POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Introduction to Python and its applications in bioinformatics Part two

Basics of Bioinformatics

Lecture 4

Adam Kuzdraliński

2026/2027

Sources: Biopython: tutorial

Questions we will answer

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

How can we compare the properties of selected proteins?

How can we compare DNA regions and transcript versions?

What can we calculate from a protein structure model?

How can we show differences in RNA measurements between samples?

Sources: Biopython: tutorial

Working with the examples

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each section has its own script, data folder, and results folder.

Code on later slides uses variables created earlier.

Run the complete script from the beginning in a prepared Python environment.

The included data lets you reproduce the results without downloading it again.

Sources: Biopython: tutorial

Example 1: human and mouse globins

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We will compare three selected proteins from humans and three from mice.

HBA and HBB are haemoglobin subunits, and MB is myoglobin.

We will calculate length, histidine percentage, and mass from each sequence.

Sources: UniProt: human HBA | UniProt: mouse HBA

Downloading two sets from UniProt

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Three identifiers refer to humans and three to mice.

The OR query retrieves all the selected records.

We will use the saved FASTA file for several analyses.

```
ids = ['P69905', 'P68871', 'P02144',
       'P01942', 'P02088', 'P04247']
query = ' OR '.join(f'accession:{x}' for x in ids)
r = requests.get(
    'https://rest.uniprot.org/uniprotkb/stream',
    params={'query': query, 'format': 'fasta'}, timeout=60)
r.raise_for_status()
(DATA / 'globins.fasta').write_text(r.text)
```

Sources: UniProt: API

Sequence properties

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Length is the number of amino acids in the stored sequence.

The histidine fraction is the number of H letters divided by the length.

ProteinAnalysis calculates mass from the amino acid composition of the sequence.

```
records = list(SeqIO.parse(DATA / 'globins.fasta', 'fasta'))
rec = next(r for r in records if 'P02144' in r.id)
s = str(rec.seq)
print(len(s))
print(round(100 * s.count('H') / len(s), 2))
print(round(ProteinAnalysis(s).molecular_weight(), 1))
```

Sources: UniProt: API | Biopython: ProtParam

A table of six proteins

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each row describes one selected UniProt record.

The columns contain the identifier, species, and protein name.

Further columns contain length, histidine percentage, and mass.

```
print(proteins[['id', 'species', 'protein',
                'length', 'histidine_pct']].round(2))
proteins.to_csv(OUT / 'globins.tsv', sep='\t', index=False)
```

Sources: UniProt: API

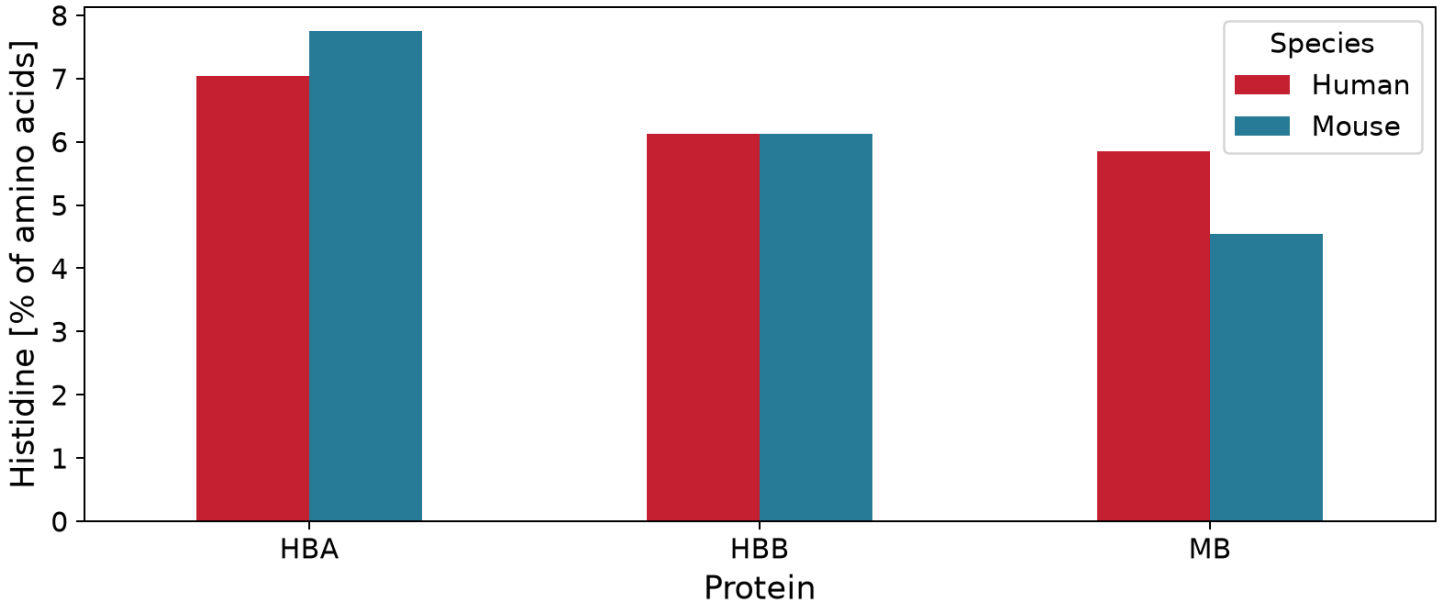
Histidine percentage in selected globins

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We compare the same type of protein between species.

Bar height shows the percentage of positions occupied by histidine.

The chart describes six selected records, rather than whole proteomes.



Sources: UniProt: API | Biopython: ProtParam

Extending a project to amino acid composition

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Instead of counting one amino acid, we can count all twenty.

This table can help us compare the composition of different protein families.

```
alphabet = 'ACDEFGHIKLMNPQRSTVWY'
composition = {aa: 100 * s.count(aa) / len(s)
               for aa in alphabet}
print(pd.Series(composition).round(2))
```

Sources: UniProt: API | Biopython: ProtParam

Example 2: two mitochondrial genomes

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We will download human and chimpanzee reference records.

NC_012920.1 and NC_001643.1 contain sequences and their annotations.

We will compare genome length, GC content, and annotated CDS regions.

Sources: NCBI: human mtDNA | NCBI: chimpanzee mtDNA

Downloading a record from NCBI

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

EFetch returns a record in the selected format.

gbwithparts includes the sequence and its annotations.

The complete script downloads records for both identifiers.

```
accession = 'NC_001643.1'
r = requests.get(
    'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi',
    params={'db': 'nuccore', 'id': accession,
            'rettype': 'gbwithparts', 'retmode': 'text'},
    timeout=60)
r.raise_for_status()
(DATA / f'{accession}.gb').write_text(r.text)
```

Sources: NCBI: reference sequences | Biopython: annotations and features

Length and composition of both genomes

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Human: 16,569 nucleotides and 44.36% GC.

Chimpanzee: 16,554 nucleotides and 43.69% GC.

Each record contains 13 annotated CDS regions.

Sources: NCBI: reference sequences | Biopython: annotations and features

A table of coding-region lengths

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We select features marked as CDS.

The extract method obtains each sequence using its annotation.

The output contains species, gene name, and length in nucleotides.

```
cds_rows = []
for label, rec in mt.items():
    for f in rec.features:
        if f.type != 'CDS':
            continue
        gene = f.qualifiers.get('gene', ['?'])[0]
        seq = f.extract(rec.seq)
        cds_rows.append({'species': label, 'gene': gene,
                         'nt': len(seq)})
```

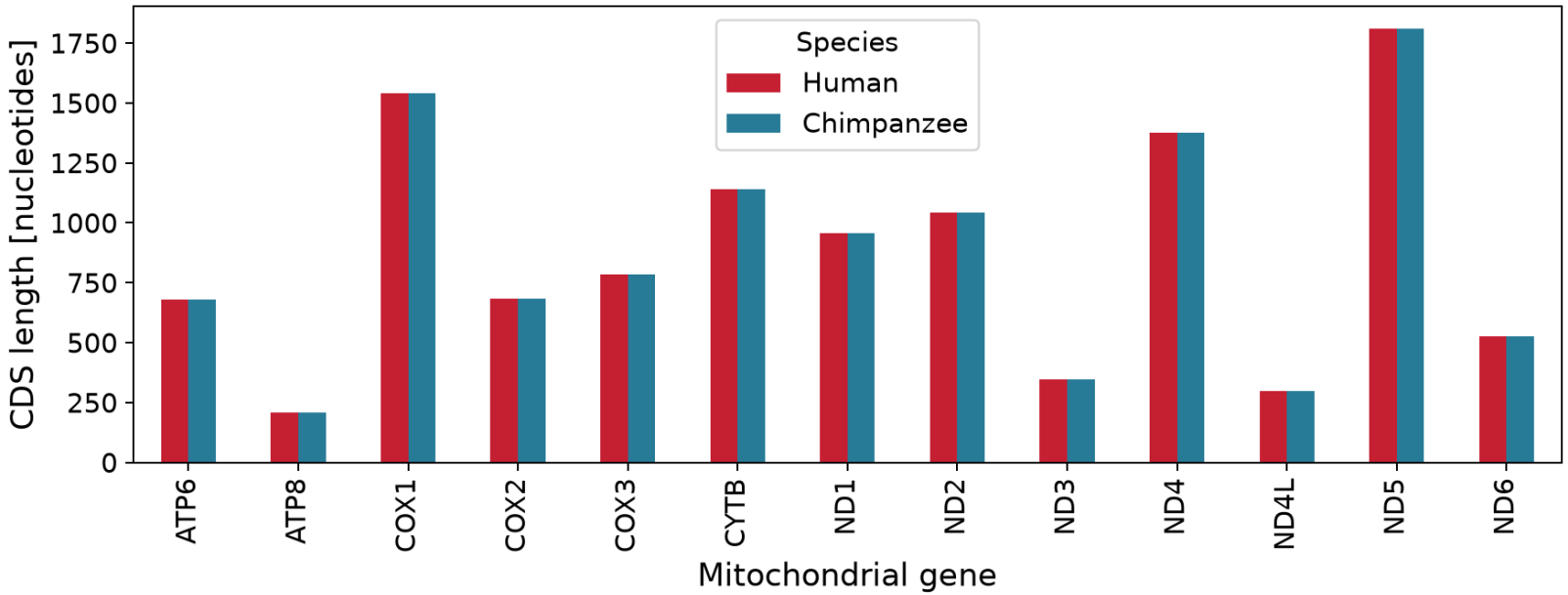
Sources: Biopython: annotations and features

Comparing mitochondrial CDS lengths

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each pair of bars represents the same gene.

The unit is the number of nucleotides in the annotated CDS region.



Sources: Biopython: annotations and features | NCBI: reference sequences

Example 3: aligning COX1 sequences

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

COX1 encodes a subunit of a protein complex in mitochondria.

We compare corresponding human and chimpanzee sequences.

An alignment places sequences together to compare corresponding positions.

Sources: Biopython: sequence alignment | Biopython: annotations and features

Global alignment in Biopython

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Global mode covers the full length of both sequences.

Match rewards and penalties for differences define the score.

The result depends on the sequences and the selected parameters.

```
aligner = Align.PairwiseAligner(
    mode='global', match_score=2, mismatch_score=-1,
    open_gap_score=-5, extend_gap_score=-0.5)
aln = aligner.align(cox['Human'], cox['Chimpanzee'])[0]
print(aln)
(OUT / 'cox1_alignment.txt').write_text(str(aln))
```

Sources: Biopython: sequence alignment

Identical positions in the DNA alignment

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The alignment has 1,542 columns and no gaps.

1,408 columns contain the same letter, and 134 contain different letters.

The percentage of identical columns is 91.31%.

```
a, b = aln[0], aln[1]
matches = sum(x == y for x, y in zip(a, b))
identity = 100 * matches / len(a)
print(len(a), matches, round(identity, 2))
# 1542 1408 91.31
```

Sources: Biopython: sequence alignment

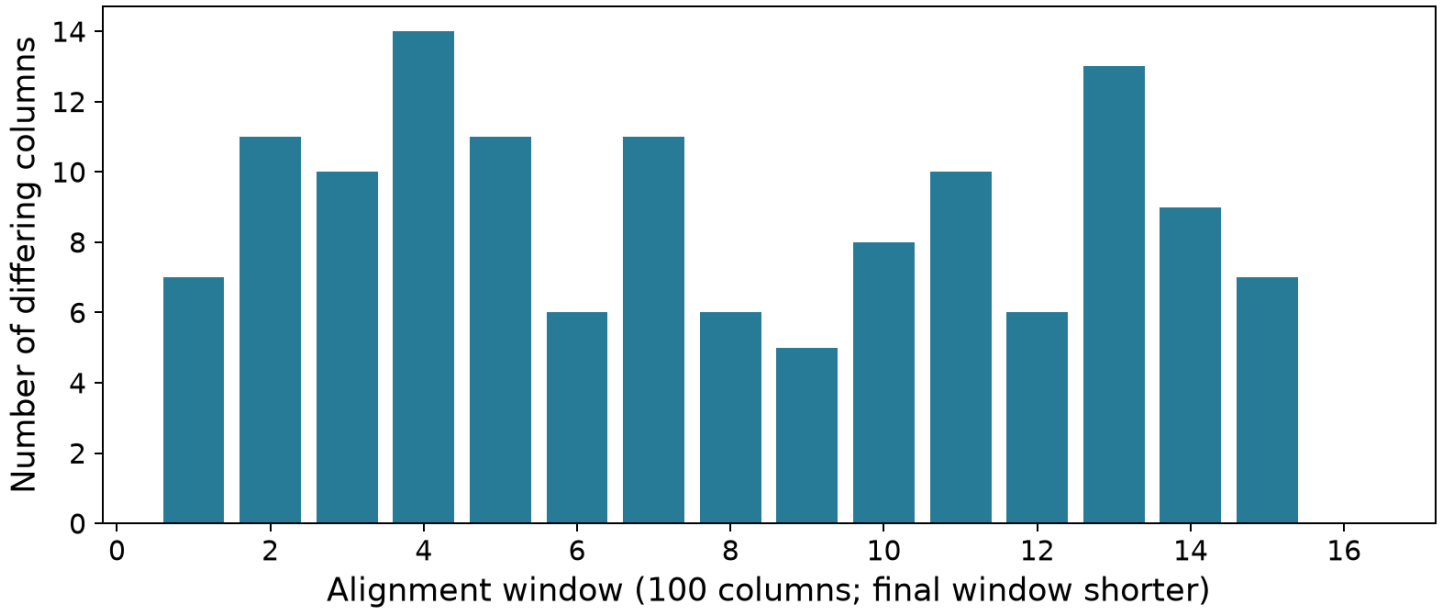
The distribution of sequence differences

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We divide the alignment into windows of 100 columns.

In each window, we count positions with different characters.

The final window is shorter, so we show counts rather than compare frequencies.



Sources: Biopython: sequence alignment

The same analysis at the protein level

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We translate the COX1 CDS using mitochondrial genetic code table 2.

Each protein sequence contains 513 amino acids.

The final stop codon does not represent an amino acid.

```
aas = {}
for label, rec in mt.items():
    f = next(f for f in rec.features if f.type == 'CDS'
             and f.qualifiers.get('gene') == ['COX1'])
    seq = f.extract(rec.seq)
    table = int(f.qualifiers['transl_table'][0])
    aas[label] = seq.translate(table=table, cds=True)
print([len(seq) for seq in aas.values()])  # [513, 513]
```

Sources: Biopython: annotations and features | NCBI: genetic code tables

DNA and protein comparisons give different results

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The COX1 proteins have 507 identical columns out of 513.

The percentage of identical columns is 98.83%.

The DNA result was 91.31%, with the stated settings for both alignments.

```
aa_aligner = Align.PairwiseAligner(
    mode='global', match_score=2, mismatch_score=-1,
    open_gap_score=-5, extend_gap_score=-0.5)
aa_aln = aa_aligner.align(aas['Human'], aas['Chimpanzee'])[0]
aa_a, aa_b = aa_aln[0], aa_aln[1]
aa_matches = sum(x == y for x, y in zip(aa_a, aa_b))
print(round(100 * aa_matches / len(aa_a), 2))  # 98.83
```

Sources: Biopython: sequence alignment

Three possible sequence-based projects

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

UniProt: compare the composition of a selected protein family.

RefSeq: examine gene lengths and composition in small genomes.

Alignments: compare homologous DNA and protein sequences.

Sources: UniProt: API | Biopython: annotations and features | Biopython: sequence alignment

Ensembl: one gene, many transcripts

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We analyse the human genes HBA1, HBB, and TP53.

A transcript is an RNA version recorded in the gene annotation.

We will compare transcript counts, exon counts, and sequence lengths.

Sources: Ensembl REST: genes and transcripts

Retrieving gene information through REST

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

lookup/symbol finds a gene by species and symbol.

The expand=1 parameter adds transcripts and exons.

We read the JSON response as a Python dictionary.

```
symbol = "HBB"
r = requests.get(
    f"{server}/lookup/symbol/homo_sapiens/{symbol}",
    params={"expand": 1, "content-type": "application/json"},
    timeout=60)
r.raise_for_status()
gene = r.json()
print(gene["id"], len(gene["Transcript"]))
```

Sources: Ensembl REST: genes and transcripts

Transcripts in a table

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each row describes one transcript.

We record its type, exon count, and protein length.

A missing Translation entry does not mean a protein of length zero.

```
for t in gene["Transcript"]:
    rows.append({
        'transcript': t["id"],
        'biotype': t["biotype"],
        'exons': len(t["Exon"]),
        'protein_length': t.get("Translation", {}).get("length")
    })
```

Sources: Ensembl REST: genes and transcripts

RNA versions recorded in the annotation

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

In the downloaded annotation, HBA1 has 26 transcripts, HBB has 19, and TP53 has 40.

Translation entries are present for 24, 18, and 37 transcripts, respectively.

These are counts of annotation entries, rather than expression levels.

```
summary = tab.groupby('gene').agg(
    transcripts=('transcript', "size"),
    with_protein=('protein_length', "count")
)
print(summary)
```

Sources: Ensembl REST: genes and transcripts

cDNA and CDS

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

In Ensembl, cDNA represents the transcript sequence, including UTR regions.

CDS is the protein-coding part.

We select sequences using specific transcript identifiers.

Sources: Ensembl REST: sequence types | Ensembl REST: genes and transcripts

Downloading a transcript sequence

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We use the same identifier with the cdna and cds options.

Lengths are measured in nucleotides.

We calculate GC content separately for the selected sequence type.

```
r = requests.get(f"{server}/sequence/id/{tid}",
    params={"type": "cds", "content-type": "text/plain"},
    timeout=60)
r.raise_for_status()
cds = r.text.strip()
print(len(cds), round(100 * gc_fraction(cds), 2))
```

Sources: Ensembl REST: sequence types

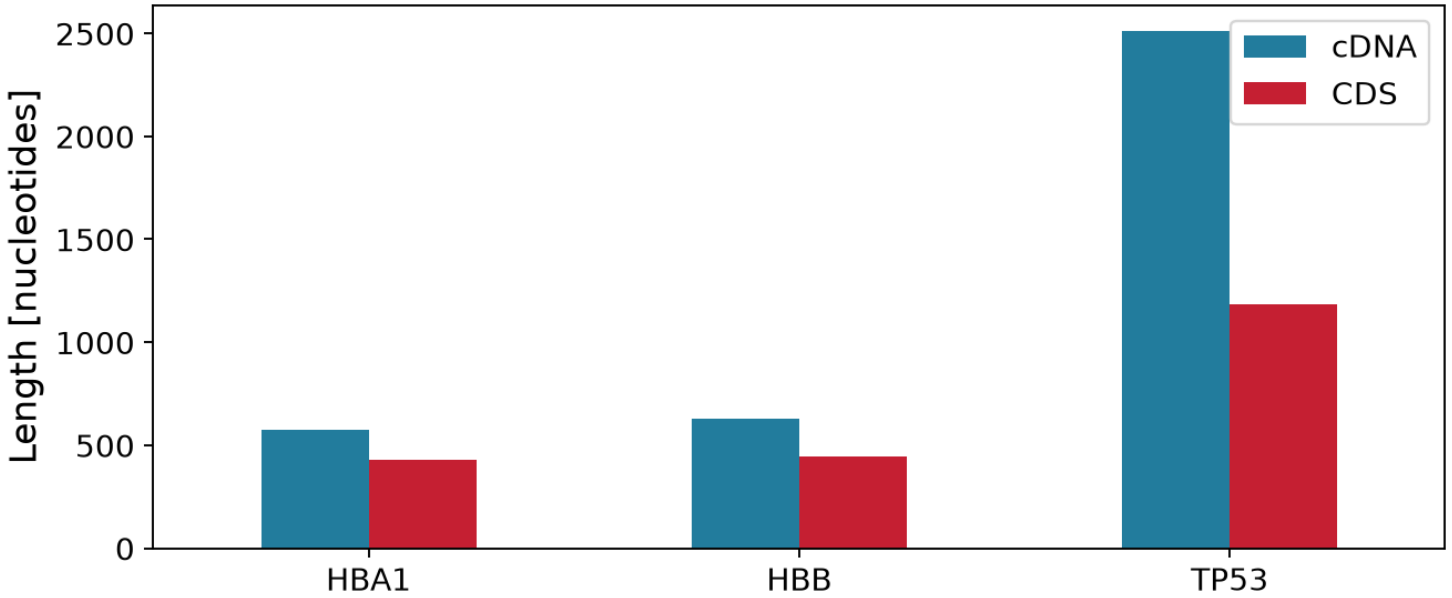
Comparing cDNA and CDS lengths

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The two bars show two sequence types for the same transcript.

The length difference covers the combined UTR regions.

This comparison describes sequence structure.



Sources: Ensembl REST: sequence types

ENA: a project without downloading gigabytes

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We compare metadata from PRJNA63817 and PRJNA229998.

Their reports describe WGS and RNA-Seq, respectively.

Small TSV tables are enough for this example, without downloading FASTQ files.

Sources: ENA: file reports

An ENA report as a pandas table

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We select a project and the fields to include in the report.

Each row describes one sequencing run.

We read the downloaded TSV directly, without an intermediate local file.

```
r = requests.get(
    "https://www.ebi.ac.uk/ena/portal/api/filereport",
    params={"accession": "PRJNA63817", "result": "read_run",
            "fields": fields, "format": "tsv"}, timeout=60)
r.raise_for_status()
ena = pd.read_csv(io.StringIO(r.text), sep="\t")
```

Sources: ENA: file reports

What can we calculate from run metadata?

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

PRJNA63817: 1 run and about 0.194 billion bases.

PRJNA229998: 16 runs and about 57.175 billion bases.

These figures describe the amount of recorded sequence data.

```
ena['gigabases'] = ena["base_count"] / 1_000_000_000
summary = ena.groupby('project').agg(
    runs=("run_accession", "size"),
    gigabases=('gigabases', "sum")
)
print(summary)
```

Sources: ENA: file reports

PDB: three proteins and their structure models

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

1GFL describes GFP, and 4HHB describes human haemoglobin.

1CRN is crambin, a small plant protein.

We will obtain methods and resolutions from metadata, and atom coordinates from mmCIF.

Sources: RCSB: Data API | RCSB: 1CRN

Structure metadata through the RCSB API

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The entry endpoint returns one record description as JSON.

We save three responses in separate files.

The same procedure works with a list of identifiers selected for a project.

```
base = 'https://data.rcsb.org/rest/v1/core/entry/'
ids = ['1GFL', '4HHB', '1CRN']
for pdb_id in ids:
    r = requests.get(base + pdb_id, timeout=60)
    r.raise_for_status()
    Path(f'data/{pdb_id}.json').write_bytes(r.content)
```

Sources: RCSB: Data API

A table of methods and resolutions

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

All three selected models were determined using X-ray diffraction.

Their resolutions are 1.90 Å, 1.74 Å, and 1.50 Å.

For this method, a lower value means that finer details can be distinguished.

```
rows = []
for pdb_id in ids:
    d = json.loads((D / f'{pdb_id}.json').read_text())
    rows.append({'PDB': pdb_id,
        'method': d['exptl'][0]['method'],
        'resolution_A': d['rcsb_entry_info']['resolution_combined'][0]})
meta = pd.DataFrame(rows)
```

Sources: RCSB: Data API

mmCIF: atom coordinates

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The 1CRN.cif file contains a crambin model with 46 amino acids.

The model is organised into chains, amino acid residues, and atoms.

Each atom has x, y, and z coordinates that we can use to calculate distances.

```
url = 'https://files.rcsb.org/download/1CRN.cif'
r = requests.get(url, timeout=60)
r.raise_for_status()
Path('data/1CRN.cif').write_bytes(r.content)
```

Sources: RCSB: 1CRN | Biopython: Bio.PDB

One atom to represent each amino acid

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We select model 0 and chain A.

The Cα atom is part of the backbone of every standard amino acid.

For 1CRN, we obtain 46 such atoms, one per position.

```
parser = MMCIFParser(QUIET=True)
structure = parser.get_structure('1CRN', 'data/1CRN.cif')
chain = structure[0]['A']
ca = [r['CA'] for r in chain if r.id[0] == ' ' and 'CA' in r]
print(len(ca))  # 46
```

Sources: Biopython: Bio.PDB | RCSB: 1CRN

Distance between the ends of a chain

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Subtracting Atom objects in Bio.PDB calculates their spatial distance.

The first and last Cα atoms in 1CRN are about 11.81 Å apart.

This is a distance in three-dimensional space.

```
distance = ca[0] - ca[-1]
print(round(float(distance), 2))  # 11.81 A
```

Sources: Biopython: Bio.PDB | RCSB: 1CRN

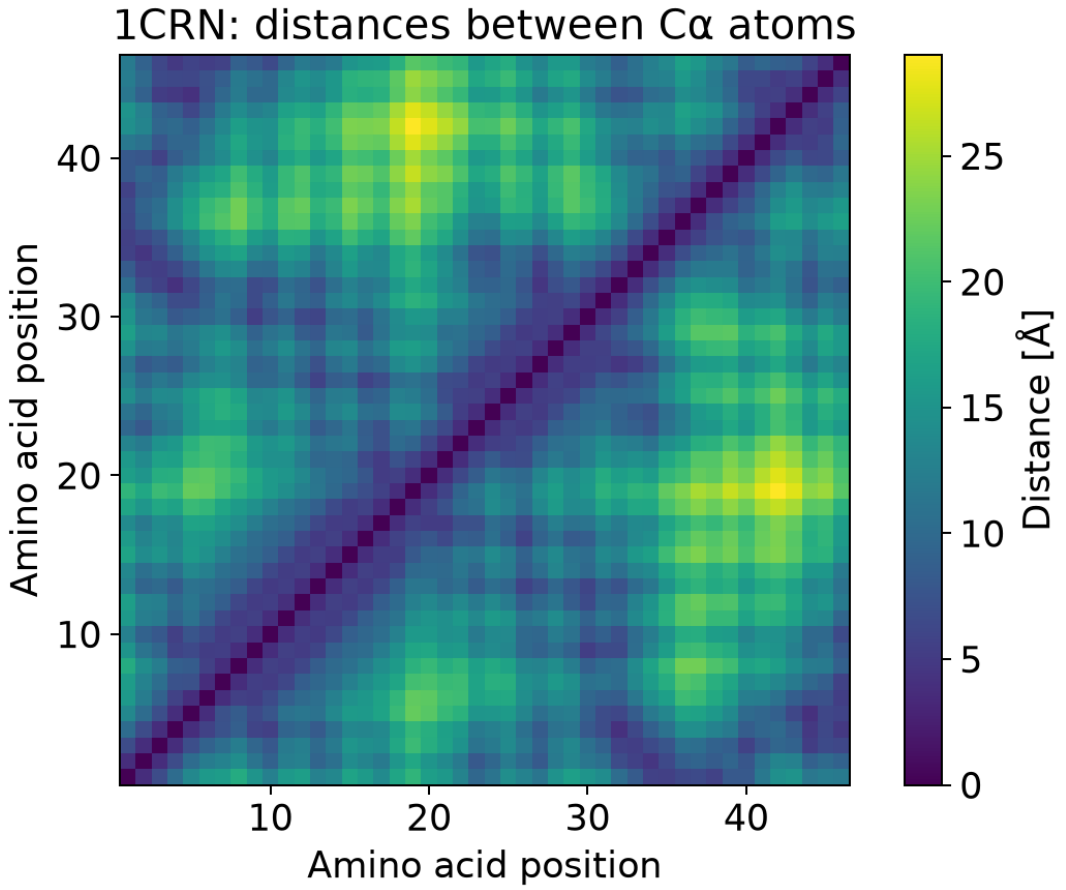
A protein distance map

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each cell shows the distance between two positions.

Distances on the diagonal are zero.

Darker areas indicate positions that are closer together in space.



Sources: Biopython: Bio.PDB | RCSB: 1CRN

A project using PDB structures

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

First analysis: summarise methods and resolutions for selected records.

Second analysis: compare the geometry of corresponding protein regions.

Possible outputs include a metadata table and a distance map.

Sources: RCSB: Data API | Biopython: Bio.PDB

GEO: RNA expression

GDS507 contains processed microarray data from kidney tissue.

The matrix contains 22,645 probe sets and 17 samples.

We compare signals from tumour tissue and adjacent tissue labelled as normal.

Sources: GEO: GDS507 | GEO: SOFT format

SOFT: study information and data in one file

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

A SOFT file contains descriptive sections and a data matrix.

We extract the text between the table's start and end markers.

pandas reads this section as a tab-separated table.

```
text = gzip.open('data/GDS507.soft.gz', 'rt').read()
table_block = text.split('!dataset_table_begin\n')[1]
table_block = table_block.split('!dataset_table_end')[0]
table = pd.read_csv(io.StringIO(table_block), sep='\t')
X = table.set_index('ID_REF').filter(regex='^GSM')
print(X.shape)  # (22645, 17)
```

Sources: GEO: GDS507 | GEO: SOFT format

Sample information

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

The RCC group contains 9 samples, and the normal group contains 8.

The metadata also includes a person identifier.

Seven people have both sample types, allowing paired comparisons.

```
sample_info = pd.read_csv(
    'data/samples.tsv', sep='\t', index_col=0, dtype=str)
print(sample_info.groupby('group').size())
# RCC       9
# normal    8
```

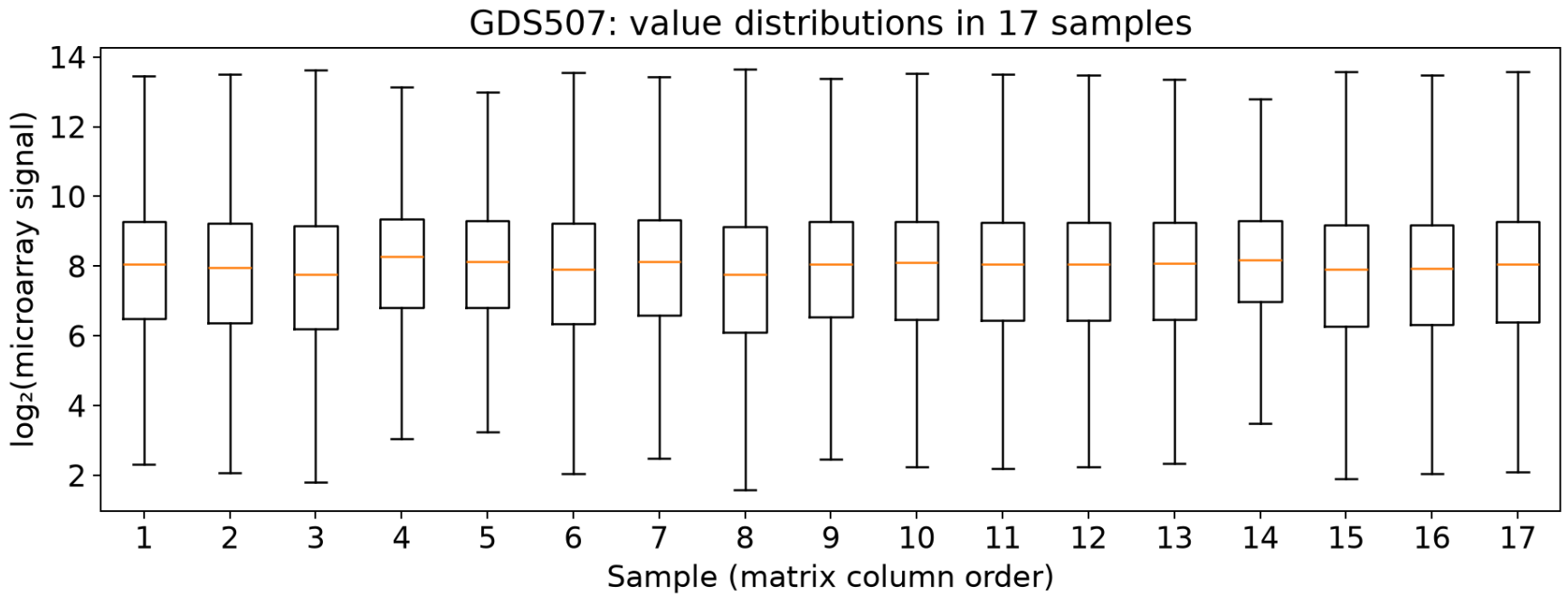
Sources: GEO: GDS507 | GEO: SOFT format

Logarithms make the data easier to view

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We use a base-two logarithm to display the data distribution.

A difference of 1 on this scale corresponds to a twofold change.



Sources: GEO: GDS507

Paired comparison of one probe set

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We select probe set 200000_s_at, annotated as PRPF8.

We compare the two tissue measurements for each person.

The difference in $\log_2$ signals equals $\log_2$ of their ratio.

```
probe_set = '200000_s_at'  # PRPF8, first row of the file
measurements = sample_info.copy()
measurements['log2_signal'] = Y.loc[probe_set]
pairs = measurements.pivot(
    index='person', columns='group', values='log2_signal')
pairs = pairs.dropna(subset=['normal', 'RCC'])
differences = pairs['RCC'] - pairs['normal']
print(len(pairs), round(differences.median(), 3))  # 7 0.383
```

Sources: GEO: GDS507

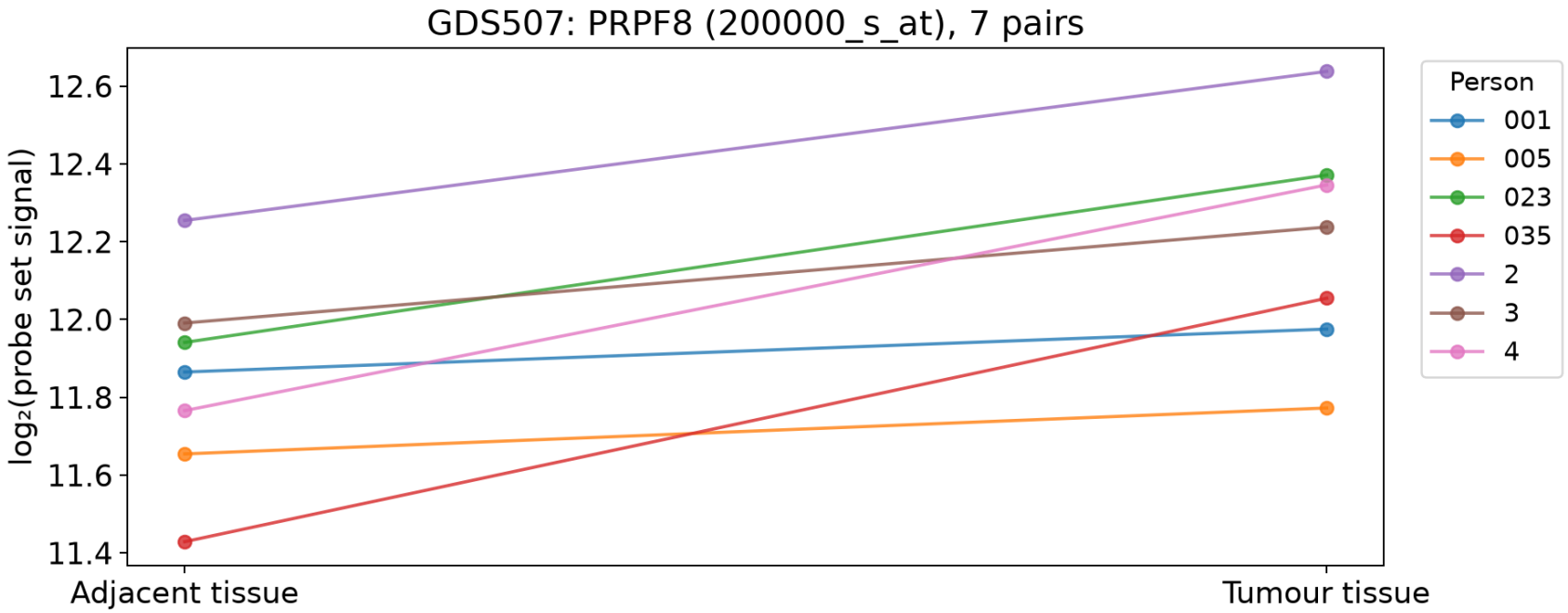
A paired chart shows each person

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Each line connects two samples from the same person.

For this probe set, all seven differences are positive.

The median $\log_2$ difference is about 0.383.



Sources: GEO: GDS507

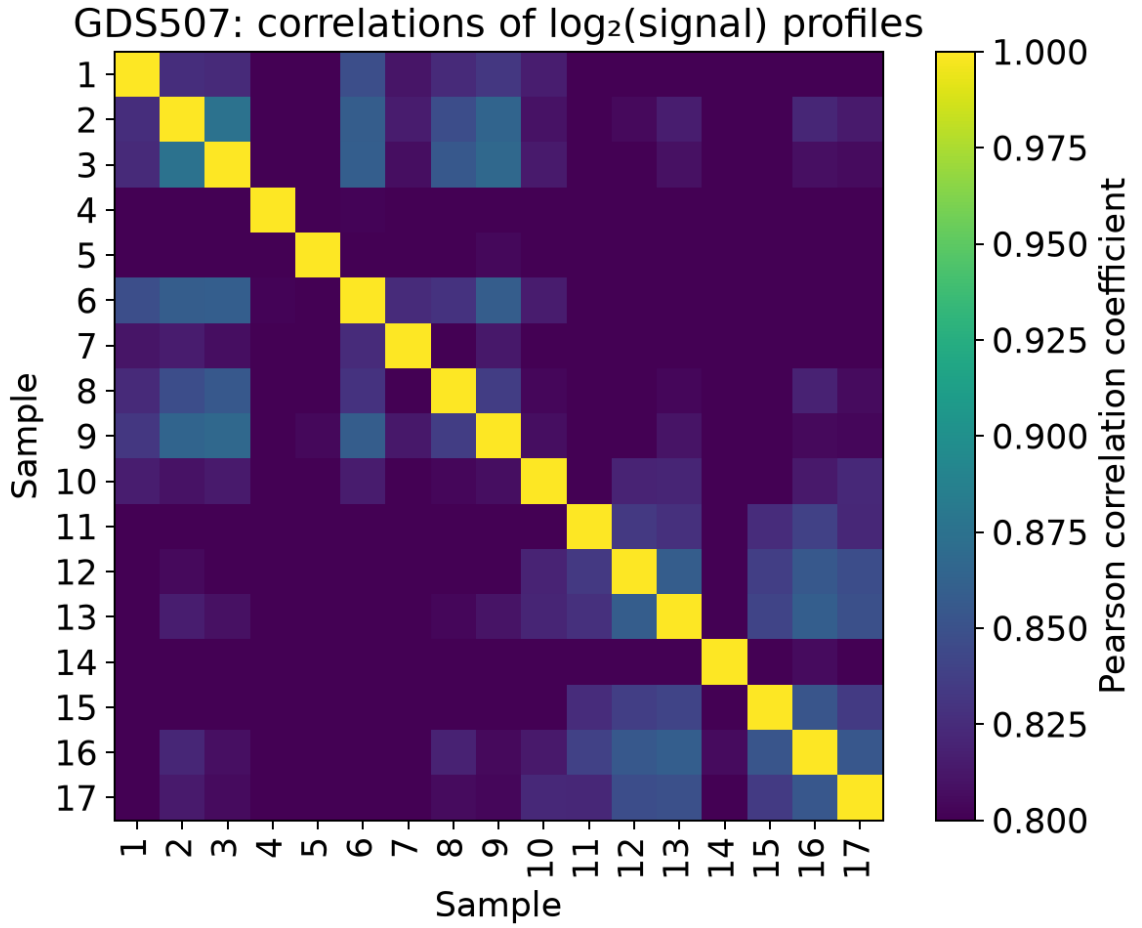
Similarity between sample profiles

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Correlation compares many probe set values between two samples.

We obtain a 17 × 17 matrix of Pearson correlation coefficients.

High correlation indicates a similar pattern of signals.



Sources: GEO: GDS507

A project using one GEO study

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We combine the measurement matrix with sample information.

We show data distributions, selected probe sets, and sample similarity.

The study design and available measurements define the scope of the analysis.

Sources: GEO: GDS507 | GEO: SOFT format

Six starting points for a project

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Protein properties and sequence similarity.

Transcript structure and sequencing metadata.

Three-dimensional structures and gene expression measurements.

Sources: Biopython: tutorial | Ensembl REST: genes and transcripts

NCBI Gene: gene descriptions for a project

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

We retrieve gene symbols and descriptions using Gene IDs.

3039 identifies HBA1, 3043 identifies HBB, and 7157 identifies TP53.

We can add this information to sequence analysis results.

```
d = json.loads((DATA / "ncbi_gene.json").read_text())["result"]
for uid in d["uids"]:
    gene = d[uid]
    print(uid, gene["name"], gene["description"])
```

Sources: NCBI: ESummary | NCBI Gene: HBB

POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Thank you for your attention

Basics of Bioinformatics

Python in bioinformatics - part two