EECS C145B / BioE C165: Image Processing and Reconstruction Tomography

Lecture 10

Jonathan S. Maltz

jon@eecs.berkeley.edu
http://muti.lbl.gov/145b
510-486-6744

Topics to be covered

1. Step-by-step description of analytical reconstruction algorithms involving filtering and backprojecting
2. Sampling requirements for tomographic reconstruction
3. Projection as a matrix operation
4. Backprojection as a matrix operation
5. Reconstructing using the pseudoinverse
6. Relationship between pseudoinverse method and algorithms based on the filtering and backprojecting
7. Introduction to iterative methods

Reading

Assigned reading:

- Course Reader, pp. 21-27, 29, 88-103.

Additional reading:

- Cho, "Foundations of Medical Imaging", John Wiley and Sons (1993), Chapter 3.
- Budinger, "A Primer on Reconstruction Algorithms" (handout), pp. 1-11, 13-26

Advanced reading:

- Natterer, "The Mathematics of Computerized Tomography", John Wiley and Sons (1986), Chapters I and II.

Backprojection of convolution-filtered projections (BCFP)

This algorithm is more commonly referred to as the **convolution backprojection** algorithm.

Given a set of I angular projections $p(\boldsymbol{\theta}_i, s)$ reconstruct $\tilde{f}(\mathbf{x})$, an approximation of $f(\mathbf{x})$ by:

1. Taking the derivatives:

$$p'(\boldsymbol{\theta}_i, s) = \frac{dp(\boldsymbol{\theta}_i, s)}{ds} \qquad i = 1, 2, \ldots, I$$

2. Applying Hilbert transform to each $p'(\boldsymbol{\theta}_i, s)$:

$$\tilde{p}(\boldsymbol{\theta}_i, s) = p'(\boldsymbol{\theta}_i, s) * \frac{1}{2\pi^2 s}$$

Backprojection of convolution-filtered projections (BCFP)

3. Backprojecting all I filtered projections:

$$\tilde{f}(\mathbf{x}) = \mathrm{BP}\Big\{\tilde{p}(\boldsymbol{\theta}_i, s)\Big\}$$

   to get the reconstructed distribution.

Note: Steps (1) and (2) can be replaced by the step:

$$\tilde{p}(\boldsymbol{\theta}_i, s) = p(\boldsymbol{\theta}_i, s) * h(s)$$

where $h(s)$ is the inverse FT of any of the modified ramp filters.

Backprojection of Fourier-filtered projections (BPFP)

This algorithm is more commonly referred to as the **filtered backprojection** algorithm.

Given a set of I angular projections $p(\boldsymbol{\theta}_i, s)$ reconstruct $\tilde{f}(\mathbf{x})$, an approximation of $f(\mathbf{x})$ by:

1. Transforming each projection using the 1D FT:

$$P(\boldsymbol{\theta}_i, w) = \mathcal{F}_1\Big\{p(\boldsymbol{\theta}_i, s)\Big\} \qquad i = 1, 2, \ldots, I$$

2. Multiplying each $P(\boldsymbol{\theta}_i, s)$ with the ramp filter or modified ramp filter $H(w)$:

$$\tilde{P}(\boldsymbol{\theta}_i, w) = P(\boldsymbol{\theta}_i, w) \times H(w)$$

Backprojection of Fourier-filtered projections (BPFP)

3. Applying the inverse 1D FT to each Fourier domain projection:

$$\tilde{p}(\boldsymbol{\theta}_i, s) = \mathcal{F}_1^{-1}\Big\{\tilde{P}(\boldsymbol{\theta}_i, w)\Big\}$$

4. Backprojecting all I filtered projections:

$$\tilde{f}(\mathbf{x}) = \mathrm{BP}\Big\{\tilde{p}(\boldsymbol{\theta}_i, s)\Big\}$$

   to get the reconstructed distribution.

Fourier-filtered backprojection algorithm (FFBP)

Given a set of I angular projections $p(\boldsymbol{\theta}_i, s)$ reconstruct $\tilde{f}(\mathbf{x})$, an approximation of $f(\mathbf{x})$ by:

1. Backprojecting all the known projections:

$$f_{\mathrm{BP}}(\mathbf{x}) = \mathrm{BP}\Big\{p(\boldsymbol{\theta}_i, s)\Big\} \qquad i = 1, 2, \ldots, I$$

2. Fourier transforming the backprojection image using the 2D FT:

$$F_{\mathrm{BP}}(\boldsymbol{\xi}) = \mathcal{F}_2\Big\{f_{\mathrm{BP}}(\mathbf{x})\Big\}$$

Fourier-filtered backprojection algorithm (FFBP)

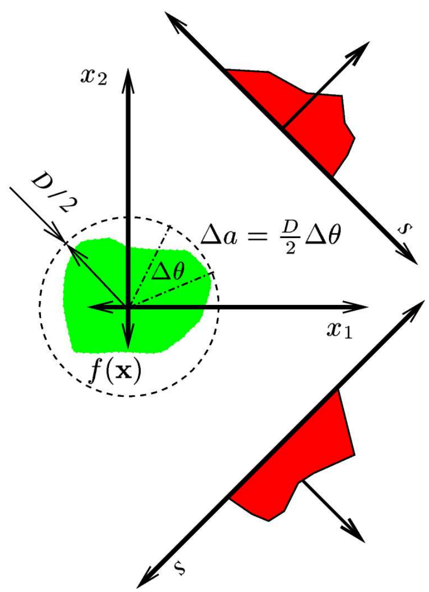
4. Multiplying $F_{\mathrm{BP}}(\boldsymbol{\xi})$ by the cone filter $H(\boldsymbol{\xi}) = \|\boldsymbol{\xi}\|$ or a modified version thereof:
$$\tilde{F}(\boldsymbol{\xi}) = F_{\mathrm{BP}}(\boldsymbol{\xi}) \times H(\boldsymbol{\xi})$$
5. Applying the inverse 2D FT to the filtered transform:
$$\tilde{f}(\mathbf{x}) = \mathcal{F}_2^{-1}\left\{\tilde{F}(\boldsymbol{\xi})\right\}$$

Space-filtered backprojection algorithm (SFBP)

Given a set of I angular projections $p(\boldsymbol{\theta}_i, s)$ reconstruct $\tilde{f}(\mathbf{x})$, an approximation of $f(\mathbf{x})$ by:

1. Backprojecting all the known projections:
$$f_{\mathrm{BP}}(\mathbf{x}) = \mathrm{BP}\left\{p(\boldsymbol{\theta}_i, s)\right\} \qquad i = 1, 2, \ldots, I$$
2. Inverse Fourier transforming the ideal or modified cone filter $H(\boldsymbol{\xi})$ to get its 2D point-spread function $h(\mathbf{x})$.
3. Convolving (in 2D) $f_{\mathrm{BP}}(\mathbf{x})$ with the PSF $h(\mathbf{x})$ to get the reconstructed image:
$$\tilde{f}(\mathbf{x}) = f_{\mathrm{BP}}(\mathbf{x}) * h(\mathbf{x})$$

Sampling considerations



Sampling considerations: Angular sampling requirements

- We now address the sampling requirements for a discrete imaging tomographic system that produces I **regularly spaced** projections over 180 degrees. We assume that the highest spatial frequency (along all directions) inside the imaged distribution $f(\mathbf{x})$ is $\rho_{\max}$.
- Consider an object that can be enclosed by a circle of diameter D centered at the center of camera rotation.
- Let $\Delta\theta$ represent the angular spacing between successive projections. Then the arc subtended by $\Delta\theta$ on the circle has length $\Delta a = \frac{D}{2}\Delta\theta$.
- This arc spans the **longest distance between adjacent samples**. To prevent aliasing, we must ensure that this arc is sampled at **over twice the Nyquist frequency** $\rho_{\max}$.

Sampling considerations: Angular sampling requirements

- To satisfy the sampling theorem, we must thus choose $\Delta\theta$ to satisfy:

$$\frac{1}{\Delta a} \quad = \quad 2\rho_{\max}$$

Substituting our expression for Δa gives:

$$\frac{2}{D\Delta\theta} \quad = \quad 2\rho_{\max}$$

and so we must have

$$\Delta\theta < \frac{1}{\rho_{\max} D}$$

- Since we do not typically know what $\rho_{\max}$ is for the distribution $f(\mathbf{x})$ (this is why we're trying to look inside it in the first place), we set $\rho_{\max}$ to the resolution of the imaging system. With this choice, we avoid all aliasing other than that aliasing that is unavoidable. (Remember that it is not possible to antialias filter the projections before we sample them, so detail smaller than twice the instrument's sampling interval will be aliased).

Sampling considerations: Radial sampling requirements

- When a camera samples a single angular projection, it collects the photons into **bins**. Each bin often corresponds physically to a single detector.
- We will assume that we have J bins along the s-axis.
- If the highest frequency along any direction in the distrubution is $\rho_{\max}$, then we need to make sure that the interbin spacing Δs satisfies:

$$\Delta s < \frac{1}{2\rho_{\max}}$$

Sampling considerations: Sampling requirements

- What is the minimum number of bins that we need?

$$J = \left\lceil \frac{D}{\Delta s} \right\rceil = \left\lceil 2D\,\rho_{\max} \right\rceil$$

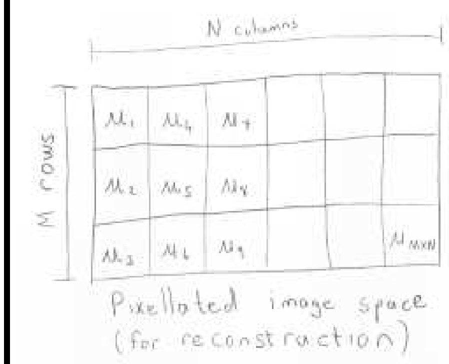
where $\lceil\cdot\rceil$ represents the "ceiling" operator (finds the next highest integer for non-integer arguments and the same integer for integer arguments)

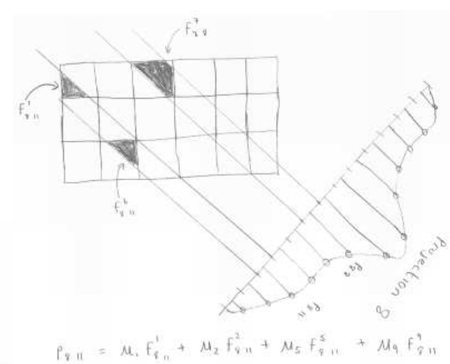
- Based on the earlier discussion, what is the minimum number of angles needed if we sample over 180 degrees (as we generally do in transmission tomography)?

$$I = \left\lceil \frac{\pi}{\Delta\theta} \right\rceil = \left\lceil \pi D\,\rho_{\max} \right\rceil$$

Discrete Radon transform as a matrix operation

Because each projection bin measurement can be expressed in terms of a linear combination of the pixel values μ_n (on the grid on which the image is reconstructed), we can represent the discrete projection operation as a matrix operation.





Discrete Radon transform as a matrix operation

- The quantities f_{ij}^n are called **geometric weighting factors**. It is proportional to the **area of intersection** of the backprojection of the jth bin of the ith projection, with the nth pixel.
- The value f_{ij}^n represents the contribution of the "mass" of nth pixel to the jth bin of the ith projection.
- Consequently, there are $N \times M \times I \times J$ weighting factors.
- We will now show how these factors can be placed in a matrix $\mathbf{F}$ so that the vector of projection measurements $\mathbf{p}$ can be expressed in terms of the pixel values as:

$$\mathbf{p} = \mathbf{F}\boldsymbol{\mu}$$

Discrete Radon transform as a matrix operation

$$\begin{bmatrix} p_{11} \\ p_{12} \\ \vdots \\ p_{1J} \\ p_{21} \\ p_{22} \\ \vdots \\ \vdots \\ p_{I1} \\ p_{I2} \\ \vdots \\ p_{IJ} \end{bmatrix} = \begin{bmatrix} f_{11}^1 & f_{11}^2 & \cdots & \cdots & f_{11}^{N\times M} \\ f_{12}^1 & f_{12}^2 & \cdots & \cdots & f_{12}^{N\times M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{1J}^1 & f_{1J}^2 & \cdots & \cdots & f_{1J}^{N\times M} \\ f_{21}^1 & f_{21}^2 & \cdots & \cdots & f_{21}^{N\times M} \\ f_{22}^1 & f_{22}^2 & \cdots & \cdots & f_{22}^{N\times M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{I1}^1 & f_{I1}^2 & \cdots & \cdots & f_{I1}^{N\times M} \\ f_{I2}^1 & f_{I2}^2 & \cdots & \cdots & f_{I2}^{N\times M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{IJ}^1 & f_{IJ}^2 & \cdots & \cdots & f_{IJ}^{N\times M} \end{bmatrix} \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \mu_{N\times M} \end{bmatrix}$$

Discrete Radon transform as a matrix operation

To allow matrix expression of the projection (discrete Radon transform) operator, we have:

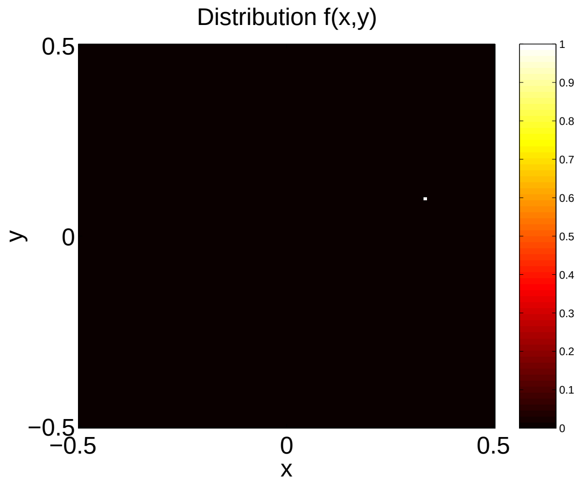
1. Stacked the N columns of the image pixel values on top of each other to form the $N \times M$ vector $\boldsymbol{\mu}$
2. Stacked the I angular projections of J bins each into the $I \times J$ vector $\mathbf{p}$.
3. Placed f_{ij}^n at row $j + (i-1) \times J$ and column n of $\mathbf{F}$.

- $\mathbf{F}$ is in practice a very large matrix of dimension $IJ \times MN$. If we have a 512×512 image to reconstruct and 512 projections of 512 bins each, $\mathbf{F}$ has dimension 262144×262144. This corresponds to 68 billion elements, requiring 549 gigabytes of memory for double precision storage.
- Because relatively few pixels contribute to each bin, $\mathbf{F}$ contains mostly zeros. It is a **sparse** matrix and so only the non-zero elements are stored in practice.

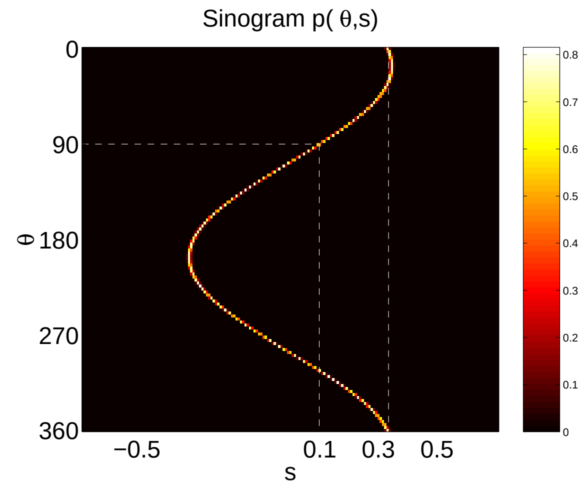
Discrete Radon transform as a matrix operation

Five minute exercise: Try to work out how we can automatically calculate the f_{ij}^n for a specific imaging geometry.

Discrete Radon transform as a matrix operation



Discrete Radon transform as a matrix operation



Discrete backprojection as a matrix operation

- We will now show that backprojection of a set of discrete projections can be found using:

$$\boldsymbol{\mu}_{\mathrm{BP}} = \mathbf{F}^T \mathbf{p}$$

- We will see that the nth row of $\mathbf{F}^T$ gives us **all** the weighting factors necessary to calculate the contributions to the nth pixel of **each of the projection bins**
- Consequently, when we multiply the nth row by the projection vector $\mathbf{p}$, we get the **intersection-area-weighted sum** of the contributions of all the bins to the nth pixel. This is, **by definition**, backprojection.

Discrete backprojection as a matrix operation

$$\begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \mu_{N\times M} \end{bmatrix} = \begin{bmatrix} f_{11}^1 & \cdots & f_{1J}^1 & \cdots & f_{I1}^1 & \cdots & f_{IJ}^1 \\ f_{11}^2 & \cdots & f_{1J}^2 & \cdots & f_{I1}^2 & \cdots & f_{IJ}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ f_{11}^{N\times M} & \cdots & f_{1J}^{N\times M} & \cdots & f_{I1}^{N\times M} & \cdots & f_{IJ}^{N\times M} \end{bmatrix} \begin{bmatrix} p_{11} \\ \vdots \\ p_{1J} \\ \vdots \\ p_{I1} \\ \vdots \\ p_{IJ} \end{bmatrix}$$

Pseudoinverse reconstruction algorithm

- Since we now have a set of simultaneous equations for the discrete Radon transform (forward problem):

$$\mathbf{p} = \mathbf{F}\boldsymbol{\mu}$$

we are in a position to reconstruct the distrubution $\boldsymbol{\mu}$ using the pseudoinverse (inverse problem):

$$\tilde{\boldsymbol{\mu}} = \mathbf{F}^{+}\mathbf{p} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{p}$$

- It is instructive to identify $\mathbf{F}^T\mathbf{p}$ with the backprojection image giving:

$$\tilde{\boldsymbol{\mu}} = (\mathbf{F}^T\mathbf{F})^{-1}\,\mathrm{BP}\{\mathbf{p}\} = (\mathbf{F}^T\mathbf{F})^{-1}\,\boldsymbol{\mu}_{\mathrm{BP}}$$

where $\boldsymbol{\mu}_{\mathrm{BP}} \triangleq \mathrm{BP}\{\mathbf{p}\}$ is the backprojection image stacked in a column vector.

Pseudoinverse reconstruction algorithm

- Remember that in the space-filtered backprojection (SFBP) algorithm we had:

$$\tilde{\boldsymbol{\mu}} = h(\mathbf{x}) * \mathrm{BP}\{\mathbf{p}\}$$

where $h(\mathbf{x})$ was the inverse FT of the cone filter or modifed version thereof.

- Now recall also that we showed that the 2D convolution operation could be expressed in matrix form as:

$$\mathbf{f} = \mathbf{H}\,\mathbf{g}$$

where $\mathbf{H}$ was a block Toeplitz matrix.

- So, we can then further identify:

$$\begin{aligned} \mathbf{H} &= (\mathbf{F}^T\mathbf{F}) \\ \mathbf{f} &= \boldsymbol{\mu}_{\mathrm{BP}} \\ \mathbf{g} &= \tilde{\boldsymbol{\mu}} \end{aligned}$$

- Therefore:

$$\boldsymbol{\mu}_{\mathrm{BP}} = (\mathbf{F}^T\mathbf{F})\tilde{\boldsymbol{\mu}}$$

and, as we had before:

$$\tilde{\boldsymbol{\mu}} = (\mathbf{F}^T\mathbf{F})^{-1}\,\boldsymbol{\mu}_{\mathrm{BP}}$$

- We can interpret:

$$\boldsymbol{\mu}_{\mathrm{BP}} = (\mathbf{F}^T\mathbf{F})\,\tilde{\boldsymbol{\mu}}$$

as a convolution operation that yields the backprojection image as the convolution of the true reconstruction with a 2D filter $h'(\mathbf{x})$. The block Toeplitz matrix $\mathbf{H} = (\mathbf{F}^T\mathbf{F})$ contains samples of $h'(\mathbf{x})$ and effects this convolution.

- This enables us to interpret:

$$\tilde{\boldsymbol{\mu}} = \mathbf{H}^{-1}\,\boldsymbol{\mu}_{\mathrm{BP}} = (\mathbf{F}^T\mathbf{F})^{-1}\,\boldsymbol{\mu}_{\mathrm{BP}}$$

as a deconvolution operation, where we deblur the backprojection image by deconvolving it and the blurring kernel $h'(\mathbf{x})$, where $H'(\boldsymbol{\xi}) = 1/H(\boldsymbol{\xi})$ (the inverse of the deblurring cone filter).

Pseudoinverse reconstruction algorithm

- If the pseudoinverse and SFBP algorithms are equivalent, then $(\mathbf{F}^T\mathbf{F})$, should be block Toeplitz, which indeed it is.
- We conclude that the pseudoinverse has a special interpretation as the matrix that carries out the **deconvolution** of the backprojection image and the PSF of the blurring kernel.
- This is equivalent to convolving the backprojection image with the PSF of the cone filter. This is identical to the SFBP algorithm.

Pseudoinverse reconstruction algorithm