Pràctica 2: Introducció a la Teoria de l'Informació i la Compressió de Dades

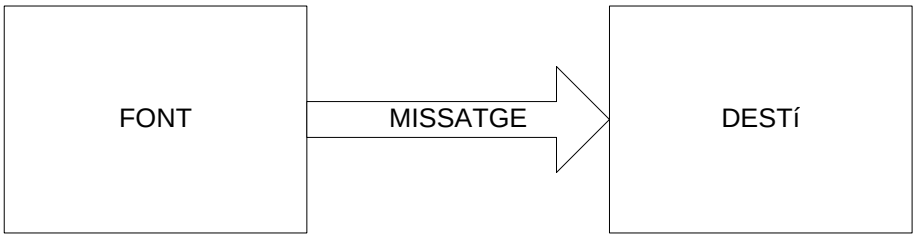
Primera part: L'Entropia

Introducció

En aquesta primera part tractarem de fer una introducció al concepte d'Entropia, concepte bàsic a la Teoria de l'Informació. Primer de tot visualitzarem el concepte de desordre o soroll inherent d'una font, i després farem servir l'eina de l'histograma sobre aquesta mateixa font. Ambdues coses ens portaran de manera intuïtiva al concepte i la definició d'Entropia, que implementarem finalment en MATLAB mitjançant una funció.

Desordre o soroll inherent d'una font

Volem estudiar el comportament d'una font generadora d'informació.



L'únic que tenim per treballar és el missatge que genera aquesta font, el mateix missatge amb el que la font transmet la informació.
La font serà discreta, es a dir, que el nombre de símbols que té el seu alfabet es finit. En aquest cas seran 256 símbols de 8 bits cadascun.

El primer que farem és carregar un fitxer de text amb el MATLAB. Aquest serà el missatge generat per la font que estudiarem. Com que treballarem amb nombres sencers de 8 bits i el MATLAB fa servir variables de tipus real, li haurem d'indicar d'alguna manera:

```
% Llegeix els primers 90.000 caràcters del Silmarillion, i guarda'ls com a sencers de 8 bits
fid = fopen('silmarillion.txt', 'r');
m1 = fread(fid, 90000,'uint8');
fclose(fid);
```

El nombre 90.000 és el nombre de mostres a llegir, i és arbitrari. Ha de ser un nombre raonable però suficient de mostres, i al hora ens ha de permetre transformar aquest vector de dades en una matriu quadrada, per poder visualitzar aquestes mostres com a una imatge.

```
% Fes que tinguin forma de matriu 300x300
M1=reshape(m1,[300 300]);

% Fes que els colors siguin una escala de grisos de 0 a 255
% Nota: això només és necessari al començament de la sessió
colormap(gray(256));

% Dibuixem les dades com a imatge
image(M1);
title('Dades com a imatge');
```

La imatge que veiem (figura 1) és una imatge grisosa, a la qual predominen els tons intermitjos amb taques ocasionals fosques o molt brillants. Podríem dir per tant que, encara que el fitxer té certa natura sorollosa, hi ha clarament patrons de similitud entre grans àrees d'aquest. Sembla lògic pensar que aprofitant d'alguna manera aquest patrons, podríem arribar a transmetre el mateix fitxer, però enviant menys bits, o el que és el mateix, enviar la mateixa informació amb un fitxer més petit.

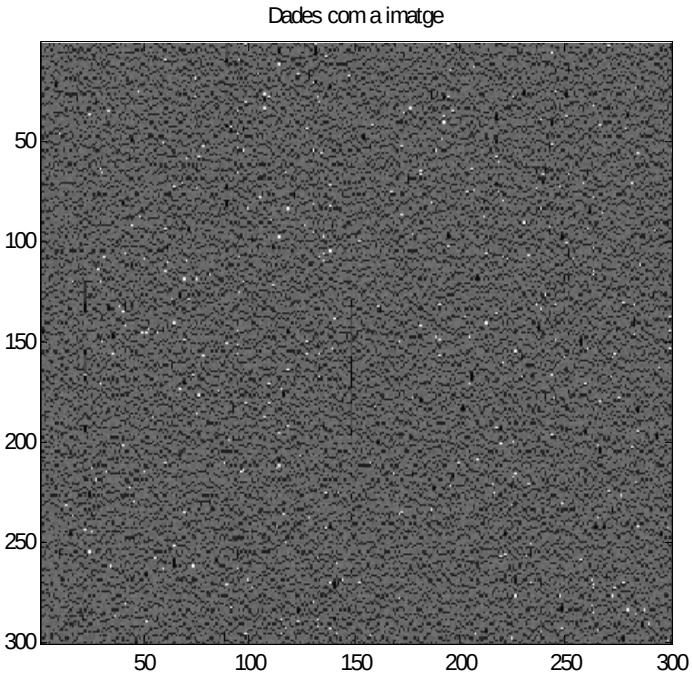


Figura 1 - Representació gràfica d'un fitxer de text

Ara anem a fer el mateix amb el mateix fitxer, però comprimit en format RAR.

```
% Llegeix els primers 90.000 bytes del Silmarillion comprimit, i guarda'ls com a sencers de 8
bits
fid = fopen('silmarillion.rar', 'r');
m2 = fread(fid, 90000,'uint8');
fclose(fid);

% Fes que tinguin forma de matriu 300x300
M2=reshape(m2,[300 300]);
```

```
% Dibuixem les dades com a imatge
image(M2);
title('Dades com a imatge');
```

El resultat (figura 2) és més proper al que entenem per soroll blanc, amb totes les tonalitats de brillo, i sense cap patró aparent. És el que veiem a un televisor quan no hi ha cap senyal sintonitzat. És un senyal més desordenat, o més sorollós, que el d'abans. Fent servir una analogia de la teoria de la termodinàmica, podríem dir que aquesta és la representació d'aquesta informació amb màxima Entropia.
Mirant aquestes imatges podríem pensar que el que fa aquest compressor de dades és, per tant, canviar la representació de dades (re-codificant les dades) per obtenir la seqüència de bits amb màxima aleatorietat = amb màxima Entropia.

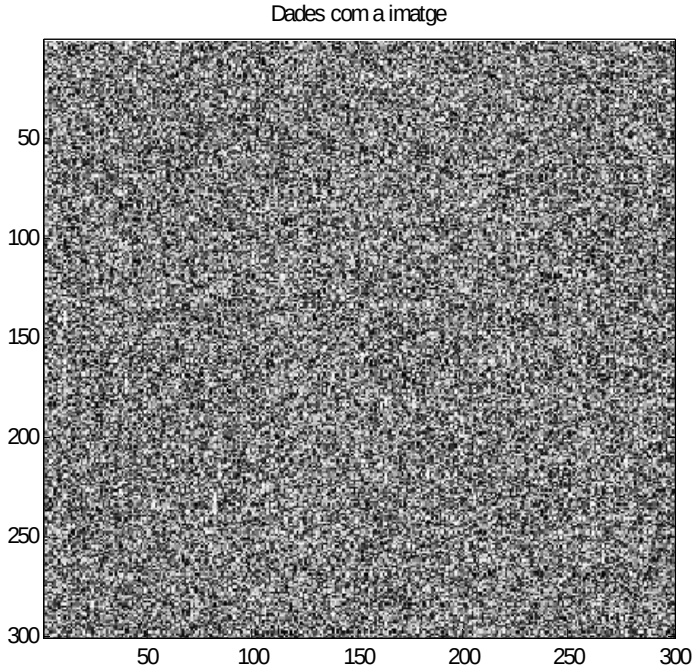


Figura 2 - Representació gràfica d'un fitxer de text comprimit

Histograma d'un missatge

Ara mirarem aquests mateixos missatges, però des d'un altre punt de vista: l'Histograma.
Així com la representació gràfica de la secció anterior ens donava, encara que fos des d'un punt de vista subjectiu, una representació temporal del missatge generat per la font que estem estudiant, l'histograma ens dóna una representació estadística d'aquest.

Tenim una funció MATLAB que ens calcula de manera directa l'histograma:

```
% Dibuixa l'histograma de les dades
bar(hist(m1,256));
title('Histograma de les dades');
```

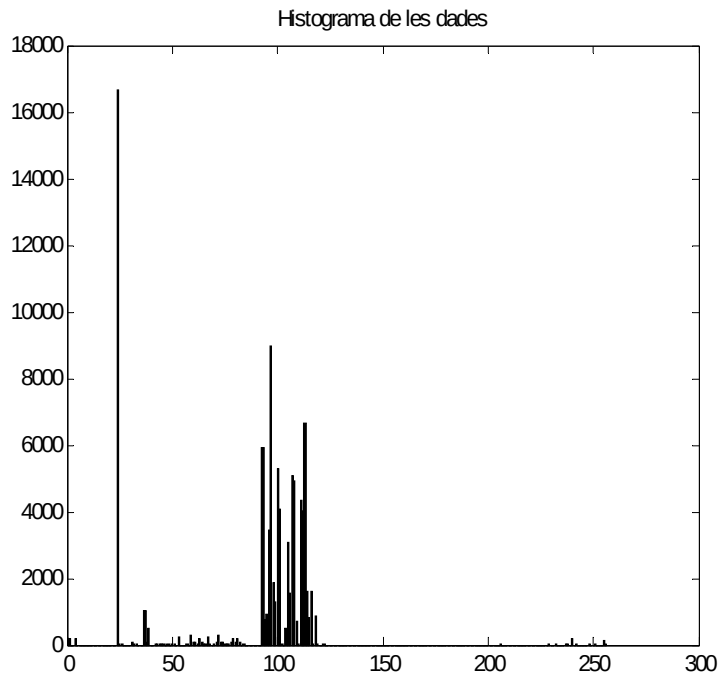


Figura 3 - Histograma d'un fitxer de text

Veiem que hi ha una gran concentració de bytes al voltant del 100, uns quants al voltant del 70, alguns a la zona del final, i finalment un pic al 32. Evidentment, això correspon a la distribució de caràcters del fitxer, codificats en codi ASCII, com podem veure a la taula de la pàgina següent. El 32 és l'espai, els caràcters al voltant del 70 són majúscules i els caràcters al voltant del 100 són minúscules. Els caràcters més grans que 127 són del codi ASCII extès, i corresponen als caràcters accentuats i amb dièresis.
Si el fitxer fos prou gros, o potser si agaféssim el llibre sencer, aquest histograma correspondria a la distribució de l'alfabet a l'idioma anglès.

Continuem amb les analogies. Si fem l'analogia de l'histograma amb la distribució freqüencial d'un senyal, ens adonem que tenim una distribució de banda estreta, o sigui, una distribució molt localitzada frequencialment, i poc aleatoritzada.

Per tant és lògic pensar que l'histograma del missatge comprimit serà un histograma quasi pla. Anem a comprobar-ho:

```
% Dibuixa l'histograma de les dades
bar(hist(m1,256));
title('Histograma de les dades');
```

ASCII	Hex	Símbolo
0	0	NUL
1	1	SOH
2	2	STX
3	3	ETX
4	4	EOT
5	5	ENQ
6	6	ACK
7	7	BEL
8	8	BS
9	9	TAB
10	A	LF
11	B	VT
12	C	FF
13	D	CR
14	E	SO
15	F	SI

ASCII	Hex	Símbolo
16	10	DLE
17	11	DC1
18	12	DC2
19	13	DC3
20	14	DC4
21	15	NAK
22	16	SYN
23	17	ETB
24	18	CAN
25	19	EM
26	1A	SUB
27	1B	ESC
28	1C	FS
29	1D	GS
30	1E	RS
31	1F	US

ASCII	Hex	Símbolo
32	20	(espacio)
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/

ASCII	Hex	Símbolo
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

ASCII	Hex	Símbolo
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O

ASCII	Hex	Símbolo
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

ASCII	Hex	Símbolo
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o

ASCII	Hex	Símbolo
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	

Tabla 1- Código ASCII

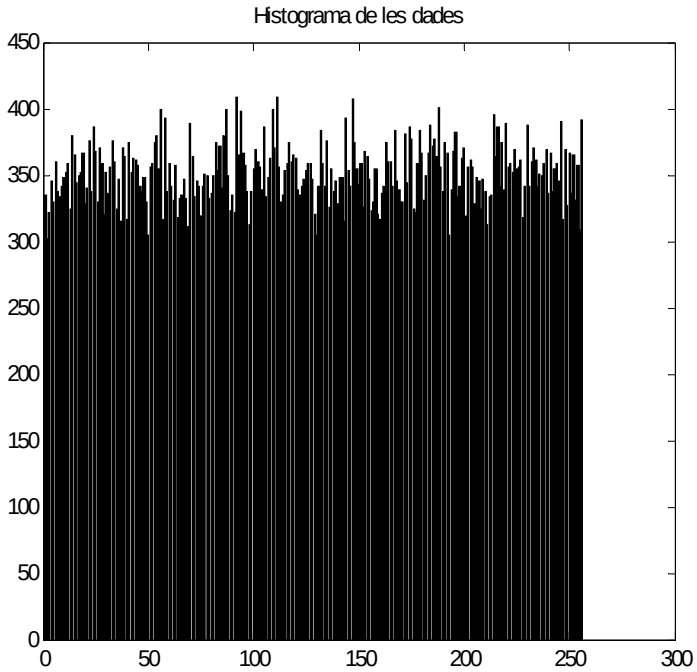


Figura 4 - Histograma d'un fitxer de text comprimit

Exercici 1.- Comprova quin tipus de soroll i quin histograma té el mateix fitxer, però amb format Word (".doc") i format Adobe (".pdf"). Explica les teves observacions, comparant els resultats amb els ja vists pel fitxer original i el fitxer comprimit (Per exemple, per què és el format Adobe el més utilitzat a Internet, en comptes del Word o el RAR?).

Exercici 2.- Fes el mateix amb un dibuix en format de mapa binari ("Dibuix.bmp") i en format JPEG ("Dibuix.JPG"). Agafa 6400 mostres al fitxer JPEG.

Exercici 3.- Fes el mateix amb un fitxer d'audio en format PCM ("Nirvana.wav") i en format MP3 ("Nirvana.mp3"). Agafa 810000 mostres al fitxer PCM.

Entropia d'una font

L'entropia és estrictament una mesura de l'informació que porta una font. Se li diu entropia perquè comparteix certes característiques amb el concepte termodinàmic d'entropia.
Si tenim el conjunt de símbols que pot generar la font com l'alfabet

$s=(s_1,s_2,...,s_N)$

i el vector de probabilitat associat

$p=(p_1,p_2,...,p_N)$

sent p_i la probabilitat de que ocurreixi el símbol *i*,
definim auto-informació o informació del símbol *i* com a

$I(p_i)=-\log_2(p_i)$

Aquest nombre es pot demostrar que representa el mínim nombre de bits amb el qual es pot codificar aquest símbol.

Llavors definirem entropia com l'esperança de l'auto-informació dels símbols:

$$H(p) = \sum_{i=0}^{N} p_i \cdot H(i) = -\sum_{i=0}^{N} p_i \cdot \log_2 p_i$$

Es mesura en bits també en bits.
Com més uniforme sigui la densitat de probabilitat, o sigui, com més iguales siguin les probabilitats dels símbols entre elles, més alta serà l'entropia, fins al límit de

$\log_2 N$

que és el nombre de bits de quantificació d'N nivells, i que correspon a un senyal tipus soroll blanc, o a un histograma pla, tal com comentàvem a les seccions anteriors.

Es pot demostrar que l'entropia és el límit de compressió per la font donada (primer teorema de Shannon): això vol dir, en altres paraules, que és el nombre de bits mínim amb el qual podríem arribar a codificar els símbols d'aquesta font, en mitja.

O sigui, en el nostre cas, com que totes les paraules de l'alfabet dels fitxers son bytes= 8 bits, si una font ens dóna H prop de 8 voldrà dir que no podrem comprimir gaire perquè ja està a prop del límit, mentre que una H baixa, de 2 per exemple, voldrà dir que podrem comprimir bastant els missatges, i enviar la mateixa informació amb molts menys bits.

Es clar, per aconseguir aquest límit de bits que representa l'entropia, el problema estarà a la definició de l'alfabet i sobretot a l'estimació del vector de probabilitat, però això ja és un altre problema...

Exercici 4.- Fes una funció MATLAB que calculi l'entropia d'una font, estimant el vector de probabilitat a partir de la freqüència d'aparició (histograma) dels símbols al missatge que passem com a paràmetre:

e=entropia(missatge)

Exercici 5.- Calcula l'entropia per cadascun dels fitxers estudiats als exercicis anteriors. Comproba si la relació entre l'entropia de diferents codificacions de la mateixa informació coincideix amb la diferència entre tamanys dels fitxers corresponents. Fes servir un nombre més gran de mostres si cal perquè l'estimació sigui millor. Als casos que les relacions no coincideixen, a què creus que és degut?