LIÈGE université
GIGA
CRC Human Imaging

Data representation & storage

Doctoral School:

The Computational Biosciences Days

Dr. Mohamed Ali Bahri, Ir Ph.D
GIGA CRC Human Imaging

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Introduction to Data Representation

Data representation is the method of organising, formatting, and storing data so that it can be processed, understood, and used effectively. In computing, this often involves converting data into a format that computers and humans can interpret, manipulate, and analyse.

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Bits & bytes

- Bit (for "binary digit") =
  - a basic unit of information used in computing and digital communications.
  - can have only one of two values → a single ***on*** and ***off*** state within an electrical circuit
  - most commonly represented as either a 1 or 0
- Byte (8 bits) =
  - a unit of digital information
  - most commonly consists of eight bits,
  - representing a binary number

Bytes

Originally,

- number of bits used to encode a single character of text in a computer
- Hardware-dependent, no definitive standard initialy fixed its size.
- convenient as power of 2 $\rightarrow$ values from 0 to 255

Now

- *de facto* standard for smallest amount of “memory unit”
- 32- or 64-bit ‘words’, built of four or eight bytes
- aka. “octet”, symbol ‘o’,

Binary vs. Decimal

Binary (Base-2) Representation:

- Computer operate using binary (base-2), meaning everything is expressed in powers of 2. In this system, each unit is a power of 2 (2^10, 2^20, etc.)
  - 1 KiB (kilobyte) = 2^10 bytes = 1,024 bytes
  - 1 MiB (mebibyte) = 2^20 bytes = 1,048,576 bytes
  - …

Decimal (Base-10) Representation:

- The decimal system is used more frequently by manufacturers of storage devices and is based on powers 10 (10^3; 10^6, etc)
  - 1 KB (kilobyte) 10^3 bytes = 1,000 bytes
  - 1 MB (megabyte) = 10^6 bytes = 1,000,000 bytes
  - …

Memory size

Expressed in binary vs. decimal base

Name	Binary (Base-2)	Decimal (Base-10)	Discrepancy
Kibibyte (KB)	2^10 = 1,024 (1 KiB)	1,000 (1 KB)	2.4%
Megabyte (MB)	2^20 = 1,048,576 (1 MiB)	1,000,000 (1 MB)	4.8%
Gigabyte (GB)	2^30 = 1,073,741,824 (1 GiB	1,000,000,000 (1 GB)	7.4%
Terabyte (TB)	2^40 = 1,099,511,627,776 (1 TiB)	1,000,000,000,000 (1 TB)	9.9%
Petabyte (PB)	2^50 = 1,125,899,906,842,674 (1 PiB)	1,000,000,000,000,000 (1 PB)	12.6%

Real-world Example of Binary vs Decimal:

Binary (Base-2) Representation:

- When you buy a 1 TB hard drive, the manufacturer uses the decimal system. However, your computer will show a capacity closer to 931 GiB instead of 1 TB.

➢ ***Marketing:*** Storage device manufactures use the decimal system because it shows a higher number, making their products seem larger.

➢ ***Operating Systems***: display memory size in binary, which can make it seem as though you have ''less'' storage than advertised.

Transfer Speed:

Refers to the rate at which data can be moved from one location to another, such as between a storage devices and memory, or over a network. It's a critical factor in computing and networking because it impacts how quickly data can be processed, transferred, or accessed.

Factors affecting Transfer Speed:

- ➢ Hardware limitations (SSD vs. HDD; or Wi-Fi vs. Ethernet)
- ➢ Bandwidth
- ➢ Latency (Delay between sending a request and receiving a response)
- ➢ Congestion and traffic (many users using the same network)
- ➢ File size and transfer efficiency

Transfer speed

Typical bandwidth

- RAM, ~10Gb per second
  $\rightarrow$ 1Gb of data in ~ 0.1 second

- Hard drive, ~0.5Gb per second
  $\rightarrow$ 10Gb of data in ~ 20 second

- Network, ~100Mbps = ~ 0.1GB per second
  $\rightarrow$ 1Tb of data in ~ 10.000 seconds = ~2.8 hours !!!

Data transfer can be a bottle neck!

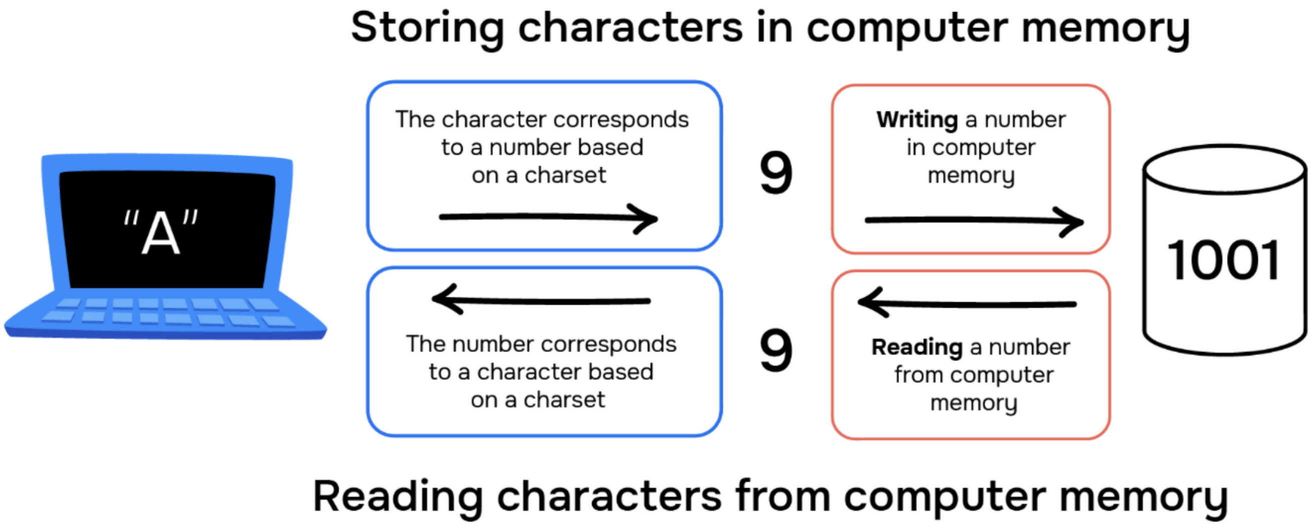
Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Character

Character encoding is the process of converting characters (letters, numbers, digit, or symbols) into a format that computers can understand, typically as binary code (0 and 1s).

With 1 byte, 1 simple character, aka. 'char'



Character

from ASCII (American Standard Code for Information Interchange)

to UTF-8 (Unicode Transformation Format – 8-bit)

Encoding	Bit size	Characters Supported	Use Case
ASCII	7 bits	128 characters	Simple text, English, control chars
Extended ASCII	8 bits	256 characters	European Languages, legacy systems
ISO-8859-1	8-bits	Western European languages	Text in Westerb European laguages
UTF-8	1-4 bytes	All Unicode characters	Web, most common today
UTF-16	2-4 bytes	All Unicode characters	Wundows, Asian scripts
UTF-32	4 bytes	All Unicode Characters	Internal processing, fixed width

b_7 b_6 b_5 →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4	b_3	b_2	b_1	Column → / Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	—	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Integer, signed or unsigned

Integers (whole numbers) are typically stored in binary form, using a fixed number of bits. The way these integers are represented determines their **range** and whether they can hold **negative values**.

Unsigned Integer: is a binary number that **only represents positive values** (including zero).

- **Number of bits:** n bits (e.g. 8-bit, 16-bit, 32-bit)
- **Range:** 0 to $2^n - 1$

Signed Integer: is a binary number that can represent both positive and negative values. In most systems, this is done using a method named two's complement.

- **Number of bits:** n-1
- **Range:** $-2^{(n-1)}$ to $2^{(n-1)}$ -1

Integer, signed or unsigned

- with **1 byte (8-bit)**,
  - ‘int8’, values ∈ [-128 127]
  - ‘unit8’, values ∈ [0 255]
- with **2 bytes (16-bit)**, or ‘**short**’,
  - ‘int16’ values ∈ [−32,768   32,767] i.e. [−(215) 215 − 1]
  - ‘uint16’, values ∈ [0 65,535] i.e. [0 216 − 1]
- with **4 bytes (32-bit)**, or ‘**long**’,
  - ‘int32’ values ∈ [−(231) 231 − 1]
  - ‘uint32’, values ∈ [0 4,294,967,295 i.e. [0 232 − 1]
- with 8 bytes,
  - …

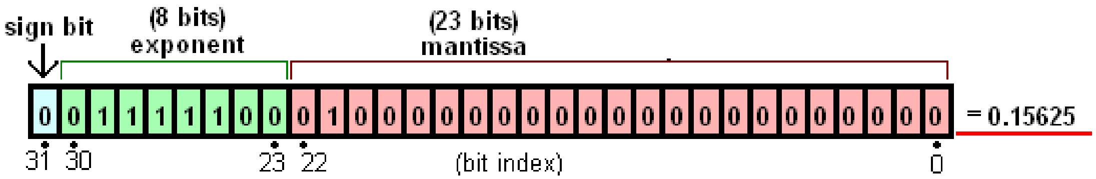
Note : *In programming,* ***short*** *and* ***long*** *are data type modifiers used to specify integer sizes, which can vary based on the system and language.*

Floating-point

Floating Point Encoding: is a method of representing real numbers in a way that computers can efficiently store and perform calculations on them.

The **IEEE 754 standard,** floating-point numbers are encoded in either 32-bit (**single precision**) or 64-bit (**double precision**) format.

$$Value = (-1)^{sign}\ x\ (1.mantissa)\ \ x\ 2^{(\text{exponent-bias})}$$



- Total bits : 32 (single-precision)
- Sign bit : 1 bit (0 ‘positive’ or 1 ‘negative’)
- Exponent width : 8 bits
- Mantissa : 24 bits (23 explicitly stored)

Floating-point

Normalization: Most floating-point numbers are stored in normalized form, meaning the mantissa is scaled to fall within a specific range.

Bias in Exponent: The exponent is stored with a bias to handle both positive and negative exponents without needing a separate sign bit.

Special values:

Zero: represented with all bits in the **exponen**t and **mantissa** set to **zero**.

Infinity: represented by setting all bits of **exponent** to **1** and all bits of the **mantissa** to **0**. the sign bit indicates +∞ or -∞.

NaN (Not a Number): Used to represent undefined or unrepresentable values like 0/0 or $\sqrt{-1}$. NaNs have an **exponent** of all **1**s and a **non-zero mantissa**

Floating-point

Example: *Encode the decimal number -5.75 in single-precision floating point:*

1. Convert to binary : -5,75 becomes -101.11_2
2. Normalize: -101.11_2 becomes -1.0111×2^2
3. Sign bit: 1 (negative)
4. Exponent: 2 + 127 (bias) = 129, which in binary is 10000001.
5. Mantissa: Drop the leading 1 and use the remaining bits 01110000000000000000000 (23 bits)

Final encoding (32 bits)

1 (sign) 10000001 (exponent) 01110000000000000000000 (mantissa)

Floating-point

Rounding Error:

Due to limited precision, floating-point numbers may introduce rounding errors, particularly when performing arithmetic operations. Certain values cannot be represented exactly in binary floating-point (e.g., 0.1), leading to small approximations.

e.g., $0.00011001100110011..._2$ infinitely repeating fraction in binary.

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Signal discretization

- Signal Discritization?
  - Process of converting a continuous signals into discrete signals for digital processing.
  - Essential for applications in digital signal processing (DSP)
- Continuous signals:
  - Can take any value within a range (e.g., Analog audio)
- Discrete signals:
  - Defined only at discrete intervals.
  - Digital audio, sampled images

Signal discretization

→ discretized signal with **finite resolution**!

Two faces of “resolution” → Different file weight!

- Sampling: Taking measurements at regular intervals
  - time/space → sampling rate
- Quantization: Mapping sampled values to discrete levels
  - amplitude → encoding precision

Sampling rate

How sparse/coarse are data sampled?

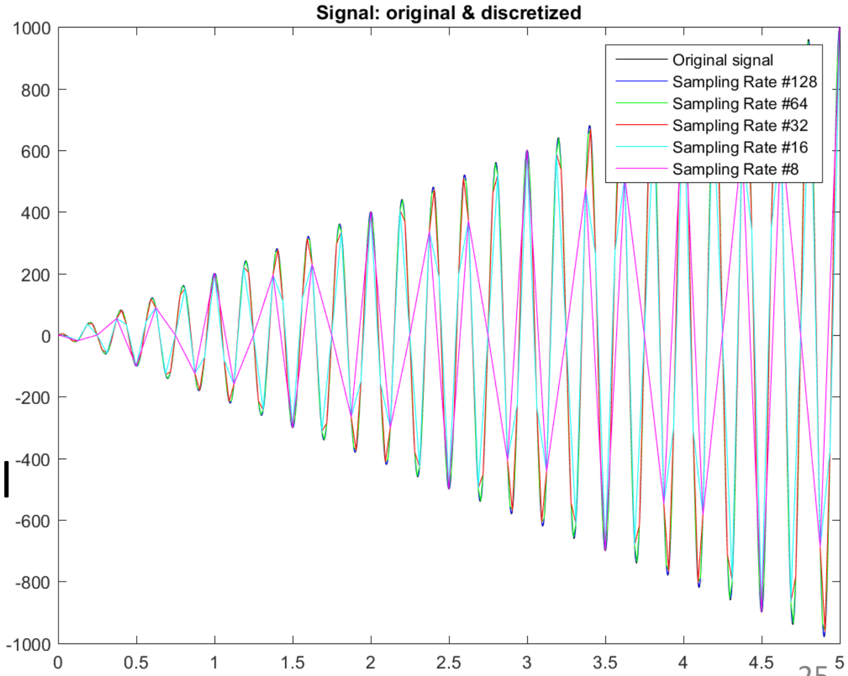
→ sampling rate

→ Nyquist theorem:

$$f_s \geq 2 * f_{max}$$

f_s : sampling rate

f_{max} : the highest frequency in the signal



Example for 3D image

Consider a 3D image with 256 x 256 x 128 = 2^{23} voxels

- 1 `int16` per voxel → 16 MB
- 1 `float32` per voxel → 32 MB

Coloured image

→ 3 RGB values par voxel, e.g. 3 `int8` per voxel → 24 MB

Resample at half the resolution, i.e. 128 x 128 x 64 voxels

→ divide sizes by 8

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

File format

- Standardised ways of encoding data for storage in files so that software and devices can easily interpret, process, and share it.
- Each file format follows specific rules for organising and encoding data, which ensures compatibility with applications that support that format.

File format

Open vs. closed file format:

- fully described vs. proprietary
- openly readable vs. requiring specific software
- community supported vs. software/company dependent

→ Stick to open format whenever possible

→ More flexibility to use with homemade software

File format

Feature	Open Format	Closed Format
Accessibility	Publicly available, no barriers	Restricted access, often requires licensing
Interoperability	High, can be used across various platforms	Low, Often limited to specific software
Transparency	Specifications are publicly documented	Specifications may be secretive or proprietary
Community involvement	Community-driven, encourages contributions	Controlled by a single organisation
Exemples	CSV, XML, JSON, ODF	.docx, .psd, .pages

The case of MS Word & Excel

Both are proprietary and cost €€€ + files are “binarized”

- Word & `.doc` files, replace by

  → ‘MarkDown’ (`.md`) files

  → open editor/reader, e.g. Typora (https://typora.io/)

- Excel & `.xls` files , replace by

  → ‘comma-separated value’ or ‘tab-separated value’ (`.csv`/`.tsv`) files

  → open editor/reader, e.g. CSVed (https://csved.sjfrancke.nl/)

Whenever possible and appropriate

DataComments.md - Typora

File Edit Paragraph Format View Themes Help

Some comments about the data.

Overall ~79Gb: (~58k files & 208 folders)

- MSHS, 37Gb, 37 subjects
- MSPA, 40Gb, 40 subjects
- MSP FLAIR/mask, 2.5Gb, 40 subjects

MSPA:

possibly to exclude.
s08825. Rather visible movement artefacts. Poor positioning in scanner -> cerebellum out of FOV?
s00349. Some movement artefact + hyper-instensities (artefact) in orbito-frontal area for MT.
s00356. hyper-instensities (artefact) in orbito-frontal area for MT + small meningiome between the frontal hemispheres.

DataComments.md - Typora

File Edit Paragraph Format View Themes Help

```
## Some comments about the data.

Overall ~79Gb: (~58k files & 208 folders)
- MSHS, 37Gb, 37 subjects
- MSPA, 40Gb, 40 subjects
- MSP FLAIR/mask, 2.5Gb, 40 subjects

#### MSPA:

possibly to exclude.
**s08825**. Rather visible movement artefacts. Poor positioning in scanner -> cerebellum out of FOV?
**s00349**. Some movement artefact + hyper-instensities (artefact) in orbito-frontal area for MT.
**s00356**. hyper-instensities (artefact) in orbito-frontal area for MT + small meningiome between the frontal hemispheres.
```

Ziemann *et al. Genome Biology* (2016) 17:177
DOI 10.1186/s13059-016-1044-7

Genome Biology

COMMENT **Open Access**

Gene name errors are widespread in the scientific literature

Mark Ziemann[1], Yotam Eren[1,2] and Assam El-Osta[1,3*]

Abstract

The spreadsheet software Microsoft Excel, when used with default settings, is known to convert gene names to dates and floating-point numbers. A programmatic scan of leading genomics journals reveals that approximately one-fifth of papers with supplementary Excel gene lists contain erroneous gene name conversions.

Keywords: Microsoft Excel, Gene symbol, Supplementary data

Abbreviations: GEO, Gene Expression Omnibus; JIF, journal impact factor

frequently reused. Our aim here is to raise awareness of the problem.

We downloaded and screened supplementary files from 18 journals published between 2005 and 2015 using a suite of shell scripts. Excel files (.xls and.xlsx suffixes) were converted to tabular separated files (tsv) with ssconvert (v1.12.9). Each sheet within the Excel file was converted to a separate tsv file. Each column of data in the tsv file was screened for the presence of gene symbols. If the first 20 rows of a column contained five or more gene symbols, then it was suspected to be a list of gene symbols, and then a regular expression (regex) search of the entire column was applied to identify gene symbol errors. Official gene symbols from Ensembl ver-

Structured data

Structured data is stored in highly organized formats, usually with a consistent schema and data types, making it easy for machines to process and analyze.

Data as

- key/value pairs
- hierarchical structure

→ use ‘JavaScript Object Notation’, i.e. `.json`, files

Example, `task-Nback_bold.json`

```
{
    "RepetitionTime": 3.0,
    "EchoTime": 0.0003,
    "FlipAngle": 78,
    "SliceTiming": [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6, 2.8],
    "MultibandAccellerationFactor": 4,
    "ParallelReductionFactorInPlane": 2
}
```

Data compression

Data compression is the process of encoding information using fewer bits than the original representation. It aims to reduce the size of data to save storage space or improve transmission speed over networks.

Two main types of data compression: **lossless** and **lossy**.

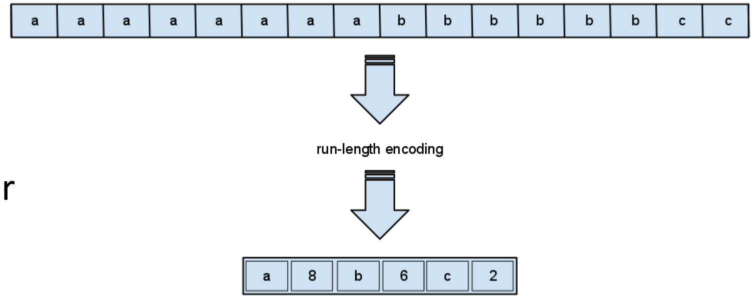
Data compression

Lossless:

- No data/signal lost
- Data reconstruction is possible

- Techniques:
  - Run-Length Encoding → replace “patterns” with fewer bytes (RLE).
  - Huffman Coding → short/long codes
  - Lempel-Ziv-Welch (LZW) → uses dictionaries

- 2-4x compression rate, depending on data
- e.g. ZIP, PNG, JPEG2000



Data compression

Lossy:

- Removes some signal → irreversible loss!
- Techniques:
  - Discrete Fourier Transform (DFT)
  - …
- Quality factor from 0 to 100 → >10x compression rate
- e.g. JPEG, MPEG, MP3

Data compression

When to Use Lossless vs. Lossy Compression

- Use **Lossless** for applications where data integrity is essential (e.g., text files, source code, financial data, medical imaging).
- Use **Lossy** for media files where some data loss is acceptable in exchange for significantly reduced file sizes (e.g., streaming music, video, web images).

Program

- Introduction
- Basic units of Data
- Data format
- Signal discretization
- File format & compression
- Storage & Safety

Storage


HDD
Actuator Arm
Read/Write Hea
Memory Chip
HARD DISK DRIVE
VS
SOLID SBATE DRIVE
Sorate Technology
SSD

Hard-disk drive

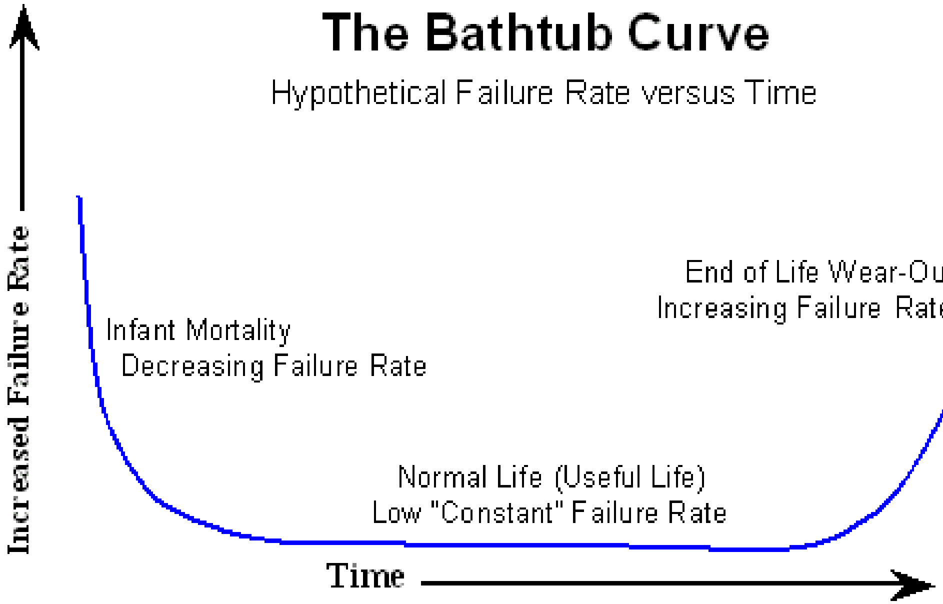
HDD = electromechanical data storag
device:

- magnetic storage to read/write data
- on one (or more rigid) rapidly rotating disks
- cheap and storage density increases (Moore's law)
- latency = ~a few ms,
- transfer rate up to ~1 Gb/s

Hard-disk drive

HDD = electromechanical data storage device:

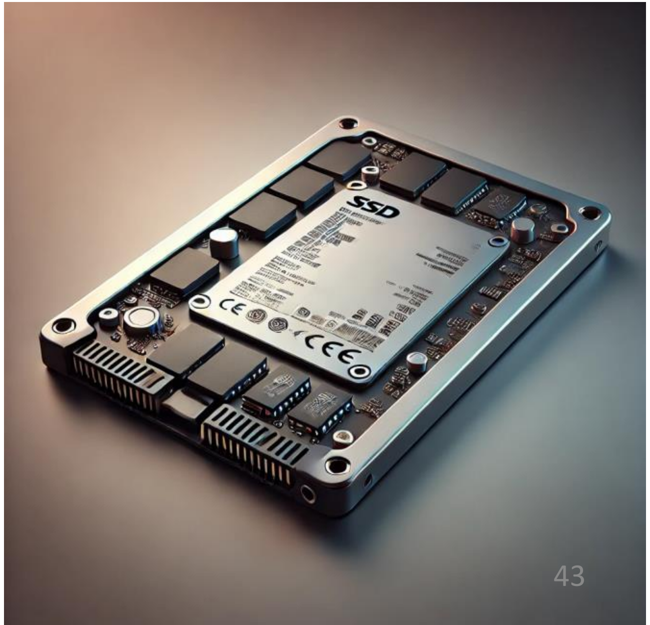
- risk of failure increases with time but…



Solid-state drive

SSD = integrated circuit data storage device:

- non-volatile NAND flash memory to read/write data
- no mechanical or moving part
- latency < ms,
- transfer rate up to a few Gb/s
- compared to HDD
  - more expensive and more reliable
  - less power consumption



ULiège mass-storage

- Personal space → your own stuff
- Platform space → raw data access
- Team space → shared data & results

Keep in mind access time

→ no direct processing of data!

Backup vs. Archive

Backup

- copy of **current** data/system
- includes files which are currently being accessed/changed

→ Restoring data/system to a previous point in time, if they are lost or become corrupted

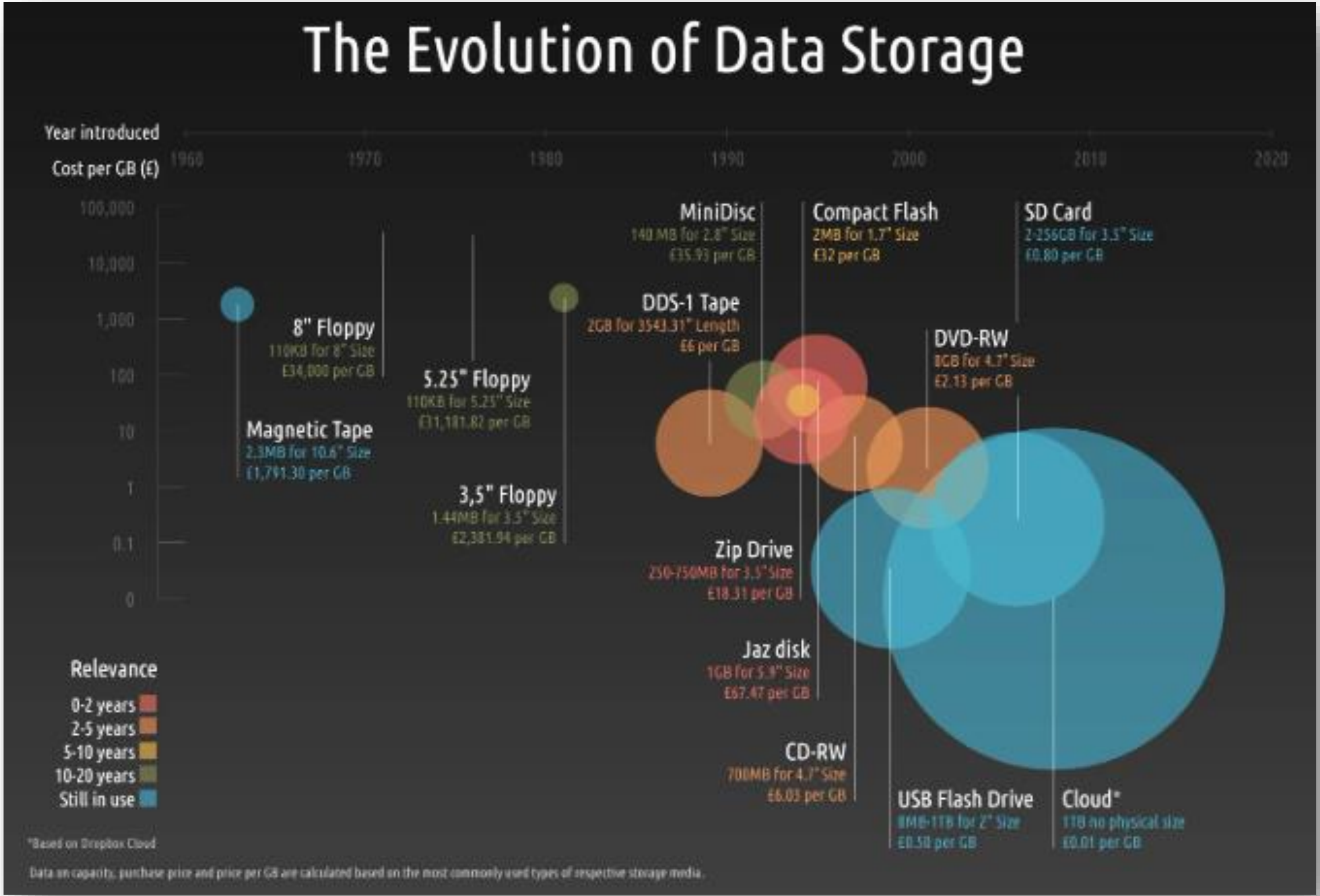
Archive

- store data/information to be kept for a long period of time
- includes files which should not be modified, accidentaly or purposely

→ Restoring the ‘original’ data/information, e.g. to re-analyse them

References

- https://en.wikipedia.org/wiki/Markdown
- https://typora.io/
- https://en.wikipedia.org/wiki/Comma-separated_values
- https://en.wikipedia.org/wiki/Tab-separated_values
- https://csved.sjfrancke.nl/
- https://en.wikipedia.org/wiki/JSON
- https://doi.org/10.1186/s13059-016-1044-7
- https://en.wikipedia.org/wiki/Run-length_encoding
- https://en.wikipedia.org/wiki/JPEG
- https://en.wikipedia.org/wiki/Hard_disk_drive
- https://en.wikipedia.org/wiki/Solid-state_drive


The Evolution of Data Storage
Year introduced
Cost per GB (£)
1960
1970
1980
1990
2000
2010
2020
100,000
10,000
1,000
100
10
1
0.1
0
8" Floppy
110KB for 8" Size
£34,000 per GB
Magnetic Tape
2.3MB for 10.6" Size
£1,791.30 per GB
5.25" Floppy
110KB for 5.25" Size
£31,181.82 per GB
3,5" Floppy
1.44MB for 3.5" Size
£2,381.94 per GB
MiniDisc
140 MB for 2.8" Size
£35.93 per GB
DDS-1 Tape
2GB for 3543.31" Length
£6 per GB
Compact Flash
2MB for 1.7" Size
£32 per GB
SD Card
2-256GB for 3.3" Size
£0.80 per GB
DVD-RW
8GB for 4.7" Size
£2.13 per GB
Zip Drive
250-750MB for 3.5" Size
£18.31 per GB
Jaz disk
1GB for 5.9" Size
£67.47 per GB
CD-RW
700MB for 4.7" Size
£6.03 per GB
USB Flash Drive
8MB-1TB for 2" Size
£0.50 per GB
Cloud*
1TB no physical size
£0.01 per GB
Relevance
0-2 years
2-5 years
5-10 years
10-20 years
Still in use
*Based on Dropbox Cloud
Data on capacity, purchase price and price per GB are calculated based on the most commonly used types of respective storage media.

Thank you for your attention!

LIÈGE université
GIGA
CRC Human Imaging