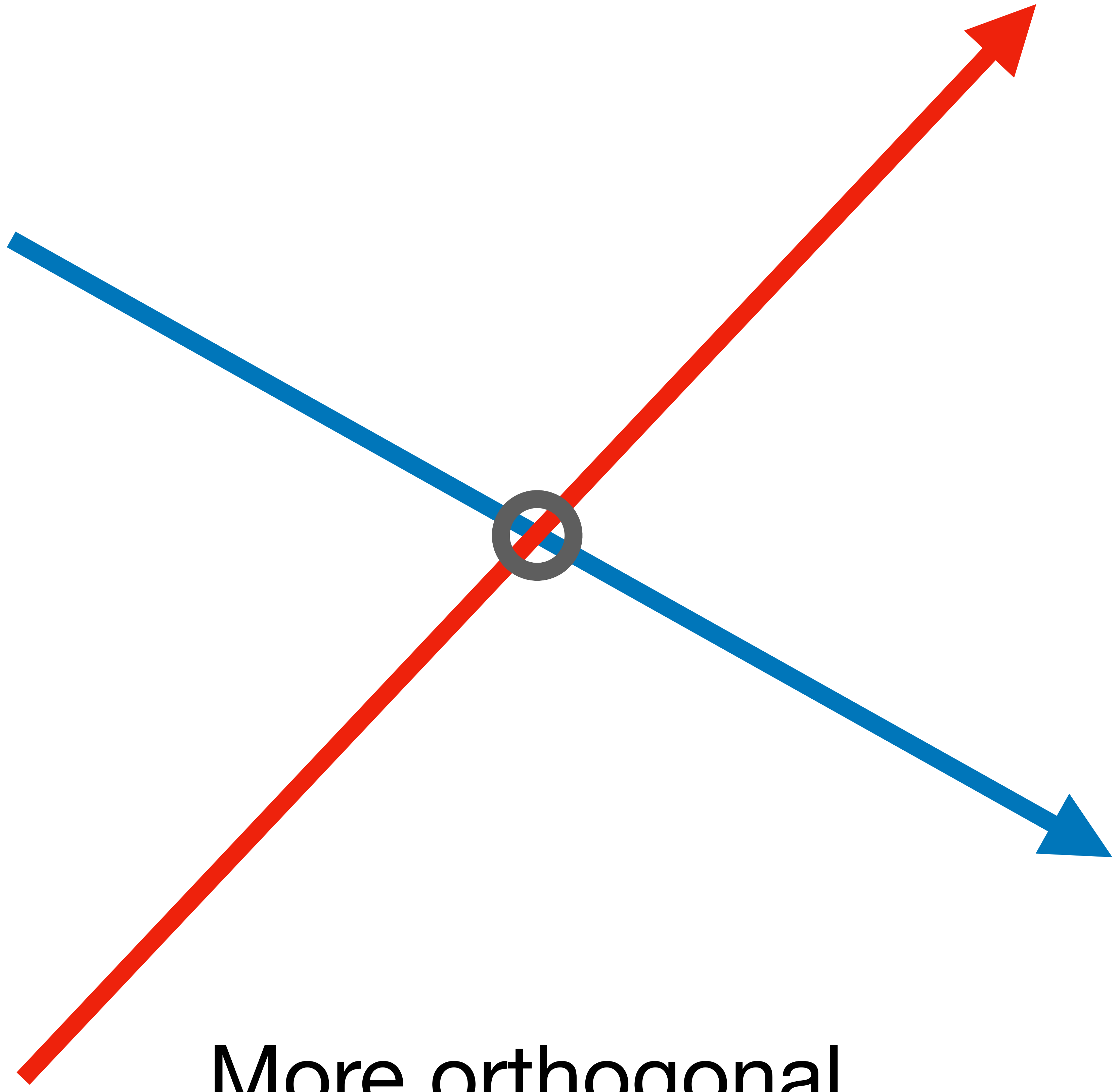
g-factor noise amplification



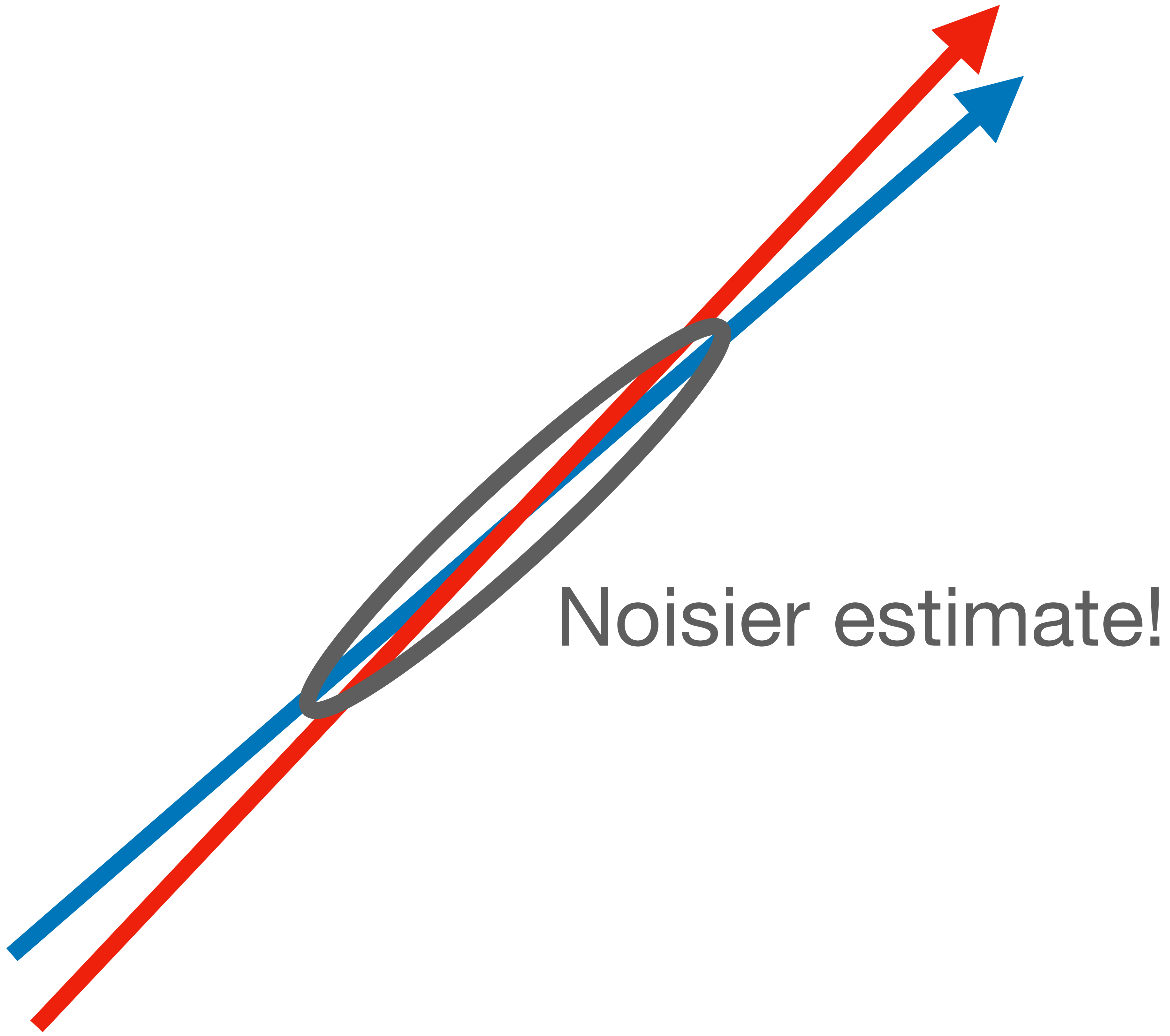
Pruessmann et al., MRM 1999

g-factor intuition

Imagine trying to localize a point at the intersection of two vectors



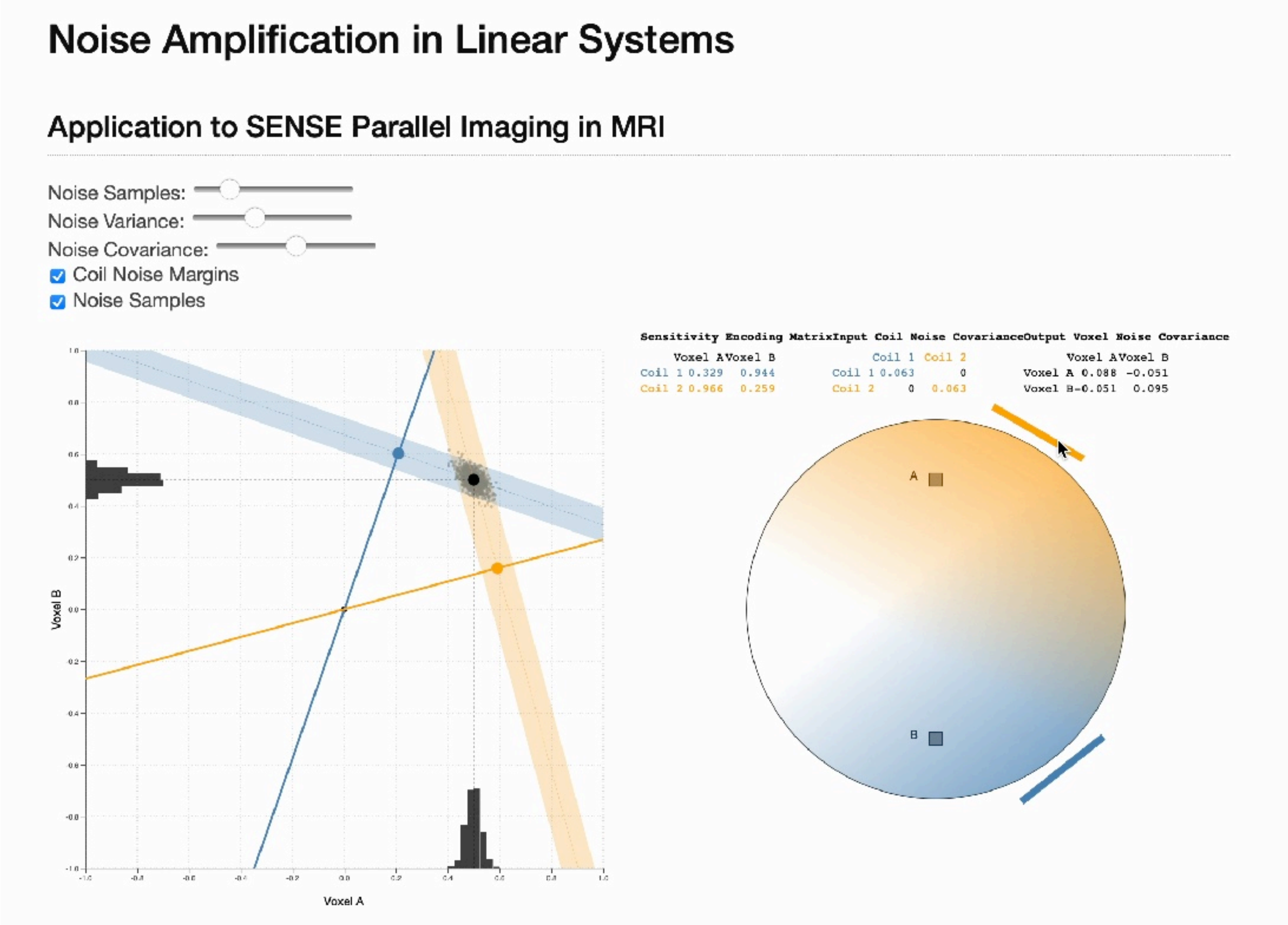
More orthogonal



More co-linear

Noise Amplification Interactive

https://mchiew.github.io/gfactor_vis.html



Correlated Noise after Unaliasing

- After Parallel Imaging reconstruction, not only do we have noise amplification due to the $\sqrt{R}$ and g-factor penalties, we also have correlated noise in our image

- Voxel locations that were aliased and subsequently separated will have correlated (non-independent) noise

- Intuitively, since the aliased voxels all shared the same noise source: $x_{aliased} = x_1 + \ldots + x_R + n$

- It is unsurprising then that the noise remains correlated in each of the unaliased voxels

Noise Propagation

Formally

- All three of the noise characteristics ($\sqrt{R}$, g-factor, correlations) can be derived by computing the variance of the linear reconstruction estimator
- Assume some encoding operator E, such that $k = Ex + n$, where $n \sim N(0,\sigma)$
- Consider a sense reconstruction of the form $\hat{x} = (E^H E)^{-1} E^H k$
- Then $M = (E^H E)^{-1} E^H$ is the linear reconstruction operator (it operates on k to estimate $\hat{x}$
- The variance of $\hat{x}$ is then computed as $var(\hat{x}) = \mathbb{E}[(\hat{x} - \bar{x})(\hat{x} - \bar{x})^H]$

Noise Propagation

Formally

- Because the noise is zero-mean, $\bar{x} = \mathbb{E}[\hat{x}] = (E^H E)^{-1} E^H (Ex) = x$
- Then $\mathbb{E}[(\hat{x} - \bar{x})(\hat{x} - \bar{x})^H] = \mathbb{E}[(\hat{x} - x)(\hat{x} - x)^H]$
- Since $\hat{x} - x = (E^H E)^{-1} E^H n$, we get $\mathbb{E}[(E^H E)^{-1} E^H n n^H E (E^H E)^{-1}]$
- Assuming whitened data ($nn^H = I$), we get $var(\hat{x}) = (E^H E)^{-1}$
- The square root of the diagonal entries of $\sqrt{(E^H E)^{-1}_{i,i}}$ tell us about the noise amplitude in each voxel (can be decomposed to $\sqrt{R} \cdot g$, $g \geq 1$)
- The off-diagonal entries tell us about the noise covariance between voxels

Noise Propagation

More informally

- If we recall that our encoding operator $E = F_R S$, with F_R being the sub-sampled Fourier transform operator at acceleration factor R, and S being the sensitivities, then $E^H E = S^H F_R^H F_R S$

- The term in the middle, $F_R^H F_R$ scales with $\frac{1}{R}$ because fewer rows in F_R means the column-wise inner products terms are summed over fewer values

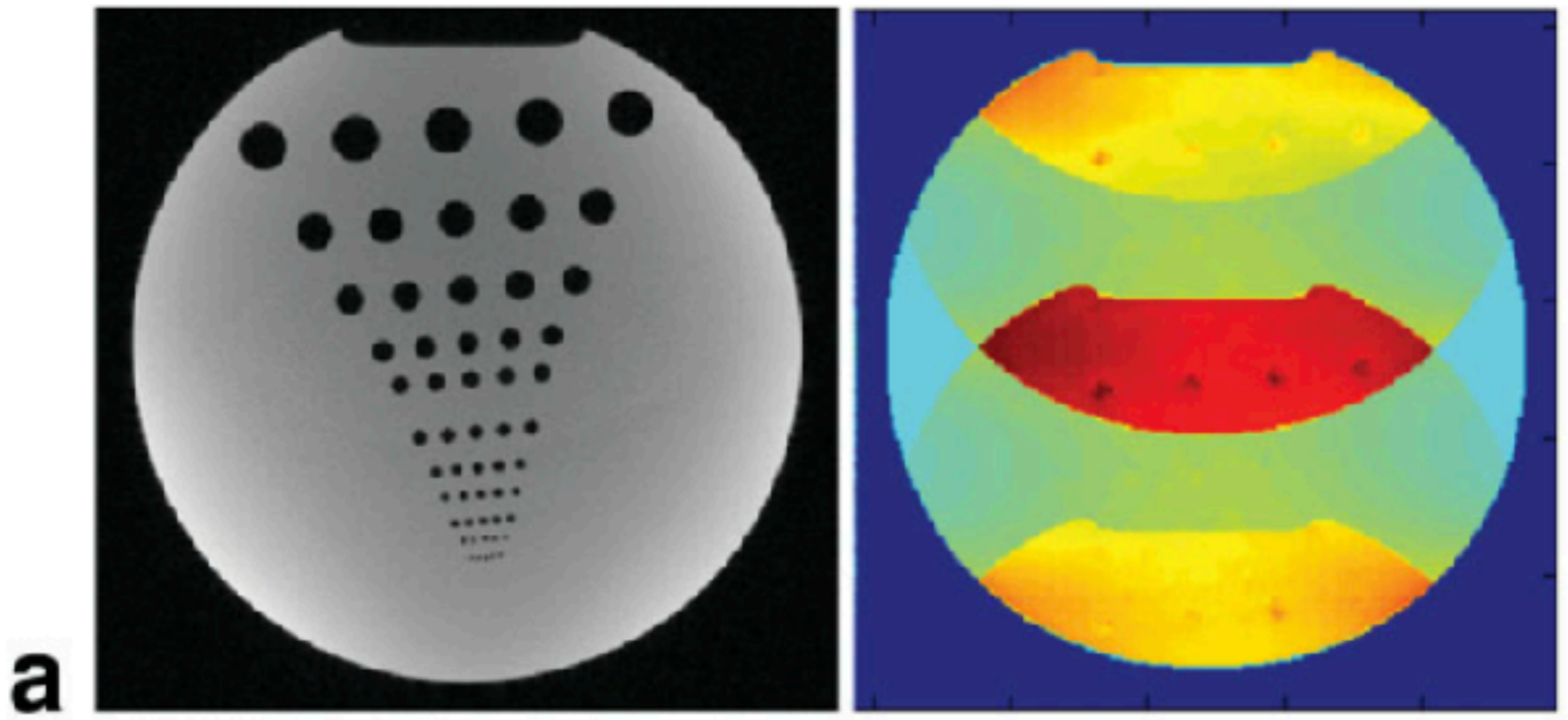
- So if $F_R^H F_R = \frac{1}{R}\Lambda$, then $(E^H E)^{-1} = R(S^H \Lambda S)^{-1}$

- Thus, we've separated the term that depends on R, and the geometry term that depends on $S^H \Lambda S$ (coil sensitivities and aliasing pattern)

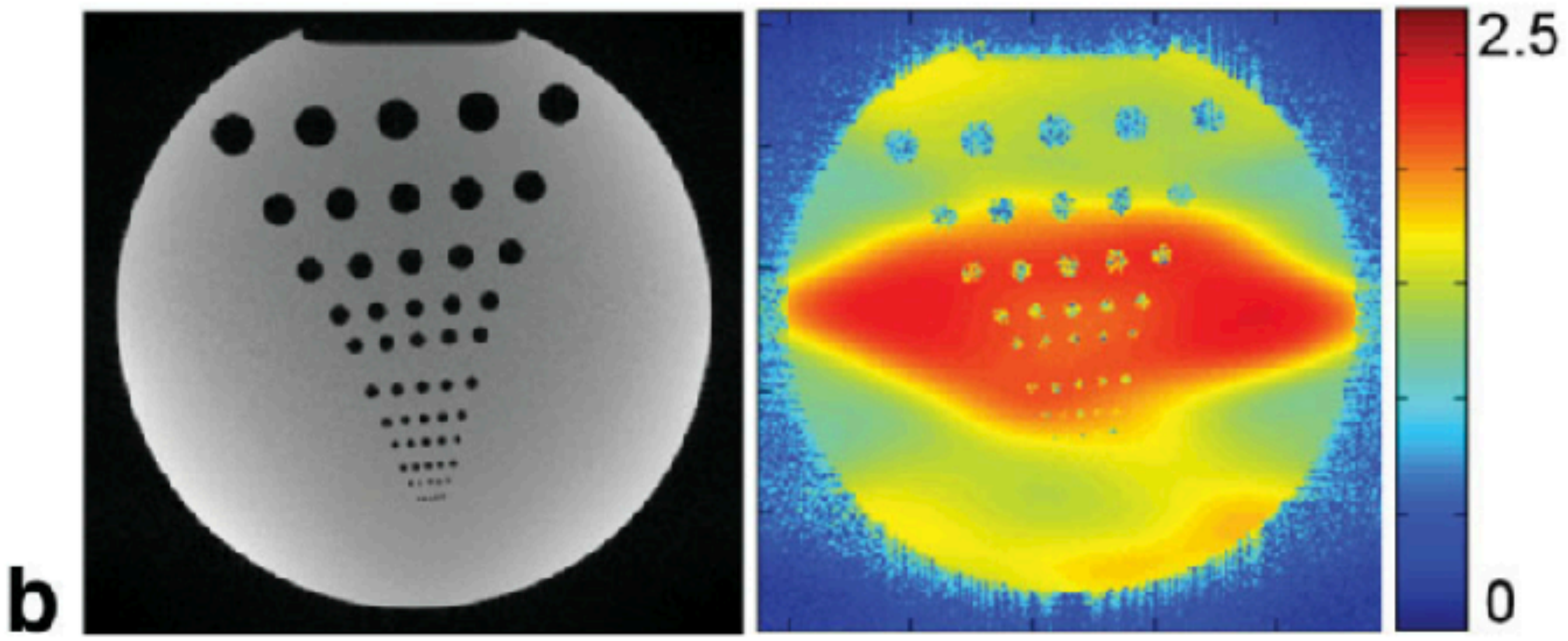
Noise Propagation

GRAPPA

- Similar analysis can be done analytically for GRAPPA
- Conceptually the noise contributions are the same: $\sqrt{R}$ and g-factor
- In practice the calculation is a bit more involved, because it involves transforming the GRAPPA kernel weights into image-domain vectors first



SENSE



GRAPPA

Breuer et al., MRM 2009

Key References

- **Multi-channel arrays, coil combination:**

Roemer, P B, W A Edelstein, C E Hayes, S P Souza, and O M Mueller. “The NMR Phased Array.” Magnetic Resonance in Medicine 16, no. 2 (November 1, 1990): 192–225.

- **SENSE, g-factors:**

Pruessmann, Klaas P, M Weiger, M B Scheidegger, and P Boesiger. “SENSE: Sensitivity Encoding for Fast MRI.” Magnetic Resonance in Medicine 42, no. 5 (November 1, 1999): 952–62.

- **GRAPPA**:

Griswold, Mark A, Peter M Jakob, Robin M Heidemann, Mathias Nittka, Vladimir Jellus, Jianmin Wang, Berthold Kiefer, and Axel Haase. “Generalized Autocalibrating Partially Parallel Acquisitions (GRAPPA).” Magnetic Resonance in Medicine 47, no. 6 (June 1, 2002): 1202–10. https://doi.org/10.1002/mrm.10171.

- **Adaptive combine coil combination weights, sensitivity estimation**

Walsh, D O, A F Gmitro, and M W Marcellin. “Adaptive Reconstruction of Phased Array MR Imagery.” Magnetic Resonance in Medicine 43, no. 5 (May 1, 2000): 682–90.

- **ESPIRiT sensitivity estimation, subspace interpretation of GRAPPA**

Uecker, M, P Lai, M J Murphy, P Virtue, M Elad, John M Pauly, S S Vasanawala, and M Lustig. “ESPIRiT - An Eigenvalue Approach to Autocalibrating Parallel MRI: Where SENSE Meets GRAPPA.” Magnetic Resonance in Medicine 71, no. 3 (January 1, 2014): 990–1001.

- **GRAPPA g-factors**

Breuer, Felix A, Stephan A R Kannengiesser, Martin Blaimer, Nicole Seiberlich, Peter M Jakob, and Mark A Griswold. “General Formulation for Quantitative G-Factor Calculation in GRAPPA Reconstructions.” Magnetic Resonance in Medicine 62, no. 3 (September 1, 2009): 739–46. https://doi.org/10.1002/mrm.22066.