

# MATH444 HW7

Jiasen Zhang

## 1 Problem 1

The given compressed image is shown in Figure 1. Since the given data only takes about 0.15% of the full image data, the given image is almost empty only with some sparse points in the image. That means we can't recover all the information of the full image.

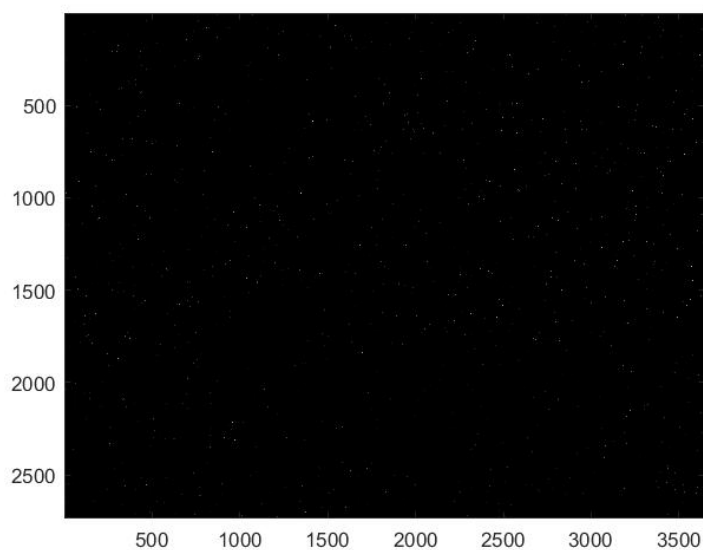


Figure 1: The given image.

Using tree regression to reconstruct the image, the best result is shown in Figure 2, in which all the leaves are pure. In Figure 2 we can see a white house and a white tower near the sea. We can recognize the blue sky with clouds, the blue sea and green trees.



Figure 2: The approximate image.

Figure 3 shows different stages of reconstruction.



Figure 3

## 2 Matlab Code

```
1 clear all;clc;
2 tic
3
4 load MysteryImage.mat;
5
6 % show the given data as an image
7 I0=zeros(m,n,3);
8 for i=1:15000
9     I0(rows(i),cols(i),1)=vals(i,1);
10    I0(rows(i),cols(i),2)=vals(i,2);
11    I0(rows(i),cols(i),3)=vals(i,3);
12 end
13 %imagesc(I0);
14
15 row0 = 1:m;
16 col0 = 1:n;
17 y = zeros(m,n,3); % the approximated image
18
19
20 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
21 % initialize the tree
22 % structure R
23 R(1).x = vals;
24 R(1).row = rows; % coordinate of sampled pixels
25 R(1).col = cols; % coordinate of sampled pixels
26 R(1).row0 = row0; % coordinate of image
27 R(1).col0 = col0; % coordinate of image
28 R(1).split_idx = [];
29 R(1).leftR = []; % left child
30 R(1).rightR = []; % right child
31 m=1;
32
33 depth=16;
34
35 figure(1);
36 for k=1:depth
37
38     for i=1:m
39         % if it's a mixed leaf
40         if isempty(R(i).leftR) && length(R(i).row)>1
41             s = find_s(R(i).row,R(i).x);
42             R(i).leftrow = R(i).row(R(i).row<=s);
43             R(i).rightrow = R(i).row(R(i).row>s);
44             R(i).leftcol = R(i).col(R(i).row<=s);
45             R(i).rightcol = R(i).col(R(i).row>s);
46             xleft=R(i).x(R(i).row<=s,:);
47             xright=R(i).x(R(i).row>s,:);
48             %
49             c1 = mean(R(i).x(R(i).row<=s,:),1);
50             c2 = mean(R(i).x(R(i).row>=s,:),1);
51             %
52             for j=1:3
53                 y(min(R(i).row0:floor(s),R(i).col0,j)=c1(j);
54                 y(ceil(s):max(R(i).row0),R(i).col0,j)=c2(j);
55             end
56
57             R(i).split_idx = s;
58
59             if k<depth
60                 R(i).leftR = m+1;
61                 R(i).rightR = m+2;
62                 R(m+1).row = R(i).leftrow;
63                 R(m+1).col = R(i).leftcol;
64                 R(m+2).row = R(i).rightrow;
```

```

65         R(m+2).col = R(i).rightcol;
66         R(m+1).x = xleft;
67         R(m+2).x = xright;
68
69         R(m+1).row0 = (min(R(i).row0):floor(s));
70         R(m+1).col0 = R(i).col0;
71         R(m+2).row0 = (ceil(s):max(R(i).row0));
72         R(m+2).col0 = R(i).col0;
73
74         m=m+2;
75     end
76 end
77 end
78 imagesc(y);drawnow;
79
80 for i=1:m
81     % if it's a mixed leaf
82     if isempty(R(i).leftR) && length(R(i).col)>1
83         s = find_s(R(i).col,R(i).x);
84         R(i).leftcol = R(i).col(R(i).col<=s);
85         R(i).rightcol = R(i).col(R(i).col>s);
86         R(i).leftrow = R(i).row(R(i).col<=s);
87         R(i).rightrow = R(i).row(R(i).col>s);
88         xleft=R(i).x(R(i).col<=s,:);
89         xright=R(i).x(R(i).col>s,:);
90         %
91         c1 = mean(R(i).x(R(i).col<=s,:),1);
92         c2 = mean(R(i).x(R(i).col>s,:),1);
93         %
94         for j=1:3
95             y(R(i).row0,min(R(i).col0):floor(s),j)=c1(j);
96             y(R(i).row0,ceil(s):max(R(i).col0),j)=c2(j);
97         end
98
99         R(i).split_idx = s;
100
101         if k<depth
102             R(i).leftR = m+1;
103             R(i).rightR = m+2;
104             R(m+1).row = R(i).leftrow;
105             R(m+1).col = R(i).leftcol;
106             R(m+2).row = R(i).rightrow;
107             R(m+2).col = R(i).rightcol;
108             R(m+1).x = xleft;
109             R(m+2).x = xright;
110
111             R(m+1).row0 = R(i).row0;
112             R(m+1).col0 = (min(R(i).col0):floor(s));
113             R(m+2).row0 = R(i).row0;
114             R(m+2).col0 = (ceil(s):max(R(i).col0));
115
116             m=m+2;
117         end
118     end
119 end
120 imagesc(y);drawnow;
121
122 fprintf('depth = %d \t m = %d\n',k,m);
123 end
124
125
126 toc
127 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
128
129
130 % function for finding s
131 function s = find_s(t,x)
132 % t: coordinate of x

```

```

133 % x: data value
134 % s: split value
135 p=length(t);
136 [t, idx] = sort(t); % sort t and x
137 x(:, :) = x(idx, :);
138 sigma=zeros(1,p-1);
139 F=zeros(1,p-1);
140 for j=1:p-1
141     sigma(j)=(t(j)+t(j+1))/2;
142     idx1=(t<=sigma(j));
143     idx2=(t>sigma(j));
144     c1 = mean(x(idx1,:),1);
145     c2 = mean(x(idx2,:),1);
146     FL = sum(sum((x(idx1,:)-c1).^2)); % sum of square of 2-norm
147     FR = sum(sum((x(idx2,:)-c2).^2)); % sum of square of 2-norm
148     F(j)=FL+FR; % MSE
149 end
150 s=sigma(F==min(F));
151 s = (s(1)+s(end))/2;
152 end

```