BRIEF ANSWERS TO ALESSANDRO'S QUESTIONS

1 (a)

15	1	2	12
4	10	9	7
8	6	5	11
3	13	14	0

4 bits/pixel

4 × 4 pixels

→ The histogram is already flat!

1 (b)

1	1	2	2
2	1	3	1
1	1	0	1
1	2	2	1

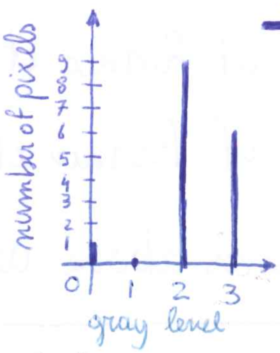
2 bits/pixel

4 × 4 pixels

gray level	probability	cumulative probability	round to nearest multiple of 1/3	output gray level
0	1/16	1/16	0/3	0
→ 1	9/16	10/16	2/3	2
2	5/16	15/16	3/3	3
3	1/16	16/16	3/3	3

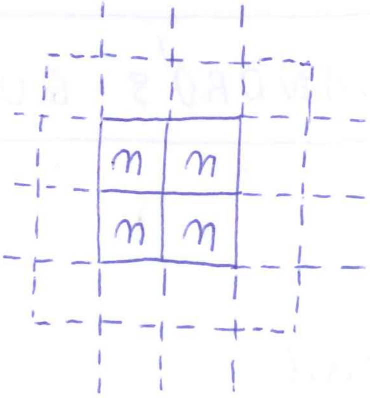
→ output:

2	2	3	3
3	2	3	2
2	2	0	2
2	3	3	2



→ The histogram is badly equalized because of discretization and the small number of bits and pixels.

1 (c)



each pixel has the same gray level n

N gray levels are possible ($2^{\#\text{bits}}$).

$0 \leqslant n \leqslant N-1$

n occurs with probability $= 1$

cumulative probability $= 1$

output gray level $= N-1$

➔ the output image is white.

1 (d)

$$\int_0^y q(y')\,dy' = \int_0^x p(x')\,dx'$$

as in histogram matching (equalization)

$$y = 1 - e^{-ax}$$

$$\Rightarrow x = -\frac{1}{a} \ln(1-y)$$

---

2 (a) See Sect. 2.1 of Romeo et al. (2004) and Fig. 1 of Romeo et al. (2003).

2 (b) See Sect. 2.2 of Romeo et al. (2004) and Fig. 2 of Romeo et al. (2003) [see also Fig. 6 of Romeo et al. (2004)].

2 (c) See Sect. 3.1 of Romeo et al. (2004).

2 (d) See Sect. 3.2 of Romeo et al. (2004).

NOTE: The references above are linked from the lecture notes.

3 (a) Single-pixel entropy $H_1 = -\sum_{i=0}^{15} p_i \log_2 p_i = 4$ bits/pixel.

Theoretical maximum compression ..... =

$$\frac{\text{\# bits/pixel in the image}}{\text{single-pixel entropy}} = 1 \rightarrow \underline{\text{no}} \text{ compression!}$$

Magic square ☺

How many pixels do we really need to transmit?

Yes	Yes	Yes	Yes
Yes	Yes	Yes	No
Yes	Yes	Yes	No
No	No	No	No

10 pixels → compression = 1.6;

or, e.g.

Y	Y	Y	Y
Y	Y	Y	N
N	Y	Y	N
N	N	N	N

9 pixels → higher compression ≃ 1.8, but more difficult to transmit.

→ compression > theoretical maximum...

because pixels are correlated and we are reducing inter-pixel redundancy.

3 (b)

gray level	probability	Huffman code	length
1	9/16	0	1
2	5/16	1 0	2
0	1/16	1 1 0	3
3	1/16	1 1 1	3

Mean Huffman-code length $N_{mean} = \sum_{i=0}^{3} p_i N_i \simeq 1.6$ bits/pixel.

→ compression $\simeq 1.3$ (the original image has 2 bits/pixel).

3 (c)

18	21	24	27
30	33	36	39
42	45	48	51
54	57	60	63

6 bits/pixel

4 × 4 pixels

→ Lossless predictive coding: $\bar{f}_n = f_{n-1}$

$e_n = f_n - \bar{f}_n = f_n - f_{n-1}$

18	3	3	3
3	3	3	3
3	3	3	3
3	3	3	3

→ Run length coding: 18 (3, 15)

6 bits + 7 bits + 4 bits = 17 bits

original # bits = 96 bits

→ compression $\simeq 5.6$.

<u>NOTE</u>: minimum number of assumptions!

3 (e) continued

Larger-size image, e.g.

18	21	24	27	30
33	36	39	42	45
48	51	54	57	60
63	66	69	72	75
78	81	84	87	90

7 bits/pixel

5 × 5 pixels

➔ same coding as before : 18 (3, 24)

7 bits + 8 bits + 5 bits = 20 bits

original # bits = 175 bits

➔ compression ≃ 8.8, higher efficiency!

NOTE: a rigorous generalization is non-trivial!

But the trend is clear!!

3 (d)

0	1	3	6
10	15	21	28
36	45	55	66
78	91	105	120

7 bits / pixel

4 × 4 pixels

→ Lossless predictive coding: $\bar{f}_n = 2 f_{n-1} - f_{n-2}$ (gradient-based)

$$e_n = f_n - \bar{f}_n = f_n - 2 f_{n-1} + f_{n-2}$$

0	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

→ Run length coding: 0 1 (1, 14)

7 bits + 7 bits + 9 bits + 4 bits = 27 bits

original # bits = 112 bits

→ compression ≃ 4.1. (same note as in 3c).

Larger-size image, e.g.

0	1	3	6	10
15	21	28	36	45
55	66	78	91	105
120	136	153	171	190
210	231	253	276	300

9 bits / pixel

5 × 5 pixels

→ same coding as before : 0 1 (1, 23)

9 bits + 9 bits + 11 bits + 5 bits = 34 bits

original # bits = 225 bits

→ compression ≃ 6.6,

higher efficiency! (same note as in 3c).