Part II

The Inverse Radon Transform and PET

Chapter 6

Introduction to Computerized Tomography

One of the major inventions in this century is the CT-scanner (Computerized Tomography). It can be mentioned that Cormack and Hounsfield got the Nobel-prize in Medicine 1979 for their work with computed axial tomography. The CT-scanner can be used for reconstruction of the X-ray absorption in the interior of structures, such as patients or machine parts with possible internal fractures.

Over the years several types of scanners have emerged such as the MR-scanner (MR - Magnetic Resonance) for measuring features related to the contents of water. This type of scanner is common in larger hospitals and it only uses magnetic fields, no X-rays or radioactive tracers are involved. Two other emerging techniques, both used to monitor the concentration of radioactive tracers in the interior of an object, are SPECT (Single Photon Emission Computed Tomography) and PET (Positron Emission Tomography). If the radioactive tracer is attached to, e.g., glucose [57], then the interesting possibility of measuring the activity of the brain arises. This possibility of dynamic measurements of the activity of the brain is now also appearing with one of the newest technologies fMRI (functional MRI).

What is common for the development of all types of scanners is that they, to some extent, have been based on inversion of Radon transform, but it should be mentioned that the MRI scanners mostly use inverse Fourier transformation. The most direct use of inverse Radon transform is found with the CT-scanner, and Section 6.1 describes the fundamental theory of physics along with the motivation of modelling the problem by use of the Radon transform. Next, Section 6.2 goes into the fundamentals of PET-imaging and some of the problems in practical scanning. An introduction to the various scanners mentioned above is given in [58].

Besides Section 7.6, which goes through the major direct inversion methods for the (p, τ) Radon transform (slant stacking, cf. Chapter 1), this part of the thesis will go into reconstruction from line integrals using normal parameters. First, in Chapter 7 some of the well-known direct 2D Radon inversion formulas are derived [14]. A large software package that includes all of the inversion methods mentioned in Chapters 7-8 has been developed [59, 60, 61]. Together with the package for generating images and their Radon transform [22] described in Appendix C.1, efficient and easy tools have been made for testing new algorithms and doing 2D reconstruction with the built-in algorithms.

During the very first years of work with the CT-scanner Hounsfield and his colleagues used an iterative reconstruction scheme based on linear algebra for reconstructing images. This strategy had a number of problems, and was abandoned when it was realized that the reconstruction problem could be modelled by the Radon transform. From the moment this was realized, most reconstruction algorithms in practical use have been based on direct reconstruction methods, especially the so called Filtered Backprojection algorithm. But in recent years, iterative reconstruction schemes have again become a very popular area of research. In Chapter 9 the basis of the iterative reconstruction algorithms are provided, and some of the more widely used methods are reviewed. A very fast implementation of these algorithms in 2D has been made [60, 5, 6]. It turns out that the methods used for iterative reconstruction of the 2D images directly can also be applied for reconstruction of 3D volumes, hence the methods both cover 2D and 3D reconstruction algorithms.

One of the newer and emerging techniques is 3D direct reconstruction from plane integrals [62] or line integrals relevant for 3D CT-scanners and 3D PET medical scanners respectively [63, 64, 65, 66, 67, 68, 69, 70, 71]. Some of the direct reconstruction methods, problems, and possibilities are shown in Chapter 10. Aspects of implementation including the use of multi processor hardware are also considered. A software packages has been developed for generating volumes and the corresponding Radon transform along with a package for reconstruction of 3D volumes using direct and iterative methods.

6.1 Fundamental Theory of the CT-Scanner

A CT-scanner can consist of a ring with one X-ray emitter and a large number of detectors positioned opposite to the emitter as shown in Fig. 6.1. Rotating the emitter and the detector array around the patient makes it possible to cover all parts of the brain. Note that the brain could be substituted by any structure with an interesting non-homogeneous interior hidden behind the surface.

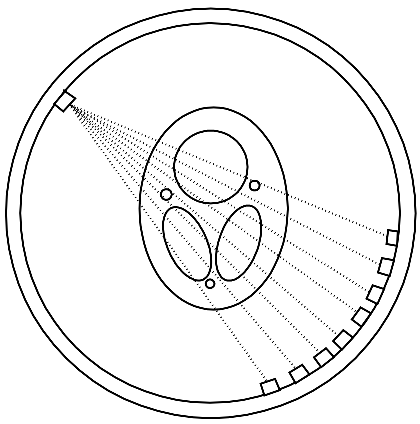

Figure 6.1 The CT Scanner.

It can be assumed that the X-rays travel in straight lines, and the beam is attenuated along that line with the attenuation coefficient μ. After the scan the attenuation coefficient is to be found by reconstruction in each point in the plane. It will be shown that this can be based on inversion of the Radon transform.

Assume a planar (2D) CT-scanner with only one slice of the brain measured, then the coordinate system can be chosen so that the slice is the $z = 0$ plane. Each line going through the scanned object is parameterized with two parameters: The normal distance from $(0, 0)$ to the line ρ, and the angle relative to the first axis θ. These two parameters are shown in Fig. 6.2.

It must be assumed, that the scanned object is non-homogeneous, hence the attenuation coefficient is a function of x and y, i.e., $\mu(x,y)$. Assume that the emitter and one of the detectors define the line (ρ,θ). In Fig. 6.3 three different objects are shown (seen along the line).

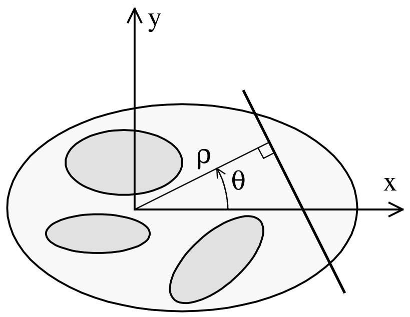

Figure 6.2 Parameters used to describe the lines.

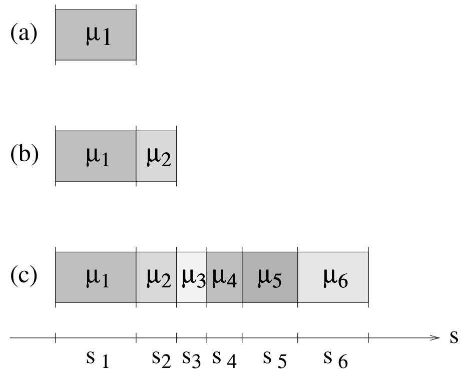

Figure 6.3 Three different media, with attenuation coefficients μ_i and widths s_i.

In case (a) the received intensity will follow the ordinary exponential decay

$$I(\rho,\theta) = I_0\ e^{-\mu_1 s_1} \tag{6.1}$$

If two different homogeneous media are penetrated as shown at (b) the received intensity is

$$I(\rho,\theta) = I_0\ e^{-\mu_1 s_1 - \mu_2 s_2} \tag{6.2}$$

If several homogeneous media are present, like in case (c), the result is simply that additional exponential factors are included for each layer

$$I(\rho,\theta) = I_0\ e^{-\sum_i \mu_i s_i} \tag{6.3}$$

In the general form, where the attenuation coefficient is a function of the line path, the sum in Eq. 6.3 will become an integral and s_i becomes the line element ds.

$$I(\rho,\theta) = I_0\ e^{-\int \mu(x,y)\ ds} \tag{6.4}$$

In this equation the parameter I_0 is the intensity of the emitter, and s denotes the parameter in the normal form of the line, where (x,y) lies on the line defined by (ρ,θ). Note that the exponential factor can be perceived as the probability of a single photon getting through the absorbing medium.

From Eq. 6.4 the projection $\mathcal{P}(\rho,\theta)$ is defined

$$\mathcal{P}(\rho,\theta) = \log\left(\frac{I_0}{I(\rho,\theta)}\right) = \int \mu(x,y)\ ds = \int_{-\infty}^{\infty} \mu(\rho\cos\theta - s\sin\theta, \rho\sin\theta + s\cos\theta)\ ds \tag{6.5}$$

It can be recognized that $\mathcal{P}(\rho,\theta)$ is the Radon transform of $\mu(x,y)$, cf. Eq. 2.8. Using the Dirac delta function the projections can be rewritten, cf. Eq. 2.8

$$\mathcal{P}(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy \tag{6.6}$$

The result is, that the attenuation coefficient $\mu(x, y)$ can be found from the projections using the inverse of the Radon transform.

In the literature [14, 72, 73] direct inversion schemes can be found. One class is based on the Fourier Slice Theorem, which is implemented using 1D Fast Fourier Transformation, 2D Fast Fourier transformation and 2D interpolation. This will normally result in a very fast algorithm, but numerical artifacts due to the interpolation can decrease the performance. Another famous inversion scheme is the Filtered Backprojection which is a combination of filtering and integration. This inversion scheme is by far the most common and used in medical scanners due to good numerical stability, but the algorithm will normally be somewhat slower. These aspects are covered in the following chapters.

In conclusion, it has been shown that the CT-scanner gives a map of the attenuation coefficient. The CT-scanners have been used several years, and one of the newer trends is to perform helical CT, where the emitter and detector ring rotate around the patient and make a uniform motion along the z-axis, as shown in Fig. 6.4. In this way a 3D-image, i.e., a volume of the interior can be obtained with a better resolution on the z-axis, compared to a large number of plane scans placed along the z-axis.

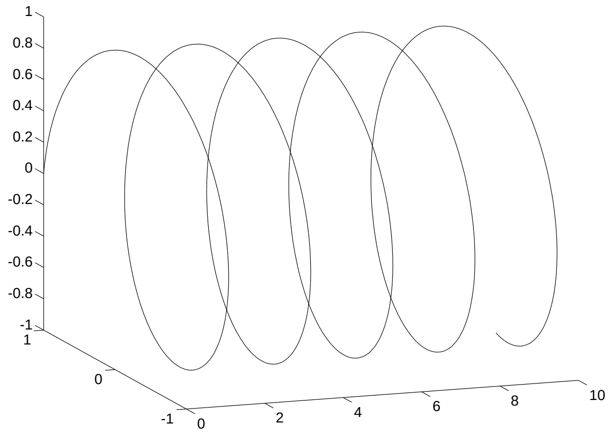

Figure 6.4 Motion of the emitter when using helical CT scanning.

It has been demonstrated that the attenuation coefficient can be measured indirectly, and the reconstruction of images can be done by use of the inverse Radon transform. In the following section another emerging technique, PET-scanning, is presented which have some similarities to the CT-scanner, but a major difference is the number of features, which can be measured, e.g., the glucose metabolism in the brain.

6.2 The PET Scanner

The PET scanner (Positron Emission Tomography) is based on another type of measurement. A small dosage of a radioactive β^+ (positron) tracer, such as F-18, O-15, or C-11, produced by use of a cyclotron is build into a larger molecule such as glucose. The β^+-labeled glucose is injected into the patient. The β^+ tracer will then be circulated in the tissue by the blood flow. If for instance the brain is to be scanned then the glucose metabolism in the individual regions of the brain will correspond to the local brain activity. In the regions with high brain activity the glucose metabolism will be high, and a corresponding number of radioactive nucleons will decay under emission of a positron. Within a few millimeters the positron will then interact with an electron and annihilate under emission of two 511 keV photons. The two photons will travel in (nearly) opposite directions outwards where the photons can be detected at detectors placed in a ring in the PET scanner, as illustrated in Fig. 6.5. Note that here only detectors are needed, as the emitter is placed within the patient.

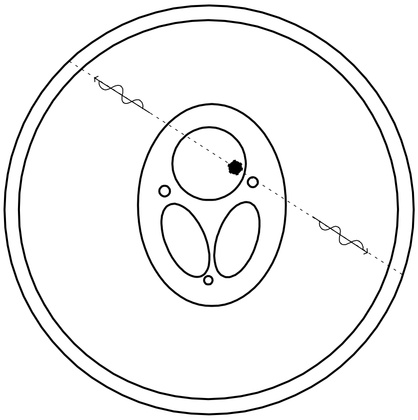

Figure 6.5 Emission of two photons from the place of decay.

In Fig. 6.6 a real PET scanner is shown - the General Electric Advance PET scanner. The photograph was taken at the PET center at the National University Hospital in Copenhagen.

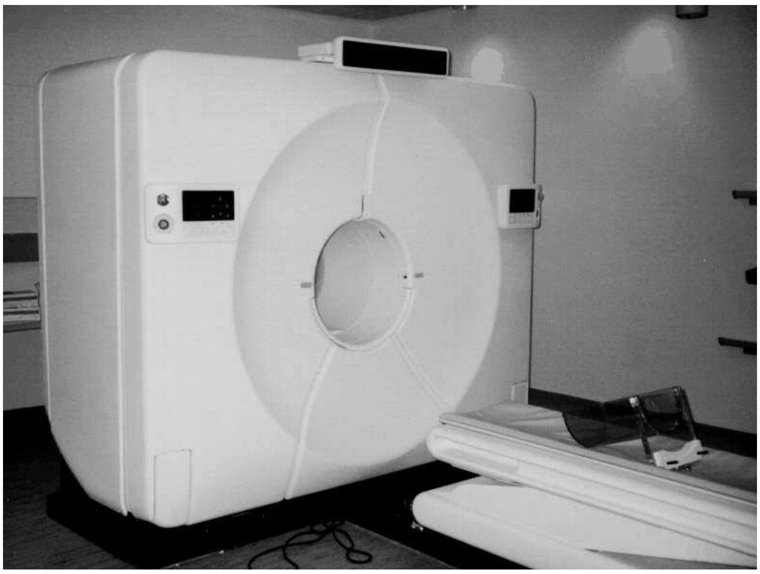

Figure 6.6 The GE Advance PET scanner.

In Fig. 6.7 is shown the same scanner with the front opened. Note the detector blocks are visible. Within a single block a number of detectors are attached, and in this particular scanner 18 rings of detector elements are found, which makes it possible to record 35 slices in 2D (number of planes in the scan) at the same time. The 35 slices is found because the scanner in 2D recording mode allows measuring photon pairs detected in the same ring or two adjacent rings.

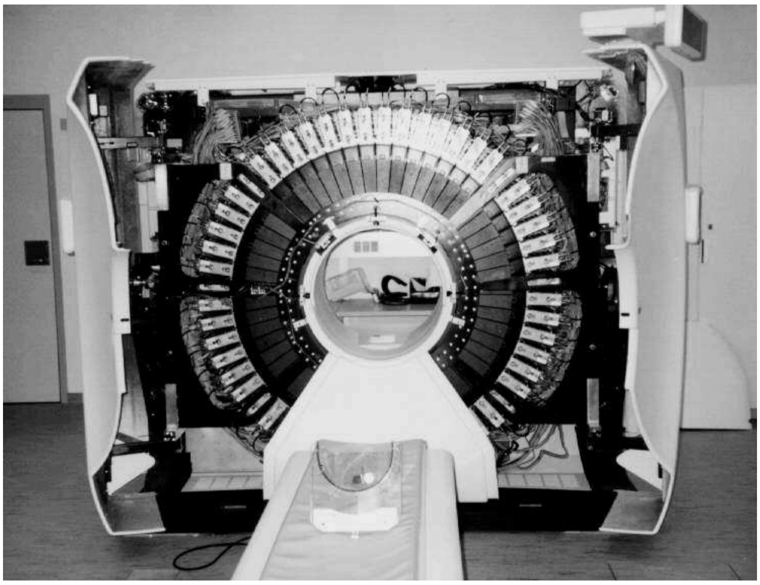

Figure 6.7 The GE Advance PET scanner opened.

The PET hardware will sort the arrival times of the photons, so only two photons that arrived (almost) at the same time at the detector ring are taken into account. The photons travel with the speed of light, and the small difference in arrival time (within approximately 2 nanoseconds) has so far only been used in the Time-of-Flight PET scanners or TOF-PET [74]. Only a very limited number of these scanners have been build. In the following the small time difference is neglected.

A two dimensional matrix of the possible detector versus detector combinations is created, and all values are initialized to zero. The number of possible detector combinations corresponds to a finite set of possible line parameters, (ρ, θ). Assume that two photons have been detected and the line between the two detectors have line parameters $(\rho, \theta) = (\rho_0, \theta_0)$, then the array is incremented with the value one at that position in the array. This is because the only obtainable information from the two photons is that the photon emission took place somewhere along that line. Depending on the radioactive dosage given to the patient, many decays take place each second in each volume element, and after a PET recording, e.g., 10^7 photon pairs can have been recorded per slice. A PET-scanning can last up to one hour or more, depending on the task of the scanning and the half-life of the tracer. Note that the majority of photon pairs are never detected, due to the limited size, and hence spherical coverage of one detector ring.

After the recording is terminated an array of emissions has been recorded. The individual emission recordings are no longer stored. For a single position or bin in the matrix at $(\rho, \theta) = (\rho_0, \theta_0)$ the array has been developed by several β^+ emissions which took place along the line. The obtained array of emissions, known as the sinogram, are in each position approximately proportional to the total emission intensity along that particular line (times the total recording time, denoted by T_e). The obtained matrix divided by T_e is in mathematical terms an approximation to the integral of the emission intensity $A(x, y)$ over that particular line. If the measured emission sinogram is named $\mathcal{E}(\rho, \theta)$ then

$$\mathcal{E}(\rho, \theta) \approx \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} A(x, y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy\ T_e \qquad (6.7)$$

Again it can be recognized that the recorded array $\mathcal{E}(\rho, \theta)$ is the Radon transform of the emission intensity $A(x, y)$. Note that this intensity will not be a constant in time as it varies with the metabolism of the brain. This may limit the allowable recording time. Note also that Eq. 6.7 is an approximation, and the approximation is not always very good, due to a very limited number of counts in the individual sinogram bin $\mathcal{E}(\rho, \theta)$. Normally the limited number of counts imply that noise is found in the reconstructed images of the activity of the patient. Nevertheless the approximation is used very often in order to apply the inverse Radon transform for reconstructing the activity pattern, even if this approach can limit the performance of the reconstruction algorithm. Later, in Chapter 9, the ML-EM algorithm is presented, which tries to incorporate the statistical nature of the emission scanning.

So far the absorption of photons in the tissue has been neglected, thus the inversion process must be modified to take this into account. The activity of the brain can be split up in a sum of point sources. Assume a point source is located in a fixed place $(x, y) = (x_0, y_0)$. Assume in the time window T_e that $A(x_0, y_0)$ pairs of photons are produced and emitted along on the line $(\rho, \theta) = (\rho_0, \theta_0)$. Assume the realistic situation that the attenuation cannot be neglected and that attenuation projections $\mathcal{P}(\rho, \theta)$ have been measured using the transmission tomography as discussed in Section 6.1. It can be mentioned that the inverse attenuation coefficient, i.e., μ^{-1}, for 511 keV photons in water, similar to tissue, is about 10 cm, and in a brain with a diameter of 20 cm approximately 86% of all photons are lost due to absorption.

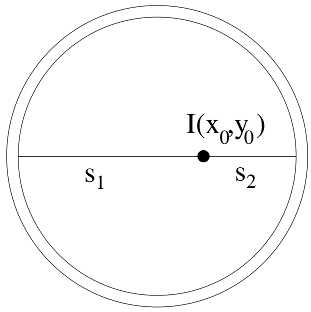

Figure 6.8 Emission of two photons from (x_0, y_0) along the line (ρ, θ).

Now consider two photons traveling from (x_0, y_0), as shown in Fig. 6.8. The photon traveling along the line piece s_1 will reach the detector ring with the probability P_1, given by

$$P_1 = e^{-\int_{s1} \mu(x,y)\; ds} \tag{6.8}$$

which is the result found in Eq. 6.4. The other photon traveling in the opposite direction along s_2 will reach the opposite detector with the probability P_2.

$$P_2 = e^{-\int_{s2} \mu(x,y)\; ds} \tag{6.9}$$

The two photons travel independently, which implies that the probability of detecting both photons is the product of P_1 and P_2. This means that the measured matrix of emissions $\mathcal{E}(\rho, \theta)$ from the point source is given by

$$\begin{aligned}
\mathcal{E}(\rho, \theta) &= A(x_0, y_0)\, T_e\, P_1\, P_2 && (6.10)\\
&= A(x_0, y_0)\, T_e\; e^{-\int_{s1} \mu(x,y)\; ds}\; e^{-\int_{s2} \mu(x,y)}\; ds && (6.11)\\
&= A(x_0, y_0)\, T_e\; e^{-\int_{s} \mu(x,y)\; ds} && (6.12)
\end{aligned}$$

where the integral in the last line is the integral of the attenuation coefficient along the line (ρ, θ), which contains (x_0, y_0). The last equation illustrates a very important conclusion, namely that attenuation correction in PET does not depend on the initial point of interest (x_0, y_0), but only on the line parameters (ρ, θ). If all point sources in the brain are considered, then the measured matrix becomes

$$\begin{aligned}
\mathcal{E}(\rho, \theta) &= \int_s A(x_0, y_0)\, T_e\, e^{-\int_s \mu(x,y)\; ds}\; ds_0 && (6.13)\\
&= e^{-\int_s \mu(x,y)\; ds} \int_s A(x, y)\, T_e\; ds && (6.14)\\
&= e^{-\mathcal{P}(\rho,\theta)} \int_s A(x, y)\; ds\; T_e \;\Rightarrow && (6.15)\\
\breve{A}(\rho, \theta) &= \int_s A(x, y)\; ds && (6.16)\\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} A(x, y)\; \delta(\rho - x\cos\theta - y\sin\theta)\; dx\; dy && (6.17)\\
&= \frac{\mathcal{E}(\rho, \theta)}{T_e}\; e^{\mathcal{P}(\rho,\theta)} && (6.18)
\end{aligned}$$

where s_0 denotes integration with respect to (x_0, y_0) along the line (ρ, θ). Note that the attenuation factor becomes a constant (along that line), and it can be moved out through the s_0-integral. Note also that the transmission correction is used in the Radon domain, thus no inversion of $\mathcal{P}(\rho, \theta)$ is needed. The important result is that the attenuation coefficient only produces a multiplicative factor in the Radon domain. This implies that the essential part of producing brain images still can be based on inversion of the Radon transform.

Here it can be mentioned that another related emission scanner, the SPECT-scanner (Single Photon Emission Computed Tomography) operates by measuring one emission photon only and here the attenuation correction is a function of position and line parameters, and not as simple as in PET.

6.2.1 Correction for Attenuation in PET

In order to obtain an estimate of $\mathcal{P}(\rho, \theta)$, the normalization factor I_0 is needed. Up to this point this factor has been assumed to be a constant. This is not true, but this problem is easily corrected by measuring a total of three scans. First an emission scan, and then two scans to correct for attenuation. A blank scan is normally made prior to the two other scans by measuring the transmitted photon intensity or rather the total number of photons transmitted along the line (ρ, θ) for T_b seconds, which gives the blank scan sinogram $\mathcal{B}(\rho, \theta)$. After the patient is placed in the scanner a transmission scan is recorded for T_t seconds (with the same setup used for the blank scan). This gives the transmission sinogram $\mathcal{T}(\rho, \theta)$. Assume that in both scans $I_0(\rho, \theta, x_r, y_r)$ is transmitted per second from the radioactive source along the line with parameters (ρ, θ) to the detector at position (x_r, y_r). Neglecting the absorption in air, then the measured matrix $\mathcal{B}$ is given by

$$\mathcal{B}(\rho, \theta) = T_b\; I_0(\rho, \theta, x_r, y_r)\; e^{-\int \mu_{\text{hardware}}\; ds} \tag{6.19}$$

where the label ‘hardware’ indicates that some parts of the scanner cannot be avoided when measuring the blank scan along the particular line. This could be the cabinet and the bed for the patient.

With the patient in the scanner, the transmission matrix $\mathcal{T}$ only differs by the absorption in the patient.

$$\mathcal{T}(\rho,\theta) = T_t \; I_0(\rho,\theta,x_r,y_r) \; e^{-\int \mu_{\text{hardware}} \; ds} \; e^{-\int \mu_{\text{patient}} \; ds} \tag{6.20}$$

From Eq. 6.19 and 6.20 the attenuation factor is easily found

$$e^{\int \mu_{\text{patient}} \; ds} = \frac{\mathcal{B}(\rho,\theta) \; T_t}{\mathcal{T}(\rho,\theta) \; T_b} \tag{6.21}$$

This implies that the Radon transform of the attenuation correction of the activity measurements, cf. Eq. 6.18, can be carried out in the Radon domain and the very important correction formula is given by

$$\check{A}(\rho,\theta) = \frac{\mathcal{E}(\rho,\theta) \; \mathcal{B}(\rho,\theta)}{\mathcal{T}(\rho,\theta)} \frac{T_t}{T_e \; T_b} \tag{6.22}$$

which reveals some potential problems. The sinogram corrected for attenuation $\check{A}$ will be highly dependent on the statistical properties of the three individual scans. Practical aspects of this issue is further discussed in Chapter 11.

6.3 Summary

Some of the fundamental properties of the CT-scanner and the PET-scanner have been presented. It was shown that the CT-scanner can measure projections of the attenuation coefficient, and the measured projections can be modelled as the Radon transform of the attenuation coefficient. This explains why the inverse Radon transform now for many years has been an important tool in medical imaging for reconstructing images of the interior of a function from a set of external measurements. In Chapter 1 of [14] it is shown that several other fields of science, such as astronomy and microscopy also can use exterior measurements and reconstruct the interior function by use of the inverse Radon transform.

Note that the description given in this chapter is somewhat simplified on several points. In practical PET several problems are common such as, scattered photons (scatter), random detection of two photons not originating from the same annihilation process (randoms), geometrical factors, penetration of photons through several detectors before detection (radial elongation), e.g., the chapter about PET in [58].

Chapter 7

Inversion of the Radon Transform

In this chapter several direct inversion schemes are presented for the Radon transform [75]. It can be noted that other inversion schemes are available but perhaps not that used, e.g., [76, 77].

7.1 The Fourier Slice Theorem

Inversion of the Radon transform can be done in several ways. One standard algorithm is based on the Fourier Slice Theorem also known as the Central Slice Theorem [78, 79]. The Radon transform $\breve{g}(\rho, \theta)$ is to be inverted into $g(x, y)$, where the inversion is based on the Fourier transform.

First the 2D Fourier Transform of $g(x, y)$ is needed.

$$G(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ e^{-j2\pi(k_x x + k_y y)}\ dx\ dy \tag{7.1}$$

and the inverse transform is given by

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y)\ e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \tag{7.2}$$

Introducing polar frequency parameters

$$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \nu\ \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \tag{7.3}$$

and inserting in Eq. 7.3 into Eq. 7.1 gives

$$\begin{aligned} G(\nu\cos\theta, \nu\sin\theta) &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ e^{-j2\pi\rho\nu}\ dx\ dy \right] d\rho \\ &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy \right]\ e^{-j2\pi\rho\nu}\ d\rho \\ &= \int_{-\infty}^{\infty} \breve{g}(\rho, \theta)\ e^{-j2\pi\rho\nu}\ d\rho \end{aligned} \tag{7.4}$$

Thus, a one-dimensional Fourier Transform of the Fourier Transform gives the spectrum, which subsequently gives $g(x, y)$, cf. Eq. 7.2. The Fourier Slice Theorem can be summarized as

$$G(\nu\cos\theta, \nu\sin\theta) = \int_{-\infty}^{\infty} \breve{g}(\rho, \theta)\ e^{-j2\pi\rho\nu}\ d\rho \tag{7.5}$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y)\ e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \tag{7.6}$$

The Fourier Slice Theorem makes it possible to invert the Radon transform by use of 1D Fourier transforms and 2D Fourier transforms. Note, that the theorem can be used in reverse order to calculate the Radon transform, if the signal $g(x, y)$ is given.

7.2 Filtered Backprojection

Another very famous inversion scheme is the Filtered Backprojection [80, 81, 14, 82] or as it more correctly should have been named Backprojection of Filtered Projections method [14]. It is derived from Eq. 7.2, by introducing polar coordinates

$$\begin{aligned} g(x,y) &= \int_0^{2\pi}\int_0^{\infty} \nu\ G(\nu\cos\theta,\nu\sin\theta)\ e^{j2\pi\nu(x\cos\theta+y\sin\theta)}\ d\nu\ d\theta \\ &= \int_0^{\pi}\int_{-\infty}^{\infty} |\nu|\ G(\nu\cos\theta,\nu\sin\theta)\ e^{j2\pi\nu(x\cos\theta+y\sin\theta)}\ d\nu\ d\theta \\ &= \int_0^{\pi}\int_{-\infty}^{\infty} |\nu|\ \left(\int_{-\infty}^{\infty}\breve{g}(\tilde{\rho},\theta)\ e^{-j2\pi\tilde{\rho}\nu}\ d\tilde{\rho}\right)\ e^{j2\pi\nu(x\cos\theta+y\sin\theta)}\ d\nu\ d\theta \end{aligned} \tag{7.7}$$

Eq. 7.7 is commonly written in two parts: A filtering part and an integration part.

$$\bar{\breve{g}}(\rho,\theta) = \int_{-\infty}^{\infty} |\nu|\ \left(\int_{-\infty}^{\infty}\breve{g}(\tilde{\rho},\theta)\ e^{-j2\pi\tilde{\rho}\nu}\ d\tilde{\rho}\right)\ e^{j2\pi\rho\nu}\ d\nu \tag{7.8}$$

$$= \mathrm{IFT}_{\nu\to\rho}\left\{|\nu|\ \mathrm{FT}_{\tilde{\rho}\to\nu}\left\{\breve{g}(\tilde{\rho},\theta)\right\}\right\} \tag{7.9}$$

$$g(x,y) = \int_0^{\pi}\bar{\breve{g}}(x\cos\theta+y\sin\theta,\theta)\ d\theta \tag{7.10}$$

$$= \int_0^{\pi}\int_{-\infty}^{\infty}\bar{\breve{g}}(\rho,\theta)\ \delta(\rho-x\cos\theta-y\sin\theta)\ d\rho\ d\theta \tag{7.11}$$

Note, the similarity of Eq. 2.4 to the forward Radon transform in Eq. 7.10.

The operation shown in Eq. 7.10 is named backprojection and it can be shown, page 120 of [14] that the backprojection operator is nearly linked to the adjoint Radon transform. To be more specific, the adjoint Radon transform is two times the backprojection operator. The concept of the adjoint Radon transform will be used in Chapter 9.

The filtering part in Eq. 7.8 is a forward Fourier transform for each of the angles. In the Fourier domain the signal is high pass filtered with the filter $|\nu|$. Finally an inverse 1D-Fourier transform is used to get the filtered Radon domain $\bar{\breve{g}}(\rho,\theta)$. The backprojection part is merely an integration along a sine-curve in the filtered Radon domain. Eqs. 7.8 and 7.10 form the Filtered Backprojection inversion scheme.

Filtered Backprojection can also be expressed without using filtering in the frequency domain. The filter $|\nu|$ is expressed as a product.

$$|\nu| = \frac{-j\ \mathrm{sign}(\nu)}{2\pi}\ j2\pi\nu\ \Rightarrow \tag{7.12}$$

$$\bar{\breve{g}}(\rho,\theta) = \mathrm{IFT}_{\nu\to\rho}\left\{\frac{-j\ \mathrm{sign}(\nu)}{2\pi}\ j2\pi\nu\ \mathrm{FT}_{\tilde{\rho}\to\nu}\left\{\breve{g}(\tilde{\rho},\theta)\right\}\right\} = \frac{1}{2\pi^2\rho} * \frac{\partial\breve{g}(\rho,\theta)}{\partial\rho} \tag{7.13}$$

where $*$ denotes a convolution in the parameter ρ. It should be noted that $|\nu|$ does not fulfill the ordinary requirements for having an inverse Fourier transform, but here the Cauchy principal value has been used in the inverse Fourier integral. Next, the Hilbert transform of a function is introduced.

$$\mathcal{H}g(t) = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{g(t')}{t-t'}\ dt'\ \Rightarrow\ \bar{\breve{g}}(\rho,\theta) = -\frac{1}{2\pi}\mathcal{H}\frac{\partial\breve{g}(\rho,\theta)}{\partial\rho} \tag{7.14}$$

In total, filtered backprojection can be rewritten on a very compact form resembling the one used by Radon [13].

$$g(x,y) = -\frac{1}{2\pi}\int_0^{\pi}\left(\mathcal{H}\frac{\partial\breve{g}(\rho,\theta)}{\partial\rho}\right)_{\rho=x\cos\theta+y\sin\theta} d\theta \tag{7.15}$$

Direct implementation from Eq. 7.15 is somewhat problematic, because of the Hilbert transformation. The Hilbert transform cannot be implemented easily without using Fourier transformation. If this is the case then the reconstruction formula matches the one given in Eq. 7.8 and 7.10.

7.3 Filtering after Backprojection

It is also possible to make the backprojection, i.e., the adjoint Radon transform before the filtering [14, 73]. In this case another filter must be used. Rewriting $g(x, y)$ using the delta function, cf. Eq. 2.16, gives

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(x-x^*)\ \delta(y-y^*)\ dx^*\ dy^* \ \Rightarrow \tag{7.16}$$

$$\breve{g}(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x^*,y^*)\ \delta(\rho - x^*\cos\theta - y^*\sin\theta)\ dx^*\ dy^* \tag{7.17}$$

Eq. 7.17 shows that each point (x^*, y^*) is transformed into a sine curve in the Radon domain. The Radon transform of $g(x, y)$ is the sum (integral) of all the sine curves in the Radon domain. In the following the integrations over x^* and y^* are omitted. This can be done because only linear transforms are used. Now assume a point source is given and the adjoint Radon transform is applied before a filtering.

$$g(x,y) = \delta(x-x^*)\ \delta(y-y^*) \ \Rightarrow \tag{7.18}$$

$$\breve{g}(\rho,\theta) = \delta(\rho - x^*\cos\theta - y^*\sin\theta) \tag{7.19}$$

$$\hat{g}(x,y) = \int_0^{\pi} \breve{g}(x\cos\theta + y\sin\theta, \theta)\ d\theta \tag{7.20}$$

$$= \left.\frac{1}{|(x-x^*)\sin\theta - (y-y^*)\cos\theta|}\right|_{(x-x^*)\cos\theta + (y-y^*)\sin\theta = 0} \tag{7.21}$$

$$= \frac{1}{\left|-(x-x^*)\sin\arctan\left(\frac{x-x^*}{y-y^*}\right) - (y-y^*)\cos\arctan\left(\frac{x-x^*}{y-y^*}\right)\right|} \tag{7.22}$$

$$= \frac{\sqrt{1+\left(\frac{x-x^*}{y-y^*}\right)^2}}{\frac{(x-x^*)^2}{y-y^*} + (y-y^*)} \ = \ \frac{1}{\sqrt{(x-x^*)^2 + (y-y^*)^2}} \tag{7.23}$$

It can be recognized that this is a two dimensional convolution.

$$\hat{g}(x,y) = g(x,y) ** h(x,y) \quad \text{where } h(x,y) = \frac{1}{\sqrt{x^2+y^2}} \tag{7.24}$$

Using the technique shown in Eq. 7.17 it can be seen that Eq. 7.24 is valid for any given $g(x, y)$, Eq. 7.24 thus enables another inversion scheme. A two dimensional Fourier transform of Eq. 7.24 gives

$$\hat{G}(k_x,k_y) = G(k_x,k_y)\ H(k_x,k_y) \Rightarrow \tag{7.25}$$

$$G(k_x,k_y) = \frac{\hat{G}(k_x,k_y)}{H(k_x,k_y)} \tag{7.26}$$

From standard 2D Fourier transform tables it can be found that

$$h(x,y) = \frac{1}{\sqrt{x^2+y^2}} \ \leftrightarrow \ H(k_x,k_y) = \frac{1}{\sqrt{k_x^2+k_y^2}} \tag{7.27}$$

This means that $g(x,y)$ can be found as

$$\hat{g}(x,y) = \int_0^{\pi} \check{g}(x\cos\theta + y\sin\theta, \theta)\ d\theta \tag{7.28}$$

$$\hat{G}(k_x,k_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \hat{g}(x,y)\ e^{-j2\pi(k_x x + k_y y)}\ dx\ dy \tag{7.29}$$

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \sqrt{k_x^2+k_y^2}\ \hat{G}(k_x,k_y)\ e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \tag{7.30}$$

The reconstruction method presented is here called Filtering after Backprojection, which makes use of an integration succeeded by a two dimensional high pass filtering. This is an inversion algorithm very similar to Filtered Backprojection.

7.4 Calculation using Operators

A number of textbooks, e.g. [14, 83], define operators. In this section relevant operators are defined, which will enable an easy way to write the inversion algorithms. In the following x, y, k_x, and k_y denotes continuous variables. To identify the continuous operators calligrafic letters are used, e.g., $\mathcal{R}$.

$$\mathcal{R}g(x,y) = \check{g}(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ dx\ dy \tag{7.31}$$

$$\mathcal{B}h(\rho,\theta) = \int_0^{\pi} h(x\cos\theta + y\sin\theta,\theta)\ d\theta \tag{7.32}$$

$$\mathcal{F}_{p1}^{-}h(\rho,\theta) = \int_{-\infty}^{\infty} h(\tilde{\rho},\theta)\ e^{-j2\pi\tilde{\rho}\nu}\ d\tilde{\rho} \tag{7.33}$$

$$\mathcal{F}_{p1}^{+}H(\nu,\theta) = \int_{-\infty}^{\infty} H(\nu,\theta)\ e^{j2\pi\rho\nu}\ d\nu \tag{7.34}$$

$$\mathcal{F}_{r2}^{-}h(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x,y)\ e^{-j2\pi(k_x x + k_y y)}\ dx\ dy \tag{7.35}$$

$$\mathcal{F}_{r2}^{+}H(k_x,k_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} H(k_x,k_y)\ e^{j2\pi(k_x x + k_y y)} dk_x\ dk_y \tag{7.36}$$

$$\mathcal{C}h(\rho,\theta) = \int_{-\infty}^{\infty} |\nu| \left(\int_{-\infty}^{\infty} h(\tilde{\rho},\theta)\ e^{-j2\pi\tilde{\rho}\nu} d\tilde{\rho}\right) e^{j2\pi\rho\nu}\ d\rho \tag{7.37}$$

$$\mathcal{D}h(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \sqrt{k_x^2+k_y^2} \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tilde{x},\tilde{y})\ e^{-j2\pi(k_x\tilde{x}+k_y\tilde{y})} d\tilde{x}\ d\tilde{y}\right) e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \tag{7.38}$$

Using the symbolic operators it is easy to express the presented inversion algorithms in a short form

Filtered Backprojection

$$\mathcal{B}\mathcal{F}_{p1}^{+}\ |\nu|\ \mathcal{F}_{p1}^{-}\ \check{g} = \mathcal{B}\mathcal{C}\ \check{g} = g(x,y)\ \Leftrightarrow\ \mathcal{B}\mathcal{F}_{p1}^{+}\ |\nu|\ \mathcal{F}_{p1}^{-}\ \mathcal{R} = \mathcal{B}\mathcal{C}\mathcal{R} = \mathcal{I} \tag{7.39}$$

where $\mathcal{I}$ is the unity operator, i.e., $\mathcal{I}g = g$.

Filtering after Backprojection

$$\mathcal{F}_{2r}^{+}\ \sqrt{k_x^2+k_y^2}\ \mathcal{F}_{2r}^{-}\ \mathcal{B}\ \check{g} = \mathcal{D}\mathcal{B}\check{g} = g(x,y) \Leftrightarrow \tag{7.40}$$

$$\mathcal{F}_{2r}^{+}\ \sqrt{k_x^2+k_y^2}\ \mathcal{F}_{2r}^{-}\ \mathcal{B}\ \mathcal{R} = \mathcal{D}\mathcal{B}\mathcal{R} = \mathcal{I} \tag{7.41}$$

Fourier Slice Theorem

$$\mathcal{F}_{r2}^{+}\,\mathcal{F}_{p1}^{-}\,\check{g} = g(x,y) \;\Leftrightarrow\; \mathcal{F}_{r2}^{+}\,\mathcal{F}_{p1}^{-}\,\mathcal{R} = \mathcal{I} \tag{7.42}$$

where a conversion between polar and rectangular frequency coordinates is implied.

7.4.1 The Zero Frequency Problem

In the previous sections a perfect inversion is assumed, e.g., Eq. 7.41, but the backprojection algorithms cannot invert perfectly, due to the filters used in the inversion. Using Filtered Backprojection it is seen that the polar spectrum at zero frequency ($\nu = 0$) is multiplied with zero. This implies that a non-zero mean-value in the sinogram is set to zero in *all* cases, hence the mean value of the reconstructed image need not be correct. Using Backprojection after Filtering, it is obvious that the zero frequency value of the inverted image $g(x,y)$ must becomes zero. The Fourier Slice Theorem also shows that problem when considering the discrete implementation.

7.5 Sampling Considerations

In the previous sections it was shown that direct reconstruction formulas use the adjoint Radon transform, in form of the backprojection operator, and/or the 1D/2D Fourier transform. This suggests that the presented direct reconstruction schemes can be implemented using a sum approximation for the adjoint Radon transform and FFT/IFFT for the Fourier transforms, but it calls for proper discretization of the continuous formulas and careful selection of the sampling parameters.

In most implementations a linear sampling of all variables is used.

$$\begin{array}{llll} x = x_m & = x_{min} + m\Delta x & m = 0,1,...,M-1 \\ y = y_n & = x_{min} + n\Delta y & n = 0,1,...,N-1 \\ \rho = \rho_r & = \rho_{min} + r\Delta\rho & r = 0,1,...,R-1 \\ \theta = \theta_t & = \theta_{min} + t\Delta\theta & t = 0,1,...,T-1 \end{array} \tag{7.43}$$

and a centered square image will be assumed

$$\Delta x = \Delta y \tag{7.44}$$

$$M = N \tag{7.45}$$

$$x_{min} = -x_{max} = -\frac{(M-1)}{2}\Delta x \tag{7.46}$$

Note, that this choice will in general not give optimum numerical stability, but implies fast and fairly easy implementations. See [72] concerning a clever use of non-linear sampling of the parameter domain.

Eq. 2.10 gives a choice when choosing the limits of the parameter domain. It is possible to choose $\rho \geq 0$, which normally implies that $0 \leq \theta < 2\pi$. It is also possible to choose $0 \leq \theta < \pi$ and ρ both positive and negative. This choice must be done considering the specific implementation. In the following it is chosen to use $0 \leq \theta < \pi$ and $\theta_{min} = 0$, which implies that $\Delta\theta = \frac{\pi}{T}$, because this choice matches the limits used in, e.g., Filtered Backprojection and those used in Chapter 2.

When implementing a Radon (or inverse Radon) transform algorithm several things must be fulfilled, to ensure a reasonable performance. Firstly sampling must be adequate in all parameters. This will imply bounds on the sampling intervals. Secondly it is assumed that the fundamental function $g(x,y)$ to be reconstructed have compact support, or more precisely is zero if $\sqrt{x^2+y^2} > |\rho_{max}|$. This demand will ensure that $\breve{g}(\rho,\theta) = 0$, if $|\rho| > |\rho_{max}|$. If this cannot be fulfilled, numerical problems must be expected, because then a numerical implementation does not have all the non-zero information necessary for reconstructing the function.

Assuming that $g(x,y)$ has compact support, then $(x,y) = (0,0)$ should be placed to minimize $|\rho_{max}|$. This will reduce the size of the data array (with respect to ρ) used for the discrete Radon transform. Normally ρ_r is placed symmetrically around $\rho = 0$, and r is chosen to be odd. In this case

$$\rho_{min} = -\Delta\rho\ \frac{R-1}{2} \tag{7.47}$$

is used. This choice implies that $\rho_{min} = -\rho_{max}$, like it was chosen in Subsection 2.2.1. The parameter M can then be chosen to fulfill that the whole sinogram is confined in the reconstructed image, in order to reconstruct an image from all of the non-zero parts of the sinogram.

$$\max_{x_m,y_n} \sqrt{x^2+y^2} = x_{min}\frac{\sqrt{2}}{\Delta x} > \rho_{max} \ \Rightarrow\ M > \left\lceil \sqrt{2}\frac{|\rho_{min}|}{\Delta x} \right\rceil + 1 \tag{7.48}$$

From a given sinogram the only parameter still needed to be determined is the sampling distance Δx. Unfortunately the sinc-interpolation strategy used in Subsection 2.2.1 cannot be used. It seems to be difficult to make a series expansion from a finite set in the Radon domain and reconstruct the set using one of the direct reconstruction schemes presented in previous sections in this chapter. Furthermore the optimal choice of image sampling distance Δx will also be determined by additional filters used in the discrete implementation. In general a lower bound is $\Delta\rho = \Delta x$, but if a low-pass filter is used on the sinogram or the structures being scanned are large, then a coarser can be used.

7.6 Inversion of the (p,τ) Radon Transform

In this section inversion formulas are derived for slant stacking. They all are equivalents forms of the ones derived for the (ρ,θ) Radon transform. This section has been included to support Chapter 1, but will not be supported further. The rest of the thesis will focus on the normal Radon transform. For further reading on the use of the inverse (p,τ) Radon transform see [36, 84, 85, 44]. It can be mentioned that the numerical implementation of the inversion schemes will resemble closely the ones presented in Chapter 8.

In this section the fundamental function is $g(x,y)$. Within the seismic literature the function in question normally depend on a distance parameter x and a time t, and the 2D Fourier transform has physical interpretation as a plane wave-expansion. Here the parameters have been chosen according to the rest of the thesis. The Radon transform using parameters (p,τ), i.e., the slant stack of a function $g(x,y)$ is assumed

$$\breve{g}(p,\tau) = \int_{-\infty}^{\infty} g(x, px+\tau)\ dx \tag{7.49}$$

7.6.1 Fourier Slice Theorem

The derivation of the inversion formula is based on the two dimensional Fourier transform of $g(x, y)$

$$G(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y)\ e^{-j2\pi(k_x x + k_y y)}\ dx\ dy \tag{7.50}$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y)\ e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \tag{7.51}$$

Now two new parameters are introduced. Firstly, the negative ratio between the two frequencies k_x and k_y is denoted p, and secondly the offset τ, given by

$$p = -\frac{k_x}{k_y} \tag{7.52}$$

$$\tau = y - px \tag{7.53}$$

If these are inserted into Eq. 7.50 it is found that

$$G(-pk_y, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, px + \tau)\ e^{-j2\pi k_y \tau}\ dx\ dy \tag{7.54}$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} g(x, px + \tau)\ dx \right]\ e^{-j2\pi k_y \tau}\ dy \tag{7.55}$$

$$= \int_{-\infty}^{\infty} \breve{g}(p, \tau)\ e^{-j2\pi k_y \tau}\ dy \tag{7.56}$$

The last equation can be recognized as the 1D Fourier transform of Radon transform, hence an inversion formula is found by combining Eqs. 7.56 and 7.51

$$\begin{aligned} G(-pk_y, k_y) &= \int_{-\infty}^{\infty} \breve{g}(p, \tau)\ e^{-j2\pi k_y \tau}\ dy \\ g(x, y) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y)\ e^{j2\pi(k_x x + k_y y)}\ dk_x\ dk_y \end{aligned} \tag{7.57}$$

This Fourier Slice Theorem shows that the function $g(x, y)$ can be reconstructed by mapping the 1D Fourier transform of the Radon transform into the 2D Fourier spectrum of $g(x, y)$. Again the problem regarding high slopes, pointed out in Section 1.7, reappears. If the Radon transform only is given in a certain interval, e.g., between -1 and 1, then only a bow tie shaped part of the 2D Fourier domain domain can be provided from the 1D Fourier transform of the Radon transform, which is shown in Fig. 7.1.

This implies that the function $g(x, y)$ only can be reconstructed if the main Fourier components are concentrated at the k_y-axis. This demand can again be translated into the image domain, demanding that the function cannot have line-like objects with a slope higher than the one used when computing the Radon transform, i.e., if the limiting slope in the parameter domain is chosen to small, then the obvious result is that the function cannot be recovered from its Radon transform.

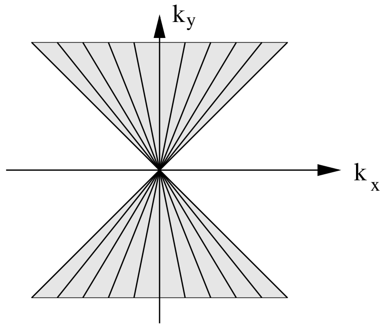

Figure 7.1 From the Radon transform only a bow tie shaped part of the The 2D Fourier domain can be covered.

7.6.2 Filtered Backprojection

Eq. 7.57 can also be combined with Eq. 7.53 in order to get a Filtered Backprojection inversion formula for slant stacking

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(-pk_y,k_y)\ e^{j2\pi k_y\tau}\ |k_y|\ dp\ dk_y\Big|_{\tau=y-px} \tag{7.58}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |k_y|\ G(-pk_y,k_y)\ e^{j2\pi k_y\tau}\ dk_y dp \tag{7.59}$$

$$= \int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty}|k_y|\left(\int_{-\infty}^{\infty}\breve{g}(p,\tilde{\tau})\ e^{-j2\pi k_y\tilde{\tau}}\ d\tilde{\tau}\right)e^{j2\pi k_y\tau}\ dk_y\right)_{\tau=y-px} dp \tag{7.60}$$

For sake of clarity, Eq. 7.60 is written in two parts; a filtering part and a backprojection part

$$\bar{\breve{g}}(p,\tau) = \int_{-\infty}^{\infty}|k_y|\left(\int_{-\infty}^{\infty}\breve{g}(p,\tilde{\tau})\ e^{-j2\pi k_y\tilde{\tau}}\ d\tilde{\tau}\right)e^{j2\pi k_y\tau}\ dk_y \tag{7.61}$$

$$g(x,y) = \int_{-\infty}^{\infty}\bar{\breve{g}}(p,y-px)\ dp \tag{7.62}$$

where the last equation also is called the backprojection operator.

This Filtered Backprojection formula for computing the inverse Radon transform implies that the parameter domain is filtered with the absolute frequency in the y-direction for all values of x, and then the backprojection part integrates up along a line. Note that the linear expression in the backprojection integral is like the one found with the point source, i.e., the point sources have by the Radon transform been spread into the parameter domain, and by the backprojection part, the filtered parameter domain is collected again.

7.6.3 An Inversion Formula using the Hilbert Transform

The next inversion formula mentioned here is on the same form as Radon derives in [13]. Eq. 7.60 is now written using that the Fourier transform of a product becomes a convolution (here marked by $*$)

$$\begin{aligned} g(x,y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}|k_y|\ G(-pk_y,k_y)\ e^{j2\pi k_y\tau}\ dp\ dk_y \text{ where } \tau = y-px \\ &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}|k_y|\ e^{j2\pi k_y\tau}\ dk_y\ *\ \int_{-\infty}^{\infty}G(-pk_y,k_y)\ e^{j2\pi k_y\tau}\ dk_y\right]dp \\ &= \int_{-\infty}^{\infty}\frac{-1}{2\pi^2\tau^2}\ *\ \breve{g}(p,\tau)\ dp \end{aligned} \tag{7.63}$$

where it again should be noted that $|k_y|$ does not fulfill the ordinary requirements for having an inverse Fourier transform, but here the Cauchy principal value has been used in the inverse Fourier integral. The result can be found in, e.g., [86] (page 13.2 F26). Now the expressions are rearranged furthermore by inserting the convolution

$$\begin{aligned}
g(x,y) &= \int_{-\infty}^{\infty} \frac{-1}{2\pi^2\tau^2} \ast \breve{g}(p,\tau)\; dp \text{ where } \tau = y - px && (7.64)\\
&= \frac{-1}{2\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\breve{g}(p,\tilde{\tau})}{(\tau-\tilde{\tau})^2}\; d\tilde{\tau}\; dp && (7.65)\\
&= \frac{1}{2\pi^2}\frac{d}{d\tau}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\breve{g}(p,\tilde{\tau})}{\tau-\tilde{\tau}}\; d\tilde{\tau}\; dp && (7.66)\\
&= \frac{1}{2\pi^2}\frac{d}{dt}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\breve{g}(p,\tilde{\tau})}{\tau-\tilde{\tau}}\; d\tilde{\tau}\; dp && (7.67)\\
&= \frac{1}{2\pi^2}\frac{d}{dy}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\breve{g}(p,\tilde{\tau})}{y-px-\tilde{\tau}}\; d\tilde{\tau}\; dp && (7.68)\\
&= \frac{1}{2\pi^2}\frac{d}{dy}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\breve{g}(p,\tilde{y}-px)}{y-\tilde{y}}\; dy\; dp && (7.69)
\end{aligned}$$

The last equation can be written on a shorter form by use of the Hilbert transform

$$\begin{aligned}
\mathcal{H}\{f(x,y)\} &= \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{f(x,\tilde{y})}{y-\tilde{y}}\; d\tilde{y} \;\Rightarrow && (7.70)\\
g(x,y) &= \frac{1}{2\pi}\frac{d}{dy}\mathcal{H}\left\{\int_{-\infty}^{\infty} \breve{g}(p,\tilde{y}-px)\; dp\right\} && (7.71)
\end{aligned}$$

This equation is very concise, but not that easy to implement on a discrete form. The inner part of the inversion formula can be discretized by use of Section 1.4. The remaining part requires a discrete approximation of the Hilbert transform and the derivative operator. Note, that the Hilbert transform has a singularity which makes the discrete implementation difficult.

7.6.4 Filtering after Backprojection

The last inversion form is equivalent to Eq. 7.71. A way to circumvent the implementation problems with the Hilbert transform is to note that Eq. 7.71 can be written in two stages

$$\begin{aligned}
\tilde{g}(x,y) &= \int_{-\infty}^{\infty} \breve{g}(p,y-px)\; dp && (7.72)\\
g(x,y) &= \frac{1}{2\pi}\frac{d}{dy}\mathcal{H}\left\{\tilde{g}(x,y)\right\} && (7.73)
\end{aligned}$$

and using that the last equation is a convolution, then the inversion scheme becomes a filter in the Fourier domain.

$$\left.\begin{array}{ll}
\text{The Hilbert transform:} & \frac{1}{\pi y} \leftrightarrow -j\,\mathrm{sign}(k_y)\\
\text{The derivative operator:} & \frac{dg(y)}{dy} \leftrightarrow j2\pi k_y G(k_y)
\end{array}\right\} \Rightarrow \qquad (7.74)$$

$$g(x,y) = \int_{-\infty}^{\infty} |k_y| \left[\int_{-\infty}^{\infty} \tilde{g}(x,y)\; e^{-j2\pi k_y y}\; dy\right] e^{j2\pi k_y \tau}\; dk_y \qquad (7.75)$$

where sign$(\cdot)$ is the sign function. The conclusion is that the inversion formula proposed by Radon can be interpreted as a backprojection followed by a filtering operation. A very important feature is that the reconstructed zero frequency component value becomes zero for any fixed value of x.

7.7 Summary

In this section several formulas for computing the inverse Radon transform have been derived. It has been shown that they can be based on Fourier techniques and integral transformations. Formulas are provided both using the normal Radon transform and for the slant stack definition. Some remarks have been given, regarding that the inverse Radon transform can not recover the mean value correctly.

Considerations concerning the sampling of the continuous parameters are given, which will be used in the following chapters.

Chapter 8

Numerical Implementation of Direct Reconstruction Algorithms

Several authors discuss implementations of the direct reconstruction schemes presented in Chapter 7, e.g., [81, 78, 79, 16]. In this chapter the implementation of Filtered Backprojection, Filtering after Backprojection and the Fourier Slice Theorem are shown. All of these algorithms can be based on the Fourier transform, which is reviewed in the next section. This theory might be considered basic, but it appears that applying the DFT as a discrete approximation to the Fourier transform is requires special care with phase and normalization.

8.1 Using the DFT to Approximate the Fourier Transformation

All of the presented direct reconstruction schemes use Fourier transformation, either for filtering or for re-mapping of spectra (Fourier Slice Theorem). For digital signals the discrete Fourier transform (DFT) can be used to estimate the spectrum, and in practice the spectrum is estimated by the Fast Fourier Transform (FFT).

Assume that the one dimensional function, of a continuous variable t, denoted $g(t)$ is sampled uniformly, i.e., $g_s(n) = g(n\Delta t)$. Furthermore, assume that only N values are non-zero. In this case the Fourier transform can be approximated by the DFT

$$G(f) = \int_{-\infty}^{\infty} g(t)\ e^{-j2\pi ft}\ dt \approx \Delta t \sum_{n=0}^{N-1} g_s(n)\ e^{-j2\pi mn/N} = G_s(m) \tag{8.1}$$

hence, $G_s(m)$ will approximate the continuous spectrum

$$\Delta t G_s(m) \approx G\left(\frac{m}{N\Delta t}\right) \tag{8.2}$$

The discrete spectrum is in Eq. 8.1 computed using the DFT of $g_s(n)$. With the same approach the inverse Fourier transform is commonly approximated by

$$g_a(t) = \int_{-\infty}^{\infty} G_a(f)\ e^{j2\pi ft}\ df \approx \frac{1}{N\Delta t} \sum_{m=0}^{N-1} G_d(m)\ e^{j2\pi mn/N} = g_d(n) \tag{8.3}$$

This technique can easily be generalized to two or more dimensions. Note that the inverse Fourier transform can also be implemented using the FFT [87].

When approximating the Fourier transform by the DFT it should be noted that the spectrum is available only in discrete samples and the samples represents the Fourier transform of a periodical repetition of the discrete signal $g_s(n)$ with period N.

Furthermore a very important factor is that the discrete spectrum as a function of m also is periodical with period N, i.e.,

$$G_s(m) = G_s(m+N) \tag{8.4}$$

This implies that the maximum absolute frequency corresponds to $m = N/2 \Rightarrow f_u = \frac{1}{2\Delta t}$, i.e., the upper frequency f_u equals half of the sampling frequency. Due to the periodical behavior of the discrete spectrum, the last half of the digital spectrum, i.e., from $m = N/2$ to $m = N-1$ will correspond to negative frequencies.

8.1.1 The FFT Applied for Filtering

Now the practical use of the DFT for implementation of the operator $\mathcal{F}_{p1}^{-}$ will be discussed. It is very important to note that the DFT uses an array of signal values $g(n),\ n = 0, 1, ..., N-1$ where the last half of the array corresponds to negative continuous variable. In this case the relevant signal is the discrete samples of the sinogram for a certain value of θ_t, i.e., $h(r) = \breve{g}(\theta_t, \rho_r)$, where the values of ρ lies symmetrically around 0, cf. Eqs. 7.43 and 7.47. In order to get the phase of the spectrum correct, the array $g(n)$ used for the DFT must be filled as shown in Fig. 8.1. Note the wrapping so negative values of ρ are filled into the last half of the array. It is also assumed that the DFT transformation length is a power of two in order to use one of the fast radix-2 FFT algorithms. For the unknown entries in the middle of the array of g(n) zeros must be filled in. This operation is called zero padding and this action will affect the spectrum.

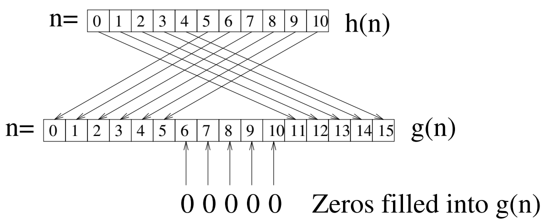

Figure 8.1 Filling an array h(n) into a larger array g(n), when using radix-2 FFT.

A large number of zeros gives a more smooth spectrum (due to a denser sampling in frequency domain), which can be desired if interpolation in the spectrum is required. Thus, the extra zeros padded can help to improve the numerical stability. In Fig. 8.2 a square signal is shown in the left upper corner, and the corresponding spectrum in the right upper corner. Zeros have been padded, and again the spectrum has been found. This is shown on the lower part of the figure. Note, that the new spectrum is more smooth.

As it briefly has been described, the use of DFT to approximate the Fourier transform also implies that the signal becomes cyclical with the period of the transformation length. This can lead to problems, as it later will be demonstrated in Subsection 8.5.1. A common strategy to reduce this cyclical influence is to multiply the signal with a window, i.e., a weight function attenuating the edges of the signal.

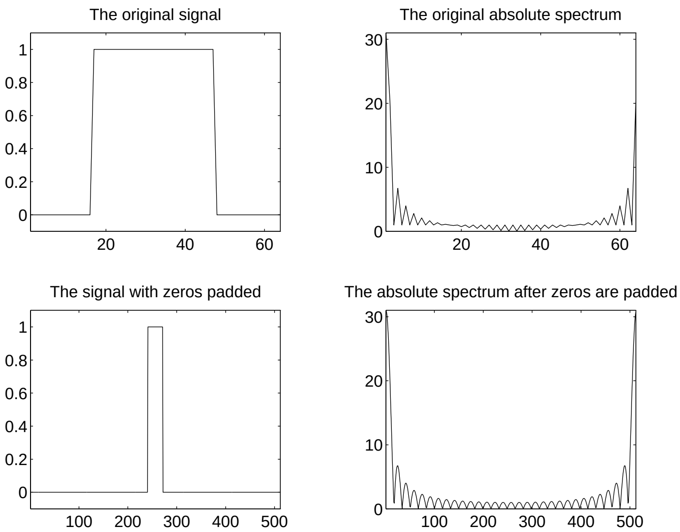

Figure 8.2 Upper left shows a square and the upper right shows the corresponding absolute spectrum. The frequency ranges from 0 to the sampling frequency (last half is the negative frequencies). Lower left shows the square where zeros have been padded, and lower right shows the corresponding absolute spectrum, which appears more smooth.

A property, which can be used to reduce the computational cost is based on real (non-complex) valued signals. Assume a discrete cyclical real valued signal $g(n)$

$$G(m) = G(-m)^* = G(N-m)^* \tag{8.5}$$

where * denotes the complex conjugate and $g(n) = g(n+N)$. This symmetry is easily proved from the DFT definition shown in Eq. 8.1, and it should be used if only a real signal is provided, such as a sinogram which should be filtered (without complex values). In this way the length of the FFT needed can be reduced by a factor of two, or two real valued signals can be transformed with the same FFT [88, 89].

Numerical algorithms are available in virtually any numerical package for efficient calculation of the FFT. Several packages furthermore also includes functions for calculating the DFT of a real valued sequence $g(n)$ of length $N = 2^p$, e.g., Numerical Recipes [90]. Often the FFT-algorithm is a radix-2 algorithm. This restriction is for reconstruction purposes not harsh, but must be remembered when zeros are padded.

In the implementation of Filtered Backprojection a 1D filtering of the sinogram is needed, where the filter is the absolute of the frequency parameter, i.e., $|\nu|$, cf. Eq. 7.8. This filter is approximated in many ways, but the structure of the FFT based filtering is the same, as shown in Algorithm 8.1. The algorithm does not demonstrate the implementation of the extra speedup gained by exploiting Eq. 8.5, hence the array g(n) has complex entries. Note also that the algorithm does not split the signal values as it was shown in Fig. 8.1, because the filtering does not use the actual phase of the spectrum. What matters, is that the filtered result is extracted from the same positions in the array, and is returned in the original sinogram (g_radon(t,r)), which is done for sake of memory efficiency.

The filter calculated in line three of Algorithm 8.1 (sampling of Eq. 8.2) is called the ramp filter or Ram-Lak filter. Often it is multiplied with a weight function or simply another function in order to get a better signal to noise ratio. The only item all of the filters have in common is that they approximate $|\nu|$ at low frequencies, and the difference is pronounced as the frequency ν approaches half of the sampling frequency, denoted by $\nu_u = \frac{1}{2\Delta\rho}$.

Algorithm 8.1 : Filtering using DFT

```
Set P to upper power of two (P≥R)          //Assuming radix-2 FFT
For r = 1 to P/2                           //Initialize Filter
  f(r) = r/(Delta_rho*P)                   //Approximate filter
End
For t = 0 to T-1                           //For all angular samples
  For r = 0 to R-1                         //For the radial samples
    g(r) = g_radon(t,r)                    //Move the sinogram values
  For r = R to P-1                         //For padding
    g(r) = 0                               //Pad with zeros
  End
  Compute FFT of g(n)                      //Using radix-2 FFT
  g(0) = 0                                 //Handle zero freq. specially
  g(P/2) = g(P/2)*f(P/2)                   //Handle half sampling freq.
  For r = 1 to P/2-1                       //Multiply with filter
    g(r) = g(r)*f(r)                       //Positive frequency
    g(P-r) = g(P-r)*f(r)                   //Negative frequency
  End
  Compute Inverse FFT of g(r)              //Using radix-2 FFT
  For r = 0 to R-1                         //For the radial samples
    g_radon(t,r)=real(g(r))                //Move the real-part
  End
End
```

Some of the filters reported in the literature [73, 58] are given below, where it is only the the part below a certain limit frequency $\nu_l \leq \nu_u$, which is sampled and used. The reason for using these weight functions also called windows or apodizing functions is to suppress the influence of noise. It is obvious that the filter $|\nu|$ is a high pass filter and it will attenuate any noise present in the sinogram. Examples of this property will be given in Subsection 8.5.5. Many windows can be presented, but here only four examples are given

The Cropped Ram-Lak/Ramp filter Sample the filter $|\nu|$ until ν_l, i.e., the filter is

$$H(\nu)_{\text{Ram-Lak}} = |\nu| \tag{8.6}$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero. The ramp filter is widely used but will amplify noise if ν_l is chosen too high.

The Shepp-Logan filter A sinc window is multiplied to the ramp filter

$$H(\nu)_{\text{Shepp-Logan}} = |\nu| \frac{1}{2} \frac{\sin\left(\frac{\pi\nu}{2\nu_l}\right)}{\left(\frac{\pi\nu}{2\nu_l}\right)} \tag{8.7}$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

The Hann filter A Hann window is multiplied to the filter

$$H(\nu)_{\text{Hann}} = |\nu| \left(1 + \cos\left(\frac{\pi\nu}{\nu_l}\right)\right) \tag{8.8}$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

The Generalized Hamming filter

$$H(\nu)_{\text{Generalized Hamming}} = |\nu|(\alpha + (1-\alpha)\cos(\pi\nu/\nu_u)) \tag{8.9}$$

where typical values of α are $0.5 - 0.54$, cf. page rec-22 of [58]. Again for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

Stochastic Filters In Section 10.8 of [73], Jain derives a parameterized filter, which is shaped to the actual noise level.

Fig. 8.3 shows three of the filters. They can all be written as a product of the theoretical derived ramp filter and an apodizing window, which will influence the performance in presence of noise. In general, if the apodizing window have a low cutoff frequency the resolution gets worse, but the noise suppression can be improved.

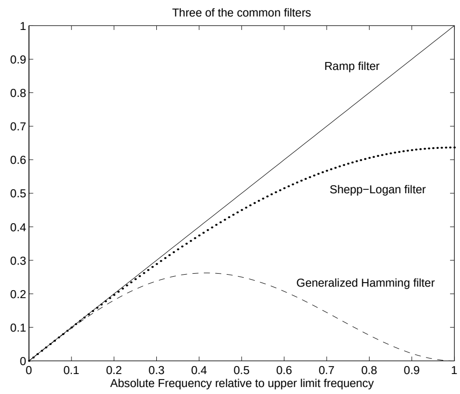

Figure 8.3 The amplitude of the Ram-Lak/Lamp filter, the Shepp-Logan filter, and the generalized Hamming filter using $\alpha = 0.5$, all three as a function of frequency normalized to the upper limit frequency ν_u.

In the top part Fig. 8.4 is shown the impulse response from the ramp filter. Note that the lower sub-figure uses logarithmic scale. The figure shows that the impulse response has long tails which implies that a number of zeros, in principle an infinite number, have to be padded to the signal or else the cyclical behavior of the DFT can influence the filtered signal.

Fig. 8.5 shows filtering of a sinogram corresponding to a circular disc in the image domain. Here a Hann window has been multiplied to the ramp filter. This filtered sinogram will later, in Fig. 8.7, be backprojected to demonstrate the full reconstruction algorithm of Filtering after Backprojection. From Fig. 8.5 it can be seen that the spectrum is very localized around zero frequency, and the high-pass filter will amplify the edges and from the last sub-figure, it can be seen that significant negative values are found. A small remark is that the data representation in these algorithms should include a sign bit.

It should be mentioned that the complexity of filtering the sinogram is the number of angular samples times the complexity of the FFT operations (plus some lower order terms)

$$\mathcal{O}_{\text{Filtering}} = \mathcal{O}(T\ R\ \log R) \tag{8.10}$$

where R should be replaced by the smallest power of two larger or equal to R if a radix-2 FFT is used.

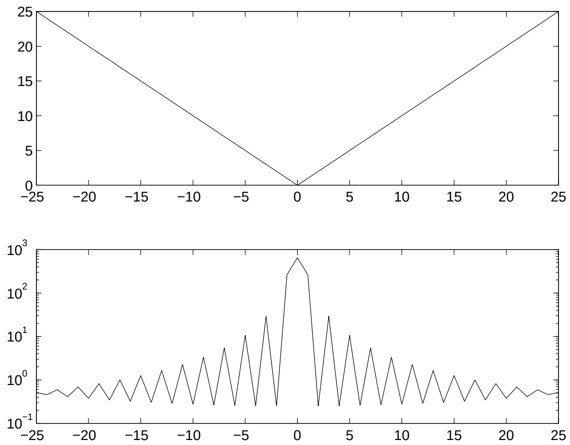

Figure 8.4 Upper shows 51 samples of a ramp filter as a function of frequency. Upper right shows the absolute value of the corresponding spectrum, found from a DFT of the same length. Here no windowing has been used to reduce the cyclical behaviour of the DFT.

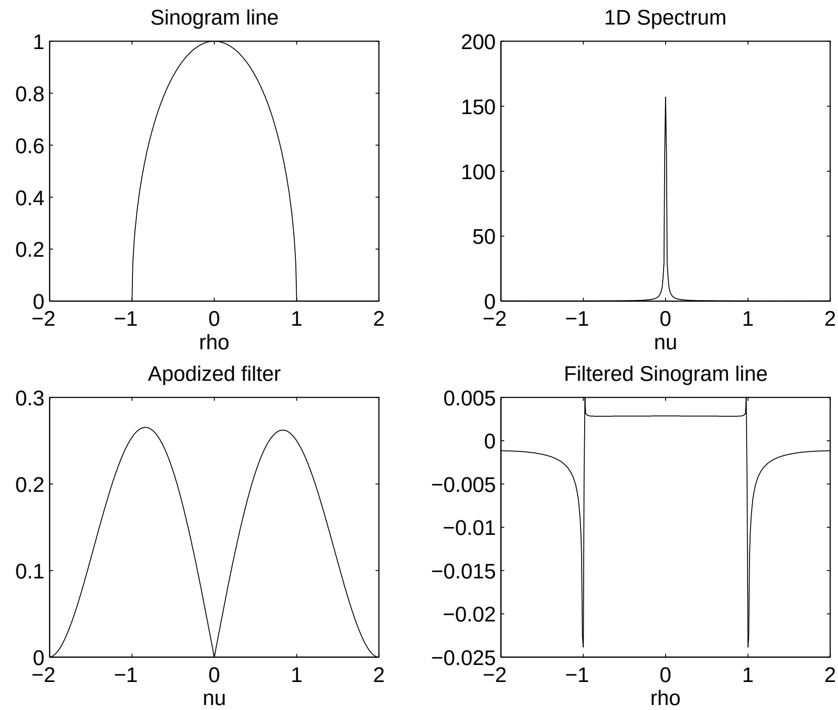

Figure 8.5 Upper left shows the sinogram for a fixed value of θ and varying ρ. Upper right shows the corresponding discrete spectrum, and it can be noted that there is a heavy low frequency dominance. Lower left shows the filter, which here is the ramp multiplied with a Hann window. Lower right shows the filtered sinogram part.

The 1D filtering could also have been done by convolving the sinogram with the proper impulse response, which is infinitely long as indicated in Fig. 8.4, hence windowing is needed. This approach is fast if the impulse response is truncated into a short signal, which will somewhat sacrifice the performance in the frequency domain. This implementation of filtering of the sinogram followed by backprojection is known as convolution backprojection.

8.2 Discrete Implementation of Backprojection

Filtered Backprojection is the easiest inversion scheme to implement. An algorithm based on Filtered Backprojection will have two parts: A filtering part and an integration part. The filtering part has been covered in Subsection 8.1.1 and in this section the implementation of the backprojection part is described.

The backprojection operator was found in Eq. 7.10

$$g(x,y) = \int_0^{\pi} \breve{\bar{g}}(x\cos\theta + y\sin\theta, \theta)\; d\theta \tag{8.11}$$

where $\breve{g}$ is the filtered sinogram in case of Filtered Backprojection and the original sinogram in Filtering after Backprojection.

A commonly used approximation of Eq. 8.11 is

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \breve{\bar{g}}(x_m\cos\theta_t + y_n\sin\theta_t, \theta_t) \tag{8.12}$$

where a one-dimensional interpolation must be used in the ρ-direction. Normally either a nearest neighbour approximation or a linear interpolation is incorporated. Now the discrete indices of the sinogram are used.

Nearest neighbour approximation

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \breve{\bar{g}}\left([r^*], t\right), \text{ where } r^*(m,n;t) = \frac{x_m\cos\theta_t + y_n\sin\theta_t - \rho_{min}}{\Delta\rho} \tag{8.13}$$

Linear Interpolation

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} (1-w)\breve{\bar{g}}(r_l, t) + w\breve{\bar{g}}(r_l+1, t) \tag{8.14}$$

$$\text{where } r^*(m,n;t) = \frac{x_m\cos\theta_t + y_n\sin\theta_t - \rho_{min}}{\Delta\rho} \tag{8.15}$$

$$r_l = \lfloor r^*(m,n;t) \rfloor \text{ and } w = r^* - r_l \tag{8.16}$$

Higher order interpolation is seldom used, due to the increased accuracy will normally not match the increase in computational cost. See also the discussions in Chapters 1 and 2 concerning interpolation. The complexity of the backprojection is given by the image size times the number of angular samples

$$\mathcal{O}_{\text{backprojection}} = \mathcal{O}(M^2 T) \tag{8.17}$$

traditionally the backprojection part has been considered to be far more computationally expensive than the 1D filtering, but the backprojection part can be optimized so the two operations use similar amount of time for relevant sized sinograms. This will naturally also depend on the actual hardware used for reconstruction.

From Eqs. 8.13 and 8.14 it can be seen, that the evaluation of the function $r^*(m,n;t)$ should be optimized, which can reduce the computational cost drastically. Note also the similarity to (ρ,θ) implementation of the Hough transform found in Section 3.2, where a single signal value was distributed along the sinusoid, here all the signal values along the same sinusoid are collected, and it is relevant to the results found in Subsection 3.2.2. Table 3.4, where a full image has to be transformed, showed that time consuming initialization of one- and two-dimensional arrays can tolerated, if the mapping procedure is very fast. Using the optimization technique shown in Case 4 in Subsection 3.2.2 implies that the backprojection operator can implemented effectively as shown in Algorithms 8.2 and 8.3. Here the pseudo code has been split in the initialization part and the true backprojection part in order get a better overview, but also to indicate that the initialization part for a given parameter setup can be computed once and only the backprojection part shown in Algorithm 8.3 reused many times in many images with the same parameter setup should be reconstructed. This is the case in multi slice 2D reconstruction of PET images.

Algorithm 8.2 : Initialization before Backprojection

```
rhooff=rho_min/Delta_rho                    //Compute offset
For t = 0 to T-1                            //For all values of theta
  theta = t*Delta_theta                     //theta is computed
  costheta(t) = cos(theta)                  //cos(theta) is stored
  sintheta(t) = sin(theta)                  //sin(theta) is stored
End
For m = 0 to M-1                            //For all values of x
  xrel = (x_min + m*Delta_x)/Delta_rho      //compute x
  For t = 0 to T-1                          //For all values of theta
    xc(m,t)=xrel*costheta(t)                //Store x times cos theta
    ys(m,t)=xrel*sintheta(t)-rhooff         //Store y times sin theta
  End
End
```

Algorithm 8.3 : Fast Backprojection

```
For m = 0 to M-1                            //For all values of x
  For n = 0 to M-1                          //For all values of y
    sum = 0                                 //Initialize simple variable
    For t = 0 to T-1                        //For all values of theta
      rm = xc(m,t)+ys(n,t)                  //Compute non-integer index
      rl = floor(rm)                        //Find lower integer
      w = (rm-rl)                           //Compute weight
      sum = sum +(1-w)*g_radon(t,rl)+w*g_radon(t,rl+1)
    End                                     //Linear interpolation finished
    g(m,n)=sum*Delta_theta
  End
End
```

Note that in Algorithm 8.3 it is not checked whether the value of rl correspond to pixels in the image or not, i.e, $0 \leq \mathsf{rl} < R-1$. This operation is very time consuming as it is evaluated in the most inner core of the loop. Checking can be avoided, if the initial sinogram is expanded in size in the ρ-direction by padding zeros, in order to fulfill

$$0 < \frac{x_m \cos\theta_t + y_n \sin\theta_t - \rho^*_{min}}{\Delta\rho} < R-1 \quad \forall\ (x_m, y_n, \theta_t) \ \Rightarrow\ R^* > \sqrt{2}(M-1)\frac{\Delta x}{\Delta\rho} + 1 \quad (8.18)$$

where R^* is the number of samples required in the new sinogram, when using the same sampling interval $\Delta\rho$, and also adjusting the value of ρ_{min} to maintain a symmetrical sampling of ρ

$$\rho^*_{min} = -\Delta\rho \frac{R^* - 1}{2} \tag{8.19}$$

The given implementation shown in Algorithms 8.2 and 8.3 will require two matrices of size MT and the computational load is lowered significantly, cf. Table 3.4 for the implementation of the Hough transform.

The discrete implementation of Filtered Backprojection is schematically shown in Fig. 8.6 using first filtering and then backprojection.

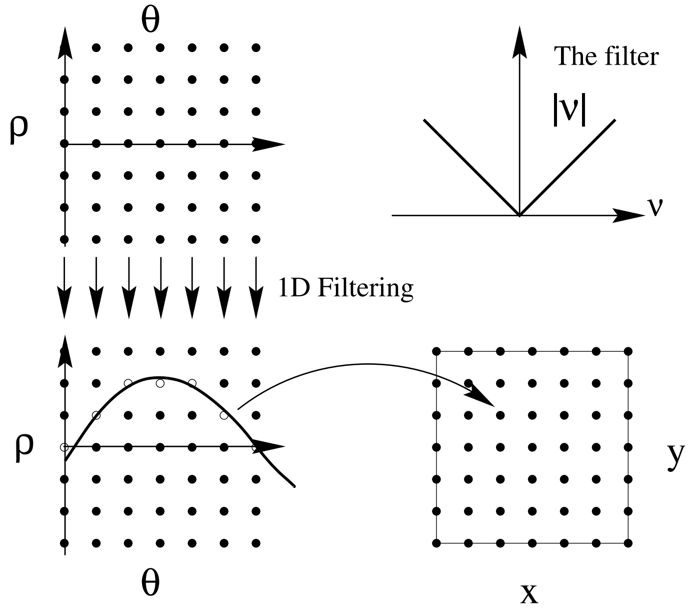

Figure 8.6 The discrete implementation of Filtered Backprojection. At the left the sinogram is filtered and then the filtered sinogram is by backprojection mapped into the reconstructed image.

Now a set of images are shown where a synthetic sinogram, corresponding to a circular disc in the image domain, must be reconstructed using Filtered Backprojection with an increasing number of angular samples. Fig. 8.5 showed the filtering of the sinogram, and here the (rotational symmetrical) disc is placed in the middle of the coordinate system, hence each of one-dimensional the filtered sinogram does not depend on θ_t, cf. Eq. B.29. From 5 angular samples in the sinogram, i.e., $T = 5$, Fig. 8.7 demonstrates the reconstructed image using Filtered Backprojection. The figure clearly shows how the 5 sinogram parts are backprojected into the image, and more angular samples are obviously needed. In Fig. 8.8, only 10 angular samples was used, and the reconstructed shape of the disc is much better recovered, but large artifacts are still very visible outside the disc. Increasing to 20 angular samples, as shown in Fig. 8.9, the artifacts are reduced in amplitude compared to Fig. 8.8, and with 100 angular samples, shown in Fig. 8.10, the disc is nearly recovered perfectly.

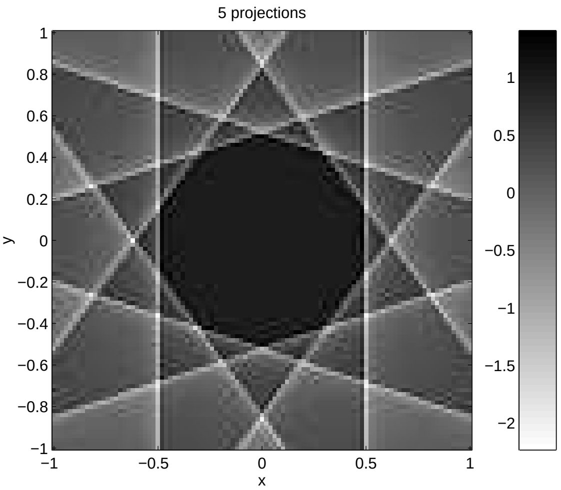

Figure 8.7 Filtered Backprojection from a sinogram with 5 angular samples (T=5).

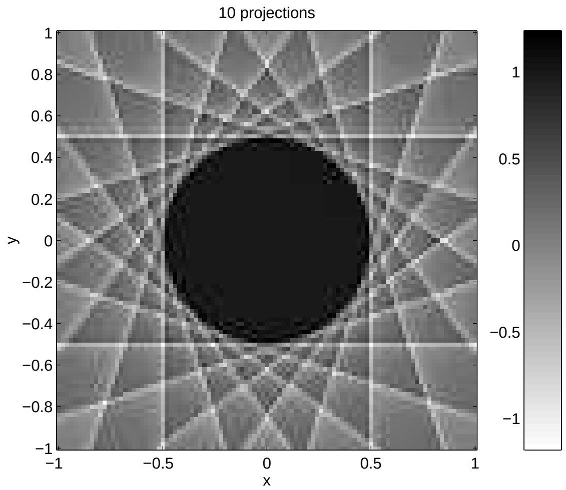

Figure 8.8 Filtered Backprojection from a sinogram with 10 angular samples (T=10).

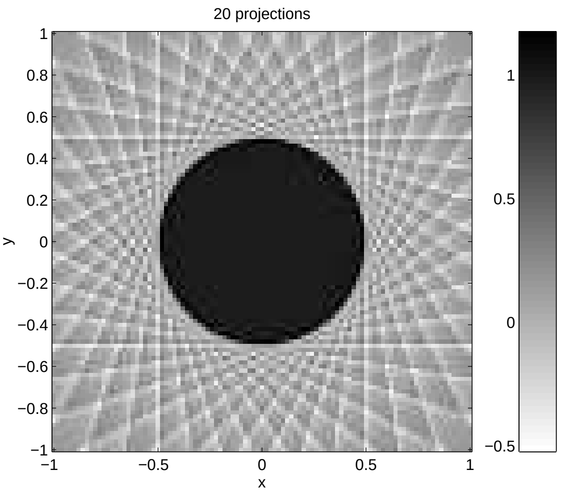

Figure 8.9 Filtered Backprojection from a sinogram with 20 angular samples (T=20).

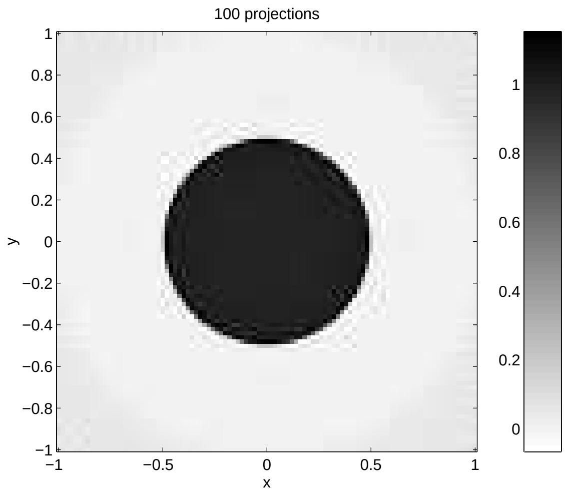

Figure 8.10 Filtered Backprojection from a sinogram with 100 angular samples (T=100).

8.3 Implementation of Filtering after Backprojection

Implementation of Filtering after Backprojection requires a discrete implementation of the backprojection operator, as shown in Section 8.2. After the backprojection the matrix $g(m,n)$ must be high pass filtered cf. Eq. 7.30. The implementation resembles the one shown in Subsection 8.1.1, but is extended to two dimensions.

The spectrum of an image can be obtained in two ways. Either by using one of the multidimensional FFT algorithms, e.g., [90], or by using a one dimensional FFT on first all the rows and then all the columns of the image, which is possible because the discrete spectrum can be written

$$G(u,v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m,n)\ e^{-j2\pi(mu/M+nv/N)} \tag{8.20}$$

$$= \sum_{m=0}^{M-1} \left[\sum_{n=0}^{N-1} g(m,n)\ e^{-j2\pi mu/M} \right] e^{-j2\pi nv/N} \tag{8.21}$$

where both image sizes M and N must be powers of two, if using a radix-2 FFT. If this is not true extra pad image samples at the edges. For filtering the additional samples should not be can be padded with zeros. Assume that the reconstructed image should look like a disc. Then the backprojected sinogram (into the image domain) will have large non-zero values away from the disc, due to the convolution shown in Eq. 7.24. This implies that the filtering in the image domain will meet problem with the cyclical behavior of the DFT (or FFT). These edge problem must be solved by backprojecting onto a larger image than necessary (if using radix-2 FFT often to the nearest upper power of 2), and then filter the expanded image, and cropping values of corresponding to the specified image size.

Note that the image is considered periodical and the spectrum will have a complex conjugate symmetry for a real valued signal, as shown in Fig. 8.11.

$$g(m,n) = g(m+M,n) = g(m,n+N) \tag{8.22}$$

$$G(u,v) = G(m+M,n) = G(m,n+N) \tag{8.23}$$

$$g(m,n) = g(m,n)^* \;\Rightarrow\; G(u,v) = G(-u,-v)^* = G(M-u,N-v)^* \tag{8.24}$$

The symmetry of the spectrum can be exploited for faster computation of the spectrum, but the symmetry can not easily be exploited for reducing the memory requirements due to a rather odd storage strategy needed.

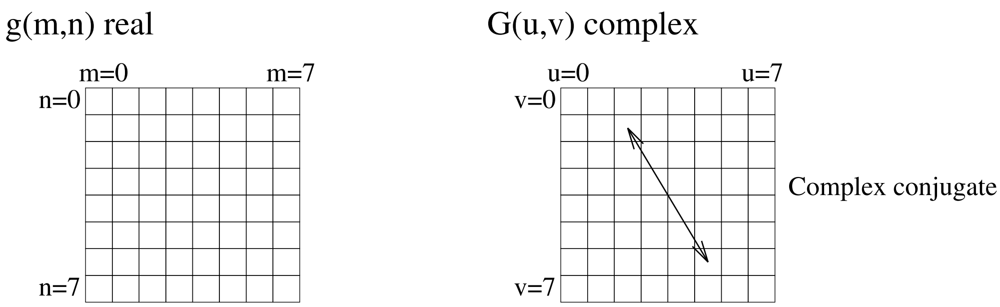

Figure 8.11 A real valued image gives a spectrum with complex conjugate pairs.

The 2D-filtering is very easy if the spectrum is calculated properly. The complex spectrum can easily be multiplied by a sampled version of the filter $\sqrt{k_x^2+k_y^2}$.

$$k_x \to \frac{u}{M\Delta x} \text{ and } k_y \to \frac{v}{N\Delta x} \tag{8.25}$$

$$\sqrt{k_x^2+k_y^2} \to \sqrt{\left(\frac{u}{M\Delta x}\right)^2 + \left(\frac{v}{N\Delta x}\right)^2} \tag{8.26}$$

If addressing the negative frequencies $k_x < 0$, u must be replaced by $u - M$ and if $k_y < 0$, v must be replaced by $v - N$. Symmetry can be used so approximately half of the complex spectrum is multiplied with the filter and the other half is duplicated from the first due to the complex conjugate symmetry. After the multiplication of the filter, the inverse two dimensional DFT (or rather FFT) is used, and the real part of the result is extracted, and the imaginary part should be zero.

The 2D DFT implementation of the high-pass filter should also incorporate multiplication by an apodizing window, in order to reduce the edge effects, due to the periodical behavior of the spectrum. Of the huge amount of windows available, two relevant choices of windows should be mentioned.

Cropped 2D Ramp filter Here the theoretically derived 2D ramp filter is cropped at a certain frequency k_l

$$H(k_x, k_y) = \begin{cases} \sqrt{k_x^2 + k_y^2} & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \tag{8.27}$$

where k_l is set below the upper limit frequency in one of the directions, i.e., $k_l < \frac{1}{2\Delta x}$.

Hanning Window The 2D ramp filter could also be multiplied by a Hanning window

$$H(k_x, k_y) = \begin{cases} \frac{1}{2}\sqrt{k_x^2 + k_y^2}\left(1 + \cos\left(\pi \frac{\sqrt{k_x^2+k_y^2}}{k_l}\right)\right) & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \tag{8.28}$$

where k_l again is set below the upper limit frequency in one of the directions, i.e., $k_l < \frac{1}{2\Delta x}$.

The two windows both use a cutoff frequency k_l, which can be varied, depending on the noise-level. A high noise level might call for a low value for k_l, which implies that the reconstructed image will be somewhat blurred.

Note again that the mean value of the reconstructed image will always be set to zero. This value might be estimated in an area of the reconstructed image where some prior knowledge implies that the value should be, e.g., zero. In brain tomography the relevant area could be outside the brain.

8.4 Implementation of The Fourier Slice Theorem

The basis of the Fourier Slice Theorem is given in Section 7.1. In the implementation, the discrete spectrum of the sinogram is calculated for each of the angular samples like it was shown in Subsection 8.1.1. Note that here the phase is important, hence the shifting shown in Fig. 8.1 must be considered. This spectrum is considered as polar samples, and must be mapped onto a quadratic frequency grid, as shown in Fig. 8.12. This operation calls for two-dimensional interpolation in the frequency domain, an item to be discussed furthermore. Finally, the two-dimensional quadratic spectrum can be inverted using 2D inverse FFT in order to get the reconstructed image.

The complexity of this implementation, where the FFT is used to computing the spectra is given by

$$\mathcal{O}_{\text{Forward 1D FFT of sinogram}} = \mathcal{O}(T\,R \log R) \tag{8.29}$$

$$\mathcal{O}_{\text{Inverse 2D FFT of spectrum}} = \mathcal{O}(M^2 \log M) \;\;\Rightarrow \tag{8.30}$$

$$\mathcal{O}_{\text{Fourier Slice Theorem}} \approx \mathcal{O}(M^2 \log M) \tag{8.31}$$

where the time used to map the polar spectrum onto the quadratic spectrum has not been considered, and it will actually be negligible if using nearest neighbour interpolation. In the last equation it has been assumed that $T \approx R \approx M$. In all three equations the values of R and M should correspond to the expanded sinogram and image (powers of two), if using radix-2 FFT. In conclusion, Eq. 8.31 indicates that the implementation of the Fourier Slice Theorem is of a lower order than Filtered Backprojection and Filtering after Backprojection.

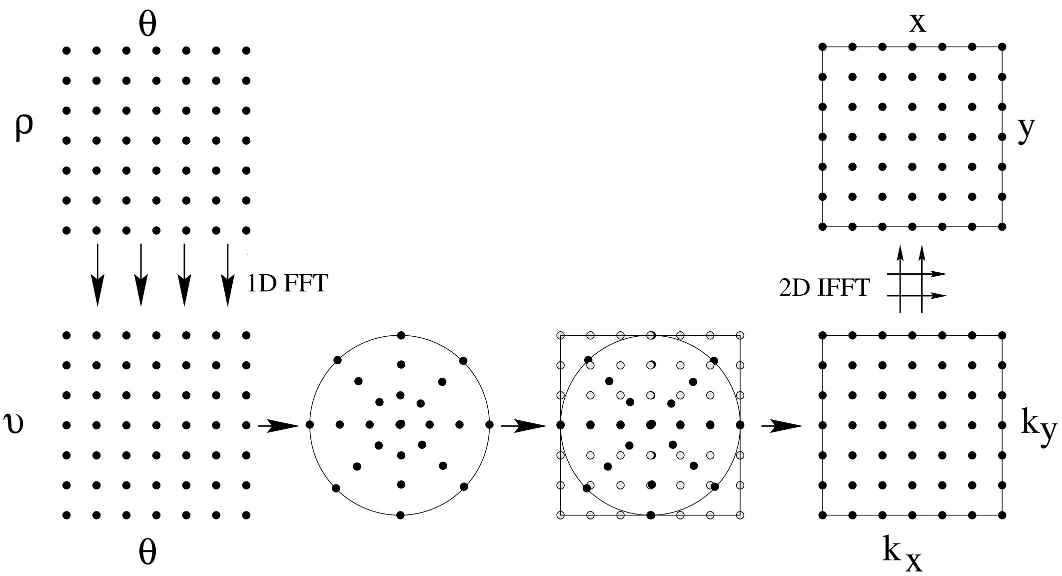

Figure 8.12 Following from upper left. At first the discrete spectrum is computed. The spectrum is then considered to be polar, and mapped onto a quadratic grid in the frequency domain using two-dimensional interpolation. Finally the 2D spectrum is inverted into the reconstructed image using 2D inverse FFT.

Next some of the problems with this implementation are discussed. If omitting the important shifting problems, illustrated in Fig. 8.1, the polar and the quadratic spectrum can match in three different ways as shown in Fig. 8.13. The boundary corresponds to the maximum frequencies (half of the sampling frequencies). Note that a square and centered reconstructed image is still assumed $M = N$, $x_{min} = y_{min}$ and $\Delta x = \Delta y$.

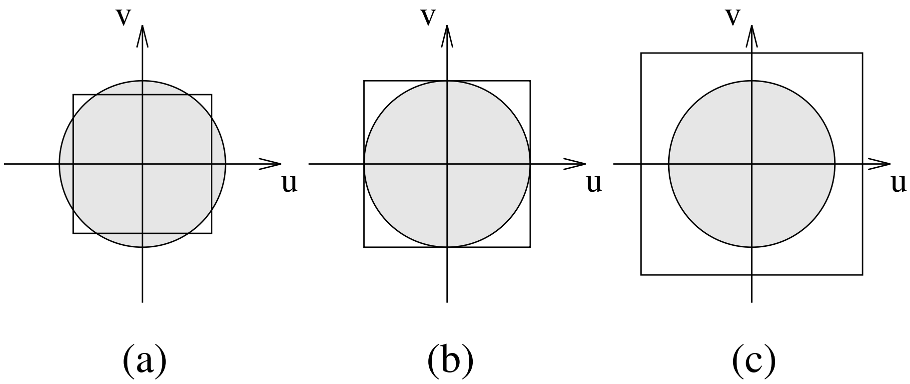

Figure 8.13 Three ways that the polar and the quadratic spectrum can match

In the first case (a) the quadratic spectrum is too small. Some parts of the polar spectrum is not mapped onto the quadratic spectrum. This is a very bad situation, and the result is unreliable when

$$\max |k_x| = \frac{1}{2\Delta x} < \nu_{max} = \frac{1}{2\Delta\rho} \;\Leftrightarrow\; \Delta x > \Delta\rho \tag{8.32}$$

In the second case (b) all of the polar spectrum is mapped onto the quadratic spectrum. This will happen when $\Delta x = \Delta\rho$. In the final case (c), where the polar spectrum is fully covered by the quadratic spectrum. This means that the output image in principle uses all of the spectrum, but the reconstructed image will appear as it was low pass filtered, because the polar spectrum must be assumed to be equal to zero for frequencies higher than half of the polar sampling frequency.

Concerning the 2D interpolation, nearest neighbour interpolation is very fast, but the cost is that artifacts must be expected in the reconstructed image. Another common choice is instead to use the slower but more stable bilinear interpolation as shown in the following equations. Fig. 8.14 illustrates, that the value of each sample in the quadratic grid is a weighted sum of the four nearest neighbours in the polar grid. First of all the frequencies in the quadratic grid are expressed in polar coordinates.

$$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \tilde{\nu} \begin{pmatrix} \cos\tilde{\theta} \\ \sin\tilde{\theta} \end{pmatrix} \tag{8.33}$$

Then the four nearest neighbours are found.

$$\text{Find the integer } t = \left\lfloor \frac{\tilde{\theta}}{\Delta\theta} \right\rfloor \Rightarrow \theta_t \leq \tilde{\theta} < \theta_{t+1} \text{ and set } w_\theta = \frac{\tilde{\theta} - \theta_t}{\Delta\theta} \tag{8.34}$$

$$\text{Find the integer } r = \left\lfloor \frac{\tilde{\nu}}{\Delta\nu} \right\rfloor \Rightarrow \nu_r \leq \tilde{\nu} < \nu_{r+1} \text{ and set } w_\nu = \frac{\tilde{\nu} - \nu_r}{\Delta\nu} = \Delta\rho(\tilde{\nu} - \nu_r) \tag{8.35}$$

where the last formula only is valid for positive frequencies. In case of negative frequencies, the periodical behavior of the spectrum must be considered and proper shifting must be used.

Finally, a bilinear interpolation in the polar coordinates of the four values are used

$$\begin{aligned} G(k_x, k_y) &= (1 - w_\theta)((1 - w_\nu)G(\nu_r, \theta_t) + w_\nu G(\nu_{r+1}, \theta_t)) \\ &\quad + w_\theta((1 - w_\nu)G(\nu_r, \theta_{t+1}) + w_\nu G(\nu_{r+1}, \theta_{t+1})) \end{aligned} \tag{8.36}$$

The sampling of the output image must furthermore be sufficiently dense to adjust to the level of information in the polar spectrum, but due to the distribution of polar samples, with an increased density towards $(0,0)$, a general problem is that the quadratic spectrum does not exploit the high number of samples near $(0,0)$ leading to aliasing artifacts. On the other hand for high polar frequencies, the density is low, so the quadratic spectrum samples the polar spectrum faster than necessary. Note, the angular distance between samples in the polar grid is $\Delta\theta$, and in the radial parameter ν the distance between samples is $\Delta\nu = \frac{1}{R\Delta\rho}$, cf. Eq. 8.2. The distance in each of the coordinates in the rectangular grid is $\Delta k_x = \Delta k_y = \frac{1}{M\Delta x}$. If choosing $\Delta\rho = \Delta x$, cf. Eq. 8.32, one criterion is that the sampling interval in the radial parameter ν must match the one in the rectangular grid, which is a reasonable tradeoff, i.e.,

$$\Delta k_x = \Delta k_y = \frac{1}{M\Delta x} = \Delta\nu = \frac{1}{R\Delta\rho} \Rightarrow M = R \tag{8.37}$$

where both M and R should correspond to the expanded values, i.e., being a power of two if using a radix-2 FFT.

The presented reconstruction algorithm will introduce more noise than accumulated in the backprojection algorithms. Especially, ringing problems are common. The problem can be reduced by use of higher-order interpolation and/or use of, e.g. a non-linear grid in the Radon domain. Note that higher-order filters can provide better numerical results, but the computational load might increase to an unacceptable level. Another method [91, 58] illustrated in Fig. 8.15 is to distribute each of the polar samples onto the rectangular map using proper weights, which implies that all of the polar samples will always be represented in the quadratic grid, but the cost of the uneven density of samples is that an additional filtering is required.

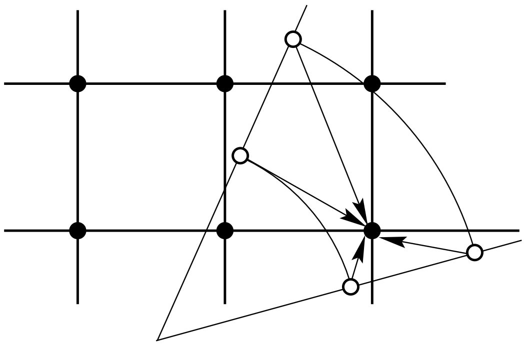

Figure 8.14 Strategy 1: A weighted sum of the four closest polar samples is used to estimate the spectrum on the quadratic grid.

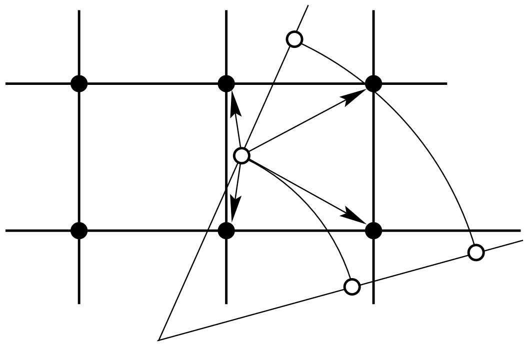

Figure 8.15 Strategy 2: Any of the samples on the polar grid are distributed with certain weights to the quadratic grid.

The 2D interpolation in the frequency domain described in this section is in general considered the major problem in implementations of the Fourier Slice Theorem, and the method has apparently found limited success in clinical use.

8.4.1 Non-linear sampling of the Radon domain

The problems by using two-dimensional interpolation in the frequency domain can be reduced by using a nonlinear grid in the Radon domain. The basic idea is to use one dimensional interpolation in the frequency domain and (perhaps) one dimensional interpolation in the Radon domain. This idea has been used in the Linogram method [92] and the method has attracted attention, because of the low complexity $M^2 \log M$, and the authors claim to provide an image quality as good as by use of Filtered Backprojection, in case of a noise-free sinogram. The Linogram method is also covered very thoroughly in [72], hence only the basic elements of the method is covered here. It can be noted that other reconstruction algorithms similar to the linogram method can be found in the literature, e.g., [93, 59].

The last half of the Fourier Slice Theorem, i.e., the operator $\mathcal{F}_{2r}^{+}$ is easily implemented using quadratic sampling of the frequency domain. Thus the spectrum $G(k_x, k_y)$ is approximated by the two dimensional DFT on a quadratic grid denoted $G(m, n)$, where the (positive) frequencies are given by

$$k_x = \frac{m}{M\Delta x} \tag{8.38}$$

$$k_y = \frac{n}{M\Delta x} \tag{8.39}$$

The polar spectrum $H(\nu, \theta)$ is as previous mapped onto the quadratic grid using Eq. 8.33, and linear sampling of θ is still applied, and the sampling of $\nu = r/(R\Delta\rho)$ can be chosen to vary with θ by allowing $\Delta\rho$ to be a function of θ. The frequency plane is divided into two sectors where the available values of k_x and k_y are used

- Sector 1

$$\sin\theta < \frac{1}{\sqrt{2}} \;:\; \begin{cases} k_x = \nu\cos\theta \Rightarrow \\ \nu = \frac{k_x}{\cos\theta} = \frac{m}{M\Delta x\cos\theta} \Rightarrow \\ \Delta\nu = \frac{1}{M\Delta x|\cos\theta|} \end{cases} \tag{8.40}$$

- Sector 2

$$\sin\theta \geq \frac{1}{\sqrt{2}} \ : \ \begin{cases} k_y = \nu \sin\theta \Rightarrow \\ \nu = \frac{k_y}{\sin\theta} \Rightarrow \\ \Delta\nu = \frac{1}{M\Delta x|\sin\theta|} \end{cases} \tag{8.41}$$

With this choice only one-dimensional interpolation of the spectrum is needed. The non-linear sampling of ν results in a non-linear sampling of ρ, i.e.

$$\sin\theta < \frac{1}{\sqrt{2}} \Rightarrow \Delta\rho = \frac{M}{R}\Delta x|\cos\theta| \tag{8.42}$$

$$\sin\theta \geq \frac{1}{\sqrt{2}} \Rightarrow \Delta\rho = \frac{M}{R}\Delta x|\sin\theta| \tag{8.43}$$

In Fig. 8.16 is shown the non-linear sampling of the frequency domain, and Fig. 8.17 shows the non-linear sampling of the Radon domain.

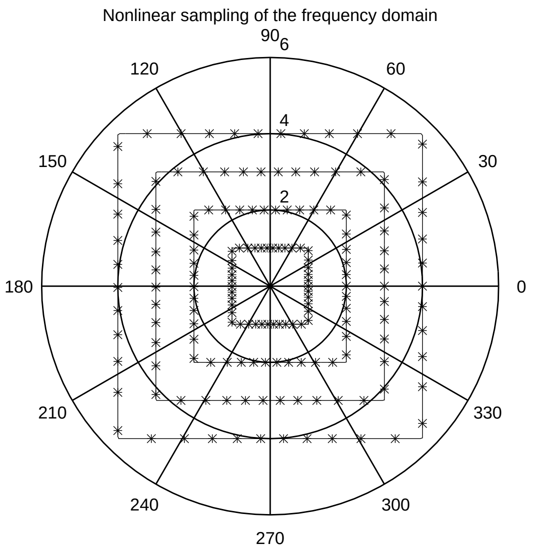

Figure 8.16 Non-linear sampling of the frequency domain.

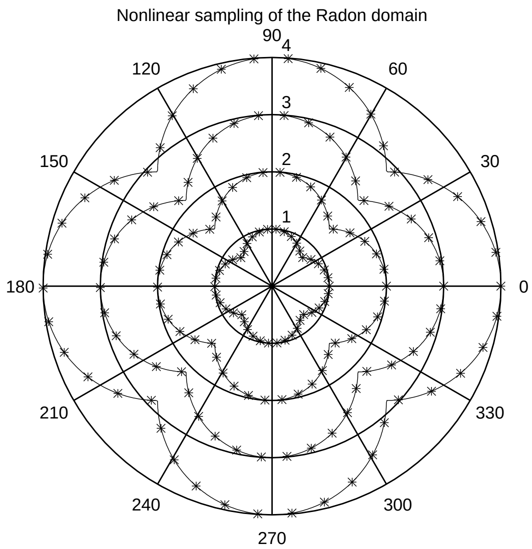

Figure 8.17 Non-linear sampling of the Radon domain. Here (ρ, θ) is used as polar coordinates.

If this technique is used, only interpolation with respect to k_x when $\frac{\pi}{4} < \theta < \frac{3\pi}{4}$ and only with respect to k_y is needed in the case $0 \leq \theta < \frac{\pi}{4} \vee \frac{3\pi}{4} \leq \theta < \pi$. The additional interpolation in the Radon domain is normally not that critical because the Radon transform has low pass characteristics, i.e., low-order interpolation is normally sufficient.

Another related strategy is to use the Chirp-z transform [94, 95] to transform directly from the sinogram (without interpolation) to the spectrum shaped as shown in Fig. 8.16. In [59] this method has been implemented and evaluated, but the conclusion was that even given that all of the involved step has complexity $\mathcal{O}(M^2 \log M)$, the method is not that fast for small sized images, e.g., 100∗100 and certainly more difficult to implement compared to, e.g., the Fourier Slice Theorem.

8.5 Examples Using Direct Reconstruction Algorithms

A software packages has been developed [59], where several direct reconstruction algorithms have been implemented, and lately additional features such as numerical forward projection (discrete Radon transform) has been implemented. The package called "iradon" is available from [10], and the usage of the programs are shown in Section C.2. All of the reconstruction examples in this chapter has been generated using "iradon". Synthetic images and corresponding sinograms are generated using another developed package named "RadonAna", described in Sections C.1 and B.3.

8.5.1 Reconstruction using Different Methods

Here reconstruction of a head phantom is examined. In Fig. 8.18 is shown the noise-free sinogram corresponding to the image shown in Fig. 8.19. Note that all of the following figures are color-scaled individually corresponding to the minimum and maximum value. First, Filtered Backprojection has been used to reconstruct the image as shown in Fig. 8.20. It can be seen that the mean value is displaced, and a ring is visible outside the head phantom with a radius corresponding to the extension of the sinogram in the ρ-direction. This is a good example showing that zeros in general must be padded to the sinogram in that direction in order to reduce the cyclical behavior of the FFT based filtering. If using Filtering after Backprojection, as shown in Fig. 8.21, the ring effect has disappeared, but otherwise the result appears to be just as good with respect to edge sharpness.

Finally, a nearest neighbour implementation of the Fourier Slice theorem has been used as shown in Fig. 8.22. The reconstructed image appears to be comparable to the two backprojection methods, but more "texture" can be found in the area outside of the head (and inside too).

The sampling parameters of the sinogram are $\Delta\rho = 0.01$, $R = 201$, $T = 200$, and for the image, $M = 201$ and $\Delta x = 0.01$ has been used.

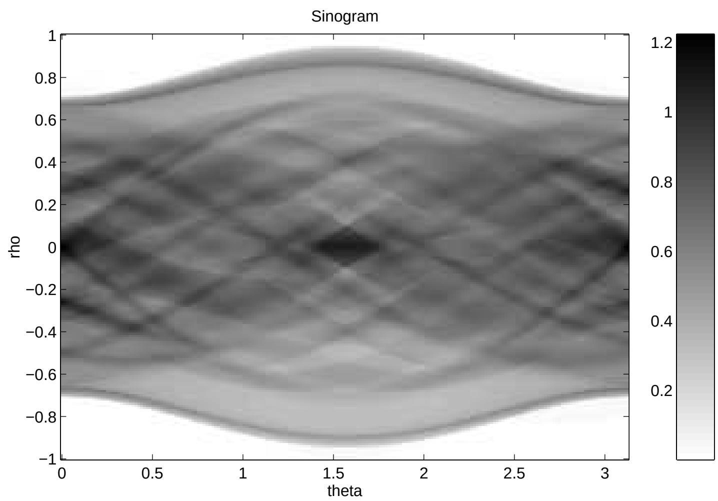

Figure 8.18 Sinogram of the phantom.

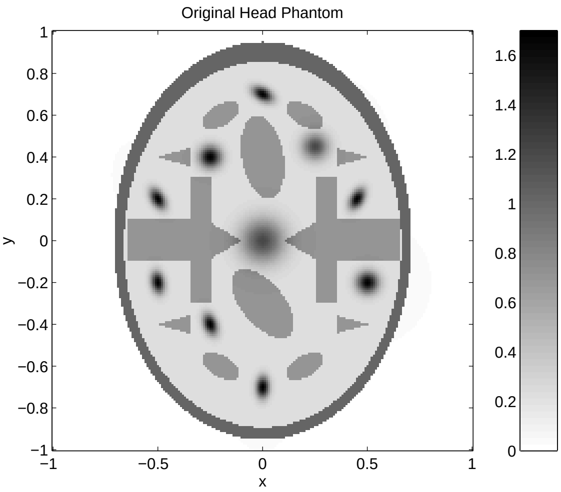

Figure 8.19 The original head phantom.

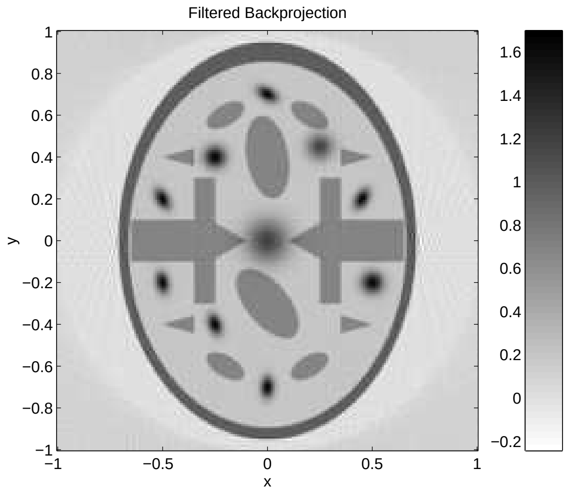

Figure 8.20 The reconstructed head phantom using Filtered Backprojection.

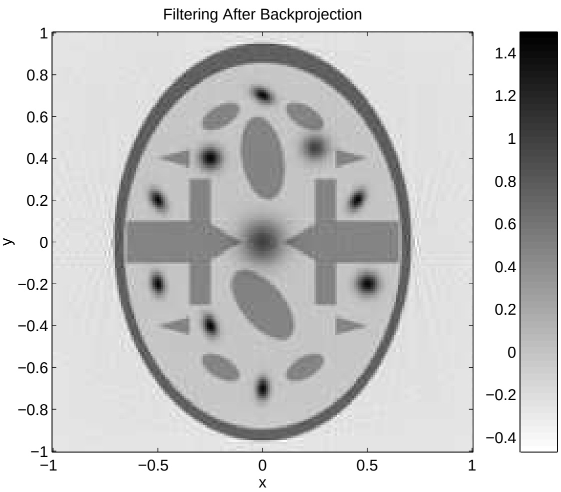

Figure 8.21 The reconstructed head phantom using Filtering After Backprojection.

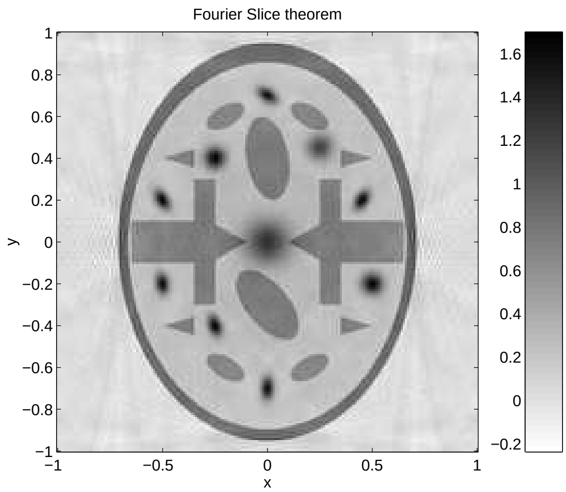

Figure 8.22 The reconstructed head phantom using Fourier Slice theorem.

8.5.2 Sinogram with very few Samples in the Angular Direction

A sinogram corresponding to a spiral of discs was created, and the number of samples in the angular direction, $T = 30$, is very small compared to $R = 151$ and $\Delta\rho = 0.2$. The sampling parameters of the reconstructed image is chosen to $M = 151$ and $\Delta x = 0.2$. First, Fig. 8.23 shows the reconstructed image using a nearest neighbour implementation of the Fourier Slice theorem. The few samples in the θ-direction can be seen as an angular blurring especially farthest away from the center of the image. Using Filtered backprojection (with a ramp filter) gives approximately the same visual impression, but without the angular blurring. Approximately the same number of discs can be identified in the image.

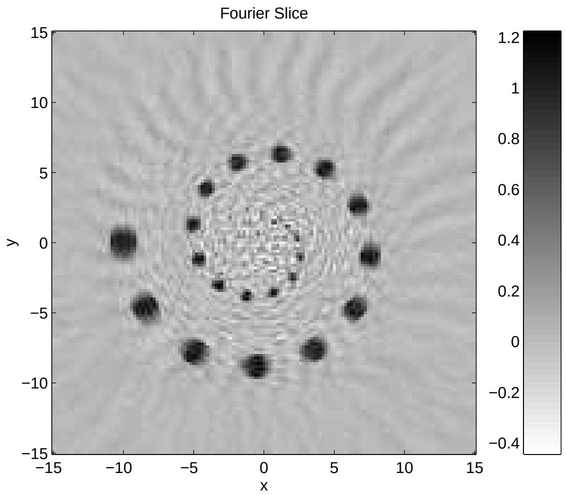

Figure 8.23 The reconstructed spiral using Fourier Slice Theorem.

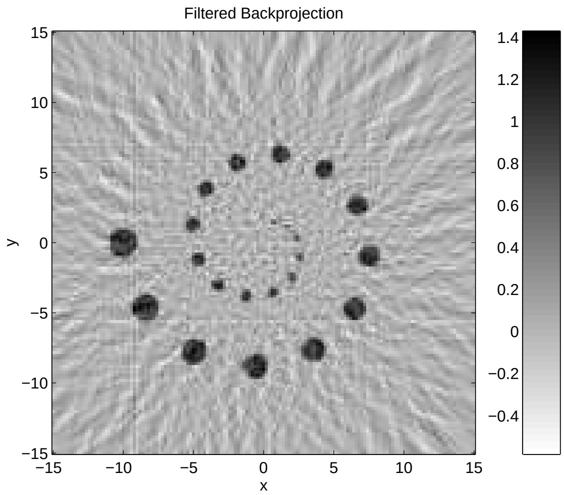

Figure 8.24 The reconstructed spiral using Filtered Backprojection.

8.5.3 Reconstruction with Varying Image Size

The next example addresses quantification of the reconstruction quality and artifacts when using a Fourier Slice algorithm, Filtered Backprojection, and Filtering after Backprojection. From the sinogram shown in Fig. 8.18, images have been reconstructed with increasing image size, going from $M = 51$ to $M = 401$ in steps of 50 samples (on each dimension of the reconstructed image), where the sampling distance has been changed in order to keep the image in focus, as shown in Fig. 8.19.

A modified L2-measure of misfit given in Eq. 8.44, has been used to quantify the error. The modification is that the DC-level does not alter the measure, due to the problems mentioned in Subsection 7.4.1.

$$\mathrm{L2} = \sqrt{\frac{\sum_{m,n}(g_{m,n} - g_{m,n}^{ref} - \bar{g} + \bar{g}^{ref})^2}{\sum_{m,n}(g_{m,n}^{ref} - \bar{g}^{ref})^2}} \tag{8.44}$$

where $g_{m,n}^{ref}$ is the reference image (the original) and the bars indicate the average over all samples.

Fig. 8.25 shows the misfit in this case as a function of the image size M. It can be seen that here the errors limited, and the Fourier Slice method has the worst error-measure. These observations are naturally also depend on the image contents. For Filtered Backprojection Fig. 8.26 shows that the error has several reasons. It is obvious that the steep edges in the image are not reconstructed perfectly, due to the use of simple linear filters. Furthermore, lines are visible in the figure, which are due to aliasing problems. The sinogram should have been sampled more densely. Finally the ring also found in Fig. 8.20 will also add to the total error.

For a Pentium 120 MHz (Linux system) the time needed to reconstruct the images are shown in Fig. 8.27. It is clear that the radix-2 FFT used for this implementation implies that several steps can be seen in the Filtering after Backprojection and the Fourier Slice implementation. The reason that Filtering after Backprojection is much slower compared to Filtered Backprojection is that backprojection must be done into a larger number of samples (power of two) in order to reduce edge effects in the subsequent filtering. It can be seen that Filtering after Backprojection from image size 151 to 251 gets slightly faster. In this range the use of radix-2 FFT implies that a 256∗256 image is generated in all three cases (151,201, and 251) when backprojecting, and due to the implementation, the subsequent cropping becomes slightly faster here.

This example shows that Filtered Backprojection can be implemented efficiently on a PC and provide fast reconstruction, approximately as fast as the Fourier Slice theorem for the images size shown in this example.

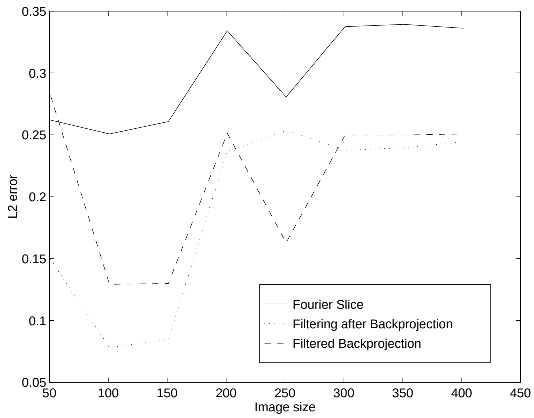

Figure 8.25 L2 error as a function of the reconstructed image size M for Filtered Backprojection, Filtering after Backprojection, and a Fourier Slice implementation.

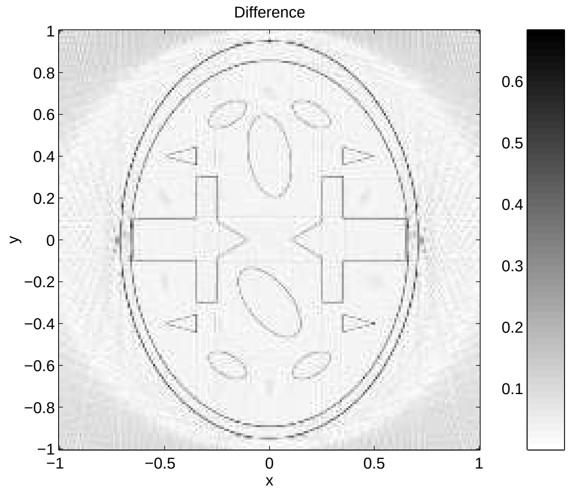

Figure 8.26 The absolute error between the original image and the reconstructed image using Filtered Backprojection.

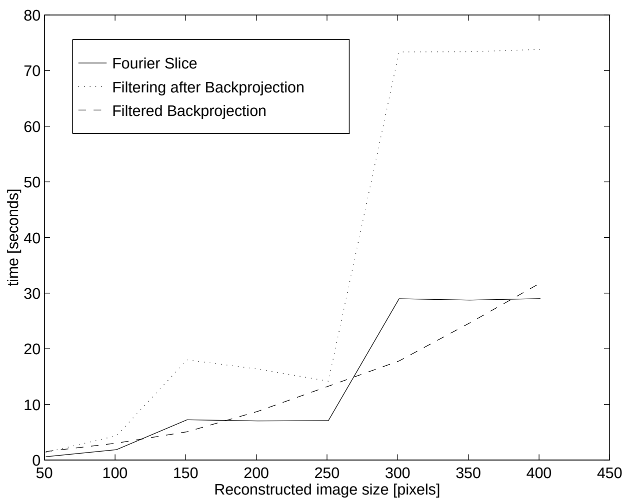

Figure 8.27 Time usage for reconstructing the sinogram in seconds as a function of the reconstructed image size M for Filtered Backprojection, Filtering after Backprojection, and a Fourier Slice Theorem implementation on a Pentium 120 MHz.

8.5.4 Reconstruction into a Oversampled Image

In order to illustrate the problems with the Fourier Slice theorem if $\Delta x > \Delta\rho$, (cf. Eq. 8.32) a sinogram with $R = 251$, $T = 200$, and $\Delta\rho = 0.0025$ has been created. The sinogram corresponding to a square in the image domain is shown in Fig. 8.28. Then two reconstruction methods have been used, namely the Fourier Slice Theorem in Fig. 8.29, and Filtering after Backprojection in Fig. 8.30. It can be seen that the filtering after backprojection has no problems, but the DC component is of course wrong. Using the Fourier Slice theorem severe artifacts can be seen, due to aliasing of the spectrum.

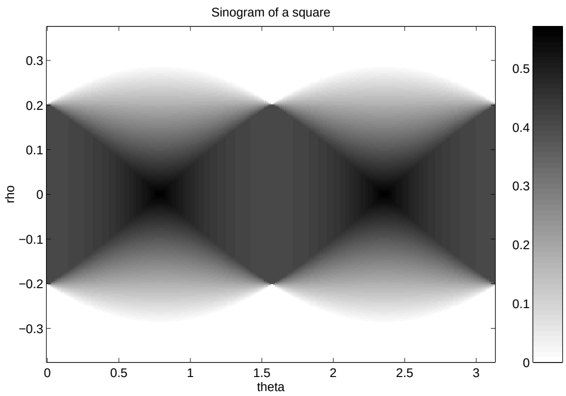

Figure 8.28 Sinogram of a square with $R = 251$, $T = 200$, and $\Delta\rho = 0.0025$.

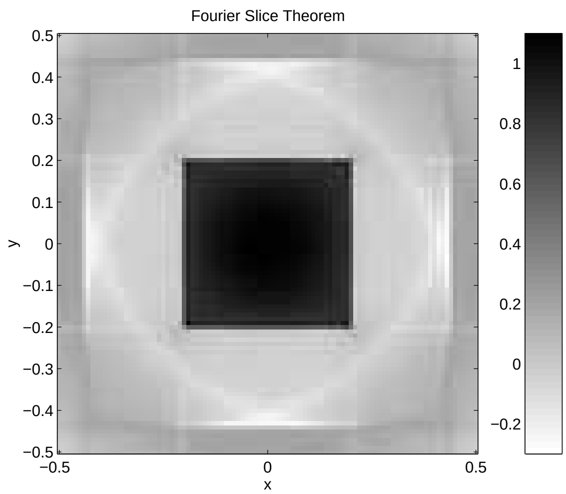

Figure 8.29 Reconstructed image with $M = 101$ and $\Delta x = 0.01$ using Fourier Slice theorem.

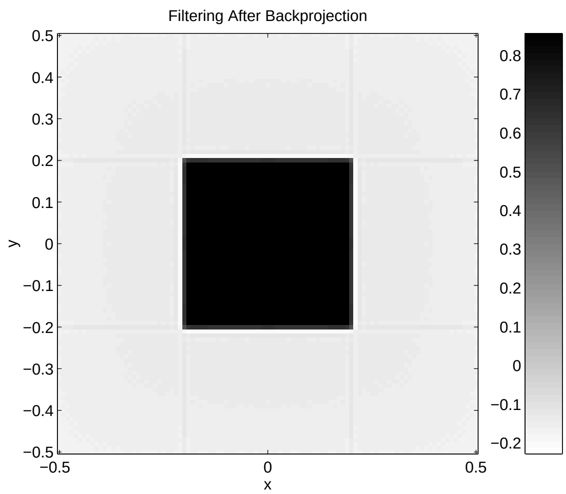

Figure 8.30 Reconstructed image with $M = 101$ and $\Delta x = 0.01$ using Filtering after Backprojection.

8.5.5 Noise in the Sinogram

One of the problems found in PET reconstruction, is that the sinograms are noisy. Assume that the noise is mainly due to finite measurement time of the emission sinogram, then each of the sinogram bins will be approximately Poisson distributed $P(\lambda_{r,t})$ with a parameter $\lambda_{r,t} = \mathcal{E}(\rho_r, \theta_t)T_e$, where $\mathcal{E}$ denotes the mean and unknown emission activity in a bin of the sinogram, and T_e is the measurement time for the emission sinogram. This situation is often assumed in the literature, but it will be shown in Chapter 11 that especially the transmission scan might add much noise.

Here the sinogram is quantified by the total number of counts in the emission sinogram, i.e., the sum over all bins in the sinogram. For four values of the total number of counts, a sinogram corresponding to a disc is generated using a Poisson generator, [90]. The sinogram is then reconstructed using Filtered Backprojection.

First Fig. 8.31 shows the $101 * 100$ samples sinogram (corresponding to a disc in the image domain) with 10^5 counts, and Fig. 8.32 shows the corresponding reconstructed $151 * 151$ samples image. Likewise Fig. 8.33 has been reconstructed from 10^6 counts in the sinogram, and finally Fig. 8.34 from 10^7 counts. The sinogram size (10100 bins) corresponds to an average number of counts per sample in the range 10-1000. All three reconstructions have been made with Filtered Backprojection using a ramp filter.

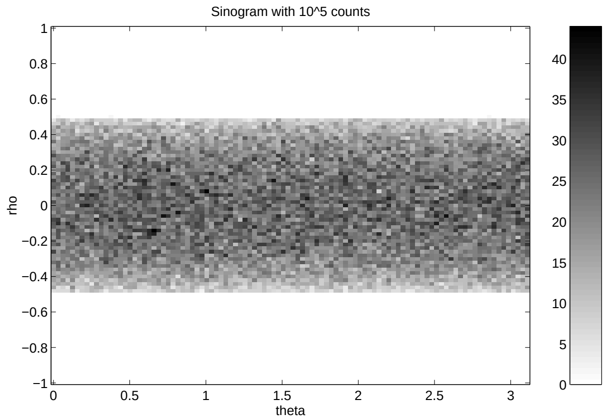

Figure 8.31 Sinogram with 10^5 counts.

From the reconstructed images it is clear that few counts implies a high noise level in the reconstructed images. As mentioned in Chapter 8 the ordinary cure is to multiply the ramp filter with an apodizing window. As an example a Hanning window will be multiplied to the ramp filter with a varying relative cutoff frequency ν_u, cf. Eq. 8.8. The L2 measure has been computed as a function of ν_l/ν_u, i.e., the cutoff frequency relative to half of the sampling frequency, and the result is shown in Fig. 8.35. In this case where the object of interest is very large a very low value of the optimal ν_l is found. Using $\nu = 0.2\nu_u$ the "optimal" reconstructed image is shown in Fig. 8.36. It can be seen that the high-frequency contents of the image has been removed, but the edges have also been blurred.

In general the choice of ν_u will always imply a tradeoff between resolution and noise suppression. To illustrate the tradeoff, the head phantom, previously shown in Fig. 8.19, has been reconstructed using a ramp filter apodized with a Hanning window with $\nu_u = 0.2\nu_u$. Here it can be seen that the value of ν_u is definitely too small, and most of the finer details have been removed by the low pass filtering.

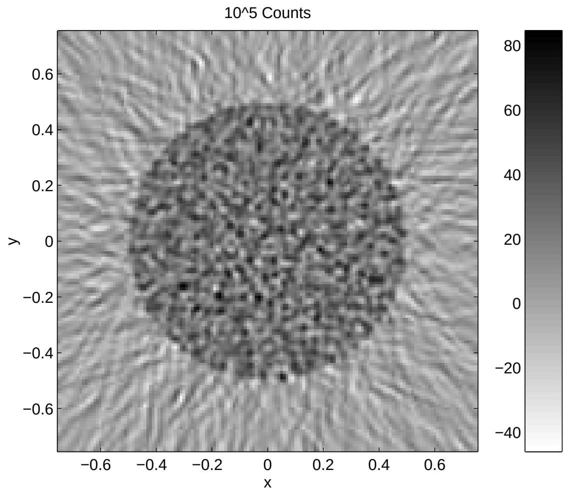

Figure 8.32 Reconstructed image from 10^5 counts in the sinogram.

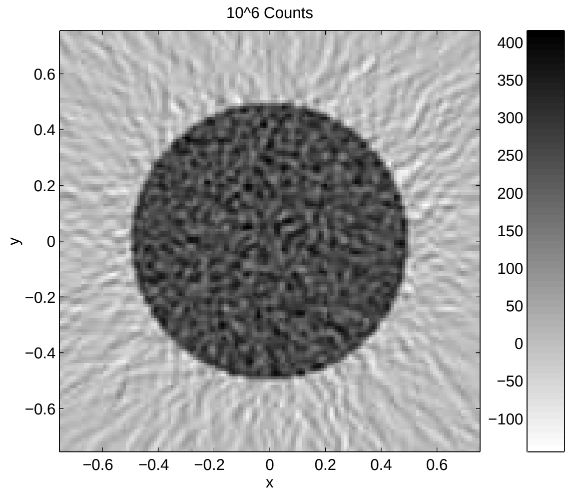

Figure 8.33 Reconstructed image from 10^6 counts in the sinogram.

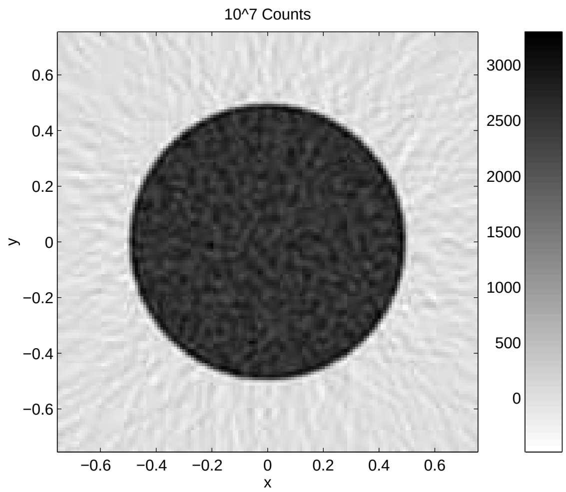

Figure 8.34 Reconstructed image from 10^7 counts in the sinogram.

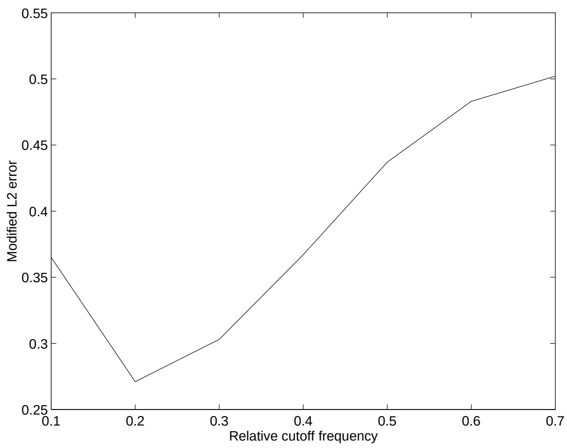

Figure 8.35 Modified L2 measure as a function of the relative cutoff frequency of the Hanning window.

8.6 Summary

In this chapter the implementation of several direct inversion schemes and hints for improving the speed have been presented. Filtered Backprojection is normally precise but somewhat slow, with complexity $\mathcal{O}(M^3)$, where M is the number of samples in one direction of the resulting image. Filtered Backprojection is the algorithm used in most scanners today. The Fourier Slice Theorem gives a fast algorithm, $\mathcal{O}(M^2 \log M)$, but does not have the same numerical stability, due to the two-dimensional interpolation. Another algorithm is a hybrid between the two inversion schemes: The Linogram method, see [72, 92]. This inversion method gives better numerical stability than implementations of the Fourier Slice Theorem and has complexity $\mathcal{O}(M^2 \log M)$. The algorithm uses the non-linear sampling of the frequency domain described above.

Finally, a set of examples using some of the direct reconstruction algorithms have been shown, demonstrating some of the possibilities and limitations with the direct reconstruction algorithms.

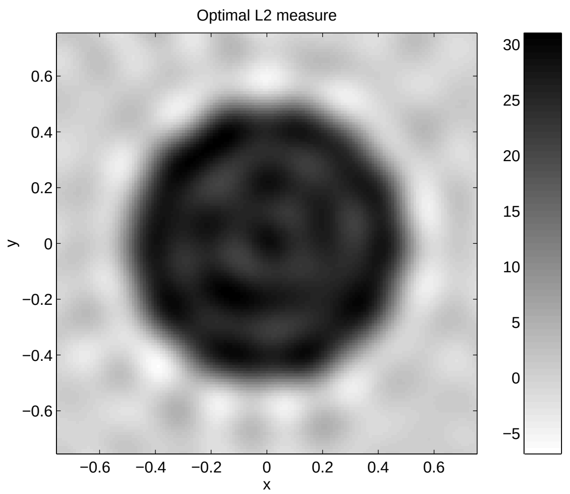

Figure 8.36 The reconstructed disc using $\nu_u = 0.2\nu_u$ from the sinogram with 10^5 counts.

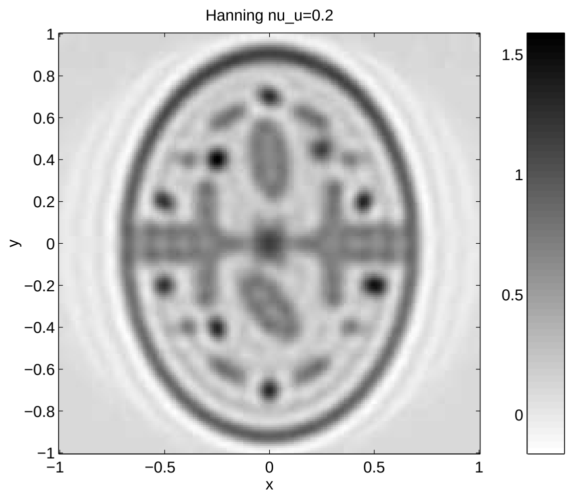

Figure 8.37 Reconstructed head phantom as shown in Fig. 8.19 but using a Hanning window with $\nu_u = 0.2\nu_u$. In this case the reconstructed images becomes very blurred.

Chapter 9

Reconstruction Algorithms Based on Linear Algebra

Inversion of the Radon transform need not be based on the direct inverse formulas as shown in Section 7.4. A different class of reconstruction schemes is based on linear algebra [96, 14, 97]. The chapter presents the linear algebra based reconstruction theory, and the modelling of the reconstruction problem. Several of the well known iterative reconstruction techniques are demonstrated, and a fast implementation of the iterative reconstruction algorithms is presented along with a set of examples.

9.1 From the Radon Transform to Linear Algebra based Reconstruction

The Radon transform is a linear transform, with respect to the function $g(x, y)$, and so is the discrete versions of the Radon transform, hence instead of considering the integral version of the Radon transform operators a matrix representation can used

$$\begin{array}{ccc} \breve{g}(\rho,\theta) & = \mathcal{R} & g(x,y) \\ \downarrow & \downarrow & \downarrow \\ \boldsymbol{b} & = \boldsymbol{A} & \boldsymbol{x} \end{array} \tag{9.1}$$

Assume a discrete set of values for the Radon transform $\breve{g}(\rho_r, \theta_t) = \breve{g}_d(r, t)$, and a given sampling of wanted image $g(x_m, y_n) = g_d(m, n)$ used for the reconstruction. If the matrix $\breve{g}(r, t)$ is rearranged into a vector, e.g.,

$$b_i = b_{rT+t} = \breve{g}(r, t) \tag{9.2}$$

and the same technique is applied to the image, e.g.,

$$x_j = x_{nM+m} = g(m, n) \tag{9.3}$$

then using a series expansion shown in Eq. 9.5 the Radon transform can be written in the form shown in equation 9.11. The vector dimensions are $I = RT$ and $J = MN$, thus the transformation matrix $\boldsymbol{A}$ have $RTMN$ elements, and will normally be very large.

$$g(x, y) = \sum_m \sum_n g(m, n)\phi(x - x_m, y - y_n) \ \Rightarrow \tag{9.4}$$

$$\breve{g}(r, t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(\sum_m \sum_n g(m, n)\ \phi(x - x_m, y - y_n)\right) \delta(\rho_r - x\cos\theta_t - y\sin\theta_t)\ dx\ dy \tag{9.5}$$

$$= \sum_m \sum_n g(m, n) \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \phi(x - x_m, y - y_n)\ \delta(\rho_r - x\cos\theta_t - y\sin\theta_t)\ dx\ dy \tag{9.6}$$

hence the matrix elements of the system matrix can be calculated as

$$\begin{aligned} a_{i,j} &= a_{rT+t,nM+m} && (9.7)\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \phi(x-x_m, y-y_n)\ \delta(\rho_r - x\cos\theta_t - y\sin\theta_t)\ dx\ dy && (9.8)\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \phi(x,y)\ \delta((\rho_r - x_m\cos\theta_t - y_n\sin\theta_t) - x\cos\theta_t - y\sin\theta_t)\ dx\ dy && (9.9)\\ &= \check{\phi}(\rho_r - x_m\cos\theta_t - y_n\sin\theta_t, \theta_t) && (9.10) \end{aligned}$$

where the function $\phi(\cdot)$ is the expansion function in the image domain specifying how the pixel at (x_m, y_n) models the image domain with the continuous positions (x, y). Eq. 9.10 shows that the matrix element is the Radon transform of the expansion function, where the ρ-parameter has been shifted according to the pixel position. Alternatives to the image pixel driven generation of matrix elements can be found in [98].

The methods to be presented all rely on a linear dependence between two vectors; the known I-dimensional vector $\boldsymbol{b}$ $(I = RT)$, containing the sinogram, and the unknown J-dimensional vector $\boldsymbol{x}$ $(J = M^2)$. For tomography the vector $\boldsymbol{b}$ will contain the sinogram values wrapped into a vector using Eq. 9.2, and $\boldsymbol{x}$ is the set of reconstructed pixels in the image formed as a vector using Eq. 9.3.

In the rest of this chapter a linear algebra formalism is used instead of deriving integrals formulations of the inverse Radon transform, which then is approximated in order to implement the algorithms. Now the reconstruction problem will can be written in a matrix vector formulation

$$\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x} \tag{9.11}$$

where $\boldsymbol{A} \in \mathbb{R}^{I\times J}$ is called the system matrix containing the weight factors between each of the image pixels and each of the line orientations from the sinogram, as illustrated in Fig. 9.1.

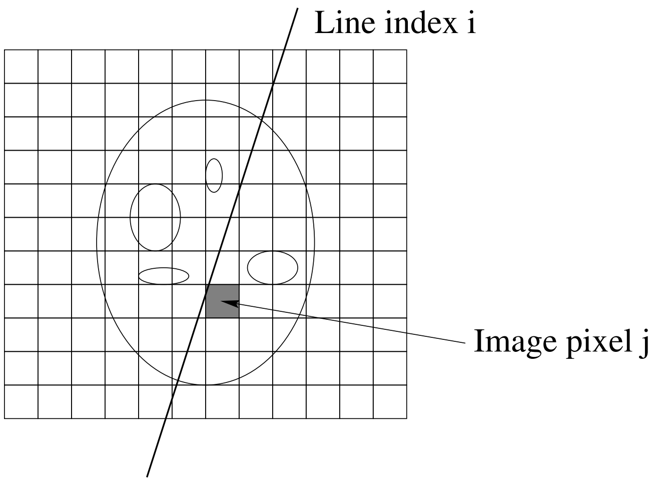

Figure 9.1 The matrix element $a_{i,j}$ can be considered as the weight factor between a certain sinogram value numbered by i and the image pixel j.

Compared to the Radon based direct reconstruction methods shown in Chapter 7, several new possibilities and problems arise

- Linear algebra is very strongly supported in mathematics, and the formalism can be used for both 2D and 3D PET reconstruction.
- An irregular scanner geometry with, e.g., limited line orientation is very bad for the direct methods, e.g., see [99], and with the linear algebra approach, problems with missing data in the sinogram can easily be incorporated into the matrix formalism.
- A finite, i.e., non-zero detector size can be modelled into the system, hence the model need no be based on a ray approximation, and varying detector sensibility can in principle be modelled into the system of equations, hence better modelling of the physical scanner setup is possible.
- The system matrix $\boldsymbol{A}$ is in general not quadratic ($I \neq J$), which limit the number of techniques applicable if not forming the normal equations, which will be shown in Eq. 9.20.
- The system matrix $\boldsymbol{A}$ will be (near) singular, i.e., have very small singular values, i.e., that reconstruction written in the linear algebra formalism is an ill-conditioned problem. Some of the image valued can be under determined and others very over determined due to the uneven coverage of the projections (sinogram lines) in the image domain. This problem must then be controlled with constraints and/or regularization.
- It turns out that $\boldsymbol{A}$ does not has a simple structure, such as a band matrix, hence the inversion of $\boldsymbol{A}$ have to use rather slow methods. The reason that $\boldsymbol{A}$ does not have a simple structure is partly due to the easy sorting schemes shown in Eqs. 9.2 and 9.3. Another reason is that line parameters (ρ, θ) does not have a very simple relation to the image coordinates (x, y).
- The matrix $\boldsymbol{A}$ is normally very large, e.g., $256^2 \times 256^2$ elements, thus calculation of a generalized inverse of $\boldsymbol{A}$ is extremely costly both in time and memory.
- The system matrix will be sparse due to the fact that only approximately M of the $M \times M$ image pixels adds weight to a certain bin in the sinogram.

9.2 The Calculation of Matrix Elements

The matrix $\boldsymbol{A}$ can be estimated in several ways. One very common approach is to use a nearest neighbour approximation [96]. But a first order approximation better interpolation can also be used. As mentioned previous, alternatives to the image pixel driven generation of matrix elements can be found in [98].

Even though that some of the following interpolation schemes are very simple, they can be attractive if they can be computed fast, due to the fact that the large matrices normally are not stored in memory but calculated many times in the iterative reconstruction schemes and a good interpolation scheme takes longer time compared to the coarse interpolation schemes.

9.2.1 Pixel Oriented Nearest Neighbour Approximation

Around each pixel (x_m, y_m) is placed a square $\Delta x * \Delta x$ wide. If the line with parameters (ρ_r, θ_t) crosses the square then the matrix element a_{ij} is set to Δx, and else to 0. From Fig. 9.2 it is easy to see that the test criterion can written as

$$\rho' = \rho_r - x_m \cos\theta_t - y_n \sin\theta_t \tag{9.12}$$

$$|\rho' \cos\theta_t| < \frac{\Delta x}{2} \quad \text{and} \quad |\rho' \sin\theta_t| < \frac{\Delta x}{2} \;\Rightarrow\; a_{rT+t,nM+m} = \Delta x \tag{9.13}$$

Note that the index $rT + t$ can easily be inverted to give r and t using truncation to lower integer and the modulus operator, and likewise with the index $nM + m$. This is relevant when generating the matrix, e.g., column-wise or row-wise. Often the matrix elements are set to 1 in case of the line crosses the pixel [96] and zero otherwise. This will imply a general scaling of the solution $\boldsymbol{x}$.

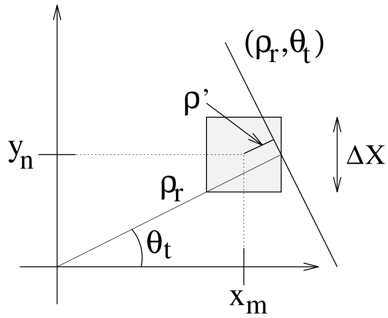

Figure 9.2 The line with index i, corresponding to (ρ_r, θ_t), crosses the square pixel centered at (x_m, y_n).

9.2.2 Discrete Radon Transform

Another interpolation scheme is to use the discrete Radon transform to approximate the forward matrix multiplication. This can be done using Algorithm 2.1, either once (requires storage of the system matrix) or every times it is needed. If multiplication with the transpose of the matrix is needed, then the discrete backprojection operator (multiplied with a factor of two) can be used, cf. Eq. 9.19 and Algorithm 8.3.

9.2.3 First Order Pixel Oriented Interpolation Strategy

Alternatively a ray-tracing strategy can be used, by assuming that the pixel at position (x_m, y_m) is a square $\Delta x * \Delta x$ wide with constant amplitude of 1. Now the length through the pixel with line orientation given by (ρ_r, θ_t) is used as the matrix element. By use of the rules for translation and scaling given in Eqs. B.6 and B.22 the result can be obtained from the Radon transform of a basic square $\breve{g}_{sq}$. If the square is centered around $(0, 0)$ with side lengths $2 * 2$, then the results found in Subsection B.3.2 can used directly

$$a_{rT+t,nM+m} = \frac{\Delta x}{2} \breve{g}_{sq} \left(2\frac{\rho_r - x_m \cos\theta_t - y_n \sin\theta_t}{\Delta x}, \theta_t \right) \tag{9.14}$$

A variation of this scheme is to use several rays and average the result from each of the rays. The rays could represent some symmetrically placed spots in the parameter domain, lying within a square corresponding to the sampling distances, which is indicated in Fig. 9.3. In this way the finite size of pixel resolution and detector size (width of lines) can be modelled.

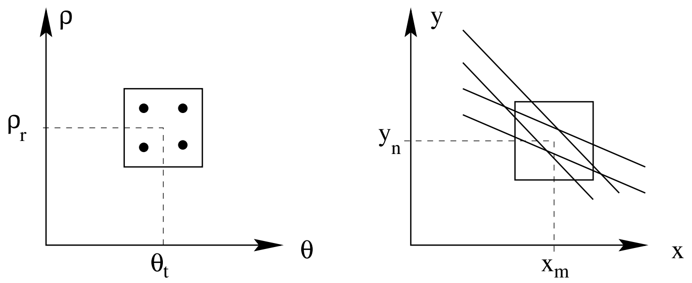

Figure 9.3 Left: A pixel in the discrete parameter domain, which here is subdivided into four center values. Right: For each of the center values the length though the square image pixel is averaged and used as the matrix value.

9.2.4 The Sinc Interpolation Strategy

Another approach is to use Eq. 2.42, which is based on a sinc interpolation scheme. This expression gives the elements of the $\boldsymbol{A}$

$$a_{rT+t,nM+m} = \frac{\Delta x}{\Psi}\sin\left(\Psi \min\left\{\frac{1}{|\sin\theta_t|},\frac{1}{|\cos\theta_t|}\right\}\right) \tag{9.15}$$

$$\Psi = \frac{\pi}{\Delta x}\left(\rho_r - x_m\cos\theta_t - y_n\sin\theta_t\right) \tag{9.16}$$

Note that this way of generating the matrix implies that some of the matrix elements will be negative, which definitely is bad from a physical point of view. Assuming that only one pixel is non-zero in the image, then negative counts will be measured for some line orientations. Nevertheless Eq. 9.15 represent a better interpolation from a signal processing point of view. Another drawback is that it is very costly to generate the matrix in this way compared to the other methods shown above.

9.3 Duality between Matrix Operations and the Radon Transform

Using the Radon transform scheme along with the matrix formalism as shown in Eq. 9.1 implies ordinary matrix operations have equivalent Radon transform operations. Eq. 9.1 is the Radon transform of discrete image $g(m,n)$ into the full discrete parameter domain $\breve{g}(r,t)$.

$$\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x} \leftrightarrow \text{Radon transform to full size parameter domain} \tag{9.17}$$

Iterative algorithms, such as ART and MART, described in the Sections 9.6 and 9.7, use the scalar product between row number i of $\boldsymbol{A}$, i.e., $\boldsymbol{a}_i$ and the current reconstructed image $\boldsymbol{x}$, i.e., $\boldsymbol{a}_i^T\boldsymbol{x}$, which is the Radon transform of the image $\boldsymbol{x}$ into one specific sample in the parameter domain.

$$b_i = \boldsymbol{a}_i^T\boldsymbol{x} \leftrightarrow \text{Radon transform to one sample in the parameter domain} \tag{9.18}$$

Another common operator in iterative algorithms is the transpose of the matrix. The operation $\tilde{\boldsymbol{x}} = \boldsymbol{A}^T\boldsymbol{b}$ is the backprojected discrete parameter domain into the image domain.

$$\tilde{\boldsymbol{x}} = \boldsymbol{A}^T\boldsymbol{b} \leftrightarrow \text{Adjoint Radon transform of the sinogram into the image domain} \tag{9.19}$$

This fact is often not recognized in the literature, but it is a direct consequence from the matrix formalism. The transpose of a matrix (without complex values) is the adjoint operator, and in this case the adjoint Radon transform is two times the backprojection operator, cf. page 134 of [14], and it is equivalent to the transpose matrix, cf. Eq. 7.10.

One well known approach to solve set of equations with a non-square system matrix is to form the normal equations.

$$\boldsymbol{A}^T \boldsymbol{b} = (\boldsymbol{A}^T \boldsymbol{A}) \boldsymbol{x} \tag{9.20}$$

An interesting analysis shown in [100] is that the solution to the normal equations

$$\boldsymbol{x} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{b} \tag{9.21}$$

can be interpreted in formalism of the direct reconstruction methods. The matrix $\boldsymbol{A}^T$ represent the backprojection operator, cf. Eq. 9.19, and then $(\boldsymbol{A}^T \boldsymbol{A})^-1$ represents the filter in Eq. 7.30. Likewise can it be shown that Filtered Backprojection can interpreted from Eq. 9.21 if assuming full rank of $\boldsymbol{A}$

$$\boldsymbol{x} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{b} \tag{9.22}$$

$$= (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T (\boldsymbol{A}\boldsymbol{A}^T)(\boldsymbol{A}\boldsymbol{A}^T)^{-1} \boldsymbol{b} \tag{9.23}$$

$$= (\boldsymbol{A}^T \boldsymbol{A})^{-1} (\boldsymbol{A}^T \boldsymbol{A}) \boldsymbol{A}^T (\boldsymbol{A}\boldsymbol{A}^T)^{-1} \boldsymbol{b} \tag{9.24}$$

$$= \boldsymbol{A}^T (\boldsymbol{A}\boldsymbol{A}^T)^{-1} \boldsymbol{b} \tag{9.25}$$

where $(\boldsymbol{A}\boldsymbol{A}^T)^{-1}$ represents the $|\nu|$ filter in Eq. 7.8 and $\boldsymbol{A}^T$ (again) represents the backprojection operator. Note that the assumption of full rank is not quite valid, due to the problems mentioned in Subsection 7.4.1 with recovering the DC level when using Filtering after Backprojection or Filtered Backprojection.

9.4 Regularization and Constraints

In linear algebra regularization or constraints of an ill-conditioned set of equations is used to stabilize the solution. The problem is that the solution to Eq. 9.11 in a two-norm sense as shown in Eq. 9.26 often is useless, due to large undesired fluctuations in the solution [101].

$$\boldsymbol{x} = \text{argmin} \left\{ \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \right\} \tag{9.26}$$

These problems are nearly related to the conditional number of the system matrix. One way to deal with this problem is to use truncated SVD, which will be described in Section 9.5. Note that Eq. 9.26 is very general least-squares fit and is not based the noise actual noise statistics found in PET sinograms.

A very simple approach to stabilize the solution is to use constraints on the solution in iterative reconstruction algorithms. If, e.g., the activity of the brain is to be reconstructed, then a non-negativity constraint can be imposed, and perhaps a upper limit on the activity also can be imposed, then the solution can be truncated to the desired values simply by setting values outside of the interval to the upper/lower limit. It seems like an ad hoc approach, but it is often used in practice [96] and constraints can definitely improve the stability of iterative reconstruction algorithms, but it seems to be difficult to analyze the implications of constraints from a theoretical point of view.

A very common way of stabilizing iterative methods is to use regularization. This can be done by adding another term to Eq. 9.26.

$$\boldsymbol{x} = \text{argmin} \left\{ \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda^2 \|\boldsymbol{L}(\boldsymbol{x} - \boldsymbol{x}^*)\|_2^2 \right\} \tag{9.27}$$

where $\boldsymbol{x}^*$ is an estimate of the solution, and if none is available it can be set to zero. The operator $\boldsymbol{L}$ is a matrix which can be selected to the identity matrix (restricts large values) or to approximate the first or second order derivative of the solution $\boldsymbol{x}$ (restricts fast variations). The parameter λ controls how much weight the regularization term should have. Methods to estimate the optimal value of λ can be found in [101].

A simple way to include regularization terms in the reconstruction algorithms is to expand the set of equation by adding a set of regularization rows to the original set of equation

$$\begin{pmatrix} \boldsymbol{A} \\ \lambda \boldsymbol{L} \end{pmatrix} \boldsymbol{x} = \begin{pmatrix} \boldsymbol{b} \\ \lambda \boldsymbol{L} \boldsymbol{x}^* \end{pmatrix} \tag{9.28}$$

But it should be noted that the noise properties of the original and the the expanded set of equations is normally are very different, and it is not easy to predict theoretically how it will influence on a particular iterative reconstruction algorithm.

9.5 Singular Value Decomposition

The system matrix $\boldsymbol{A}$ can be analyzed by use of the SVD (Singular Value Decomposition) [102, 73, 63]. The basic idea of the SVD is to decompose the matrix $\boldsymbol{A} \in \mathbb{R}^{I \times J}$ as

$$\boldsymbol{A} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^T \tag{9.29}$$

where the matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{I \times J}$ is a diagonal matrix with the singular values of $\boldsymbol{A}$ on the diagonal, $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \sigma_2, ..., \sigma_J)$ and the matrices $\boldsymbol{U} \in \mathbb{R}^{I \times I}$ and $\boldsymbol{V} \in \mathbb{R}^{J \times J}$ are orthogonal matrices. The singular values are normally ordered in a non-increasing order, $\sigma_i \geq \sigma_{i+1}$, and the conditional number of $\boldsymbol{A}$ is the ratio between the largest and the smallest singular value.

$$\mathrm{cond}(\boldsymbol{A}) = \frac{\sigma_1}{\sigma_J} \quad J \leq I \tag{9.30}$$

To illustrate the properties of the system matrix, $\boldsymbol{A}$ has been calculated from Eq. 9.15, cf. Section 9.2. The sampling parameters are been chosen as shown in Eqs. 9.31-9.34, and the corresponding system matrix shown in Fig. 9.4.

Note that this example is a toy example, as it only concerns a very small $21 * 21$ image reconstructed from a $21 * 20$ sinogram, but it illustrates some of the basic properties of the system matrix. In this example the SVD could be computed in approximately 80 sec on a Pentium 120 MHz using the SVD-routine in Matlab, and the decomposition required approximately additionally 4 MBytes of memory.

$$x_m = -1 + m \cdot 0.1, \; m = 0, 1, ..., 20 \tag{9.31}$$
$$y_n = -1 + n \cdot 0.1, \; n = 0, 1, ..., 20 \tag{9.32}$$
$$\rho_r = -1 + r \cdot 0.1, \; r = 0, 1, ..., 20 \tag{9.33}$$
$$\theta_t = t \cdot \frac{\pi}{20}, \; t = 0, 1, ..., 19 \tag{9.34}$$

The singular values is shown in Fig. 9.5, and it can be seen that approximately 20 large singular values are succeeded by approximately 255 slowly decaying ones, and here a dramatic change occurs and then the singular values are decaying rapidly. The 20 large singular values corresponds to the size of the system matrix and the actual sampling. It can be seen that the

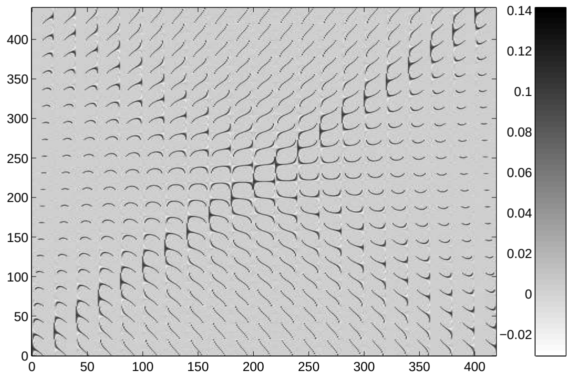

Figure 9.4 The system matrix $\boldsymbol{A}$.

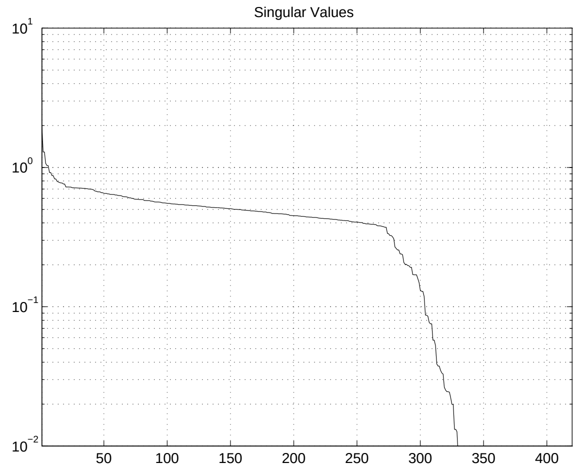

Figure 9.5 The singular values of the system matrix $\boldsymbol{A}$.

problem is ill-conditioned. This is normally mended by restricting the variations of $\boldsymbol{x}$, by use of regularization [101].

One way to invert an ill-conditioned problem is to separate the matrix $\boldsymbol{\Sigma}$ in a part with large singular values $\boldsymbol{\Sigma}_1$ and a part with small singular values $\boldsymbol{\Sigma}_2$.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma_1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma_2} \end{bmatrix} \tag{9.35}$$

i.e., the singular values in $\boldsymbol{\Sigma}_2$ are insignificant. Choosing a rank p approximation, i.e. $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{p \times p}$. The generalized inverse of $\boldsymbol{\Sigma}$ is defined as

$$\boldsymbol{\Sigma}^+ = \begin{bmatrix} \boldsymbol{\Sigma_1}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \tag{9.36}$$

Note that this inversion is easily computed. The generalized inverse can be used for inversion of Eq. 9.11.

$$\boldsymbol{x} = \boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^T\boldsymbol{b} \tag{9.37}$$

This solution is a minimum norm solution to the least square problem $\min_{\boldsymbol{x}} \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2$.

From Fig. 9.5 it could be suggested to separate large and small singular values at singular value number 275. In Fig. 9.6 is shown the generalized inverse matrix using the first 275 singular values. A very stable structure is found looking something like the one found in Fig. 9.4, which could be expected due to the underlying line transformation model. Next, Fig. 9.7 shows the generalized inverse matrix using the first 320 singular values. Very little structure can be seen and here the inverted matrix is dominated by fast oscillations.

The suggested inversion strategy is not viable for varying geometries because SVD has complexity $\mathcal{O}(4I^2J + 8IJ^2)$, cf. page 248 of [102], where I and J is the number of rows and columns of the matrix $\boldsymbol{A}$ respectively. This implies that the SVD used for inversion of the Radon transform has complexity approximately $\mathcal{O}(M^6)$, where M still is the number of pixels on each of the axes in the image domain. Here it should be noted that for a fixed scanner geometry the SVD only has to be computed once. After the SVD decomposition of the system matrix the minimum norm solution can easily be found using matrix multiplications, and a varying rank or filtering the singular values more softly by use of, e.g., the popular Tikhonov regularization, e.g., [101, 103] is

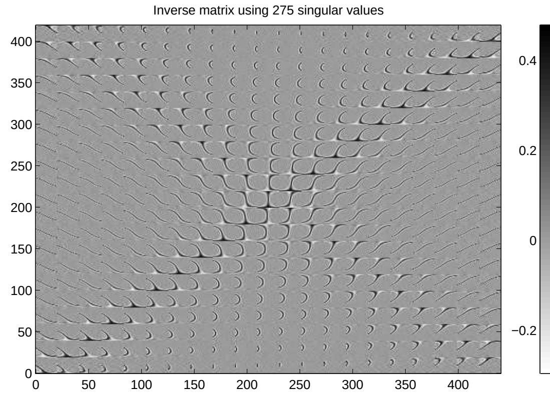

Figure 9.6 The generalized inverse of the system matrix using 275 singular values. A structure resembling the transpose of the system matrix can be seen.

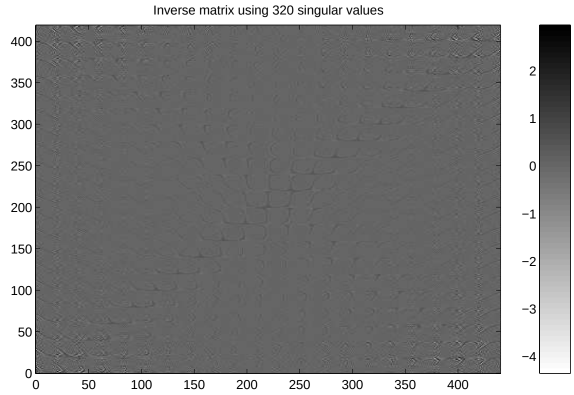

Figure 9.7 The generalized inverse of the system matrix using 320 singular values. The structure has vanished and note the span in amplitude compared to Fig. 9.6.

easily incorporated in the inversion scheme. Despite the high computational complexity, the SVD has been used for reconstruction of SPECT images [104, 105], and it has been reported that the SVD of matrices with size 7080×4436 can be computed in approximately 35 hours on a SUN Sparc 10/40. For 3D reconstruction with a significantly higher number of sinogram values and voxels, the SVD based reconstruction requires a change in computing technology.

9.6 Iterative Reconstruction using ART

A well-known way to solve Eq. 9.11 is named ART, which stands for Algebraic Reconstruction Technique. Many articles, e.g., [96, 106, 73] demonstrate the algorithm. ART was published in the biomedical literature in 1970, and Cormack and Hounsfield used ART for reconstructing the very first tomography images. They probably did not know of the work of Johann Radon at that point, and later it was discovered that ART is a reinvention of the algorithm introduced by Kaczmarz [107] in back in 1937.

The basis operator required in ART is the scalar product between to certain row i of the system matrix, $\boldsymbol{a}_i$ and a solution vector $\tilde{\boldsymbol{x}}$

$$\tilde{b}_i = \sum_{j=1}^{J} a_{ij}\ \tilde{x}_j = \boldsymbol{a}_i^T \tilde{\boldsymbol{x}} \tag{9.38}$$

ART is formulated as an iterative reconstruction algorithm, where the solution vector in iteration k is updated by adding a scaled version of row i of the system matrix.

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \frac{b_i - \boldsymbol{a}_i^T \boldsymbol{x}^{(k)}}{\boldsymbol{a}_i^T \boldsymbol{a}_i}\, \boldsymbol{a}_i \tag{9.39}$$

The main idea of ART is that the equation i is fulfilled in iteration k, which easily is shown:

$$\boldsymbol{a}_i^T \boldsymbol{x}^{(k)} = \boldsymbol{a}_i^T \boldsymbol{x}^{(k-1)} + \frac{b_i - \boldsymbol{a}_i^T \boldsymbol{x}^{(k-1)}}{\boldsymbol{a}_i^T \boldsymbol{a}_i}\, \boldsymbol{a}_i^T \boldsymbol{a}_i = b_i \tag{9.40}$$

If using the Radon transformation terminology, then ART will modify the reconstructed image in iteration k, in order to give the correct Radon transform at (ρ_r, θ_t), where (r, t) is found from the sinogram sorting scheme shown in Eq. 9.2. In this way the reconstruction quality of ART in a given area can be altered by choosing that i matches the lines passing through an interesting area frequently.

Often [73] i as a function of k is chosen to $i = k$ MOD I, where MOD is the modulus operator, but this choice is not very good [106, 108]. Another very common and easy strategy is to choose i randomly using a uniform probability density function. Initially $\boldsymbol{x}^{(0)}$ can be chosen to zero or some good guess on the solution, e.g., in 2D from a fast algorithm like one based on Fourier Slice Theorem. Yet another strategy is to initialize all solution values with a constant, as it will be shown in Subsection 9.6.2.

The ART algorithm can also be interpreted from a geometrical point of view. Fig. 9.8 shows the use of ART in a two parameter estimation problem with two projection lines. As shown in Eq. 9.40 the current solution in iteration k, i.e., $\boldsymbol{x}^{(k)}$ is projected perpendicularly (along the direction of $\boldsymbol{a}_i^T$) onto the hyperplane (in this case a line) determined by $\boldsymbol{b}_{i(k)} = \boldsymbol{a}_i^T \boldsymbol{x}^{(k)}$. As seen from Fig. 9.8 the speed of convergence is very dependent on the angle between the hyperplanes.

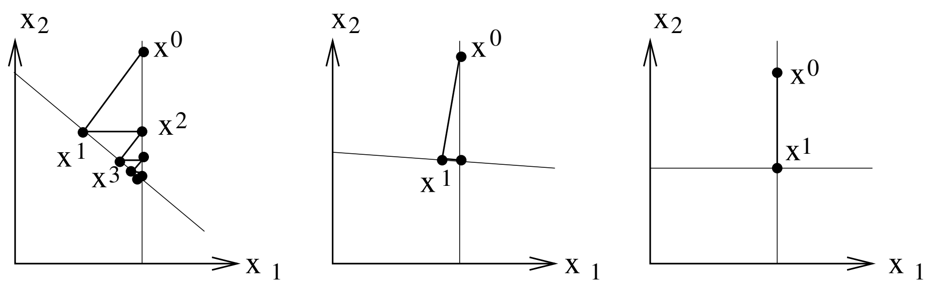

Figure 9.8 Three different cases of iteration in a two parameter problem using ART. Depending of the orthogonality of the hyperplanes (here lines) the convergence can be slow (left most figure) or fast (right most figure).

Note that ART in each iteration only requires the scalar product between $\boldsymbol{a}_i$ and the current solution $\boldsymbol{x}^{(k)}$, which can be compared to calculating one value in the parameter domain using discrete Radon transform, hence each iteration is very fast but the quality enhancement gained from one iteration is in general very limited. Often when comparing ART to other iterative algorithms that requires computation of a full discrete parameter domain, the iteration number used for ART is a full loop through all I rows, i.e., I times the actual number of iterations in ART.

A common alteration of ART is to introduce a relaxation parameter in form of a weight factor as shown in Eq. 9.41.

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \lambda_k \, \frac{b_i - \boldsymbol{a}_i^T \boldsymbol{x}^{(k-1)}}{\boldsymbol{a}_i^T \boldsymbol{a}_i} \, \boldsymbol{a}_i^T \tag{9.41}$$

where λ_k can be set as a simple function of k, such as a linear function or a exponential decay. Even given the result Eq. 9.40 it has experimentally been proven [109], that setting λ_k to values different from one can improve the speed of convergence. It should be noted that the optimum value of λ_k is a function of k, the sinogram values, and the sampling parameters of the reconstructed image. In [106] it is shown that optimizing the value of λ_k in each iteration and careful selection of row index i as a function of k results in a reconstructed image quality as good as using the EM-algorithm, at a order of magnitude less computational cost (The EM-algorithm is presented in Section 9.8).

In the literature, authors have optimized ART and/or EM, and for some years it was not clear whether the one was better than the other. Now it seems as if ART is loosing popularity compared to EM for 2D reconstruction, but for 3D reconstruction the opposite seems to be the case [110] and Section 10.6.

As shown in Algorithms 9.1 and 9.2 ART is very easy to implement. Only a scaler product is really needed. In the algorithm the matrix elements are used over and over again, and note for a huge problem which cannot be stored in memory all at one time each matrix element has be calculated in a function like it was shown in Section 9.2. In Algorithm 9.1 the denominator values $\boldsymbol{a}_i^T \boldsymbol{a}_i$ are computed once as a function of i. The reason for showing two algorithms is that the first part can be computed once, and if several sinograms with the same geometry should be reconstructed, then it is only Algorithm 9.2 which should be used several times.

Algorithm 9.1 : Initialization of the ART algorithm

```
For i = 0 to I-1                        //For all values of i
  sum = 0                               //Calculate the denominator
  For j = 0 to J-1                      //For all values of j
    sum = sum + a(i,j)*a(i,j)           //Accumulate value
  End
  anorm(i)=sum                          //Store denominator
End
For j = 0 to J-1                        //For all values of j
  x(j) = c                              //Initialize with a constant
End
```

Algorithm 9.2 : The ART algorithm

```
For k = 0 to K-1                        //For K iterations of ART
  Set i as a function of k              //Choose row index in iteration k
  sum = 0                               //Initialize scalar product
  For j = 0 to J-1                      //For all values of j
    sum = sum + a(i,j)*x(j)             //Update scalar product
  End
  w = lambda(k)*(b(i)-sum)/anorm(i)     //Calculate common weight
  For j = 0 to J-1                      //For all values of j
    x(j) = x(j) + w*a(i,j)              //Project solution
  End
End
```

9.6.1 ART with Constraints

It is not guaranteed, that ART will provide a non-negative solution, as required by the physical meaning of the solution; in PET emission activity and in CT absorption coefficient. As mentioned in Section 9.4 a crude, but very easily implemented strategy is to restrict the solution after some iterations from a upper limit estimate on the solution in each pixel, i.e., vector element or maybe just using a upper limit constant in each iteration. A non-negativity constraint can be imposed as shown in Algorithm 9.3, where the initialization part shown in Algorithm 9.1 has been removed. Other constraining schemes for ART can be found in [96].

Algorithm 9.3 : The ART algorithm with Constraints

```
For k = 0 to K-1                        //For K iterations of ART
  Set i as a function of k              //Choose row index in iteration k
  sum = 0                               //Initialize scalar product
  For j = 0 to J-1                      //For all values of j
    sum = sum + a(i,j)*x(j)             //Update scalar product
  End
  w = lambda(k)*(b(i)-sum)/anorm(i)     //Calculate common weight
  For j = 0 to J-1                      //For all values of j
    x(j) = x(j) + w*a(i,j)              //Project solution
  End
  If k MOD kc = 0                       //Every kc iterations use constraints
    For j = 0 to J-1                    //For all values of j
      If x(j) < 0                       //Negative solution is found
        x(j) = 0                        //which is set to zero
      End
      If x(j) > Maxx(j)                 //Too large solution is found
        x(j) = Maxx(j)                  //which is set to max limit
      End
    End
  End
End
```

9.6.2 Initialization

Using iterative algorithms also implies that the solution vector $\boldsymbol{x}$ should be initialized with a constant value or a good start guess, which need not be a constant. One possibility is to use fast direct algorithms, such as Fourier Slice based methods, for producing a start guess. If the start guess is good the iterative algorithms in general will converge faster, but it also implies that the behavior of the algorithm will be biased by the direct method. Another very common starting guess is to initialize the solution with a constant [111]. For the ART algorithm no restrictions are made concerning the initial value, but for other iterative algorithms such as MART (Section 9.7) and EM (Section 9.8) the initial value has to be positive. Assuming that the solution is a constant, then by averaging in Eq. 9.11 the value should be

$$x_j^0 = \frac{\sum_{i=1}^{I} b_i}{\sum_{j=1}^{J}\sum_{i=1}^{I} a_{i,j}} \;\forall\, j \tag{9.42}$$

This initialization requires that the system matrix is computed an additional time or can be combined with the first part of Algorithm 9.1. A faster scheme is to use the approximation that for a certain value of i a line approximately crosses M pixels each with a value of approximately Δx, hence

$$\sum_{j=1}^{J}\sum_{i=1}^{I} a_{i,j} \approx IM\Delta x \;\Rightarrow\; x_j^0 = \frac{1}{IM\Delta x}\sum_{i=1}^{I} b_i \;\forall\, j \tag{9.43}$$

9.7 Multiplicative ART

Another technique is MART (Multiplicative Algebraic Reconstruction Technique), that can be found in the literature, but MART has apparently not had a major impact in practical reconstruction applications. The basis of MART is to maximize the entropy of the solution

$$-\sum_{j=1}^{J} x_j \log x_j \tag{9.44}$$

under the constraints that

$$b_i = \boldsymbol{a}_i^T \boldsymbol{x}^{(k)} \text{ and } \boldsymbol{x}_j \geq 0 \ \forall j \tag{9.45}$$

Like ART, the algorithm is very easy. At first all image values are initialized to the constant value

$$x^0 = e^{-1} \tag{9.46}$$

and then the general iteration step looks like

$$\boldsymbol{x}_j^{(k+1)} = \left(\frac{b_i}{\boldsymbol{a}_i^T \cdot \boldsymbol{x}^{(k)}} \right)^{\lambda_k a_{i,j}} \boldsymbol{x}_j^{(k)} \tag{9.47}$$

where λ_k is a positive tunable relaxation parameter. Just like ART will MART require a scheme for selecting the row index i as a function of the iteration index k. Again two strategies are commonly used; cyclical ($i = k$ MOD I) or randomly with a uniform probability distribution over all indices.

According to [96] the behavior of the algorithm is not properly known, especially with respect to the influence of noise in the sinogram. Another problem is that the denominator in Eq. 9.47 can become zero, which indicates that problems with instability can be expected.

9.8 The EM algorithm

So far the reconstruction methods have modelled the projections as line integrals, which was reconstructed by use of discretization of the inverse Radon transform. Especially in emission tomography with limited counts in each sinogram bin, the statistical noise can dominate the reconstructed images when using direct reconstruction methods, cf. Subsection 8.5.5. This lead many scientists to consider statistical approaches to derive reconstruction algorithms.

One of the most prominent iterative reconstruction methods for emission tomography is Maximum Likelihood Reconstruction using the EM algorithm, which stands for Expectation Maximization. In the very famous articles [80] by Shepp and Vardi and [112] by Vardi, Shepp, and Kaufman a statistical framework for reconstruction is given. It is assumed that the measurements originate from uncorrelated Poisson generators, which ideally model the underlying physics, but problems like attenuation correction are not modelled in this framework. Another feature of the EM-algorithm is that the model includes a non-negativity constraint.

The key idea of the EM-algorithm is to maximize the Likelihood

$$L(\boldsymbol{x}) = P(\boldsymbol{b}|\boldsymbol{x}) = \prod_{i=1}^{I} \frac{(b_i^*)^{b_i}}{b_i!} e^{-b_i^*} \tag{9.48}$$

where $\boldsymbol{b}^*$ contains the unknown mean values of $\boldsymbol{b}$, i.e., the true sinogram without noise, and it is assumed that $\boldsymbol{b}^*$ fit the solution perfectly

$$\boldsymbol{b}^* = \boldsymbol{A}\boldsymbol{x} \tag{9.49}$$

where the coefficients of the system matrix are considered as transition probabilities normalized as

$$1 = \sum_{i=1}^{I} a_{i,j} \tag{9.50}$$

in PET meaning that a photon pair at detector pair i originates from one of the regions in the brain (pixels in the image) with the probability of one.

Now the expectation of the Likelihood is maximized from the log Likelihood $l(\boldsymbol{x})$ by setting the derivatives of $l(\boldsymbol{x})$ to zero

$$\frac{\partial l(\boldsymbol{x})}{\partial x_j} = -\sum_{i=1}^{I} a_{i,j} + \sum_{i=1}^{I} \frac{a_{i,j} b_i}{\sum_{j'=1}^{J} a_{i,j'} x_{j'}} = -1 + \sum_{i=1}^{I} \frac{a_{i,j} b_i}{\sum_{j'=1}^{J} a_{i,j'} x_{j'}} = 0 \tag{9.51}$$

In the literature an additional non-negativity constraint is imposed and it can be shown that this results in the Kuhn-Tucker conditions,

$$x_j \frac{\partial l(\boldsymbol{x})}{\partial x_j} = 0 \quad \forall\ j \text{ where } x_j > 0 \tag{9.52}$$

$$\frac{\partial l(\boldsymbol{x})}{\partial x_j} \leq 0 \quad \forall\ j \text{ where } x_j = 0 \tag{9.53}$$

where the first equation is used to formulate an iterative mapping scheme

$$0 = x_j \frac{\partial l(\boldsymbol{x})}{\partial x_j} = x_j \left(-1 + \sum_{i=1}^{I} \frac{a_{i,j} b_i}{\sum_{j'=1}^{J} a_{i,j'} x_{j'}} \right) \Rightarrow \tag{9.54}$$

$$x_j^{(k)} = x_j^{(k-1)} \sum_{i=1}^{I} \frac{a_{i,j} b_i}{\sum_{j'=1}^{J} a_{i,j'} x_j^{(k-1)}} \tag{9.55}$$

This equation is very often found in statistical reconstruction literature, but it should be noted that it requires that the elements of the system matrix is properly normalized. Here another version of the EM algorithm proposed in [113], and found to give very good results in [114], is used. It does not require the assumption shown in Eq. 9.50, and works fine with the normalization used in Section 9.2.

$$x_j^{(k)} = \frac{x_j^{(k-1)}}{\sum_{i'=1}^{I} a_{i',j}} \sum_{i=1}^{I} \frac{a_{i,j} b_i}{\sum_{j'=1}^{J} a_{i,j'} x_j^{(k-1)}} \tag{9.56}$$

Eq. 9.56 is a very compact form to show the EM-algorithm, but it does not show that each iteration consists of four steps

$$\boldsymbol{b}^f = \boldsymbol{A}\boldsymbol{x}^{(k-1)} \tag{9.57}$$

$$b_i^q = \frac{b_i}{b_i^f} \tag{9.58}$$

$$\boldsymbol{x}^b = \boldsymbol{A}^T\boldsymbol{b}^q \tag{9.59}$$

$$x_j^{(k)} = \frac{x_j^{(k-1)}x_j^b}{s_j} \tag{9.60}$$

where $s_j = \sum_{i=1}^{I} a_{i,j}$, forms a normalization vector that can be calculated once for all.

In Subsection 9.11.4 it will be demonstrated, that Eq. 9.58 can lead to instability. Assume a PET imaging situation where a certain i corresponds to a line not entering regions of activity, hence the forward projected value b_i^f becomes zero, thus the denominator in Eq. 9.58 is a potential stability problem.

The EM algorithm is computationally demanding. Eq. 9.57 shows that each iteration requires a forward projection (Radon transform) of the current solution, cf. Eq. 9.17. The quotient in each point between the measured sinogram b_i and the forward projected solution b_i^f, i.e., $\boldsymbol{b}^q$ is then backprojected into the image domain. Finally, in Eq. 9.60, the next estimate of the solution is generated by multiplying for each index j, the current estimate of the solution $x_j^{(k-1)}$ with the backprojected solution x_j^b weighted with s_j. This shows that each iteration of EM actually costs more than using, e.g., Filtered Backprojection. Note also that the EM algorithm is nonlinear, hence additive noise in the sinogram will not automatically lead to an additive noise term in the reconstructed images, as it has been the case for all the previous methods. For further reading on the EM algorithm for Maximum Likelihood iterative reconstruction methods [80, 112, 115, 116, 111, 113] are recommended.

In Algorithms 9.4 and 9.5 are shown the implementation of the EM-algorithm. It has been split into two parts indicating that the first part has to be calculated once, and the last part can then be used to reconstruct several images with the same geometry.

Algorithm 9.4 : Initialization of the EM algorithm

```
For j = 0 to J-1                    //For all values of j
  sum = 0                           //Calculate the values of s(j)
  For i = 0 to I-1                  //For all values of i
    sum = sum + a(i,j)              //Accumulate value
  End
  s(j)=sum                          //Store value
End
For j = 0 to J-1                    //For all values of j
  x(j) = c                          //Initialize with a constant
End
```

Note that the initialization of the EM-algorithm requires that the system matrix is generated once in the initialization part, but this can be avoided by initializing the solution vector as as shown in Algorithm 9.6, and then use Algorithm 9.5 for the remaining iterations. Note that in the first line of the Algorithm 9.5 the counter should then be For k = 1 to K-1.

Algorithm 9.5 : The EM algorithm

```
For k = 0 to K-1                              //For K iterations of EM
  For i = 0 to I-1                            //For all values of row index
    sum = 0                                   //Initialize scalar product
    For j = 0 to J-1                          //For all values of j
      sum = sum + a(i,j)*x(j)                 //Update scalar product
    End
    bq(i) = b(i)/sum                          //Store scaled forward projection
  End
  For j = 0 to J-1                            //For all values of column index
    sum=0                                     //Initialize backprojection sum
    For i = 0 to I-1                          //For all values of row index
      sum = sum + a(i,j)*bq(i)                //Accumulate sum
    End
    x(j) = x(j)*sum/s(j)                      //Update solution
  End
End
```

Algorithm 9.6 : The first iteration of the EM algorithm

```
For i = 0 to I-1                              //For all values of row index
  sum = 0                                     //Initialize scalar product
  For j = 0 to J-1                            //For all values of j
    sum = sum + a(i,j)*x(j)                   //Update scalar product
  End
  bq(i) = b(i)/sum                            //Store scaled forward projection
End
For j = 0 to J-1                              //For all values of column index
  s(j) = 0                                    //Initialize s-values
  sum=0                                       //Initialize backprojection sum
  For i = 0 to I-1                            //For all values of row index
    la = a(i,j)                               //Store value in simple variable
    sum = sum + la*bq(i)                      //Accumulate sum
    s(j) = s(j) + la                          //Increment s
  End
  x(j) =x(j)*sum/s(j)                         //Update solution
End
```

9.9 The Conjugate Gradient Method

The Conjugate Gradient algorithm (or CG for short) is a well-documented technique for iteratively solving equation systems with a symmetrical positive definite system matrix [101, 117]. For reconstruction purposes the Conjugate Gradient algorithm is not well suited, due to the restrictions on the system matrix, but reconstruction can form the normal equations, shown in Eq. 9.20, which gives the least squares solution to the original set of equations, shown in Eq. 9.11. Using normal equations the effective system matrix $\boldsymbol{A}^T\boldsymbol{A}$ is symmetrical and the singular values are now squared compared to the singular values of $\boldsymbol{A}$.

Given that the system matrix is sparse, which can be exploited for effective storage, the system matrix of the normal equations $\boldsymbol{A}^T\boldsymbol{A}$ should not be generated, because it in general will not be sparse, thus it requires a lot of storage space. In several textbooks, e.g., [117] an algorithm is found implementing the Least Squares Conjugate Gradient method (LSCG) without computing $\boldsymbol{A}^T\boldsymbol{A}$.

$$\text{Initialize } \boldsymbol{x}^0 \tag{9.61}$$
$$\boldsymbol{s}^0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^0 \tag{9.62}$$
$$\boldsymbol{r}^0 = \boldsymbol{p}^0 = \boldsymbol{A}^T\boldsymbol{s}^0 \tag{9.63}$$
$$\boldsymbol{q}^0 = \boldsymbol{A}\boldsymbol{p}^0 \tag{9.64}$$

Then for each iteration the LSCG algorithm on the normal equations becomes

$$\alpha = \frac{\|\boldsymbol{r}^{k-1}\|_2^2}{\|\boldsymbol{q}^{k-1}\|_2^2} \tag{9.65}$$
$$\boldsymbol{x}^k = \boldsymbol{x}^{k-1} + \alpha\boldsymbol{p}^{k-1} \tag{9.66}$$
$$\boldsymbol{r}^k = \boldsymbol{A}^T\boldsymbol{s}^{k-1} \tag{9.67}$$
$$\boldsymbol{r}^k = \boldsymbol{r}^{k-1} - \alpha\boldsymbol{q}^{k-1} \tag{9.68}$$
$$\beta = \frac{\|\boldsymbol{r}^{k}\|_2^2}{\|\boldsymbol{r}^{k-1}\|_2^2} \tag{9.69}$$
$$\boldsymbol{p}^k = \boldsymbol{r}^k + \beta\boldsymbol{p}^{k-1} \tag{9.70}$$
$$\boldsymbol{q}^k = \boldsymbol{A}\boldsymbol{p}^k \tag{9.71}$$

Note that each step of the iteration requires several vector updates and scalar products, by the bulk of the computational load lies in the forward projection and the backprojection operation. So the computational complexity per iteration resembles that of the EM-algorithm. No algorithm is given here, due to the fact that Eqs. 9.61-9.71 only includes simple operations, but many, hence the pseudo code will be long.

Note that the LSCG algorithm estimates a solution for the equation system shown in Eq. 9.20, which implies that the effective system matrix $\boldsymbol{A}^T\boldsymbol{A}$ squares the singular values, and a bad conditional number gets squared. According to [101] the CG method has an inherent regularization and in [118] the regularization properties are examined.

In [100], it is demonstrated that for the LSCG Method, it really does not matter, whether a constant (here zeros) or an image generated by use of Filtered Backprojection is used.

9.10 Accelerated Iterative Reconstruction

From an implementation point of view the iterative algorithms shown in the previous sections, so far have been treated as if all the values in the system matrix $a_{i,j}$ are available in general. One problem is that the system matrix is normally huge. A 2D sinogram from, e.g., a GE Advance PET scanner contains $I = 281 * 336$ values, and reconstructed into a $J = 301 * 301$ grid, i.e., the system matrix has approximately 8.5 billion elements, requiring over 34 GBytes of memory, when using 4 bytes per matrix element. This is a lot of memory, even looking some years into the future. Besides, it would not be wise to store all that data, due to the fact that approximately 98% of the matrix entries will be zeros. This knowledge should be incorporated into the reconstruction schemes.

Assuming that memory is not available for storing the full system matrix, one possibility is to compute the individual matrix elements in each iteration when needed. This can be done by use of the discrete Radon transform, as described in Section 9.2 or other modelling schemes for the scanner. This approach is viable, is rather easily implemented, and the storage requirements are reduced to a minimum. Memory is only required for the sinogram ($\boldsymbol{b}$) and the current solution ($\boldsymbol{x}$), and perhaps some additional temporary variables of the same size or smaller, but no system matrix is stored in memory. It will be demonstrated that this implementation has a major drawback in speed, since the system matrix will be computed many times during an iterative reconstruction. Each time with the same high computational cost.

Here a new hybrid solution is proposed [5, 6] for accelerating the iterative reconstruction algorithms, but requiring as much memory as modern workstations are currently equipped with, or will be soon. The idea is to only store the non-zero elements of the system matrix in the main memory using sparse matrix techniques. In this way the core of the reconstruction algorithms, highly based on matrix vector multiplications, can be accelerated significantly, and thereby removing one of the major drawbacks of the iterative methods.

It is proposed that the system matrix $\boldsymbol{A}$ is calculated one time only using all the modifications found for the actual scanner setup. If no specific scanner model is provided then the system matrix can be modelled and generated using the Radon transform or other simpler schemes as shown in Section 9.2. From the system matrix the very small values are truncated to zero.

$$\tilde{a}_{i,j} = \begin{cases} a_{i,j} & \text{if } a_{i,j} > \gamma \\ 0 & \text{Otherwise} \end{cases} \tag{9.72}$$

where the threshold γ can be chosen to a small fraction of the maximum matrix value, e.g., $\gamma = 0.05 \max_{i,j}\{a_{i,j}\}$. If γ is chosen sufficiently low, a good compromise between resolution and the sparseness of the matrix can be reached, and normally this does not alter the behavior of the algorithms, and for some of the schemes used to estimate the matrix values the value of γ can be set to 0, due to a finite interpolation width in, e.q., Eq. 9.13.

The sparse structure of $\boldsymbol{A}$ can be exploited by only storing non-zero values in the fast memory. For a certain row, number i, all of the matrix elements are calculated, stored, and truncated using Eq. 9.72. Hereby will the number of non-zero elements in the row, denoted by Z_i, be much smaller than the image size $J = M^2$. The values of Z_i are stored in a simple one dimensional vector. Two vectors of length Z_i, indexed by an integer z, can then be allocated and stored containing the non-zero matrix value a_z and the corresponding column index j_z. The procedure is repeated for all rows.

Assuming a nearest neighbour approximation with one pixel for each point along the integration lines and using 4 bytes for storing each of the vector elements, then the total storage requirement is reduced to approximately $8\sum_{i=1}^{I} z_i \approx 8IM$ bytes. In the example shown above approximately 100 MBytes of memory is required. Assuming that amount of memory is present most iterative algorithms can be implemented from three basic operations: Matrix vector multiplication $\boldsymbol{A}\tilde{\boldsymbol{x}}$, scalar product between the i'th row of the system matrix and a vector $\boldsymbol{a}_i^T\tilde{\boldsymbol{x}}$, and finally multiplication with the transpose of the matrix $\boldsymbol{A}^T\tilde{\boldsymbol{b}}$.

In the following, the implementations of the scalar product (Algorithm 9.7), the matrix vector multiplication (Algorithm 9.8), and the multiplication with the transpose the system matrix (Algorithm 9.9) are shown.

ALGORITHM 9.7 : $\boldsymbol{b}_i^f = \boldsymbol{a}_i^T\tilde{\boldsymbol{x}}$

```
Set a and j to correct row                //Use pointer technique
sum = 0                                   //Initialize
For z=0 to Z(i)-1                         //For row i
  sum=sum+a(z)*x(j(z))                    //Increment sum
End
bt=sum                                    //Store value
```

ALGORITHM 9.8 : $b^f = \boldsymbol{A}\boldsymbol{x}$

```
For i = 0 to I-1                          //For all rows
  sum = 0                                 //Initialize
  Set a and j to correct row              //Use pointers
  For z=0 to Z(i)-1                       //For row i
    sum=sum+a(z)*x(j(z))                  //Increment sum
  End
  bt(i)=sum                               //Store value
End
```

ALGORITHM 9.9 : $\boldsymbol{x}^b = \boldsymbol{A}^T\boldsymbol{b}$

```
For j=0 to J-1                            //For all columns
  xb(j)=0                                 //Initialize
End
For i=0 to I-1                            //For all rows
  Set a and j to correct row              //Using pointers
  For z=0 to Z(i)-1                       //Compute sum
    xb(j(z))=xb(j(z))+a(z)*b(i)           //Update sum
  End
End
```

9.10.1 A Fast 2D Iterative Reconstruction Package

A software package, named “it” has been written in the C-language. A short description is provided in Section C.3. The package includes the proper structures for manipulating sparse matrices and vectors, along with optimized code for computing matrix vector like products, well suited for iterative reconstruction algorithms. The software package are provided for free from [10].

In the package ART, EM, and LSCG are implemented both in a fast version using sparse matrix storage of the system matrix and in a slow version where the system matrix is not stored and required matrix entries are computed in each part of the iterative steps. In this way the package can be used to reconstruct any size of images, and if enough memory is available the reconstruction is very fast.

The package also features constraining of the solution using either a constant minimum and a constant maximum for each of the solution values, or an image can be supplied masking the ROI (region of interest), used to limit the solutions differently in several areas.

Currently one regularization option is found in the package, namely a discrete version of the Laplace operator, shown in Table 9.1, used in the reconstructed image. The operator and the image are convolved, and for each pixel in the image the regularization term will add one line to the system matrix, cf. Eq. 9.28. Likewise is the $\boldsymbol{b}$-vector expanded with zeros match the extra number of rows of the system matrix.

0	-1	0
-1	4	-1
0	-1	0

Table 9.1 Discrete approximation to the Laplace operator, used in the image domain.

In the software package "it" it has been done by appending extra rows to the system matrix at runtime, after the system matrix has been generated or read from the disk. Due to the use of sparse storage techniques and the small support of the operator, the appended part to the system matrix can be generated very fast and will not require much memory, hence the regularization factor λ can be supplied at runtime and the software package can be used number of times with different regularization factors and use the same system matrix stored on the disk.

Several interpolation methods have been implemented for generation of the system matrix.

- Binary estimation of the matrix matrix, cf. Eq. 9.13.
- Nearest Neighbour interpolation based on the Radon transform, cf. Algorithm 2.1.
- Linear interpolation based on the Radon transform.
- Analytical Radon transform of a square, i.e., the length through a quadratic pixel is used, cf. Subsection B.3.2.
- The Radon transform of a sinc expansion in the image domain, cf. Eq. 9.15.

9.11 Examples Using Iterative Reconstruction Algorithms

In this section a set of examples are presented where iterative reconstruction methods have been used. The examples have been generated with the package "it". The artificial sinograms used for some examples is generated using "RadonAna". Both packages are described in Appendix C.

The program has been used on two types of machines. A Linux machine with a 120 MHz Pentium processor and an Silicon Graphics (SGI) Onyx equipped with four 200 MHz R4400 processors, where the program was running on one processor.

9.11.1 A Small Reconstruction Example

In the first example the (synthetic) sinogram has $125 * 101$ samples and the reconstructed image has $101 * 101$ samples. In Table 9.2 the reconstruction times on both machines are shown for the fast and the slow method as well as the ratio between the execution times (slow/fast).

Times are measured for ART, EM, and the LSCG-method, when EM and LSCG were running (arbitrarily) 20 iterations, and ART 20 full iterations, i.e., 20 times the number of rows (chosen randomly), which is 20∗125∗101 iterations in Eq. 9.39. Note that all times only correspond to the actual iterations. For the fast versions of the iterative reconstruction algorithms, the time to generate the system matrix once should be added if changing the system matrix, e.g., when changing the sampling parameters of the reconstructed image.

For this example the system matrix was modelled using discrete Radon transformation with linear interpolation, where the threshold γ was chosen to zero, hence the slow and the fast methods give exactly the same results. Note that the large difference in speedup between ART and EM/LSCG is due to the implementation of the discrete Radon transform is more efficient than the multiplication with the transpose of the system matrix. The slow methods can be accelerated some by implementing multiplication with the transpose of the system matrix (adjoint operator) as a backprojection integral, but note that this implies that the approximation of the system matrix will be different in the forward and the backprojection part. The sparse system matrix for this transformation geometry required approximately 13 MBytes of memory, and each iteration requires approximately one second.

Machine	Type	ART	EM	LSCG
Pentium	Fast	26 sec	17 sec	17 sec
	Slow	2218 sec	5776 sec	5250 sec
	Ratio	85	340	309
SGI Onyx	Fast	16 sec	16 sec	16 sec
	Slow	1306 sec	2722 sec	2715 sec
	Ratio	82	170	170

Table 9.2 Time usage for 20 iterations of EM and LSCG. For ART the time is for 20 full iterations, i.e., 20∗125∗101 of the iterations used in Eq. 9.39. The time measurements are for a sinogram with $125 * 101$ samples reconstructed into a $101 * 101$ samples image.

9.11.2 A Larger Reconstruction Example

In Fig. 9.9 a 2D sinogram with 281∗336 samples is shown, which was measured on a GE Advance PET scanner. The sinogram is reconstructed into a (large) image with 301∗301 samples. The system matrix has 94416∗90601 elements of which 0.23% are non-zero when modelling then system matrix using the Radon transform of a square. On the Onyx it required 1466 seconds to generate sparse matrix requiring 157 MBytes of memory, and each iteration of EM required 15 sec in the fast version and 11678 sec in the slow implementation. After 10 iterations the algorithm was stopped. The reconstructed image is shown in Fig. 9.10. For sake of comparison, the same sinogram reconstructed image using Filtered Backprojection is shown in Fig. 9.11, and this reconstruction used 6 seconds on the Onyx.

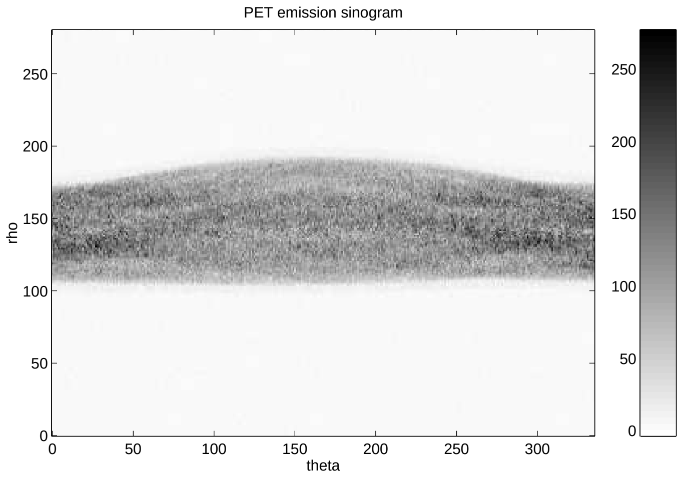

Figure 9.9 A slice of a 2D PET sinogram of a human brain.

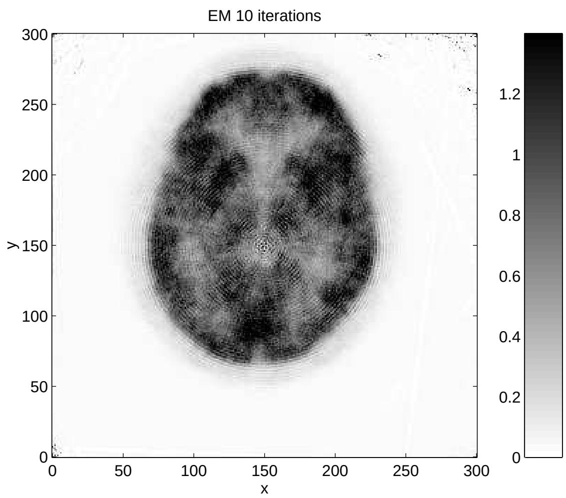

Figure 9.10 The reconstructed image after 10 iterations of EM.

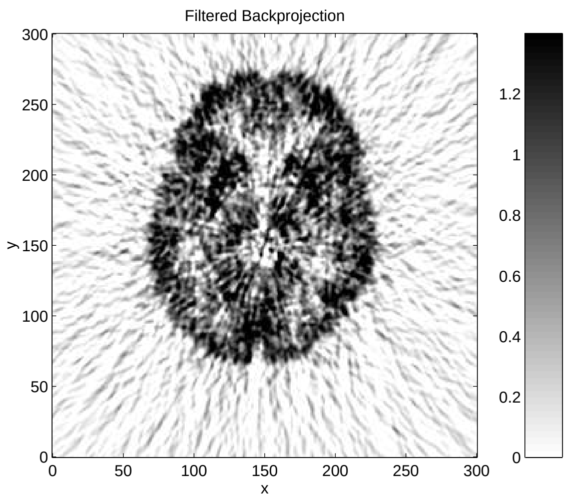

Figure 9.11 The reconstructed image using Filtered Backprojection with a ramp filter.

The front page of the thesis is generated from the reconstruction of 35 slices of 2D sinograms, as the one shown in Fig. 9.9. The reconstructed slices have been put into a single volume. The volume has been segmented using a marching cubes based program [119], which makes a iso-surface corresponding to a chosen threshold level. A 3D model of the reconstructed brain is available from [120].

9.11.3 Comparison with Different Noise Levels

In this subsection an (synthetic) image with $101 * 101$ pixels shown in Fig. 9.12 should be reconstructed from the noisy sinogram shown in Fig. 9.13. The sinogram has been generated by use of the program "RadonAna", described in Section C.1, where an additive uncorrelated Gaussian noise ($\sigma = 1$) term was added. Likewise has a sinogram been generated without the noise term, and one where the Gaussian deviance is 5 ($\sigma = 5$). It can very well be argued, that this noise term is very unphysical, due to (negative) sinogram values clearly lying outside of the head, but some of the correction algorithms used in practice can also produce artifacts like this.

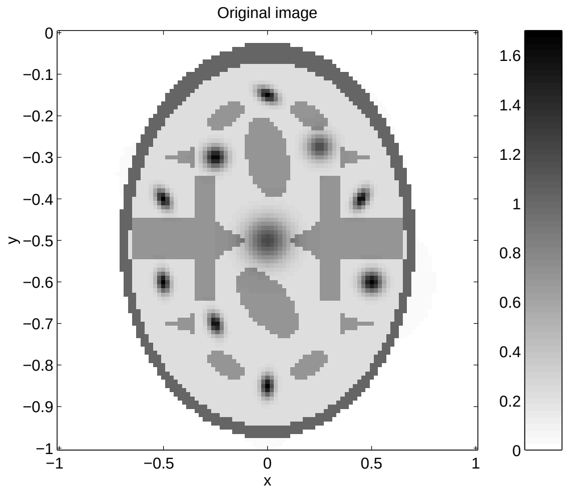

Figure 9.12 The original head phantom containing $101 * 101$ pixels.

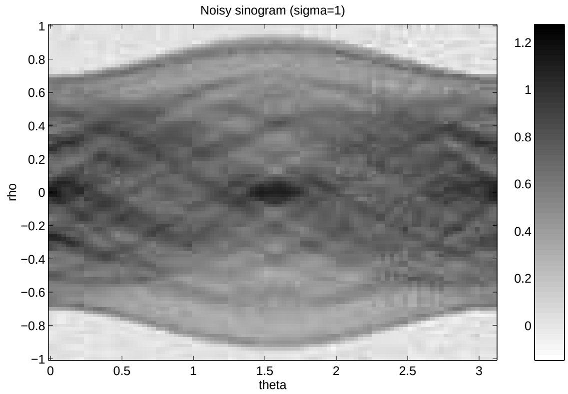

Figure 9.13 Noise corrupted sinogram with $100 * 101$ pixels.

In order to quantify the quality of images here it is chosen to use the L2 measure of misfit.

$$\mathrm{L2} = \frac{\|\boldsymbol{x}^* - \boldsymbol{x}_{ref}\|_2}{\|\boldsymbol{x}_{ref}\|_2} \tag{9.73}$$

where $\boldsymbol{x}^*$ is the estimated solution and $\boldsymbol{x}_{ref}$ is the true solution. It should also be mentioned that constraints have been used to generate the following figures. After each iteration the solution has been limited to the interval between 0 and 10.

In Fig. 9.14 ART, EM, and LSCG has been used for reconstruction of the sinogram without noise and the L2 measures of misfit have been shown. For ART, both cyclical and random selection of row index are shown. It can be seen that ART is very fast, and using random selection of row index ends up with the lowest error after three full iterations. The figure also shows that EM approximately gets to the same level, but so much slower. Here the LSCG algorithm does not do a good job, and more constraining or regularization is needed here. It can be mentioned that Filtered Backprojection gets an L2-measure of 0.266 (and only requiring less than one second).

Fig. 9.15 has the same setup as Fig. 9.14 for the noise corrupted sinogram with $\sigma = 1$. Here ART and LSCG reaches approximately the same error level (but ART is by far the fastest here), and EM clearly demonstrates a better performance. Note also that EM has a very flat plateau at the lowest error, which is very nice when designing criteria for selection of the number of iterations. For the same sinogram Filtered Backprojection gives L2=0.437, so EM is the better algorithm in this case, even though the error is not Poisson distributed.

Finally for the high noise level $\sigma = 5$, Fig. 9.16 shows a very different behavior for ART. It clearly diverges, but the constraints implies that the solution stays limited (but does not approximate the true solution). Both LSCG and EM converges in the first nine iterations, and ends up with L2=0.62 and L2=0.57 respectively and in this case Filtered Backprojection ends up with L2=1.75. But note that the comparison is somewhat harsh, because that the filter in Filtered Backprojection is a ramp filter amplifying the noise, and in general the sinogram could be low-pass filtered with lowered cutoff at high noise levels.

For the previous three figures the system matrix has been modelled using the Radon transform of a square, cf. Subsection B.3.2. Now the results are compared to another choice of model for the noisy sinogram $\sigma = 1$. In Fig. 9.17 the binary model found in Eq. 9.13 and the Radon transform

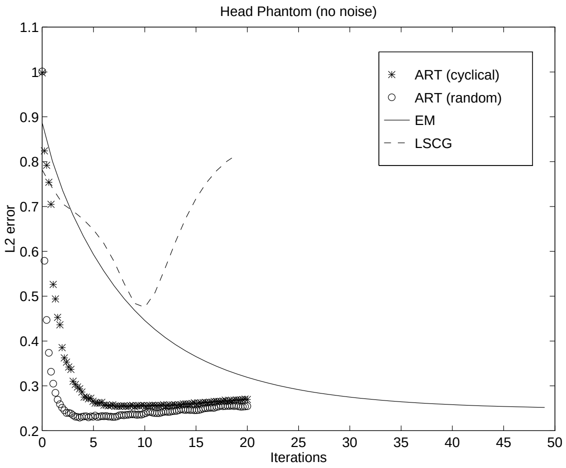

Figure 9.14 L2 measure of misfit as a function of number of full iterations when using ART, EM, and LSCG to reconstruct from the sinogram without noise. For ART both cyclical and random selection of row index are shown.

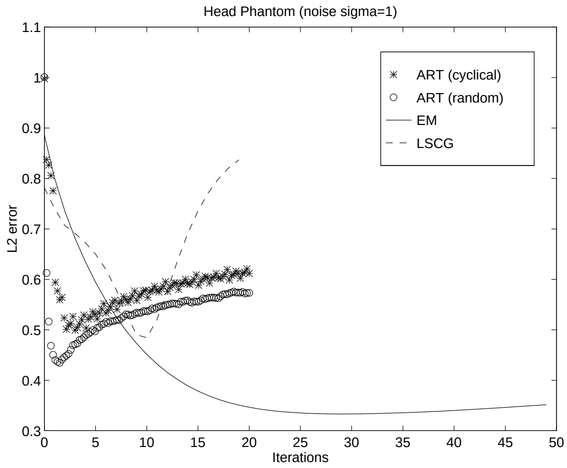

Figure 9.15 L2 measure of misfit as a function of number of full iterations when using ART, EM, and LSCG to reconstruct from the sinogram noisy sinogram ($\sigma = 1$).

of a square has been used to model the system matrix, and for ART is is apparently the simple binary model which provides the lowest error, but not much difference is found.

For EM, three models have been tried. The binary model, the Radon transform of a square, and the sinc-interpolation based model, cf. Eq. 9.15. For the sinc model a cutoff value $\gamma = 0.2\Delta x$ was used to limit the size of the sparse stored system matrix to 20 MBytes (lowering γ with an additional factor of two implied twice as much memory was needed). Fig. 9.18 shows that EM gets to a lower error when the model is improved, and the sinc interpolation scheme and the Radon transform of a square gives the same convergence until iteration 16, where the truncation of the matrix values implies that the two models split up in performance. For LSCG shown in Fig. 9.19 an odd convergence is found. The two models ends up with the same error, but after a very different number of iterations, but it is clear that LSCG can and should be improved.

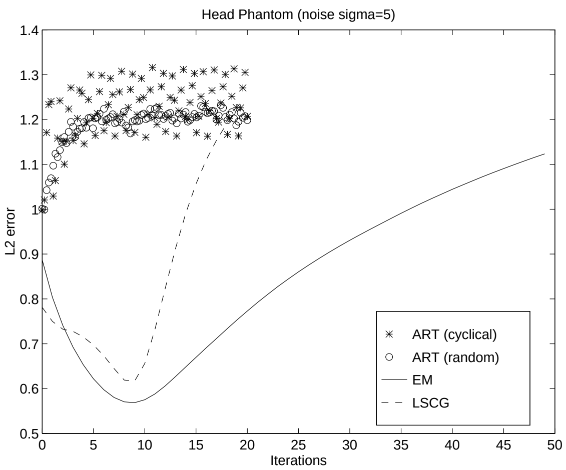

Figure 9.16 L2 as a function of iterations when using ART, EM, and LSCG ($\sigma = 5$).

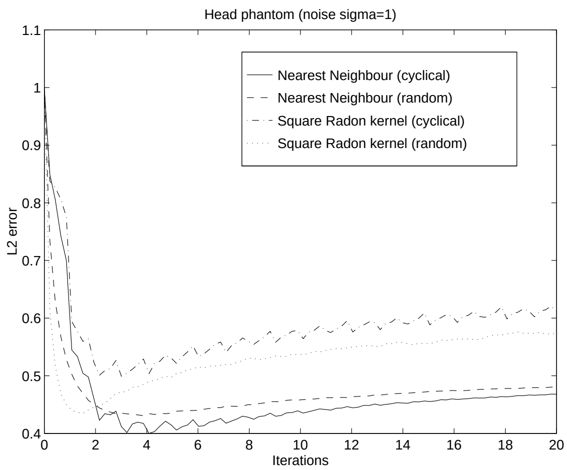

Figure 9.17 L2 as a function of iterations for ART for two models of the system matrix.

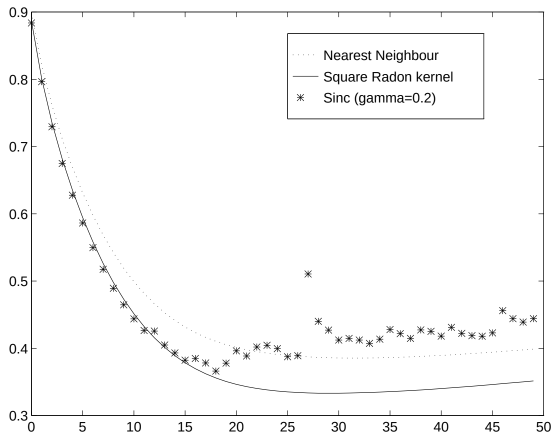

Figure 9.18 L2 as a function of iterations for EM for three models of the system matrix.

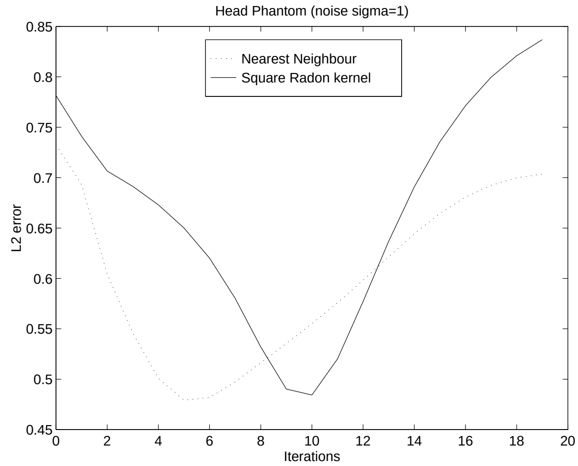

Figure 9.19 L2 as a function of iterations for LSCG for two models of the system matrix.

9.11.4 Reconstruction Using Constraints

In this example the EM and the LSCG algorithm has been to reconstruct images of $201 * 201$ samples from the sinogram shown in Fig. 9.9.

Figs. 9.20-9.25 first show the reconstructed images after 5, 10, and 15 iterations using EM with and without a simple constraints. In this case the constraints are implemented by placing an ellipse around the reconstructed head. After each iteration the values will be truncated to zero outside of the head. This is a very simple prior, which actually works out fine. The following figures use the same color scaling in a row, but not in a column. The color scaling has been truncated to see a reasonable image of the brain. Already after 5 iterations of EM is instability found in a single pixel originating from pixels and lines outside of the head. Fig. 9.22 uses a truncated color scaling, but a line can be seen to the right of the head, and Fig. 9.24 many artifacts are found in the image, which at some places have extremely high values. In the figures to the right the ellipse is clearly marked and no artifacts are found until after 10 iterations. The reconstructed images appears to be very nice and far better than the one shown in Fig. 9.11 where Filtered Backprojection was used. The intrinsic noise in the sinogram is not dominating in the reconstructed image, and edges appear to be more sharp, which is a feature that often is reported in the reconstruction literature.

Next reconstructed images using LSCG are shown in Figs. 9.26-9.31. Note that Figs. 9.26 and 9.27 show a very different result than Figs. 9.20 and 9.21, but after 10 iterations the reconstructed images from EM and LSCG look almost identical in the area of the brain. After 15 iterations LSCG shows signs of instability: Low-frequency components dominate the area outside of the brain, which is major source for the bad performance of LSCG showed in Section 9.11.3. But it is probably even worse that the values in the interior of the brain (the so called the white matter) are forced very close to zero. With even more iterations the reconstructed images gets worse.

This example clearly demonstrates the potential of the iterative methods compared to the (classical) direct reconstruction methods, but it is not a simple task to choose the number of iterations in a real-world case due to the numerous noise sources.

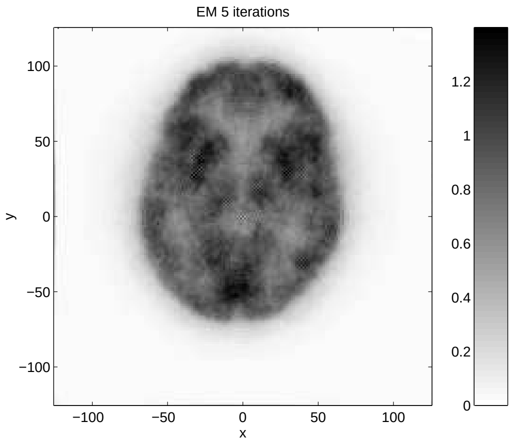

Figure 9.20 EM reconstructed image after 5 iterations.

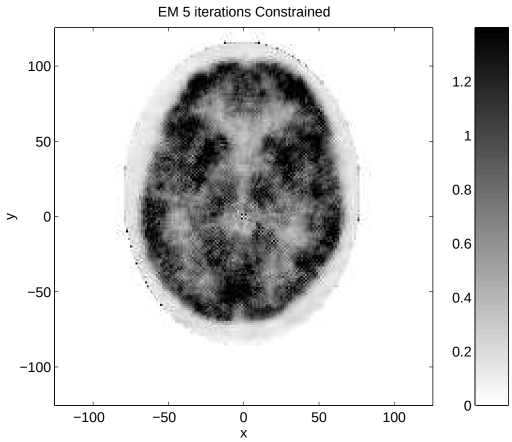

Figure 9.21 EM reconstructed image after 5 iteration with constraints.

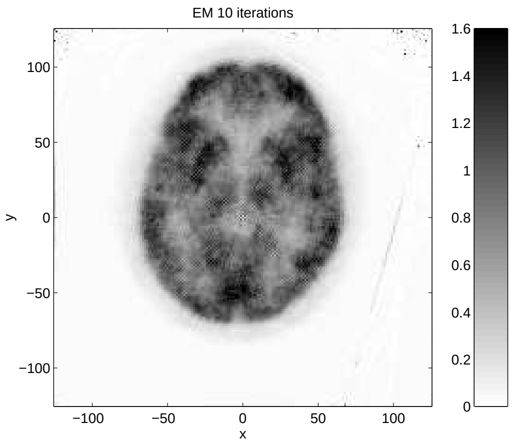

Figure 9.22 EM reconstructed image after 10 iterations.

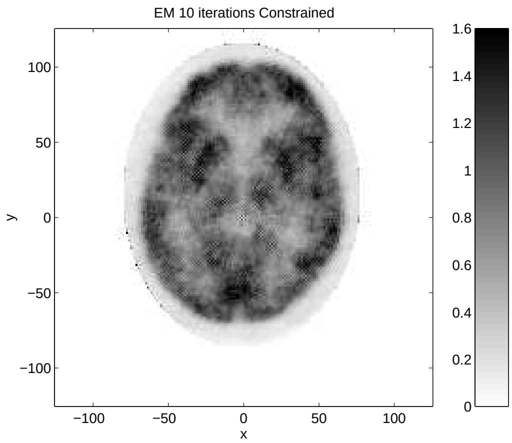

Figure 9.23 EM reconstructed image after 10 iteration with constraints.

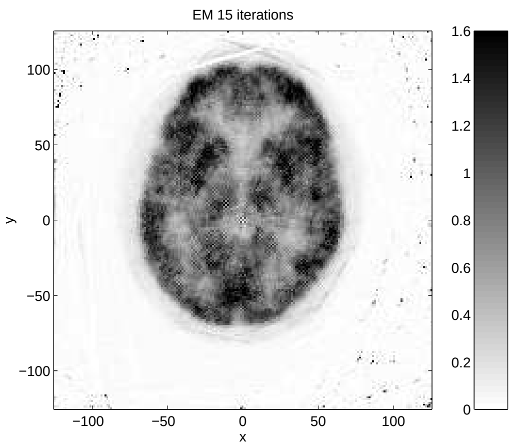

Figure 9.24 EM reconstructed image after 15 iterations.

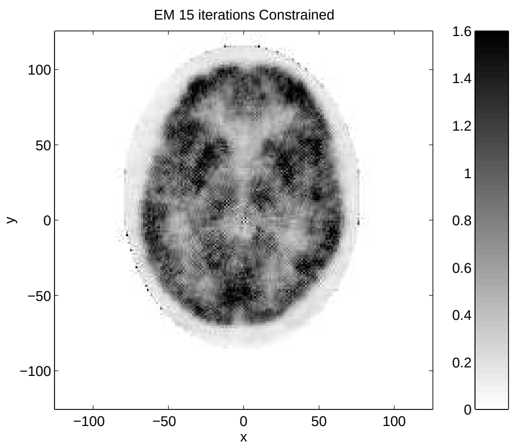

Figure 9.25 EM reconstructed image after 15 iteration with constraints.

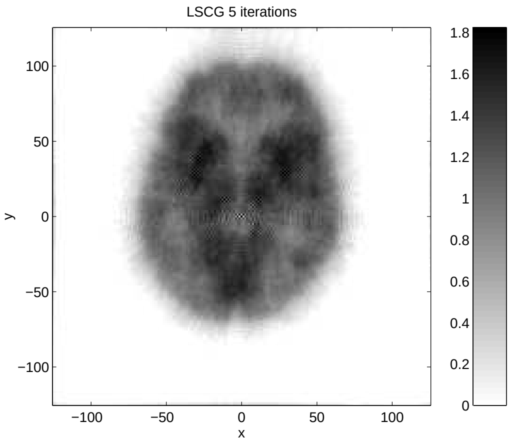

Figure 9.26 LSCG reconstructed image after 5 iterations.

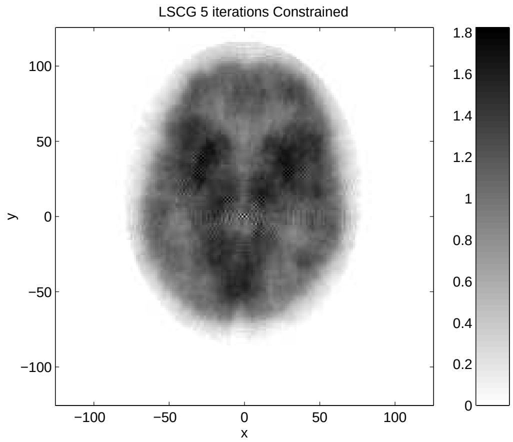

Figure 9.27 LSCG reconstructed image after 5 iteration with constraints.

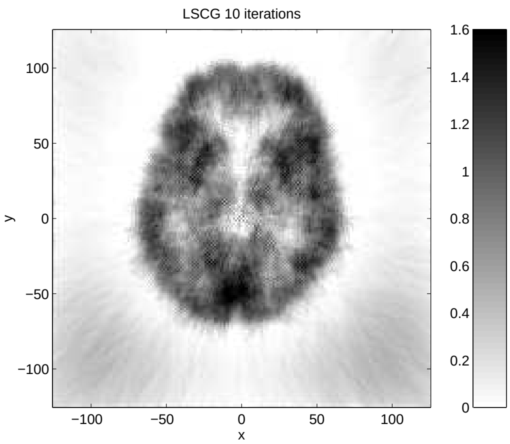

Figure 9.28 LSCG reconstructed image after 10 iterations.

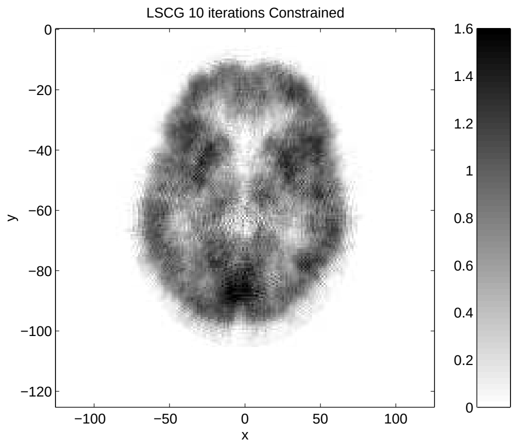

Figure 9.29 LSCG reconstructed image after 10 iteration with constraints.

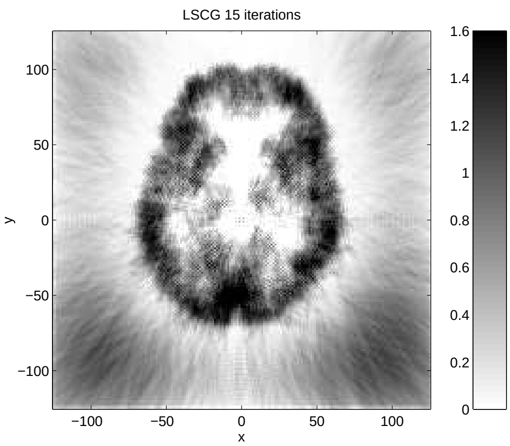

Figure 9.30 LSCG reconstructed image after 15 iterations.

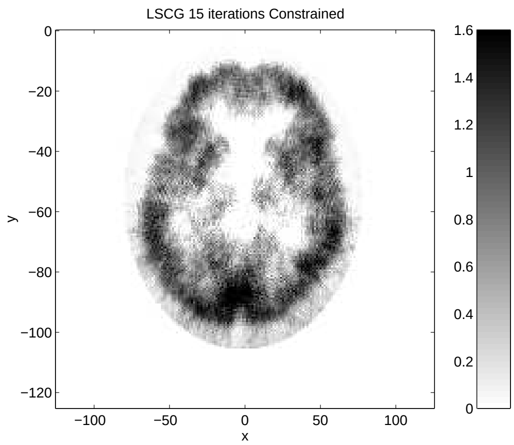

Figure 9.31 LSCG reconstructed image after 15 iteration with constraints.

9.11.5 Reconstruction Using Regularization

For the EM and the LSCG methods tests have been conducted using the Laplace operator as a regularizing operator L. For ART it can be found that the regularization term will add some (fixed) stability, but the regularization parameter λ is has no effect, due to the update scheme

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \frac{0 - \lambda \boldsymbol{l}_i^T \boldsymbol{x}^{(k)}}{\lambda \boldsymbol{l}_i^T \lambda \boldsymbol{l}_i} \lambda \boldsymbol{l}_i^T = \boldsymbol{x}^{(k)} - \frac{\boldsymbol{l}_i^T \boldsymbol{x}^{(k)}}{\boldsymbol{l}_i^T \boldsymbol{l}_i} \boldsymbol{l}_i^T \tag{9.74}$$

where $\boldsymbol{l}_i^T$ is certain row of the regularization matrix $\boldsymbol{L}$. The conclusion is that regularization of ART will have to be devised with another scheme.

In Fig. 9.32 is shown that EM responds very negatively to the extra regularization rows in the (expanded) system matrix. The sinogram is the same as used in Section 9.11.3 with the noise amplitude $\sigma = 2$. From the figure is can be seen that a non-zero λ actually implies a faster divergence and a larger error. This is due to the fact that the extra rows of the system matrix have a very different statistical nature compared to the original system matrix.

For LSCG with the same sinogram, as shown in Fig. 9.33, the regularization term will stabilize the solution, i.e., the change in error per iteration is minimized, but it does not lead to a better solution in terms of L2 error.

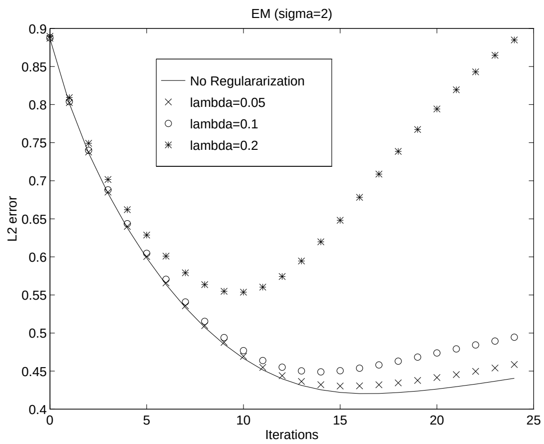

Figure 9.32 L2 measure as a function of the iteration number when reconstructing a noisy sinogram with EM and four different values of the regularization parameter λ.

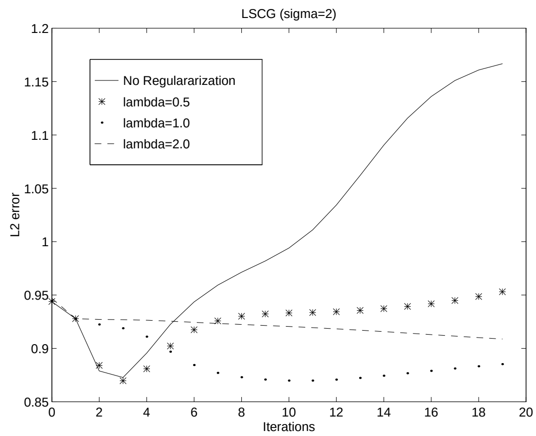

Figure 9.33 L2 measure as a function of the iteration number when reconstructing a noisy sinogram with LSCG and four different values of the regularization parameter λ.

9.12 Summary

This chapter demonstrated reconstruction methods based on linear algebra. It was shown that the system matrix can be modelled in different ways, here mostly shown from the Radon transform point of view.

The SVD was then used to indicate the structure of the singular values of the system matrix, but the SVD is very computationally demanding, hence it has currently a limited value for practical reconstruction purposes.

Some of the popular iterative reconstruction algorithms have been presented, and a very fast implementation of iterative reconstruction algorithms, comparable in speed to direct reconstruction methods, has been presented with theory and examples. The implementation is based on storing of the system matrix in fast memory using sparse techniques. The approach is mainly applicable to 2D reconstruction, due to the requirements of a sufficient amount of memory, but in principle the method can also be applied for 3D reconstruction.

Examples have been shown demonstrating how different modelling schemes of the system matrix influences on the solution, and the use of constraints and regularization have been shown in a number of examples. It is also clear the many open ends exist in this field, which the number of publications at conferences and relevant journals clearly demonstrate.

Chapter 10

The 3D Radon Transform for Lines

So far the reconstruction techniques have been related to 2D reconstruction. One of the problems with 2D PET scanners is the very low photon efficiency. The problem is that only a small fraction of photon pairs are actually detected on the same detector ring. One obvious possibility is to use a multi ring detector setup where photons pairs can be detected at different rings, as shown in Fig. 10.1. The basic physics is as described in Section 6.2, but now the photon pairs can form a line with (in principle) arbitrary orientation, which leads to a much higher rate of pairs of photons.

This chapter will first describe lines in 3D, which then is used to generalize the Radon transform for lines in 3D. Some direct reconstruction methods are also shown. The implementation of some of the elements in direct and iterative reconstruction algorithms will be described. A software package has been developed providing both direct and iterative reconstruction methods. The chapter will also present some examples to illustrate 3D reconstruction.

In recent years commercial 3D PET scanners have become available, such as the GE Advance PET scanner and the Siemens/CTI Ecat Exact scanners. Some theoretical issues are still not fully resolved [67] and some practical, such as attenuation correction, can still be improved in 3D scanners. Currently the attenuation correction is often based on 2D measurements.

In Chapter 2 it was shown that the 2D Radon transform used lines are the fundamental shape. The Radon transform is also defined in higher dimensions [14], and in 3D the fundamental shape is a plane (and not a line in the 3D space), hence the ordinary 3D Radon transform cannot be used. Appendix D describes some of the fundamental properties, along with some reconstruction methods, and a method for using these techniques to reconstruct volumes from line integrals is shown.

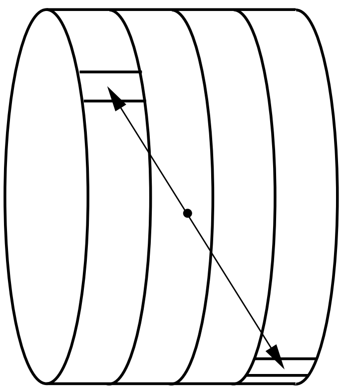

Figure 10.1 A multi ring PET scanner where pairs of photons can be detected at different rings.

10.1 Lines in a Three Dimensional Space

Lines in a three dimensional space can not be written in a single normal form as in the two dimensional case. Instead a parameter description can be used. In the following a line description using four parameters is shown. This description is the basis of direct inversion schemes. In general a line can be described by

$$\boldsymbol{r} = \boldsymbol{r}_0 + s\boldsymbol{\tau} \tag{10.1}$$

where s is the free parameter, and $\boldsymbol{r}_0$ is an offset vector. The vector $\boldsymbol{\tau}$ is a directional vector which can be normalized to unit length, i.e., $|\boldsymbol{\tau}| = 1$. The vector can, e.g., be described by

$$\boldsymbol{\tau} = \begin{pmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi \end{pmatrix} \tag{10.2}$$

The base point vector (or offset vector) $\boldsymbol{r}_0$ is described by two parameters u and v using two directional vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, both normalized to unit length.

$$\boldsymbol{r}_0 = u\boldsymbol{\alpha} + v\boldsymbol{\beta} \tag{10.3}$$

The coordinate system is shown in Fig. 10.2

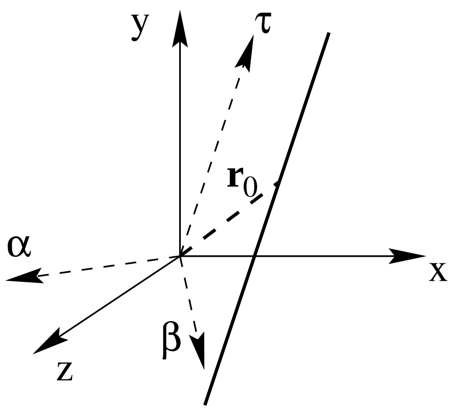

Figure 10.2 The (x, y, z) coordinate system and a line lying along the $\boldsymbol{\tau}$-axis. The base point vector $\boldsymbol{r}_0$ can be written as a linear combination of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

It can be chosen that the three vectors $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ form an orthogonal basis, and the last rotational degree of freedom can be used for specifying that the z-component of $\boldsymbol{\alpha}$ is zero. In [68], $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are defined as

$$\boldsymbol{\alpha} = \begin{pmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{pmatrix} \text{ and } \boldsymbol{\beta} = \begin{pmatrix} -\cos\theta\sin\phi \\ -\sin\theta\sin\phi \\ \cos\phi \end{pmatrix} \tag{10.4}$$

It should be mentioned that other symbols of the vectors can be found in the literature, e.g., [64, 65, 63].

Following [67], line integrals through a three dimensional space can be expressed by the symbols defined in Eqs. 10.1 and 10.3

$$\breve{g}(\theta,\phi,u,v) = \int_{-\infty}^{\infty} g(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta})\ ds \tag{10.5}$$

where (θ,ϕ,u,v) is the four-dimensional parameter domain and in PET the (measured) values are still named the sinogram, like in 2D.

The mapping from the (x,y,z) domain to (s,u,v) is very useful

$$\boldsymbol{r} = s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} \ \Rightarrow \tag{10.6}$$

$$\boldsymbol{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos\theta\cos\phi & -\sin\theta & -\cos\theta\sin\phi \\ \sin\theta\cos\phi & \cos\theta & -\sin\theta\sin\phi \\ \sin\phi & 0 & \cos\phi \end{pmatrix} \begin{pmatrix} s \\ u \\ v \end{pmatrix} = \boldsymbol{Q}\boldsymbol{p} \tag{10.7}$$

Using that the base vectors are orthogonal and normalized to unit length, hence the rotation matrix $\boldsymbol{Q}$ is unitary, i.e.,

$$\boldsymbol{p} = \boldsymbol{Q}^{-1}\boldsymbol{r} = \boldsymbol{Q}^{T}\boldsymbol{r} \tag{10.8}$$

One feature of the matrix $\boldsymbol{Q}$ is that only two angles are used compared to a general rotation matrix which uses three degrees of freedom, i.e., three angles. The definition of the line integrals in Eq. 10.5 implies that a three dimensional function $g(x,y,z)$ is transformed into a four dimensional parameter domain $\breve{g}(\theta,\phi,u,v)$. This change in dimensions will naturally impose some inherent difficulties to be analyzed later.

It has been chosen to denote the line integrals of $g(\boldsymbol{r})$ with the symbol $\breve{g}$. It can be argued that Eq. 10.5 is not a Radon transform of the function $g(x,y,z)$. It is not, cf. Eq. D.1, a three dimensional Radon transform nor a four dimensional, but it can be seen as a hybrid or generalized Radon transform for lines through a three dimensional space.

In the following, Eq. 10.5 is called the 3D line Radon transform or simply the Radon transform. In the rest of this chapter only the 3D line Radon definition will be considered, and the actual parameters will (as previously) uniquely determine which type of transformation is used. From the definition of the 3D line Radon transform some basic rules are derived and shown in Appendix E. Additionally, the Radon transform of simple geometrical functions are given in the same appendix.

The 3D line Radon transform can be perceived as a convolution between a function $g(\boldsymbol{r})$ and a kernel h expressed in the coordinates $\boldsymbol{p}$ for a given combination of the angles θ and ϕ.

$$h(\boldsymbol{r}) = h(\boldsymbol{p}) = h(s,u,v) = \delta(u)\delta(v) \tag{10.9}$$

which can be seen from the following, where $***$ implies a three dimensional convolution in the parameters (s,u,v).

$$\begin{aligned} \breve{g}(\theta,\phi,u,v) &= g(\boldsymbol{r}) *** \delta(u)\delta(v) && (10.10) \\ &= \int_{v'=-\infty}^{\infty}\int_{u'=-\infty}^{\infty}\int_{s'=-\infty}^{\infty} g(s'\boldsymbol{\tau} + u'\boldsymbol{\alpha} + v'\boldsymbol{\beta})\ \delta(u-u')\delta(v-v')\ ds'\ du'\ dv' \\ &= \int_{s'=-\infty}^{\infty} g(s'\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta})\ ds' && (10.11) \end{aligned}$$

Besides the new integration variable s', the result matches Eq. 10.5.

At this time it could be appropriate to look back to the two dimensional results. This can be done by choosing $\phi = 0$, $v = z_0$ and $u = \rho$, thus the line integral reduces to

$$\breve{g}(\rho, 0, \theta) = \int_{-\infty}^{\infty} g(s\cos\theta - \rho\sin\theta, s\sin\theta + \rho\cos\theta, z_0)\ ds \tag{10.12}$$

From Eq. 2.8, this can be recognized as the ordinary two dimensional Radon transform of the function in the plane $z = z_0$, where the angular parameter θ has been rotated 90 degrees compared to the definition of the two dimensional Radon transform.

10.1.1 Limiting the 3D line parameters

The angular parameters θ and ϕ control the orientation of the line and it is obvious that they can be bounded as normal spherical angles, i.e., $0 \leq \theta < 2\pi$ and $-\pi/2 \leq \phi < \pi/2$. This corresponds to a full angular coverage, but the interval of the θ-parameter can further be restricted to $0 \leq \theta < \pi$, due to Eqs. 10.3 and 10.4

$$\breve{g}(\theta, \phi, u, v) = \breve{g}(\theta + \pi, -\phi, -u, v) \tag{10.13}$$

This little trick will enable a reduction of the parameter domain with a factor of two. The trick is not used in [67] and [66], hence the filters found in these papers will differ with a constant (a factor of two) from those presented in the following. This problem relates to the difference between the adjoint and the backprojection operator, which for 2D was mentioned in Section 9.3.

The reconstruction geometry will now be based on truncated elliptical tubes, relevant to multi ring PET scanners, where the geometry Ω_ψ is defined as

$$\Omega_\psi = \{0 \leq \theta < \pi \ \vee\ |\phi| \leq \psi\} \tag{10.14}$$

where ψ is called the axial acceptance angle. The geometry, where $\psi = 0$, i.e., Ω_0 corresponds to the two dimensional case treated in the previous chapters (see also Eq. 10.12), and $\Omega_{\pi/2}$ is full angular coverage. Actually Ω_ψ is of interest due to the possibility to derive direct reconstruction formulas, but actually 3D PET scanners do not measure in this geometry - some of the line orientations are missing. This aspect will be covered in Section 10.6.

The two remaining parameters u and v are translation parameters controlling the position of the line in a plane perpendicular to the directional vector $\boldsymbol{\tau}$. Theoretically u and v, can only be bounded if it can be assumed that the scanned object is finite. The origin of the three dimensional coordinate system is placed (approximatively) at the center of the object. This implies that the distance from the origin to the outer edge of the object is minimized to $r_{\max}$, or in mathematical terms

$$g(x, y, z) = 0 \text{ for } x^2 + y^2 + z^2 > r_{\max}^2 \tag{10.15}$$

If $r_{\max}$ is found then u and v can be restricted, because the distance from the origin to the volume element at $\boldsymbol{r} = s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}$ are $|\boldsymbol{r}| = \sqrt{s^2 + u^2 + v^2}$. This is due to othonormality of the three expansion vectors. Assuming that $u^2 + v^2 > r_{\max}^2$, implies that the integral is zero, because the function g is zero along the line. This result implies that each of the translation parameters can be bounded as

$$-r_{\max} \leq u \leq r_{\max} \ \text{ and } \ -r_{\max} \leq v \leq r_{\max} \tag{10.16}$$

10.2 Fourier Slice Reconstruction in 3D

From the line integrals defined in Eq. 10.5, the function $g(\boldsymbol{r})$ can be recovered using Fourier techniques, and the Fourier Slice Theorem is now derived for 3D line integrals by applying the two dimensional Fourier transform to each of the (u, v)-planes in Eq. 10.5.

$$\check{G}(\theta,\phi,\nu_u,\nu_v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \check{g}(\theta,\phi,u,v)\ e^{-j2\pi(u\nu_u+v\nu_v)}\ du\ dv \tag{10.17}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(s\boldsymbol{\tau}+u\boldsymbol{\alpha}+v\boldsymbol{\beta})\ e^{-j2\pi(u\nu_u+v\nu_v)}\ du\ dv\ ds \tag{10.18}$$

which indicates a close connection to the three dimensional Fourier transform of the function $g(\boldsymbol{r})$.

$$G(\boldsymbol{\nu}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(\boldsymbol{r})\ e^{-j2\pi\boldsymbol{r}\cdot\boldsymbol{\nu}}\ d\boldsymbol{r} \tag{10.19}$$

$$g(\boldsymbol{r}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(\boldsymbol{\nu})\ e^{j2\pi\boldsymbol{r}\cdot\boldsymbol{\nu}}\ d\boldsymbol{\nu} \tag{10.20}$$

where $\boldsymbol{\nu}$ is the three dimensional frequency vector.

Now the integration parameters in Eq. 10.18 is changed into x, y and z, i.e., $\boldsymbol{r}$. It is used that $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$ are orthogonal.

$$\boldsymbol{r} = s\boldsymbol{\tau}+u\boldsymbol{\alpha}+v\boldsymbol{\beta}\ \Rightarrow \tag{10.21}$$

$$u = \boldsymbol{\alpha}\cdot\boldsymbol{r}\ \text{ and }\ v=\boldsymbol{\beta}\cdot\boldsymbol{r}\ \Rightarrow \tag{10.22}$$

$$u\nu_u+v\nu_v = \boldsymbol{r}\cdot(\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta}) \Rightarrow \tag{10.23}$$

$$\check{G}(\theta,\phi,\nu_u,\nu_v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(\boldsymbol{r})\ e^{-j2\pi\boldsymbol{r}\cdot(\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta})}\ d\boldsymbol{r} \tag{10.24}$$

This interesting result shows that the two dimensional Fourier transform of the line integrals is the three dimensional Fourier transform of the function to be reconstructed [67].

$$\check{G}(\theta,\phi,\nu_u,\nu_v) = G(\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta}) \tag{10.25}$$

which is a Fourier Slice Theorem for line integrals in a three dimensional space. It implies that the function $g(\boldsymbol{r})$ can be recovered by applying a two dimensional Fourier transform to the sinogram for all values of (θ, ϕ), followed by a mapping of the spectrum, and finally recovering the desired volume by using a three dimensional inverse Fourier transform.

$$G(\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \check{g}(\theta,\phi,u,v)\ e^{-j2\pi(u\nu_u+v\nu_v)}\ du\ dv \tag{10.26}$$

$$g(\boldsymbol{r}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(\boldsymbol{\nu})\ e^{j2\pi\boldsymbol{r}\cdot\boldsymbol{\nu}}\ d\boldsymbol{\nu} \tag{10.27}$$

A non-trivial problem arises when implementing the mapping of the spectrum. Each frequency point $\boldsymbol{\nu}$ is mapped into $\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta}$, but $\boldsymbol{\nu}$ is a three-dimensional vector and $\nu_u\boldsymbol{\alpha}+\nu_v\boldsymbol{\beta}$ has four degrees of freedom. This implies that each $\boldsymbol{\nu}$ matches an infinite set of parameters (θ, ϕ, u, v). One possible solution is to use weighted averages of the possible 4D frequency vectors for each value of $\boldsymbol{\nu}$, and this problem is still an area of active research, though in [69] several aspects have been covered.

10.3 Backprojection Based Inversion of Line Integrals in 3D

Analogous to the 2D Filtered Backprojection, it is possible to filter the line projections and then make a backprojection of the sinogram into the volume domain (x, y, z) [67]. The backprojection operator in 3D is now analyzed from the transformation of a point source. Again, it is used that any function can be resolved into a weighted sum (integral) of delta functions.

$$g(\boldsymbol{r}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(\boldsymbol{r}_0)\ \delta(x-x_0)\delta(y-y_0)\delta(z-z_0)\ dx_0\ dy_0\ dz_0 \tag{10.28}$$

$$\equiv \int_{-\infty}^{\infty} g(\boldsymbol{r}_0)\ \delta(\boldsymbol{r}-\boldsymbol{r}_0)\ d\boldsymbol{r}_0 \tag{10.29}$$

This implies that the corresponding Radon transform is given by

$$\breve{g}(\theta,\phi,u,v) = \int_{-\infty}^{\infty} g(\boldsymbol{r}_0)\left(\int_{s=-\infty}^{\infty}\delta(s\boldsymbol{\tau}+u\boldsymbol{\alpha}+v\boldsymbol{\beta}-\boldsymbol{r}_0)\ ds\right) d\boldsymbol{r}_0 \tag{10.30}$$

This shows that any function $g(\boldsymbol{r})$ can be Radon transformed, if the point source can be transformed. Now this will be done.

$$g_p(\boldsymbol{r}) = \delta(\boldsymbol{r}-\boldsymbol{r}_0) \quad \Rightarrow \tag{10.31}$$

$$\breve{g}_p(\theta,\phi,u,v) = \int_{-\infty}^{\infty}\delta(s\boldsymbol{\tau}+u\boldsymbol{\alpha}+v\boldsymbol{\beta}-\boldsymbol{r}_0)\ ds \tag{10.32}$$

$$= \int_{-\infty}^{\infty}\delta((s-s_0)\boldsymbol{\tau}+(u-u_0)\boldsymbol{\alpha}+(v-v_0)\boldsymbol{\beta})\ ds \tag{10.33}$$

$$= \int_{-\infty}^{\infty}\delta(\tilde{s}\boldsymbol{\tau}+(u-u_0)\boldsymbol{\alpha}+(v-v_0)\boldsymbol{\beta})\ d\tilde{s} \tag{10.34}$$

Here the unambiguous substitution $\boldsymbol{r}_0 = s_0\boldsymbol{\tau} + u_0\boldsymbol{\alpha} + v_0\boldsymbol{\beta}$ has been used. Using that the delta function will be non-zero if and only if the argument is a zero length vector implies that the result will be non-zero only if $u = u_0 = \boldsymbol{r}_0\cdot\boldsymbol{\alpha}$ and $v = v_0 = \boldsymbol{r}_0\cdot\boldsymbol{\beta}$, and because the base vectors are orthogonal the result can be expressed using the delta function.

$$\breve{g}_p(\theta,\phi,u,v) = \delta(u-u_0)\delta(v-v_0) = \delta(u-\boldsymbol{r}_0\cdot\boldsymbol{\alpha})\delta(v-\boldsymbol{r}_0\cdot\boldsymbol{\beta}) \tag{10.35}$$

This can also be found directly from Eq. 10.10. It is a very important result which can be used to formulate a backprojection operator in 3D [67].

$$\hat{g} = \int_{\Omega}\breve{g}(\theta,\phi,u=\boldsymbol{r}\cdot\boldsymbol{\alpha},v=\boldsymbol{r}\cdot\boldsymbol{\beta})\ d\Omega = \int_{\theta=0}^{\pi}\int_{\phi=-\psi}^{\psi}\breve{g}(\theta,\phi,u=\boldsymbol{r}\cdot\boldsymbol{\alpha},v=\boldsymbol{r}\cdot\boldsymbol{\beta})\ \cos\phi\ d\phi\ d\theta \tag{10.36}$$

For convenience the integration over angles is written as a single integral with index Ω. If the geometry is, e.g., Ω_ψ the integration becomes the last part of Eq. 10.36, where the term $\cos\phi$ is the Jacobian, found when converting to spherical coordinates.

10.4 Filtering after Backprojection of Line Integrals in 3D

Like in two dimensional backprojection algorithms, a high pass filter is needed either before or after the backprojection [67]. In this section the aim is to find a condition for the filter used after backprojection, i.e., on the volume. It can be derived by backprojecting the Radon transform of a function g. Inserting Eq. 10.5 in 10.36 gives

$$\hat{g}(\boldsymbol{r}) = \int_\Omega \int_{s=-\infty}^{\infty} g((\boldsymbol{r}\cdot\boldsymbol{\alpha})\boldsymbol{\alpha} + (\boldsymbol{r}\cdot\boldsymbol{\beta})\boldsymbol{\beta} + s\boldsymbol{\tau})\ ds\ d\Omega \tag{10.37}$$

Now it is utilized that the s-integration can be shifted along the $\boldsymbol{\tau}$-axis, like it was done from Eq. 10.33 to 10.34. Here the offset is chosen to $\boldsymbol{r}\cdot\boldsymbol{\tau}$.

$$\hat{g}(\boldsymbol{r}) = \int_\Omega \int_{s=-\infty}^{\infty} g((\boldsymbol{r}\cdot\boldsymbol{\alpha})\boldsymbol{\alpha} + (\boldsymbol{r}\cdot\boldsymbol{\beta})\boldsymbol{\beta} + (\boldsymbol{r}\cdot\boldsymbol{\tau})\boldsymbol{\tau} + s\boldsymbol{\tau})\ ds\ d\Omega \tag{10.38}$$

$$= \int_\Omega \int_{s=-\infty}^{\infty} g(\boldsymbol{r} + s\boldsymbol{\tau})\ ds\ d\Omega \tag{10.39}$$

which can be recognized to be a convolution, thus a 3D Fourier transform can be applied on both sides.

$$\tilde{G}(\boldsymbol{\nu}) = \int_\Omega \int_{s=-\infty}^{\infty} \int_{\boldsymbol{r}=-\infty}^{\infty} g(\boldsymbol{r} + s\boldsymbol{\tau})\ e^{-j2\pi\boldsymbol{\nu}\cdot\boldsymbol{r}}\ d\boldsymbol{r}\ ds\ d\Omega \tag{10.40}$$

$$= \int_\Omega \int_{s=-\infty}^{\infty} \int_{\tilde{\boldsymbol{r}}=-\infty}^{\infty} g(\tilde{\boldsymbol{r}})\ e^{-j2\pi\boldsymbol{\nu}\cdot(\tilde{\boldsymbol{r}}-s\boldsymbol{\tau})}\ d\tilde{\boldsymbol{r}}\ ds\ d\Omega \tag{10.41}$$

$$= \int_\Omega \int_{s=-\infty}^{\infty} G(\boldsymbol{\nu})\ e^{j2\pi s\boldsymbol{\nu}\cdot\boldsymbol{\tau}}\ ds\ d\Omega \tag{10.42}$$

$$= \int_\Omega G(\boldsymbol{\nu}) \left(\int_{s=-\infty}^{\infty} e^{j2\pi s\boldsymbol{\nu}\cdot\boldsymbol{\tau}}\ ds \right) d\Omega \tag{10.43}$$

$$= G(\boldsymbol{\nu}) \int_\Omega \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega \equiv \frac{G(\boldsymbol{\nu})}{H_a(\boldsymbol{\nu})} \tag{10.44}$$

This result is a 3D Filtering after Backprojection reconstruction method. The spectrum of the backprojected sinogram is multiplied with the filter $H_a(\boldsymbol{\nu})$, shown in Eq. 10.45. Finally Eq. 10.20 is used recover the desired volume.

$$H_a(\boldsymbol{\nu}) = \frac{1}{\int_\Omega \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega} \tag{10.45}$$

In the geometry Ω_ψ, the filter can also be found on an analytical form. Several papers, e.g., [66, 67], have shown different filters, which again have been shown to only differ with a normalization constant. Due to the circular symmetry around the z-axis the filter can be calculated, with ν_y, i.e., the y-component of the frequency vector, set to zero.

$$\int_\Omega \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega = \int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi} \delta(\nu_x\cos\theta\cos\phi + \nu_z\sin\phi)\cos\phi\ d\theta\ d\phi \tag{10.46}$$

$$= 2\int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi/2} \delta(\nu_x\cos\theta + \nu_z\tan\phi)\ d\theta\ d\phi \tag{10.47}$$

$$= 2\int_{\phi=-\gamma}^{\gamma} \frac{1}{|\nu_x\sin\theta_0|}\ d\phi \text{ where } \begin{cases} \gamma = \min\{\psi, |\arctan(\nu_x/\nu_z)|\} \\ \nu_x\cos\theta_0 + \nu_z\tan\phi = 0 \end{cases} \tag{10.48}$$

$$= 2\int_{\phi=-\gamma}^{\gamma} \frac{\cos\phi}{\sqrt{\nu_x^2 - (\nu_x^2+\nu_z^2)\sin^2\phi}}\ d\phi \tag{10.49}$$

The symmetry of the integral is used again to get the general result, i.e., in the last line $|\nu_x|$ is substituted back to $\sqrt{\nu_x^2+\nu_y^2}$, and the result is

$$H_a(\boldsymbol{\nu}) = \begin{cases} \dfrac{\boldsymbol{\nu}}{\pi} & \text{if } \sqrt{\nu_x^2+\nu_y^2} < |\boldsymbol{\nu}|\sin\psi \\ \boldsymbol{\nu}\left(2\arcsin\left(\frac{|\boldsymbol{\nu}|}{\sqrt{\nu_x^2+\nu_y^2}}\sin\psi\right)\right)^{-1} & \text{otherwise} \end{cases} \tag{10.50}$$

Two things can be noted in Eq. 10.50. First, letting $\psi \to \pi/2$ implies full angular coverage and Eq. 10.50 becomes very easy, i.e., $H_a(\boldsymbol{\nu}) = \boldsymbol{\nu}/\pi$. On the other hand letting $\psi \to 0$ implies that the filter becomes close to $H_a(\boldsymbol{\nu}) = \sqrt{\nu_x^2+\nu_y^2}/(2\psi)$. If neglecting the normalization constant 2ψ the filter can be recognized as the filter used in 2D Filtering after Backprojection, cf. Eq. 7.24. In Fig. 10.3 is shown the angular dependence of the filter from Eq. 10.50. The filter has been multiplied by $\psi/|\boldsymbol{\nu}|$ and on the first axis is shown the angle $\phi_{\boldsymbol{\nu}} = \arcsin(\nu_z/|\boldsymbol{\nu}|)$.

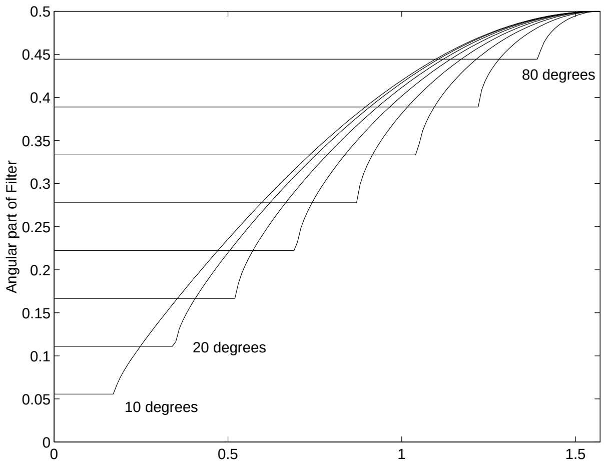

Figure 10.3 Normalized angular part of the filter as a function of $\phi_{\boldsymbol{\nu}} = \arcsin(\nu_z/|\boldsymbol{\nu}|)$ for $\psi = 10°, 20°, \ldots, 80°$.

10.5 Filtered Backprojection of Line Integrals in 3D

Analogous to the two dimensional Filtered Backprojection method, the filtering can be done before the backprojection of the line integrals in 3D. Here a two dimensional filter $h_b(\theta,\phi,u,v)$ is convolved in the (u,v)-plane of the sinogram for each value of (θ,ϕ). After the filtering shown in Eq. 10.51, the backprojection operator is used as shown in Eq. 10.53.

$$\bar{\breve{g}}(\theta,\phi,u,v) = h_b(\theta,\phi,u,v) **\breve{g}(\theta,\phi,u,v) \tag{10.51}$$

$$= \int_{u'=-\infty}^{\infty}\int_{v'=-\infty}^{\infty} h_b(\theta,\phi,u-u',v-v')\ \breve{g}(\theta,\phi,u',v')\ du'\ dv' \tag{10.52}$$

$$g(\boldsymbol{r}) = \int_\Omega \bar{\breve{g}}(\theta,\phi,\boldsymbol{r}\cdot\boldsymbol{\alpha},\boldsymbol{r}\cdot\boldsymbol{\beta})\ d\Omega \tag{10.53}$$

The criterion that the filter in 3D Filtered Backprojection will have to satisfy can be derived by choosing $g(\boldsymbol{r})$ as a point source and requiring that Eqs. 10.51 and 10.53 are self-consistent, i.e.,

$$g(\boldsymbol{r}) = \delta(\boldsymbol{r}) \;\Rightarrow\; \breve{g}(\theta,\phi,u,v) = \delta(u)\delta(v) \;\Rightarrow \tag{10.54}$$

$$\delta(\boldsymbol{r}) = \int_\Omega h_b(\theta,\phi,\boldsymbol{r}\cdot\boldsymbol{\alpha},\boldsymbol{r}\cdot\boldsymbol{\beta})\; d\Omega \tag{10.55}$$

The last equation can also be viewed in the 3D Fourier domain

$$1 = \int_\Omega \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_b(\theta,\phi,\boldsymbol{r}\cdot\boldsymbol{\alpha},\boldsymbol{r}\cdot\boldsymbol{\beta})\; e^{-j2\pi\boldsymbol{\nu}\cdot\boldsymbol{r}}\; d\boldsymbol{r}\right)\; d\Omega \tag{10.56}$$

$$= \int_\Omega \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_b(\theta,\phi,u,v)\; e^{-j2\pi(s\boldsymbol{\nu}\cdot\boldsymbol{\tau}+u\boldsymbol{\nu}\cdot\boldsymbol{\alpha}+v\boldsymbol{\nu}\cdot\boldsymbol{\beta})}\; d\boldsymbol{p}\right)\; d\Omega \tag{10.57}$$

$$= \int_\Omega H_b(\theta,\phi,\boldsymbol{\nu}\cdot\boldsymbol{\alpha},\boldsymbol{\nu}\cdot\boldsymbol{\beta}) \int_{-\infty}^{\infty} e^{-j2\pi s\boldsymbol{\nu}\cdot\boldsymbol{\tau}}\; ds\; d\Omega \tag{10.58}$$

$$= \int_\Omega H_b(\theta,\phi,\boldsymbol{\nu}\cdot\boldsymbol{\alpha},\boldsymbol{\nu}\cdot\boldsymbol{\beta})\; \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\; d\Omega \tag{10.59}$$

where the 2D Fourier transform of the filter has been used

$$H_b(\theta,\phi,\nu_u,\nu_v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_b(\theta,\phi,u,v) e^{-j2\pi(u\nu_u+v\nu_v)} du\; dv \tag{10.60}$$

In a given geometry, several filters are valid [67], due to the 4D to 3D transformation during reconstruction. In general, part of the filters belong to a null-space, which could be used to improve noise performance without altering the signal reconstruction, though more research is needed to derive appropriate filters.

In [67] a criterion is given which makes Filtered Backprojection in a sense equivalent to Filtering after Backprojection.

$$H_b(\nu_u,\nu_v) = H_a(\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}) \tag{10.61}$$

Defining $\nu_x = \nu_u\alpha_x + \nu_v\beta_x$ and $\nu_y = \nu_u\alpha_y + \nu_v\beta_y$, then Eqs. 10.50 and 10.61 imply that one valid filter is

$$H_b(\nu_u,\nu_v) = \begin{cases} \frac{\sqrt{\nu_u^2+\nu_v^2}}{\pi} & \text{if } \sqrt{\nu_x^2+\nu_y^2} < |\boldsymbol{\nu}|\sin\psi \\ \dfrac{\sqrt{\nu_u^2+\nu_v^2}}{2\arcsin\left(\dfrac{\sqrt{\nu_u^2+\nu_v^2}}{\sqrt{\nu_x^2+\nu_y^2}}\sin\psi\right)} & \text{otherwise} \end{cases} \tag{10.62}$$

In the geometry $\Omega_{\pi/2}$, the filter is very simple

$$H_b(\theta,\phi,\nu_u,\nu_v) = \frac{1}{\pi}\sqrt{\nu_u^2+\nu_v^2} \tag{10.63}$$

The $\Omega_{\pi/2}$-filter can be validated by inserting Eq. 10.63 in 10.59.

$$\int_{\Omega_{\pi/2}} H_b(\theta, \phi, \boldsymbol{\nu}\cdot\boldsymbol{\alpha}, \boldsymbol{\nu}\cdot\boldsymbol{\beta})\ \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega \tag{10.64}$$

$$= \frac{1}{\pi}\int_{\Omega_{\pi/2}} \sqrt{(\boldsymbol{\nu}\cdot\boldsymbol{\alpha})^2 + (\boldsymbol{\nu}\cdot\boldsymbol{\beta})^2}\ \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega \tag{10.65}$$

$$= \frac{1}{\pi}\int_{\Omega_{\pi/2}} \sqrt{(\boldsymbol{\nu}\cdot\boldsymbol{\alpha})^2 + (\boldsymbol{\nu}\cdot\boldsymbol{\beta})^2 + (\boldsymbol{\nu}\cdot\boldsymbol{\tau})^2}\ \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega \tag{10.66}$$

$$= \frac{1}{\pi}\int_{\Omega_{\pi/2}} |\boldsymbol{\nu}|\ \delta(\boldsymbol{\nu}\cdot\boldsymbol{\tau})\ d\Omega = 1 \tag{10.67}$$

In the last line the result from Eqs. 10.46-10.50 are used, with $\psi = \pi/2$.

10.6 Reconstruction Scheme for 3D Multi Ring PET Scanners

One of the fundamental problems in 3D PET reconstruction is that the Ω_ψ geometry defined in Eq. 10.14 does not perfectly match the measuring situation with multi ring PET scanners. In order to detect many events (detection of paired photons) a large scanner is needed, which leads to a high value of ψ, but it can be shown, e.g., [70] that a large value of ψ implies that scanners will lack some areas of the sinogram. The relative importance of this problem increases with ψ.

In the very famous article [70] by Kinahan, Harrop and Rogers, a three-stage process is proposed, namely that a crude volume, actually a set of axial slices, is reconstructed from the events lying (approximately) in the same axial plane, cf. Eq. 10.12, using 2D Filtered Backprojection. From this volume of reconstructed slices, the missing parts of the four-dimensional sinogram can be generated using the Radon transform, Eq. 10.5. Finally, the complete sinogram are used to reconstruct the volume using either Section 10.4 or 10.5. An implementation is reviewed in [71], where it is shown that the approach is viable and it is shown that an improvement of the volumes quality is obtained, compared to 2D reconstruction of the axial planes from the sinogram. The implementation was done on a VAX 8650 where the total reconstruction algorithm requires 236 minutes for a volume of $100 * 100 * 50$ voxels and a sinogram with $9 * 60 * 100 * 50$ samples.

This technique has become very popular. Currently, almost all 3D reconstruction articles relate their results to this method and the Kinahan & Rogers reprojection technique (called 3DRP in [110]) is also used in commercial scanners.

It should be mentioned that other strategies exist to reconstruct volumes from line integrals. In [121] the SSRB method (Single Slice Rebinning) is presented. A simple approximation is used, namely that the four-dimensional sinogram first is rebinned into a set of 2D sinograms, by projecting onto the nearest axial slice. These 2D sinograms are then reconstructed using 2D techniques. This approach is much faster, and can be implemented using significantly less memory, compared to the "true" 3D reconstruction techniques. In [110] a comparison has been made showing that for a PET scanner with $\psi = 9°$ the SSRB method can essentially be as accurate as the 3DRP method, and for the HEAD PENN-PET scanner with $\psi = 27°$ (developed at the University of Pennsylvania) a more accurate reconstruction is found using 3DRP.

Another strategy has recently been developed by Defrise [122]. Basically the idea is like the SSRB, i.e., to map the 4D sinogram onto a set of 2D sinograms, but here the mathematical development is based on a frequency-distance relation in the sinogram. In this method the Fourier transform of the 4D sinogram is mapped onto a set of 2D sinograms in the Fourier domain. For practical PET usage this method will also need additional parts of the sinogram found by use of, e.g., the Kinahan & Rogers technique.

As stated in Section 9.1, one of the benefits of using iterative reconstruction methods is that an irregular geometry is (in principle) easily supported, hence the reprojection step described above can be avoided. In [110] 3D iterative reconstruction algorithms (ART and EM) have been tested along with the 3DRP method. The conclusion was that ART gave the highest Signal to Noise ratio, closely followed by 3DRP. Another conclusion was that the EM method apparently performed worse. In their example the reconstruction required 2-10 hours for ART, 3.3 hours for 3DRP, and 22-64 hours for EM. These measurements correspond to a SUN Sparc 10.

10.7 A 3D Reconstruction Package

A software package for 3D reconstruction of volumes from line integrals has been developed. The package has been written in C for Unix systems, and is available for free at [10]. The package has been tested on Linux systems, and on an Onyx-computer from Silicon Graphics (SGI).

Going from 2D to 3D reconstruction one major difference is the size of the inversion problem, as the examples mentioned in Section 10.6 indicated. A practical reconstruction program (direct or iterative) will include several steps, of which filtering, Radon transform, and the backprojection is the most time consuming (the actual elements depend on the algorithm). In the implementation these central elements have been written to run efficiently on a single processor system, as well as in parallel on an Onyx using the Iris Power-C compiler. Compiler options has been added to the code to enable parallel execution, and one of the nice features of the SGI implementation is that the code still can be compiled on any other Unix machine, e.g., a Linux box, where the additional parallel compiler options will be ignored.

The reconstruction program "Recon3D" is intended for testing of different 3D reconstruction algorithms, and no Kinahan & Rogers reprojection step is currently included, in order not to mix 2D and 3D algorithms, but all of the tools for implementing the reprojection step are available.

The program will require a uniformly sampled sinogram, as it will shown in Eq. 10.68, based on the Ω_ψ-geometry and the reconstructed volumes will also be sampled uniformly. The value of ψ is defined by the user. The package also includes the program "3D_RadonAna", which has been developed for generating a volume and the corresponding four-dimensional sinogram from a set of scaled, translated, and rotated primitives. This program is based on the properties shown in Appendix E and especially Section E.2. The usage of the program is shown in Appendix F. In the following section the implementation of the Radon transform operator and the backprojection operator are shown, which cover two of the central elements of the software package.

10.8 Implementation of the 3D Reconstruction Methods

It has been shown that reconstruction of volumes from 4D line integrals can be done by filtering, either the projections, cf. Eq. 10.62, or the backprojected volume, cf. Eq. 10.50. The implementation is a straight forward generalization of the 2D reconstruction implementation discussed in Chapter 8, but the filters are not as simple, if the limiting angle $\psi < \frac{\pi}{2}$. Again, apodizing windows should be multiplied to the filters in order to limit the influence of noise.

Here the reconstruction algorithms are based on the geometry Ω_ψ, and the parameters in the parameter domain are sampled uniformly

$$\begin{array}{llll} \phi & = \phi_p & = -\psi + p\frac{2\psi}{P-1}, & p = 0,1,\ldots,P-1 \\ \theta & = \theta_t & = t\frac{\pi}{T}, & t = 0,1,\ldots,T-1 \\ u & = u_i & = u_{min} + i\Delta u, & i = 0,1,\ldots,I-1 \\ v & = v_j & = v_{min} + j\Delta v, & j = 0,1,\ldots,J-1 \end{array} \tag{10.68}$$

where $\Delta\phi = \frac{2\psi}{P-1}$, $\Delta\theta = \frac{\pi}{T}$, $\theta_{min} = 0$, and $\phi_{min} = -\psi$ have been inserted.

The parameters of the reconstructed volume are also samples uniformly

$$\begin{aligned} x &= x_k &&= x_{min} + k\Delta x, && k = 0, 1, \ldots, K-1 \\ y &= y_l &&= y_{min} + l\Delta y, && l = 0, 1, \ldots, L-1 \\ z &= z_m &&= z_{min} + m\Delta z, && m = 0, 1, \ldots, M-1 \end{aligned} \tag{10.69}$$

Note that a measured sinogram must be resampled (rebinned) into this geometry and missing parts of the sinogram can be estimated using the Kinahan & Rogers reprojection technique.

10.8.1 Implementation of the Backprojection Operator

The backprojection operator can be approximated by a sum, here shown using a nearest neighbour approximation

$$\hat{g}(\boldsymbol{r}_{k,l,m}) = \int_{\theta=0}^{\pi} \int_{\phi=-\psi}^{\psi} \breve{g}(\theta, \phi, u = \boldsymbol{r} \cdot \boldsymbol{\alpha}, v = \boldsymbol{r} \cdot \boldsymbol{\beta}) \ \cos\phi \ d\phi \ d\theta \tag{10.70}$$

$$\approx \Delta\theta\Delta\phi \sum_{p=0}^{P-1} \cos\phi_p \sum_{t=0}^{T-1} \breve{g}\left(t, p, \left[\frac{\boldsymbol{r}_{k,l,m} \cdot \boldsymbol{\alpha}_{p,t} - u_{min}}{\Delta u}\right], \left[\frac{\boldsymbol{r}_{k,l,m} \cdot \boldsymbol{\beta}_{p,t} - v_{min}}{\Delta v}\right]\right) \tag{10.71}$$

In Algorithm 10.1 the implementation of the backprojection operator is shown, when using nearest neighbour interpolation. Better interpolation schemes, such as bilinear interpolation in the u and v parameters, can easily be included.

It has been shown that all of the reconstruction methods are heavily based on projections of the directional vectors $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\tau}$ onto the volume coordinates $\boldsymbol{r}$. In order to speed up the reconstruction algorithms, the vectors $\boldsymbol{\alpha} = (\alpha_x, \alpha_y, \alpha_z)^T$, $\boldsymbol{\beta} = (\beta_x, \beta_y, \beta_z)^T$, and $\boldsymbol{\tau} = (\tau_x, \tau_y, \tau_z)^T$, should be computed once, for all values of (p, t). This can be done rather efficiently and will only require six matrices. In Algorithm 10.1 the arrays are assumed given, e.g., $\mathsf{alpha\_x(t,p)} = \alpha_x(\theta_t, \phi_p)$. For further optimization, the double loop of all possible angles (θ, ϕ) can be combined into one loop, and the values of $\mathsf{x(k)}$, $\mathsf{y(l)}$, and $\mathsf{z(m)}$ should be moved to simple variables before entering the inner loops. In this way many of the array calculations can be avoided. Additional optimization techniques have been shown in Chapter 1 for the 2D Radon transform.

For each value of (t, p), Eq. 10.71 requires that the u- and the v- values lies within the bounds shown in Eq. 10.68. One scheme to avoid this time-consuming testing is to expand the sinogram in the u and v direction (by padding with zeros at the edges), though this might not be desirable, due to the memory requirements (a measured sinogram from a GE Advance PET scanner can require about 65 MBytes of memory). For the expansion, it can easily be shown that

$$\left\lfloor \frac{-\max|\boldsymbol{r}_{k,l,m}| - u_{min}}{\Delta u} \right\rfloor \le i \le \left\lceil \frac{\max|\boldsymbol{r}_{k,l,m}| - u_{min}}{\Delta u} \right\rceil \tag{10.72}$$

$$\left\lfloor \frac{-\max|\boldsymbol{r}_{k,l,m}| - v_{min}}{\Delta v} \right\rfloor \le j \le \left\lceil \frac{\max|\boldsymbol{r}_{k,l,m}| - v_{min}}{\Delta v} \right\rceil \tag{10.73}$$

Another scheme to avoid the index testing, could be to compute the intervals of (p, t), which will give legal values of u and u, cf. Eq. 10.68. In principle this is viable, but might require more computations compared to the reduction being offered by this alteration.

ALGORITHM 10.1 : BACKPROJECTION OPERATOR IN 3D

```
For k=0 to K-1                                                //For all values of x
  For l=0 to L-1                                              //For all values of y
    For m=0 to M-1                                            //For all values of z
      sum=0                                                   //Initialize sum
      For p=0 to P-1                                          //For all values of phi
        sump=0                                                //Initialize sump
        For t=0 to T-1                                        //For all values of theta
          u=x(k)*alpha_x(t,p)+y(l)*alpha_y(t,p)+z(m)*alpha_(t,p)  //Calculate u value
          i=round((u-u_min)/Delta_u)                          //Calculate i index
          If 0≤i<I                                            //Check if index is valid
            v=x(k)*beta_x(t,p)+y(l)*beta_y(t,p)+z(m)*beta_(t,p)   //Calculate v value
            j=round((v-v_min)/Delta_v)                        //Calculate j index
            If 0≤j<J                                          //Check if index is valid
              sum=sum+g_radon(t,p,i,j)                        //Update sum
            End
          End
        End
        sum=sum+sump*cosphi(p)                                //Update sump
      End
      g_backproject(k,l,m)=sum*Delta_rho*Delta_phi            //Store result
    End
  End
End
```

The number of loops shown in Algorithm 10.1 illustrates that the complexity of backprojection is high, namely

$$\mathcal{O}_{\text{3D Backprojection}} = \mathcal{O}(KLMPT) \tag{10.74}$$

Hence, the complexity increases with the number of voxels in the reconstructed volume times the number of angular samples in the sinogram, and it indicates that the backprojection operator is a rather demanding operation.

One nice feature is that Algorithm 10.1 can be parallelized very easily. On the Onyx-computer using Iris Power-C, it was concluded that the parallel chunks of code should be as large as possible, and in Section 10.9 it will be shown that splitting the outer-most loop of the algorithm, i.e., the k-loop on four processors leads to a very good performance.

10.8.2 Implementation of the Radon Transform Operator

An implementation of the Radon transform defined in Eq. 10.5 will now be shown. This operator is needed for the Kinahan & Rogers reprojection methods and for any of the iterative methods presented in Chapter 9.

$$\breve{g}(\theta,\phi,u,v) = \int_{-\infty}^{\infty} g(s\boldsymbol{\tau}+\boldsymbol{r}_0)\,ds, \quad \text{where } \boldsymbol{r}_0 = (x_0,y_0,z_0)^T = u\boldsymbol{\alpha}+v\boldsymbol{\beta} \tag{10.75}$$

$$= \int_{-\infty}^{\infty} g(s\tau_x+x_0, s\tau_y+y_0, s\tau_z+z_0)\,ds \tag{10.76}$$

In order to avoid the problems with too high slopes, analogous to the problems pointed out in Section 1.7, the line integral is projected onto the axis with maximum absolute component of the directional vector relative to its sampling interval. It can be mentioned that this strategy also was applied for the 2D Radon transform in Section 2.2. Assume that the projection is made onto the x-axis

$$\frac{|\tau_x|}{\Delta x} > \frac{|\tau_y|}{\Delta y} \quad \text{and} \quad \frac{|\tau_x|}{\Delta x} > \frac{|\tau_y|}{\Delta y} \tag{10.77}$$

Then the 3D Radon transform can be rewritten

$$\breve{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} \frac{1}{|\tau_x|} g\left(x, xy^{(x)} + y_0^{(x)}, xz^{(x)} + z_0^{(x)}\right)\, dz \tag{10.78}$$

where

$$y^{(x)} = \frac{\tau_y}{\tau_x} \quad \text{and} \quad y_0^{(x)} = y_0 - y^{(x)}x_0 \tag{10.79}$$

$$z^{(x)} = \frac{\tau_z}{\tau_x} \quad \text{and} \quad z_0^{(x)} = z_0 - z^{(x)}x_0 \tag{10.80}$$

where the exponent (x) merely means with respect to x.

A simple discretization, which will require few calculations each time, can now be derived from Eqs. 10.69, 10.76, 10.79, and 10.80.

$$\breve{g}(\theta, \phi, u, v) \approx \Delta x \sum_{k=0}^{K-1} g\left(k, \left[\frac{x_k y^{(x)} + y_0^{(x)} - y_{min}}{\Delta y}\right], \left[\frac{x_k z^{(x)} + z_0^{(x)} - x_{min}}{\Delta z}\right]\right) \tag{10.81}$$

$$= \Delta x \sum_{k=0}^{K-1} g\left(k, \left[a_y^{(x)}k + b_y^{(x)}\right], \left[a_z^{(x)}k + b_z^{(x)}\right]\right) \tag{10.82}$$

$$\text{where} \quad a_y^{(x)} = \frac{\tau_y}{\tau_x}\frac{\Delta x}{\Delta y} \quad \text{and} \quad b_y^{(x)} = \frac{\tau_y}{\tau_x}\frac{(x_{min} - x_0)}{\Delta y} + \frac{y_0 - y_{min}}{\Delta y} \tag{10.83}$$

$$a_z^{(x)} = \frac{\tau_z}{\tau_x}\frac{\Delta x}{\Delta z} \quad \text{and} \quad b_z^{(x)} = \frac{\tau_z}{\tau_x}\frac{(x_{min} - x_0)}{\Delta z} + \frac{z_0 - z_{min}}{\Delta z} \tag{10.84}$$

With respect to the high slope problem, Eq. 10.77 implies that $\left|a_y^{(x)}\right| \leq 1$ and $\left|a_z^{(x)}\right| \leq 1$, i.e., a step from k to $k+1$ in the sum shown in Eq. 10.82, will not lead to a step in l or m greater than 1.

The discrete implementation of the Radon transform is also quite demanding, due to the complexity

$$\mathcal{O}_{\text{3D Radon transform}} = \mathcal{O}(PTIJK) \tag{10.85}$$

which accounts for the number of times where the projection is made onto the x-axis. For projection onto the y- and z-axis similar expressions are found, with K substituted by L and M, respectively.

If either the absolute y- or z-component of $\boldsymbol{\tau}$ is the greatest, then it is very easy to derive almost identical formulas, and the formulas will only require swapping of symbols compared to the ones shown above. In Algorithm 10.2 the implementation of the discrete 3D Radon transform is shown using a nearest neighbour approximation, but, e.g., bilinear interpolation in the y and z coordinate can easily be incorporated. The algorithm only shows the case where Eq. 10.77 is fulfilled. The algorithm uses the discrete variables a_y $= a_y^{(x)}$, a_z $= a_z^{(x)}$, b_y $= b_y^{(x)}$, and b_z $= b_z^{(x)}$. Two very similar algorithms are needed to cover projection onto the y-axis and the z-axis, respectively.

Algorithm 10.2 : Discrete 3D Radon Transform

```
For p=0 to P-1                                          //For all values of phi
  For t=0 to T-1                                        //For all values of theta
    For i=0 to I-1                                      //For all values of u
      For j=0 to J-1                                    //For all values of v
        Assuming Eq. 10.77 is fulfilled
        Set a_y, b_y, a_z, and b_z from Eqs. 10.83 and 10.84
        sum=0                                           //Initialize sum
        For k=0 to K-1                                  //For all values of x
          l=round(a_y*k+b_y)                            //Calculate y-index
          If 0≤l<L                                      //Check if index is valid
            m=round(a_z*k+b_z)                          //Calculate z-index
            If 0≤m<M                                    //Check if index is valid
              sum=sum+g(k,l,m)                          //Update sum
            End
          End
        End
        g_radon(p,t,i,j)=sum*Delta_x                    //Update Radon domain
      End
    End
  End
End
```

10.9 Examples of 3D Reconstructed Volumes

In this section a few examples of 3D reconstruction using “Recon3D” are given. The 3D visualization has been done using either the commercial package IDL, or combination of the “Polyr” program [119] and the excellent Geomview program, available for free at [123]. Polyr can convert volumes into a mesh of triangles. The mesh is a iso-surface though the volume for a certain user-defined threshold level. The mesh can be imported directly into Geomview for interactively inspection of the 3D object.

10.9.1 Reconstruction of a Ball

The first example concern reconstruction of a homogeneous ball, cf. Subsection E.2.1. In this case the axial limiting angle ψ was set to $90°$. Here the reconstructed volume has $51 * 51 * 51$ voxels and the (noise free) sinogram uses $P = 12$, $T = 10$, and $51 * 51$ samples in each of the (u, v) planes. Both the volume and the sinogram requires roughly 0.5 MBytes of memory.

In Fig. 10.4, 3D Filtered backprojection has been used to reconstruct the ball, and in Fig. 10.5 3D Filtering after Backprojection. In both cases an iso-surface has been generated using IDL, and a slice has been inserted to show the volume values in that plane. The figures indicate that the ball has been recovered well, though the edges of the ball is somewhat blurred. This could be expected, due to the low number of samples in both domains. The figures are not intended for quantitatively evaluation of the reconstruction results, but illustrates which kind of tools can be used to visualize 3D reconstructed volumes.

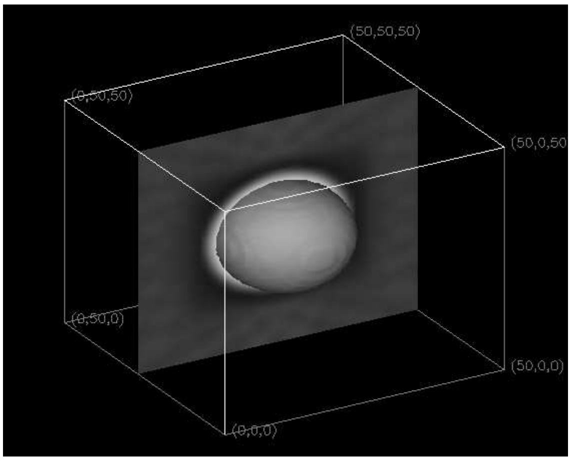

Figure 10.4 Reconstructed ball using 3D Filtered Backprojection.

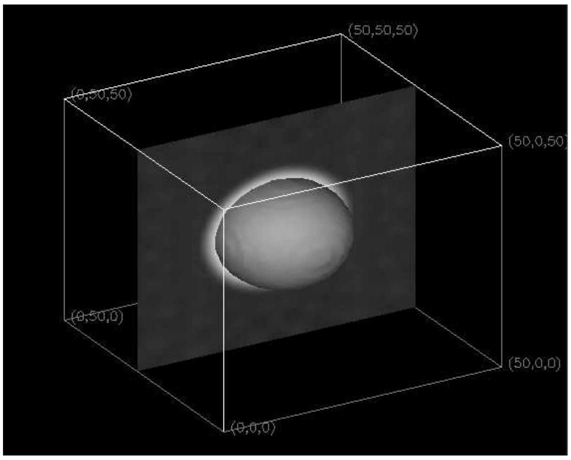

Figure 10.5 Reconstructed ball using 3D Filtering after Backprojection.

Next the iterative reconstruction methods ART and EM have been used to reconstruct the same ball. For ART some artifacts were be found, while EM is very successful. The values outside of the ball are very close to zero. Here figures like Fig. 10.4 has been omitted, due to the high similarity to the one shown above. For ART the sinogram entries were chosen randomly, though with a large axial acceptance angle ψ, it was experimentally found useful to use a weight function favoring the line orientations perpendicular to the z-axis, i.e., favoring a small value of ϕ. Assume $\phi = 90°$, and a uniform random weighing scheme to select the sinogram index in ART, cf. Section 9.6. In that case many of the lines orientations through the scanned object will be close, and ART will converge very slowly or produce artifacts, cf. Fig. 9.8.

For the Onyx, Table 10.1 shows the times needed to reconstruct the ball, obtained with and without parallel options enabled. When running in parallel, the program will use four 200 MHz MIPS R4400 processors.

For Filtering Backprojection running in parallel the filtering required 0.24 sec and the backprojection required 3.95 sec. The same figures for one processor are 0.62 sec and 13.91 sec. In the software package the efforts were concentrated on the backprojection, and here a speedup factor of 3.5 was obtained on four processors. Filtering after Backprojection gets slower because the sinogram is being backprojected into a larger volume in order to avoid edge effects, and the speedup factor is somewhat lower due to a larger part of the total time is assigned to 3D filtering, based on a multi-dimensional FFT (currently) running on a single processor. For reference, the much cheaper Pentium 120 MHz running Linux can reconstruct the same volume in 28 sec, when using Filtered Backprojection.

From Table 10.1 is can be seen that ART actually gets slower when running in parallel mode. In each iteration of ART, the part of the program running in parallel is the Radon transform of the volume for one single parameter vector. The size of code assigned to each processor is here very small, and most of the time the four processors are waiting to synchronize their operation.

10.9.2 Reconstruction of the Mickey Phantom

In this example the axial acceptance angle has been reduced to merely $\psi = 9°$. The volume was centered around $(0, 0, 0)$ and sampled with $\Delta x = \Delta y = \Delta z = 0.5$, and $K = L = M = 71$, which requires 1.4 MBytes of memory. In the sinogram $T = 90$, $P = 11$, $I = J = 61$, and $\Delta u = \Delta v = 1$ were used, and the sinogram requires 14.7 MBytes of memory.

Algorithm	FB	FAB	ART	EM
Time in seconds on one processor	14.5	26	6	6
Time in seconds running parallel	4.2	10	39	18
Speedup factor	3.5	2.6	0.15	3

Table 10.1 Reconstruction time for reconstruction of the ball using Filtered Backprojection (FB), Filtering after Backprojection (FAB), one full iteration of ART, and finally one iteration of EM. All times are measured in seconds on an Onyx.

Using 3D Filtering after Backprojection, the reconstructed volume has been visualized using Polyr and Geomview. Fig. 10.6 shows the result, which looks like the original Mickey Mouse phantom. In Figs. 10.7 and 10.8 the reconstructed central (x, y) and (y, z) planes are shown. The following figures use individual color scale according to the minimal and maximal value. This phantom has the value 0.2 inside the "skull" and a small "tumor"-ball with radius of 1 and value 0.4 placed at $(0, 1, 0)$. The tumor is visible but blurred, but it is clear that the general structures have been recovered well. The reconstructing required 6754 sec on a 120 MHz Pentium, and here the sinogram was backprojected onto a $128 * 128 * 128$ volume in order to use radix-2 FFT filtering. No apodizing windows have been used, and is probably the reason for the ripples outside of the phantom.

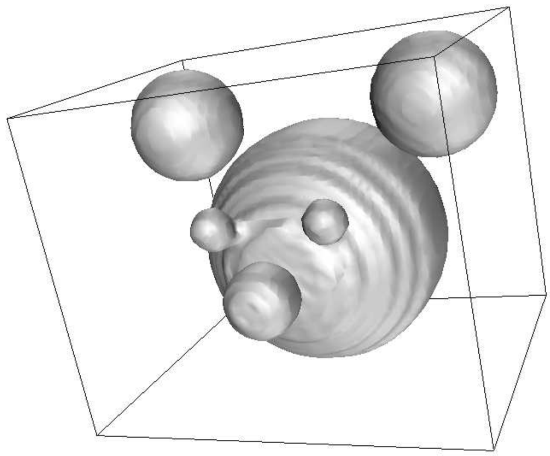

Figure 10.6 The Mickey phantom.

Next Filtered Backprojection has been used to reconstruct the same phantom. The total reconstruction used 1189 sec on the 120 MHz Pentium, divided in 75 sec for the filtering, 1111 sec for the backprojection and 3 sec for additional minor operations. Figs. 10.9 and 10.10 show the central (x, y) and (y, z) planes for Filtered Backprojection.

Finally, five iterations using the EM algorithm has been used. Fig. 10.11 and 10.12 again show the (x, y) and (y, z) planes. It is obvious that additional iterations are needed, and the images look blurred.

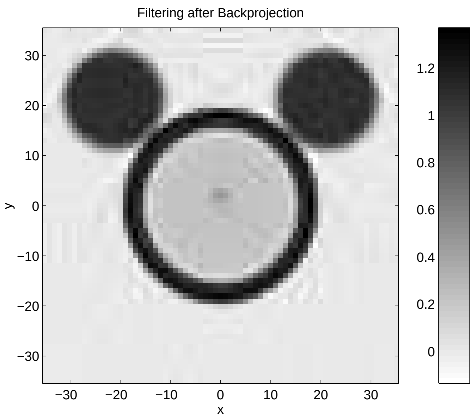

Figure 10.7 Reconstructed phantom in the central (x, y)-plane using Filtering after bacprojection.

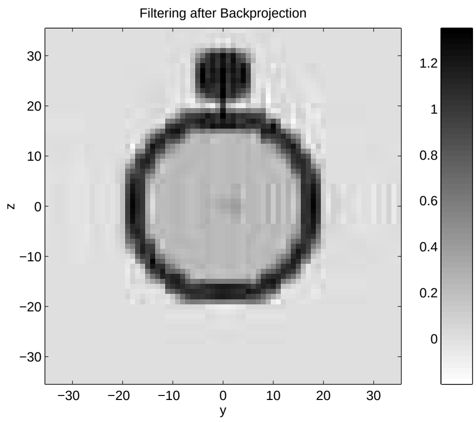

Figure 10.8 Reconstructed phantom in the central (y, z)-plane using Filtering after bacprojection.

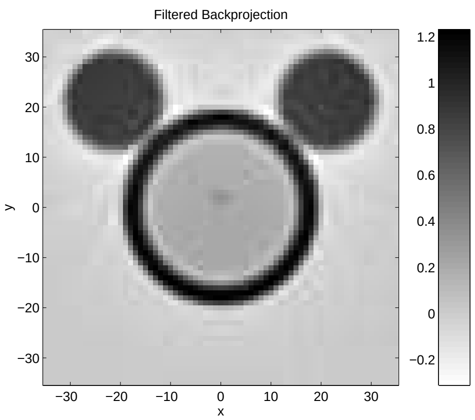

Figure 10.9 Reconstructed phantom in the central (x, y)-plane using Filtered Backprojection.

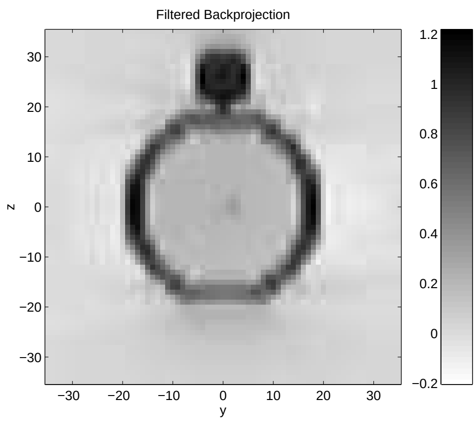

Figure 10.10 Reconstructed phantom in the central (y, z)-plane using Filtered Backprojection.

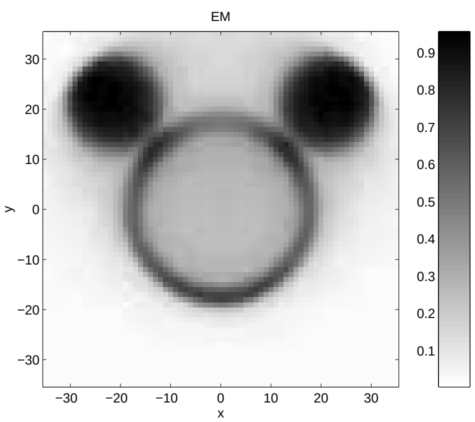

Figure 10.11 Reconstructed phantom in the central (x, y)-plane using five iterations of EM.

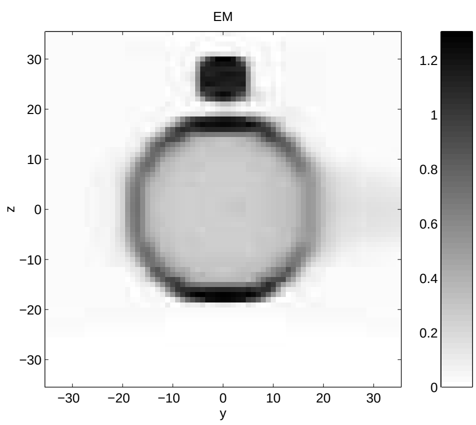

Figure 10.12 Reconstructed phantom in the central (y, z)-plane using five iterations of EM.

10.10 Summary

In this chapter several methods have been described for reconstruction of volumes from their line integrals, which is very relevant for 3D PET scanners. It has been shown that the measured sinogram in this case becomes four dimensional, and the reconstructed volumes are three dimensional. The problems with this change in dimensions have been outlined, but more research is needed in this area.

The implementation of two of the central elements of reconstruction algorithms have been shown, and aspects of parallel implementation are given. The theory has been implemented in a software package, where both direct and iterative reconstruction methods are available. The package also includes a program for generation of synthetic sinograms and the corresponding volume. This is very useful when testing the reconstruction methods.

The software package is working, but some parts of the package can be improved with respect to the tradeoff between the degree of approximation and the time needed to reconstruct the volumes. Especially the use of apodizing windows in the algorithms could be investigated, and this will definitely alter how the reconstruction algorithms are influenced by noise. That is why only a few examples have been presented with sparse information on the quantitatively performance of the individual algorithms.

Chapter 11

Noise Contributions from Blank, Transmission and Emission Scans in PET

This chapter describes quantification of noise contributions in PET, and is more practically oriented than the previous chapters. This work has been carried out at the National University Hospital in Copenhagen (Rigshospitalet) [7, 8].

11.1 Introduction

PET scans require correction for attenuation in order to be quantitative, cf. Eq. 6.22. Even in brain activation studies where absolute units are often replaced by relative values (i.e., normalized to a mean value), inter-individual comparisons still assume that values across an image reflect the local tracer concentration. Corrections can be applied, e.g., by assuming uniform values of attenuation within geometrically defined boundaries or in segmented areas of the image, but a more accurate description of the attenuation requires measurement by a transmission scan.

Unfortunately the attenuation correction also adds noise to the image and adds to the total procedure time. In some cases, mainly in body scanning, the increase in noise is immediately noticed and may even be so high that quantification must be sacrificed in order to provide a reasonable visual impression. However, in other situations, effects that are invisible for the human observer might still be of importance when examining the subtle differences of brain activation by statistical methods. The purpose of this work is to establish guidelines for the relative importance of the noise contributions from blank, transmission, and emission scans in typical imaging situations.

Although much information can be deduced theoretically from existing knowledge of reconstruction algorithms [124, 125, 126], here a mainly experimental approach has been applied. The limitations in generality caused by the obvious difficulty of varying all of the many possible parameters of acquisition and reconstruction are outweighed by the advantage of the close reproduction of actual scan setups, including some human studies. The following section describes the model used for fitting to the observed data. It should not be considered as a rigorous derivation; it merely lists the different structures of the datasets included in the study comprising homogeneous as well as inhomogeneous objects.

11.2 Theory

Due to the the linearity of the reconstruction process when using direct reconstruction methods, such as Filtered Backprojection, the PET images (volumes) denoted x_i, can be described as a weighted sum of the sinogram values s_j, with weight factors ϕ_{ij} corresponding to the linear reconstruction algorithm, cf. 9.25.

$$x_i = \sum_j \phi_{ij} s_j \tag{11.1}$$

where the actual image pixel (voxel) is denoted by i, and j denotes the pixel position in the sinogram.

In order to compensate for attenuation the sinogram used for reconstruction is calculated from an emission sinogram e_j, a transmission sinogram t_j, and a blank scan b_j. In Eq. 6.18 it is shown that calculation basically amounts to

$$s_j = \frac{e_j b_j}{t_j} \tag{11.2}$$

although this in practice usually is qualified by a filtering of the factor $\frac{b_j}{t_j}$.

All three types of sinograms in principle inherit their statistical properties from the Poisson statistics of the radioactive decay. A number of corrections that are performed before the stage of reconstruction described here, however, adds to the raw counts' noise. The overall effect of correction for randoms and scatter can be described as a decrease in effective counts, the resulting figure being widely known as Noise Equivalent Counts (NEC) [127, 126]. A practical formulation used in this work is:

$$\mathrm{NEC} = \mathcal{T} \frac{(1-\mathrm{SF})^2}{1 + 2f\frac{\mathcal{R}}{\mathcal{T}}} \tag{11.3}$$

where $\mathcal{T}$ is true counts (including scatter), $\mathcal{R}$ is random counts from the full field-of-view, f is the fraction of the sinogram covered by the object under investigation, and SF is the scatter fraction. In all applications of NEC in this study, the value of SF has been set to zero.

While NEC as a global measure may be adequate for the noise description in the case of homogeneous phantoms extending the full axial field-of-view (AFOV), objects of limited extension may be better characterized by a value per slice. In the following, all NEC values quoted correspond to a single slice.

11.2.1 One Emission and One Transmission Scan

In the simplest case where the image noise is estimated from a single image set, using a Taylor expansion and assuming no noise correlation between pixels in the reconstructed images implies that the pixel noise variance $V\{x_i\}$ can be estimated by

$$V\{x_i\} \simeq \sum_j \phi_{ij}^2 V\left\{\frac{e_j b_j}{t_j}\right\} \simeq \sum_j \phi_{ij}^2 E^2\{s_j\}\left(\frac{V\{e_j\}}{E^2\{e_j\}} + \frac{V\{b_j\}}{E^2\{b_j\}} + \frac{V\{t_j\}}{E^2\{t_j\}}\right) \tag{11.4}$$

Due to the Poisson statistics the variance in the emission sinogram relative to the squared mean value $\frac{V\{e_j\}}{E^2\{e_j\}}$ is inversely proportional to the mean of an infinite amount of experiments $E\{e_j\}$, i.e., inversely proportional to number of counts NEC in the emission sinogram. Similar arguments can be used for the transmission and blank scan terms, hence the total pixel variance normalized with the mean of the image can be modelled by

$$\frac{V\{x_i\}}{E^2\{x_i\}} = \frac{a}{\mathrm{NEC}} + b + \frac{c}{T_t} + \frac{d}{T_b} \tag{11.5}$$

where T_t is the transmission scan time in seconds, T_b the blank scan time in seconds (for the given geometry and source strength). In the model a base term b is included to accommodate effects in the reconstruction process, not explained by the other terms, e.g., varying sensitivity of the detectors, not properly corrected for by normalization. For fixed values of (either) T_t or T_b the corresponding terms may also be thought of as part of this constant term. The model applies to regions with limited structural variation, hence measuring variance in a heterogeneous object like the brain or the thorax is problematic.

11.2.2 Two Emission Scans

To overcome the problem with heterogeneous structures the difference between two PET images, corresponding to two equivalent but independent emission scans $e_j^{(1)}$ and $e_j^{(2)}$, can be calculated. Due to Eqs. 11.1 and 11.2 the difference image is given by

$$x_i = \sum_j \phi_{ij} \left(e_j^{(2)} - e_j^{(1)} \right) \frac{b_j}{t_j} \tag{11.6}$$

Using a Taylor expansion, and using that the two emission scan have the same mean values $E\{e_j\}$ and variance $V\{e_j\}$ implies that

$$x_i \simeq \sum_j \phi_{ij} \left(\left(e_j^{(2)} - E\{e_j\} \right) - \left(e_j^{(1)} - E\{e_j\} \right) \right) \frac{E\{b_j\}}{E\{t_j\}} \tag{11.7}$$

Thus the variance $V\{x_i\}$ is given by

$$V_{\text{two e-scans}}\{x_i\} \simeq 2 \sum_j \phi_{ij}^2 E^2\{s_j\} \frac{V\{e_j\}}{E^2\{e_j\}} \tag{11.8}$$

Normalized with the squared mean of the heterogeneous structure and using the same definition of constants as in Eq. 11.5 gives that

$$\frac{1}{2} \frac{V_{\text{two e-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} \tag{11.9}$$

Thus only the emission term of the noise variance can be estimated from two emission scan and one transmission scan. This result somewhat surprising, but is also found in the measurements. Since all structural information is removed by using differences, no constant term (b) can appear.

If instead two transmission scans (and one emission scan) are available, it is easy to show that two times the transmission variance can be estimated (and only that) from a normalized difference image.

11.2.3 Two Emission and Two Transmission Scans

Using the same Taylor technique with two sets of emission scans and two transmission scans, it can be shown that

$$\frac{1}{2} \frac{V_{\text{two e and t-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} + \frac{c}{T_t} \tag{11.10}$$

This implies that both the emission and transmission term can be estimated in the inhomogeneous case if a double set of independent and equivalent emission and transmission scans are measured and reconstructed pairwise. Note that any normalized variance shown in the following has been divided by two if measured from two scans, cf. Eqs. 11.5, 11.9 and 11.10.

11.2.4 Zeroes in the Transmission Sinogram

A common problem especially in body scanning is zeros in the transmission sinogram due to huge attenuation leading to few counts and poor statistics in the transmission sinogram. One option is to replace the zeros with a small number T_0, so the attenuation correction can be done using Eq. 11.2. Although overflow is avoided this may imply that reconstructed images contain strong lines causing a significant change in the noise structure. Let $\mathcal{Z}$ be the set of j, where the measured t_j is zero. Assume that the variance of the blank scan is negligible and further that only the terms with zeros effectively contribute to the noise. In this case

$$V\{x_i\} \simeq \sum_{j\in\mathcal{Z}} \phi_{ij}^2 V\left\{\frac{e_j b_j}{T_0}\right\} \simeq \sum_{j\in\mathcal{Z}} \phi_{ij}^2 V\{e_j\}\frac{E^2\{b_j\}}{T_0^2} \tag{11.11}$$

This indicates that the number of zeros (through the sum) and the variance on the emission scan determines the noise level. Hence the variance normalized with the squared mean is approximately proportional to the number of zeros and inversely proportional to NEC. In the limit where this kind of noise is dominating, however, the reconstructed images consist of many lines and become useless anyway.

11.2.5 Limitations

Using either of Subsections 11.2.1, 11.2.2, or 11.2.3 to estimate the noise terms, we estimate the noise variance from a chosen ROI (region of interest) and normalize with the squared mean of the ROI. Due to the global features in the reconstruction process the noise tends to have only a very small correlation with the underlying structure and often it is approximately evenly distributed across the ROI's used. This implies that that the normalized noise variance of the ROI depends strongly on the selected ROI, through the squared mean.

11.3 Overview of the Measurements

Measurements were made on the GE Advance PET scanner [128, 129, 130] with 3D acquisition and reconstruction capability. Its 9 GBytes raw data disk can hold about 180 3D frames (recordings with 35 slices), suitable for phantom decay studies. Reconstruction in our configuration (with 10 i860 processors) takes approximately 8 minutes per frame [131]. For blank and transmission scanning, which is always performed in 2D, the scanner applies two pin sources. During the initial experiments the pin source activities were 389 MBq and 134 MBq, respectively. The blank scan sinogram count rate was 0.91 Mcps, corresponding to an average count rate per sinogram element of $\approx$ 0.23 cps (counts per sec) in the center. On later phantom and human studies, blank and transmission scan times have been scaled in accordance with the currently observed blank scan count rates and the values quoted in seconds are therefore directly comparable.

11.3.1 Phantom studies

Three different water-filled phantoms have been used: the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm (shown in Fig. 11.1), and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm, as shown in Fig. 11.2.

Using initially the two homogeneous phantoms in accordance with Eq. 11.5, emission scans were made as decay series with F-18 in 2D (total of 22 measurements) and 3D (limited to 10 time frames). Transmission scans were made starting from T_t=64 seconds and doubling the time

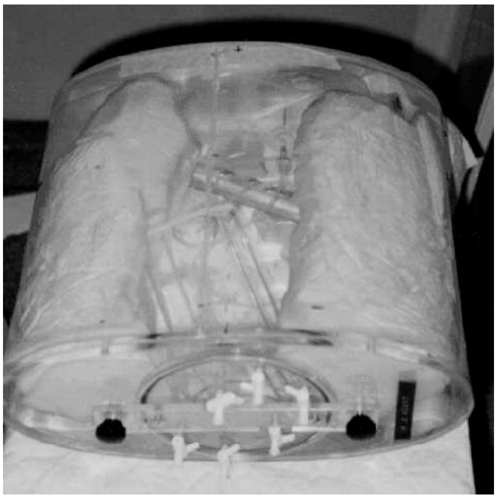

Figure 11.1 The torso-like phantom, here shown with lung-phantoms.

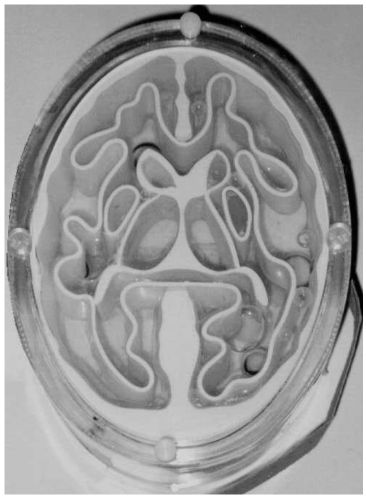

Figure 11.2 The elliptical brain phantom (Capintec) with axes 15 and 20 cm.

for each step up to 32768 seconds. Blank scans were made from T_b=64 to 4096 seconds and complemented with one 100000 seconds scan.

Reconstructions were made for all relevant combinations of emission and transmission scans using the 2048 seconds blank scan. All emission scans were further reconstructed with a calculated attenuation correction (CAC) using a circular or elliptical contour respectively. For the 2D cylinder case the emission image with highest count was reconstructed with the longest transmission scan and all blank scans. The cylinder was reconstructed in a 128^2 matrix with pixel size 2.0 mm, and a 4 mm Hann filter. For the ellipse, a 128^2 matrix of pixel size 4.0 mm was used with a 8 mm Hann filter. The 3D axial filter (Hann) was also set to its minimum value of 8.5 mm. The attenuation data were preprocessed with a 8 mm Gaussian filter for the cylinder case, and 10 mm for the ellipse. All available corrections were applied, including detector normalization, randoms subtraction, and scatter correction. In all slices of all reconstructed images (order of 25,000 images) the mean and variance was calculated from an ROI extending 70% of the diameter or ellipse axes. In 2D, the data were averaged over 31 (of 35) slices avoiding edge effects but ignoring the minor differences between direct and cross slices. In 3D only the central 15 slices having almost identical noise were included in the average. For each reconstructed dataset the noise was represented by the normalized variance, i.e., variance/mean^2, in subsequent plots and fitting. From the total rates curves a Noise Equivalent Count (NEC) value per slice was calculated for each scan frame. The rate dependent correction between trues and NECs due to randoms did not exceed 15%.

The brain phantom has two separate chambers ('grey' and 'white' matter, respectively). It was filled to resemble the usually quoted ratio of 4:1 between these two substances for flow or metabolism. Two different sets of measurements have been performed. One set (repeated in 2D and 3D) was originally designed to address count rate performance [132, 133]. The phantoms were loaded with C-11 carbonate well above the expected saturation limit of the scanner and pairs of (60:62) seconds scans were performed every 10 minutes for about 5 hours (15 half lives). The maximum number of slice counts observed in this study was limited to 10^6 which would not allow a sufficiently clear distinction between the transmission scan times. One more decay series was therefore prepared in 3D with F-18 measuring for 12 half lives, each split in two frames with a ratio of (0.415:0.585) yielding approximately equal NEC. The C-11 series were reconstructed using an 'infinite' (32768 sec) transmission scan. For the F-18 series, a set of 6 pairs of transmission scans, 64 - 16384 sec, were obtained. Reconstructions were made to provide datasets with independent transmission and emission noise according to Eq. 11.10 by combining the E-T frames as odd-odd, even-even. Reconstruction parameters for the brain phantom were identical to those for the cylinder except that the axial Hann filter was replaced by a Ramp filter. From both the C-11 series and the F-18 series difference images were calculated, and ROI's placed over grey matter (central and peripheral), white matter, and whole brain in the original as well as the difference images for mean and standard deviation calculation respectively.

11.3.2 Human studies

For one person (case KL) included in a count rate performance and dose optimization study with O-15 [132], the usual 90 second integration time (starting 20 seconds after bolus injection) was supplemented by 2 short frames (20-22 sec). The study comprised of injections with 25-800 MBq in 2D and 50-800 MBq in 3D. Reconstruction and analysis was performed as above, although only with one 12 minutes transmission scan. The number of NECs in the short scans were about 16% of the corresponding 90 seconds frames normally used.

One person (case MN) injected with 240 MBq of FDG had a double set of emission scans (8-512 sec) of the brain starting 2 hours after injection, followed by a double set of transmission scans (64-512 sec) yielding a total of 28 data points in (E,T)-plane. Reconstruction and data analysis was made to match the brain phantom F-18 series.

One patient (case BB) undergoing a dynamic FDG scanning of the heart additionally had a series of 6 transmission scans (0.5 - 20 min). Pairs of emission scans (0.5,1,2,5 min) from different parts of the dynamic study were reconstructed with all the transmission scans, and subsequent data analysis made as described above (Eq. 11.9) in a large body circumferential ROI.

11.3.3 Fitting data

From the emission data (see Subsections 11.2.1, 11.2.2, and 11.2.3) the parameter a was estimated by linear regression in a log-log diagram of the normalized variance using the longest transmission scan available. Depending on the setup the transmission parameter c (and the base term b) was subsequently estimated by subtracting the estimated emission part of the normalized variance, a/NEC, and again applying linear regression. This approach is very simple and was considered adequate for the purpose. Non-linear data fitting was also tried and found to give similar results.

11.4 Results

In this section data and parameters are presented for the previously described experiments and compared to the model. Again it should be noted that any variance shown has been divided by two if measured from paired difference scans so that the results presented are comparable and represent noise in the single images.

Estimated model parameters from the Cylinder phantom in 2D and 3D modes, the Elliptical Body phantom in 2D and 3D, The Brain Phantom, and Case MN, are given in Table 11.1. This table give parameters valid for any practically used value of NEC and T_t.

In Fig. 11.3 the measured normalized variance in the Cylinder (2D) case is shown. It can be seen that the transmission scans will effectively start to add to the variance if NEC is larger than approximately 10^5 counts per slice and that they - for transmission scan lengths normally encountered - are dominating at 10^7. Note also, that the emission noise model as judged from the

Case	Mode	a [k counts]	b [ ]	c [sec]
Cylinder	2D	47.7	0.00188	6.40
Cylinder	3D	27.0	0.00035	1.64
Ellipse	2D	24.6	0.00167	8.85
Ellipse	3D	33.8	0.00154	4.16
Brain Phantom	3D	16.7		1.01
Case MN	3D	18.5		1.59

Table 11.1 Estimated model constants.

linearity in the log-log plot remains valid down to the level where the noise exceeds the mean by an order of magnitude, i.e., beyond any practical application of the images as such.

In the Cylinder (2D) case the blank scan parameter d has also been estimated from reconstructed images corresponding to varying blank scan length T_b, and the estimated value is $d = 0.96$. By comparison with the parameter c from Table 11.1 the transmission scan in the cylinder case (2D) is seen to contribute approximately 6.7 times more to the variance than a blank scan with the same duration, in accordance with the average attenuation of the central region of a 20 cm water-filled phantom.

In Fig. 11.4, corresponding to the Brain phantom scanned in 3D, a contour plot shows the normalized variance as a function of NEC and transmission length T_t. Shading is used to indicate the actual variance, which can be seen from the rightmost grey scale bar. Contours show approximate equidistant level of noise. Furthermore two lines are inserted to show where the transmission term equals 10% and 100% respectively of the emission term.

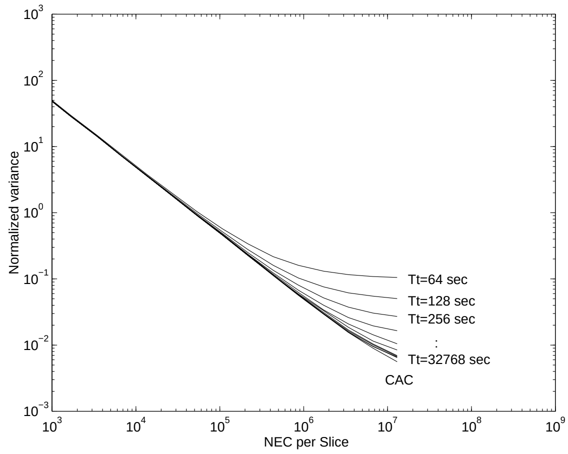

Figure 11.3 Normalized variance from 2D PET scan of the 20 cm Cylinder phantom as a function of NEC per slice and duration of transmission scan T_t. CAC means calculated attenuation correction.

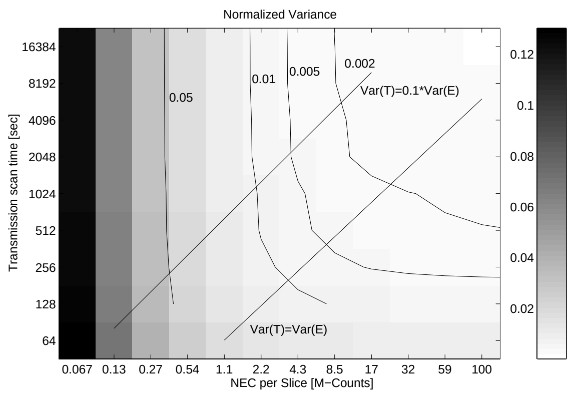

Figure 11.4 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t. Contours show iso-noise curves with levels indicated. Lines show where the ratio of transmission to emission noise terms is unity and 10% respectively.

In the Brain phantom (3D) case and in case MN two emission and two transmission scans were measured, thus both emission and transmission parameters have been estimated. The parameters are listed in Table 11.1. Despite the approximations made for deriving the models Figs. 11.5 and 11.6 demonstrate an excellent match between the measured data and the model for many decades of NEC. Note that the very low noise level, compared to Fig. 11.3, is due to the use of difference images made from paired emission and transmission scans, which eliminates the b-term.

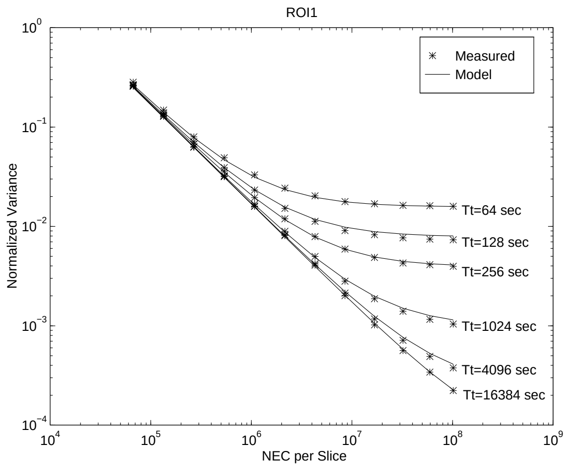

Figure 11.5 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t. Stars show the measured data and lines the fitted model.

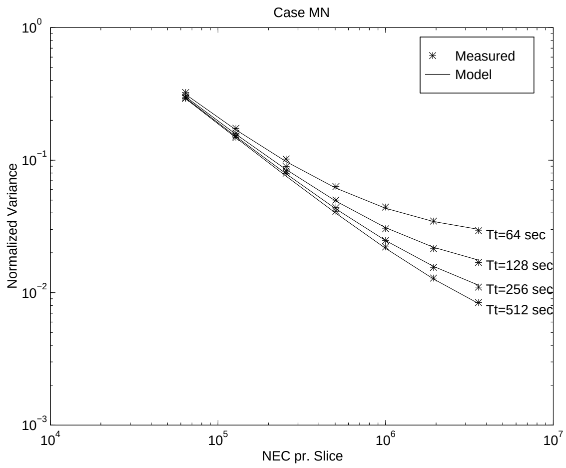

Figure 11.6 Normalized variance of case MN (3D) as a function of NEC per slice and duration of transmission scan T_t. Stars show the measured data and lines the fitted model.

Fig. 11.7 illustrates how well the emission data measured on human brains agree with the phantom studies. The figure shows case KL (2D/3D), case MN (2D/3D), and the Brain Phantom (2D/3D). In all cases data corresponding to only one transmission scan are shown.

Table 11.4 shows noise estimates obtained from different ROIs. The emission parameter a and the transmission parameter c have been estimated for the Brain Phantom (3D) in four different ROIs. The Head ROI is an ellipse just surrounding the activity of the brain, WM is a small region within white matter, $\mathrm{GM_p}$ and $\mathrm{GM_c}$ are two small grey matter regions, peripheral and central, respectively. The averages of the measured regions are normalized to the average of WM. Note the approximately constant ratio between a and c, indicating the major influence of the average on the coefficients.

ROI	a [k counts]	c [sec]	Average
Head	17	1.0	2.07
WM	72	3.7	1.00
$\mathrm{GM_p}$	4.4	0.24	3.98
$\mathrm{GM_c}$	7.2	0.49	4.05

Table 11.2 Brain phantom data (3D) with different ROIs.

Finally Fig. 11.8 shows data from body measurements. The 2D Elliptical Body phantom follows the previous noise model, but in case BB (large patient with arms in FOV, only a single set of transmission scans) the noise curves for the 1 and 2 minutes transmission scans are shifted upwards. Inspection of the reconstructed images reveals large number of lines, due to (clusters of) zeros in the transmission scan (cf. Eq. 11.11).

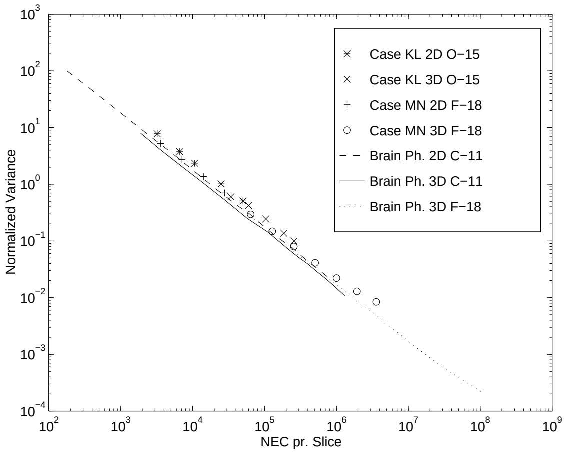

Figure 11.7 Normalized variance of Case KL (2D/3D), case MN (2D/3D), and Brain phantom (2D/3D) as a function of NEC per slice.

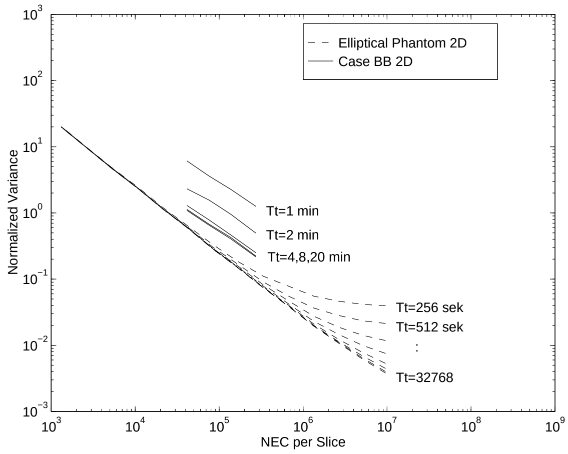

Figure 11.8 Normalized variance of body studies case BB (2D) and Elliptical phantom (2D) as a function of NEC per slice. Note that case BB (1,2 and 4 min) shows a different noise behavior.

11.5 Optimization

One application of the estimated noise parameters is optimization of the patient time used in the scanner [126]. Assume that the total time available for the examination of a patient is T seconds and that the transmission scan and the emission scan are distinct. The duration of the two scans are T_t and T_e respectively, and $T = T_t + T_e$. For simplicity assume that the rate of Noise Equivalent Counts is constant and equals R_{NEC}. This implies that the sum of the emission and transmission term are given by

$$\frac{V}{E^2} = \frac{a}{R_{\text{NEC}}(T - T_t)} + \frac{c}{T_t} \tag{11.12}$$

Thus the optimum duration of the transmission scan is given by

$$T_t = \frac{T}{1 + \sqrt{\frac{a}{c\,R_{\text{NEC}}}}} \tag{11.13}$$

Figs. 11.9 and 11.10 show the normalized variance of the individual noise terms and their sum as a function of transmission scan time T_t in two cases. Both correspond to the parameters found in Table 11.1 for the Brain Phantom (3D) and the sum of the transmission scan time and emission scan time is arbitrarily set to 1800 sec. In Fig. 11.9 it is assumed that $R_{\text{NEC}} = 1000$ counts per slice per sec, which gives an optimized normalized variance of 0.014 when $T_t = 354$ sec. Assuming $R_{\text{NEC}} = 10000$ counts per slice per sec as in Fig. 11.10 implies that the optimum is found at $T_t = 786$ sec and the normalized variance is 0.0029.

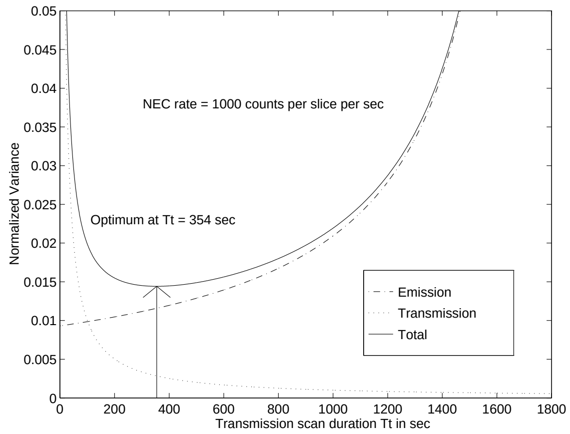

Figure 11.9 The transmission and emission noise terms and their sum as a function of the transmission scan duration, when the total (patient) scan time is 1800 sec. Parameters correspond to the Brain Phantom (3D). The rate of Noise Equivalent counts is 1000 counts per slice per sec.

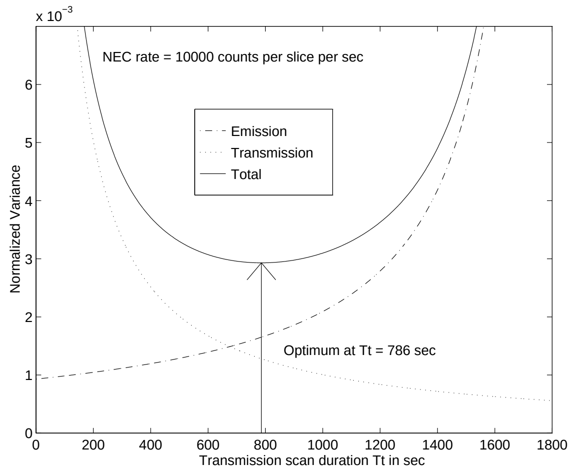

Figure 11.10 The transmission and emission noise terms and their sum as a function of the transmission scan duration, when the total (patient) scan time is 1800 sec. Parameters correspond to the Brain Phantom (3D). The rate of Noise Equivalent counts is 10000 counts per slice per sec.

11.6 Discussion of the Results

The general noise properties of reconstructive tomography are well described in the literature [124, 125]. Previous descriptions of the Advance scanner also include count rate dependence in phantoms and humans [129, 132, 133] but the noise characterization so far [134] focused on the emission noise. A theoretical derivation [126] using data from the GE 2048 scanner formed part of the basis for the design of the Advance scanner's 2-pin source transmission scanning system. In the present paper we have measured the relative importance of the noise contributions from blank, transmission, and emission scans for a range of imaging situations encountered. Transmission scan values are presented as scan durations in seconds for a given, specified set of sources. Optimal conditions with 2 sources of maximal strength (400 MBq) would almost double the transmission count rate. Given the current expensive Germanium-68 pin sources, the presented configuration represents a likely mean value over source life time. Results can be scaled according to the observed blank scan count rate, to accommodate for differences in number or activity of pin sources and their rotation radius.

An excellent fit to the empirical variance has been found both in 2D and 3D studies. The numerical values of the measure chosen for the noise examination is strongly dependent on reconstruction parameters, in part because of the neighbour pixel correlation disregarded in the theory section. Most parameters, however, will affect the emission and transmission contributions in the same way (through ϕ_{ij} only), and therefore the model and its output is considered adequate for the purpose of identifying areas in the E-T plane where one source is dominating. The analysis largely ignores the fact that the tomographic noise is non-stationary with higher values found towards the center as exemplified by the differently located ROIs in the brain phantom, but also in this respect is the ratio between the emission and the transmission term approximately constant.

From the estimated value of $d = 0.96$ it is seen that the blank scan contribution to the noise as expected is only a small fraction of the transmission noise for same duration, the ratio being well explained by the average attenuation of the (central part of) the 20 cm Cylinder phantom. For all practical purposes, therefore, the application of a 20 minutes blank scan will ensure that the blank scan contribution is negligible since the cases where a longer transmission scan might be applied are those with a higher attenuation.

A typical 3D brain activation study with O-15 water in our setup will contain 0.5 Mcounts for a central slice, (Fig. 11.7,[132]). With a 10 minute transmission scan, the transmission variance calculated from the phantom data in Table 11.1 is 0.00167 compared to the emission contribution of 0.0334, i.e., the transmission adds (only) 5% to the final result. If difference images are made, the effect becomes even smaller. For the case MN the corresponding figures would be 0.00265 (transmission), 0.037 (emission) and 7%, respectively. This higher value is due to the lack of a skull for the Brain phantom.

A 10 minute, 2D FDG brain scan typically would also have 0.5-1 M counts yielding the same 5-10% ratio, while acquiring the same data in 3D would make the two contributions almost equal and therefore call for a more detailed analysis of timing. It should be noted that in this imaging condition no attribute has been made to the small additional noise from the emission correction of the transmission scan which is necessary if the transmission is performed with activity present.

Fig. 11.8 suggests that within the emission count range of a typical dynamic heart FDG scan, the emission noise contribution (when using a 20 minute transmission scan) is dominating. Data are, however, currently not available that can demonstrate the region in which the transmission noise curves split up. It should be emphasized that the observed shifts of the curves with short transmission time are due to line-artifacts caused by (clusters of) zeros in the transmission sinogram. This deteriorating effect is usually more important and calls for an improved methods of attenuation correction, e.g., by using image segmentation as described in [135].

11.7 Summary

The measured image noise variance from a GE Advance PET scanner has been modelled as a sum of terms corresponding to the noise in the emission scan, the transmission scan, and the blank scan. The weight parameters in this simple model have been determined from a large number of experiments in both 2D and 3D scan mode for the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm, and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm. Furthermore, noise model parameters have been found from some human studies. For brain studies there is an excellent agreement between model and observation, and between phantoms and human studies. The estimated parameters have been used to optimize the durations of the emission and transmission scan under the constraint that their sum is constant. For studies of the heart or other body regions the model is still generally valid, but the parameters not as well documented, and the transmission noise is often dominated by line artifacts.

Conclusion and Topics for Further Research

This thesis has demonstrated that the Radon transform and its inverse can be implemented on ordinary computers and provide a stable and useful tool for image processing.

Two common parameter definitions have been considered with the linear Radon transform, and the central issues concerning sampling requirements have been covered thoroughly in Chapters 1 and 2. The literature often neglect to consider the implications of discretization, and a new sinc-interpolation strategy has been used to show that simple interpolation strategies in general are sufficient for line detection algorithms.

In Chapter 3 the relationship between the Radon and the Hough transform has been covered, and it was shown that the discrete forms of the two algorithms can give exactly the same parameter domain, if some restrictions are made on the sampling parameters. Many variations of the Hough transform can be found in the literature, and some of them are described in the last part of the chapter. From the literature, it is not clear how the Radon and the Hough transform are related. It has been demonstrated that the two methods act very differently, when changing the sampling distances in the parameter domain.

A new computationally efficient algorithm, named the FCE-algorithm, for fast curve parameter estimation has been presented in Chapter 4. The algorithm uses the Radon transform and the Hough transform in a generalized form, and exploits the best properties of the two. The algorithm initially uses the generalized Hough transform to estimate a coarsely sampled parameter domain, which is exploited to reduce the computational cost for the subsequently used generalized Radon transform. The algorithm has been validated in two examples, in which hyperbolas are detected from seismic images. The FCE-algorithm has been derived using a specific form of the generalized Radon transform, but the FCE-algorithm can be modified to cover virtually all parameterized curve shapes.

Novel approaches to analyze the influence of noise on the discrete Radon transform have been presented in Chapter 5. For the discrete Radon transform, it has been shown that parameterized curves can be detected even for extremely bad signal to noise ratios. It was also shown that the discrete Radon transform is robust to random fluctuations on the line pixels. Analytical expressions have been given in Chapter 5 to quantify the influence from these noise sources. It could also be of interest to make a combined analysis of these two noise sources.

A large software package has been developed for 2D Radon based reconstruction. Several Radon based reconstruction methods have been implemented, based on the theory presented in Chapters 7 and 8. The package also includes two other programs. The first is a program for generation of an image and the corresponding sinogram from a set of primitives. This program is based on the analytically derived Radon transform of a set of primitives, as shown in Section B.3. With the program it is simple to combine shifted, scaled, and rotated primitives to build a complex image and its sinogram. A noise model can also be added to the noise free sinogram, hence the program is suitable for testing reconstruction algorithms.

The last program in the 2D reconstruction package contains several iterative reconstruction methods, implemented using a new and computationally efficient technique, shown in Chapter 9. A huge acceleration has been obtained with the last package compared to a traditional approach, where the individual values of the system matrix are not stored but calculated when needed.

Chapter 9 covered some parts of the linear algebra based reconstruction methods. Even though many aspects have been covered or mentioned in the thesis, several issues should be examined. A modification of the iterative methods that might aid performance is to incorporate different regularization techniques. This topic should be investigated in the future. Lately a very interesting article [118], based on the combination of the Least Squares Conjugate Gradient method and regularization techniques, showed interesting results.

One major question in all iterative methods, is the choice of a stopping criterion. Like it was assumed for the derivation of the EM algorithm, work has been done, where the fundamental assumption is that the values in the sinogram are generated in a Poisson process, but it seems doubtful that methods narrowly based on the Poisson assumption is the way to proceed. In practical PET reconstruction the noise has many sources, such as the noise from the finite length of the emission, transmission, and blank scan, cf. Chapter 11. As previously mentioned additional effects like scatter, randoms, and detector variations (Chapter PET in [58]) also contribute to a total noise model being far more complicated.

An important issue in iterative reconstruction theory is the modelling of the system matrix. In Chapter 9, examples mostly using Radon based approaches have been presented, but given that a reconstructed image is a result of both the sinogram, the system matrix, the number of iterations, iterative method chosen, it could also be of interest to take the interactions of the individual parts into account.

An interesting path to follow could be to incorporate prior knowledge of edges in the model of the reconstructed image and use mean field theory [136, 137] or [11, 12]. In the last two papers, shown in Appendix N, the model only operates on the reconstructed image. However, a combination of the reconstruction process and a more detailed model of the image seems to be a very interesting research area. This model is very relevant to PET reconstruction, where prior information could be included, e.g., found from a high resolution MR-scan. The MR-scan can be used to extract edges between localized anatomical structures, which can be used to stabilize the adaptive reconstruction methods for PET. Adaptive Finite Element methods [138] might also be a way to incorporate information about edges, in order to enhance and segment the volumes.

Chapter 10 was concerned with 3D reconstruction methods, where the volume of interest should be recovered from a set of line integrals through the volume. The derivation of several reconstruction methods has been shown. The methods are not contained within the normal Radon transform definition, but it has been shown that the 2D Radon based reconstruction methods can be generalized to the 3D case, and the 2D iterative methods are almost directly applicable for 3D reconstruction. For this purpose yet another software package has been developed, providing both direct and iterative reconstruction methods. The package also includes a program for generation of volumes and the corresponding sinogram from a set of primitives like in 2D.

The area of reconstruction methods is still under development, where new adaptive methods and very computer demanding methods will become feasible, due to the rapid development in computer performance. It has, e.g., been reported [139], that reconstruction methods can be implemented efficiently using texture mapping hardware. Techniques as VRML (Virtual Reality Modelling Language) and volume rendering, will soon add new dimensions to the interpretation of volume scans.