FONDAMENTI DI FISICA MEDICA

PARTE 1: BASI FISICHE DELLA RADIOLOGIA (1CFU)

LECTURE 4
COMPUTED TOMOGRAPHY (CT)

Luigi Rigon
University of Trieste and INFN

Computed Tomography

- Part I - From projections to slices
- Part II – Iterative Methods
- Part III – Back projection: LSBP and FBP
- Appendix: calculating the Ram-Lak filter
- Sources:
  - The Physics of medical imaging / edited by Steve Webb (1988)
  - The essential physics of medical imaging / Jerrold T. Bushberg ... [et al.] (2012)

Part I - From projections to slices

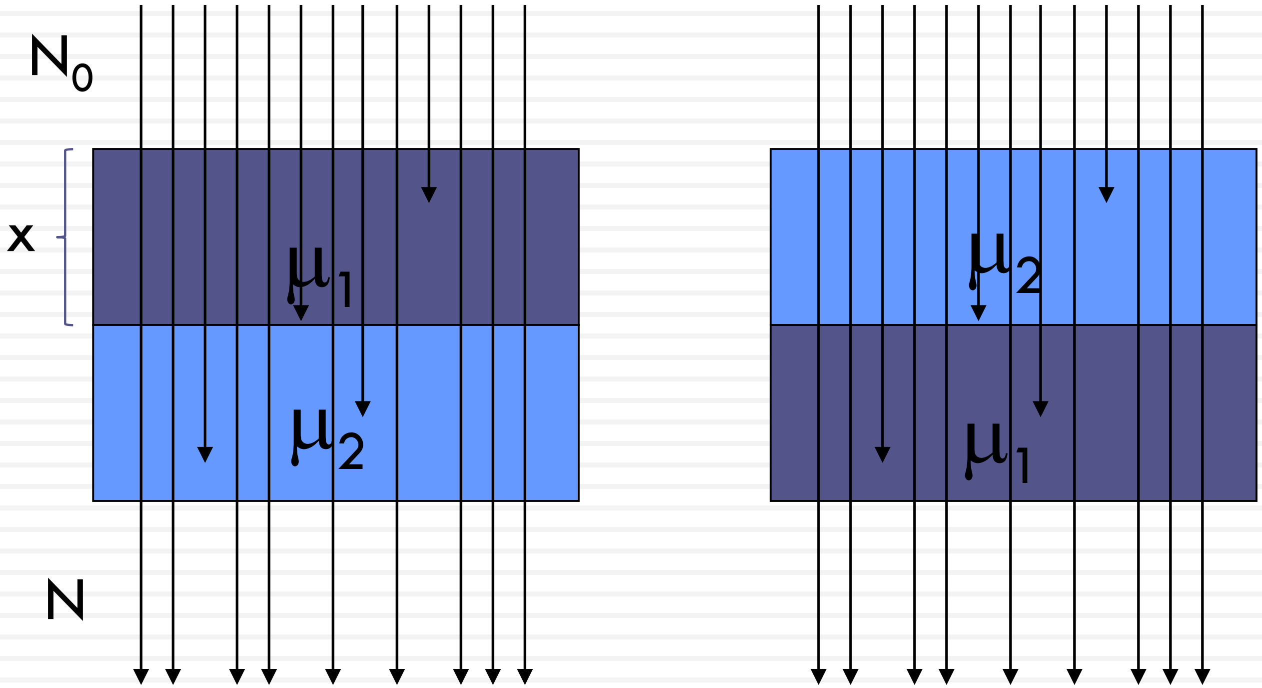
- The essential physics of medical imaging
  Jerrold T. Bushberg ... [et al.] (2012)

Why tomography?

- Planar radiography is inadequate in
  - visualize structures along the direction of x-ray propagation
  - distinguish soft tissues (with similar absorption coefficients), which contrast is of the order of 2% or smaller
- Computed Tomography can overcome these two difficulties. The general principle is always based on "cutting" narrow slices (the name tomography comes from the Greek word "tomo", to cut).
- Actually different methods can be called tomographic imaging (namely image of a <u>section</u>). Here we focus on (Axial) Computed Tomography, or simply CT

Limitations of planar radiography

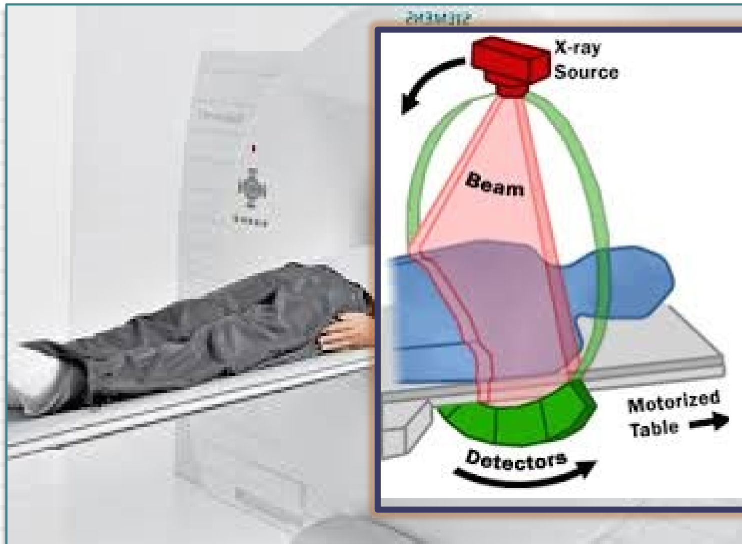
- Lack of depth information



- In both cases we have

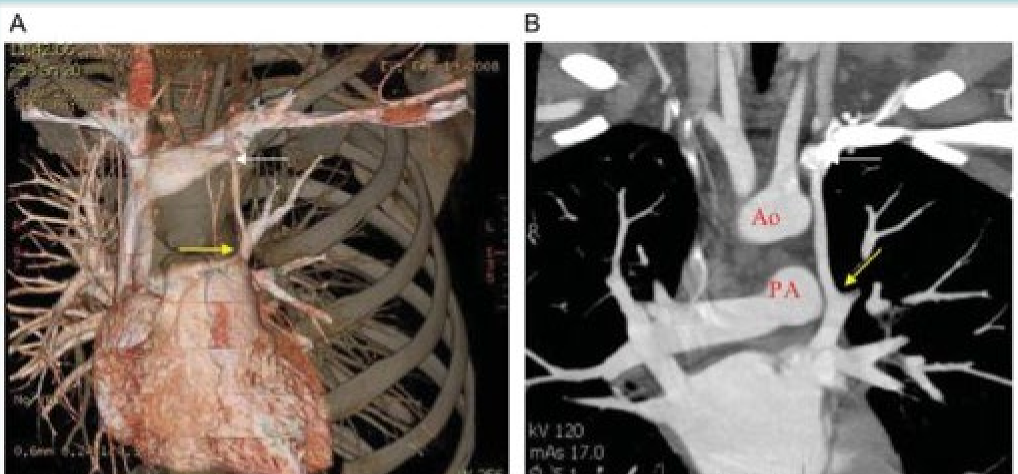
$$N = N_0 e^{-\mu_1 x} e^{-\mu_2 x} = N_0 e^{-\mu_2 x} e^{-\mu_1 x} = N_0 e^{-(\mu_1 + \mu_2) x}$$

Axial Computed Tomography (CT)



Several (~ 100) different projections are acquired, while x-ray source and detectors rotate around an axis corresponding to the longitudinal direction of the patient

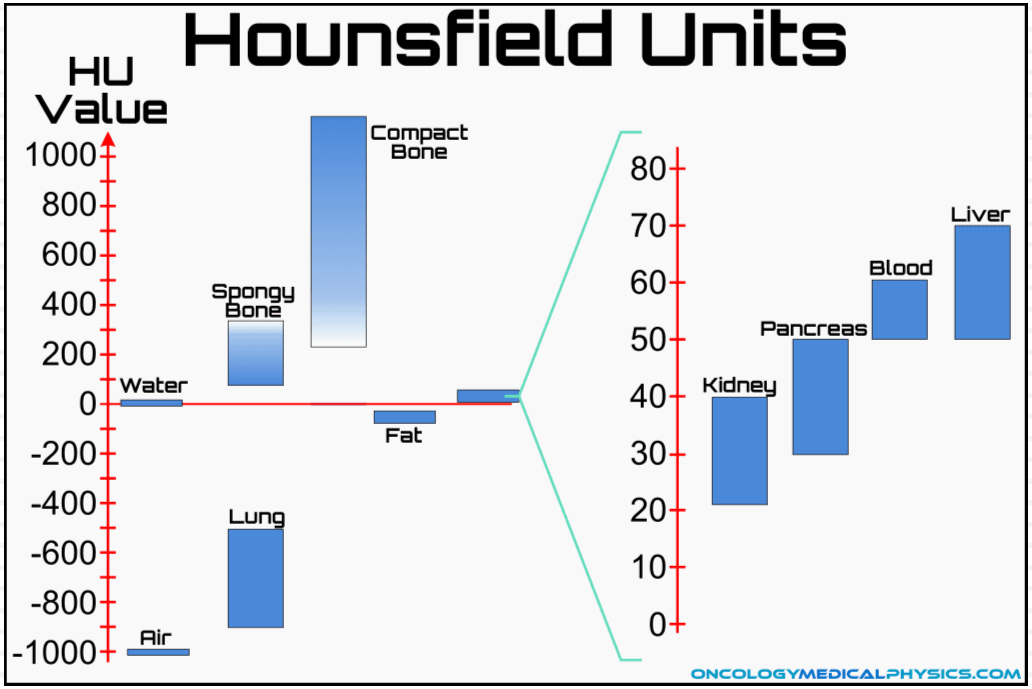
The digital projection data are used to obtain a 2D image of the irradiated slice



Stacking several different slices allows to obtain 3D images using volume rendering (A) and to virtually cut additional 2D images on planes oriented along any desired direction (e.g. B)

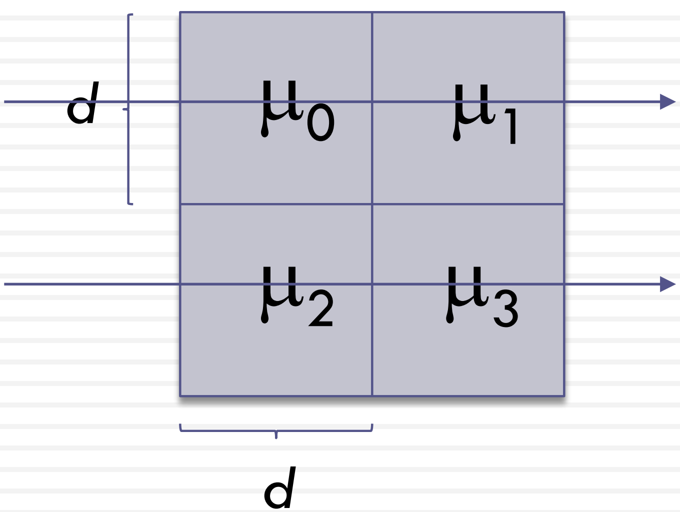
CT images and Hounsfield Units

- CT images are essentially maps of the linear attenuation coefficient (μ) of each voxel.
- In clinics, μ is measured in Hounsfield Units sometimes also called CT numbers: $HU_{(x,y,z)} = 1000\frac{\mu_{(x,y,z)} - \mu_{water}}{\mu_{water}}$



Taking projections means making sums

- Consider a 2x2 pixel axial slice with:
  - square pixels *d x d* in size
  - unknown linear attenuation coefficients
- Horizontal projections are as follows:



$$N = N_0 e^{-\mu_0 d} e^{-\mu_1 d} = N_0 e^{-(\mu_0 + \mu_1) d}$$

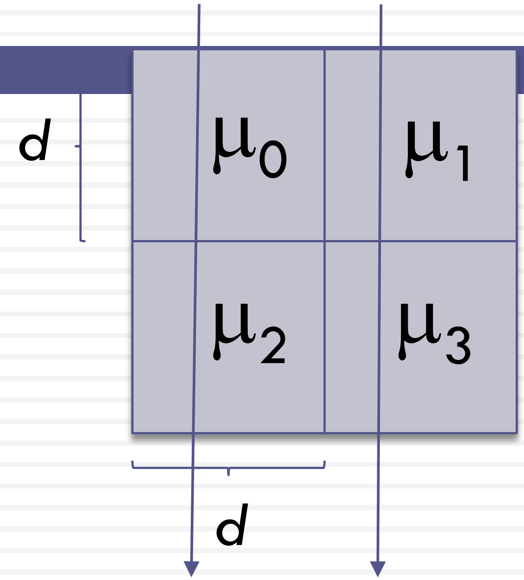
$$-\frac{1}{d} \ln\left(\frac{N}{N_0}\right) = \mu_0 + \mu_1$$

$$N = N_0 e^{-\mu_2 d} e^{-\mu_3 d} = N_0 e^{-(\mu_2 + \mu_3) d}$$

$$-\frac{1}{d} \ln\left(\frac{N}{N_0}\right) = \mu_2 + \mu_3$$

Taking projections means making sums

- The same applies for vertical projections:



$$N = N_0 e^{-\mu_0 d} e^{-\mu_2 d}$$
$$= N_0 e^{-(\mu_0 + \mu_2) d}$$

$$N = N_0 e^{-\mu_1 d} e^{-\mu_3 d}$$
$$= N_0 e^{-(\mu_1 + \mu_3) d}$$

$$-\frac{1}{d} \ln\left(\frac{N}{N_0}\right) = \mu_0 + \mu_2$$

$$-\frac{1}{d} \ln\left(\frac{N}{N_0}\right) = \mu_1 + \mu_3$$

- …and for all directions

From projections to slices

- The mathematical problem of reconstructing the images of the slices from the projections is:
  - an inverse problem
  - an ill-posed problem
    (a unique and stable solution is not guaranteed)
- In principle the problem can be modeled as a linear system of n^2 unknowns, where n is the number of rows (and columns) of the digital image
- A simple example with $n=2$ is shown in the following

From projections to slices

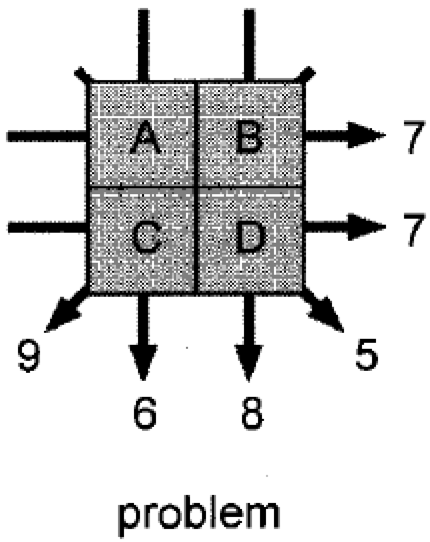


FIGURE 13-27. The mathematical problem posed by computed tomographic (CT) reconstruction is to calculate image data (the pixel values—A, B, C, and D) from the projection values *(arrows)*. For the simple image of four pixels shown here, algebra can be used to solve for the pixel values. With the six equations shown, using substitution of equations, the solution can be determined as illustrated. For the larger images of clinical CT, algebraic solutions become unfeasible, and filtered backprojection methods are used.

From projections to slices

- In the simple example with $n=2$ the algebraic approach is simple and effective
- However, with typical values of $n=512$ or $n=1024$, this approach may be unpractical
- Two practical approaches are:
  - Iterative methods (where the algebraic approach is applied iteratively – computationally demanding)
  - filtered back projection (an entirely different approach that can be applied also with relatively small computer power)

Part II – Iterative methods in CT

- The Physics of medical imaging / edited by Steve Webb (1988)

Iterative methods

- Basic idea: the section consists of an array of unknowns, and we set up algebraic equations for the unknowns in terms of the measured projection data.
- This can be applied also when:
  - it is not possible to measure a large number of projections, or
  - the projections are not uniformly distributed over 180° (or 360°)

  both these conditions being necessary requirements for the back-projections techniques to produce results with the accuracy desired in medical imaging.

Iterative methods

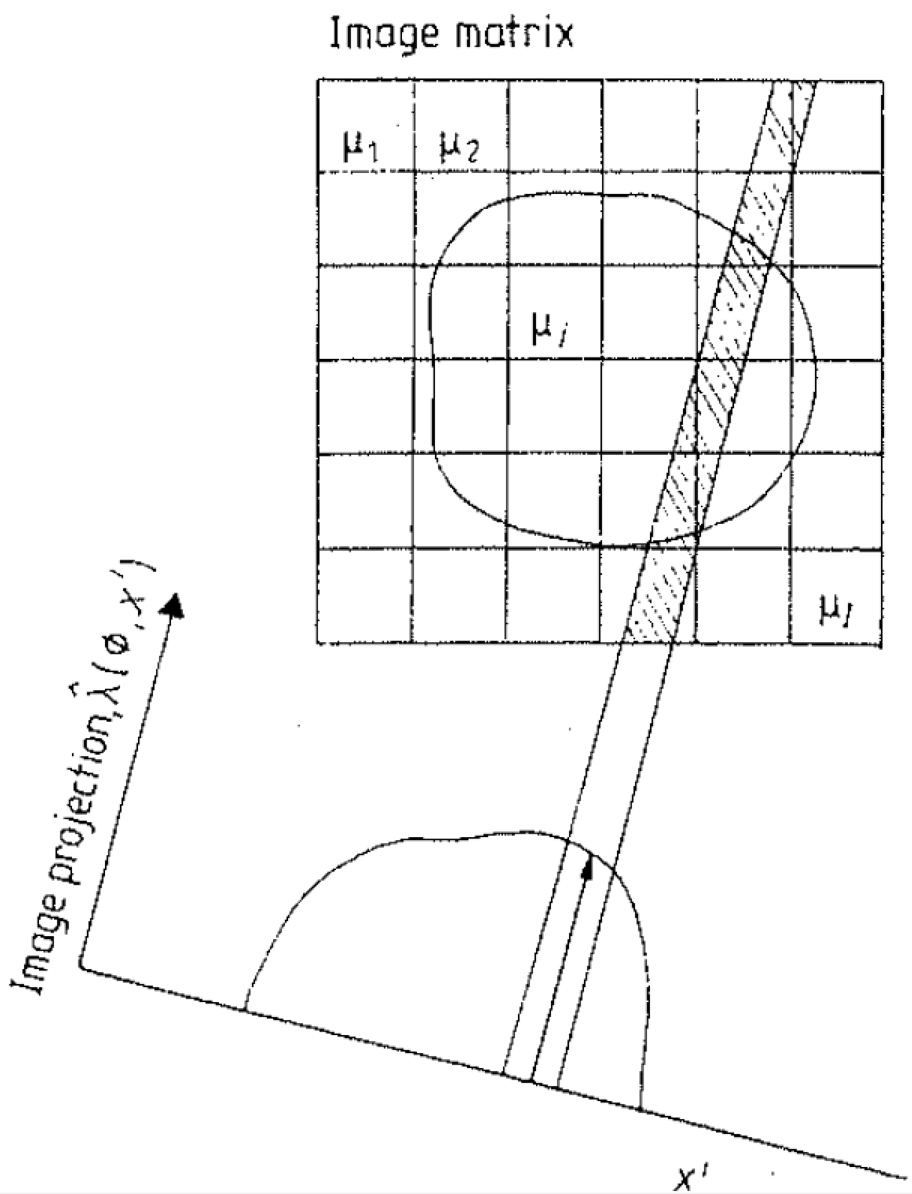
- Recently, the fast development of computers has allowed to approach the reconstruction problem in CT by means of iterative methods, obtaining sufficient accuracy and speed of implementation.
- Iterative methods are of increasing importance. For instance, they can be used when the data are incomplete or affected by a large background noise, as in the case of
  - Low-dose Tomography
  - Emission Tomography
- Various variants of iterative method exist, e.g.
  - **ART**- **A**lgebraic **R**econstruction **T**echnique
  - **SIRT** - **S**imultaneous **I**terative **R**econstruction **T**echnique
  - **SART** - **S**imultaneous **A**lgebraic **R**econstruction **T**echnique
  - **ILST** - **I**terative **L**east **S**quare **T**echnique

Iterative methods

- The image matrix represents an <u>estimate</u> of the object
  - I 2D square pixels
  - $\mu_i \quad 1 \le i \le I$
- The projections $\hat{\lambda}(\phi, x')$ that would occur if this <u>were</u> the real object are:

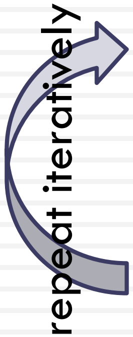
$$\hat{\lambda}(\phi, x') = \sum_{i=1}^{I} \alpha_i(\phi, x') \mu_i$$

where $\alpha_i(\phi, x')$ is the average path length traversed by the (ϕ, x') projections through the ith cell.



Iterative methods

- Let us consider the equation: $\hat{\lambda}(\phi, x') = \sum_{i=1}^{I} \alpha_i(\phi, x')\mu_i$
  - it symbolically represents a set of thousands equations
  - it cannot be solved by numerically inverting $\alpha_i(\phi, x')$
- Iterative methods consist in solving this equation by:



  - guess/adjust the values of the μ_i
  - calculate the computed projections $\hat{\lambda}(\phi, x') = \sum_{i=1}^{I} \alpha_i(\phi, x')\mu_i$
  - evaluate the difference between calculated and measured projections
  - stop when a "good agreement" is found
- These final values μ_i are then taken to be the solution, i.e. the image.

Iterative methods

- Because of measurement noise, and the approximations engendered by the model, there will not be a single exact solution.
- Part of the difficulty of implementing iterative methods is in deciding upon the correct criteria for testing the convergence of the intermediate steps and knowing when to stop (ill-posed problem).
- The many iterative algorithms may differ in the manner in which the corrections are calculated and reapplied during each iteration.
  - additively or multiplicatively
  - immediately after being calculated or stored and applied only at the end of each round of iteration
- The order in which the projection data are taken into consideration may differ as well.

Iterative methods: a simple example

- In the following, we give a simple example of a additive, immediate-correction iterative method:

Consider the matrix O (= the **O**bject)

$$O = \begin{matrix} 1 & 2 & 3 \\ 8 & 9 & 4 \\ 7 & 6 & 5 \end{matrix}$$

The Object matrix

Take the 4 projections P_1= E-W, P_2= N-S, P_3= NE-SW e P_4= NW-SE. They give

$$P_1 = \begin{matrix} 6 \\ 21 \\ 18 \end{matrix} \; ; \; P_2 = \begin{matrix} 16 & 17 & 12 \end{matrix} \; ; \; P_3 = \begin{matrix} 10 & & \\ & 19 & \\ & & 10 \end{matrix} \; ; \; P_4 = \begin{matrix} & & 6 \\ & 15 & \\ 14 & & \end{matrix}$$

Iterative methods: a simple example

As a first iteration, we assign to all the elements in a row the average of the result obtained from the projection P_1; we have the first iteration of the Image matrix (I = the Image).

$$I_1 = \begin{matrix} 2 & 2 & 2 \\ 7 & 7 & 7 \\ 6 & 6 & 6 \end{matrix}$$

$$I_1 = \langle P_1 \rangle$$

For I_1 we have

$$\text{m.s.e.}(I_1) = ([1 + 0 + 1 + 1 + 4 + 9 + 1 + 0 + 1]/9)^{1/2} = (18/9)^{1/2} = 1.4$$

The new I_1 matrix gives obviously for the E-W projection again the value P_1; instead for the N-S projection it gives a different value P_2' = 15 15 15; these values differ from those of P_2 by the amounts (-1) (-2) (+3).

Iterative methods: a simple example

We now iterate the I_1 matrix adding to the elements in each column one third of this difference, obtaining the iterated matrix I_2

$$I_2 = \begin{matrix} 2+1/3 & 2+2/3 & 1 \\ 7+1/3 & 7+2/3 & 6 \\ 6+1/3 & 6+2/3 & 5 \end{matrix}$$

$$I_2 = I_1 + \langle P_2 - P_2' \rangle$$

and for I_2

m.s.e.(I_2)= $([16/9 + 4/9 + 4 + 4/9 + 16/9 + 4 + 4/9 + 4/9 + 0]/9)^{1/2} = (120/81)^{1/2} = 1.2$

We see that already after the first iteration the error has decreased. We continue the process, computing the new oblique **NE-SW** projection P_3'', and we get$^{(*)}$

Iterative methods: a simple example

10

$P_3'' =$ 15

12+2/3

The nev **NE-SW** projection

The differences with the P_3 projection of the object are (0); (-4); (+2+2/3); correcting for these differences the matrix I_2, we get the third iteration I_3

	2+1/3	2+2/3	2+1/3
$I_3 =$	7+1/3	9	4+2/3
	7+2/3	5+1/3	5

$$I_3 = I_2 + \langle P_3 - P_3'' \rangle$$

The error in the I_3 matrix is

m.s.e.(I_3)= $([16/9 + 4/9 + 4/9 + 4/9 + 0 + 4/9 + 4/9 + 4/9 + 0]/9)^{1/2} = (36/81)^{1/2} = 0.67$

Iterative methods: a simple example

We use now the P_4 = **NW-SE** projection.

The I_3 matrix gives the values

$7_{+1/3}$

$P_4''' =$ $16_{+1/3}$

$12_{+2/3}$

The differences with the P_4 projection of the object are $(1_{+1/3})$; $(1_{+1/3})$ **-(1+1/3)** correcting for these differences the matrix I_3, we finally get the fourth iteration I_4

Iterative methods: a simple example

	1+8/9	2	2+1/3
I_4 =	8	8+5/9	4
	7+2/3	6	4+5/9

The fourth iteration

$$I_4 = I_3 + \langle P_4 - P_4''' \rangle$$

The error in the I_4 matrix is

m.s.e.(I_4)= ([64/81 + 0 + 4/9 +0+ 16/81 + 0 + 4/9 + 0 + 16/81]/9)$^{1/2}$ =(168/729)$^{1/2}$= =0.48

We see that the successive iterations improve the values obtained, as it is proven by the decrease of the mean square error.

Part III – Back projection: LSBP and FBP

- The Physics of medical imaging / edited by Steve Webb (1988)

Linear superposition of back projections: a simple numerical example

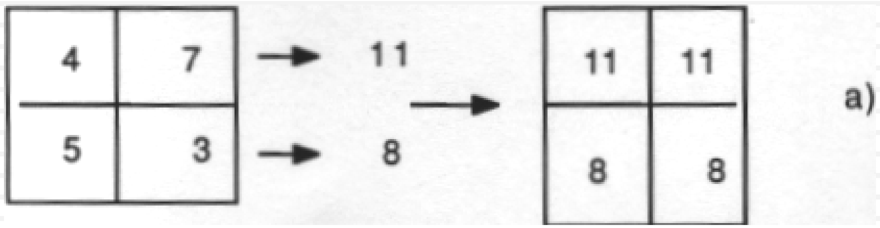
- In this simple numerical example, our section (object plane) consists of a square, represented by a 4-elements matrix. The matrix values represent the (unknown) absorption coefficients:

4	7
5	3

- We begin letting our beam cross the square from left to right (from "West" to "East"). As already said, one single measurement provides only the sum of the absorption coefficients. The projection "West to East" gives, for the upper and lower rows respectively, the values 11 and 8 (the sum of the values of the elements in each row).
- We shall call "profile" of our object the sum of the intensities obtained in one given direction. Therefore the profile measured from W to E consists in the two numbers 11 an 8.

Linear superposition of back projections: a simple numerical example – a)

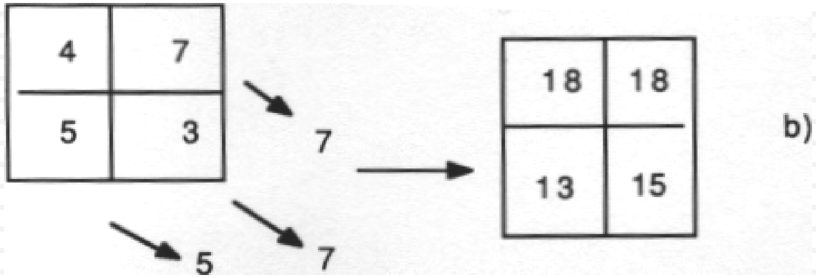
- These values obtained (namely the profile obtained with the first exposure) are initially assigned, row by row, to each element of each row, getting the values



- The values obtained as a sum of the elements by rows (the profile) have been "back projected" (in the E-W direction) on the various matrix elements.
- We indicate with “a)” the temporary result of this back-projection procedure

Linear superposition of back projections: a simple numerical example - b)

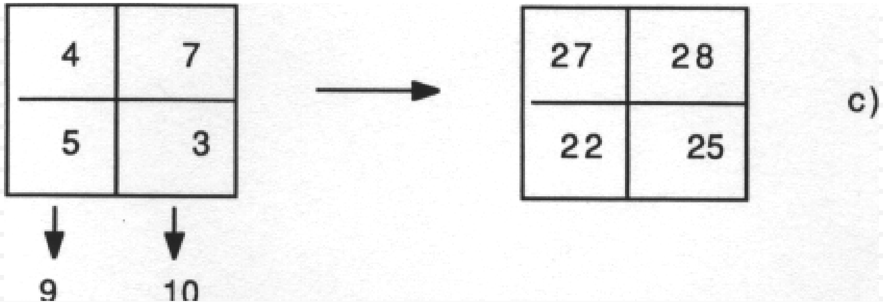
- Next the beam crosses the square in an oblique direction NW-SE. The projections from NW to SE give a new profile with the values 7, 7, 5
- These are back projected in the oblique SE-NW direction and added to the values previously obtained [matrix a) in previous slide]



- A <u>"linear superposition"</u> has been performed. We indicate with "b)" the temporary result of this procedure

Linear superposition of back projections: a simple numerical example – c)

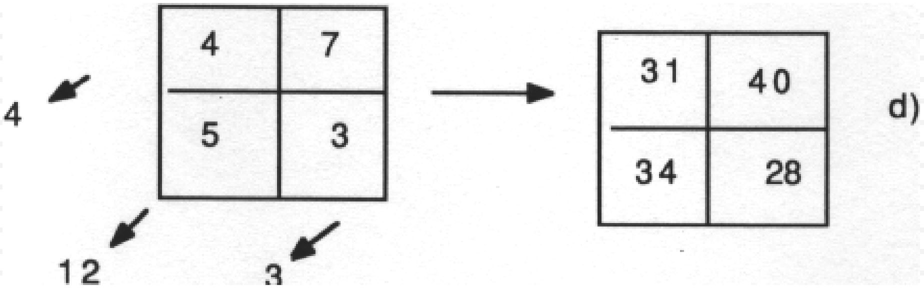
- We now perform the N-S projections. They give for the N-S profiles the values 9, 10
- Back projecting in the vertical S-N direction, and adding these values to matrix "b)" we get the values:



- We indicate with "c)" the temporary result of this back-projection procedure

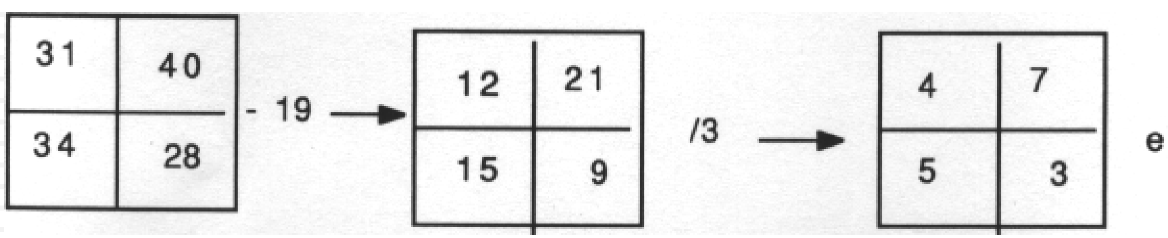
Linear superposition of back projections: a simple numerical example – d)

- The last projections are the NE-SW, giving for the profiles the values 4, 12, 3
- Back projecting in an oblique SW-NE direction and adding these values to those of matrix c), the following matrix "d)" is obtained:



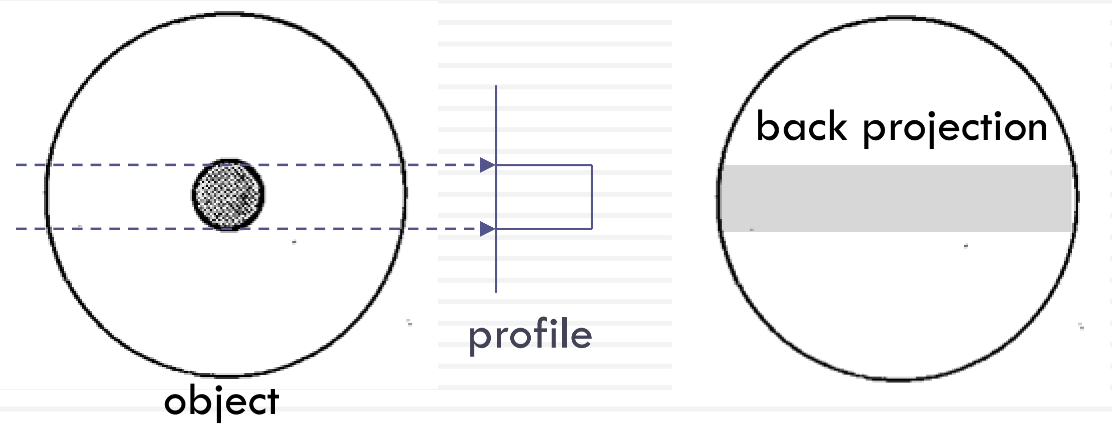
Linear superposition of back projections: a simple numerical example – e)

- Finally we subtract from each element in the matrix d) the number 19 (the total sum of all the unknown elements of the starting matrix, obtained as the sum of all the projections), which represents the "background“
- We further divide each matrix element by 3 (the number of the projections after the first one), obtaining eventually the numbers 4, 7, 3, 5, namely exactly the matrix to be found.



Linear Superposition of Back Projections

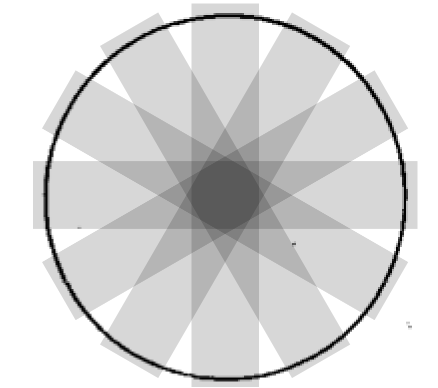
- We now consider a very simple object that will give the same profile (*) when seen from any view
- One single back projection of such profile is a stripe of uniform density:



- (*) sometimes we will use “profile” instead of “projection” when referring to a 1D projection

Linear Superposition of Back Projections

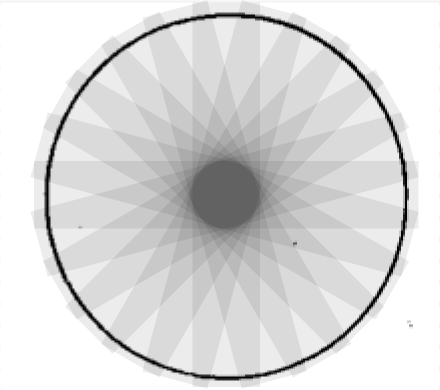
- We now consider the linear superposition of 6 such back projections (corresponding to 6 profiles taken 30° from one another)



- The resulting image shows a central build-up, corresponding to the object image, but surrounded by a figure roughly star-like with 12 spikes
- It is quite easy to realize that the back projections taken from n angular directions will produce, once added, a figure with 2n spikes

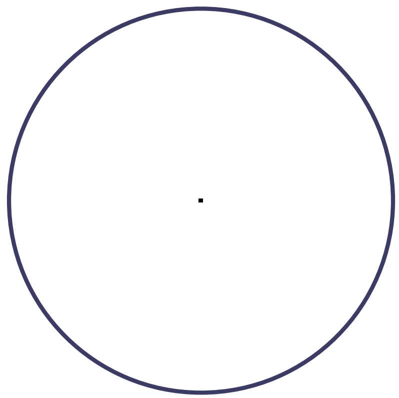
Need for a filtered back projection

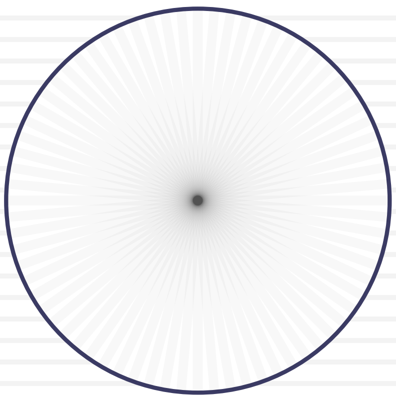
- Increasing the number of projections (12 in the figure below), the number of spikes increases but every single spike becomes less visible
- Overall, we still obtain a central figure with a high density, corresponding to the object image, surrounded by a background halo, whose density decreases with the distance from the center



Need for a filtered back projection

- This is even more apparent for a point-like object and in the limit of many angles; here a linear superposition of 36 back projections has been used





- Again, an halo of density decreasing with the distance from the center is obtained
- It can actually be demonstrated that the halo corresponding to the LSBP of a point-like object in the limit of many angles is given by an $1/r$ function

IRF of the LSBP

point-like object $\delta(x,y)$

impulse response function $1/r(x,y)$

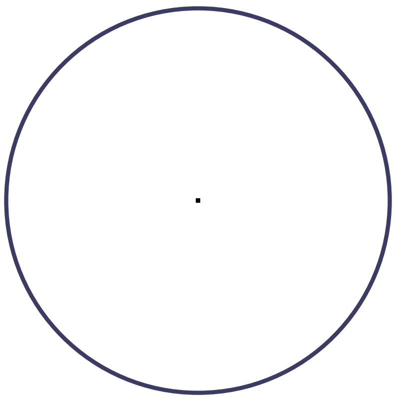
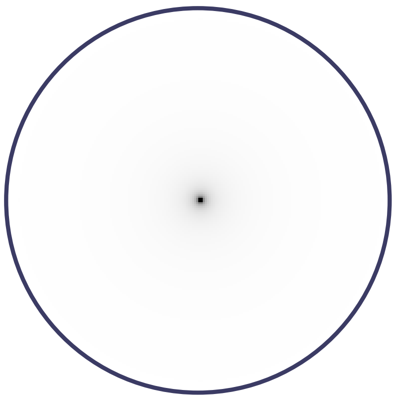
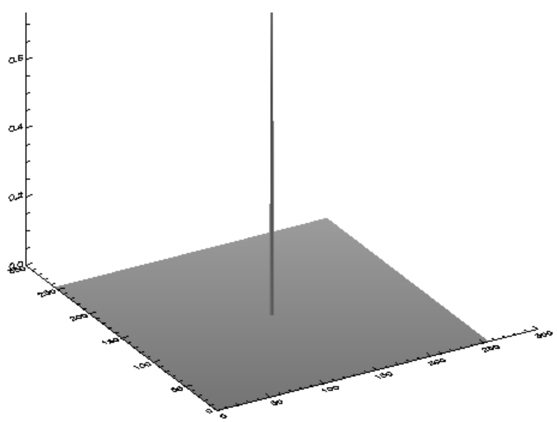


image view

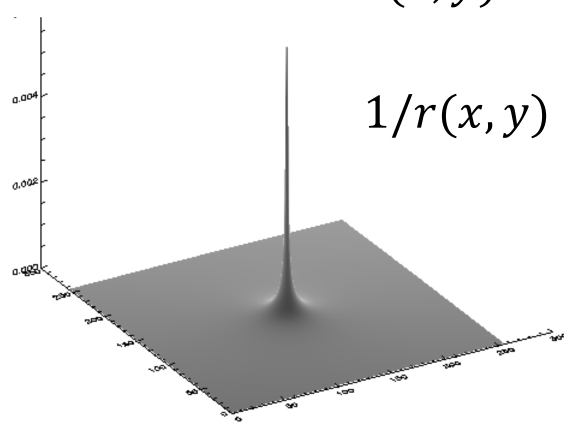


$$r(x,y) = \sqrt{x^2 + y^2}$$

$$1/r(x,y) = 1/\sqrt{x^2 + y^2}$$



2D function view



Need for a filtered back projection

- Resolving our object plane in many points, the resulting image will consist in the images of the corresponding points in the image plane, each of them surrounded by its halo
- Thus the LSBP method requires appropriate corrections: we will now derive a <u>filtered</u> back projection method that overcomes this limit
- The derivation of the <u>filtered</u> back projection method will use the tools we have introduced in the applied linear system theory, and in particular we will use 2D Fourier Transforms to move from the image plane (spatial coordinates) to the 2D spatial-frequency plane and vice-versa

2D Notation

- We will use the following notation:
  - *f(x,y)* is a 2D function representing the object
  - *g(x,y)* is a 2D function representing the image
  - *irf(x,y)* is a 2D function representing the impulse response function of the imaging process
- As we have seen, in LSI systems the image is a convolution of the object with the *irf*. In 2D this can be written as:

$$g(x,y) = f(x,y) ** irf(x,y)$$

where ** indicates a 2D convolution, or more explicitly as :

$$g(x,y) = \int\int_{-\infty}^{+\infty} d\alpha d\beta f(\alpha,\beta) irf(x-\alpha, y-\beta)$$

where α and β are Cartesian spatial coordinates (as are *x* and *y*)

Filtered back projection

- The image obtained from the LSBP *g(x,y)* is related to the "true image" *f(x,y)* of the object by the relation:

  LSBP Image = True Image ** (1/r)

  where the symbol ** means a convolution, i.e.

$$g(x,y)=\int\int_{-\infty}^{+\infty} d\alpha d\beta f(\alpha,\beta)/r[(x-\alpha),(y-\beta)]$$

$$=\int\int_{-\infty}^{+\infty} d\alpha d\beta f(\alpha,\beta)[(x-\alpha)^2+(y-\beta)^2]^{-1/2}$$

- Moving to the spatial-frequency domain, and indicating with *T(u,v)* the characteristic function, i.e. the FT of the *irf(x,y)* we have:

$$G(u,v)=F(u,v)\cdot T(u,v)$$

- It is important to point out that we would like to obtain *f(x,y)* [or equivalently *F(u,v)*] while the LSBP provides *g(x,y)* [or *G(u,v)*]

Filtered back projection

- It can be shown that in this case the FT of the *irf* $1/r$ is given by $1/\rho$, where ρ is the "distance" in the (u,v) space, namely:
  - $\rho\,(u,v) = (u^2 + v^2)^{1/2}$
  - $1/\rho\,(u,v) = (u^2 + v^2)^{-1/2}$
- Thus $G(u,v) = F(u,v)/\rho(u,v)$
- This means that the *F(u,v)* and *f(x,y)* can - in principle - be obtained simply as $F(u,v) = G(u,v)\cdot\rho(u,v)$

$$f(x,y) = \int\int_{-\infty}^{+\infty} dudv e^{i2\pi(ux+vy)} F(u,v)$$

$$= \int\int_{-\infty}^{+\infty} dudv e^{i2\pi(ux+vy)} \rho(u,v) G(u,v)$$

- This procedure is called filtered back projection since the back projection *G(u,v)* is "filtered" by the ρ function

Filtered back projection (in principle)

- In principle, the "true" image $f(x,y)$ can be obtained as follows
  - perform the LSBP obtaining $g(x,y)$
  - FT $g(x,y)$ to yield $G(u,v)$
  - filtering in the spatial frequency domain with the ρ function to obtain $\rho G(u,v)$
  - perform the FT^{-1} of $\rho G(u,v)$

The ramp filter

- As it can be seen, the filter ρ has the role of giving more weight to the higher frequencies, to compensate the defocusing process of LSBP
- However, due to the finite sampling of $f(x,y)$, the filtering function ρ must be truncated at a maximum frequency ρ_{max}.
- Therefore, a "ramp" filter will be used:

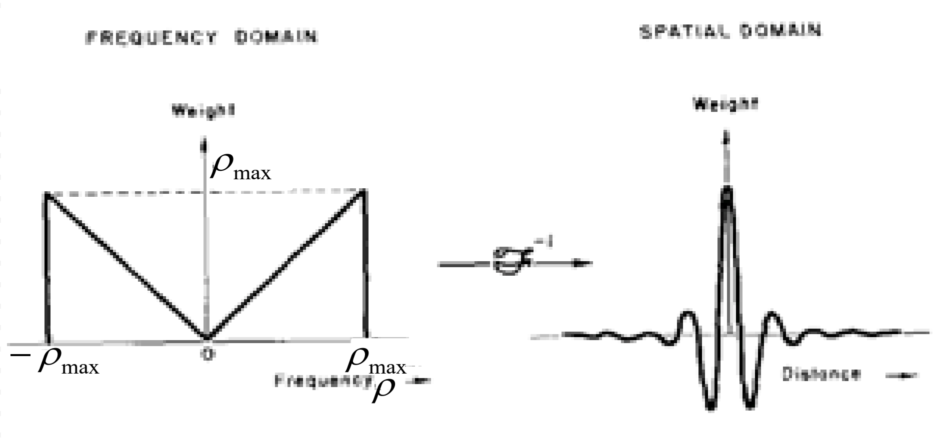
$$ramp(\rho) = \begin{cases} |\rho| & |\rho| \leq \rho_{\max} \\ 0 & |\rho| > \rho_{\max} \end{cases}$$

Filtered back projection (in practice)

- As a matter of fact, the filtering is done on the space domain rather than in the Fourier domain
- the relation: $F(u,v) = G(u,v) \cdot ramp(\rho)$

  is equivalent to: $f(x,y) = g(x,y) ** R(r)$

  where *R(r)* is the FT^{-1} of the ramp filter, and is known as the Ram-Lak filter (Ramachandran – Lakshminarayanan)
- It can be shown that $R(r) = \rho_{max}^2[2\,\mathrm{sinc}(2r\rho_{max}) - \mathrm{sinc}^2(r\rho_{max})]$



Filtered back projection (in practice)

- Therefore, in practice, the "true" image $f(x,y)$ could be obtained as follows:
  - perform the LSBP obtaining $g(x,y)$
  - evaluate $R(r)$
  - perform a 2D convolution $f(x,y) = g(x,y) ** R(r)$
- But this is not the end of the story, either

Filtered back projection (in practice)

- Actually, as a matter of fact, the filter is applied on the single profiles <u>before</u> performing the back projections
- Moreover, different filters can be used:
  - The Ram-Lak filter derives from a simple truncation of the <u>ideal</u> filtering function ρ at the frequency ρ_{max}, to take the sampling into account
  - However, <u>in practice</u> the high frequencies components must be suppressed also because they carry the noise contribution
  - Different filter may be chosen, depending on the quality of the data, with the aim to optimize the image by modulating the contributions of the high frequencies

Filter comparison (in frequency space)

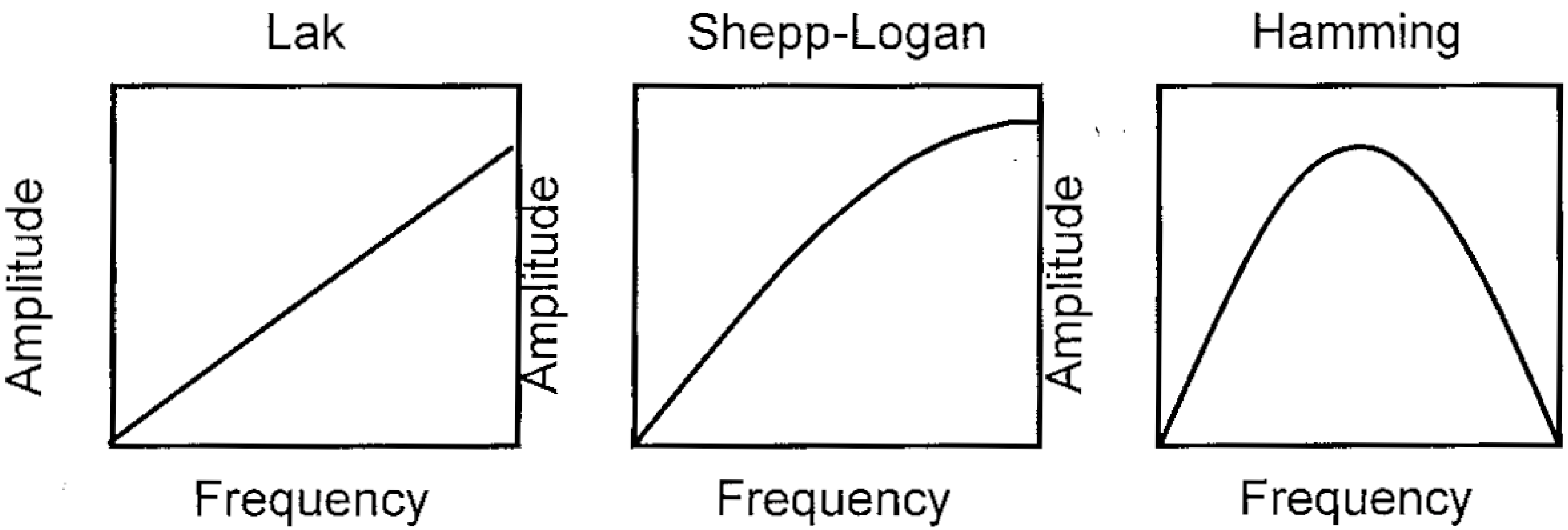


FIGURE 13-29. Three computed tomography (CT) reconstruction filters are illustrated. The plot of the reconstruction filter is shown below, with the image it produced above. **(A)** The Lak is the ideal reconstruction filter in the absence of noise. However, with the quantum noise characteristic in x-ray imaging including CT, the high-frequency components of the noise are amplified and the reconstructed image is consequently very noisy. **(B)** The Shepp-Logan filter is similar to the Lak filter, but roll-off occurs at the higher frequencies. This filter produces an image that is a good compromise between noise and resolution. **(C)** The Hamming filter has extreme high frequency roll-off.

Filtered back projection (in practice)

- Therefore, in practice, the "true" image $f(x,y)$ can be obtained as follows:
  - choose a suitable filter $\varphi(x)$ (which includes optimizing the filter parameters)
  - apply the filter on the single profiles (performing a 1D convolution between the filter and the profiles in the real space)
  - perform a backprojection of the filtered profiles, i.e. a Filtered Back Projection (FBP)

Part I – More on FBP

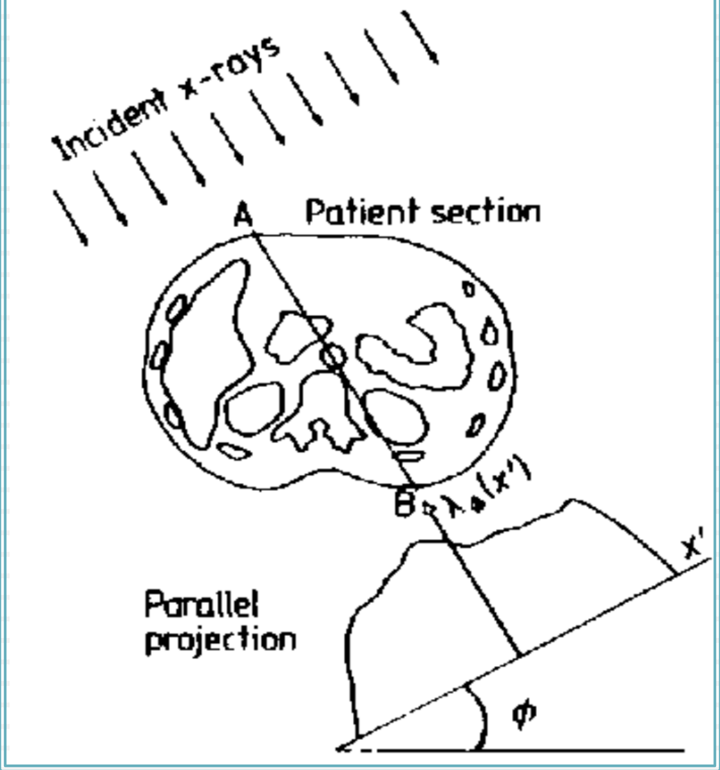
- The Physics of medical imaging / edited by Steve Webb (1988)
- Lecture notes of prof. L. Bertocchi
- Lecture slides of prof. R. Longo
- The essential physics of medical imaging / Jerrold T. Bushberg ... [et al.] (2012)

Profiles as line integrals

- Monochromatic and collimated beam
- No scattered radiation is recorded by the detector
- *f[x,y]* distribution of the linear attenuation coefficient within the imaged slice
- $\lambda_\phi(x')$ object profile at angle ϕ

$$I_\phi(x') = {}^0I_\phi(x')\, e^{-\int_A^B f[x,y]\,dy'}$$

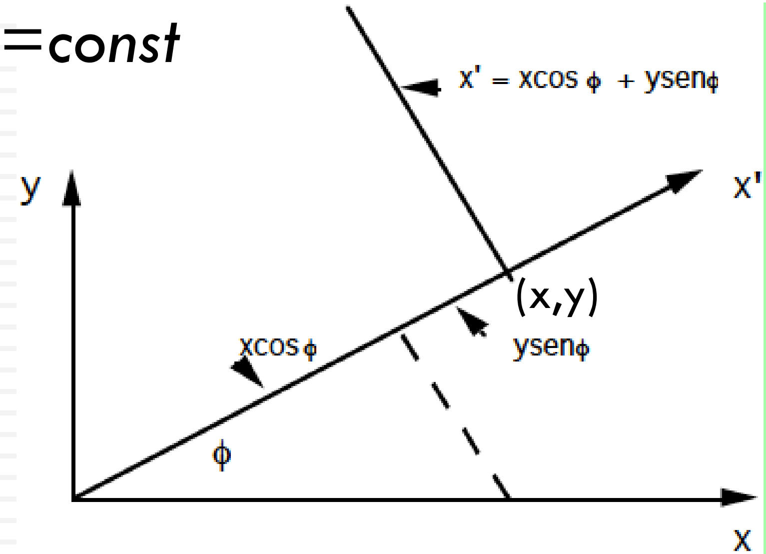
$$\lambda_\phi(x') = -\ln\left[\frac{I_\phi(x')}{{}^0I_\phi(x')}\right] = \int_A^B f[x,y]\,dy'$$



Profiles as line integrals

- $x' = x\cos\phi + y sen\phi$ is the straight line $x' = const$
- Thus the line integral

$$\lambda_\phi(x') = -\ln\left[\frac{I_\phi(x')}{{}^{0}I_\phi(x')}\right] = \int_A^B f[x,y]dy'$$



- can be written as

$$\lambda_\phi(x') = \iint dxdyf[x,y]\delta(x' - x\cos\phi - y sen\phi)$$

(Radon Transform)

- This equation expresses a <u>linear relation</u> between the object function *f[x,y]* and the measured profiles $\lambda_\phi(x')$.
- The goal of the reconstruction is indeed to invert this equation, i.e. to express the function *f[x,y]* from a set of profiles $\lambda_\phi(x')$.

Central Section Theorem

Technicalities

$$\lambda_\phi(x') = \iint dxdy f[x,y]\delta(x' - x\cos\phi - y sen\phi)$$

$$\lambda_0(x') = \iint dxdy f[x,y]\delta(x' - x)$$

$$\lambda_0(x') = \int dy f[x', y]$$

- ϕ can be considered as a free parameter
- For the moment we set $\phi=0$ so that calculations are simpler

$$\Lambda_0(u) = \int dx e^{-i2\pi ux} \lambda_0(x)$$

$$\Lambda_0(u) = \iint dxdy e^{-i2\pi(ux+vy)} f[x,y] \Big|_{v=0}$$

$$\Lambda_0(u) = F[u,0]$$

Central Section Theorem

Technicalities

- Polar coordinates (ρ,ϕ) can be used in the Fourier space instead of the usual Cartesian coordinates [u,v]
- Generally speaking, the equations that relate these coordinates systems are

$$u = \rho\cos(\phi) \qquad \rho = (u^2 + v^2)^{1/2}$$

$$v = \rho\sin(\phi) \qquad \phi = \tan^{-1}(v/u)$$

- However, with $\phi=0$, they reduce to $u = \rho$ and $v=0$
- Thus, we have: $$\Lambda_0(u) = F[u,0] = F^p(\rho,0)$$

- And, more generally, the Central Section Theorem

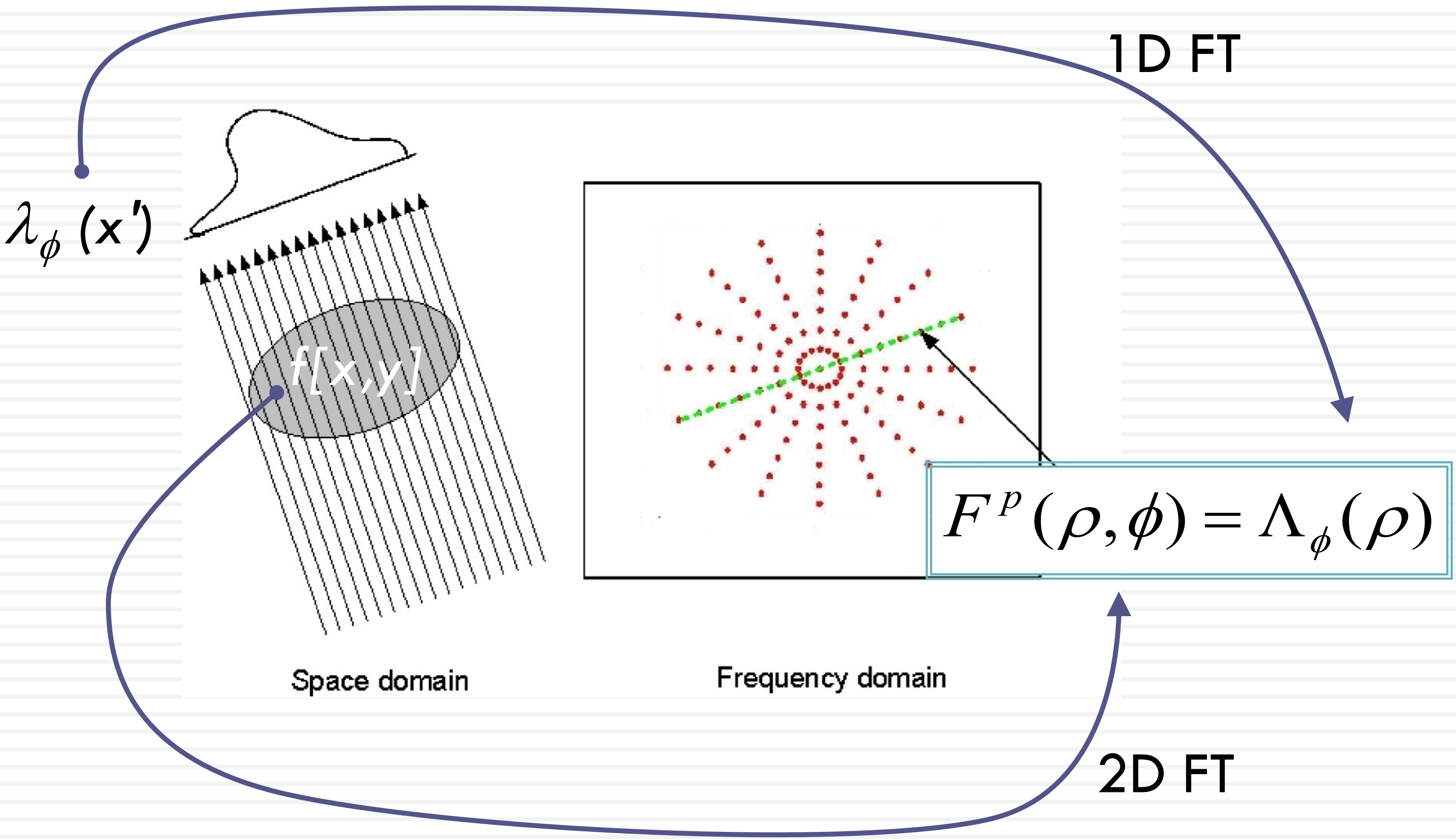
$$\Lambda_\phi(\rho) = F^p(\rho,\phi)$$

Central Section Theorem

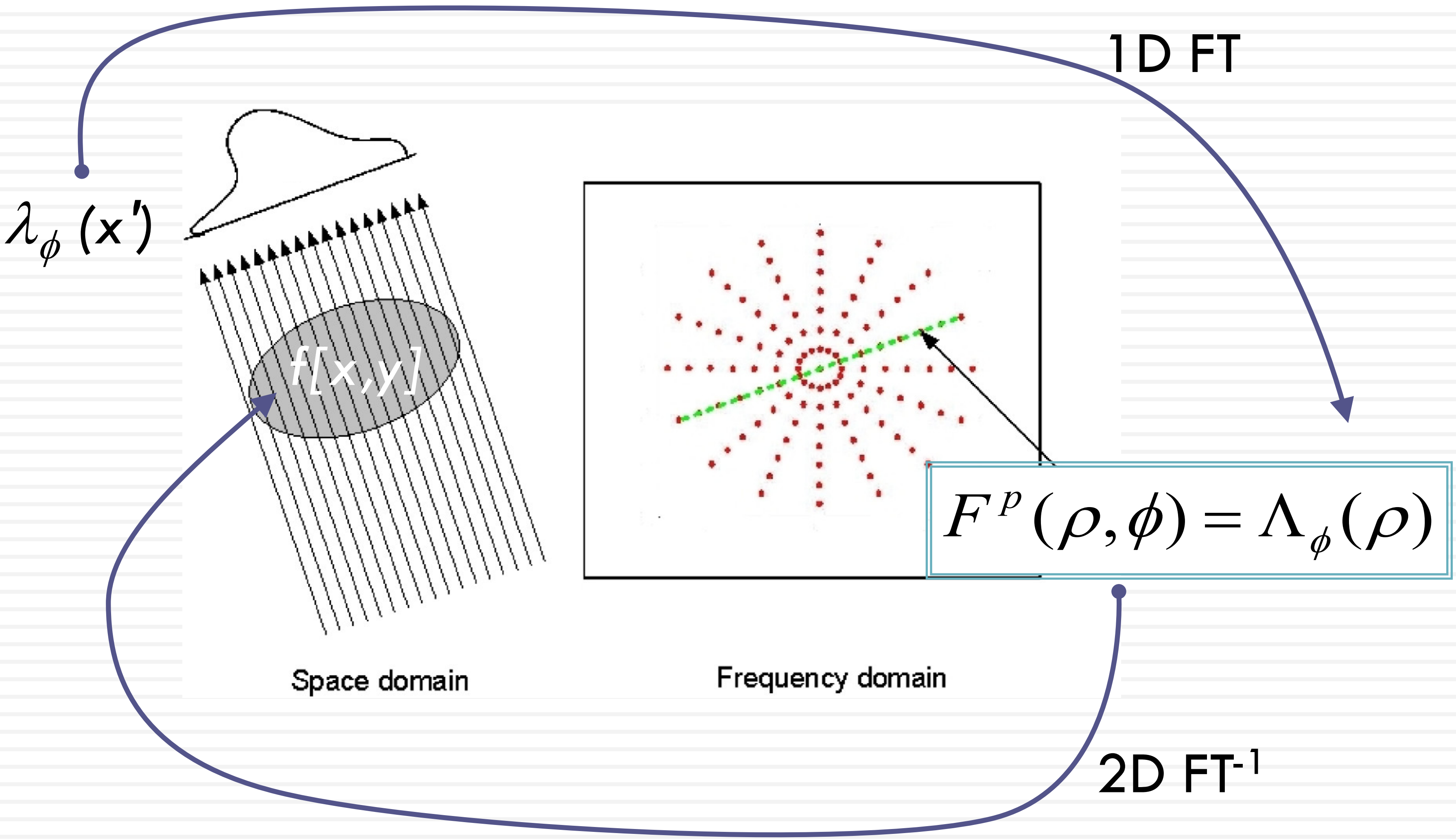
$$\Lambda_{\phi}(\rho) = F^{p}(\rho, \phi)$$

- This theorem is also known as:
  - projection-slice theorem
  - central slice theorem
  - Fourier slice theorem
- It states that the results of the following two calculations are equal:
  - Take a two-dimensional function *f[x,y]*, project it onto a (one-dimensional) line (Radon transform), and do a 1D Fourier transform of that projection.
  - Take that same function, do a 2D Fourier transform first, and then cut off a slice through its origin, which is parallel to the projection line.

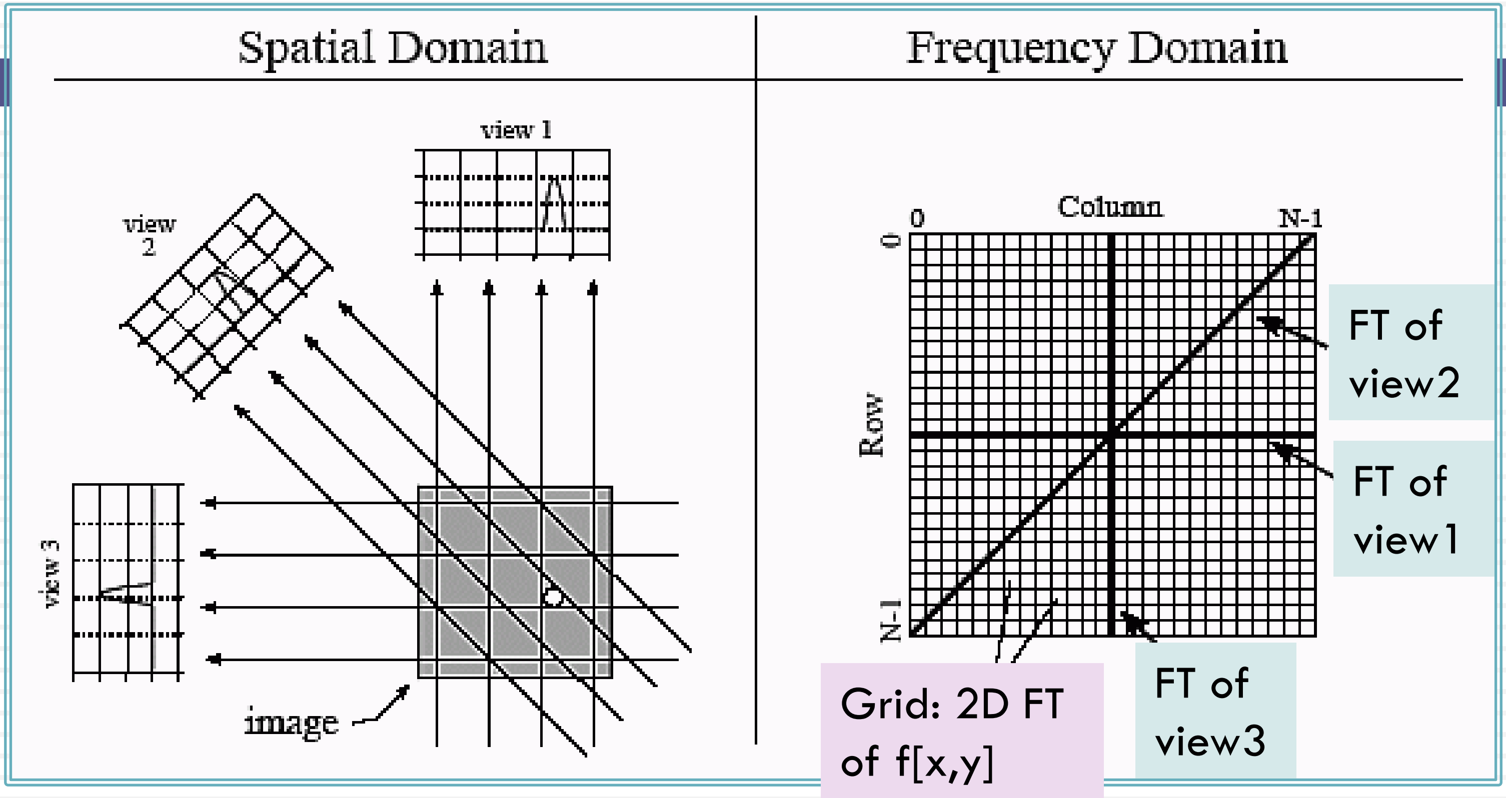
Central Section Theorem



Central Section Theorem



Central Section Theorem



- Thus, each profile collected in the spatial domain contributes to a section of the 2D FT of *f[x,y]* in the frequency domain
- By collecting a sufficient number of profiles, we can obtain sufficient knowledge on the 2D FT of *f[x,y]*
- Eventually a 2D FT^{-1} can allow us to obtain the *f[x,y]* itself

Obtaining *f[x,y]* in practice

- FT^{-1}

$$f[x,y] = \iint du\,dv\,e^{i2\pi(xu+yv)} F[u,v]$$

- FT^{-1} in polar coord.

$$f[x,y] = \int_0^{2\pi} d\phi \int_0^{\infty} d\rho\,\rho F^p(\rho,\phi) e^{i2\pi\rho(x\cos\phi + y sen\phi)}$$

- change of the integration limits

$$f[x,y] = \int_0^{\pi} d\phi \int_{-\infty}^{\infty} d\rho\,|\rho| F^p(\rho,\phi) e^{i2\pi\rho(x\cos\phi + y sen\phi)}$$

- Applying the Central Slice Theorem $F^P(\rho,\phi) = \Lambda_\phi(\rho)$

$$f[x,y] = \int_0^{\pi} d\phi \int_{-\infty}^{\infty} d\rho\,|\rho| \Lambda_\phi(\rho) e^{i2\pi\rho(x\cos\phi + y sen\phi)}$$

Obtaining *f[x,y]* in practice

- The equation

$$f[x,y] = \int_0^\pi d\phi \int_{-\infty}^{\infty} d\rho \, |\rho| \, \Lambda_\phi(\rho) e^{i2\pi\rho(x\cos\phi + ysen\phi)}$$

$$= \int_0^\pi d\phi \int_{-\infty}^{\infty} d\rho \, |\rho| \, \Lambda_\phi(\rho) \, e^{i2\pi\rho x'} \Big|_{x' = x\cos\phi + ysen\phi}$$

can be split in two parts:

$$f[x,y] = \int_0^\pi d\phi \, {}^{\dagger}\lambda_\phi(x') \qquad \textit{with} \qquad {}^{\dagger}\lambda_\phi(x') = \int_{-\infty}^{\infty} d\rho \, |\rho| \, \Lambda_\phi(\rho) e^{i2\pi\rho x'} \Big|_{x' = x\cos\phi + ysen\phi}$$

Linear Superposition of Filtered back projections

Obtaining *f[x,y]* in practice

$$^{\dagger}\lambda_\phi(x') = \int_{-\infty}^{\infty} d\rho\, |\rho| \Lambda_\phi(\rho) e^{i2\pi\rho x'}\Big|_{x'=x\cos\phi+y sen\phi} = \int_{-\infty}^{\infty} d\rho\, {}^{\dagger}\Lambda_\phi(\rho) e^{i2\pi\rho x'}$$

$$\lambda_\phi(x') = \int_{-\infty}^{\infty} du\, \Lambda_\phi(u) e^{i2\pi u x'} = \int_{-\infty}^{\infty} d\rho\, \Lambda_\phi(\rho) e^{i2\pi\rho x'}$$

$$^{\dagger}\Lambda_\phi(\rho) = |\rho| \Lambda_\phi(\rho)$$

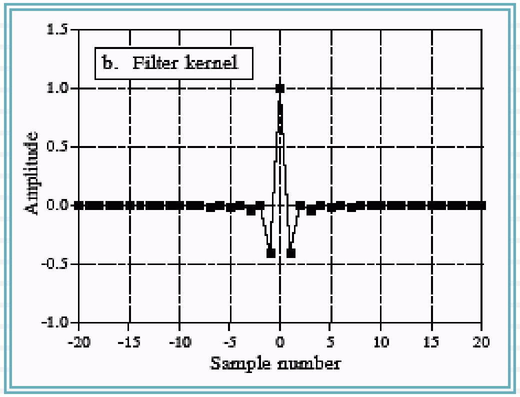
$$^{\dagger}\lambda_\phi(x') = \lambda_\phi(x') * p(x')$$

$$p(x') = FT^{-1}(|\rho|)$$

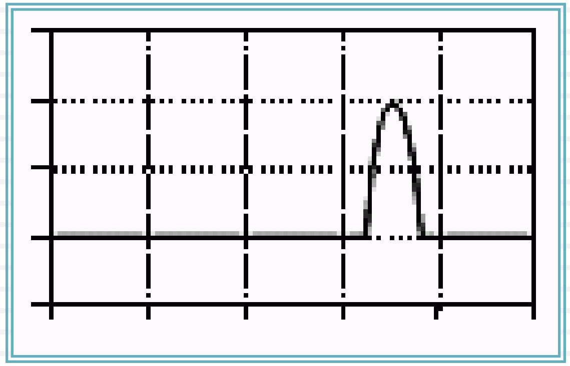
- In practice:
  - first the profile is convolved with the filter
  - then *f[x,y]* is obtained by back projecting the filtered profiles:

$$f[x,y] = \int_0^{\pi} d\phi\, {}^{\dagger}\lambda_\phi(x')\Big|_{x'=x\cos\phi+y sen\phi}$$

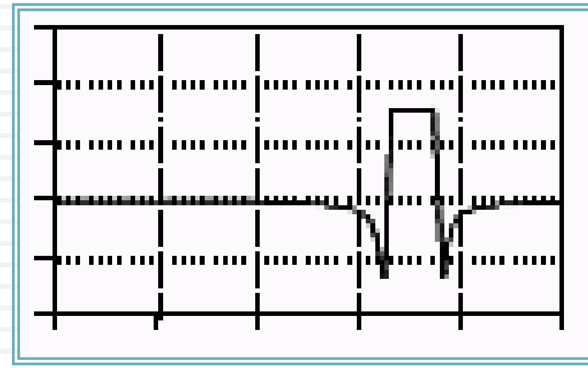
LS of filtered BP (LSFBP => simply FBP)



*

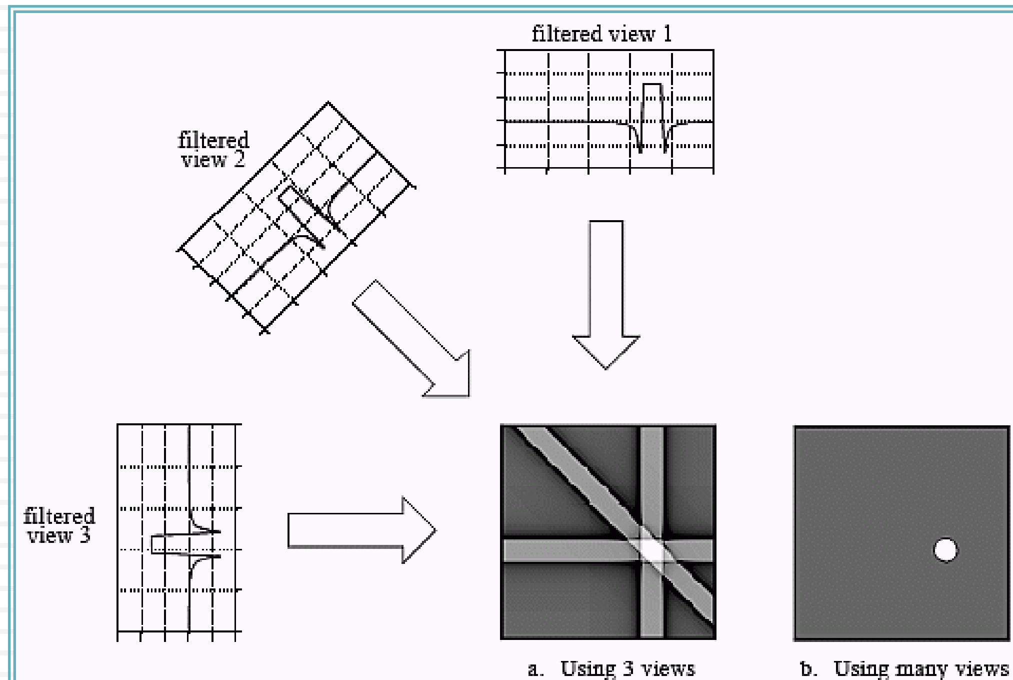


=



$$^{\dagger}\lambda_\phi(x') = \lambda_\phi(x') * p(x')$$

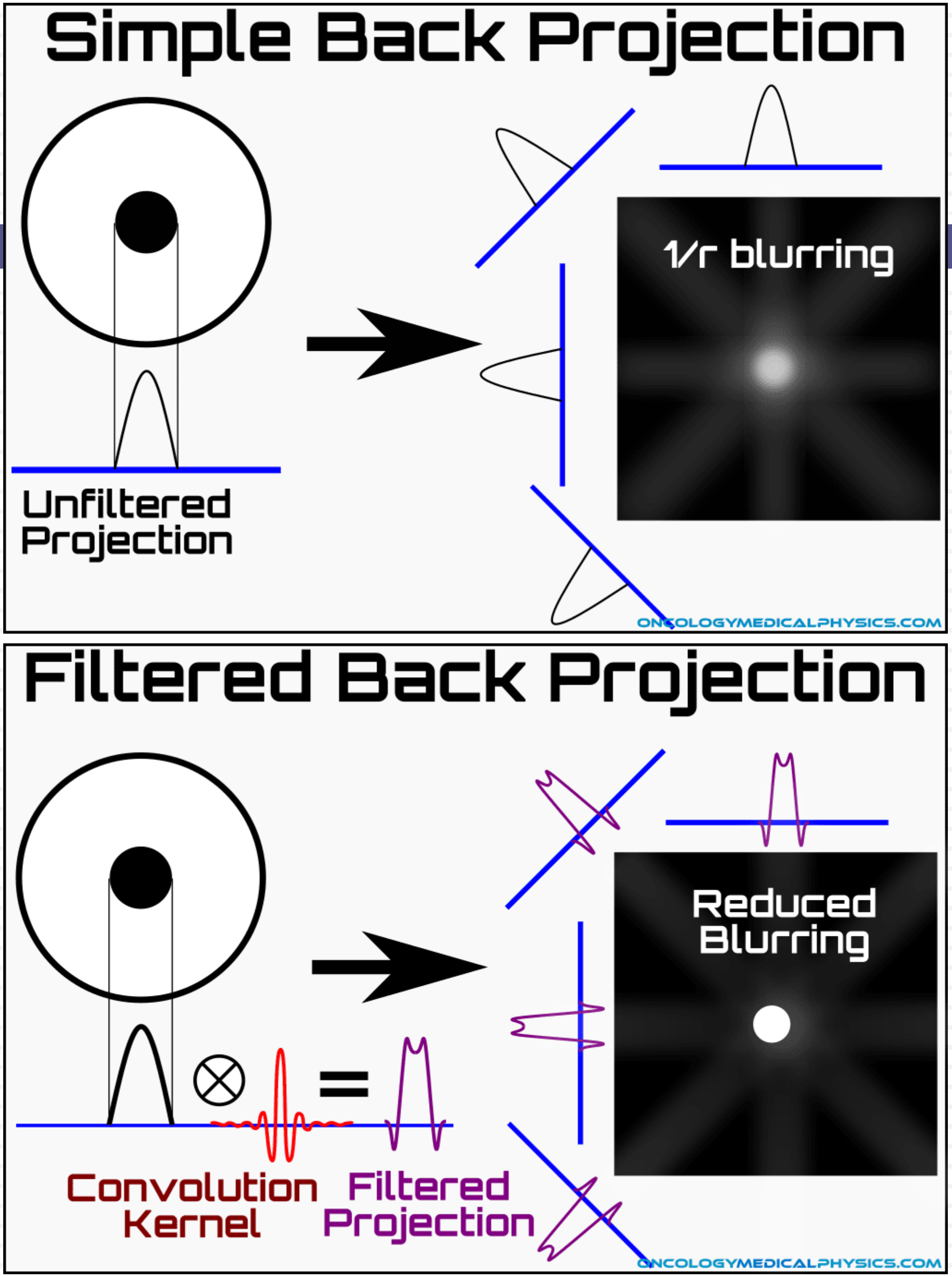
$$f[x,y] = \int_0^{\pi} d\phi \, ^{\dagger}\lambda_\phi(x') \Big|_{x'=x\cos\phi + y sen\phi}$$



LSBP

Vs

FBP



oncologymedicalphysics.com

Appendix: calculating the Ram-Lak filter

- The Physics of medical imaging / edited by Steve Webb (1988)

Calculating the Ram-Lak Filter (I)

$$R(x') = FT^{-1}(ramp(\rho))$$

$$= \int_0^{\rho_{\max}} \rho \exp(2\pi i \rho x')\, d\rho - \int_{-\rho_{\max}}^0 \rho \exp(2\pi i \rho x')\, d\rho$$

$$= \left[\frac{\rho \exp(2\pi i \rho x')}{2\pi i x'}\right]_0^{\rho_{\max}} - \int_0^{\rho_{\max}} \frac{\exp(2\pi i \rho x')\, d\rho'}{2\pi i x'}$$

$$- \left[\frac{\rho \exp(2\pi i \rho x')}{2\pi i x'}\right]_{-\rho_{\max}}^0 + \int_{-\rho_{\max}}^0 \frac{\exp(2\pi i \rho x')\, d\rho'}{2\pi i x'}$$

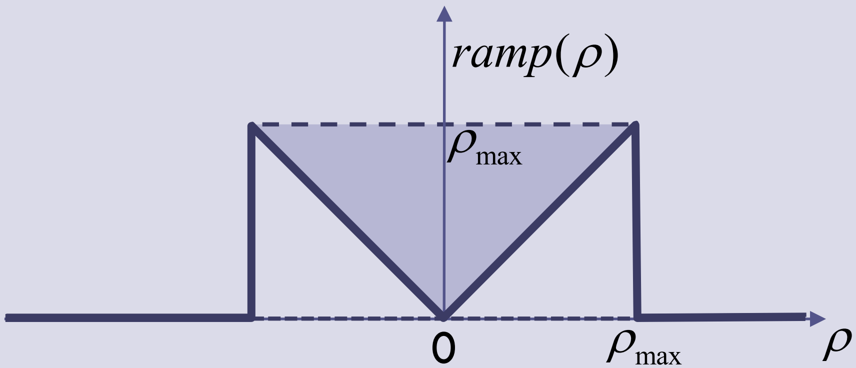
$$= \frac{\rho_{\max} \sin(2\pi \rho_{\max} x')}{\pi x'} - \left[\frac{\exp(2\pi i \rho x')}{(2\pi i x')^2}\right]_0^{\rho_{\max}} + \left[\frac{\exp(2\pi i \rho x')}{(2\pi i x')^2}\right]_{-\rho_{\max}}^0$$

$$= 2\rho_{\max}^2 \operatorname{sinc}(2\rho_{\max} x') + \frac{\cos(2\pi \rho_{\max} x') - 1}{2\pi^2 x'^2}$$

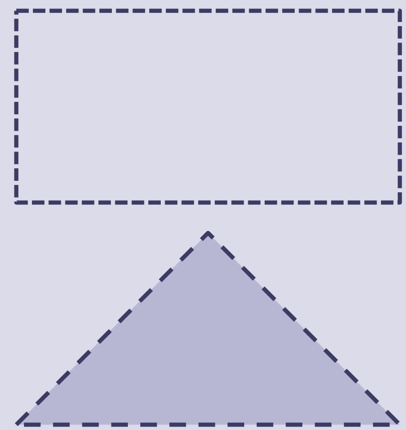
$$= 2\rho_{\max}^2 \operatorname{sinc}(2\rho_{\max} x') - \rho_{\max}^2 \operatorname{sinc}^2(\rho_{\max} x').$$

Calculating the Ram-Lak Filter (II)

$$R(x') = FT^{-1}(ramp(\rho)) = FT(ramp(\rho))$$



$$ramp(\rho) = \rho_{\max}\Pi(\frac{\rho}{2\rho_{\max}}) - \rho_{\max}\Lambda(\frac{\rho}{\rho_{\max}})$$



$$\rho_{\max}\Pi(\frac{\rho}{2\rho_{\max}})$$

$$\rho_{\max}\Lambda(\frac{\rho}{\rho_{\max}})$$

$$R(x') = FT(ramp(\rho)) = \rho_{\max} FT\left[\Pi(\frac{\rho}{2\rho_{\max}}) - \Lambda(\frac{\rho}{\rho_{\max}})\right]$$

$$= \rho_{\max}\left[2\rho_{\max}\,\mathrm{sinc}(2\rho_{\max}x') - \rho_{\max}\,\mathrm{sinc}^2(\rho_{\max}x')\right]$$

$$= 2\rho_{\max}^2\,\mathrm{sinc}(2\rho_{\max}x') - \rho_{\max}^2\,\mathrm{sinc}^2(\rho_{\max}x')$$