Tomografia

Tópicos a serem abordados:

1. Tomografia
   1.1. Retroprojeção Filtrada (Filtered Backprojection)
   1.2. Reconstrução Algébrica (ART)

Bibliografia:

[Russ, 1995, chap. 9] John C. Russ, *The Image processing Handbook*, Second Edition, CRC Press, 1995.
[Herman, 1980] Gabor T. Herman, *Image Reconstruction from Projections*, Academic Press, 1980.

Novos algoritmos:

E. J. Candès, J. Romberg and T. Tao, Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information, IEEE T. Information Theory, vol. 52, no. 2, Feb. 2006, pp. 489-509
J. Trzasko, A. Manduca, Highly Undersampled Magnetic Resonance Image Reconstruction via Homotopic l0-Minimization, IEEE T. Medical Imaging, vol, 28, no, 1, Jan 2009, pp. 106-121.

Meu trabalho:

[Cn20] H. I. A. Bustos and H. Y. Kim, “Reconstruction-Diffusion: An Improved Maximum Entropy Reconstruction Algorithm Based on the Robust Anisotropic Diffusion,” in Proc. Sibgrapi - Brazilian Symp. on Comp. Graph. and Image Proc., 2005.

Tomografia é o processo para obter imagem de uma fatia do objeto usando qualquer tipo de onda penetrante.

- O equipamento usado na tomografia é tomógrafo.
- A imagem obtida é tomograma.
- O processo usado é algoritmo de reconstrução tomográfica.

$\mu(x, y)$ = coeficiente de atenuação do objeto. $r(l,\theta)$ = reta na posição l e ângulo θ.

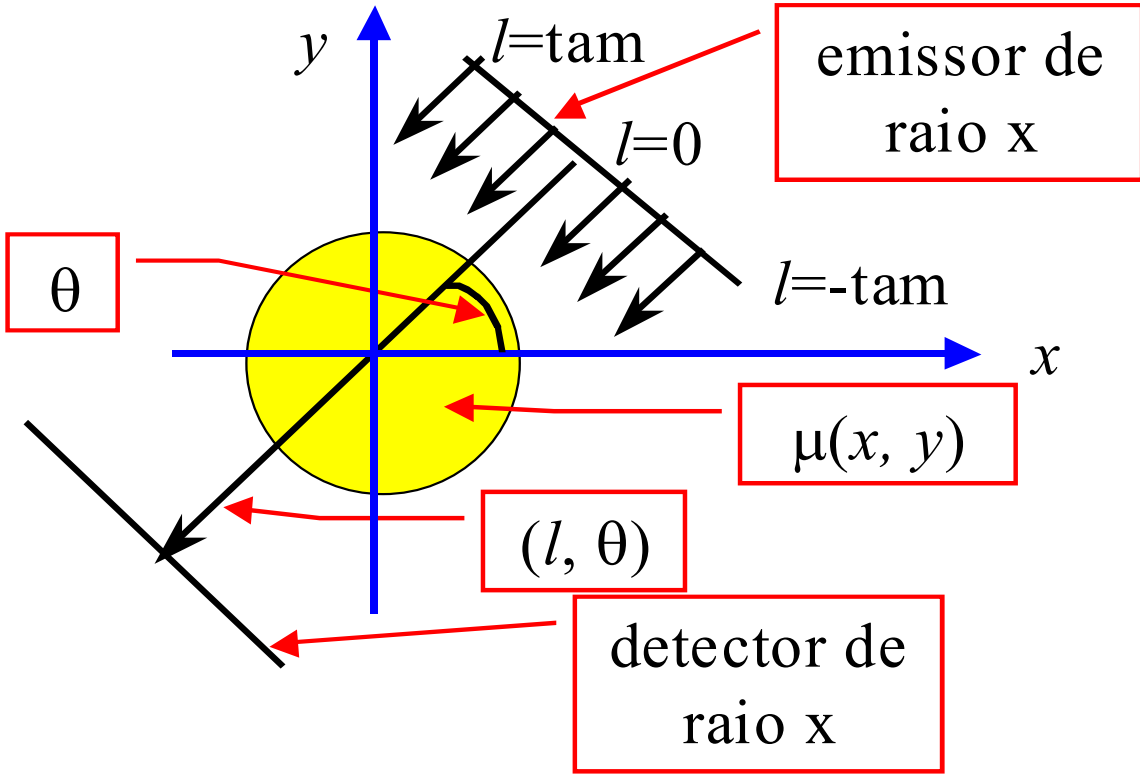
$$S(l,\theta) \propto \int \mu(x,y)dr = \ln \frac{I_o}{I_d},$$

$S(l,\theta)$ = atenuação total de um raio que segue a reta $r(l,\theta)$.

I_d = intensidade detectada I_o = intensidade emitida

supondo que a distância d entre fonte e detector permaneça constante.

Nota: Se levar em conta a distância entre fonte e detector d em relação a distância "padrão" d_o: $\frac{I_d}{I_o} \propto \left[\frac{d}{d_o}\right]^2 \exp\left[-\int \mu(x,y)dr\right]$



Problema: A partir dos $S(l,\theta)$ determinar $\mu(x,y)$.

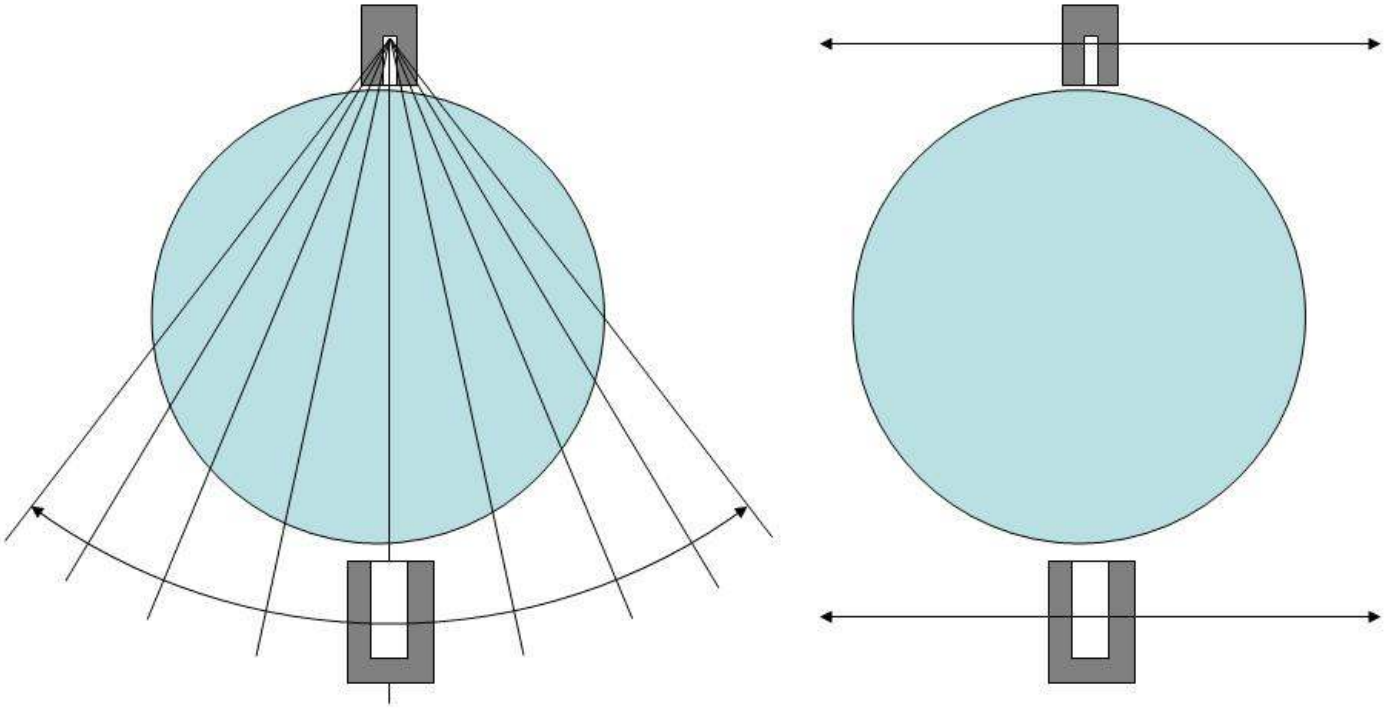
Let μ(x, y) be the spatial distribution of the coefficient of attenuation of gamma ray; I_o the emitted radiation intensity; and I_d the detected radiation intensity. Then, the following proportionality holds:

$$\frac{I_d}{I_o} \propto \left[\frac{d}{d_o}\right]^2 \exp\left[-\int \mu(x,y)dr\right]$$

where r is the path followed by the gamma ray; d/d_o is the relative distance between the ray emissor and detector in relation to a "standard distance" d_o. Rewriting the equation we obtain:

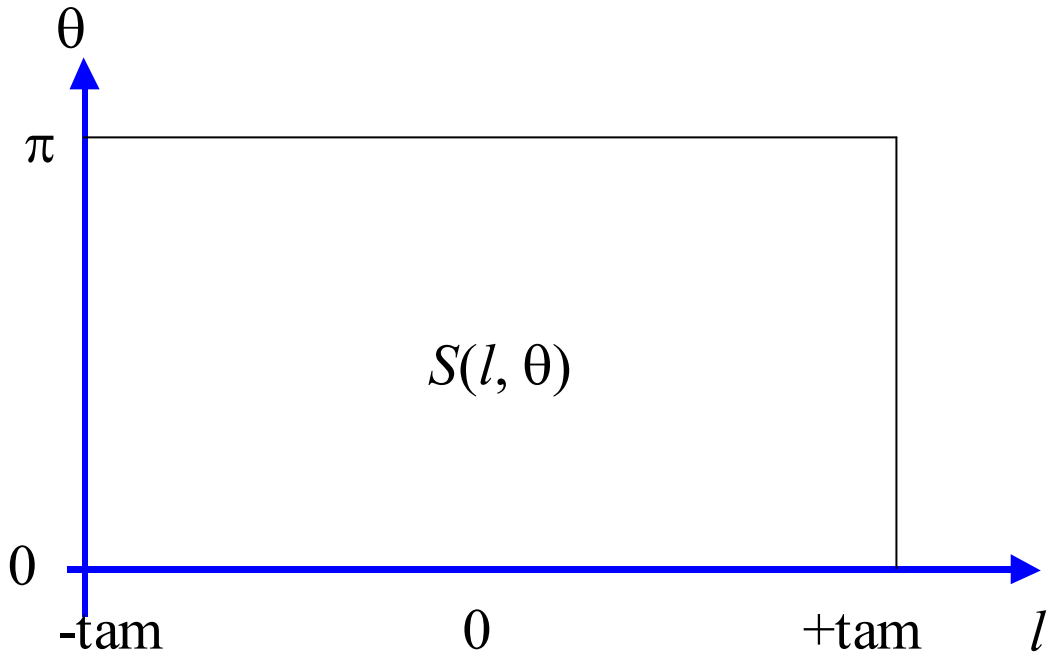
$$b = -\ln\left(\frac{I_d}{I_o}\left[\frac{d}{d_o}\right]^2\right) \propto \int \mu(x,y)dr$$

Let us call the left term of the proportionality "measured projection" b. The tomographic reconstruction problem is to determine the spatial distribution of attenuation μ(x, y) given many measured projections b in many geometrically different ray paths r.



Geometria em leque e paralela.

Sinograma:πθ



Problema: Dado sinograma $S(l, \theta)$, obter a densidade $\mu(x, y)$.

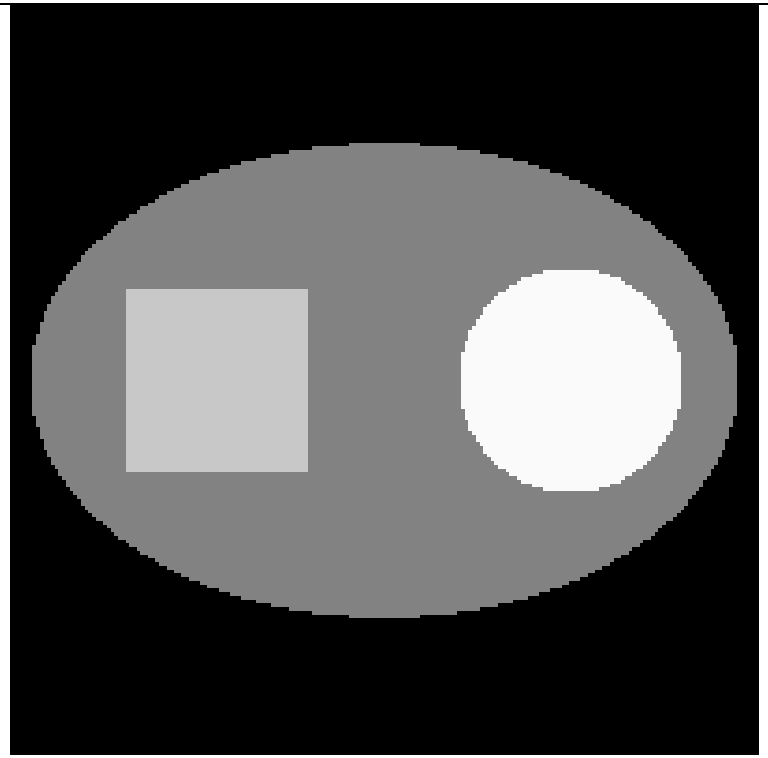
Geração de phantom (f. densidade)

```
#include <proeikon>

double f(double x, double y)
{ if (-70.0<x && x<-20.0 && -25.0<y && y<25.0)
    return 200.0;
  if (sqrt(elev2(x-50.0)+elev2(y))<30.0) return 250.0;
  if (sqrt(elev2(x)+elev2(1.5*y))<95.0) return 130.0;
  else return 0.0;
}

int main()
{ int tam=100; int tam21=2*tam+1;

  IMGGRY a(tam21, tam21, 0);
  for (int l=-tam; l<=tam; l++)
    for (int c=-tam; c<=tam; c++) {
      double t=f(c,-l);
      a(l+tam,c+tam)=double2G(t);
    }
  imp(a,"phantom.tga");
}
```

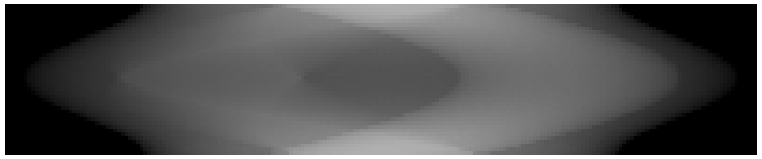
Geração de sinograma:

```
#include <proeikon>
int main()
{ IMGGRY a;
  le(a,"phantom.tga");
  if (a.nl()!=a.nc()) erro("Erro: phantom deve ser quadrada");
  if (a.nl()%2!=1) erro("Erro: nl nc devem ser impares");
  int tam=a.nl()/2; int tam21=a.nl(); int angulos=40;
  IMGGRY b(angulos, a.nc(), 0);
  for (int t=0; t<angulos; t++) {
    double theta=t*M_PI/angulos;
    double c=cos(theta); double s=sin(theta);
    for (int l=-tam; l<=tam; l++) {
      double soma=0.0;
      for (int p=tam; p>=-tam; p--) {
        double x=p*c+l*s; double y=-p*s+l*c;
        soma=soma+a(x,y,'C');
      }
      b(angulos-1-t,l+tam)=double2G(soma/tam21);
    }
  }
  imp(b,"sinogram.tga");
}
```

Rotação por θ: $\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \times \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$, onde $\begin{cases} c = \cos(\theta) \\ s = \text{sen}(\theta) \end{cases}$

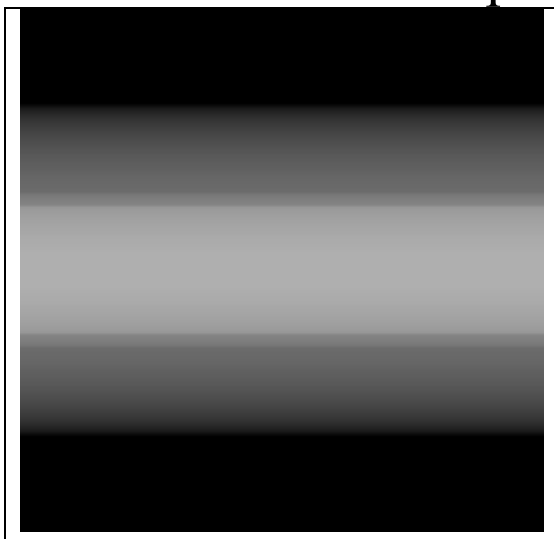


Phantom
(coeficientes de atenuação)

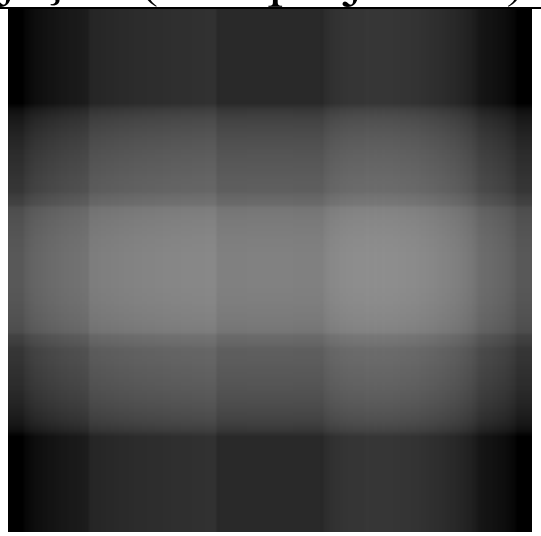


Sinograma

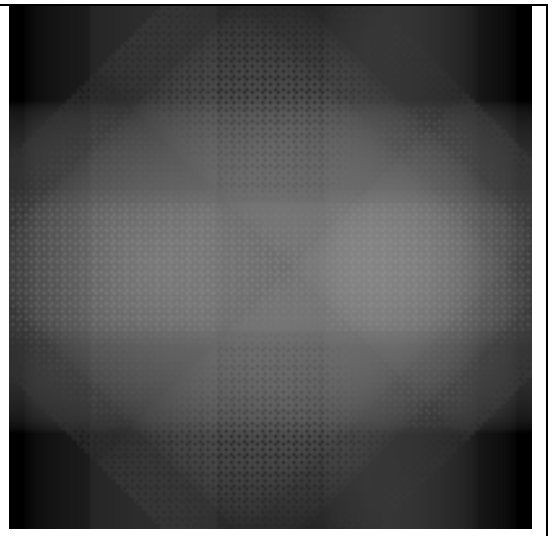
Primeira idéia: retroprojeção (backprojection)



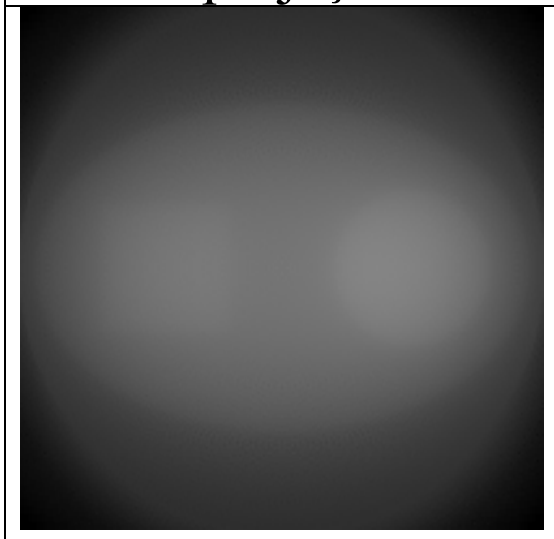
1 projeção



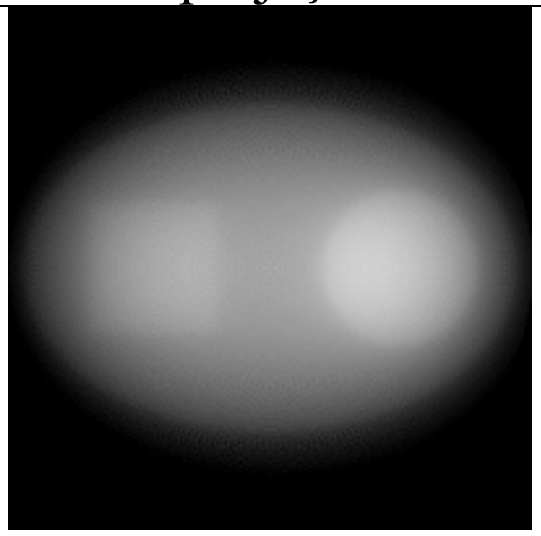
2 projeções



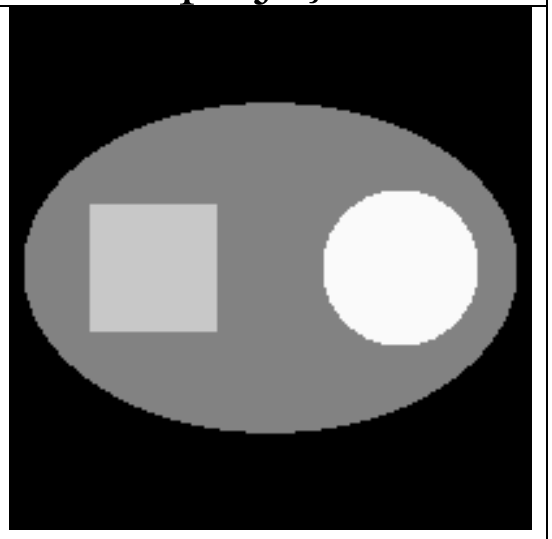
4 projeções



100 projeções



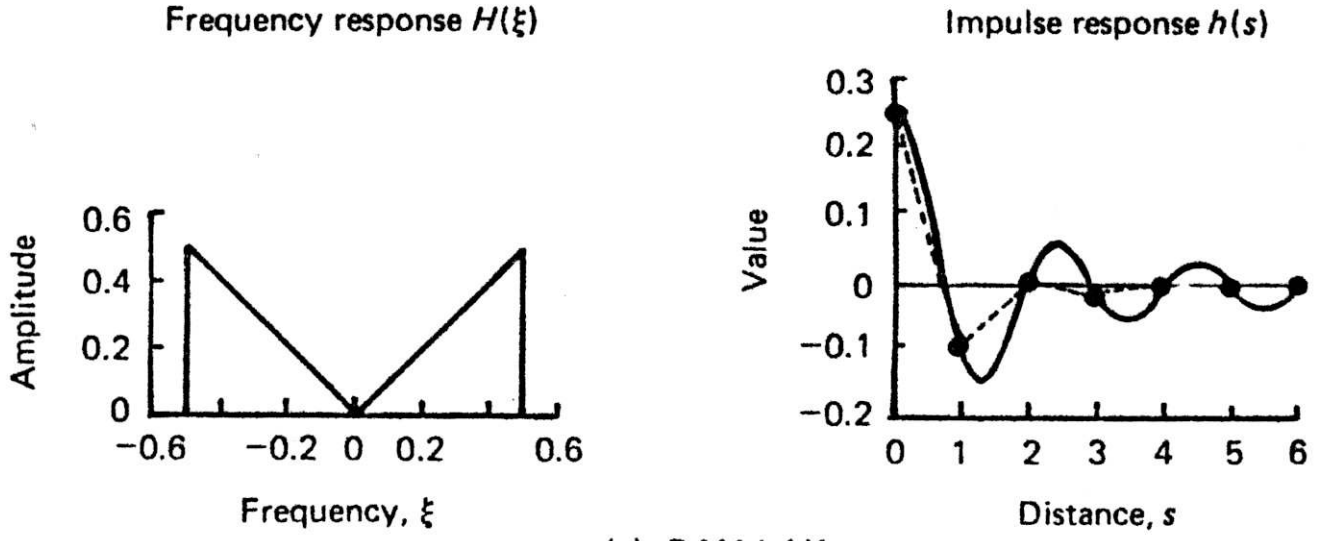
Aj. brilho/contraste



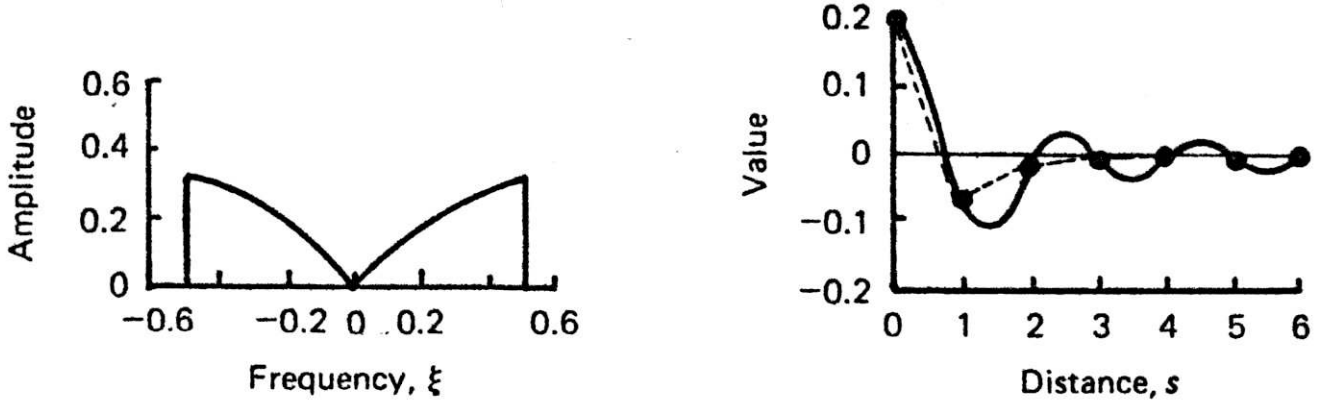
Phantom original

```
#include <proeikon>
int main()
{ IMGFLT sino; le(sino,"sinogram.tga"); sino.backg()=0.0;
  int tam21=sino.nc();
  if (tam21%2!=1) erro("Numero de raios deve ser impar");
  int tam=tam21/2; int angulos=sino.nl();
  // Filtrar aqui vetor "sino" para obter filtered backprojection
  IMGFLT peso(sino.nc(),sino.nc(),0.0);
  IMGFLT reco(sino.nc(),sino.nc(),0.0);
  for (int t=0; t<angulos; t=t+1) {
    double theta=t*M_PI/angulos;
    double c=cos(theta); double s=sin(theta);
    for (int l=-tam; l<=tam; l++) {
      for (int p=tam; p>=-tam; p--) {
        double x=p*c+l*s; double y=-p*s+l*c;
        int xi=arredonda(x); int yi=arredonda(y);
        if (-tam<=xi && xi<=tam && -tam<=yi && yi<=tam) {
          double p=peso(xi,yi,'C');
          reco(xi,yi,'C')=
            ( p*reco(xi,yi,'C') + sino(angulos-1-t,tam+l) ) / (p+1.0);
          peso(xi,yi,'C') = p+1;
        }
      }
    }
  }
  imp(reco,"backp.tga");
}
```

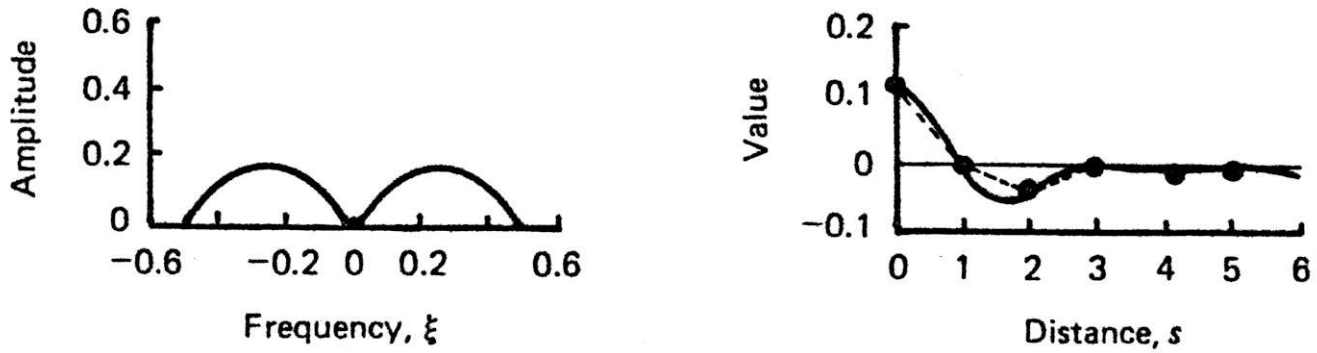
Reconstrução borrada. Precisa filtrar.



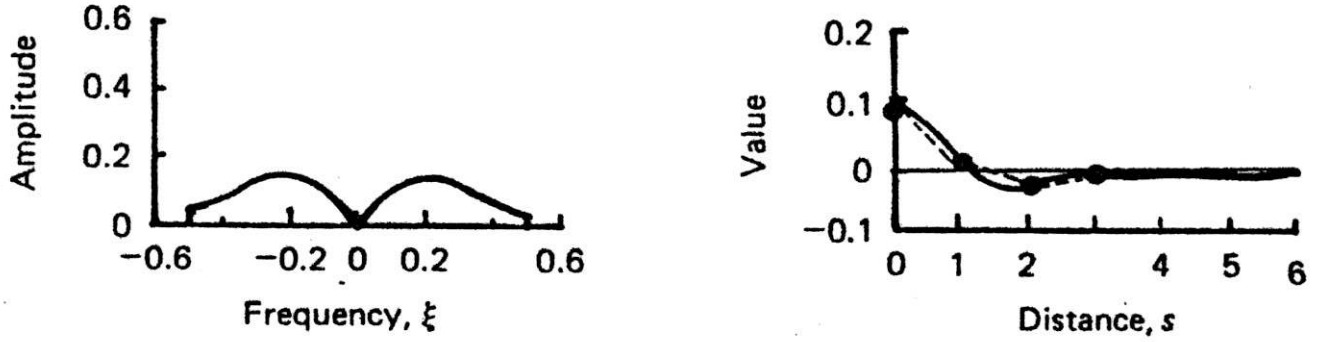
(a) RAM-LAK



(b) Shepp-Logan



(c) Lowpass cosine



(d) Generalized hamming

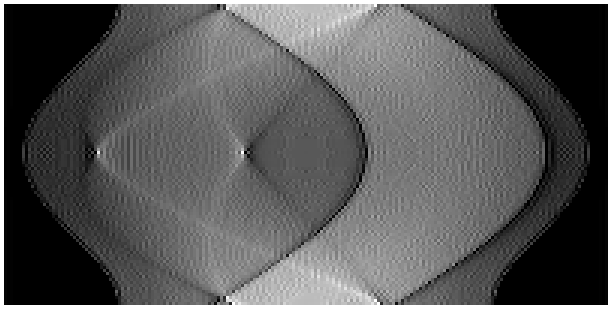
Figure 10.10 Reconstruction filters. Left column: Frequency response; right column: Impulse response; dotted lines show linearly interpolated response.

10.3 Filter Functions for Convolution/Filter Back-Projection Algorithms, $d \triangleq 1/2\xi_0$

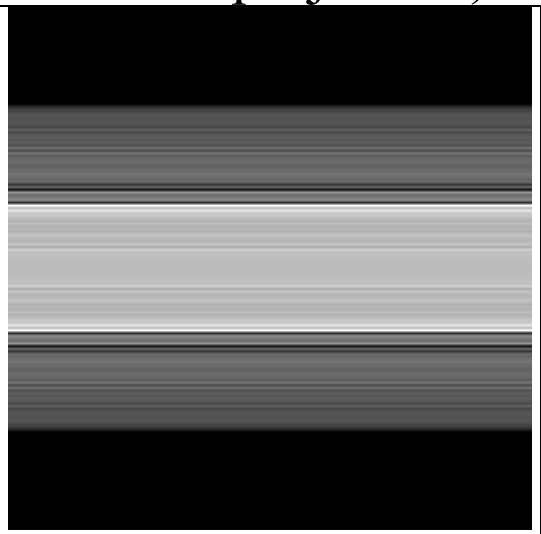
ter	Frequency response $H(\xi)$	Impulse response $h(s)$	Discrete impulse response $\hat{h}(m) \triangleq dh(md)$
Ram - Lak	$H_{RL}(\xi) \triangleq \lvert\xi\rvert \operatorname{rect}(\xi d)$	$h_{RL}(s) = \xi_0^2 \cdot [2\operatorname{sinc}(2\xi_0 s) - \operatorname{sinc}^2(\xi_0 s)]$	$\hat{h}_{RL}(m) = \begin{cases} \frac{1}{4d}, & m = 0 \\ \frac{-\sin^2(\pi m/2)}{\pi^2 m^2 d}, & m \neq 0 \end{cases}$
Shepp - Logan	$\lvert\xi\rvert \operatorname{sinc}(\xi d) \operatorname{rect}(\xi d)$	$\frac{2(1 + \sin 2\pi\xi_0 s)}{\pi^2(d^2 - 4s^2)}$	$\frac{2}{\pi^2 d(1 - 4m^2)}$
Low-pass (cosine)	$\lvert\xi\rvert \cos(\pi\xi d) \operatorname{rect}(\xi d)$	$\frac{1}{2}\left[h_{RL}\left(s - \frac{d}{2}\right) + h_{RL}\left(s + \frac{d}{2}\right)\right]$	$\frac{1}{2}[h_{RL}(m - \frac{1}{2}) + h_{RL}(m + \frac{1}{2})]$
Generalized Hamming	$\lvert\xi\rvert[\alpha + (1 - \alpha)\cos 2\pi\xi d] \cdot \operatorname{rect}(\xi d), 0 \leq \alpha \leq 1$	$\alpha h_{RL}(s) + \frac{1 - \alpha}{2} \cdot [h_{RL}(s - d) + h_{RL}(s + d)]$	$\alpha h_{RL}(m) + \left(\frac{1 - \alpha}{2}\right) \cdot [h_{RL}(m - 1) + h_{RL}(m + 1)]$
stochastic	See eq. (10.70) and Example 10.6		

```
// Janela Hamming - selecione alpha razoavel
// alpha=1.0 => vira ramlak
// Fator faz correcao de brilho
double alpha=0.9;
double cinza=4.0;
VETOR<CPXD> b(tam21);
for (int i=-tam; i<=tam; i++) {
  double x=abs(double(i)/tam);
  x=cinza*x*(alpha+(1.0-alpha)*cos(2.0*M_PI*i/tam));
  b(i+tam)=CPXD( x, 0.0);
}
b=ifftshift(b);
b=infft(b);
b=fftshift(b);
int mtam=10; int mtam21=2*mtam+1;
VETOR<double> mascara(mtam21);
for (int i=-mtam; i<=mtam; i++) mascara(mtam+i)=real(b(tam+i));
sino=ConvH(sino,mascara);
```

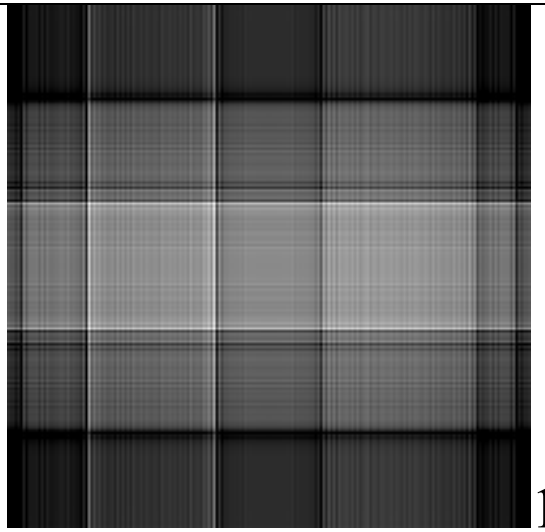
Retroprojeção filtrada (filtered backprojection)



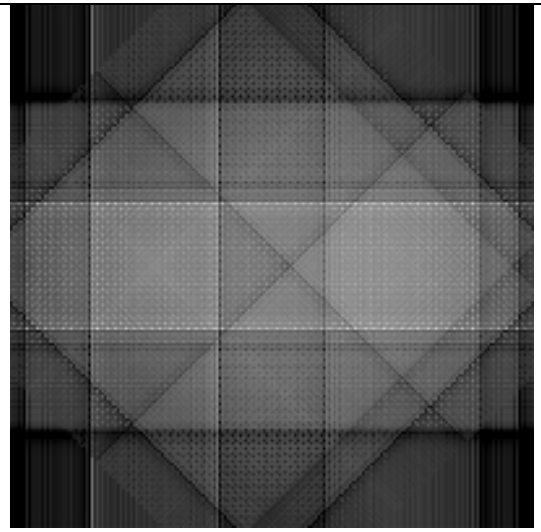
Sinograma filtrada



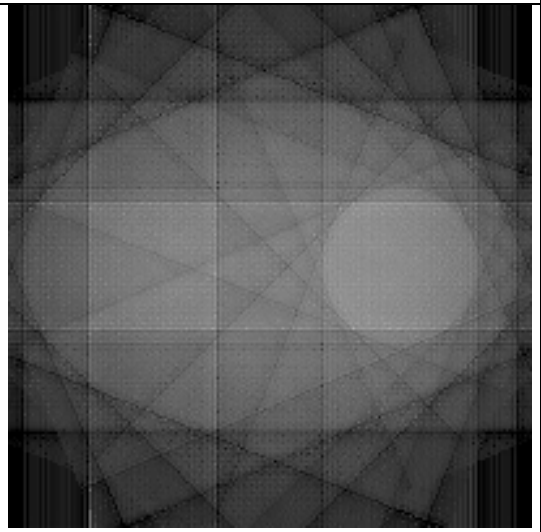
1 projeção



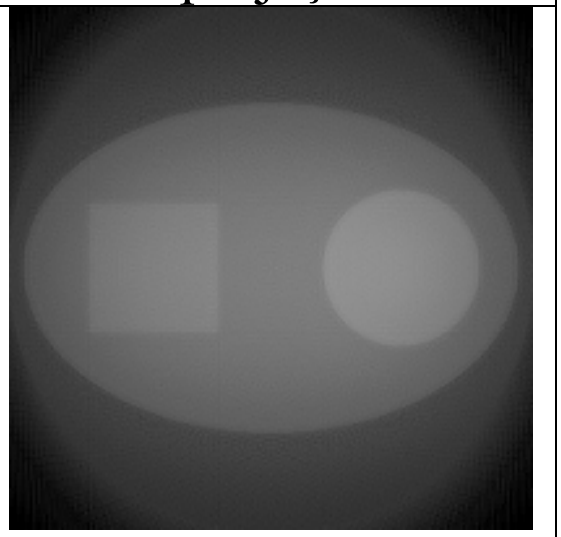
2 projeções



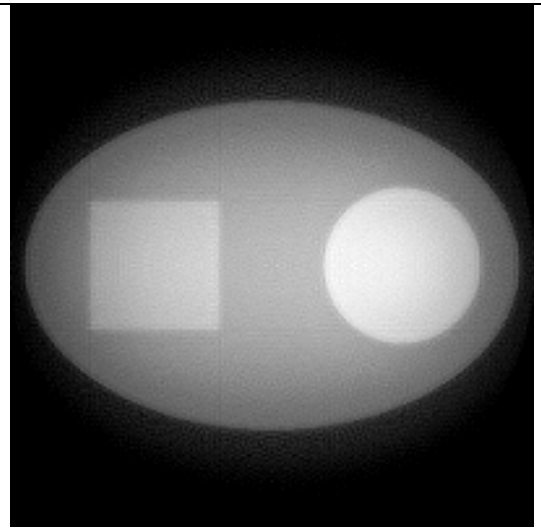
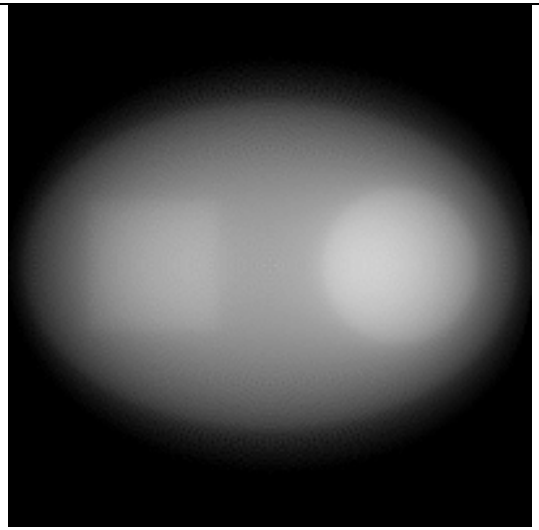
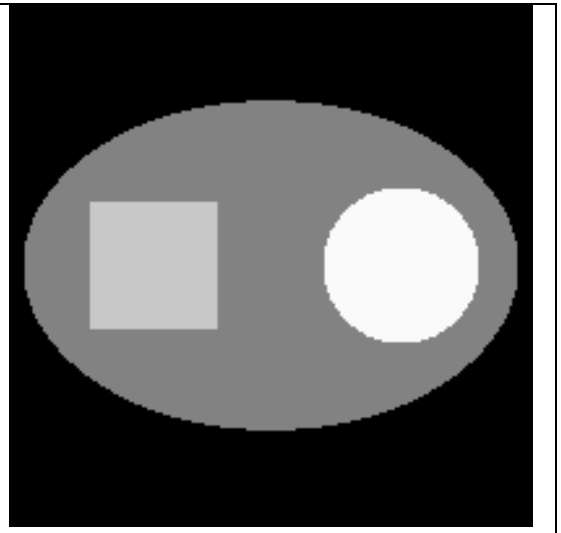
4 projeções



8 projeções



100 projeções

		
Aj. brilho/contraste	Backproj. sem filtro	Phantom original

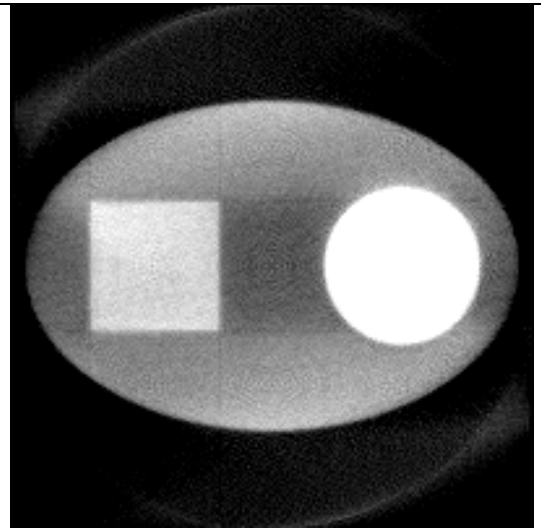
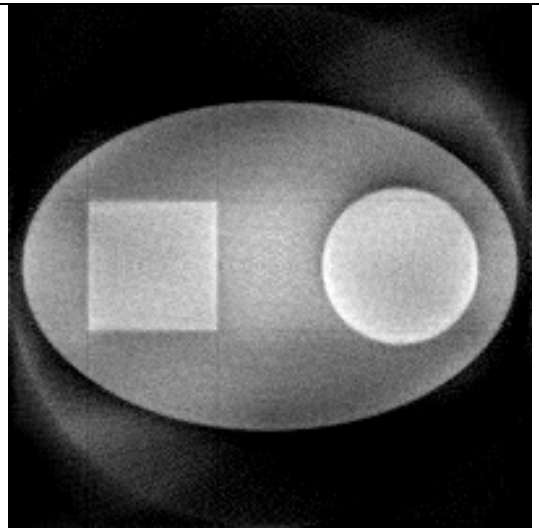
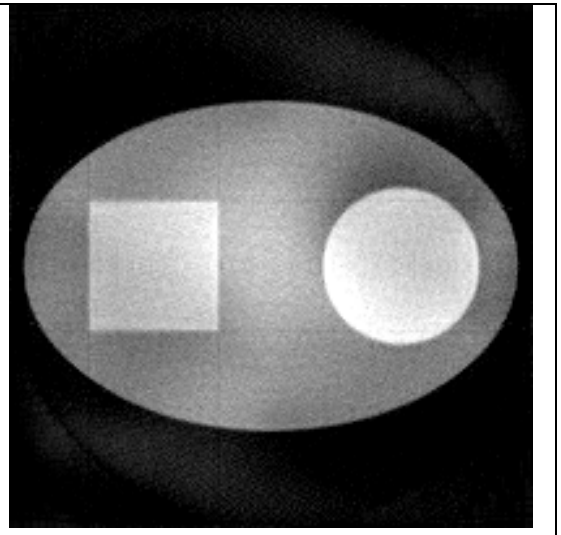
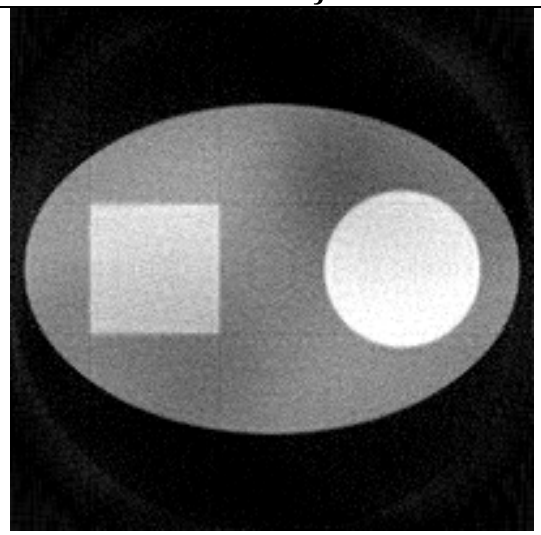
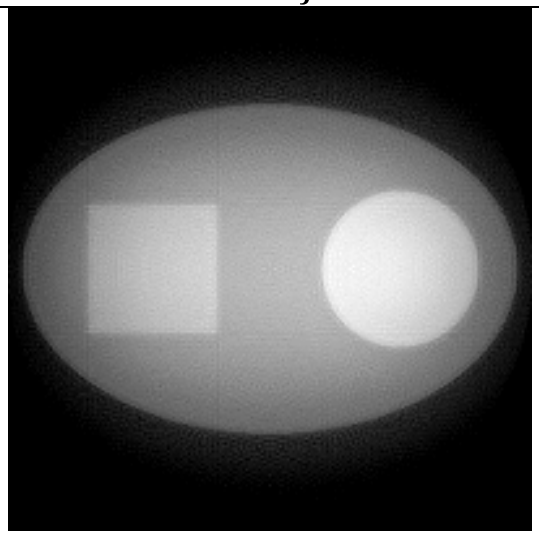
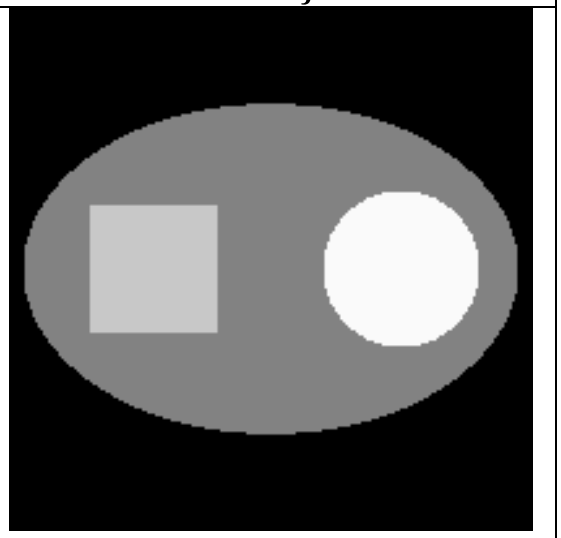
Reconstrução Algébrica (ART)

7	10	(17)			?	?	(17)		
6	8	(14)			?	?	(14)		
(13)	(18)				(13)	(18)			
7.75	7.75	15.5	(17)	↑1.5	8.5	8.5	(17)		
7.75	7.75	15.5	(14)	↓1.5	7	7	(14)		
(13)	(18)				15.5	15.5			
					(13)	(18)			
					↓2.5	↑2.5			
7.25	9.75	17	(17)						
5.75	8.25	14	(14)						
13	18								
(13)	(18)								

Note que tem 4 equações e 4 incógnitas. Mas as equações são LD.
Normalmente, o erro é distribuído apenas parcialmente ($0<\lambda<1$).

```
#include <proeikon>

int main()
{
  IMGGRY gray; le(gray,"sinogram.tga");
  IMGFLT sino=gray; sino.backg()=0.0;
  int tam21=sino.nc();
  if (tam21%2!=1)
    erro("Numero de raios deve ser impar");
  int tam=tam21/2; int angulos=sino.nl();

  IMGFLT reco(sino.nc(),sino.nc(),0.5);

  int iteracoes=8;
  for (int i=0; i<iteracoes; i++) {
    for (int t=0; t<angulos; t++) {
      double theta=t*M_PI/angulos;
      double c=cos(theta);
      double s=sin(theta);
```

```
      for (int l=-tam; l<=tam; l++) {
        double soma=0.0;
        for (int p=tam; p>=-tam; p--) {
          double x=p*c+l*s; double y=-p*s+l*c;
          int xi=int(floor(x+0.5));
          int yi=int(floor(y+0.5));
          if (-tam<=xi && xi<=tam &&
              -tam<=yi && yi<=tam)
            soma+=reco(tam-yi,tam+xi);
        }
        double dist=sino(angulos-1-t,tam+l)-soma/tam21;
        for (int p=tam; p>=-tam; p--) {
          double x=p*c+l*s; double y=-p*s+l*c;
          int xi=int(floor(x+0.5));
          int yi=int(floor(y+0.5));
          if (-tam<=xi && xi<=tam &&
              -tam<=yi && yi<=tam)
            reco(tam-yi,tam+xi)+=dist;
        }
      }
    }
  }
  gray=reco; imp(gray,"aritmet.tga");
}
```

		
1 iteração	2 iterações	4 iterações
		
8 iterações	Filt. backproj.	Phantom original

The Algebraic Reconstruction Technique (ART) is an iterative algorithm used in tomography. Gordon et al. (1970) first showed its use in image reconstruction. ART can be considered as an iterative solver of a system of linear equations ***Ax=b***. The values of reconstructed image pixels are the variables of the vector ***x*** (the attenuation coefficients μ(x, y)), each row a_i of the matrix ***A*** is the path of the *i*-th gamma ray (the geometry of ray *r*) and the measured projections form the vector ***b***. The formula below computes an approximation of the solution of the linear systems of equations:

$$x^{k+1} = x^k + \lambda_k \frac{b_i - \langle a_i, x^k \rangle}{\|a_i\|^2} (a_i)^t$$

where $i = (k \bmod m) + 1$ and λ_k is a relaxation parameter. In simple case, the matrix ***A*** consists of only 1's and 0's:

$$a_{ij} = \begin{cases} 1, & \text{if ray } i \text{ passes through pixel } j. \\ 0, & \text{otherwise.} \end{cases}$$

and $\|a_i\|^2$ becomes the number of pixels that the ray *i* passes through (number of 1's in a_i). In this simple case, MART can be written as:

$$x^{k+1} = x^k \otimes \left[\left(\frac{b_i}{\langle a_i, x^k \rangle} \right)^{\lambda_k} (a_i)^t \right]$$

where ⊗ indicates Hadamard product (elementwise product). While ART converges in the consistent case to a minimal norm solution, MART is designed to converge to a solution that minimizes the entropy (Natterer 1998).

Replacing $i = (k \bmod m) + 1$ by a randomly chosen indice, we obtain randomized ART/MART that we actually use in our experiments. Moreover, in our implementation, the relaxation parameter λ_k decreases as the iteration step *k* increases. We noted experimentally that this improves the quality of the reconstruction.

Reconstrução Algébrica Multiplicativa (MART)

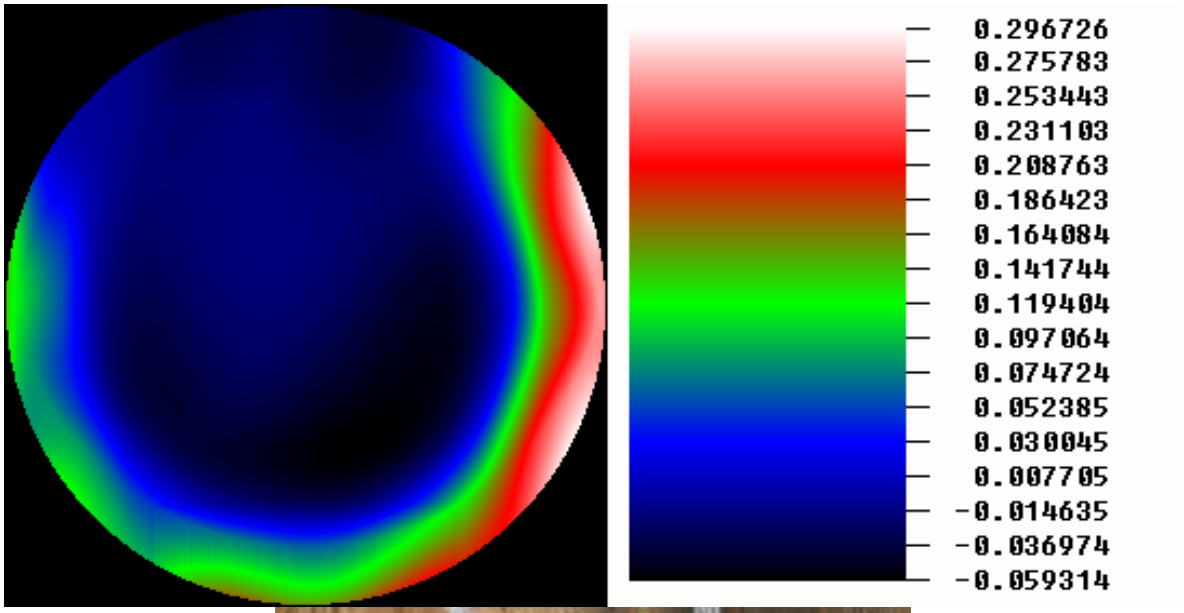
7	10	(17)			?	?	(17)		
6	8	(14)			?	?	(14)		
(13)	(18)				(13)	(18)			
7.75	7.75	15.5	(17)	*1.097	8.5	8.5	(17)		
7.75	7.75	15.5	(14)	*0.903	7	7	(14)		
(13)	(18)				15.5	15.5			
					(13)	(18)			
					*0.839	*1.161			
7.13	9.87	17	(17)						
5.87	8.13	14	(14)						
13	18								
(13)	(18)								

Note que tem 4 equações e 4 incógnitas. Mas as equações são LD.
Normalmente, o erro é distribuído apenas parcialmente ($0<\lambda<1$).

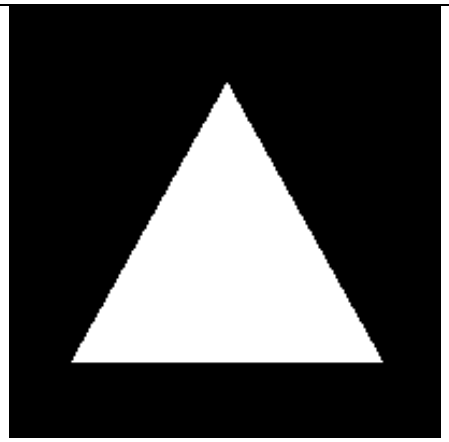
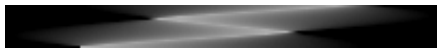
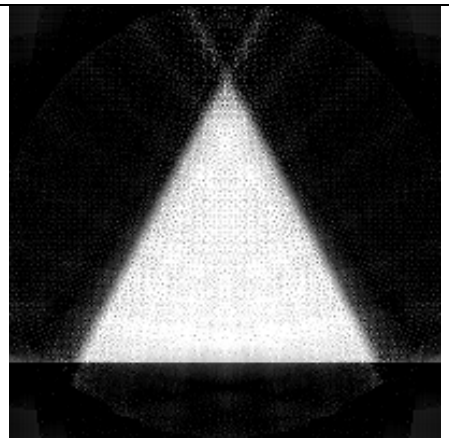
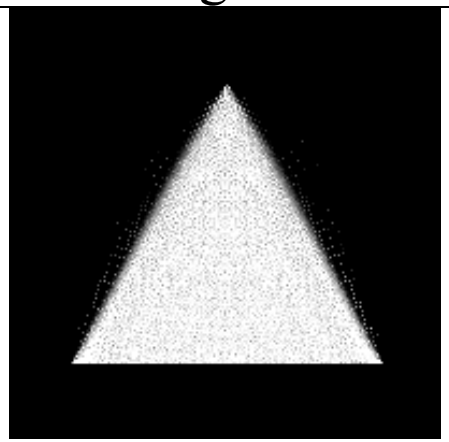
Reconstrução Algébrica Multiplicativa (com médias)

7	10	(8.5)			?	?	(8.5)		
6	8	(7)			?	?	(7)		
(6.5)	(9)				(6.5)	(9)			
7.75	7.75	7.75	(8.5)	*1.097	8.5	8.5	(8.5)		
7.75	7.75	7.75	(7)	*0.903	7	7	(7)		
(6.5)	(9)				7.75	7.75			
					(6.5)	(9)			
					*0.839	*1.161			
7.13	9.87	8.5	(8.5)						
5.87	8.13	7	(7)						
6.5	9								
(6.5)	(9)								

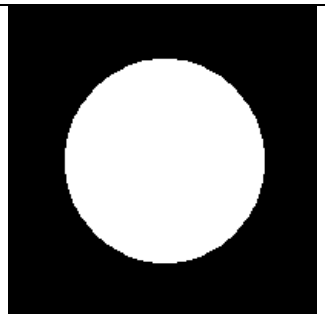
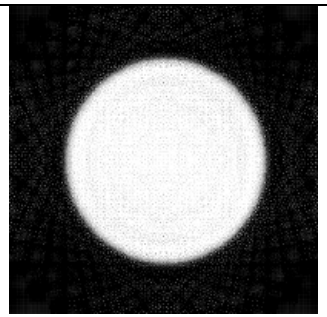
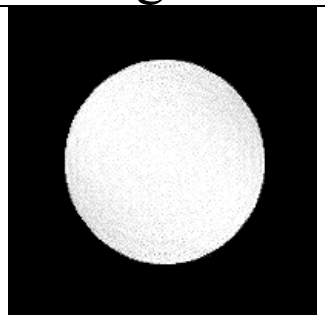
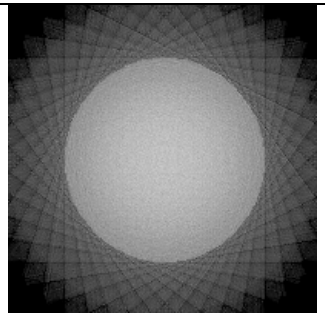
Nota: MART não pode ter valores negativos.



Reconstrução industrial de raio gama com falsa cor

 original	 sinograma com 20 linhas (projeções)	 ART
 MART	 ART com filtragem	 MART com filtragem

MART gera imagem mais nítida que ART se a imagem original tem contraste alta.

 original	 sinograma com 20 linhas (projeções)	 ART
 MART	 Filtered backprojection	