Chapter 2-Bioinformatics Databases

Reza Rezazadegan
Shiraz University, Fall 2025
www.dreamintelligent.com

Learning Outcomes:

- **Define** bioinformatics databases and articulate their indispensable role in modern biological research.
- **Trace** the historical development of biological databases, recognizing key pioneers and early initiatives.
- **Differentiate** between various database structures (flat-file, relational, data warehouses, XML) and discuss their respective advantages and disadvantages.
- **Categorize** biological databases based on their content (primary, secondary, specialized) and provide examples for each.
- **Explain** how biological databases are interconnected and how integrated retrieval systems facilitate data access and knowledge discovery.
- **Discuss** the critical issues of data quality, redundancy, and annotation errors in biological databases, along with strategies for mitigation.
- **Utilize** Biopython tools for programmatic access and interaction with major online biological databases.

1. Introduction: Navigating the Ocean of Biological Data

The explosion of biological data, particularly from high-throughput sequencing projects since the 1970s, has created an unprecedented **"data deluge"**. This flood of information, ranging from nucleotide and protein sequences to gene expression profiles, is far too vast and complex for manual management or interpretation. This challenge led to the emergence of **bioinformatics databases** as critical infrastructure for modern biology.

- **What is a Biological Database?** A biological database is a **computerized archive specifically designed for the storage, retrieval, manipulation, and distribution of information related to biological macromolecules** such as DNA, RNA, and proteins. The emphasis on computers is paramount, as most genomic data analysis tasks are either *highly repetitive, mathematically complex, or involve datasets too large* for human processing.
- **Core Goals of Biological Databases**: The primary goals of these databases are twofold: **information retrieval** and **knowledge discovery**.
    1. **Information Retrieval**: This involves *straightforward querying* to find specific data points, such as a particular gene sequence or protein structure.
    2. **Knowledge Discovery**: This goes beyond simple retrieval, aiming to *identify previously unknown connections, patterns, or relationships* between data items that were not apparent when the data was first entered. For instance, raw sequence databases can perform computational tasks to *identify sequence homology or conserved motifs*, leading to new biological insights.
- **Databases as the Backbone of Bioinformatics**: Databases are the very starting point of much bioinformatics research. They are powerful tools for storing, sharing, and describing data, and for extracting information for further understanding and analysis. They can be regarded both as data repositories and as online libraries.

2. How Data is Organized: Types of Database Structures

Databases employ various underlying structures, each offering distinct advantages for storage, retrieval, and analysis. A **database management system (DBMS)** is the software used to control a computerized database.

1. **Flat-File Databases**:
    - **Description**: These are the simplest form, storing data as **plain text files**. Each record (or entry) is typically separated by a delimiter, and data within a record are organized into fields.
    - **Advantages**: Easy to create and distribute because they can be read and analyzed by many different programs without requiring specialized, often expensive, software.

- **Disadvantages**: Lack internal organization for efficient computer-based information retrieval, making complex queries on large datasets cumbersome.
- **Examples**: Margaret Dayhoff's *Atlas of Protein Sequence and Structure* was an early flat-file database. GenBank output files are still distributed in flat-file format.
- These flat file formats are discussed further in Section 5.

2. **Relational Databases**:
   - **Description**: Data are organized into a **collection of interconnected tables**. Each table comprises rows (records) and columns (fields). Tables are linked through shared fields called **keys** (often a primary key that is unique to each record). Figure 3.3 in provides an example with `protab1` and `protab2` linked by a `Protein-code` key.
   - **Advantages**: Allow for faster and more efficient retrieval of specific information by combining data from multiple tables through these keys. They support powerful **Structured Query Language (SQL)** for complex data manipulation and analysis.
   - **Example**: Many biological databases use this model. For instance, a query might extract protein names from `protab1` and sequences from `protab2` using the shared `protein-code`. The **mmCIF (macromolecular Crystallographic Information File)** format, used by the PDB, is similar to a relational database in its explicit tagging of each item of information.

Table A

Student #	Name	State
1	John Smith	Texas
2	Jane Doe	Kansas
3	William Brown	Illinois
4	Jennifer Taylor	New York
5	Howard Douglas	Texas

Table B

Student #	Course #
1	Biol 689
2	Bich 441
3	Chem 289
4	Hort 201
5	Math 172

Table C

Course #	Course name
Biol 689	Bioinformatics
Bich 441	Biochemistry
Chem 289	Organic chemistry
Hort 201	Horticulture
Math 172	Calculus

3. **Data Warehouses**:
   - **Description**: These systems integrate information from various sources into a **single, unified database**, typically to overcome data heterogeneity issues from multiple sources. They are, in effect, the opposite of distributed databases.
   - **Example**: The **Macromolecular Structure Database (MSD)** at EMBL-EBI uses a data warehouse model for its structural information, integrating various data types like secondary structure, active sites, and ligand information.

4. **eXtensible Markup Language (XML)**:
   - **Description**: XML is a powerful system for **marking up (annotating) data** using a plain file format, making files highly portable and accessible. It allows for the definition of arbitrary tags to classify data, unlike HTML, which uses a restricted set of tags for presentation.
   - **Advantages**: Its flexibility in defining bespoke data classifications makes it an attractive alternative for specialized data storage and exchange. Many bioinformatics databases are increasingly distributed in XML format.
   - **Applications**: XML plays a role in bridging databases with heterogeneous structures. The **Distributed Annotation System (DAS)**, for example, uses a specialized protocol for bioinformatics data exchange that allows integration of dispersed sequence annotation from multiple servers. Biopython's `Bio.Blast.NCBIXML` parser handles XML output from BLAST searches.

4. What Data is Stored: Categories of Biological Databases

Biological databases are commonly categorized based on the nature of their content.

1. **Primary Databases**:
   - **Definition**: These repositories contain **original, experimentally derived biological data**, submitted directly by the scientific community, with minimal annotation.
   - **Examples**:
     - **Nucleotide Sequence Databases**: **GenBank (NCBI)**, the **EMBL Nucleotide Sequence Database (EMBL-EBI)**, and the **DNA Data Bank of Japan (DDBJ)**. These three major databases form the International *Nucleotide Sequence Database Collaboration*, exchanging new data daily to ensure consistent and comprehensive access to nucleotide sequences worldwide. Sequence submission to one of these databases is often a precondition for publication in scientific journals.

- **Protein Data Bank (PDB)**: The global archive for **three-dimensional atomic coordinates of biological macromolecules** (proteins and nucleic acids) determined experimentally by methods like X-ray crystallography and NMR spectroscopy. PDB entries are identified by a unique four-character PDBid. Newer formats like **PDBx/mmCIF** and **MMDB** (used by NCBI for integration with Entrez) address limitations of the older PDB format, offering more flexibility and explicit tagging of data items.

2. **Secondary Databases**:
   - **Definition**: These databases contain **computationally processed, analyzed, or manually curated information** derived from primary sources. They offer enriched annotations, classifications, and derived insights. Because they rely on existing data, their entries can sometimes be proved incorrect if new primary data emerges.
   - **Examples**:
     - **Protein Sequence Databases**: **UniProt Knowledgebase (UniProtKB)**, a comprehensive protein resource combining **Swiss-Prot** and **TrEMBL**.
       - **Swiss-Prot**: Known for its **high-quality, manually curated, and extensively annotated protein sequences**. Annotation includes *function, domains, catalytic sites, disease associations, and cross-references.*
       - **TrEMBL**: Contains automatically annotated protein sequences translated from the EMBL nucleotide sequence database, with more entries but less accurate annotation than Swiss-Prot.
     - **Protein Families, Motifs, and Domains**:
       - **Pfam** and **BLOCKS**: Databases of *protein families and conserved regions* represented by multiple sequence alignments, profiles, or Hidden Markov Models (HMMs).
       - **PROSITE**: Focuses on motifs and patterns specific to protein families.
     - **Structural Classification Databases**: **SCOP (Structural Classification of Proteins)** and **CATH (Class, Architecture, Topology, Homologous)** classify protein folds and domains hierarchically.

3. **Specialized Databases**:
   - **Definition**: These databases cater to a **particular research interest, organism, or data type**. They may contain overlapping data with primary databases but often include unique organizations and additional expert annotations relevant to their niche.
   - **Examples**:
     - **Genome-Specific Databases**: Such as **Flybase** (for *Drosophila*), **WormBase** (*Caenorhabditis elegans*), **AceDB**, and **TAIR** (*Arabidopsis* information database).
     - **Expressed Sequence Tag (EST) Databases**: Like **dbEST** (NCBI), containing partial cDNA sequences that indicate gene expression.
     - **Microarray and Gene Expression Databases**: Repositories for gene and protein expression data, often with visualization and analysis tools. Examples include the **Microarray Gene Expression Database (EBI)**, **ArrayExpress**, and the **Stanford Microarray Database (SMD)**.
     - **Protein Interaction Databases**: Such as **DIP (Database of Interacting Proteins)**, **MINT (Molecular INTeraction database)**, **BIND (Biomolecular Interaction Network Database)**, and **pSTIING (Protein, Signaling, Transcriptional Interactions & Inflammation Networks Gateway)**, For most proteins to carry out their function they have to interact with other molecules, including other proteins.
     - **Ontologies**: Formal and explicit specifications of terms and relationships, such as the **Gene Ontology (GO)** project, which provides a *controlled vocabulary to describe gene and gene-associated information (molecular function, cellular component, biological process)* across organisms.
     - **Systems Biology Databases**: Resources like **KEGG (Kyoto Encyclopedia of Genes and Genomes)** organize genes into functional hierarchies and biochemical pathways. The **BioModels Database** stores curated, published, quantitative kinetic models of biological systems.

4. Key File Formats for Biological Sequences

Biological data is stored and exchanged using various file formats, each with its own structure and purpose.

- **FASTA Format**:
  - **Description**: One of the simplest and most widely used sequence formats in bioinformatics because it contains plain sequence

information that is readable by many analysis programs.

- **Structure**:

```
>gi|18203677|sp|Q9ZGE9|BCHN
MERVERENGCFHTFCPIASVAWLHRKIKDSFFLIVGTHTCAHFIQTALDVMVYAHSRFGFAVLEESDLVS
ASPTEELGKVVQQVVDEWHPKVIFVLSTCSVDILKMDLEVSCKDLSTRFGFPVLPASTSGIDRSFTQGED
AVLHALLPFVPKEAPAVEPVEEKKPRWFSFGKESEKEKAEPARNLVLIGAVTDSTIQQLQWELKQLGLPK
VDVFPDGDIRKMPVINEQTVVVPLQPYLNDTLATIRRERRAKVLSTVFPIGPDGTARFLEAICLEFGLDT
SRIKEKEAQAWRDLEPQLQILRGKKIMFLGDNLLELPLARFLTSCDVQVVEAGTPYIHSKDLQQELELLK
ERDVRIVESPDFTKQLQRMQEYKPDLVVAGLGICNPLEAMGFTTAWSIEFTFAQIHGFVNAIDLIKLFTK
PLLKRQALMEHGWAEAGWLE
```

- Each record begins with a **definition line** (or header) that starts with a **right angle bracket (>)**.
- The angle bracket is followed by a **sequence name**.
- Sometimes, **extra information** like a GI number (GenInfo identifier) or comments can be included on the definition line, often separated from the sequence name by a `|` symbol. This extra information is generally considered optional and is often ignored by sequence analysis programs.
- The actual **plain sequence** in standard one-letter symbols (e.g., A, T, C, G for DNA; amino acid codes for proteins) starts on the second line.
- Each line of sequence data is typically limited to **sixty to eighty characters** in width.

- **Advantages**: Simple, human-readable, widely compatible with many bioinformatics analysis programs.

- **Disadvantages**: Lacks much structured annotation information. Extracting specific details from the variable definition line can be challenging due to format variations from different sources.

- **Biopython Handling**: `Bio.SeqIO.parse()` and `Bio.SeqIO.read()` fully support FASTA files. For large files, `SimpleFastaParser` can provide faster processing by returning simple tuples of strings.

• **GenBank Format**:

- **Description**: A very popular and richly annotated method of holding information about sequences, their features, and associated biological details. Although GenBank is internally a relational database, its search outputs for sequence files are often produced as flat files for ease of reading.

GenBank Flat-File Format

```
LOCUS       SCU49845     5028 bp    DNA
DEFINITION  Saccharomyces cerevisiae TCP1-beta gene, partial cds, and
             Axl2p
            (AXL2) and Rev7p (REV7) genes, complete cds.
ACCESSION   U49845
VERSION     U49845.1  GI:1293613
KEYWORDS    .
SOURCE      Saccharomyces cerevisiae (baker's yeast)
  ORGANISM  Saccharomyces cerevisiae
            Eukaryota; Fungi; Ascomycota; Saccharomycotina;
           Saccharomycetes;
            Saccharomycetales; Saccharomycetaceae; Saccharomyces.
```

- **Structure**: A GenBank flat file typically consists of three major sections:

```
  - **Header**: Provides an overview of the record, including critical identifiers such as a unique **accession number** (which remains stable), a **version number** (incremented with annotation revisions), and a **gene index (GI) number** (also incremented). It also includes the `DEFINITION` (description), `SOURCE` organism, taxonomy ID, and bibliographic references.

  - **Features**: This highly informative section contains detailed annotation about the gene and gene product, as well as regions of biological significance reported in the sequence. It uses unique identifiers and qualifiers. Key fields include `gene` (gene name), `CDS` (coding sequence boundaries), and information about exon locations for eukaryotic DNA.

  - **Sequence Entry ("ORIGIN")**: This section contains the nucleotide sequence itself, often accompanied by a `BASE COUNT` report (numbers of A, G, C, T). This section (and the entire record) ends with two forward slashes (`//`).
```

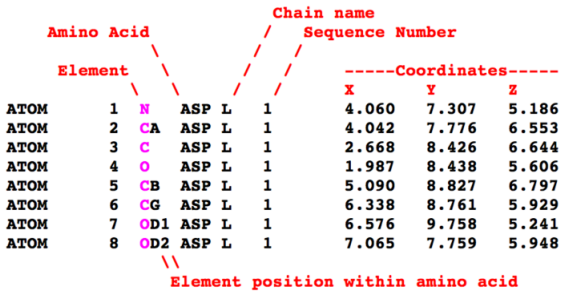
- **Advantages**: Provides rich, structured, and detailed annotation, making it highly valuable for comprehensive analysis. Its structure allows for relatively easy indexing and computer parsing.

- **Disadvantages**: Records can be very long and complex, potentially challenging for manual interpretation.

- **Biopython Handling**: `Bio.SeqIO.parse()` and `Bio.SeqIO.read()` fully support GenBank files. The `SeqRecord` object is ideal for capturing its rich annotation, including `.features`, `.dbxrefs`, and the `.annotations` dictionary. `Bio.Entrez.efetch()` can download GenBank records.

• **PDB (Protein Data Bank) Format**:

- **Description**: The standard format for experimentally determined 3D structures of biological macromolecules.



- **Structure**: Historically, a rigid 80-character-per-line structure with an explanatory header section (overview of protein, quality, methods) followed by an atomic coordinate section. The `ATOM` part refers to protein atom information, while `HETATM` refers to atoms of cofactors or substrates.

- **Limitations**: The original PDB format has inherent limitations, such as restricted field widths (e.g., max 99,999 atoms, 26 chains), which complicated representing large complexes like ribosomes. It also lacks explicit bonding information and is difficult for computer software to parse efficiently.

- **Newer Formats**: To overcome these limitations, new formats have been developed:

- **MMDB (Molecular Modeling Database)**: Developed by NCBI, it is written in the **ASN.1 format** and includes bond connectivity information, allowing for faster retrieval and easier structure drawing.

- **Biopython Handling**: The `Bio.PDB` module is specifically designed to work with PDB files. It can parse PDB and mmCIF files, manipulate atomic coordinates based on an SMCRA (Structure/Model/Chain/Residue/Atom) architecture, and download structures directly from the PDB.

- **Other Important Formats**:

- **FASTQ Format**: Widely used for holding nucleotide sequencing reads with associated quality scores. It stores DNA sequence and per-letter quality scores in a single plain text file. Biopython supports various FASTQ variants.

5. Accessing and Interconnecting Databases: A Network of Knowledge

Biological databases are rarely isolated; they are increasingly interconnected, forming a vast network of information. This interconnectedness is crucial for comprehensive data gathering and knowledge discovery.

- **Local vs. Online Access**: Databases can be accessed locally (faster, more flexible queries, security) or externally via the Internet (preferred by most users due to ease of access and maintenance). Major bioinformatics hubs like NCBI and EMBL-EBI offer web-based interfaces to numerous databases.
- **Linking and Cross-Referencing**: Most biomedical databases include **links** (URL pointers or identifiers) to relevant entries in other databases. This cross-referencing allows users to efficiently navigate between different types of information (e.g., *from a gene sequence to its translated protein, associated literature, or 3D structure*). This integrated network greatly enhances the power of public databases as a research resource.
- **Integrated Retrieval Systems**:
  - **Entrez (NCBI)**: Developed and maintained by the NCBI, Entrez is a powerful gateway providing text-based searches and integrated *access to a wide variety of NCBI databases*. Databases accessible through Entrez include PubMed (biomedical literature), GenBank (nucleotide sequences), GEO (Gene Expression Omnibus), OMIM (human disease genes), the Taxonomy database, and the PDB. Entrez integrates information through cross-referencing based on logical relationships between individual entries. Users can perform complex queries using **Boolean operators** (AND, OR, NOT) and **field qualifiers** (e.g.,

`Jones[AUTH]` for author searches, `200000[SeqLength]` for sequence length). The "WebEnv history feature" helps manage complex search results by allowing users to fetch results by reference to previous searches.
    - **Sequence Retrieval Systems (SRS)**: Another integrated system that provides access to multiple databases for retrieval of search results. SRS allows complex queries like finding human genes larger than 200 kilobase pairs with poly-A signals.
    - **ExPASy** (Swiss Institute of Bioinformatics): A proteomics database and portal for accessing tools and databases (like Swiss-Prot and PROSITE).
- **Biopython for Database Access**: The Biopython library provides convenient programmatic interfaces to interact with these online databases.
    - `Bio.Entrez`: Enables access to NCBI's Entrez databases from Python scripts, allowing users to search PubMed, download GenBank records (`efetch`), obtain database information (`einfo`), search related items (`elink`), and more.
    - `Bio.ExPASy`: Provides code to extract information from ExPASy, including Swiss-Prot entries and Prosite searches.
    - `Bio.PDB`: A module focused on working with crystal structures, allowing downloading of structures from the PDB.
    - `Bio.SeqIO`: A module for reading and writing sequence files in various formats (e.g., FASTA, GenBank). It can parse downloaded records from Entrez.
    - `Bio.SearchIO`: A submodule for parsing results from various sequence search tools like BLAST and BLAT, handling different output formats and statistics.
    - `Bio.AlignIO`: For reading and writing multiple sequence alignment files.

6. Data Quality and Pitfalls: Challenges in Database Management

Despite their immense utility, biological databases are not infallible. It is crucial to recognize their **inherent limitations** and approach computational predictions with a critical eye.

- **Redundancy**:
    - **Problem**: Primary sequence databases often contain numerous identical or nearly identical entries due to repeated submissions, overlapping sequences, or different versions of the same data. This can make databases excessively large and unwieldy for information retrieval. For example, well-studied proteins like hemoglobins and myoglobins can have hundreds of entries in PDB.
    - **Solutions**:
        - **Nonredundant Databases**: Efforts are made to reduce redundancy by creating databases like **RefSeq (NCBI)**, which merges identical sequences from the same organism into single entries.
        - **Sequence Clustering**: Databases like **UniGene** (NCBI) cluster EST sequences derived from the same gene.
        - **Careful Curation**: Swiss-Prot is known for its minimal redundancy due to careful curation.
- **Erroneous or Incomplete Annotations**:
    - **Problem**: Errors in original sequence data or their annotations can propagate through linked databases, leading to misleading research conclusions. Common issues include:
        - Misnaming genes or having unrelated genes with the same name, causing confusion.
        - Incomplete or incorrect functional assignments based on automated predictions.
    - **Solutions**:
        - **Manual Curation**: Exemplified by Swiss-Prot, *human experts meticulously review and annotate entries*, producing the highest quality and most accurate information, although this is time-consuming.
        - **Automated Consistency Checks**: Databases can automatically check for data consistency, such as valid base characters in DNA sequences, agreement between sequence length and molecular weight, or the existence of cross-references. These checks typically identify errors for manual resolution.
        - **Controlled Vocabularies**: To alleviate gene naming problems, ontologies like **Gene Ontology (GO)** provide consistent and unambiguous naming systems.
        - **Data Standards**: Standards like **MIAME (Minimum Information About a Microarray Experiment)** aim to ensure comprehensive reporting of experimental details for high-throughput data, crucial for interpretation and reproducibility.
- **Outdated Information**:

    - **Problem**: Databases need **regular updates** to incorporate new findings, correct existing entries, and reflect evolving scientific understanding. Analysis based on outdated data can be incorrect.
    - **Solution**: Databases implement version numbering (e.g., accession number, version number, GI number) to track changes, and major sites like NCBI and EBI are regularly updated.
- **Algorithmic Limitations**: Bioinformatics predictions are not formal proofs; they are computational hypotheses that **require experimental verification**. The quality of predictions depends on the sophistication of algorithms, many of which are still under development. Critical interpretation of results and a realistic perspective on the role of bioinformatics are paramount.

7. Active Learning Components for Enhanced Understanding

Python Workshop

- **Online Resources**: Explore the major database websites (NCBI, EBI, PDB, UniProtKB, Expasy) and their documentation (e.g., Entrez help pages, `Bio.PDB` documentation).
- **Code Challenges**: Students can implement simple database querying scripts (e.g., using `Bio.Entrez` or parsing flat files) or create basic in-memory "databases" using Python dictionaries, then test them on Rosalind.
- **Final Challenges**: Students design a comprehensive strategy to find all known information (sequences, structures, interactions, literature) for a specific protein from a new organism, using multiple databases and Biopython tools.

Conclusion:

Bioinformatics databases are the indispensable digital repositories that underpin nearly all modern biological research. From their origins with pioneers like Margaret Dayhoff to their current sophisticated structures and interconnections, these databases have transformed biology into an information-rich science. Understanding their diverse structures, content, access methods, and inherent challenges related to data quality is paramount for any aspiring bioinformatician or biologist. By critically engaging with these vast resources, and leveraging computational tools like Biopython, researchers can effectively retrieve information, discover new knowledge, and drive the future of biological inquiry.