Elaborazione di Segnali Multimediali
a.a. 2017/2018

Quantizzazione
Soluzioni

1 Quantizzazione uniforme

```
function xq = quant_unif(x,R);

N = 2^R;
xmax = max(x(:));
xmin = min(x(:));
Delta = (xmax - xmin)/N;
y = [xmin+Delta/2:Delta:xmax];    % livelli di restituzione

t  = [min(x(:))-1 (y(1:N-1)+y(2:N))/2 max(x(:))+1]; % soglie di decisione
xq = zeros(size(x));
for i = 1:N,
   xq = xq+((t(i)<=x)&(x<t(i+1)))*y(i);
end;
```

1.1 Esercizi proposti

1. *Codifica e Decodifica.*

```
function I = cod(x,R)

N = 2^R;
xmax = max(x(:));
xmin = min(x(:));
Delta = (xmax - xmin)/N;
y = [xmin+Delta/2:Delta:xmax];    % livelli di restituzione

t = [min(x(:))-1 (y(1:N-1)+y(2:N))/2 max(x(:))+1]; % soglie di decisione
I = zeros(size(x));
for i = 1:N,
    I = I+((t(i)<=x)&(x<t(i+1)))*i;
end;
```

```
function xq = decod(I,R,xmin,xmax)

N = 2^R;
Delta = (xmax - xmin)/N;
y = [xmin+Delta/2:Delta:xmax];    % livelli di restituzione
xq = y(I);
```

2. *Curve tasso-distorsione.*

```
fid = fopen('lena.y','rb');
x = fread(fid, [512 512], 'uint8');
for R=1:7,
   xq = quant_unif(x,R);
   psnr(R) = 10*log10(255.^2/mean((x(:)-xq(:)).^2));
end
plot([1:7],psnr,'-*'); xlabel('R'); ylabel('PSNR');
```

3. *Quantizzazione predittiva.* Cominciamo col modificare fase di decodifica. Supponendo le stime pari proprio al pixel precedente, risulta:

```
% Decodifica:
xr = cumsum(eq')';
```

In questo modo però l'immagine non si ricostruisce bene, a causa della propagazione (lungo le righe) dell'errore di quantizzazione. Infatti, codificatore e decodificatore non sono allineati, per cui l'errore di predizione può crescere a tal punto da far ricostruire valori molto diversi da quelli desiderati (disallineamento del decodificatore). E' allora necessario, anche in fase di codifica, produrre le stime dai valori quantizzati (figura 13) nel modo seguente:

```
% Codifica:
xq(:,1) = x(:,1);
eq(:,1) = x(:,1);
for i = 2:Nc
    xp(:,i) = xq(:,i-1);
    e(:,i)  = x(:,i)-xp(:,i);
    eq(:,i) = quant_laplace(e(:,i),R);
    xq(:,i) = xp(:,i)+eq(:,i);
end
```

Noterete come a parità di tasso di codifica (variabile da 1 a 4) la quantizzazione predittiva permette di ricostruire meglio l'immagine rispetto alla sola quantizzazione uniforme.

4. *Confronto strategie di predizione.*

```
x = double(imread('Rice.tif'));  % leggo l'immagine da un file
[M,N] = size(x);                 % determino le dimensioni della matrice x

%strategia 1
x2 = [];
x2(1,1) = x(1,1);                %inizializzo la matrice di predizione x2
x2(1,2:M) = x(1,1:M-1);
x2(2:N,1) = x(1:N-1,1);
for i = 2:M
    x2(i,2:N) = x(i,1:N-1);
end
MSE = mean2((x2-x).^2);
subplot(221),imshow(x2-x,[]),title(['Strategia 1  MSE = '  num2str(MSE)])

%strategia 2
x2 = [];
x2(1,1) = x(1,1);
x2(1,2:M) = x(1,1:M-1);
x2(2:N,1) = x(1:N-1,1);
for i = 2:N
    x2(2:M,i) = x(1:M-1,i);
end
MSE = mean2((x2-x).^2);
subplot(222),imshow(x2-x,[]),title(['Strategia 2  MSE = '  num2str(MSE)])

%strategia 3
x2 = [];
x2(1,1) = x(1,1);
x2(1,2:M) = x(1,1:M-1);
x2(2:N,1) = x(1:N-1,1);
for i = 2:M
    for j = 2:N
    x2(i,j) = (x(i-1,j) + x(i,j-1) + x(i-1,j-1))/3;
    end
end
MSE = mean2((x2-x).^2);
subplot(223),imshow(x2-x,[]),title(['Strategia 3  MSE = '  num2str(MSE)])
```

```
%strategia 4
x2 = [];
x2(1,1) = x(1,1);
x2(1,2:M) = x(1,1:M-1);
x2(2:N,1) = x(1:N-1,1);
for i = 2:M
    for j = 2:N
        dr = x(i-1, j)-x(i -1, j - 1);
        dc = x(i, j-1)-x(i -1, j - 1);
        if (dr < dc)
            x2(i,j) = x(i,j-1);
        else
            x2(i,j) = x(i-1,j);
        end
    end
end
MSE = mean2((x2-x).^2);
subplot(224),imshow(x2-x,[]),title(['Strategia 4  MSE = '  num2str(MSE)])
```