BASIRA

CANCER METABOLISM
RESEARCH & TRAINING

بصيرة

TRAINING CURRICULUM

CHAPTER FOUR

Data & Tools

What exists, where it comes from, and how to handle it without breaking it

The least glamorous chapter and one of the most consequential. A great deal of what separates reliable computational biology from unreliable happens before any statistics are run — at the moment data enters the analysis.

Founder & Programme Lead	Aroob Gomosani
Prepared & delivered by	Haneen Marghalani
Version	2.0 · second cohort
Date	2026

Gomosani, A., & Marghalani, H. (2026). Basira Curriculum. Zenodo.
https://doi.org/10.5281/zenodo.21791751

4·0

Before you begin

Nobody becomes a scientist because they were excited about file formats. But the errors this chapter prevents are the worst kind: they do not crash, they do not warn, and they produce numbers that look entirely plausible and are entirely wrong. A misread unit or a matrix turned the wrong way round will sail through the most sophisticated statistics you can throw at it.

So read this one as a chapter about not being fooled — the same subject as Chapter 1, moved down to the level of the file on disk.

WHERE THIS SITS

	Chapter	What it answers
	1 · Thinking	How does science reason, and how do I avoid fooling myself?
	2 · Practice	How do we behave — integrity, collaboration, communication.
	3 · The Science	What is cancer, and how does a cell power itself?
▶️	4 · Data & Tools	What data exists, and how do we handle it without breaking it?
	5 · The Toolkit	The statistics and code, as a reference you return to.
	6 · Build Your Own	Now go find a question of your own and run it.

WHAT YOU SHOULD BE ABLE TO DO BY THE END

- Name the four main data types and say precisely which question each one can and cannot answer.
- Recognise the major public resources and know what each holds.
- Explain why a gene can silently vanish from an analysis, and what prevents it.
- Read a TCGA barcode and identify the field that separates tumour from normal.
- Detect what unit an expression matrix is on, and say why guessing is dangerous.
- Explain the orientation trap and how a loader should defend against it.
- List the four habits that make an analysis rerunnable months later.
- Describe what a defensive loader does, in four verbs.

4.1
Four kinds of data, four questions

The first thing to fix in your mind is that each data type answers a different question. Confusing what a dataset can and cannot tell you is a more common error than any coding mistake, and it starts here rather than in the analysis.

DATA TYPE	WHAT IT MEASURES	THE QUESTION IT ANSWERS
Bulk RNA-seq	the summed mRNA of every cell in a tissue sample	what is this tissue doing, on average?
Single-cell RNA-seq	each individual cell's mRNA, kept separate	what is this cell type doing?
Somatic mutations	which genes are altered in a tumour, and how	what is broken here?
CRISPR screens	the fitness cost of disabling each gene, one at a time	what does this cell actually need?

No single type is sufficient. The design of any good study is the braiding of several.

Figure 4.1 Four measurements, four questions. Notice that only one of them measures necessity.

Look again at the last row. Expression tells you what a tissue is doing; a CRISPR screen tells you what a cell cannot survive without. These are different claims about the same gene, and Chapter 3 has already given you the biology behind why they can diverge.

4.2
Where the data comes from

Each data type comes from a major public resource. Knowing them by name and character is part of literacy in this field, and you will meet all of them again.

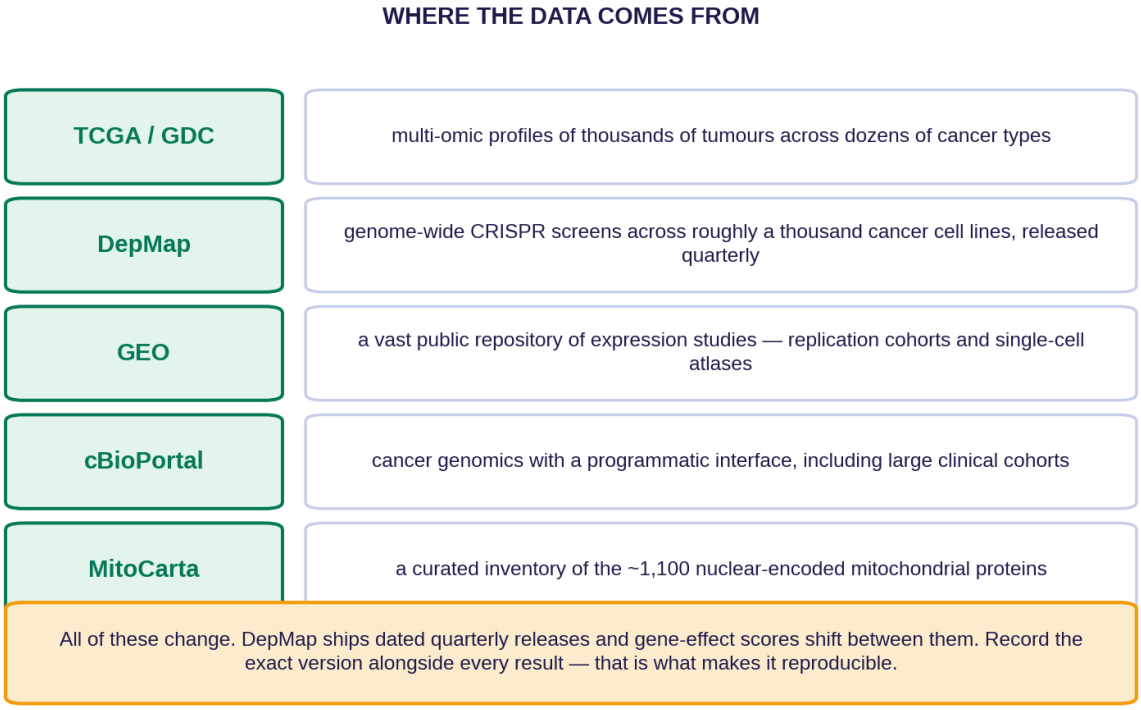


Figure 4.2 The five you will actually use, and the one rule that applies to all of them.

4.2.1 Public data and responsibility

Much of this data originates from patients. The tumour and clinical records in these resources come from real people who consented to specific research uses under specific protections, and Chapter 2's principles apply here directly and without exception. We use only appropriately consented, de-identified, publicly released data, strictly within the terms of each resource's data-use agreement, and we never attempt to re-identify anyone or combine datasets in ways those terms forbid.

Cell-line data carries fewer restrictions but still has licence terms worth reading. Treating these obligations as part of the science, rather than as an obstacle to it, is what keeps the door open for patients to keep helping.

4.3 One gene, three names

Data does not arrive analysis-ready. It arrives in particular formats, labelled with particular identifier systems, on particular scales — and reconciling all of that correctly is where careful work is won or lost. The recurring theme of this chapter is that the most dangerous errors here are silent: they produce wrong numbers rather than crashes.

Start with the genes themselves. A single gene can be named in several incompatible systems, and different datasets choose differently.

ONE GENE, THREE NAMES

the same gene, in the three systems you will actually meet

Ensembl	ENSG00000141510	stable and unambiguous, but versioned — the trailing .N usually has to be stripped
HUGO symbol	TP53	human-readable, but changes over time and carries aliases, so mapping is imperfect
Entrez	7157	a numeric NCBI identifier; often embedded inside a column header

A gene that fails to map does not raise an error. It simply vanishes from the analysis.

Figure 4.3 The same gene in three systems. Any join between two datasets has to bridge them.

▲ CAUTION — **Why a failed mapping is worse than a crash**

If your analysis draws on several resources, they will not agree on identifiers. A gene that fails to map does not raise an error — it simply drops out, and your result is computed over a slightly smaller gene set than you believe.

So map deliberately with a dedicated tool, and then check the count of genes that mapped rather than assuming it. When a gene you expected is missing from a result, a mapping failure is the first thing to suspect — before biology.

4.4 Reading a sample barcode

Samples are named by convention too, and the TCGA barcode is worth being able to read because its structure carries meaning — including the one field your entire comparison may rest on.

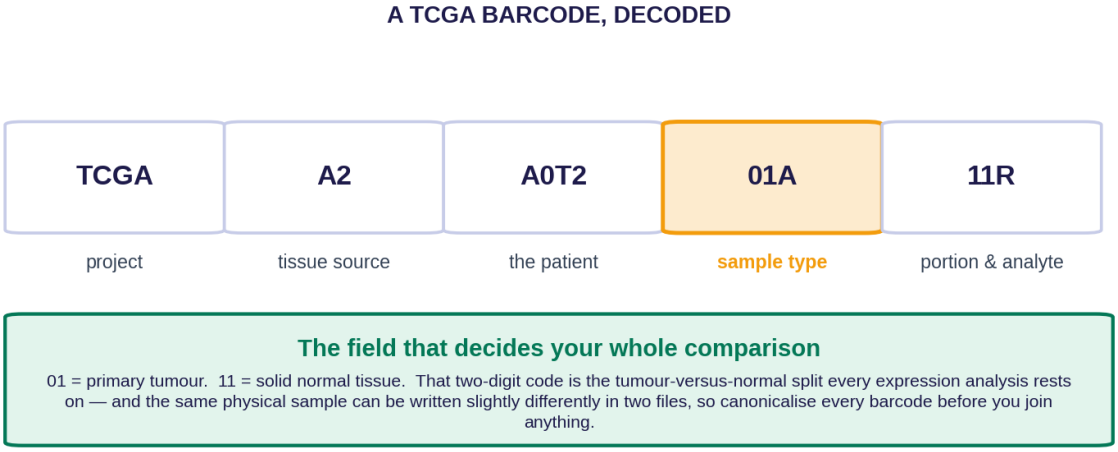


Figure 4.4 The dash-separated fields, and the two digits that decide what you are comparing.

▲ CAUTION — **The barcode-namespace trap**

The same physical sample can be written with slightly different barcode strings in two different files — differing in punctuation, or in trailing fields you did not think mattered.

Joined naively, the two files appear to share no samples at all, and your data silently disappears. The fix is to canonicalise every barcode to one standard form before any join. It is the difference between analysing your whole cohort and unknowingly analysing an empty intersection of it.

4.5
The two silent killers

If you take one practical lesson from this chapter, take this one. Two properties of a data matrix — what its numbers mean, and which way it is turned — can be wrong in ways that never raise an error and always corrupt the result.

4.5.1 Units

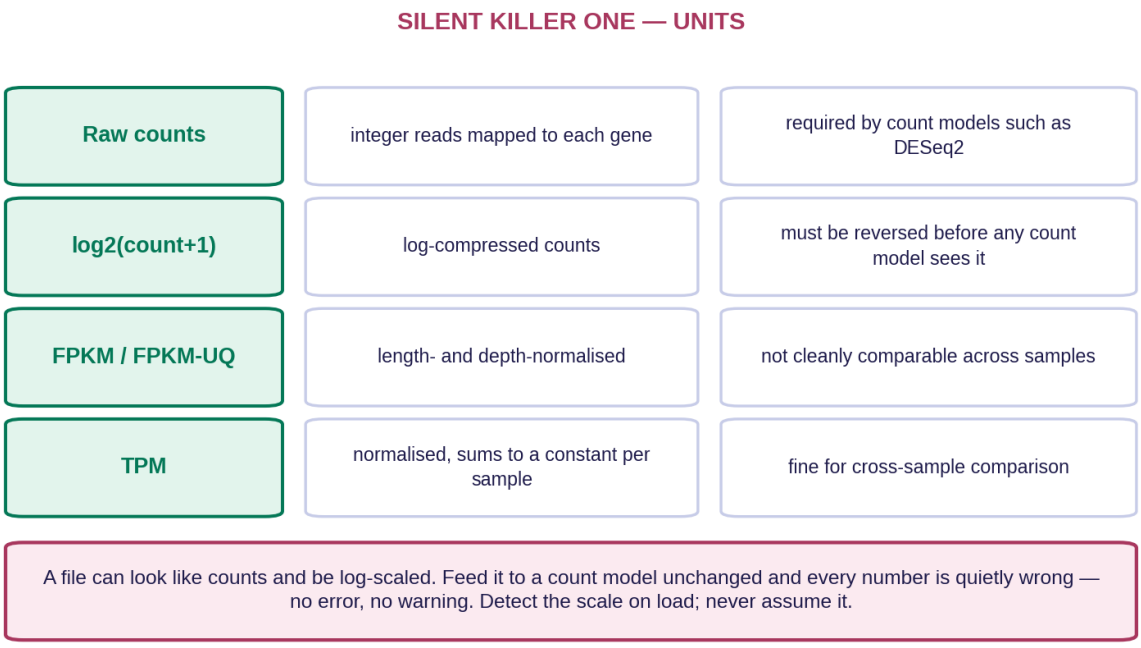


Figure 4.5 Four scales a matrix of expression values might be on. A method built for one produces nonsense on another, silently.

The practical reflex to build: on loading any expression matrix, look at its numbers before you use them. Are they integers? What is the maximum value? A matrix whose entries are non-integer and whose maximum is small has almost certainly been log-transformed already, whatever its filename claims.

4.5.2 Orientation

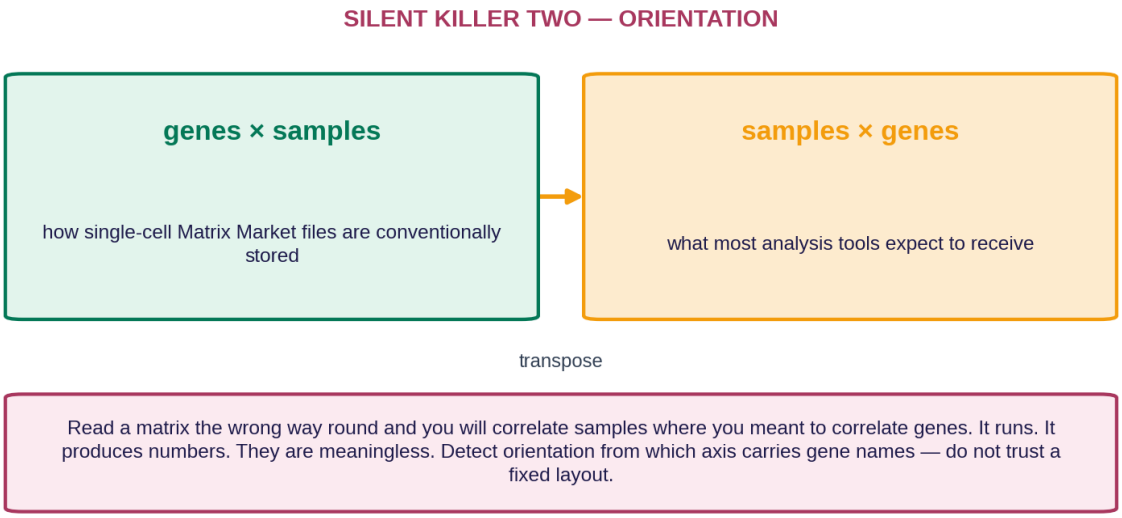


Figure 4.6 The same data, two layouts. Only one of them is what your tool expects.

? CHECKPOINT — Checkpoint 1

- For each of the four data types, write one sentence naming a question it can answer and one it cannot. (§4.1)
- Why is recording the exact version of a public dataset part of the science rather than pedantry? (§4.2)
- You join two datasets on gene name and get far fewer rows than expected. Name three things you would check. (§4.3)
- In the barcode TCGA-A2-A0T2-01A-11R, which field tells you this is a tumour and not normal tissue? (§4.4)
- You open an expression matrix and the largest value is 14.2, with decimals throughout. What scale is it probably on, and what must you do before running a count model? (§4.5.1)

4.6 The toolkit

Our analyses are written in Python, using a scientific computing stack that has become standard in the field, plus domain-specific packages for genomics. You do not need mastery of all of them on day one, but you should know what each is for so that unfamiliar code stops looking like noise.

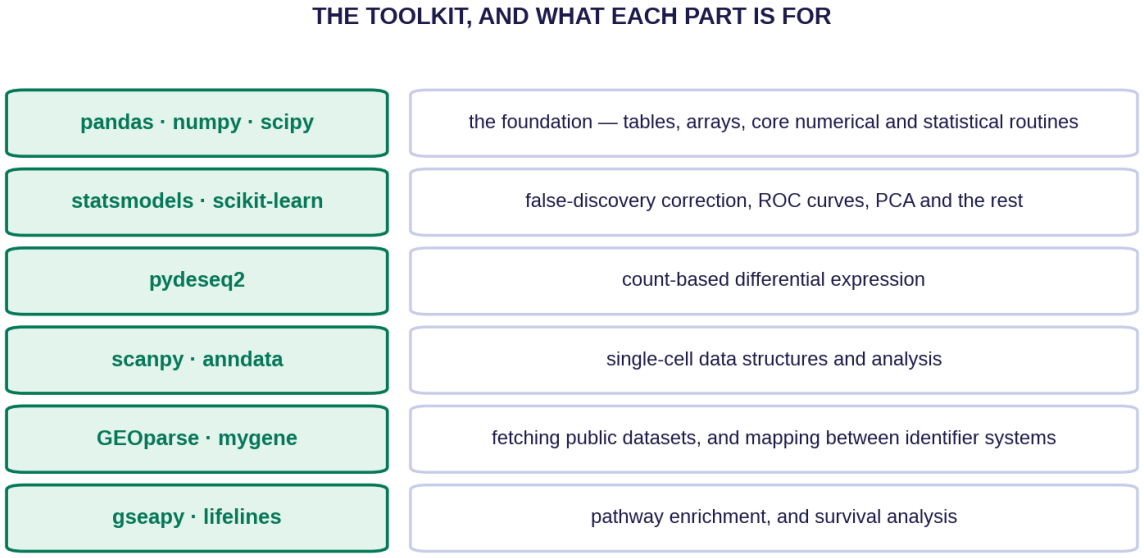


Figure 4.7 The stack. Chapter 5 teaches the statistics these implement.

The work runs in a notebook environment of the Jupyter or Colab family, which interleaves code, output and explanation. That format suits exploratory analysis, and it suits the literate, auditable style Chapter 2 asked for — a notebook is a lab notebook that happens to execute.

4.7 Making it rerunnable

Chapter 2 argued that reproducibility is a scientific obligation. This is where it becomes concrete practice. An analysis is reproducible when someone else — or you, six months later — can run it and obtain the same numbers. That takes deliberate habits, not luck.

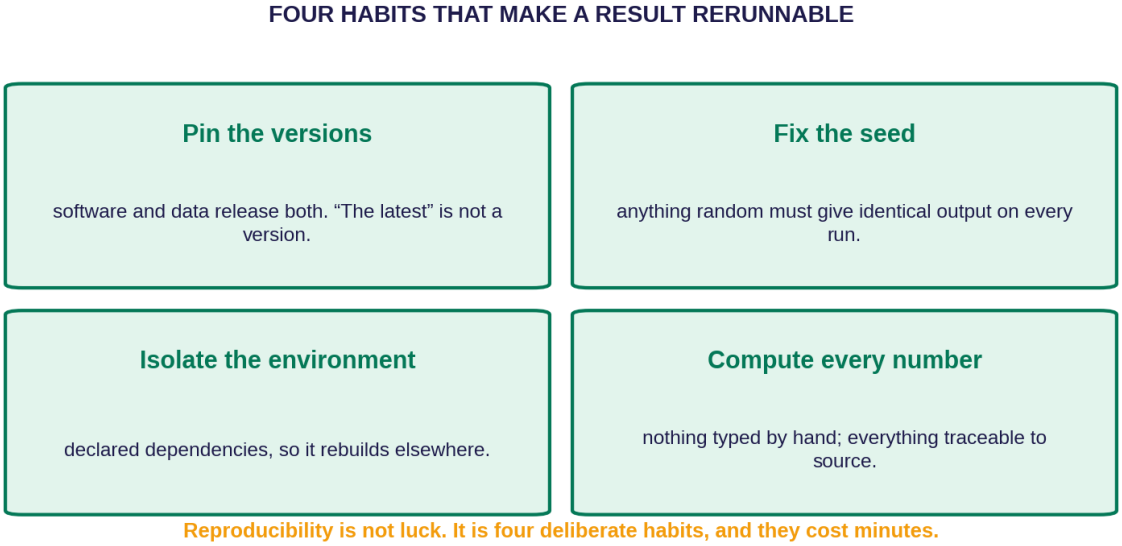


Figure 4.8 Four habits. None of them is difficult; all of them are easy to skip.

Version control closes the loop. Code evolves, and without a record of how, a result becomes impossible to reconstruct. Keeping the history of every change means any figure can be traced to the exact code that produced it — and combined with recorded data versions and pinned software, that completes the provenance chain: from a claim, back through the figure, the code and the data, to something anyone can re-run.

4.7.1 The realities of compute

Genomic data can be large, and a few practical realities shape how the work is organised. Single-cell datasets contain enormous numbers of cells and are stored sparsely — recording only the non-zero entries — because a dense representation simply would not fit in memory. Some steps are heavy enough that they are run once and their outputs cached, so later analyses read a small saved result rather than repeating an expensive computation. Being mindful of memory, runtime and what can be cached is part of doing this work at any scale.

4.8 Loading defensively

Everything in this chapter converges on a single practice: the moment data enters an analysis is the moment to check it, loudly. A loader that silently accepts a malformed file, a wrong unit or an empty join hands a corrupt input to everything downstream, where the damage is far harder to trace.

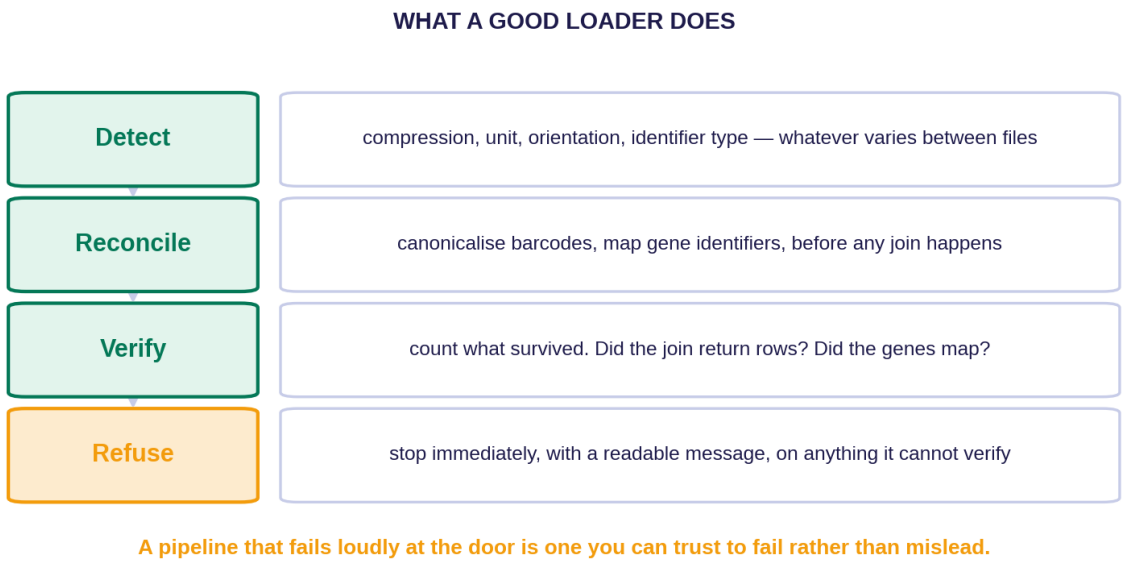


Figure 4.9 Four verbs. The fourth is the one people leave out, and it is the one that saves you.

◆ KEY IDEA — **The loader's job, in one sentence**

A good loader turns an unknown file into a known, validated object — detecting what varies, reconciling what must match, and refusing to proceed on anything it cannot verify.

Every hour spent making a loader strict is repaid many times over in errors that never happen downstream.

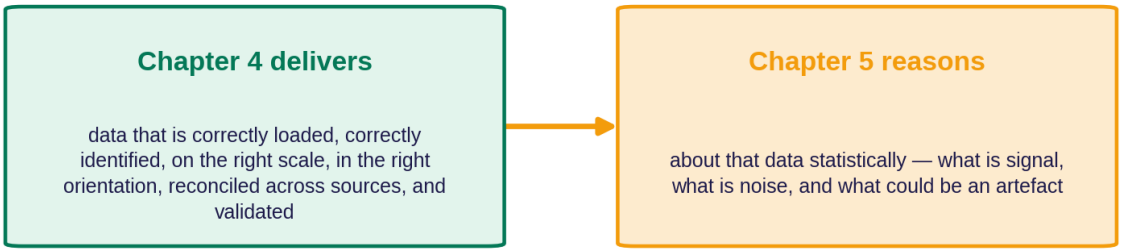
4.9
The preprocessing you should understand

Between a validated raw matrix and a statistical analysis sit a few ideas you should understand conceptually, even though Chapter 5 handles their mathematics.

- **Normalisation.** Raw measurements are not directly comparable — samples are sequenced to different depths, genes have different lengths. Different normalisations correct different distortions, and using the wrong one, or normalising twice, corrupts the comparison rather than fixing it.
- **Batch effects.** Systematic technical differences between groups of samples processed separately. These can masquerade as biology, and they are most dangerous when a batch happens to align with the comparison you care about — at which point the technical and the biological become impossible to separate after the fact.

You do not need to implement these here. You need to know they exist, and why they matter, so that Chapter 5's treatment of them makes sense when you meet it.

4.10
The handoff



The boundary matters because the most sophisticated statistics in the world cannot rescue a corrupted input.

Garbage in, garbage out is not a cliché here. It is a daily risk.

Figure 4.10 A clean boundary between two chapters, and the reason it matters.

? CHECKPOINT — Checkpoint 2

- Name three tools in the stack and say what each is for. (§4.6)
- Why must a random seed be fixed, and what goes wrong if it is not? (§4.7)
- Give the four verbs that describe a defensive loader, and say what each prevents. (§4.8)
- What is a batch effect, and when is it most dangerous? (§4.9)

- “Garbage in, garbage out.” Explain why no amount of statistical sophistication rescues a bad input. (§4.10)

4.11 End-of-chapter questions

Write a paragraph on each. Bring them to the session.

- You are handed an expression matrix with no documentation. Write the checklist you would run before trusting a single number in it.
- A collaborator sends you results but not the data version or the package versions. What can and cannot be concluded from their work?
- Describe a scenario where a batch effect would be impossible to separate from the biological signal, and say what should have been done differently at the design stage.
- Why might a defensive loader be described as an ethical practice rather than merely a technical one? Connect it to Chapter 2.
- Pick one public resource from Figure 4.2, visit it, and write a short description of what it holds and what its data-use terms require of you.
- Someone argues that checking units is a waste of time because “the filename says counts.” Write your reply.

4.12 Go deeper

Every link is live. [Read] is written documentation, [Source] is the paper behind a resource.

THE RESOURCES THEMSELVES

[Read] **NCI Genomic Data Commons** — Where TCGA lives. Browse a project and look at what a single case actually contains.

[Read] **DepMap portal** — Search any gene and see which cell lines depend on it. Worth an hour of clicking.

[Read] **Gene Expression Omnibus (GEO)** — The public repository of expression studies. Find one in your area of interest and read its metadata.

[Read] **cBioPortal** — Cancer genomics with a friendly interface and a programmatic API behind it.

[Read] **MitoCarta** — The mitochondrial protein inventory referenced in Chapter 3.

LEARNING THE TOOLS

[Read] **pandas — 10 minutes to pandas** — The fastest useful introduction to the library you will use most.

[Read] **scanpy tutorials** — Single-cell analysis, worked end to end.

[Read] **The Turing Way — guide to reproducible research** — A free, well-written handbook on exactly the habits in §4.7.

THE ORIGINALS

[Source] **Tsherniak et al. (2017) — Defining a Cancer Dependency Map** — How genome-wide dependency screening works, and why it is powerful.

[Source] **Love, Huber & Anders (2014) — DESeq2** — The count model referenced throughout. Read it once you have met Chapter 5.

4.13 Glossary

Batch effect A systematic technical difference between groups of samples processed separately, which can imitate biology.

Bulk RNA-seq Sequencing that measures the summed RNA of every cell in a sample, without separating cell types.

Canonicalisation Rewriting identifiers to one standard form before joining datasets, so that matching records actually match.

cBioPortal A portal for cancer genomics data with a public programmatic interface.

CRISPR dependency screen An experiment disabling each gene in turn to measure the fitness cost to the cell.

DepMap The Cancer Dependency Map: genome-wide CRISPR screens across roughly a thousand cancer cell lines.

Ensembl ID A stable, versioned gene identifier of the form ENSG00000141510.

Entrez ID A numeric gene identifier maintained by NCBI.

FPKM / TPM Length- and depth-normalised expression units. TPM sums to a constant per sample; FPKM does not.

GEO Gene Expression Omnibus, NCBI's public repository of expression studies.

HUGO symbol A human-readable gene symbol such as TP53. Readable, but changes over time and carries aliases.

Normalisation Adjusting raw measurements so they can be compared across samples or genes.

Orientation Whether a matrix is stored genes × samples or samples × genes. Getting it wrong fails silently.

Provenance The traceable chain from a reported claim back through figure, code and data to something re-runnable.

Raw counts Integer numbers of sequencing reads mapped to each gene. Required by count-based models.

Sparse matrix A storage format recording only non-zero entries, necessary for single-cell data to fit in memory.

TCGA The Cancer Genome Atlas: multi-omic profiles of thousands of tumours across many cancer types.

TCGA barcode A structured sample identifier encoding project, tissue source, patient and sample type.