



Tópicos sobre Sinais 2D

Processamento Digital de Sinal II

Artur Ferreira e Paulo Marques
(Dezembro 2003)



Tópicos a abordar

- n DFT2D – Discrete Fourier Transform
- n DCT2D – Discrete Cosine Transform
- n Conceito de transformada separável
- n Cálculo matricial
- n Aplicações

A DFT2D e IDFT2D

n A DFT2D define-se como

$$\begin{aligned} Y[k_1, k_2] &= DFT2D[x[m, n]] \\ &= \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x[m, n] e^{-\frac{j2\pi k_1 m}{N}} e^{-\frac{j2\pi k_2 n}{N}} \end{aligned}$$

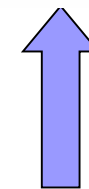
n A IDFT2D define-se como

$$\begin{aligned} x[m, n] &= IDFT2D[Y[k_1, k_2]] \\ &= \frac{1}{N^2} \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} Y[k_1, k_2] e^{\frac{j2\pi k_1 m}{N}} e^{\frac{j2\pi k_2 n}{N}} \end{aligned}$$

Cálculo matricial da DFT2D

n Atendendo a que

$$Y[k_1, k_2] = \sum_{m=0}^{N-1} e^{-\frac{j2\pi k_1 m}{N}} \sum_{n=0}^{N-1} x[m, n] e^{-\frac{j2\pi k_2 n}{N}}$$



DFT 1D sobre as linhas
da imagem x

Tem-se que a transformada é separável, ou seja, calcula-se a DFT 1D sobre as linhas da imagem. Em seguida, calcula-se a DFT 1D sobre esse resultado. Em notação matricial, fica-se com

$$Y = DFT2D[X] = \mathbf{F}\mathbf{X}\mathbf{F}^T,$$

sendo F o operador que calcula a DFT 1D. Dado que F é matriz simétrica:

$$Y = \mathbf{F}\mathbf{X}\mathbf{F}$$



DFT2D: transformada separável

- n Conclui-se que a DFT2D pode ser calculada à custa de duas DFT1D – é uma **transformada separável**
- n A FFT é um algoritmo rápido para o cálculo da DFT 1D
- n Logo, a DFT2D pode ser calculada através da FFT
- n Esta é a opção seguida na implementação do MATLAB

Troço de código da função `fft2`

A função MATLAB `fft2` calcula a DFT2D, através de duas chamadas à `fft` (**1D**)

```
function f = fft2(x, mrows, ncols)  
  
%FFT2 Two-dimensional discrete Fourier Transform.  
% FFT2(X) returns the two-dimensional Fourier transform of matrix X.  
% If X is a vector, the result will have the same orientation.  
%  
% FFT2(X,MROWS,NCOLS) pads matrix X with zeros to size MROWS-by-NCOLS  
% before transforming.  
% See also IFFT2, FFT, FFTSHIFT.  
(....)  
    if nargin==1  
        f = fft(fft(x,[],2),[],1);  
    else  
        f = fft(fft(x,ncols,2),mrows,1);  
    end  
end
```

DFT2D: exemplos de cálculo

Imagem DC

$$\mathbf{x1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$\text{fft2}(\mathbf{x1}) =$

$$\begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Linhas: Sempre iguais (frequência zero)

Colunas: Frequência máxima (\mathbf{p})

$$\mathbf{x2} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

$\text{fft2}(\mathbf{x2}) =$

$$\begin{bmatrix} 0 & 0 & 16 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

DFT2D: exemplos de cálculo

Linhas: Frequência máxima (p)

Colunas: Sempre iguais (frequência zero)

$$x3 = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\text{fft}(x3) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Linhas: Frequência máxima (p)

Colunas: Frequência máxima (p)

$$x4 = \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

$$\text{fft}(x4) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -16 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Cálculo matricial da IDFT2D

n Atendendo a que

$$1. Y = DFT2D[X] = \mathbf{F}X\mathbf{F}$$

$$2. \mathbf{F}^{-1} = \frac{1}{N}\mathbf{F}^*$$

Tem-se:

$$\begin{aligned}\mathbf{F}^{-1}\mathbf{Y}\mathbf{F}^{-1} &= \mathbf{F}^{-1}(\mathbf{F}X\mathbf{F})\mathbf{F}^{-1} \\ &= (\mathbf{F}^{-1}\mathbf{F})X(\mathbf{F}\mathbf{F}^{-1}) \\ &= X\end{aligned}$$

Logo:

$$\begin{aligned}X &= IDFT2D[Y] = \mathbf{F}^{-1}\mathbf{Y}\mathbf{F}^{-1} \\ &= \frac{1}{N^2}\mathbf{F}^*\mathbf{Y}\mathbf{F}^*\end{aligned}$$

DCT2D e IDCT2D N x N

- A DCT2D define-se como

$$F[u, v] = C[u]C[v] \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x[m, n] \cos\left(\frac{(2m+1)up}{2N}\right) \cos\left(\frac{(2n+1)vp}{2N}\right)$$

- A IDCT2D define-se como

$$f[m, n] = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C[u]C[v]F[u, v] \cos\left(\frac{(2m+1)up}{2N}\right) \cos\left(\frac{(2n+1)vp}{2N}\right)$$

$$C[l] = \begin{cases} \frac{1}{\sqrt{N}}, & l = 0 \\ \sqrt{\frac{2}{N}}, & l \neq 0 \end{cases}$$

DCT2D e IDCT2D 8x8

Utilizada no JPEG – codificação com perda, de imagem

DCT2D 8x8

$$F[u, v] = \frac{1}{4} C[u]C[v] \sum_{m=0}^7 \sum_{n=0}^7 f[m, n] \cos\left(\frac{(2m+1)up}{16}\right) \cos\left(\frac{(2n+1)vp}{16}\right)$$

IDCT2D 8x8

$$f[m, n] = \frac{1}{4} \sum_{u=0}^7 \sum_{v=0}^7 C[u]C[v] F[u, v] \cos\left(\frac{(2m+1)up}{16}\right) \cos\left(\frac{(2n+1)vp}{16}\right)$$

$$C[l] = \begin{cases} \frac{1}{\sqrt{2}}, & l = 0 \\ 1, & l \neq 0 \end{cases}$$

DCT2D e IDCT2D: cálculo matricial

À semelhança do que se demonstra para a DFT2D tem-se a separabilidade da transformada:

$$\mathbf{Y} = DCT2D[\mathbf{X}] = \mathbf{CXC}^T,$$

sendo \mathbf{C} o operador que calcula a DCT 1D. Ao contrário de \mathbf{F} (operador DFT1D), \mathbf{C} não é matriz simétrica. No entanto, \mathbf{C} é ortogonal: $\mathbf{C}^T = \mathbf{C}^{-1}$, $\mathbf{CC}^T = \mathbf{I}$, $\mathbf{C}^T\mathbf{C} = \mathbf{I}$.

$$\begin{aligned} \text{Tem-se: } \mathbf{C}^T\mathbf{Y}\mathbf{C} &= \mathbf{C}^T(\mathbf{CXC}^T)\mathbf{C} \\ &= (\mathbf{C}^T\mathbf{C})\mathbf{X}(\mathbf{CC}^T) \\ &= \mathbf{X} \end{aligned}$$

$$\mathbf{X} = IDCT2D[\mathbf{Y}] = \mathbf{C}^T\mathbf{Y}\mathbf{C}$$



DCT2D: transformada separável

- n Pode ser calculada à custa de duas DCT1D – é uma **transformada separável**
- n Dado que a DCT1D pode ser calculada via FFT tem-se que a DCT2D pode ser calculada através de duas FFT1D
- n Esta é a opção seguida na implementação em MATLAB

Troço de código da função dct2

A função MATLAB **dct2** calcula a DCT2D, através de duas chamadas à **dct (1D)** que por sua vez utiliza a **fft (1D)**

```
function b=dct2(arg1,mrows,ncols)
% DCT2 Compute 2-D discrete cosine transform.
% B = DCT2(A) returns the discrete cosine transform of A.
% The matrix B is the same size as A and contains the
% discrete cosine transform coefficients.
% References:
%
[m, n] = size(arg1);
% Basic algorithm.
if (nargin == 1),
    if (m > 1) & (n > 1),
        b = dct(dct(arg1).').';
        return;
    else
        (...)
```

Troço de código da função dct

A função MATLAB **dct** calcula a DCT1D, através da **fft (1D)**

```
function b=dct(a,n)
%DCT Discrete cosine transform.
%
% Y = DCT(X) returns the discrete cosine transform of
% X.
% The vector Y is the same size as X and contains the
% discrete cosine transform coefficients.
(...)

% Compute weights to multiply DFT coefficients
ww = (exp(-i*(0:n-1)*pi/(2*n))/sqrt(2*n)).';
ww(1) = ww(1) / sqrt(2);
if rem(n,2)==1 | ~isreal(a), % odd case
    % Form intermediate even-symmetric matrix
    y = zeros(2*n,m);
    y(1:n,:) = aa;
    y(n+1:2*n,:) = flipud(aa);
else % even case
    % Re-order the elements of the columns of x
    y = [ aa(1:2:n,:); aa(n:-2:2,:) ];
    yy = fft(y);
    ww = 2*ww; % Double the weights for even-length
    case
end

% Multiply FFT by weights:
b = ww(:,ones(1,m)) .* yy;
if isreal(a), b = real(b); end
if do_trans, b = b.'; end
```

DCT1D: cálculo matricial

```
function dct_matrix(N)

Alpha      = zeros(N,N);
Alpha(1,:) = 1/sqrt(N);
Alpha(2:N,:) = sqrt(2/N);

[n,k] = meshgrid ( 0:1:N-1, 0:1:N-1 );
C = Alpha .* cos( (pi*(2.*n + 1).*k) / (2*N) );

% % ----- DCT1D -----
x = [1 2 3 4]';
y = C * x
Ex = sum ( x.* x )
Ey = sum ( y.* y )
% dct(x)
% % ----- DCT1D -----
.....
```

```
>> dct_matrix(4)
```

```
y =
```

```
5.0000
-2.2304
-0.0000
-0.1585
```

```
Ex = 30
```

```
Ey = 30.0000
```

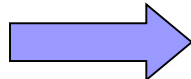
```
>>
```

A DCT1D preserva a energia
Transformada unitária

DCT2D: cálculo matricial

```
% ----- DCT2D -----  
x = magic(N)  
y = C * x * C'  
% ----- IDCT2D -----  
z = C' * y * C  
  
Ex = sum(sum ( x.* x ))  
Ey = sum(sum ( y.* y ))  
% ----- DCT2D -----
```

- A DCT2D preserva a energia
- Geralmente, concentra a energia num número reduzido de coeficientes
- Transformada unitária



```
>> dct_matrix(4)  
x =  
    16     2     3    13  
     5    11    10     8  
     9     7     6    12  
     4    14    15     1  
  
y =  
 34.0000   0.0000  -0.0000  -0.0000  
  0.0000   0.0000  13.5140   0.0000  
 -0.0000   3.3785   0.0000   2.9302  
 -0.0000  -0.0000  11.7206   0.0000  
  
z =  
 16.0000   2.0000   3.0000  13.0000  
  5.0000  11.0000  10.0000   8.0000  
  9.0000   7.0000   6.0000  12.0000  
  4.0000  14.0000  15.0000   1.0000  
  
Ex = 1496  
Ey = 1496
```

Cálculo da DCT2D

Imagem DC

$$x1 = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix}$$

dct2(x1)=

$$\begin{matrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

Variação (frequência)
apenas em coluna

$$x2 = \begin{matrix} 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \end{matrix}$$

dct2(x2) =

$$\begin{matrix} 0 & 1.53 & 0 & 3.69 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

Cálculo da DCT2D

Variação (frequência)
apenas em linha

$$x3 = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

dct2(x3) =

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -1.53 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -3.69 & 0 & 0 & 0 \end{bmatrix}$$

Variação (frequência)
em linha e em coluna

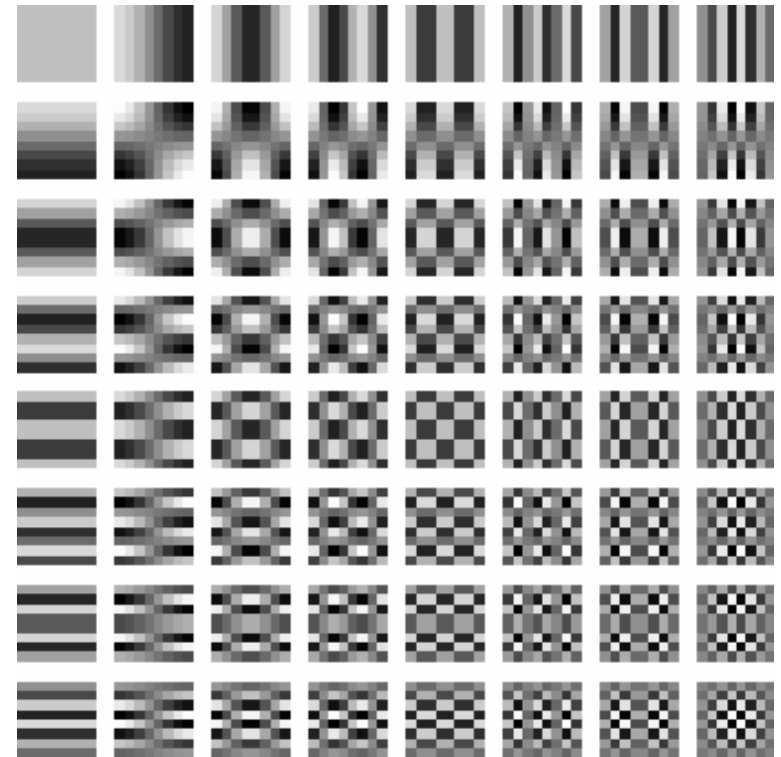
$$x4 = \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

dct2(x4) =

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -0.58 & 0 & -1.41 \\ 0 & 0 & 0 & 0 \\ 0 & -1.41 & 0 & -3.41 \end{bmatrix}$$

DCT 2D 8x8 – Imagens de base

```
pix=0;
N=8;
for v=0:N-1;
  for u=0:N-1;
    for y=0:N-1;
      for x=0:N-1;
        pix(v*(N+2)+1+y,u*(N+2)+1+x)=
          cos(((2*x+1)*u*pi)/(2*N))
            *cos(((2*y+1)*v*pi)/(2*N));
      end
    end
  end
end
mx=max(max(pix));mn=min(min(pix));
pix = pix-mn; pix = pix * 255 / (mx-mn);
colormap(gray); image(pix)
```



Qualquer imagem de resolução 8x8 pode ser obtida por combinação linear destas imagens de base. Os coeficientes da combinação linear são os coeficientes da DCT2D.

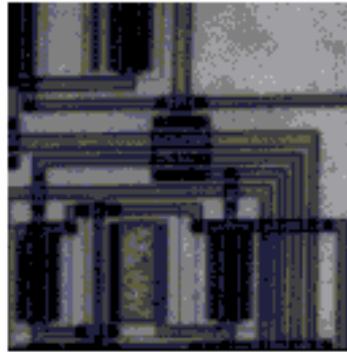
DCT2D: concentração de energia

No MATLAB:

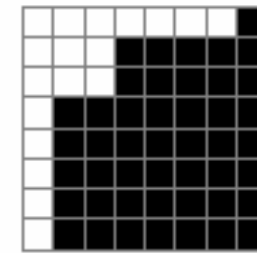
```
>> dctdemo
```

- Indicada para a codificação com perda
- Usada na norma JPEG

Original Circuit Image



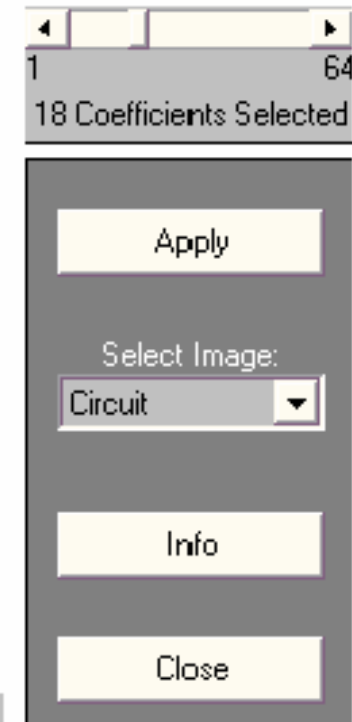
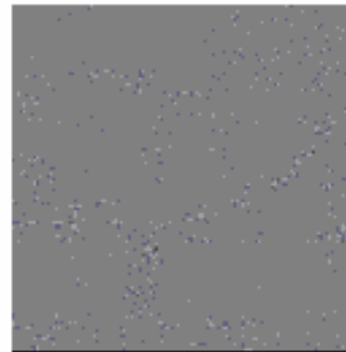
DCT coefficients



Reconstructed Image



Error Image



A screenshot of the MATLAB DCT Demo GUI. It features a slider at the top with values 1 and 64, and the text "18 Coefficients Selected". Below the slider are three buttons: "Apply", "Info", and "Close". A "Select Image:" dropdown menu is set to "Circuit".

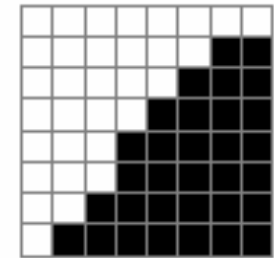
The MSE (with images normalized) is 0.000328 .

DCT2D: concentração de energia

Original Circuit Image



DCT coefficients



Reconstructed Image



Error Image



◀ | | ▶

1 | 64

32 Coefficients Selected

Apply

Select Image:

Circuit ▼

Info

Close

The MSE (with images normalized) is 0.000119.