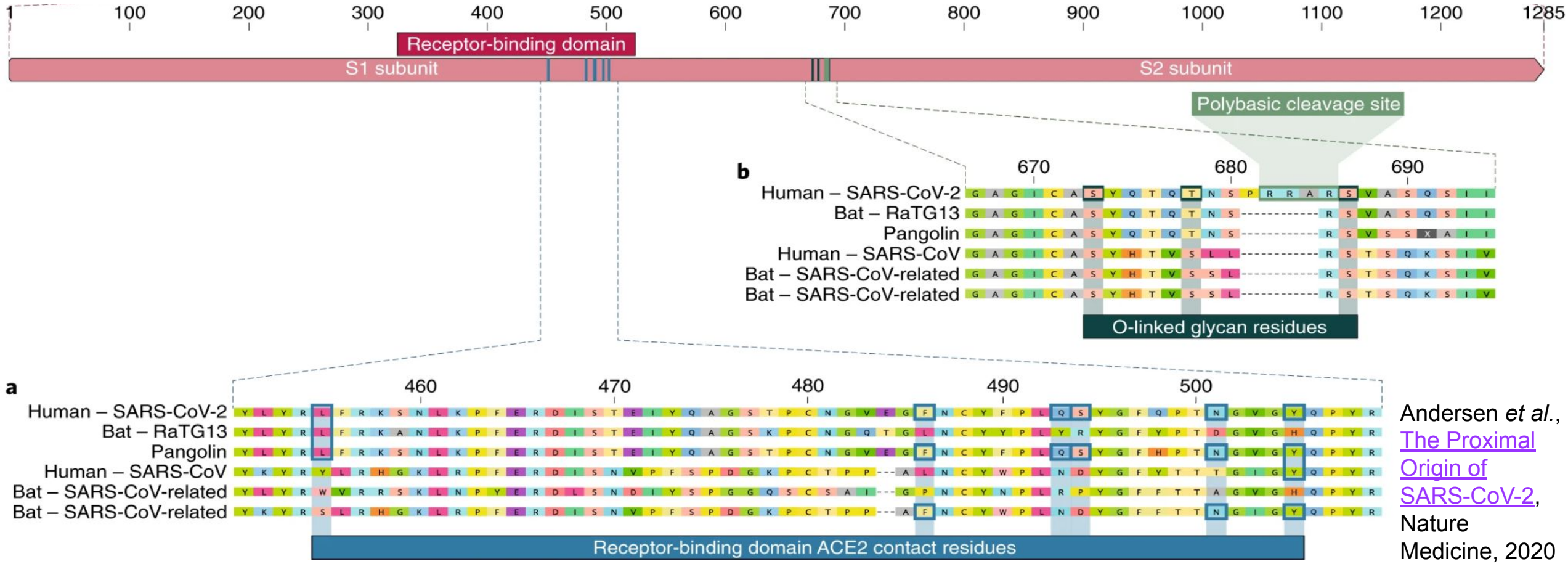
AMIDD Lecture 4: From Sequences To Structures



Andersen *et al.*, The Proximal Origin of SARS-CoV-2, Nature Medicine, 2020

Dr. Jitao David Zhang, Computational Biologist

[1] Pharmaceutical Sciences, Pharma Research and Early Development, Roche Innovation Center Basel, F. Hoffmann-La Roche

[2] Department of Mathematics and Informatics, University of Basel

Today's goals

- Sequence analysis with dynamic programming
- Sequence analysis with hidden Markov chains
- Protein domains and structure

Recap

We learned important concepts in drug discovery:

- *Indication*
- *Gene target*
- *Biological pathway and network*
- *Mechanism of Action*
- *Formulation*
- *Dosing regimen*

We learned about mutations

- Mutations and natural selection drive evolution.
- Mutations in either pathogens or human can cause diseases of varying severities.
- Presence and absence of mutations in a gene in a healthy human population inform about its evolutionary selection pressure.
- Levenshtein distance models biological sequences with three basic operations: *insertion, deletion,* and *substitution.*

Questions from the last lecture

Why beagles? It follows a historical standardization process. The beagles were chosen for several reasons, among others

- They are cooperative instead of aggressive;
- They are of moderate size, not too big nor too small.

Based on these and other reasons, breeders such as *Marshall Farms* chose Beagles and offered them to research labs around the world. The standardization simplifies data comparison, interpretation, and reproducibility. (shared by Dr. Tobias Schnitzer).

About the gene population study, what is so special about the genes that don't allow for mutations? Can we learn something from them about preventing/improving mutations on the other genes that cause disease?

Genes do not allow for mutations may be under strong evolutionary pressure (i.e. no mutation is tolerated up to reproductive age). A possible application of this piece of knowledge is that If a drug candidate modulate the function of such genes, its safety profile may need special attention.

With *Levenshtein distance* we can compare any two pieces of DNA

Levenshtein distance: The minimum number of operations required to transform string a to string b with following operations:

- Insertion, *e.g.* **bat → bait**
- Deletion, *e.g.* **boat → bot**
- Substitution, *e.g.* **pig → big**

The Levenshtein distance between two strings a, b of length $|a|$ and $|b|$ respectively is given by $\mathrm{lev}_{a,b}(|a|, |b|)$ where

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

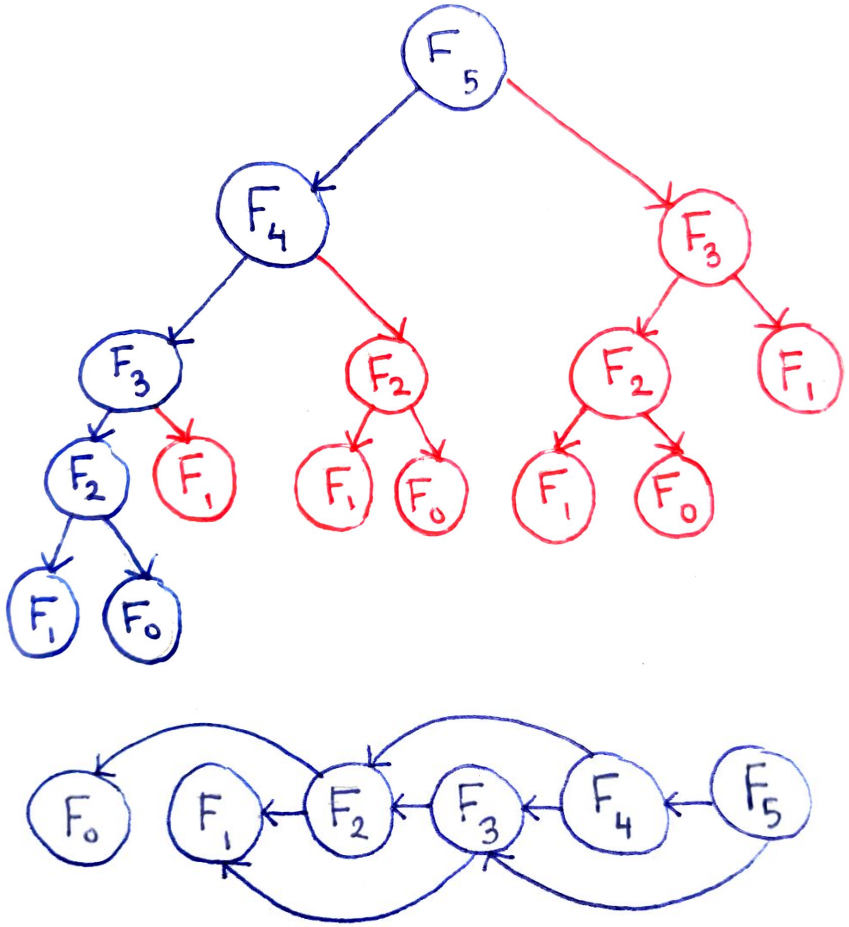
where $1_{(a_i \neq b_j)}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise, and $\mathrm{lev}_{a,b}(i,j)$ is the distance between the first i characters of a and the first j characters of b.

Dynamic programming turns a exponential problem into linear

```python
def recursive_fib(n: int):
    """A recursive version of Fibonacci"""
    if n <= 1:
        return n
    return recursive_fib(n-1) + recursive_fib(n-2)


def dp_fib_ls(n: int):
    """A dynamic programming version of Fibonacci, linear space"""
    res = [0, 1]
    for i in range(2, n+1):
        res.append(res[i-2] + res[i-1])

    return res[n]
```

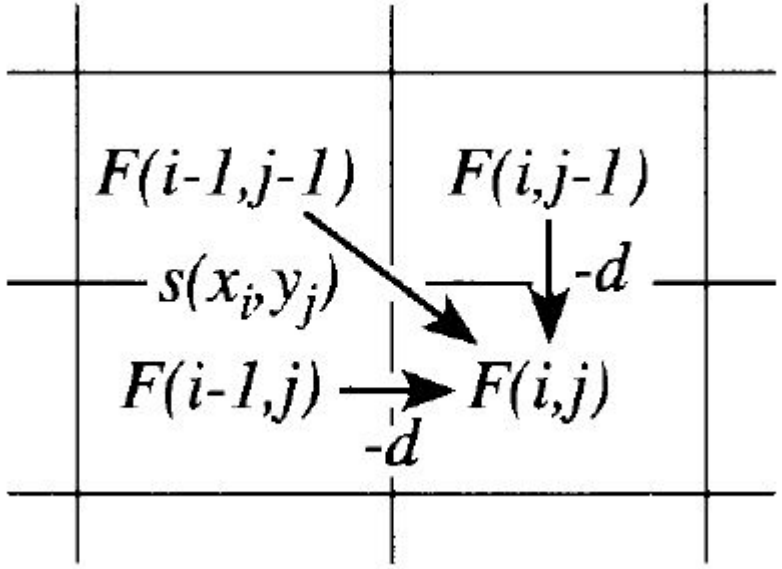


Directed Acyclic Graphs (DAGs)
by Avik Das

For fib(25), I observed a difference in the order of 10^4 in run time.
Check out the source code for the constant-space version.

Dynamic programming for biological sequence analysis

- The calculation of Levenshtein distance is *repeated* to fill a matrix of distances between two sequences.
- At each step, we keep a pointer in each cell back to the cell from which its value is derived.
- The value in the final (bottom right) cell of the matrix is the best score of an alignment.
- The best alignment is done by *traceback*, i.e. building the alignment in reverse, starting from the final cell, and following the pointers of the optimal path.



Chapter 2, Durbin, Richard, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge: Cambridge University Press, 1998. https://doi.org/10.1017/CBO9780511790492.

Calculate the Levenshtein distance with dynamic programming

What is the Levenshtein distance between `ATGC` and `AGC`?

		A	T	G	C
	0				
A					
G					
C					

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

where $1_{(a_i \neq b_j)}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise, and $\mathrm{lev}_{a,b}(i,j)$ is the distance between the first i characters of a and the first j characters of b.

Solution:

`ATGC`

`A-GC`

Dynamic programming breaks down a complex problem in sub-problems.

With a cost function (e.g. the Levenshtein distance), we enumerate all possible operations *at any position*, and record the operation that minimizes the cost. When we finish all positions, we find the solution(s) by working backwards the optimal path(s).

Calculate the Levenshtein distance with dynamic programming

What is the Levenshtein distance between `ACTGCTT` and `ACATT`?

		A	C	T	G	C	T	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
C	2	1	0	1	2	3	4	5
A	3	2	1	1	2	3	4	5
T	4	3	2	1	2	3	3	3
T	5	4	3	2	2	3	3	3

```
ACTGCTT          ACTGCTT          ACTGCTT
AC--ATT          ACA--TT          AC-A-TT
```

The Needleman-Wunsch algorithm uses dynamic programming for *global alignment* of two sequences

Compared with the Levenshtein distance, the Needleman-Wunsch algorithm uses biologically meaningful parameters to score insertion or deletion (gap penalty *d*), and substitution or mutation events (a substitution matrix *M*). The dynamic programming technique is used in a similar way.

Task: align two sequences `ATCGAC` and `CATAC`.

Parameters: d=-4, $M = \begin{pmatrix} & A & C & T & G \\ A & 5 & -3 & -3 & -3 \\ C & -3 & 5 & -3 & -3 \\ T & -3 & -3 & 5 & -3 \\ G & -3 & -3 & -3 & 5 \end{pmatrix}$

Solution:

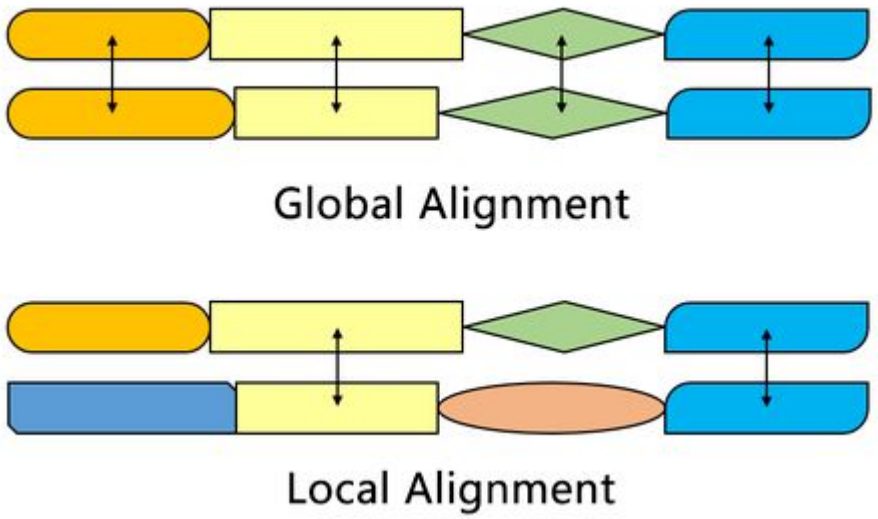
```
ATCGAC
||  ||
CAT--AC
```

-	-	A	T	C	G	A	C
-	0	-4	-8	-12	-16	-20	-24
C	-4	-3	-7	-3	-7	-11	-15
A	-8	1	-3	-7	-6	-2	-6
T	-12	-3	6	2	-2	-6	-5
A	-16	-7	2	3	-1	3	-1
C	-20	-11	-2	-1	0	-1	8

Source: https://commons.wikimedia.org/wiki/File:Needleman-wunsch.jpg

Check this video out if a step-by-step tutorial may help you: Link to YouTube (thanks to the contribution of Robert Do)

The Smith-Waterman algorithm uses dynamic programming for *local alignment* of two sequences



	Needleman-Wunsch	**Smith-Waterman**
Initialization	Gap penalty in first column and first row	0 in first column and first row
Scoring	Scores can be negative	Negative scores are set to 0
Traceback	Begin at the bottom right, and end at the top left cell.	Begin at the cell with the highest score, end when 0 is met.

Major differences between the Needleman-Wunsch and the Smith-Waterman algorithm. See this animation for an example.

Richard Bell on the origin of the name *Dynamic Programming*

I spent the Fall quarter (of 1950) at RAND. My first task was to find a name for multistage decision processes. An interesting question is, Where did the name, dynamic programming, come from?

The 1950s were not good years for mathematical research. We had a very interesting gentleman in Washington named Wilson. He was Secretary of Defense, and he actually had a pathological fear and hatred of the word, research. I'm not using the term lightly; I'm using it precisely. His face would suffuse, he would turn red, and he would get violent if people used the term, research, in his presence. You can imagine how he felt, then, about the term, mathematical. The RAND Corporation was employed by the Air Force, and the Air Force had Wilson as its boss, essentially. Hence, I felt I had to do something to shield Wilson and the Air Force from the fact that I was really doing mathematics inside the RAND Corporation. What title, what name, could I choose? In the first place I was interested in planning, in decision making, in thinking. But planning, is not a good word for various reasons. I decided therefore to use the word, "programming" I wanted to get across the idea that this was dynamic, this was multistage, this was time-varying I thought, lets kill two birds with one stone. Lets take a word that has an absolutely precise meaning, namely dynamic, in the classical physical sense. It also has a very interesting property as an adjective, and that is its impossible to use the word, dynamic, in a pejorative sense. Try thinking of some combination that will possibly give it a pejorative meaning. Its impossible. Thus, I thought dynamic programming was a good name. It was something not even a Congressman could object to. So I used it as an umbrella for my activities.

Dreyfus, Stuart. "Richard Bellman on the Birth of Dynamic Programming." Operations Research 50, no. 1 (February 2002): 48–51. https://doi.org/10.1287/opre.50.1.48.17791.

Sequence alignment is fundamental for many bioinformatics tasks and tools

Examples:

- BLAST (Basic Local Alignment Search Tool) is used for almost all biological sequence analysis tasks. At its core, a heuristic algorithm approximates the Smith-Waterman algorithm for local alignment.
- Software tools such as Bowtie/Bowtie2 (Langmead *et al.*, Genome Biology, 2009; source code on GitHub), STAR (Dobin *et al.*, Bioinformatics, 2013; source code on GitHub) and GSNAP (web link) use more sophisticated methods to map sequencing reads (usually ~30-200 nucleotides) to large genomes (*e.g.* ~10^9 base pairs of mouse or human).

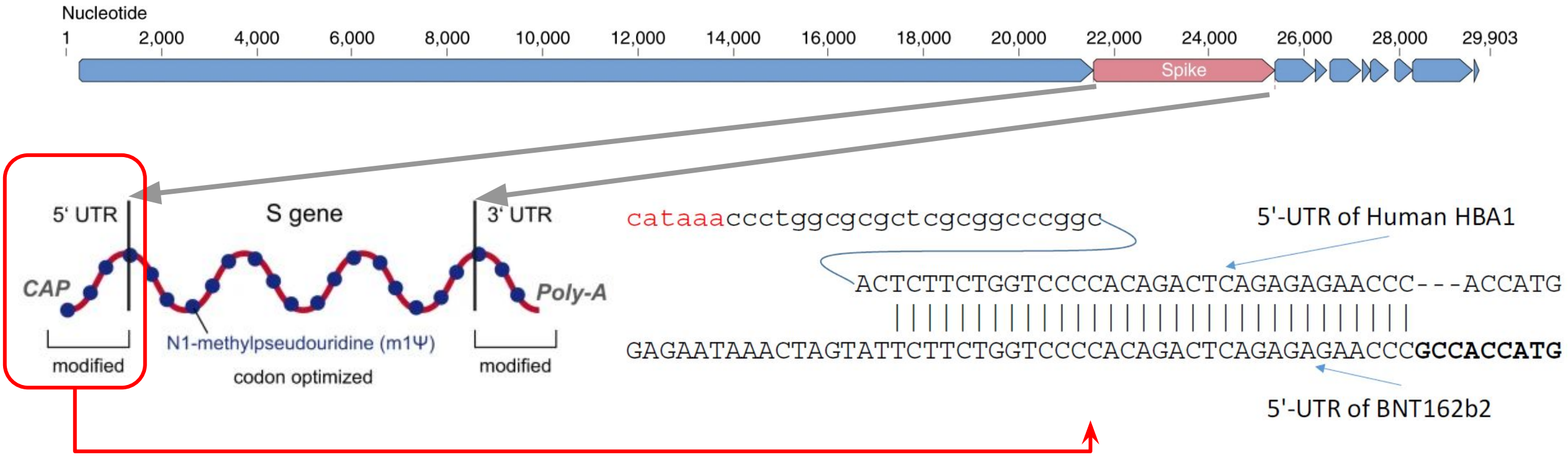
A typical case for Blast: we have a RNA sequence (see below). How can we know the original genome of the sequence, and ideally the gene encoding the sequences?

```
ATGTTTGTTTTTCTTGTTTTATTGCCACTAGTCT
CTAGTCAGTGTGTTAATCTTACAACCAGAACTCA
ATTACCCCCTGCATACACTAATTCTTTCACACGT
GGTGTTTATTACCCTGACAAAGTTTTCAGATCCT
CAGT
```

Tip: go to the NCBI BLAST tool (https://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastn&PAGE_TYPE=BlastSearch&LINK_LOC=blasthome), copy and paste the sequence as the query sequence, and try your luck. The default parameter would do.

The Wiki page of BLAST is a good start to understand how it works

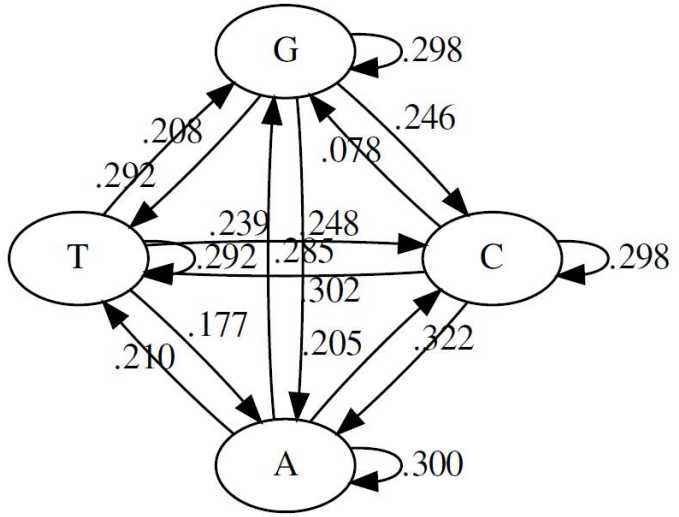
Further applications of biological sequence analysis in drug discovery: sequence-based drug design



References: Heinz, Franz X., and Karin Stiasny. "Distinguishing Features of Current COVID-19 Vaccines: Knowns and Unknowns of Antigen Presentation and Modes of Action." Npj Vaccines 6, no. 1 (August 16, 2021): 1–13. https://doi.org/10.1038/s41541-021-00369-6; Assemblies of putative SARS-CoV2-spike-encoding mRNA sequences for vaccines BNT-162b2 and mRNA-1273 (github.com/NAalytics); Xia, Xuhua. "Detailed Dissection and Critical Evaluation of the Pfizer/BioNTech and Moderna MRNA Vaccines." Vaccines 9, no. 7 (July 3, 2021): 734. https://doi.org/10.3390/vaccines9070734.

A probabilistic view of biological sequence analysis with Markov chains

- A **discrete-time** Markov chain is a sequence of random variables with the Markov property, namely that the probability of moving to the next state depends only on the present state and not on the previous states.
- A Markov chain is often represented by either **a directed graph** or **a transition matrix**.
- Two sides of application
  - Given a string, assuming that the Markov chain model is suitable, we can easily construct a Markov chain, for instance by counting transitions and normalize the count matrix (variants possible).
  - Given a Markov chain model and a string, we can calculate the probability that the string is generated by the specific Markov chain model with the **chain rule of conditional probability.**



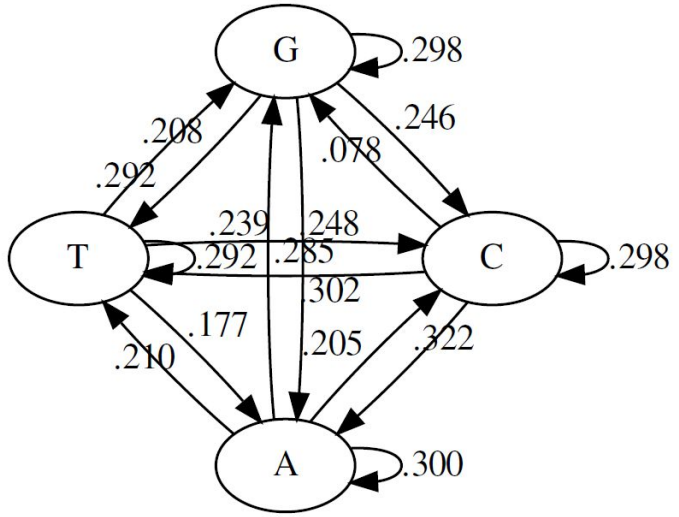
	A	C	G	T
A	.300	.205	.285	.210
C	.322	.298	.078	.302
G	.248	.246	.298	.208
T	.177	.239	.292	.292

Data: https://web.stanford.edu/class/stats366/exs/HMM1.html

A probabilistic view of biological sequence analysis with Markov chains

Given a Markov chain model and a string, we can calculate the probability that the string is generated by the specific Markov chain model with the **chain rule of conditional probability.**

Example: Compare *p(ACGTGGT*|M*)* and *p(ACCTGGT*|M*)*, where M stands for the model on the right side.

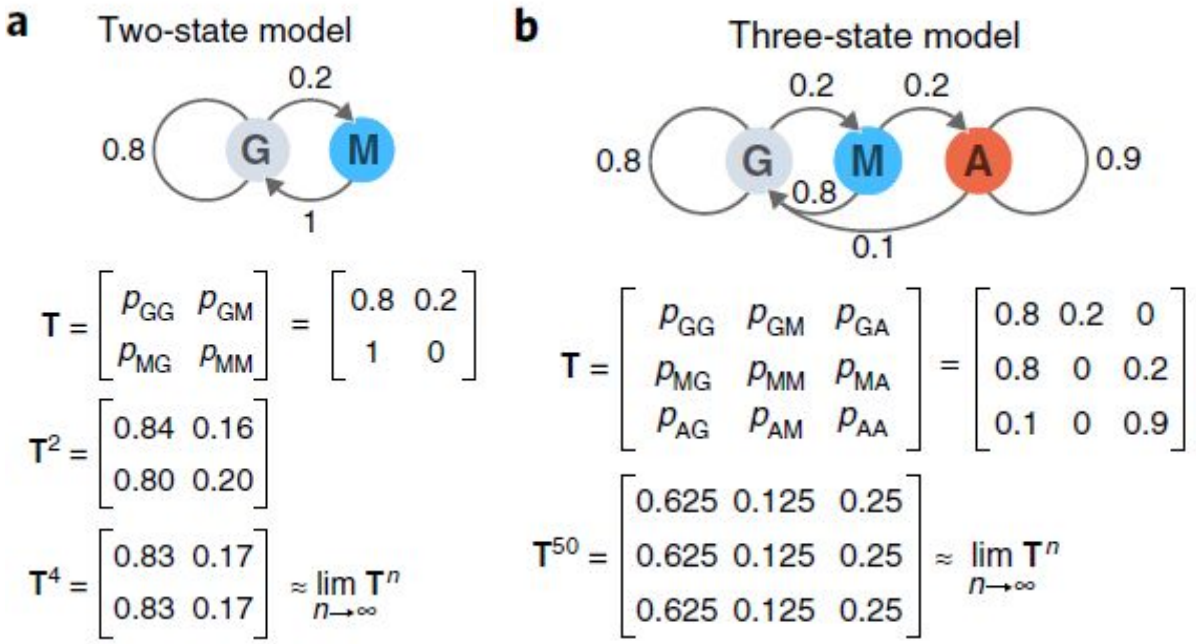


	A	C	G	T
A	.300	.205	.285	.210
C	.322	.298	.078	.302
G	.248	.246	.298	.208
T	.177	.239	.292	.292

Data: https://web.stanford.edu/class/stats366/exs/HMM1.html

Stationary distribution exist for ergodic (irreducible and aperiodic) Markov Chains

- A Markov Chain has stationary *n*-step transition probabilities, which are the *n*th power of the one-step transition probabilities. Namely, $P_n=P^n$.
- A stationary distribution π is a row vector whose entries are non-negative and sum to 1. It is unchanged by the operation matrix P on it, and is defined by πP=π.
  - In another word, it is the limit of the transition matrix multiplying itself.
  - Note that is has the form of the left eigenvector equation, $uA=\kappa u$, where κ is a scalar and u is a row vector. In fact, π is a normalised (sum to 1) multiple of a left eigenvector e of the transition matrix P with an eigenvalue of 1.
- Markov chains capture dependencies within a system and reveal interesting long-term behavior. They are subjects of the study of **stochastic processes.**

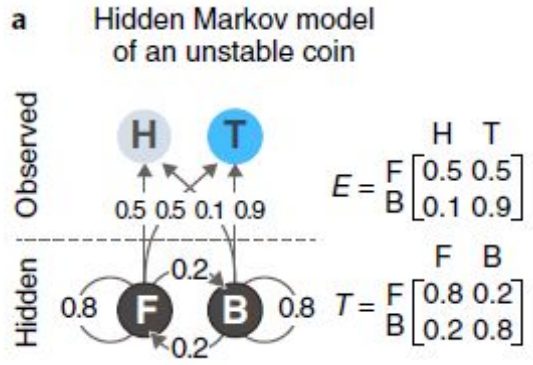


G=Growth, M=Mitosis, A=Arrest

Grewal, Jasleen K., Martin Krzywinski, and Naomi Altman. 2019. "Markov Models—Markov Chains." Nature Methods 16 (8): 663–64. https://doi.org/10.1038/s41592-019-0476-x.

Hidden Markov Chains

A Hidden Markov Model consists of two graphs (matrices): one of hidden states (corresponding to the transition matrix), and one of observed states (emitted by the hidden states according to the emission matrix). The Viterbi algorithm (based on dynamic programming), or the Baum-Welch algorithm (a special case of EM algorithms) is used to estimate its parameters.



A Hidden Markov Model of an unstable coin that has a 20% chance of switching between a fair state (F) and a biased state (B). Source: Grewal, Jasleen K., Martin Krzywinski, and Naomi Altman. 2019. "Markov Models — Hidden Markov Models." Nature Methods 16 (9): 795–96.

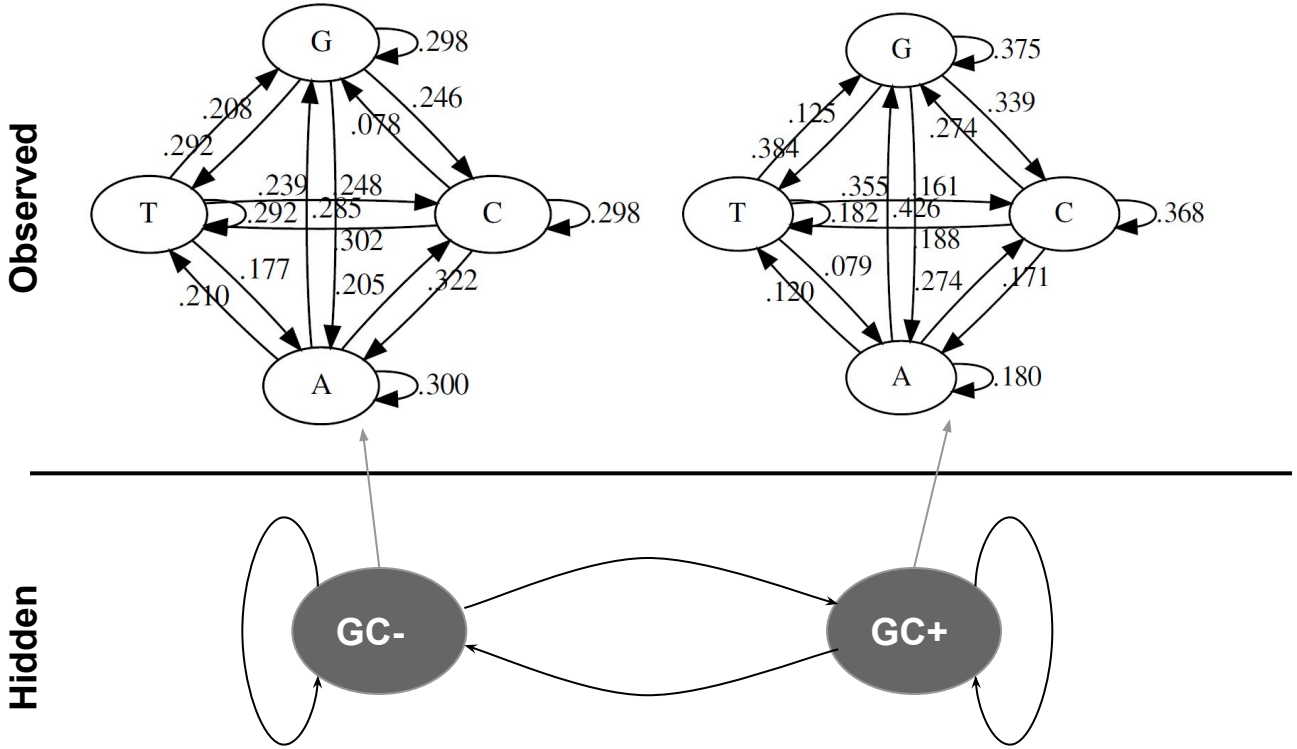


Illustration of a Hidden Markov Model predicting CpG islands in genomic sequences

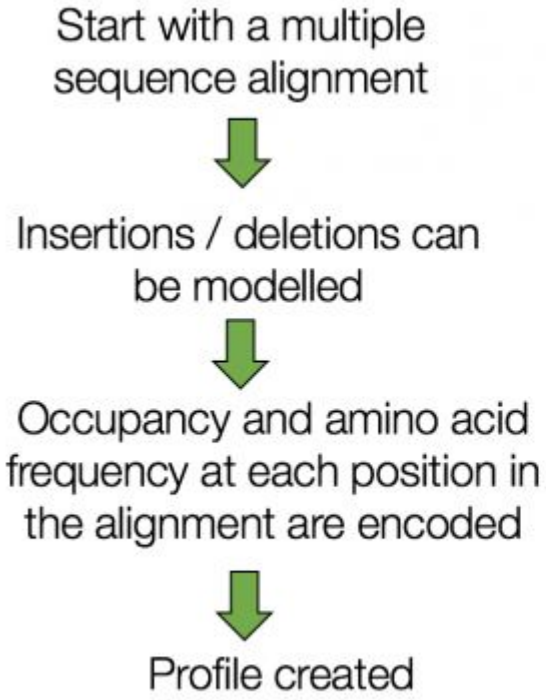
Profile Hidden Markov models capture evolutionary changes in homologs

M: match states. In the match state, the probability distribution is the frequency of the amino acids in that position.

I: insert states, which model highly variable regions in the alignment

D: delete states, which allows gaps and deletion.

Profile HMMs belongs to ***generative models***.



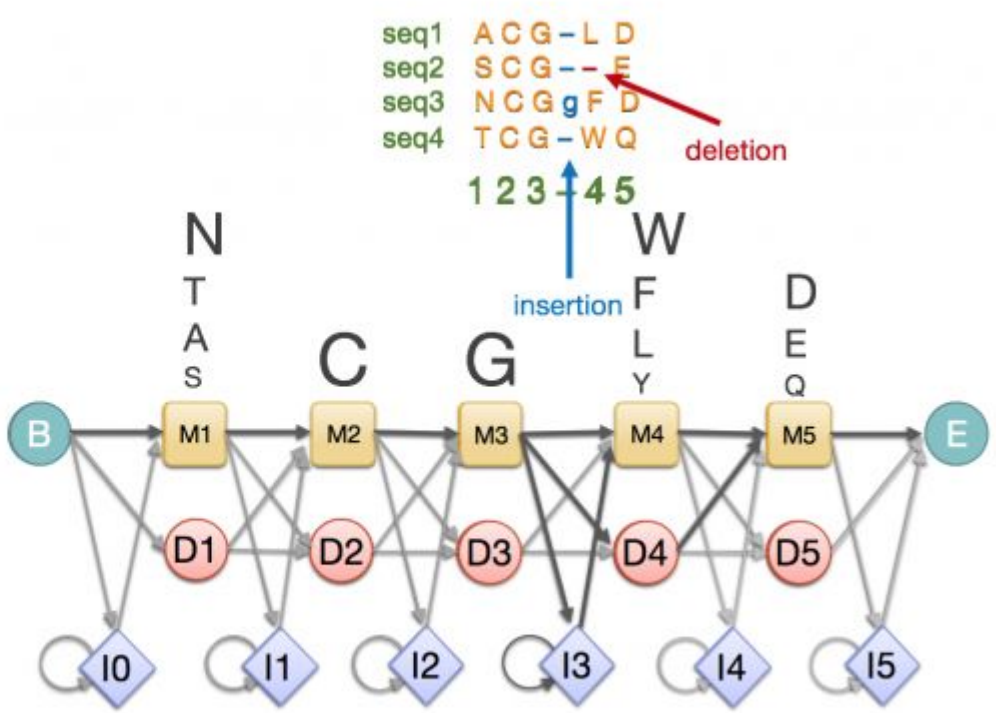
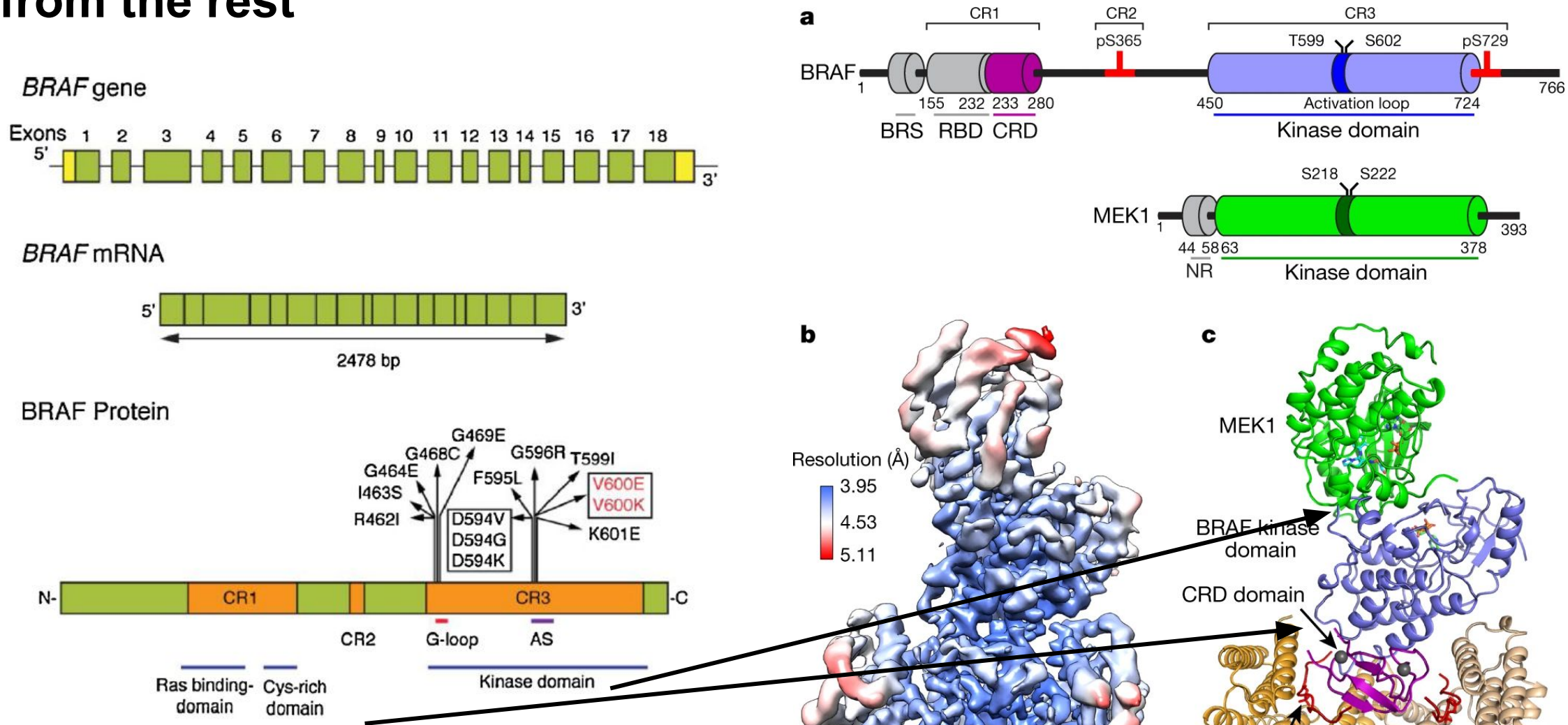


Figure from Pfam

Protein domains: self-stabilizing and folding independently from the rest

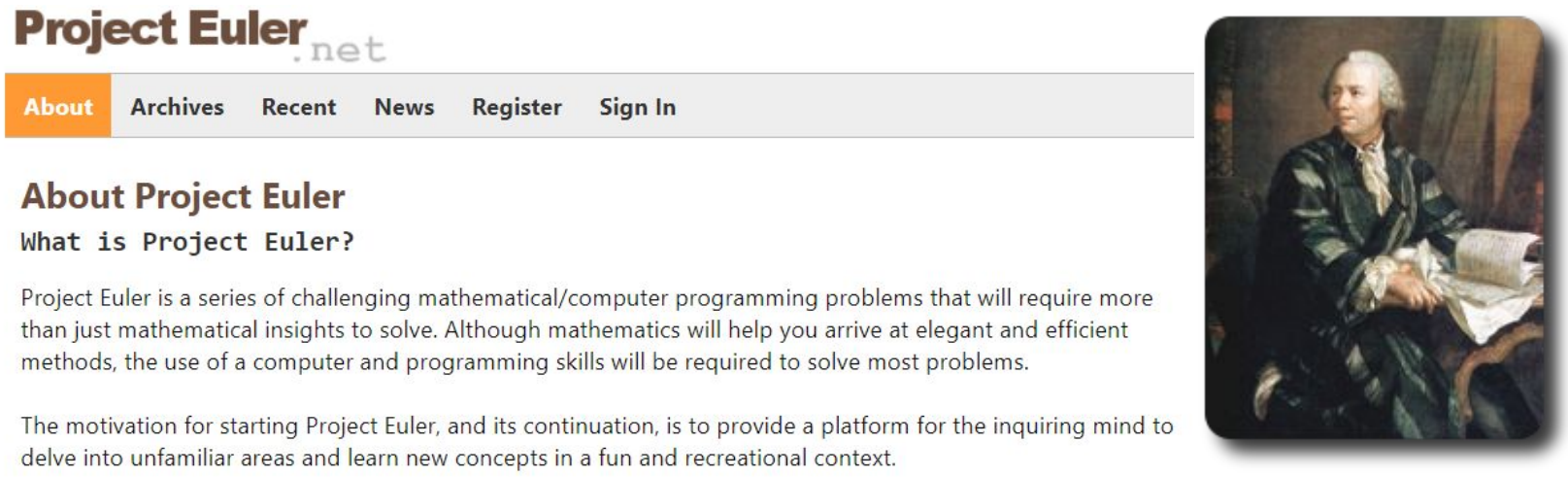


Left: Frisone, *et al.*, A BRAF New World, Critical Reviews in Oncology/Hematology (2020); Right: Park, *et al.*, Architecture of Autoinhibited and Active BRAF–MEK1–14-3-3 Complexes, Nature (2019)..

Software tools

- **General biological sequence analysis**
  - EMBOSS software suite: http://emboss.sourceforge.net/, also available online at European Bioinformatics Institute (EBI): https://www.ebi.ac.uk/services
  - BLAST (=Basic Local Alignment Search Tool) can be run at many places, for instances from EBI and National Center for Biotechnology Information (NCBI): https://blast.ncbi.nlm.nih.gov/Blast.cgi
  - Programming access, for instance the Biopython project: https://biopython.org
- **RNA biology**
  - ViennaRNA package (https://www.tbi.univie.ac.at/RNA/)
  - RNA processing tools available at U Bielefeld, for instance RNAhybrid, which finds minimum free energy hybridization using dynamic programming (https://bibiserv.cebitec.uni-bielefeld.de/rnahybrid)
- **Profile Hidden Markov Models (HMMs)**
  - The HMMER package: http://hmmer.org/

The Euler Project



https://projecteuler.net/

- **Learning by problem-solving**
- **Free**
- **Math + CS**

Problem 1: Multiples of 3 and 5

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.

Rosalind: a great scientist, and a platform for learning bioinformatics and programming through problem solving

ROSALIND http://rosalind.info/problems/locations/

Rosalind Elsie Franklin

1920-1958

A Rapid Introduction to Molecular Biology click to expand

Problem

A **string** is simply an ordered collection of symbols selected from some **alphabet** and formed into a word; the **length** of a string is the number of symbols that it contains.

An example of a length 21 **DNA string** (whose alphabet contains the symbols 'A', 'C', 'G', and 'T') is "ATGCTTCAGAAAGGTCTTACG."

Given: A DNA string s of length at most 1000 nt.

Return: Four integers (separated by spaces) counting the respective number of times that the symbols 'A', 'C', 'G', and 'T' occur in s.

Sample Dataset

```
AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC
```

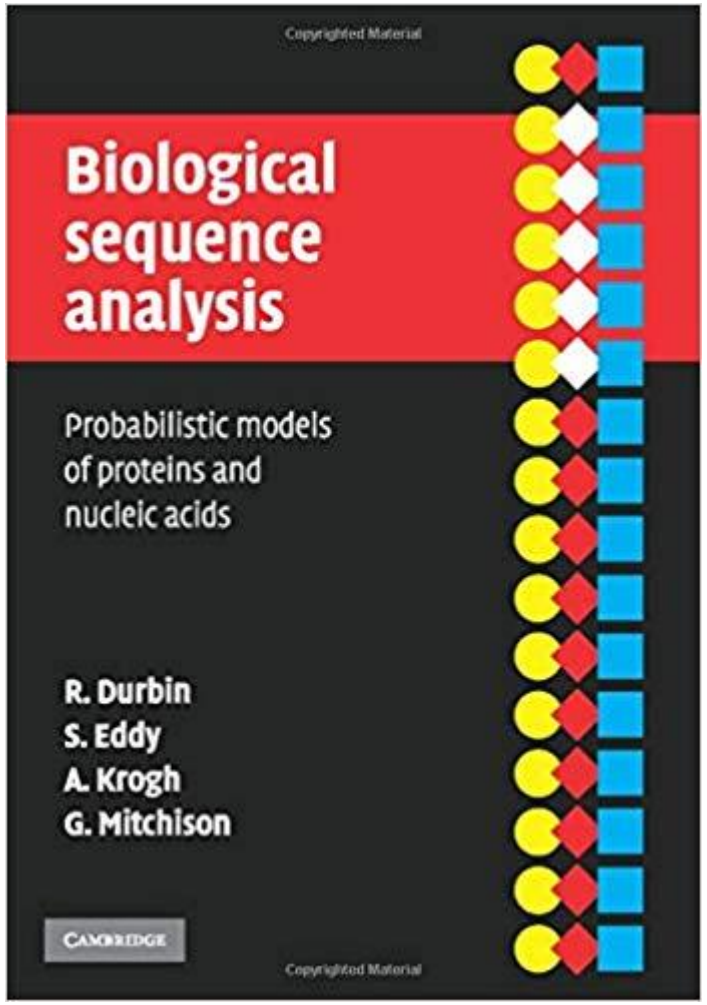
Sample Output

```
20 12 17 21
```

Please login to solve this problem.

Further resources

***Biological Sequence Analysis* by Durbin, Eddy, Krogh, and Mitchison**



Teaching RNA algorithms by the Backofen Lab at U Freiburg, with source codes available on GitHub.

The website hosts among others an interactive tool to visualize how dynamic programming (DP) helps to predict RNA secondary structure.

For a gentle introduction, see also *How Do RNA Folding Algorithms Work?* by Eddy, Sean R, *Nature Biotechnology* 22, Nr. 11 (November 2004): 1457–58. https://doi.org/10.1038/nbt1104-1457.

An Introduction to Applied Bioinformatics by Greg Caporaso (NAU)

The tutorial is written in Python using Jupyter. It introduces concepts in (a) pairwise sequence alignment, (b) sequence homology searching, (c) generalized dynamic programming for multiple sequence alignment, (d) phylogenetic reconstruction, (e) sequence mapping and clustering, as well as (f) machine learning in bioinformatics. Applications and exercises are also available.

Summary and Q&A

- Biological sequence analysis is used throughout the drug discovery process, and is essential for molecular modelling in the multiscale-modelling view of drug discovery.
- We explored both deterministic views of sequence analysis, with the example of Levenshtein distance, and probabilistic views, with the example of Markov chains and hidden Markov chains.
- Mathematical techniques such as dynamic programming, when implemented as algorithms software tools, are important for many tasks. We discussed in particular the Needleman-Wunsch algorithm, the Smith-Waterman algorithm, the BLAST software, and sequencing read alignment tools such as Bowtie2, STAR, and GSNAP.
- We provided further resources for further study and exploration.

Offline activities

Required reading:

- Tsai, et al. “Discovery of a Selective Inhibitor of Oncogenic B-Raf Kinase with Potent Antimelanoma Activity.” PNAS (2008): 3041–46. https://doi.org/10.1073/pnas.0711741105.

Optional reading:

- Dolgin, Elie. “The Tangled History of MRNA Vaccines.” Nature 597, no. 7876 (September 14, 2021): 318–24. https://doi.org/10.1038/d41586-021-02483-w.

Offline activities (for Lecture 3 and Lecture 4):

- Exercises about (1) the genetic code, (2) the Levenshtein distance and dynamic programming, (3) about the BLAST program;
- Questions about the required readings
- Submit your replies via Google Form: https://forms.gle/VzGGbXBvy9ZhTSVK8

Backup slides

Continuous-time Markov Chains

- Continuous-time Markov Chains are used for **phylogenetic analysis**, for instance of orthologous genes and of bacterial/viral genomes. They satisfy the Markovian property: $P(t+\tau)=P(t)P(\tau)$.
- The process makes a transition from the current state *i* after an amount of time modelled by an *exponential random variable* E_i, known as the **holding time**. Random variables of each state is independent.
- When a transition is made, the process moves according to the **jump chain**, a discrete-time Markov chain with a transition matrix.
- If there are *n* states, then at the time of transition, there are *n-1* competing exponentials. Since the distribution of the minimum of exponential random variables is also exponential, the continuous-time Markov chain changes its state from *i* according to a parameter $E_{i,j} \sim \text{Exp}(q_{i,j})$ ($i \neq j$). The parameters are known as the Q-matrix, or the **rate matrix**. The transition rate *E* is the product of holding time and the transition probability.
- Whereas the row sums of a transition matrix are always 1, the row sums of a rate matrix are always zero.

Given the transition matrix

$$P(t) = \begin{pmatrix} p_{AA}(t) & p_{AG}(t) & p_{AC}(t) & p_{AT}(t) \\ p_{GA}(t) & p_{GG}(t) & p_{GC}(t) & p_{GT}(t) \\ p_{CA}(t) & p_{CG}(t) & p_{CC}(t) & p_{CT}(t) \\ p_{TA}(t) & p_{TG}(t) & p_{TC}(t) & p_{TT}(t) \end{pmatrix}$$

We model the probability of seeing the same alphabet after a small increment of time as the sum of the starting probability, minus its loss, and plus its gain

$$\mu_x = \sum_{y \neq x} \mu_{xy}$$

$$p_A(t + \Delta t) = p_A(t) - p_A(t)\mu_A \Delta t + \sum_{x \neq A} p_x(t)\mu_{xA}\Delta t\,.$$

$$\mathbf{p}(t + \Delta t) = \mathbf{p}(t) + \mathbf{p}(t)Q\Delta t\,,$$

where

$$Q = \begin{pmatrix} -\mu_A & \mu_{AG} & \mu_{AC} & \mu_{AT} \\ \mu_{GA} & -\mu_G & \mu_{GC} & \mu_{GT} \\ \mu_{CA} & \mu_{CG} & -\mu_C & \mu_{CT} \\ \mu_{TA} & \mu_{TG} & \mu_{TC} & -\mu_T \end{pmatrix}$$

Source: https://en.wikipedia.org/wiki/Models_of_DNA_evolution

Interaction of drug and target: an example with HIV-1 Protease Inhibitor

Protein atoms: ball and stick, in blue and green

The small-molecule drug: ball and stick with traditional atomic coloration

Water: small red spheres