Discrete Tomography as 2-D Phase Retrieval Solved Using Hybrid Input-Output Algorithm

Andrew E. Yagle
Department of EECS, The University of Michigan, Ann Arbor, MI 48109-2122

Abstract— **The discrete tomography problem is to reconstruct a binary image from a limited set of its projections (sums of subsets of pixels). We formulate it as a one-bit phase retrieval problem and solve it using the (iterative) hybrid input-output algorithm of Fienup. Numerical examples illustrate the algorithm.**

Keywords— **Binary image, discrete tomography**
Phone: 734-763-9810. Fax: 734-763-1503.
Email: aey@eecs.umich.edu. **EDICS: 2-REST.**

I. INTRODUCTION

A. Problem Statement

The goal is to reconstruct a binary image x_{ij} from a limited set of projections (sums of subsets of x_{ij})

- $y_n = \sum_{i,j} H_{n,i,j} x_{ij}$ is underdetermined;
- x_{ij}=zero or one everywhere (binary image);
- $H_{n,i,j}$ is a sparse binary matrix (sums of x_{ij}).

We also generalize to arbitrary underdetermined linear combinations of x_{ij} (arbitrary matrices $H_{n,i,j}$).

B. Review of Discrete Tomography

The discrete tomography problem is to reconstruct a binary image from its row and column sums. This is equivalent to reconstructing the image from its discrete Fourier transform (DFT) values on the axes.

The solution to this problem is not unique. But all solutions are related to each other by "switches" of four pixels at the corners of rectangles, as follows:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{1}$$

It is easy to see that any such switch leaves the row and column sums unchanged, for another solution.

This can be solved using a greedy algorithm:

- Let argmaxima of row and column sums be (I,J);
- Set $x_{IJ} = 1$ and subtract one from the (I,J) sums;
- Continue until the binary image x_{ij} is finished.

However, if additional projection information is available, then this approach usually no longer works. We will make no attempt to review all approaches here.

C. Review of Hybrid Input-Output Algorithm

The hybrid input-output algorithm (HIO) was developed by Fienup to solve the phase retrieval problem of reconstructing an image from knowledge of the magnitude of its DFT and image information, e.g.,

- *Support:* x_{ij}=0 outside a known region;
- *Non-negativity:* $x_{ij} \geq 0$ everywhere;
- *Symmetry:* x_{ij} has rotational symmetry.

All of these are convex sets and also subspaces.

The HIO algorithm is iterative and has the form:

$$x \leftarrow x + \mathcal{Q}(2\mathcal{P}(x) - x) - \mathcal{P}(x) \quad \text{where} \tag{2}$$

- $\mathcal{P}x$ implements the image constraint on x;
- $\mathcal{Q}x$ sets DFT magnitude of x to its known value;
- These projections can be exchanged.

II. REFORMULATION OF DISCRETE TOMOGRAPHY AS PHASE RETRIEVAL

A. Discrete Projection-Slice Theorem

The projection-slice theorem of tomography states that projections of an image are samples of its Fourier transform along radial lines. The discrete projection-slice theorem is derived from the 2-D DFT, which is:

$$X_{k_1,k_2} = \sum_{i_1=0}^{N-1} \sum_{i_2=0}^{N-1} x_{i_1,i_2} e^{-j\frac{2\pi}{N}(i_1 k_1 + i_2 k_2)} \tag{3}$$

Setting $k_2 = 0$ gives the 1-D DFT of row sums:

$$X_{k_1,0} = \sum_{i_1=0}^{N-1} e^{-j\frac{2\pi}{N}(i_1 k_1)} \sum_{i_2=0}^{N-1} x_{i_1,i_2} \tag{4}$$

Setting $k_1 = 0$ gives the 1-D DFT of column sums:

$$X_{0,k_2} = \sum_{i_2=0}^{N-1} e^{-j\frac{2\pi}{N}(i_2 k_2)} \sum_{i_1=0}^{N-1} x_{i_1,i_2} \tag{5}$$

For more projections, setting $k_1 = k_2 = k$ gives

$$X_{k,k} = \sum_{i_1=0}^{N-1} \sum_{i_2=0}^{N-1} x_{i_1,i_2} e^{-j\frac{2\pi k}{N}(i_1 + i_2)} \tag{6}$$

which is the 1-D DFT of the 135^o projection

$$p_n = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} x_{i_1,i_2}\delta[n-(n_1+n_2)\mathrm{mod}(N)] \qquad (7)$$

Similarly, setting $k_1 = -k_2 = k$ gives

$$X_{k,-k} = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} x_{i_1,i_2} e^{-j\frac{2\pi k}{N}(i_1-i_2)} \qquad (8)$$

which is the 1-D DFT of the 45^o projection

$$p_n = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} x_{i_1,i_2}\delta[n-(n_1-n_2)\mathrm{mod}(N)] \qquad (9)$$

Thus knowledge of various projections of x_{ij} is equivalent to knowledge of the 2-D DFT of x_{ij} along radial lines of (k_1, k_2) at angles of $45^o, 90^o, 135^o$ and 180^o.

Note that the 45^o and 135^o projections are aliased; to avoid this we must zero-pad the 2-D DFT. This would make the iterative HIO algorithm much more difficult to implement, since the forward and inverse 2-D DFTs could no longer be used exclusively here.

B. Binary Image as One-Bit Phase

The binary image $x_{ij} = \{0, 1\}$ can be mapped to a bilevel image $x_{ij} = \pm 1$ by simply converting the data:

- **General:** $y_i \leftarrow 2y_i - \sum_{j=1}^{N} h_{ij}$
- **Row sums:** $y_i \leftarrow 2y_i - N$;
- **Col. sums:** $y_i \leftarrow 2y_i - N$;
- 45^o **sums:** $y_i \leftarrow 2y_i - (N - |N - i|)$;
- 135^o **sums:** $y_i \leftarrow 2y_i - (N - |N - i|)$.

Double each projection and subtract off the number of elements being summed in that projection.

Then the projection of x_{ij} onto ± 1 is performed by

$$x_{ij} \leftarrow \mathrm{sign}[x_{ij}] = 1e^{j\arg[x_{ij}]} \qquad (10)$$

which is the same as the projection $\mathcal{Q}$ in the HIO algorithm of setting the magnitude of x_{ij} to its known value (here, unity everywhere): $|x_{ij}| = 1$ everywhere.

C. Application of HIO to Discrete Tomography

The HIO algorithm can thus be applied to discrete tomography by reformulation as phase retrieval:

1. Initialize with random x_{ij};
2. $X_{k_1,k_2} = \mathrm{DFT}[x_{ij}]$ (see (3));
3. Set $X_{k,0}, X_{0,k}, X_{k,k}, X_{k,-k}$ to known values;
4. $\mathcal{P}(x_{ij}) = \mathrm{DFT}^{-1}[X_{k_1,k_2}]$;
5. x←x+sign[$2\mathcal{P}(x_{ij})$–x]–$\mathcal{P}(x_{ij})$
6. Continue until convergence (x_{ij} stops changing).

D. Application of HIO to General Matrix H

The HIO algorithm can also be applied to (1) with a more general matrix H. Given data y_n as defined in (1), we now redefine the projection $\mathcal{P}(x)$ to be

$$\mathcal{P}(x) = x + H^T(HH^T)^{-1}(y - Hx) \qquad (11)$$

which is the projection of x onto the affine hyperplane $y = Hx$. If the matrix H does not have full row rank, $H^T(HH^T)^{-1}$ must be replaced by the appropriate pseudo-inverse, as in the second example below.

The problem with this is that computation of the 2-D DFT is relatively fast, requiring $N^2 \log_2 N$ computations. However, the matrix-vector multiplications each require MN^2 computations, where M is the number of observations, summed over all of the projections. And of course the pseudo-inverse must be computed and stored, or a linear system of equations solved at each recursion. Hence the HIO algorithm seems much more suitable for discrete tomography.

III. NUMERICAL EXAMPLES

A. From Radial Fourier Samples

The 2-D DFT values $\{X_{k,0}, X_{0,k}, X_{k,k}, X_{k,-k}\}$ are equivalent to the *aliased* projections at $45^o, 90^o, 135^o$ and 180^o (see above). However, his is not enough data to ensure a unique solution; the two images

$$\begin{bmatrix} 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&1&0&0&0&1\\ 1&0&0&0&1&0\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0 \end{bmatrix} \qquad \begin{bmatrix} 0&0&0&1&0&0\\ 0&0&1&0&0&0\\ 1&0&0&0&1&0\\ 0&1&0&0&0&1\\ 0&0&0&1&0&0\\ 0&0&1&0&0&0 \end{bmatrix}$$

have the same (unaliased) projections (note the four switches). Hence the problem is solved when a binary image whose projections agree with the data is found.

```
clear;N=10;N1=2*N-1;
X=round(rand(N,N));C=sum(X,1);R=sum(X,2);
for I=1-N:N-1;NE(N+I)=sum(diag(X,-I));
NW(N+I)=sum(diag(flipud(X),I));end
C1=2*C-N;NE1=2*NE-[1:N-1 N:-1:1];
R1=2*R-N;NW1=2*NW-[1:N-1 N:-1:1];
FR1=fft(R1);FNE1=fft(NE1(N:N1)+[0 NE1(1:N-1)]);
FC1=fft(C1);FNW1=fft(NW1(1:N)+[NW1(N+1:N1) 0]);
Z=randn(N,N);%Hybrid Input-Output Recursion:
for J=1:100;FZ=fft2(Z);FZ(:,1)=FR1;FZ(1,:)=FC1;
for I=1:N;FZ(I,I)=FNW1(I);end %Diagonal project
for I=2:N;FZ(I,N+2-I)=FNE1(I);end %Antidiagonal
PZ=real(ifft2(FZ));W=sign(2*PZ-Z);Z=Z+W-PZ;
FW=fft2(W);%Monitor L1 error in Fourier domain:
E1=sum(abs(FW(:,1)-FR1));E2=sum(abs(FW(1,:)-FC1));
E3=sum(abs(diag(FW)-FNW1.'));
E4=sum(abs(diag(fliplr(FW),-1)-FNE1(2:N).'));
E(J)=E1+E2+E3+E4;if(E(J)<0.001);break;end;end
W=(W+1)/2;W=reshape(W,N,N);
subplot(221),imagesc(X),axis off,colormap(gray)
subplot(222),imagesc(W),axis off,colormap(gray)
subplot(212),plot(E)
```

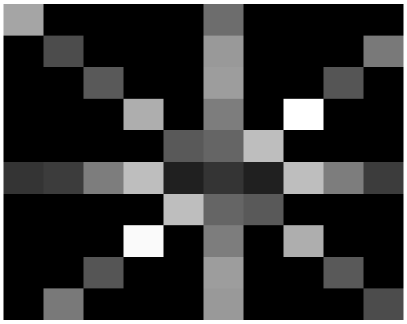

Fig. 1. Depiction of Radial Fourier Data

B. From Unaliased Projections

Here we formulate the problem as an underdetermined linear algebra problem $y = Hx$, where

- H is a binary matrix that sums subsets of x_{ij};
- HH^T is diagonal, with some zeros on it, hence:
- H is always row-rank-deficient by seven, since:
- The projections all have $X_{0,0}$ in common, and:
- $X_{0,0}$ is real-valued, for a deficiency of 2(4)–1=7;
- Hence the pseudo-inverse must be used in $\mathcal{P}$;
- H can be replaced with an arbitrary matrix below.

```
clear;N=10;N1=2*N-1;N2=N*N;
X=round(rand(N,N));C=sum(X,1);R=sum(X,2);
for I=1-N:N-1;NE(N+I)=sum(diag(X,I));
NW(N+I)=sum(diag(flipud(X),I));end
%Convert from [0,1] to [-1,1]:
C1=2*C-N;NE1=2*NE-[1:N-1 N:-1:1];
R1=2*R-N;NW1=2*NW-[1:N-1 N:-1:1];
Y=[NW1 NE1 R1' C1]';Z=randn(N2,1);
%Create projection matrix H:
%Note H is rank-deficient by 7.
for I=0:N-1;H(I+1:I+N,I*N+1:I*N+N)=eye(N);
H(N1+I+1:N1+I+N,I*N+1:I*N+N)=fliplr(eye(N));end
H=[H;[kron(ones(1,N),eye(N));kron(eye(N),ones(1,N))]];
PH=pinv(H);Z0=PH*Y;A=eye(N2)-PH*H;
%Hybrid Input-Output Recursion:
for J=1:100;PZ=A*Z+Z0;
W=sign(2*PZ-Z);Z=Z+W-PZ;
E(J)=sum(abs(Y-H*W));
if(E(J)<0.001);break;end;end
%Convert from [-1.1] to [0,1]:
W=(W+1)/2;W=reshape(W,N,N);
subplot(221),imagesc(X),axis off,colormap(gray)
subplot(222),imagesc(W),axis off,colormap(gray)
subplot(212),plot(E)
```

Typical runs of both programs are shown:

- The error tends to vary up and down for awhile, and then suddenly plunge down to zero (solution);
- The number of iterations required for convergence increases rapidly with problem size, but is finite;
- Original and reconstructed images are shown;
- The reconstructed image satisfies all projections;
- The reconstructed image is related to the original image by multiple switches of the kind shown above.

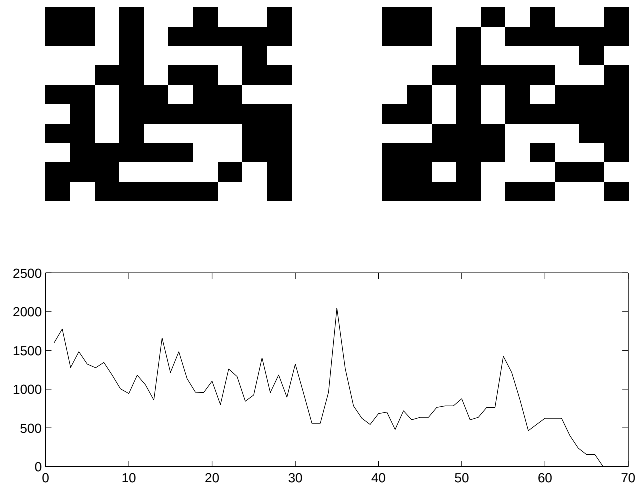

Fig. 2. Typical Run: Radial Fourier Data

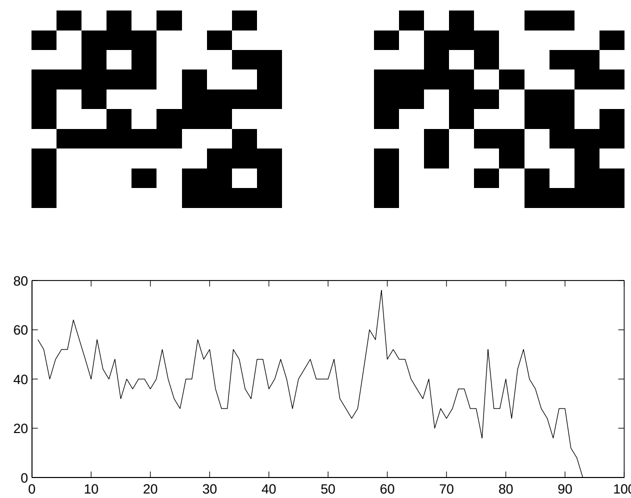

Fig. 3. Typical Run: Matrix Projection Data

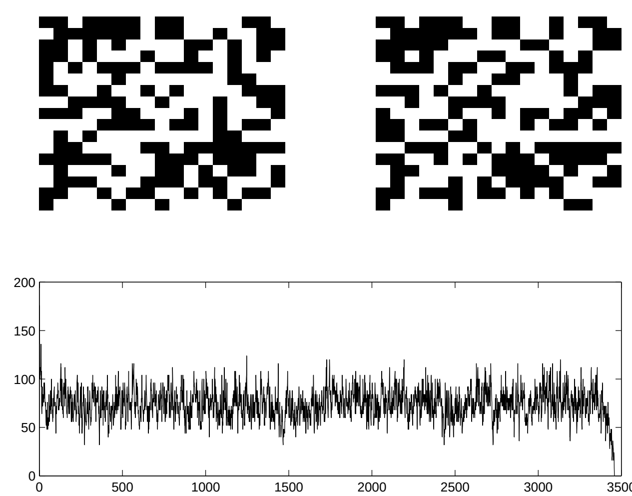

Fig. 4. Typical Run: Matrix Projection Data