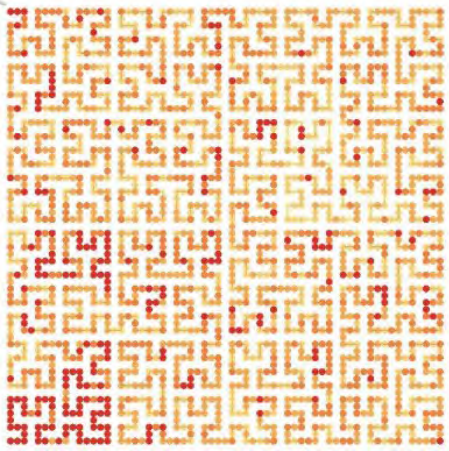
БІОІНФОРМАТИКА: АНАЛІЗ ГЕНЕТИЧНИХ ПОСЛІДОВНОСТЕЙ

Богдан ОСТАШ



Львівський національний університет
імені Івана Франка

Міністерство освіти і науки України

Львівський національний університет
імені Івана Франка

Богдан ОСТАШ

БІОІНФОРМАТИКА: АНАЛІЗ ГЕНЕТИЧНИХ ПОСЛІДОВНОСТЕЙ

Електронний підручник

Львів

2022

УДК [575.112:004](075.8)

О 76

Рецензенти:

Сиволоб А. В., проф., д-р біол. наук, кафедра загальної та медичної генетики ННЦ "Інститут біології та медицини" Київського національного університету імені Тараса Шевченка;

Дмитрук К. В., ст. наук. співроб., д-р біол. наук, заступник директора з наукової роботи Інституту біології клітини НАН України;

Волков Р. А., проф., д-р біол. наук, завідувач кафедри молекулярної генетики та біотехнології Інституту біології, хімії та біоресурсів Чернівецького національного університету імені Юрія Федьковича.

Рекомендовано до друку
Вченою Радою Львівського національного університету імені Івана Франка
(протокол № 33/6 від 29 червня 2022 р.)

Осташ Богдан

О 76 Біоінформатика: аналіз генетичних послідовностей : електронний підручник / Богдан Осташ. – Львів : ЛНУ ім. Івана Франка, 2022. – 232 с.

ISBN 978-617-10-0729-1

Електронний підручник ознайомлює з практичними знаряддями комп'ютерного аналізу нуклеотидних та амінокислотних послідовностей, а також подає стислий опис математичних та статистичних концепцій, що є теоретичним підґрунтям цих знарядь. Розглянуто сучасні бази біомедичних даних та основні математичні моделі, що застосовують в аналізі генетичних послідовностей. У наступних трьох розділах викладено методи біоінформатики: попарні й множинні вирівнювання, моделі на основі вирівнювань (зокрема, позиційно специфічні вагові матриці й приховані моделі Маркова).

Для студентів та науковців природничого та медико-біологічного профілів.

УДК [575.112:004](075.8)

ISBN 978-617-10-0729-1

ЗМІСТ

Передмова

Мабуть, не буде перебільшенням сказати, що біологія входить в той етап, де обчислювальні методи стають новим мікроскопом (10.1371/journal.pbio.0020439). На вістрі цих змін знаходиться біоінформатика – дисципліна, що сформувалася в 60-х роках XX ст. як набір обчислювальних методів, покликаних автоматизувати зберігання, аналіз і порівняння нуклеотидних послідовностей генів та кодованих ними білків. Сам термін з'явився дещо пізніше, на початку 70-х років, у працях голландської науковиці Полін Гогевег (Paulien Hogeweg). Примітно, що вона визначала біоінформатику ширше, як науку про інформаційні процеси в біотичних системах (https://www.ncbi.nlm.nih.gov/pmc/articles/ PMC3068925/). Лише зараз, на початку третього десятиліття XXI ст., ми бачимо, як лавина біологічних даних різного типу наповнює біоінформатику тим змістом, який у неї початково вкладали її засновники. Втім, основні концепції і підходи біоінформатики створені в епоху, коли не було Інтернету, і все базувалося саме на аналізі генетичних послідовностей – тобто нуклеотидних послідовностей генів та амінокислотних послідовностей білків, кодованих генами. Цілком приймаючи критику про певну ненауковість терміна "генетична послідовність", я все ж зважився його залишити з огляду на символізм (ми зосереджуємося на полімерах/послідовностях, що мають спадкову, або генетичну, природу) і ощадність (порівняйте з висловом "нуклеотидні й амінокислотні послідовності").

Підручник "Біоінформатика: аналіз генетичних послідовностей" підсумовує десятирічний досвід його автора у викладанні загального курсу біоінформатики студентам-біологам ЛНУ імені І. Франка. Виклад матеріалу відбувається за схемою, що прийнята в найпопулярніших закордонних підручниках з біоінформатики, зокрема за редакцією Баксеваніса та Аулетт (ISBN 0-471-38390-2) й Гіґґса та Еттвуд (ISBN 1–4051–0683–2). Потреба у виданні україномовного підручника з біоінформатики зумовлена кількома чинниками. По-перше, він необхідний як майданчик для обговорення і поширення україномовної термінології в галузі біоінфоматики. По-друге, наявні закордонні підручники мають певні недоліки. Підручникові Баксеваніса притаманний більш поверховий виклад, що має прикладний характер, і тому навіть останнє видання неминуче застаріле в описі наявних знарядь чи веб-застосунків. Підручник Гіґґса робить наголос на описі і навіть повному математичному виведенні рівнянь, і такий рівень може бути заскладним для рівня бакалавра/магістра-біолога. У нашому підручнику намагаємося дати докладний опис концептуальних засад біоінформатичних підходів без занурення в подробиці математичного доведення. Для тих, кого цікавить останнє, створені т.зв. "блоки" – односторінкові описи з глибшим викладом певної концепції – читання яких необов'язкове для розуміння основного тексту. Сподіваємося, це допоможе подолати проблему певної ізоляції біології від математичних дисциплін, принаймні у реаліях України. По-третє, існує потреба у донесенні проблем і питань сучасної біології до фахівців інших галузей. Для багатьох студентів з математично орієнтованих спеціальностей біологія виглядає не надто цікавою, бо у вітчизняній біологічній освіті не окреслено сучасні проблеми, до вирішення яких вони могли б долучитися зі своїм багажем знань. Плекаємо щиру надію, що цей підручник сприятиме подоланню таких штучних перешкод і стане поштовхом для фізиків, математиків та ІТ-фахівців поринути у світ обчислювальної біології!

Наш підручник забрав багато годин, які можна було б використати інакше – можливо, краще, з більшою користю. Гіршим від цього є усвідомлення, що, мабуть, ми ніколи не дізнаємося, чи він приніс бодай якусь користь нашим читачам, якщо такі будуть. Ця думка переслідує мене досить давно, і тому я намагався писати цей текст так добре, як тільки можу. Далі я бачу дві розради. Перша полягає в тому, що точно є одна людина, яка таки чогось навчилася – це сам автор. Усвідомлення тонкощів та елегантності підходів біоінформатики варті принаймні дещиці того часу, які я витратив на їхнє вивчення і опис. Це було цікаво, приємно, повчально і корисно; я зичу решті читачів пережити такі відчуття. Друга розрада полягає в тому, що тягар сумнівів про доцільність такої роботи я несу не сам. Цей підручник, у тому вигляді, який він зараз є перед читачем, став можливим завдяки внеску і підтримці дуже багатьох людей. За усі роки викладання курсу я отримав від своїх студентів безліч правок, коментарів і запитань, за що їм дуже вдячний. Викладання дало змогу зрозуміти, що саме потребує глибшого викладу, додаткового ілюстрування чи окремого переліку задач для самостійного опрацювання. Студенти направду стали моїми анонімними співавторами. Свою подяку адресую також рецензентам і колегам по роботі, слушні зауваження яких змусили мене переосмислити виклад матеріалу деяких розділів і докладніше описати певні методи. Весь текст став граматично правильним, легким і доступним для розуміння лише після внесення усіх виправлень, зроблених редакторами Видавництва ЛНУ імені Івана Франка п. Людмилою Макітринською та п. Іриною Лоїк. Незважаючи на сказане, одразу зауважу, що всі недоліки і помилки цього підручника – виключно моя відповідальність. Будь ласка, повідомляйте про них на електронну пошту: bohdan.ostash@lnu.edu.ua. Насамкінець складаю найглибшу подяку своїм рідним, передусім дружині. Іра, як доцентка кафедри генетики та біотехнології, що веде практичні з біоінформатики, виявила не один десяток недоліків моїх рисунків і пояснень; як найближча мені людина, вона змогла заплющити очі тоді, коли я “трохи” (mea culpa!) засиджувався над цим текстом. Вдячний своїм батькам, які навчили, що всяку розпочату справу – чи то перерваний дощем осінній медозбір, чи незавершений текст – треба доводити до кінця. Дякую своїм ангеликам – донькам Анні й Христині – за допомогу в ілюструванні книги, несподівані розмови і жарти, які пожвавлювали монотонну роботу над цим підручником, й оптимістично утвердили у вірі в те, що комп’ютеру –навіть якщо це суперкомп’ютер – є ще багато чому повчитися в людини ☺.

Львів, липень 2022 Богдан Осташ

Вступ

Станом на липень 2022 року в базі даних Genome (http://www.ncbi.nlm.nih.gov/genome/ – найбільший відкритий депозитарій нуклеотидних послідовностей) містилась інформація про первинну послідовність геномів більше 500 тисяч організмів, з яких 430 000 – бактерії, 24 501 – евкаріоти (одноклітинні, гриби, водорості, тварини, рослини тощо), решта – віруси, плазміди й органели. У базі зберігається інформація про більш ніж 16 трильйонів *пар нуклеотидів* (п.н.). Крім цього, у Genome містяться нуклеотидні послідовності фрагментів геномів, метагеномні дані й архіви частково секвенованих геномів. Це колосальна кількість загальнодоступної інформації про генетичну основу життя, і її кількість невпинно зростає. Гени і геноми є лише лінійною послідовністю нуклеотидних залишків, і ця послідовність не має змісту доти, доки дослідник не визначить її функціональне навантаження – здатність кодувати білок, бути промотором, інтроном тощо (рис. 1). Систематичний аналіз усього масиву даних експериментальними методами – генні нокаути, функціональна геноміка, протеоміка тощо – наразі (і, мабуть, в осяжному майбутньому також) нереалістичний як економічно, так і технологічно, що, зрештою, і не є метою більшості проєктів геномного секвенування. Розширення таких баз даних стимулювало розвиток методів комп'ютерного аналізу нуклеотидних послідовностей, продуктів їхньої транскрипції і трансляції (РНК і білки). Далі, поряд зі словосполученням *нуклеотидні й амінокислотні послідовності* (НАП), задля лаконічності вживатимемо вираз *генетичні послідовності*. Обидва терміни мають однаковий зміст, оскільки описують послідовності генетичного матеріалу (генів, геномів), а також продукти його експресії (РНК, амінокислотні послідовності). Ці методи дали змогу упорядкувати дані у межах різних спеціалізованих веб-сервісів і класифікувати їх відповідно до різних критеріїв і потреб дослідників. Однак найважливішим результатом синтезу молекулярної біології, геноміки, комп'ютерних технологій, теорій інформації та ймовірності – *біоінформатики* – стало те, що ця нова галузь перетворилася у самостійне знаряддя наукового відкриття. Використовуючи біоінформатичні методи, виявляють у масивах генетичних послідовностей нові закономірності структурного і функціонального організування геномів і протеомів, їхньої еволюції – ті закономірності, які залишаються за межами досяжності експериментальної біології. Наприклад, варто згадати виявлення третього домену життя – архей – на основі аналізу послідовностей рРНК; відкриття системи набутого імунітету в бактерій (CRISPR-Cas); відпрацювання удосконалених моделей частот вживання кодонів, що покращило методи оптимізації генів для експресії промислових білків; впровадження новітніх підходів до діагностування генних захворювань на основі геномного сканування і повного секвенування; коректні моделі тривимірної структури білків; створення нових, реалістичних популяційно-генетичних моделей, що є в основі світових природоохоронних та протиепідемічних програм. Тому не буде перебільшенням стверджувати, що повноцінна освіта біолога, передусім фахівця-генетика, сьогодні неможлива без бодай поверхового ознайомлення з біоінформатичними методами пошуку й аналізу генетичних послідовностей. Саме з цією метою написано підручник.

TCTGTTTCTGTGTTGAATTTGTGTAGCAACAGAGCAAGTATTTCTCCTGAAACTCAGACTCACTCAGGCC
ACCTCAGCCCTTCTAGTTCCTTTCGTTTATTATAGAGTAACCTACAGCAGGAATACCGCGTTCATTTACT
GTTAATAGAAGATCCACAAAAGATTTTGAAATCAAATACATTCTAAAACTAGGCAAGACACGCTGCAACT
CCCCTTGTCCTCTGGGTTATGAGGGGTGAGCGTTGGTTTCCGTGTGCTCAGATCCCATGAATGAGACCTC
ACAGGGAAGCGCCCAGTGACTGGCAGGCTCCTTAATGCCTCCAGGTCAGGCTCAGGAAGTCTCTGGATGC
CTGGGTAGAGTCTGGGTCTTCTCAGAGCGGGAGGCACCCAGGGCCTGTGGAGGCCTGAGGGCTGCTCTGG
GGTTCCAGGGCCCCAGATAGTGGCTTAGCATCCTGGTGACACAGAAGGGAGAGCACTCACCGCGGTCCCA
CGTGTGTGTTTTGAAGCCTAAGATGTTTGCAATGACTCTCAACAGTCGCTTTGACCTTTTCCCCTAGATC
TTTAGCAGTTAAGGAGTTTTTTTCAAAGCCCAAGTATAACCACATTTTGAAACCAGAAGACTGTCTCTCT
GAACCGTGCACTATATTGCAGTTGGACATGAGAACCGTGCAAATTTCTGATCTAGAGGTGAGAAAAAGAT
GAATTGCTCCTTACATTCGATAATCAGTGACCACGAAACACTCAGACCAGAGCCTGGCTTATCAAAAACC
TTCAGTGAGTGCTGGGGGTGTGAGTGAATAACTAATTATTTTATTATGCAAATAAGTGAATTTATAAAAC
GTTTGCTACTGATTTTTTCCAGTCTTTTTTCTTTTTTACGTTCTATTTTGATTCTTTCATATTGTACACC
ATTTTATGTCTCCAGCGTCTTCATTTTAGATTTATGTTTAATATTCTCAGCATCTTCAAAATCAAATAAA
TTATATTTCGTTTCATTATGGTATTTGCATTACAGTTTTTTTCAAGTATTTTAAGCCACTCTTCATGACA
GAATCGTTAGAATTCCCTCCGTGAATCCCTGTCTGTGGCCGCGCGGTGGCGCTCGTGGGCTCTGGGCGGT
TTCCCCGCGCCCCCGCCCTGCCTGGCGCCCCAGCTGGCGCCGCGTCTGGAGATGGGGCGGGCGCGGTGGC
GCCGGCGCGCTTTGGTTTTTAGTTCGTTCTGCTGTGGAAAGGCCCAGCCTGGAGCTCTCCAACTCCTTAA
CTGGAAGAGCTGAGCGCCGAAAAGATTCCGCAGAATCTGGGGTAGAAAATTGGGCGCAAGAAGCTCAGAT
GCTTGAAACGTTTCTGACTGAAGCCTCTGAACTCACGATGCGGGTTTTGGGGGTGGTTTCCCCAGGCTGG
GGGGCTTTTCTACGGTTCCTGACGCATTCAGCGTCTGCGCGGGGCGCGTGGAGGGGGCCGCCCCAACCCG
GGCCCTGCGGTGCCACGCGGTGCCCACGCGTGCCTTGTCATCTGCTTGACCCAGACCCTGAGGGGCGAGC
TGCGCTTCGACATCAGGAAGGCGGGGACCCTGCACGGCTTCACGGCCTGGTTTAGCGTCCACTTCCAGAG
CCTGCAGGAGGGGCAGCCGCCGCAGTGCTCAGCACCGGGCCCTTCCACCCGTGAGTGTGCGGGGCGCGGG
CACGGGGTGCGGGGTGGGGGGCAGGGGAGTAGGCGGGGTGCGGGGATGGGGCGCGCAGGAGGGGGTGGGA
CGCGGCGTGGCGGTCGTGGGGGGGTGGGGCACGTGGGGGCGCGGGCAGTGGGGTGGGGGCGGCGCGCATG
GGGGCACGGGTGGTGTAGACACCGCAGCCTGTGTGCCCCCTTCCGGGTCTCGCCACCACTCTGGGCCCAG
CCTGGGCGTCCCTCCGACAACAGCCCCAACGGGAGGAGCTCCCTGTCTTCCACCCCGCGAGGAGCACCAC
AGCTCTTCATGGTCTGACCCAAAGCGCCTCCACGGGGCGTCGGGGTTCACCTGACGTTTTTGGAGGTGCC
TGTTTTTTGGGACACATTTCCCCGCCAGCACCCTGGGCATGTGTGGCAGTGGAGCTGCCTTCAGGGCCCT
GCAGGCCTTCCCTTTGGGTGGGGTTGGCTACCCCGCTCTGGCACCTGCCCCGCTGACTTCCCAGCGTGGT
GGCCCCGCGAGGGGACCCCGGCTCCAGGGAGGTCGCAGAGGTGCGGGGTTGATGGCCTGGCCTGGCATGT
GGTCCCCATCACAAGAGTCCTGTGGTGGCCCCTCCAGTGCCATCACCGGGTGGAGGTGGGCAGCTTGGAG
GCCAGTTATACCCATCCCCGAGTCTGGGGGCGTGGAGGGGGCAGTGACCTGGGGCACAGGCTGGGGTGAG
GGGTGTGACCAGAGTCAGGGAAAAACTGGAGACCTGAGGAGCCCCCCACTTCACTGCTGGCCCCACCGTG
GACAGTGGGAGTTTGAGGGAAGACAGAGCCTGAGGAACTTAGAGATGTTTCCAACGCAACCCAGTGCAGA
ATCATAGCGTTTTACTATTTCCCCCCAAATGTGACAGTTGTCCTCACCTTGTAACCTTCACAGAAGCGAC
TCCTTGGACCTGATTCTGTCTGTGTACACCACAGTTTAACAGCAGGAGCTAGATCACCAGCCTCTGCAGC

Рис. 1. Фрагмент 21-ої хромосоми *Homo sapiens*

Проілюстрована вище на рисунку послідовність (2 660 п.н.) становить приблизно 0,0001 % довжини гаплоїдного генома людини, і вона є частиною кодувальної послідовності фермента аргінін-N-метилтрансферази 2. Визначення точки ініціації транскрипції цього гена і трансляції її мРНК за допомогою візуального аналізу розглянутої послідовності – неможливе, оскільки ДНК не має (ані візуально на папері, ані цитологічно) недвозначних ознак (маркерів), які б без додаткових перевірок свідчили про початок гена, чи інші функціональні ділянки. Передбачати різноманітні ознаки генів і геномів можна, якщо застосовувати біоінформатичні методи.

Біоінформатику можна визначити як набір обчислювальних (комп'ютерних) підходів до аналізу біологічних даних. Більш формальне визначення біоінформатики має двошарову структуру. Це, зокрема, розроблення обчислювальних методів для вивчення структури, функції й еволюції генетичних послідовностей і цілих геномів, а також розроблення методів організування, оброблення і зберігання біологічної інформації, що накопичується внаслідок геномних й інших високопропускних досліджень (транскриптоміки, протеоміки

тощо). Охопити все різноманіття концепцій, підходів і програм у межах однієї книги неможливо, враховуючи те, що практично всі напрями експериментальної біології мають спеціалізовані "продовження" в біоінформатичному вимірі. Наш підручник дає відповідь на такі загальні питання, які постають перед дослідником, що вирішив дослідити певну проблему з генетичного боку: де шукати інформацію про послідовність білка/гена(-ів), що цікавить; яку інформацію можна отримати про досліджувану генетичну послідовність, використовуючи лише загальнодоступні ресурси Інтернету. У тексті подано адреси спеціальних веб-сервісів, які на час написання цієї книги перевірено автором особисто. Однак потрібно усвідомлювати певний ступінь плинності веб-сервісів – деякі з них не змінюються протягом десятиліть, інші швидко зникають або ж видозмінюються. Тому доцільно використати пошукові інтернет-програми, наприклад Google, для виявлення їхніх аналогів. Добрим провідником в океані біоінформатичних ресурсів є журнал Nucleic Acids Research, (http://nar.oxfordjournals.org), який на початку кожного року друкує спеціалізований випуск, присвячений перелікові усього нового (чи суттєво оновленого) у світі біоінформатики.

Щодо структури. Підручник складається з п'яти основних розділів. У першому описані основні бази даних, де зберігається інформація про НАП. Другий ознайомлює з концепцією моделювання щодо НАП, із наголосом на імовірнісні моделі. Третій і четвертий розділи присвячені методам попарного і множинного порівняння генетичних послідовностей. П'ятий розділ – це розгляд моделей, які ґрунтуються на множинних вирівнюваннях НАП. Поряд із основним текстом наявні односторінкові "блоки" – змістовно автономні розділи, присвячені поглибленому розгляду обраних концепцій, на які спирається основний текст. Ознайомлення з "блоками" рекомендоване (але не обов'язкове) для його глибшого розуміння. Наприкінці більшості розділів також є задачі, вирішення яких має продемонструвати практичне застосування розглянутих понять і концепцій. Також акцентуємо увагу на тих підходах, які мають передусім стати в пригоді генетикові та генетичному інженерові. Обрання методів і прикладів їхнього застосування є результатом досліджень автора книги, і його можна розцінювати дещо суб'єктивним і зосередженим на бактеріях. Однак сучасні експериментальні дослідження у галузі генетичної інженерії не обходяться без бактерій – вони є об'єктом або знаряддям інженерії, або ж їх використовують як проміжну ланку, як господарів для створення певних рекомбінантних молекул ДНК, підтримання геномних бібліотек, експресії білків тощо. Багато методологічних підходів біоінформатичного аналізу – універсальні, як універсальний генетичний код життя. Висловлюємо сподівання, що цей розділ слугуватиме стартовим майданчиком для дослідників, які у своїй роботі послуговуються аналізом НАП. Практично вся біоінформатична термінологія – англомовна у вихідному стані; тому при першому вживанні спеціалізованого терміна українською мовою в дужках наведено оригінальний англомовний варіант, щоб читач згодом міг вільно розуміти його й оперувати ним у контексті англомовної наукової літератури. Завершує видання покажчик, у якому зібрані всі терміни, вжиті на сторінках підручника. Для повноцінного розуміння книги читачеві необхідні базові знання у галузі алгебри (логарифми, ступені), теорії імовірності, статистики та молекулярної біології (хімічна природа генів і білків, механізми реплікації і трансляції).

РОЗДІЛ 1

Бази даних генетичних послідовностей – модель Національного центру біотехнологічної інформації США

Сьогодні дослідницькі групи у всьому світі розшифровують нуклеотидні послідовності генів, геномів і груп геномів і щодня подають їх у загальнодоступні цифрові бази даних. Інформація про всі депоновані НАП зберігається у певних форматах і в незмінному стані. Що рухає окремими дослідниками у їхньому прагненні розміщувати генетичні послідовності на доступних через Інтернет ресурсах, а не зберігати у роздрукованому вигляді чи, наприклад, на власних серверах чи комп'ютерах? Очевидно, що спеціалізовані *бази даних* (databases) є надійнішим і доступнішим способом зберігання. Так, перші нуклеотидні послідовності в електронні бази даних занесені 1992 р., цю інформацію зберігають досі, вона доступна всім зацікавленим, незважаючи на зміни способів і форматів зберігання електронної інформації. Крім "увіковічнення" даних (що само по собі є важливим у науковій діяльності), дії дослідника зумовлені низкою практичніших чинників. По-перше, тільки після подання генетичної послідовності у базу даних вона по-справжньому доступна науковій спільноті – адреса персонального веб-сервера може бути менш відомою, менш доступною, ніж адреса великого електронного архіву. По-друге, практично всі наукові журнали зараз вимагають навести номери доступу до нових генетичних послідовностей, що є предметом публікації. По-третє, найбільші державні грантодавчі фонди Європи і США фінансують проєкти секвенування геномів за умови розміщення розшифрованих послідовностей у загальнодоступних базах даних. Тому необмежений доступ до результатів секвенування генів і геномів сьогодні розглядають як основну вимогу і гарантію того, що ці результати будуть найкориснішими для людства. Єдиний спосіб цього досягти – внести інформацію про встановлену НАП у певну базу даних.

Науковий світ послуговується великою кількістю баз даних, в яких зібрані НАП чи інформація про них. Усіх їх можна класифікувати на декілька груп. Перша і найважливіша – це первинні, або *архівні* бази даних. Саме ці бази дають змогу будь-якому дослідникові помістити нову генетичну послідовність (вперше просеквенований ген чи геном), визначену ним у ході роботи, у загальнодоступний цифровий архів. Інформацію не перевіряють, тобто її зміст повністю визначений науковцем, котрий її подав. До таких баз даних належать бази GenBank, ENA (European Nucleotide Archive), DDBJ (DNA Data Bank of Japan), PDB (Protein Data Bank) тощо. Вторинні, або *куровані* (curated), бази даних створюють на основі первинних. Зміст цих баз визначається експертами (кураторами й анотаторами), які добирають інформацію з первинних баз даних за певним принципом (наприклад, шукають усе, що стосується генома кишкової палички), і перевіряють точність цієї інформації, аналізуючи наукову літературу тощо. Зокрема, до таких баз даних належать: SWISS-PROT – одна із найякісніших баз даних про амінокислотні послідовності; FlyBase – база даних про плодову мушку *Drosophila*; COG – база даних про кластери ортологічних генів; StrepDB –

база даних генома модельного об'єкта генетики актиноміцетів – *Streptomyces coelicolor*, а також низки інших стрептоміцетів.

Ще один тип – *інтегровані* бази даних, де міститься як первинна, так і курована інформація. Приклад такої бази даних – EcoCyc. Це енциклопедія бактерії *Escherichia coli*, де, увівши у пошукове вікно назву певного гена, можна знайти усю пов'язану з ним інформацію.

Основним предметом цього підрозділу стануть первинні бази даних, які ми розглянемо на прикладі бази GenBank. Ця база є частиною міжнародного консорціуму INSDC (International Nucleotide Sequence Databases Collaboration; http://www.insdc.org/), куди входять вже згадані попередньо ENA і DDBJ. Ці три первинні бази даних приймають інформацію про НАП від окремих дослідницьких груп і великих центрів секвенування та щодня обмінюються нею між собою. Так досягають узгодження у репрезентації наявної публічної інформації про НАП у трьох найбільших первинних базах даних.

GenBank є частиною Національного центру біотехнологічної інформації США (NCBI; http://www.ncbi.nlm.nih.gov/). NCBI (рис. 1.1) зберігає та обробляє чотири основні типи даних: бібліографічні дані, нуклеотидні послідовності, амінокислотні послідовності і тривимірні структури. Крім того, у межах NCBI підтримується робота над низкою інших проєктів, зосереджених на таксономії, геномних картах, малих біоактивних молекулах тощо.

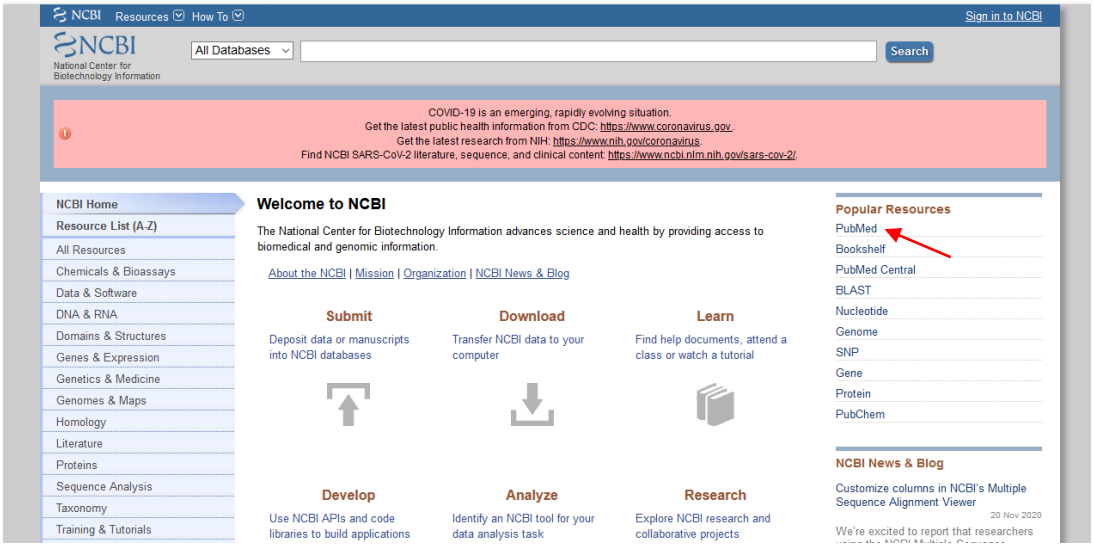


Рис. 1.1. Загальний вигляд початкової веб-сторінки NCBI (червоною стрілкою позначений гіперлінк до сторінки PubMed)

1.1. PubMed. Розгляньмо, як можна використати NCBI для пошуку наукової інформації. Для цього необхідно скористатися PubMed – гіперлінк до цього сервісу розміщений справа на початковій сторінці NCBI (див. рис. 1.1), у рубриці Popular Resources. PubMed – це електронний архів бібліографічних цитувань абсолютної більшості біомедичних видань усього світу. Наповнення цього архіву бібліографічними даними відбувається завдяки видавцям журналів, які у такий спосіб отримують найширший опис змісту своїх видань. Натискання на цей

гіперлінк відкриває веб-сторінку PubMed (рис. 1.2). Центральним елементом сторінки є пошукове вікно, куди можна ввести ключове слово, автора статті тощо й отримати список наукових публікацій, де програма знайшла задане слово.

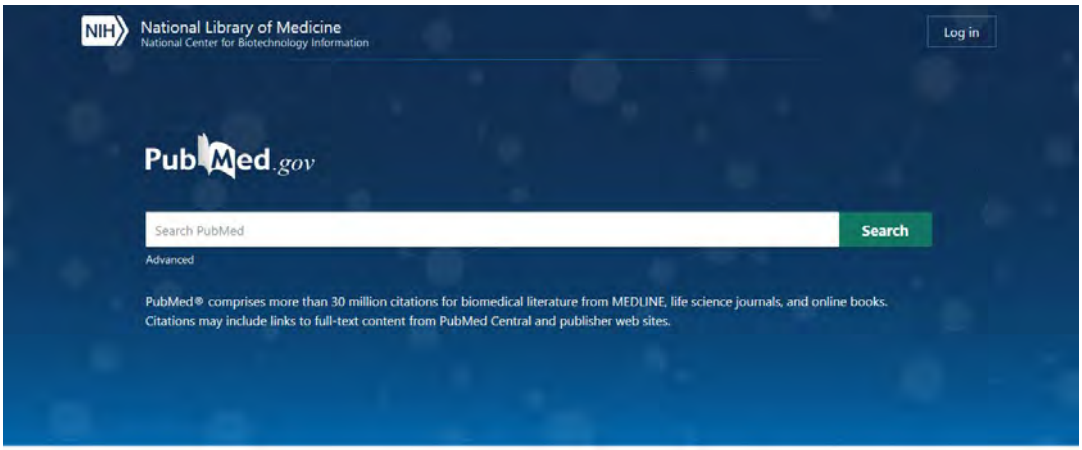


Рис. 1.2. Загальний вигляд початкової веб-сторінки PubMed

Якщо ввести у пошукове вікно слово *Streptomyces* і натиснути кнопку Search (справа від пошукового вікна, див. рис. 1.2), то програма висвітить список усіх статей, де використано це слово (рис. 1.3). Список таких статей буває дуже довгий (у цьому випадку – понад 28 тисяч), і часто потрібно звузити чи уточнити масив пошуку. PubMed пропонує широке меню засобів для цього. На початковій сторінці можна скористатися функцією Advanced, лінк до якої розміщений під пошуковим вікном (див. рис. 1.2). За допомогою цієї функції можна підібрати пошукову фразу і скерувати PubMed на пошук слова *Streptomyces* лише у назві статті, чи в поєднанні з іншим словом. На сторінці, що відображає список бібліографічних посилань, знайдених за ключовим словом (рис. 1.3), можна звузити коло пошуку за рахунок вибору категорії статей – наприклад, можна шукати це слово лише в оглядових статтях чи клінічних випробуваннях тощо. У першому випадку для цього є функція Review – зліва від списку, в рубриці My NCBI filters. Використання *операторних слів* AND, OR і NOT у пошуковому вікні PubMed є простим і корисним способом уточнення пошуку. Наприклад, оператор AND є командою пошуку статей, де обов'язково має бути вжито два певні задані слова. Тому якщо задати у пошуковому вікні словосполучення streptomyces AND peptidoglycan, то програма відобразить значно коротший список посилань (у цьому випадку – 289, рис. 1.4), ніж під час пошуку одного слова streptomyces. Зауважимо, що у простих пошуках можна обійтися без операторних слів. Наприклад, програма відобразить однакову кількість статей під час пошуку streptomyces peptidoglycan чи streptomyces AND peptidoglycan. Вживання операторних слів доречне у складних пошуках, де використані повнозначні фрази чи великі набори слів.

У відповідь на задане слово чи фразу PubMed відображає список цитувань, кожне з яких можна переглянути докладніше. Для цього натискають на назву статті, що є гіперлінком до наступного вікна (рис. 1.5). На ньому відображені

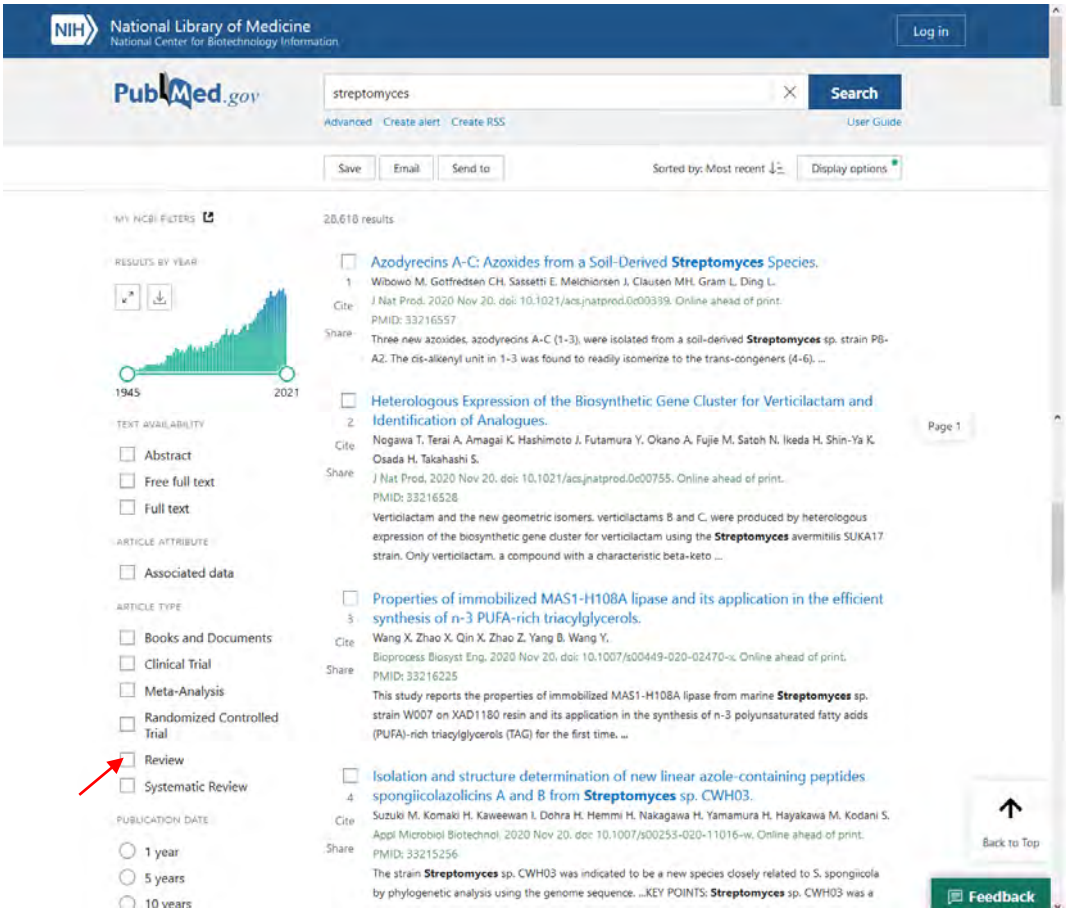


Рис. 1.3. Список статей, де вжито слово Streptomyces (червоною стрілкою позначений фільтр Review, див.: основний текст)

бібліографічні дані й анотація до статті. Часто на цій сторінці подають гіперлінк до сайта журналу, в якому надруковано статтю. Так і в нашому прикладі: у правому верхньому куті є гіперлінк до сайта журналу Scientific Reports.

PubMed можна використати для пошуку наукової інформації і створення списків літературних джерел. Бібліографічний опис у текстовому форматі отримують зі сторінки, яку зображено на рис. 1.4, за допомогою натискання гіперлінку Cite (ліворуч від опису статті), вибору в меню необхідного формату бібліографічного опису (AMA, APA, MLA і NLM). У відповідь програма завантажить на комп'ютер файл у форматі ".txt", що міститиме бібліографічний опис у вигляді тексту. Цього ж кінцевого результату можна досягти, починаючи зі сторінки з розгорнутою анотацією статті (рис. 1.5), для чого спочатку натискають гіперлінк Save (над описом статті), потім обирають у меню необхідний формат (якщо потрібен лише бібліографічний опис – то це Summary (text)). У результаті цих дій отримуємо txt-файл такого змісту:

Nuzzo D, Makitrynskyy R, Tsypik O, Bechthold A. Cyclic di-GMP cyclase SSFG_02181 from Streptomyces ghanaensis ATCC14672 regulates antibiotic biosynthesis and morphological differentiation in streptomycetes. Sci Rep. 2020 Jul 21;10(1):12021. doi: 10.1038/s41598-020-68856-9. PMID: 32694623; PMCID: PMC7374567.

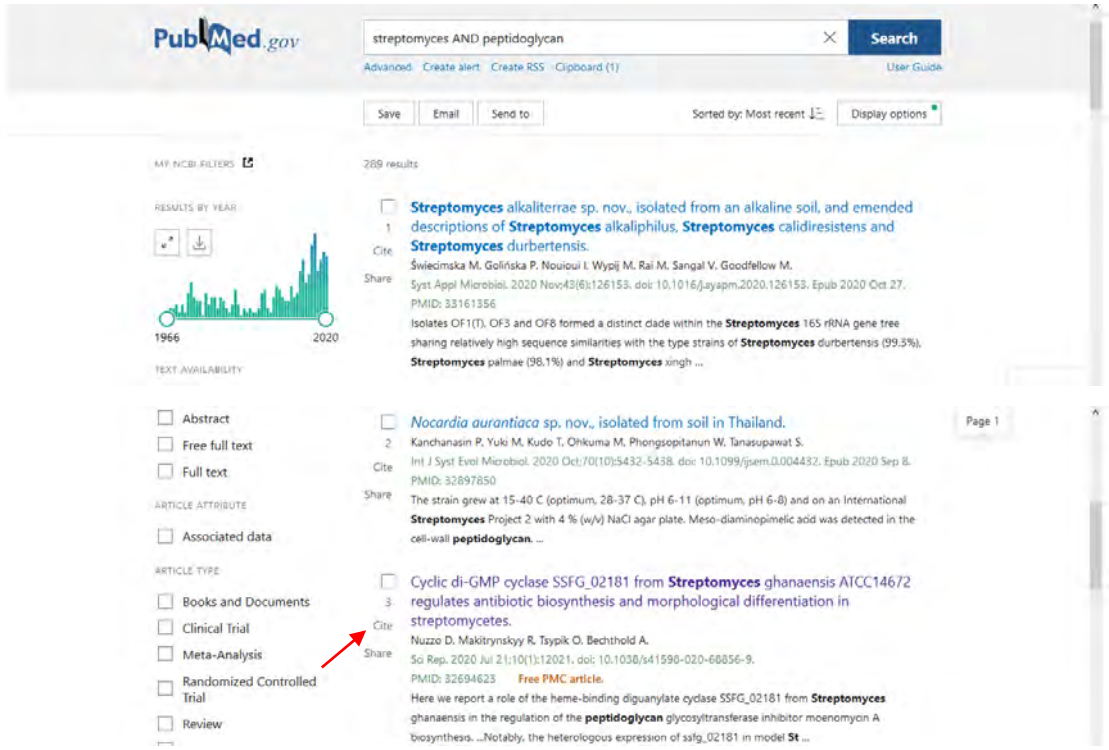


Рис. 1.4. Список статей, де вжито два слова – streptomyces AND peptidoglycan. Для ілюстрування можливостей бібліографічного опису статей в PubMed використано третю статтю у списку. Червоною стрілкою позначено лінк Cite (див.: основний текст)

Отож файл містить авторів статті, назву журналу і час видання. Опис може містити згадку про час першої електронної публікації статті – Epub, що, зазвичай, передує появі статті у друкованому варіанті журналу. В описі фігурує том журналу і сторінки статті, однак останній елемент в сучасних онлайн-виданнях часто замінюють на номер статті в журналі (як у розглянутому нами прикладі). Далі розміщена низка *ідентифікаторів статті* – комбінацій цифр та літер, що позначають цю статтю як унікальний цифровий об'єкт і за якими її можна знайти в Інтернеті. Перший ідентифікатор присвоює видавець статті (журнал, видавництво) – це т.зв. Digital Object Identifier ("цифровий ідентифікатор об'єкта" – doi), аналог універсального десяткового класифікатора (УДК), яким зараз послуговуються в українській бібліографії. Далі йдуть два інші ідентифікатори, присвоєні статті при надходженні у PubMed (PMID) та PubMed Central (PMC). PubMed Central – один з електронних сервісів NCBI, де містяться повні тексти статей, зібраних фахівцями NCBI із загальнодоступних джерел. Отже, для описаної у прикладі статті doi становить 10.1038/s41598-020-68856-9, PMID – PMID32694623, а PMCID – PMC7374567. Використання будь-якого з цих ідентифікаторів у загальних пошукових програмах (Google) має привести до самого об'єкта – тексту статті чи посилання на неї. З метою покращення пошуку інформації деякі видавництва присвоюють окремі ідентифікатори не тільки всій статті, а й її частинам (наприклад, рисункам).

Рис. 1.5. Розгорнута анотація статті в PubMed. Червоними стрілками позначено лінки до завантаження бібліографічної інформації про статтю

1.2. GenBank. База даних GenBank (https://www.ncbi.nlm.nih.gov/genbank/; рис. 1.6) поповнюється за рахунок подання нуклеотидних послідовностей. Їх

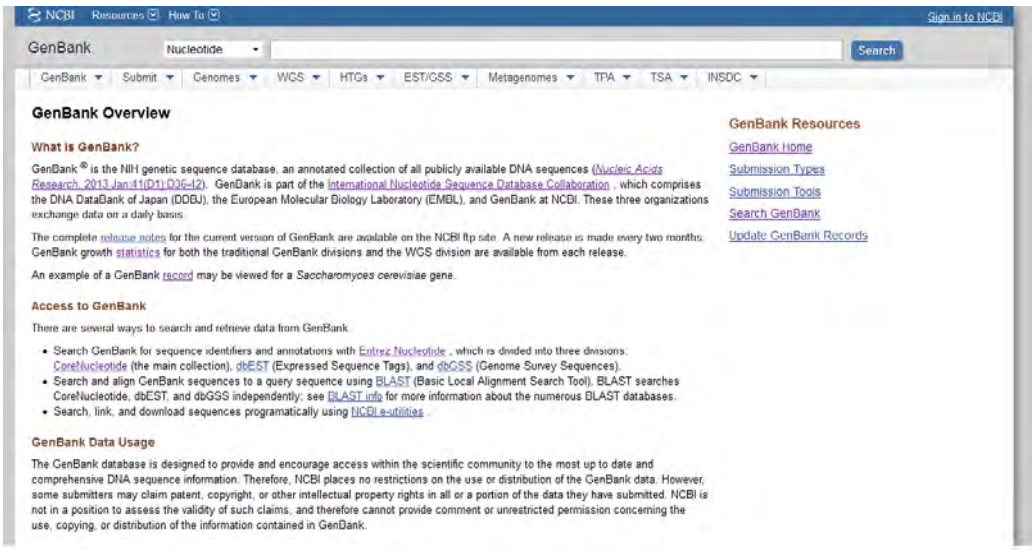


Рис. 1.6. Початкова сторінка GenBank

“віртуально” транслюють у відповідні білкові послідовності згідно з наданою авторами інформацією про координати відкритих рамок зчитування у послідовностях. Запис про нуклеотидну послідовність у базах даних (і в GenBank зокрема) містить власне саму послідовність і додаткові дані, і цей запис може бути у різних форматах. У найпростішій формі інформацію про ДНК-послідовність можна подати у вигляді стрічки нуклеотидів із коротким коментарем до неї. Подаємо файл з нуклеотидною послідовністю у форматі *FASTA*:

```
>PMI3114

TTAAGAAGCAATAGGTGTCCATAGTTCACCTAATTTATAGTCCCACTCTTTTGGTTTAATGGGAATGCG
CCCCGCATTGGGGGTAAAGGTCATTTCACTAAACAGTAGTGTGGTTTCTGACATTAAAAAATCGACCCG
ACAATAGGCAAAACCGTTAGCTAATTTGTTGGCTAATGTGAGCATATTGTCAAATTGTAGCGGTTTTTC
AATATATTCTGGGGTATTGGGATCTTCTAAAGTAAAAGGCTGTAATTGCCATTGTGTATCATAAACATT
GATA
```

Аналогічно, запис про амінокислотну (білкову) послідовність у форматі FASTA виглядатиме так:

```
>WemR

MKKLKKYLTRKKKENYIIFSIYYFIKVTSSIFISDSLYRKYIFKRKYKLNLKKPTSFNEKIHYRILNDH
NPIYTKLADKLLVRDYVREKIGEKYLIKLINHYNTPSEINFNTLPKSFVLKCNHDVGSVMIINDKSKIN
EKAIKKKLKIALKNNIYYQNREWHYKNIKPKIICEELINIFPHNKKNYPEDYKIHCFNGIPRYIELQFS
RFSHDRRINIYDFNWNLQPFLMGYKNTNESIEKPKKLQEIYNISKTLSADFDYCRVDFYITPDDSIYFG
ELTFTPCNGMDDFYPNEWDYLFGKEWIIDDSRK
```

Більшість молекулярно-біологічних і біоінформатичних сервісів розпізнають і використовують дані у форматі FASTA. Найпростіший вигляд цього формату вже показано. Значок “більше ніж” – > – позначає початок файла. Ідентифікатор (PMI3114 у першому з двох описаних прикладів) продовжений з нового рядка генетичною (нуклеотидною чи амінокислотною) послідовністю. Ці послідовності можна подавати як малими, так і великими літерами англійської абетки. Користувачі і бази даних можуть при бажанні додати свої коментарі до опису послідовності у рядок, що розпочинається з >. Цей запис може набути такої форми:

```
>gi|197283915:3424858-3425733|PMI3114| Proteus mirabilis HI4320
chromosome, complete genome
TTAAGAAGCAATAGGTGTCCATAGTTCACCTAATTTATAGTCCCACTCTTTTGGTTTAATGGGAATGCG
CCCCGCATTGGGGGTAAAGGTCATTTCACTAAACAGTAGTGTGGTTTCTGACATTAAAAAATCGACCCG
ACAATAGGCAAAACCGTTAGCTAATTTGTTGGCTAATGTGAGCATATTGTCAAATTGTAGCGGTTTTTC
AATATATTCTGGGGTATTGGGATCTTCTAAAGTAAAAGGCTGTAATTGCCATTGTGTATCATAAACATT
GATA
```

FASTA-файл містить унікальний номер-ідентифікатор gi197283915, який присвоюється фахівцями NCBI послідовності під час її потрапляння у їхню базу даних. Далі іде назва локусу (LOCUS NAME – PMI3114) і визначення (коментар до) послідовності, взятої з запису в GenBank. У такому випадку коментарем є походження цієї послідовності. На основі елементарного FASTA-файла GenBank відображає при запитах *флет-файл* (GenBank flatfile; GBFF), що є елементарною

одиницею інформації в GenBank. Розглянемо структуру GBFF, задаючи у пошуковому вікні (рис. 1.6) назву певного гена, наприклад, *moeH5*. У відповідь програма відобразить перелік записів, у яких трапляється така назва (рис. 1.7).

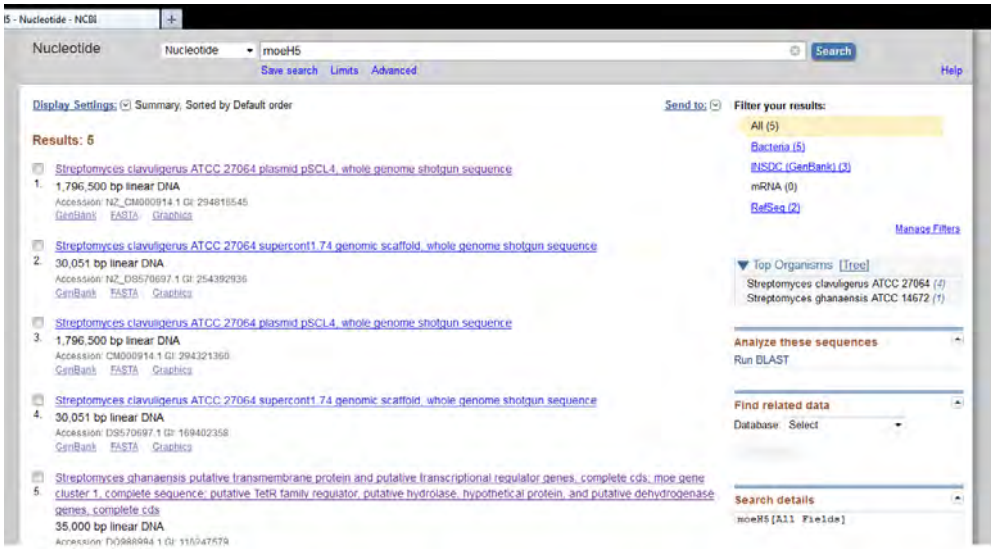


Рис. 1.7. Частковий перелік записів у GenBank, які містять інформацію про ген *moeH5*

Назви цих записів є гіперлінками до самих флет-файлів. Обираючи останній із записів, у результаті отримаємо флет-файл із записом інформації про певну нуклеотидну послідовність (рис. 1.8). Флет-файл складається з трьох основних частин: заголовка (header), опису певних властивостей цієї генетичної послідовності (features) і, власне, самої нуклеотидної послідовності. Флет-файл закінчується знаком // в останньому рядку запису.

У першому рядку заголовка флет-файла розміщене слово-ідентифікатор LOCUS, і першим елементом цього рядка є назва локусу. Зміст цього терміна змінювався з часом, і сьогодні, зазвичай, назвою локусу є номер доступу до цієї послідовності в GenBank, хоча старіші флет-файли (з "догеномної" епохи) можуть мати певну літерну абревіатуру, що відображає функцію відповідної генетичної послідовності. Після назви локусу наведено довжину генетичної послідовності – у *парах нуклеотидів* (п.н.; bp), або числі амінокислотних залишків. У нашому прикладі маємо справу з нуклеотидною послідовністю розміром 35 000 п.н. (або 35 т.п.н.). У GenBank можна депонувати послідовності в усьому діапазоні розмірів – від коротких некодувальних РНК (< 50 п.н.) до цілих геномів евкаріотів (сотні мільйонів п.н.). Для кожного типу послідовності фахівці GenBank розробили спеціалізовані підрозділи бази даних і знаряддя для їхнього коректного подання та відображення, про що докладніше опишемо далі.

Третім елементом рядка є тип молекули. Зазвичай це ДНК або РНК. Прийнятні типи молекул – ДНК, РНК, тРНК, рРНК, мРНК і нРНК (некодувальна РНК; ncRNA), і ці позначення мають відображати природу оригінальної молекули, послідовність (первинну структуру) якої встановлювали. Тобто, якщо послідовність рРНК встановлена за допомогою ПЛР геномної ДНК, то типом молекули має бути ДНК.

```
LOCUS       DQ988994               35000 bp    DNA     linear   BCT 04-APR-2007
DEFINITION  Streptomyces ghanaensis putative transmembrane protein and putative
            transcriptional regulator genes, complete cds; moe gene cluster 1,
            complete sequence; putative TetR family regulator, putative
            hydrolase, hypothetical protein, and putative dehydrogenase genes,
            complete cds.
ACCESSION   DQ988994
VERSION     DQ988994.1  GI:116247579
KEYWORDS    .
SOURCE      Streptomyces ghanaensis ATCC 14672
  ORGANISM  Streptomyces ghanaensis ATCC 14672
            Bacteria; Actinobacteria; Actinobacteridae; Actinomycetales;
            Streptomycineae; Streptomycetaceae; Streptomyces.
REFERENCE   1  (bases 1 to 35000)
  AUTHORS   Ostash,B., Saghatelian,A. and Walker,S.
  TITLE     A streamlined metabolic pathway for the biosynthesis of moenomycin
            A
  JOURNAL   Chem. Biol. 14 (3), 257-267 (2007)
   PUBMED   17379141
REFERENCE   2  (bases 1 to 35000)
  AUTHORS   Ostash,B.O. and Walker,S.
  TITLE     Direct Submission
  JOURNAL   Submitted (08-SEP-2006) Microbiology and Molecular Genetics,
            Harvard Medical School, 200 Longwood Ave., Boston, MA 02115, USA
 FEATURES             Location/Qualifiers
     source          1..35000
                     /organism="Streptomyces ghanaensis ATCC 14672"
                     /mol_type="genomic DNA"
                     /strain="ATCC14672"
                     /db_xref="ATCC:14672"
                     /db_xref="taxon:566461"
     CDS             complement(79..1614)
                     /note="ORF6"
                     /codon_start=1
                     /transl_table=11
                     /product="putative transmembrane protein"
                     /protein_id="ABJ90150.1"
                     /db_xref="GI:116247580"
                     /translation="MKTAVTDIETYGVEQIPDEDRTARPLDLFRLAFGGANTFATCVL
                     GTFPILFGLSFWQGLAATLLGLVAGALLLAPMALFGPTNGTSNSVSSSAHLGVHGRVV
                     GSFLALLTAIAFFSISVWSSGDALVGGAHRLLGVPESTVTYALAYAVFAGLVLVVCIY
                     GFRFMLLVNKIAVVAASALFVLGAFAFAGDFDPGYAGTFASTADPLFWPSFIGSALIV
                     LSNPVSFGAFLGDWSRYIPAATPRRRVMGAAFLAQIATLLPFVFGLATASIIATRAGA
                     YLDPDAPNYVGGLLAIAPGWYFLPLCLIALIGGLSTGTTALYGTGLDFCSVFTRFSRV
                     QATIFIGVLSIVFIFAGRFWLNLTQSISTFATLIITCTAPWMVVMILGYVTRRGWYDP
                     EALQVFNQRRTGGRYWFAHGWNWRGLATWLVSAAVALLFTNLPGQFVGPLGDLADGMD
                     VSLPVGLALAAVLYLVLLTLFPEPRGVYGPDGPRFVRTSDAEVLPISGGPREAPAQPA
                     VVG"
     ..................
     gene            13627..15168
                     /gene="moeH5"
     CDS             13627..15168
                     /gene="moeH5"
                     /note="putative amidotransferase subunit 2"
                     /codon_start=1
                     /transl_table=11
                     /product="MoeH5"
                     /protein_id="ABJ90159.1"
                     /db_xref="GI:116247589"
                     /translation="MTVRRPAASAPRVLLTAGPDGVRVEGDGEARLGHPLTGDHLDPG
                     PPAEGVFAGWRWDGERLVARNDRYGVCPLFYRAGGGSLALSPDPLALLPEDGPVELDH
                     DALAVFLRTGFFLAEDTAFAQVRALPPAATLTWDTGGLRLRSDGPPRPGAAAMTEAQA
                     VDGFVDLFRASVARRLPGEPYDLPLSGGRDSRHILLELCRRGAPPRRCVSGAKFPPDP
                     GADARVAAALAGRLGLPHTVVPRPRSQFRAELAALPAQGMTTLDGAWTQPVLAHLRRH
                     SRISYDGLGGGELVQNPSVEFIRANPYDPADLPGLADRLLAASRTGPHVEHLLSPRTN
                     ALWSRQAARRRLVTELARHADSASPLSSFFFWNRTRRSISAAPFALGDGRVLTHTPYL
                     DHALFDHLASVPHRFLVDGTFHDRALHRAFPEHADLGFASSVPQRHGPVLVAHRLAYL
                     LRFLAHATVVEPGWWRGPDRFLQRLLAAGRGPGAPQRVSRLQPLALYLLQLEDLAVRR
                     ARRRP"
     ..................

ORIGIN
        1 cgcgcccctc cggaggctgt ccggaagggc gcggcacggg tgggggcggg tcggccgggt
       61 tccgccggcc gggtcgggtc agccgacgac ggccggctgt gcgggcgcct cccgcgggcc
      121 gccgctgatc ggcaggacct cggcgtccga ggtccgcacg aagcgggggac cgtccggccc
      181 gtacacccca cgcggctccg ggaacaggat gagcaggacg aggtagagca ccgccgccag

     ..................

    34681 cgcctccgtg aggtacagct ccgccttgcg gtcggcgtcg gtcacgaggt cgtgcgggcc
    34741 gctcttcagg tccacctcgt gttcggcgag ccggcgccag cgcggcatga tctcctgcgc
    34801 ggcggccttg cggacggctt cctccacgtc gacggcgtgc cggtcgagaa actcttcgat
    34861 ggtttcgttg tccttgatca tgcctccatg agaccacgcc cggccgacgt tccccaccgt
    34921 cccggtgcac tgcgggggga atcggcatga atatggggtg ccggaccacg ggcgggtggg
    34981 ctcagcggcc caccgcgtag
//
```

Рис. 1.8. Структура флет-файла, що містить дані про ген moeH5. Рядком з крапок позначені видалення (для зручності перегляду) однотипних елементів файла і його розміщення на одній сторінці

Четвертий елемент рядка – трилітерна абревіатура, яку GenBank присвоює послідовності. Ця абревіатура (код) має таксономічне або класифікаційне значення (у нашому випадку – BCT), та означає, що послідовність походить з бактерій (**B**A**CT**ERIA). Крім суто таксономічних кодів, у флет-файлах можна натрапити й на такі, що позначають різновид експерименту, під час якого отримано інформацію про послідовність. Наприклад, коди EST (Expressed Sequence Tags), STS (Sequence Tagged Sites), GSS (Genome Survey Sequences), HTG (High-Throughput Genome sequences), WGS (Whole Genome Sequences), CON (Contigged sequences) тощо. Останній із кодів використовують у флет-файлах, які не містять інформації про генетичні послідовності. Замість того, вони містять інструкцію, наприклад, як скласти певний геном із окремих послідовностей, що описані у складі інших флет-файлів. Підрозділи GenBank, які містять записи з певним кодом, можна знайти і переглянути з початкової сторінки (див. рис. 1.6).

Останнім елементом першого рядка є дата оприлюднення запису про генетичну послідовність (флет-файл). Якщо будь-яку частину послідовності чи її анотацію змінювали, а флет-файл повторно оприлюднений, то дата у першому рядку збігається з часом внесення останніх змін. Дата оприлюднення не збігається з датою подання послідовності у базу GenBank – останню подаємо у флет-файлі (опис структури флет-файла наведемо далі). База даних не гарантує точності відображених дат, їх подають як зручні орієнтири для всіх, хто користується цими даними. Отож вони не мають юридичної сили, їх не можна використати як аргумент у судовій суперечці чи для встановлення пріоритету у патентних поданнях.

Другий рядок флет-файла розпочинається терміном DEFINITION:

```
DEFINITION  Streptomyces ghanaensis putative transmembrane protein and
            putative transcriptional regulator genes, complete cds;
            moe gene cluster 1, complete sequence; putative TetR
            family regulator, putative hydrolase, hypothetical
            protein, and putative dehydrogenase genes, complete cds.
```

У цьому рядку описують біологічний зміст певної НАП (що саме кодує ця послідовність). У нашому випадку, 35-т.п.н. послідовність клоновано з продуцента антибіотика моеноміцину *Streptomyces ghanaensis*, і вона містить гени біосинтезу цього антибіотика (*moe*-гени), а також декілька генів, що фланкують *moe*-гени. Цей рядок є й у FASTA-файлах, що відображає NCBI під час пошуку якихось послідовностей (наприклад, якщо задати назву гена *moeH5* у пошуковому вікні NCBI (див. рис. 1.3); результат пошуку (див. рис. 1.7)). Як видно з усіх цих прикладів, рядок запису – доволі місткий, але й він інколи не дає змоги адекватно описати всі аспекти біології розглядуваної послідовності. Цей рядок генерується автоматично на основі інформації, поданої авторами послідовності, а потім перевіряється фахівцями NCBI для гарантування точності і змістовності запису. Сьогодні розроблено правила, які врегульовують зміст рядка DEFINITION. Наприклад, якщо типом молекули є мРНК, то в DEFINITION мають бути такі елементи опису:

```
Genus species product name (gene symbol) mRNA, complete cds.
```

Якщо ж флет-файл репрезентує запис послідовності з генома:

```
Genus species product name (gene symbol) gene, complete cds.
```

Згідно з сучасними правилами, необхідно подавати повну родову і видову назви організму (тобто *Homo sapiens*, а не *H. sapiens*). Винятком з цього правила наразі є вірус імунодефіциту людини – його позначають в рядку DEFINITION як HIV1 чи HIV2.

Третій рядок флет-файла:

```
ACCESSION   DQ988994.
```

Номер доступу – ключ до певної НАП у базі даних. Цей номер цитують у публікаціях, де вперше описано НАП, або в яких спираються на неї у своїх дослідженнях. Номер доступу завжди асоційований з записом (флет-файлом). Якщо послідовність певним чином оновлена (наприклад, доданий чи видалений один нуклеотид), то номер доступу не зміниться. У деяких випадках флет-файл містить більше одного номера доступу (переважно стосується старіших записів). У цих випадках перший номер доступу вважають первинним, усі інші – вторинними. Протягом років значення вторинних номерів змінювалось, і воно не підлягає простому набору правил. Це, передусім, виняткові випадки, що не стосуються абсолютної більшості даних.

Наступний рядок треба розглядати разом із рядком ACCESSION:

```
ACCESSION   DQ988994
VERSION     DQ988994.1  GI:116247579.
```

Рядок VERSION містить інформацію про версію номера доступу (скільки разів мінялася вихідна послідовність). У нашому прикладі – це версія 1 (DQ988994.1), тобто послідовність перебуває у незмінному стані з моменту її подання. Далі йде унікальний ідентифікатор *gi/GI* (geninfo). Він асоційований лише з однією нуклеотидною послідовністю. Станом на вересень 2016 р. NCBI припинив використання gi у флет-файлах, але ще якийсь час він буде відображений у старих записах. Для білків також є ідентифікатор (protein_ids). Якщо послідовність змінюватиметься, то номер версії кожен раз зростатиме на одиницю, а GI змінюватиметься на нове, вакантне на той момент, число. Номер доступу (ACCESSION) залишатиметься незмінним.

Далі йде рядок KEYWORDS:

```
KEYWORDS.
```

Це історичний “релікт”, зараз NCBI/GenBank загалом не вимагає переліку ключових слів (обмеження не стосується інших баз даних).

Далі подають два рядки, в яких описують джерело НАП:

```
SOURCE      Streptomyces ghanaensis ATCC 14672
  ORGANISM  Streptomyces ghanaensis ATCC 14672
            Bacteria; Actinobacteria; Actinobacteridae;
Actinomycetales; Streptomycineae; Streptomycetaceae; Streptomyces.
```

Рядок SOURCE містить загальновживану або ж наукову назву організму. У підрядку ORGANISM наведено таксономічну позицію організму. Родова і видова назви організму слугують гіперлінком до таксономічної сторінки NCBI.

Кожен запис GenBank мусить мати принаймні одне посилання чи цитування. Таку інформацію відображено у рядку REFERENCE, який має низку підрядків:

```
REFERENCE   1  (bases 1 to 35000)
   AUTHORS   Ostash,B., Saghatelian,A. and Walker,S.
   TITLE     A streamlined metabolic pathway for the biosynthesis of
             moenomycin A
   JOURNAL   Chem. Biol. 14 (3), 257-267 (2007)
    PUBMED   17379141
 REFERENCE   2  (bases 1 to 35000)
   AUTHORS   Ostash,B.O. and Walker,S.
   TITLE     Direct Submission
   JOURNAL   Submitted (08-SEP-2006) Microbiology and Molecular
             Genetics, Harvard Medical School, 200 Longwod Ave,
             Boston, MA02115, USA.
```

Цей рядок містить інформацію про авторів НАП, статтю, де її вперше описано, ідентифікатор статті в PubMed (див. попередній підрозділ 1.1). Здебільшого флет-файли містять два цитування, як і у цитованому попередньо випадку: перше – посилання на статтю, де опублікована ідентифікація НАП; друге – посилання на дату прямої подачі НАП (Direct Submission) у GenBank. Тобто послідовність, що містить moe-гени, подана авторами статті в базу даних 8 вересня 2006 р. (підрядок JOURNAL у REFRENCE 2), а стаття вийшла згодом, у березні 2007 р. Запис про послідовність (флет-файл, що слугує тут прикладом) став доступним у GenBank з 4 квітня 2007 р. (перший рядок флет-файла, див. текст і рис. 1.8).

Останнім (необов'язковим) рядком заголовка у деяких випадках є рядок COMMENT. Його немає у нашому прикладі. Цей рядок часто є у записах про повну послідовність геномів, де автори подають свою контактну інформацію чи інші нотатки. У рядку може міститися інформація про зміни в нуклеотидній послідовності і її попередні gi-ідентифікатори. Якщо ж дослідник знаходить у базі даних старий запис, то в ньому, у рядку COMMENT, міститиметься переадресування на новіші версії запису.

Усі розглянуті рядки є частиною заголовка флет-файла, і після них іде т.зв. "таблиця ознак" (feature table), або "опис ознак" (FEATURES) – центральний елемент флет-файла, що є безпосередньою репрезентацією біологічної інформації в записі (файлі). Першим елементом таблиці ознак є рядок source. Він складається з низки підрядків:

```
FEATURES             Location/Qualifiers
     source          1..35000
                     /organism="Streptomyces ghanaensis ATCC 14672"
                     /mol_type="genomic DNA"
                     /strain="ATCC14672"
                     /db_xref="ATCC:14672"
                     /db_xref="taxon:566461".
```

Усі НАП мають певне походження, і в розглянутому випадку послідовність походить з певного організму. Відтак у записі фігурує термін organism. Трапляється також термін synthetic – стосується хімічно синтезованої ДНК. Термін organism далі розкривається в описі родової і видової наукової назви організму. Нижче описують інші ознаки організму, з якого походить послідовність (тип молекули, колекційний номер організму, таксон тощо).

Після рядка source описують інші ознаки НАП. У розглянутому випадку (стосується більшості записів у GenBank) – це опис розміщення *кодувальних послідовностей* (coding sequences; CDS) у наведеному фрагменті ДНК. Тобто послідовність розміром 35 000 п.н. кодує низку білків, а опис цієї ознаки включає координати першого кодону і стоп-кодону ймовірних (чи експериментально встановлених) *відкритих рамок зчитування* (open reading frames; ORF/orf). Ці координати надають автори послідовності. Відповідно до цих координат GenBank відображає продукти концептуальної трансляції окремих сегментів послідовності:

```
CDS             complement(79..1614)
                     /note="ORF6"
                     /codon_start=1
                     /transl_table=11
                     /product="putative transmembrane protein"
                     /protein_id="ABJ90150.1"
                     /db_xref="GI:116247580"

/translation="MKTAVTDIETYGVEQIPDEDRTARPLDLFRLAFGGANTFATCVLTFPILFGLSFW
QGLAATLLGLVAGALLLAPMALFGPTNGTSNSVSSSAHLGVHGRVVGSFLALLTAIAFFSISVWSSGDA
LVGGAHRLLGVPESTVTYALAYAVFAGLVLVVCIYGFRFMLLVNKIAVVAASALFVLGAFAFAGDFDPG
YAGTFASTADPLFWPSFIGSALIVLSNPVSFGAFLGDWSRYIPAATPRRRVMGAAFLAQIATLLPFVFG
LATASIIATRAGAYLDPDAPNYVGGLLAIAPGWYFLPLCLIALIGGLSTGTTALYGTGLDFCSVFTRFS
RVQATIFIGVLSIVFIFAGRFWLNLTQSISTFATLIITCTAPWMVVMILGYVTRRGWYDPEALQVFNQR
RTGGRYWFAHGWNWRGLATWLVSAAVALLFTNLPGQFVGPLGDLADGMDVSLPVGLALAAVLYLVLLTL
FPEPRGVYGPDGPRFVRTSDAEVLPISGGPREAPAQPAVVG"
     CDS             1780..3171
                     /note="ORF5"
                     /codon_start=1
                     /transl_table=11
                     /product="putative transcriptional regulator"
                     /protein_id="ABJ90151.1"
                     /db_xref="GI:116247581"

/translation="MRAGKSGLSRSVSWAHTSELADPTPWLLGAEVIMTTGLAIPRTATGQRRYLERLD
DAGVSALALSAQLHMPPLHDAFFKAAEERGFPVLEVPLAVPFIAVSQEVAAAVQEDARHRLGAQLQVFG
SLRWMVAEDLDTPTLLRRLERLSGYNVFLCTPQGRPLLPGVPTPDPGVLPASVDAPPTVPGGFVLPVPA
PGGPAGFLVAYERQGAQPAGLAVVQHIATVAALRLAMVRNERETLRREGAETLAELLREVLDPDAARRR
LARHAIEGETVLLVVRNTTDEALLHCLEDRPHLLLTRGDDRYVLGAPELAPAIGELPGVAAGMSRALPP
GAALKVAEREALWALSKAVESGRPLVRYGDDATGRWLPEDPAVLSALVEHVLGEVLRYDLAHGSQLLVS
VRTWLERDRRTETAAAALHIHPNTLAYRLRRFGALSRRDLSSTGALAEVWLAIQAAGTLGLTD".
```

У наведеному прикладі відображені продукти трансляції перших двох ORF у складі 35-т.п.н. послідовності. Кодувальною ниткою ДНК називають ту, нуклеотидна послідовність якої ідентична мРНК, що транслюватиметься рибосомою. Кодогенна нитка – та, котру розпізнає РНК-полімераза і до котрої добудовує комплементарну нитку мРНК. Координати ORF подані у рядку CDS, що також слугує гіперлінком до нуклеотидної послідовності ORF. В описі першої з двох ORF вжите слово complement, а це означає, що ген кодується комплементарною ниткою ДНК, тобто напрям його транскрипції (завжди відбувається в керунку 5′→3′) протилежний напряму транскрипції іншого гена. Відтак зображення напрямів транскрипції генів ORF6 і ORF5 (з їхніх кодогенних ниток) у вигляді стрілок буде таким, як проілюстровано на рис. 1.9.

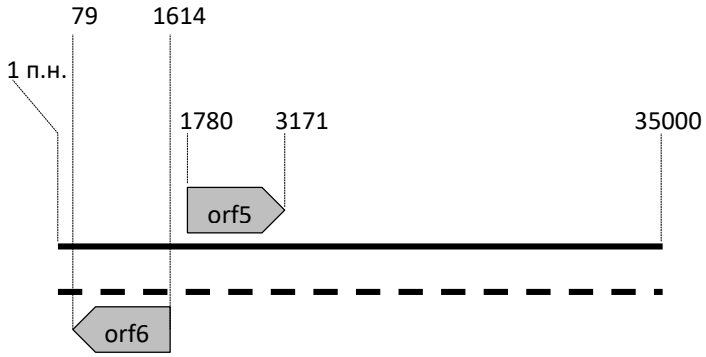


Рис. 1.9. Розміщення генів orf6 і orf5 у межах 35-т.п.н. нуклеотидної послідовності, яку розглядаємо у цьому розділі. Неперервною жирною чорною лінією позначено нуклеотидну послідовність, подану у GenBank (DQ988994), яка є кодувальною для orf5 і кодогенною для orf6 (комплементарна нитка зображена штриховою чорною жирною лінією; перший нуклеотид старт-кодону й останній нуклеотид стоп-кодону orf6 – у позиціях 1 614 і 79)

У переліку усіх CDS в межах розглянутої послідовності є *moeH5* – вихідний ген, що шукали в GenBank:

```
gene            13627..15168
                     /gene="moeH5"
     CDS             13627..15168
                     /gene="moeH5"
                     /note="putative amidotransferase subunit 2"
                     /codon_start=1
                     /transl_table=11
                     /product="MoeH5"
                     /protein_id="ABJ90159.1"
                     /db_xref="GI:116247589"

/translation="MTVRRPAASAPRVLLTAGPDGVRVEGDGEARLGHPLTGDHLDPGPPAEGVFAGWR
WDGERLVARNDRYGVCPLFYRAGGGSLALSPDPLALLPEDGPVELDHDALAVFLRTGFFLAEDTAFAQV
RALPPAATLTWDTGGLRLRSDGPPRPGAAAMTVDGFVDLFRASVARRLPGEPYDLPLSGGRDSRHILLE
LCRRGAPPRRCVSGAKFPPDPGADARVAAALAGRLGLPHTVVPRPRSQFRAELAALPAQGMTTLDGAWT
QPVLAHLRRHSRISYDGLGGGELVQNPSVEFIRANPYDPADLPGLADRLLAASRTGPHVEHLLSPRTNA
LWSRQAARRRLVTELARHADSASPLSSFFFWNRTRRSISAAPFALGDGRVLTHTPYLDHALFDHLASVP
HRFLVDGTFHDRALHRAFPEHADLGFASSVPQRHGPVLVAHRLAYLLRFLAHATVVEPGWWRGPDRFLQ
RLLAAGRGPGAPQRVSRLQPLALYLLQLEDLAVRRARRRP"
```

Тут потрібно звернути увагу на те, що поряд з терміном CDS у наведеному попередньо прикладі вжитий також термін gene. У межах рядка, що розпочинається цим терміном, можна подати назву гена чи навести іншу інформацію про нього.

Завершує таблицю ознак сама нуклеотидна послідовність, яку подано у базу даних. Запис нуклеотидної послідовності розпочинається терміном ORIGIN і закінчується двома скісними рисками //:

```
ORIGIN
        1 cgcgcccctc cggaggctgt ccggaagggc gcggcacggg tgggggcggg tcggccgggt
       61 tccgccggcc gggtcgggtc agccgacgac ggccggctgt gcgggcgcct cccgcgggcc
      121 gccgctgatc ggcaggacct cggcgtccga ggtccgcacg aagcggggac cgtccggccc
      181 gtacaccccg cgcggctccg ggaacagggt gagcaggacg aggtagagca ccgccgccag
      241 ggccaggccg accggcagcg agacgtccat gccgtcggcc aggtcgccga gcggtccgac
```

```
     .....
    34801 ggcggccttg cggacggctt cctccacgtc gacggcgtgc cggtcgagaa actcttcgat
    34861 ggtttcgttg tccttgatca tgcctccatg agaccacgcc cggccgacgt tccccaccgt
    34921 cccggtgcac tgcgggggga atcggcatga atatggggtg ccggaccacg ggcgggtggg
    34981 ctcagcggcc caccgcgtag
//
```

Для отримання нуклеотидної послідовності окремого гена чи продукту його трансляції необхідно використати таблицю ознак. Наприклад, щоб отримати нуклеотидну послідовність *moeH5*, натискають на гіперлінк CDS поблизу опису цього гена. У відповідь програма виділить у 35-т.п.н. послідовності той сегмент, що відповідає шуканим генам, а у правому нижньому куті вікна відобразить набір дій, які можна виконати над виділеною послідовністю (рис. 1.10). Зокрема, програма дає змогу відобразити послідовність гена *moeH5* у форматі FASTA чи окремого флет-файла (GBFF), присвяченого виключно цьому гену, зображеному на рис. 1.9. Він містить усі елементи, які щойно розглянуто на прикладі усієї 35-т.п.н. послідовності. Крім того, на рис. 1.11 наведено всі гіперлінки (справа від флет-файла), за допомогою яких можна виконати низку корисних дій над досліджуваною послідовністю. Команда Change region shown корисна для отримання нуклеотидної послідовності, що прилягає до кодувальної ділянки *moeH5*. Наприклад, якщо досліднику цікаво вивчити нуклеотидну послідовність розміром 200 п.н. у 5′-напрямі від старт-кодона (ймовірна промоторна ділянка), то у вікнах під командою Change region shown треба змінити координати відображуваної послідовності. Зокрема, координату from 13 627 треба замінити на 13 427, а потім натиснути знизу кнопку Update View. У результаті програма відобразить нову нуклеотидну послідовність, що містить усю кодувальну частину *moeH5* (1 542 п.н.), а також 200 перед старт-кодоном – у сумі 1 742 п.н.

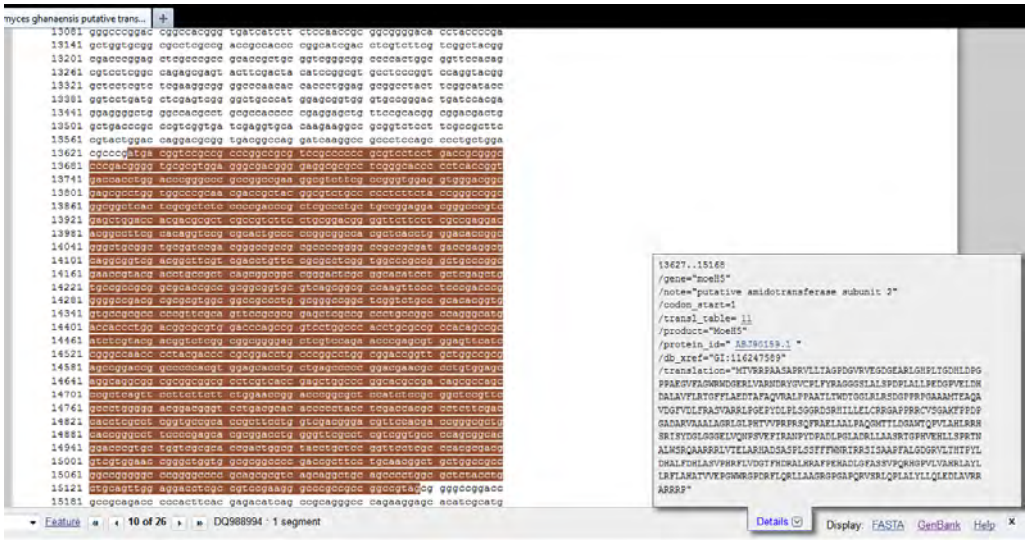


Рис. 1.10. Вікно, що виникає у відповідь на натискання на гіперлінк CDS (зліва від опису гена *moeH5*, див. рис. 1.8). Кольором виділено послідовність *moeH5* (координати 13 627 – 15 158) у межах 35-т.п.н. нуклеотидної послідовності. У правому нижньому куті – меню подальших дій

Так можна відобразити будь-який фрагмент 35-т.п.н. послідовності. Варто пам’ятати, що у розглянутому випадку ген розміщений на кодувальній нитці (умовно кажучи, транскрибується зліва направо, див. рис. 1.9). Тому для отримання промоторної послідовності збільшують район послідовності "ліворуч" від початку гена (координата from, рис. 1.11). Якщо ген розміщено на

комплементарній нитці, то для отримання промоторної послідовності збільшують на 200 п.н. район послідовності справа (поблизу координати to, рис. 1.11). Це застереження пов'язане насамперед з особливостями презентації послідовностей у базі GenBank. В інших базах даних можуть бути свої правила презентації даних, які необхідно враховувати при роботі з ними.

```
    gene            1..1542
                    /gene="moeH5"
    CDS             1..1542
                    /gene="moeH5"
                    /note="putative amidotransferase subunit 2"
                    /codon_start=1
                    /transl_table=11
                    /product="MoeH5"
                    /protein_id="ABJ90159.1"
                    /db_xref="GI:116247589"
                    /translation="MIVRRPAASAPRVLLTAGPDGVRVEGDGEARLGHPLTGDHLDPG
                    PPAEGVFAGWRWDGERLVARNDRYGVCPLFYRAGGGSLALSPDPLALLPEDGPVELDH
                    DALAVFLRTGFFLAEDTAFAQVRALPPAATLTWDTGGLRLRSDGPPRPGAAAMTEAQA
                    VDGFVDLFRASVARRLPGEPYDLFLSGGRDSRHILLELCRRGAPPRRCVSGAKFPFDP
                    GADARVAAALAGRLGLPHTVVPRPRSQFRAELAALPAQGMTTLDGAWTQPVLAHLRRH
                    SRIGYDGLGGGELVQNPSVEFIRAMPYDPADLPGLADRLLAASRTGPSVEHLLSPRTN
                    ALNSRQAARRRLVTELARHADSASPLSSFFFWNRTRRSISAAPFALGDGRVLTHTPYL
                    DHALFDHLASVPHRFLVDGTFHDRALHRAFPEHADLGFASSVPQRHGPVLVAHRLAYL
                    LRFLAHATVVEPGWWRGPDRFLQRLLAAGRGPGAPQRVSRLQPLALYLLQLEDLAVRR
                    ARRRP"
ORIGIN
        1 atgacggtcc gccgccccggc cgcgtccgcc ccccgcgtcc tcctgaccgc gggccccgac
       61 ggggtgcgcg tggagggcga cggggaggcg cgcctcgggc acccctcac cggtgaccac
      121 ctggacccgg gcccgccggc cgaaggcgtc ttcgccgggt ggaggtggga cggcgagcgc
      181 ctggtggccc gcaacgaccg ctacggccgtc tgccccctct tctaccgggc cggcggcggc
      241 tcactcgcgc tctccccega cccgctcgcc ctgctgccgg aggacgggcc cgtcgagctg
      301 gaccacgacg cgctcgccgt cttcctgcgg acgggtttct tcctcgccga ggacacggcc
      361 ttcgcacagg tccgccgcact gccccccggcg gccacgctca cctgggacac cggcgggctg
      421 cggctgcggt ccgacgggcc gccgccccc ggggccgccg cgatgaccga ggcgcaggcg
      481 gtcgacggct tcgtcgacct gttccgcgcc tcggtggccc gccggctgcc cggcgaaccg
      541 tacgacctgc cgctcagcgg cggccgggac tcgcggcaca tcctgctcga gctgtgccgc
      601 cgcggcgcac cgccgcggcg gtgcgtcagc ggcgccaagt tccctcccga ccccgggggcc
      661 gacgcgcgcg tggcggccgc cctggcgggc cggctcggtc tgccgcacac ggtggtgccg
      721 cgccccgtt cgcagttccg cgcggagctc gccgcctgc cggcccaggg catgaccacc
      781 ctggacgggcg cgtggaccca gccggtcctg gcccacctgc gccgccacag ccgcatctcg
      841 tacgacggtc tcggcggcgg ggagctcgtc cagaacccga gcgtggagtt catccgggcc
      901 aaccccctacg acccgcggga cctgcccggc ctggcggacc ggttgctggc cgcgagccgg
      961 accgcccccc acatcgaaca cctactgaac ccccgaacaa acaccctatc gacgacgcag
      961 accggccccc acgtggagca cctgctgagc cccggacga acgccctgtg gagcaggcag
     1021 gcggcggcgc ggcgcctcgt caccgagctg gcccggcacg ccgacagcgc cagccccgtc
     1081 agttccttct tcttctggaa ccggaccccgg cgctccatct ccgcggctcc gttcgccctg
     1141 ggggacggac gggtcctgac gcacaccccc tacctcgacc acgccctctt cgaccacctc
     1201 gcctcggtgc cgcaccgctt cctggtcgac gggacgttcc acgacggggc gctgcaccgg
     1261 gccttccccg agcacgcggga cctggggttc gcctcgtcgg tgccccagcg gcacggaccc
     1321 gtgctggtcg cgcaccgact ggcgtacctg ctccggttcc tcgcccacgc gacggtcgtg
     1381 gaaccgggct ggtggcgcgg cccgacccgc ttcctgcaac ggctgctggc cgccggccgg
     1441 gggccccggg cccgcagcg cgtcagcagg ctgcagcccc tggcgctcta cctgctgcag
     1501 ttggaggacc tcgccgtccg aagggcccgc cgccggccgt ag
//
```

Рис. 1.11. Структура флет-файла з послідовністю *moeH5*

Флет-файли зручні для перегляду й отримання інформації про НАП, але зберігати їх і використовувати для подальшого аналізу у такому форматі незручно. Нуклеотидну послідовність, описану у флет-файлі, можна перевести у простий FASTA-формат, користуючись гіперлінком FASTA (зліва під заголовком флет-файла, див. рис. 1.11). У результаті отримаємо сторінку, що міститиме типовий FASTA-файл (рис. 1.12).

Рис. 1.12. FASTA-файл з послідовністю *moeH5*

Користуючись командою Send (вгорі у правому куті, див. рис. 1.10), описаний файл можна завантажити на комп'ютер у текстовому (.txt) форматі.

Сучасний варіант GenBank є "білок-центричним" – це помітно зі структури уже розглянутих флет-файлів, оскільки їхнім основним елементом є опис координат генів, що кодують білки, і продуктів їхньої трансляції. Шлях до виявлення нуклеотидної послідовності окремого гена за його назвою був непрямим – пошук здійснювали від виявлення більшої послідовності, у межах якої знаходився *moeH5*. Якщо поставлене завдання "знайти ген, що кодує білок", то коротшим шляхом стане пошук в GenBank за назвою білка. Сьогодні особливості генетичної номенклатури різних груп організмів найкраще висвітлені у спеціалізованих базах даних. Зокрема: для плодової мушки – FlyBase, для нематод – WormBase; для кукурудзи і рослин – MaizeGDB; для людини – база HGNC (http://www.genenames.org/). Стосовно розглянутого у цьому розділі прикладу зазначимо, що в бактерійній генетиці назви генів і білків відрізняються, здебільшого, тим, що білки записують з великої літери, у розглянутому вже прикладі – ген *moeH5* і білок MoeH5. Якщо ввести "MoeH5" у пошукове вікно GenBank та обрати команду Protein у меню зліва від вікна, то отримаємо список усіх записів, що містять MoeH5/*moeH5* у назві флет-файла. Перший із записів буде за структурою і змістом подібний до зображеного на рис. 1.11, за винятком того, що він не міститиме запису нуклеотидної послідовності – лише амінокислотної. Нуклеотидну послідовність *moeH5* можна отримати, скориставшись гіперлінком CDS. Як бачимо, цей шлях пошуку не передбачає відображення всієї 35-т.п.н. послідовності (що було вихідним джерелом інформації про *moeH5*/MoeH5), а одразу спрямовує дослідника до інформації про певний білок і ген.

Як здійснюють подання генетичних послідовностей? Якщо це нуклеотидні дані, то їх подають у будь-яку з трьох баз даних міжнародного консорціуму INSDC – GenBank, DDBJ чи ENA. Між ними немає різниці, отож обрання бази може визначатися географічною близькістю чи звичкою дослідника. Якщо подають амінокислотні послідовності (є дослідницькі групи, які секвенують безпосередньо білки), то найкращою базою буде SWISSPROT. Далі увагу буде зосереджено на поданні нуклеотидних послідовностей, оскільки вони становлять абсолютну більшість усієї нової інформації, генерованої зараз. Як приклад розглянемо подання послідовностей через GenBank.

На початковій сторінці GenBank (див. рис. 1.6) зверху справа є рубрика GenBank Resources, яка налічує низку гіперлінків, зокрема Submission Types. Користуючись цим гіперлінком, дослідник довідається, які типи нуклеотидних послідовностей приймає GenBank. GenBank не приймає: перервані послідовності, послідовності праймерів, лише амінокислотні послідовності без відповідних нуклеотидних, суміші послідовностей геномів і мРНК (складно інтрепретувати у випадку інтрон-вмісних генетичних послідовностей), послідовності до 200 п.н. Для деяких із цих типів даних, наприклад, для коротких послідовностей, що генеруються ультрашвидкими методами секвенування (454, Illumina, SOLiD тощо), NCBI має додаткові портали. Це Sequence Read Archive і Trace Archive (для них розроблені свої правила подання, які ми не розглядатимемо).

За гіперлінком Submission Tools описані наявні знаряддя (сервіси) для подання нуклеотидних послідовностей. Зараз GenBank пропонує чотири різновиди сервісів. Перший з них – BankIt – онлайн-знаряддя для подання однієї чи декількох простих (коротких) послідовностей, що не потребують складної анотації (інтрони-екзони, регуляторні ділянки тощо). Це проста інтерактивна програма, яка потребує від дослідника введення у вікна інформації про себе, послідовність, а також опис її ознак – число CDS, їхні координати тощо. Уведена інформація проходить формальну перевірку співробітниками GenBank, які мають упевнитися в тому, що подано всю необхідну інформацію про послідовність, для її подальшої коректної презентації у флет-файлах. Якщо об’єктом подання є великий набір послідовностей (або дуже довгі послідовності) та існує потреба у розширеному наборі знарядь їхнього аналізу та графічного опису, то варто скористатися SequIn. Цю програму можна завантажити зі сторінки GenBank. Програма Tbl2asn має набір функцій як і в SequIn, однак її роботу контролюють уведенням команд у діалогове вікно програми. Її використовують для автоматизації процесу подання в GenBank послідовностей цілих геномів. Знаряддя Submission Portal – уніфікована система депонування нуклеотидних послідовностей рРНК та деяких вірусів. Геномні та транскриптомні дані можна подати через рубрику Genomes (https://www.ncbi.nlm.nih.gov/genbank/genomesubmit/).

Незалежно від типу подання, перед початком дослідникові необхідно знати відповідь на низку питань, що стосуються походження послідовності та її біологічного змісту. Як ми зауважили попередньо, сучасні первинні бази даних – “білок”-центричні. Якщо подати в базу даних нуклеотидну послідовність без її анотації (не зазначаючи координат бодай одного гена (orf)), то програма BankIt висуне попередження, яке попросить дослідника підтвердити, що наявність генів у цій послідовності залишається невідомою.

1.3. Genome. Корисний ресурс у складі NCBI – база даних геномів, які повністю/частково секвеновані чи перебувають у процесі секвенування – Genome. Гіперлінк до цього ресурсу розміщений на початковій сторінці NCBI, де й гіперлінк до PubMed. Початкова сторінка Genome (рис. 1.13) містить пошукове

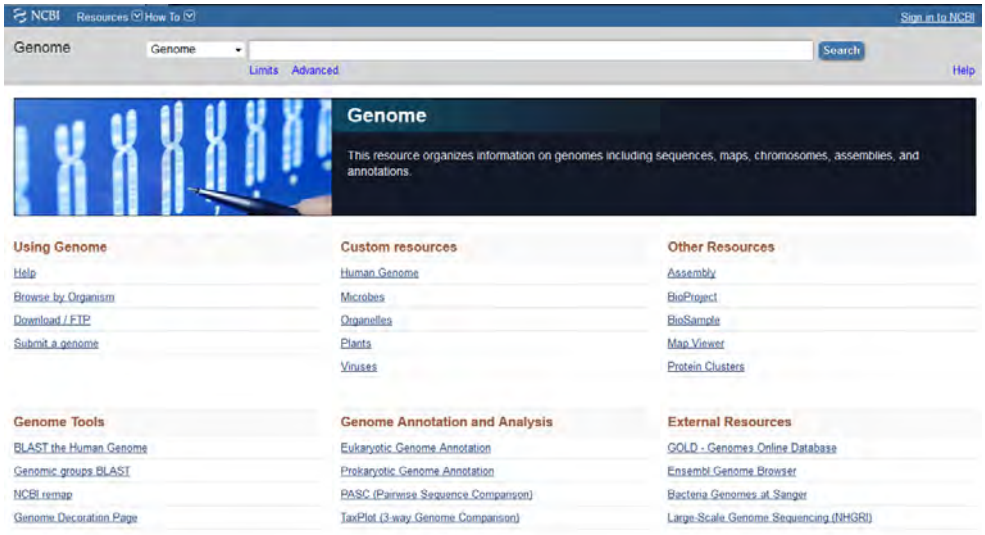


Рис. 1.13. Початкова сторінка бази даних Genome

вікно, куди можна увести родову і видову назви організму, геном якого цікавить. Наприклад, уведення *Streptomyces coelicolor* у вікно скеровує програму до сторінки, де відображена інформація про геном цього організму (рис. 1.14). Сторінка є своєрідним порталом для пошуку докладнішої інформації щодо обраного генома. Зверху поданий опис організму чи відповідної групи, і таксономічне місце. Далі описані наявні проєкти секвенування геномів цього виду – як завершені, так і на різних стадіях роботи. У цьому випадку завершене секвенування генома *S. coelicolor* M145, одного зі штамів цього виду. У таблиці подано номер доступу до повної нуклеотидної послідовності хромосоми цього виду – NC003888.3 і AL645882.2. За цими номерами дослідник знайде флет-файли типу CON, які містять інформацію про контіги генома і порядок їхнього складання у повний геном (див. попередній текст). Далі у таблиці перераховано основні характеристики цього генома: його розмір, GC-склад тощо. Також є номери доступу до двох плазмід, що входять до складу генома *S. coelicolor*, але не M145 (цей штам є безплазмідним варіантом розглянутого виду).

Подаємо ще одну таблицю, де перераховані всі проєкти секвенування геномів *S. coelicolor*, зокрема й ті, на яких базується повний опис генома у попередній таблиці.

У рубриці Genome Region (рис.1.14) можна знайти інформацію про повну нуклеотидну послідовність генома M145 у різних форматах. Можна переглянути FASTA-файл послідовності (команда FASTA), завантажити флет-файл типу CON (GenBank) або ж перейти до графічної сторінки генома (Graphics). За останнім гіперлінком міститься графічно насичена інтерактивна сторінка (рис. 1.15), яка дає змогу аналізувати різні ділянки генома: отримувати нуклеотидну послідовність окремих сегментів, анотацію генів тощо.

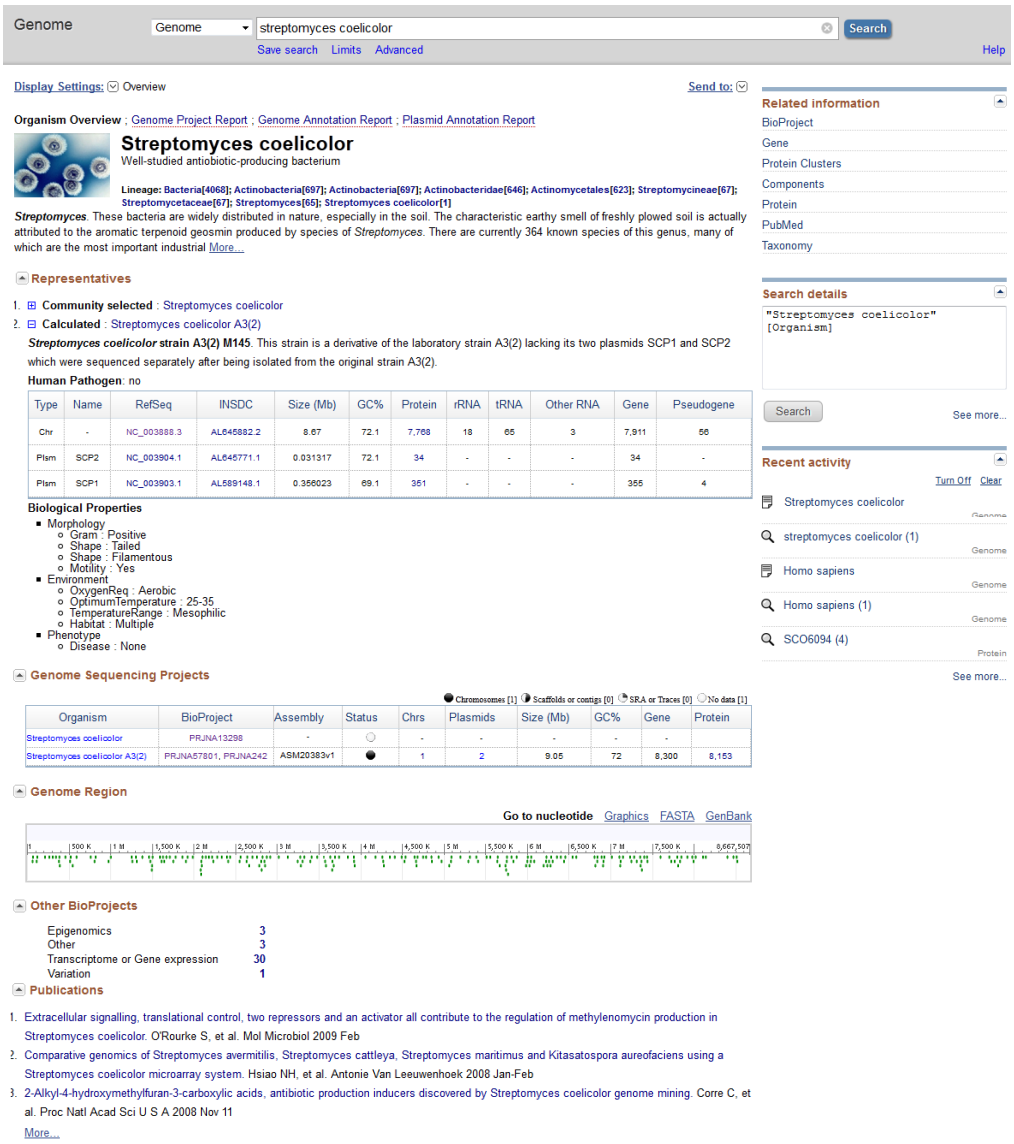


Рис. 1.14. Сторінка бази даних Genome з інформацією про геном *S. coelicolor*

Ця сторінка – типовий приклад *геномного переглядача* (genome viewer, genome browser), онлайн- чи офлайн-програми, яка дає змогу переглядати геноми і всі пов'язані з ним ознаки (анотації генів та інших генетичних елементів). Низка знарядь геномного переглядача (їх винесено на панель сторінки) дає змогу дослідникові обрати певну ділянку генома і вивчити його вміст. Наприклад, пересуваючи червоний прапорець уздовж "лінійки" генома, можна обрати те, що цікавить. Знаряддя Tools, ATG, Configure дають змогу деталізувати пошук до певної нуклеотидної ділянки і вивчити її властивості. Червоними прямокутниками зі стрілками-напрямами зображено гени, розміщені в обраній ділянці генома, їхні назви і номери доступу до їхніх послідовностей. Наприклад, у відповідь на наведення стрілки мишки на ген *SCO0217* програма відобразить лінк до послідовності гена (GeneID: 1095641 (SCO0217)). Або натисканням правої клавіші мишки при наведенні стрілки на назву гена можна отримати меню

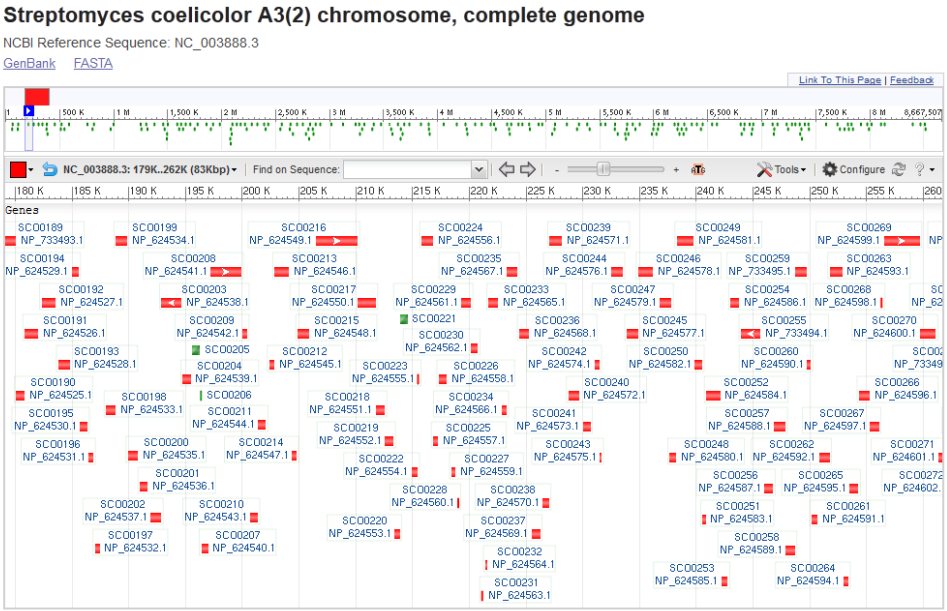


Рис. 1.15. Графічна сторінка бази даних Genome з інформацією про геном *S. coelicolor*

дій, які є змога виконати над ним. Так, у меню є команди FASTA View (рис. 1.16) для відображення нуклеотидної (номер доступу NC_003888.3, далі нуклеотидні координати) й амінокислотної послідовностей (NP_624534.1). Як і в багатьох інших онлайн-сервісах, у геномних переглядачах доступ до певного типу інформації можливий кількома різними шляхами. Ми розглянемо лише деякі з метою ілюстрування основних можливостей цього різновиду баз даних.

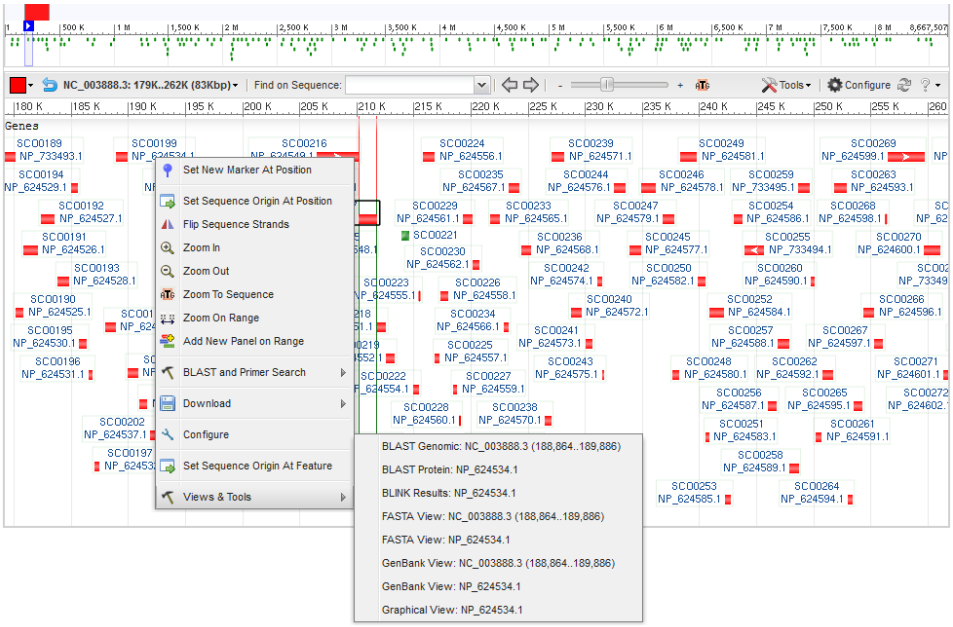


Рис. 1.16. Меню дій над об’єктами бази Genome

1.4. GEO Datasets. Цей ресурс NCBI (https://www.ncbi.nlm. nih.gov/gds) зберігає інформацію про транскриптомні дані – результати вимірювання рівнів транскрипції усіх або більшості генів у геномі. Такі вимірювання виконують методом гібридизації з мікроматрицями (microarray experiments) або (найчастіше) методами секвенування РНК (RNA-seq). Уведення назви виду у діалогове вікно скерує до списку експериментів чи масиву даних, у яких фігурує заданий вид. Наприклад, якщо ввести streptomyces albus й обрати перший експеримент, то внизу сторінки з'явиться перелік вихідних транскриптомних даних. У нашому випадку це будуть гіперпосилання, що розпочинаються з літер GSM. Після натискання на таке гіперпосилання отримаємо наприкінці нової сторінки гіперпосилання для завантаження даних у вигляді заархівованого txt-файла. У нашому випадку транскриптомний аналіз виконано методом РНК-секвенування, а файл фактично містить інформацію про те, скільки разів просеквенована мРНК певного гена. Ці значення, зазвичай, нормалізують щодо довжини гена і кількості отриманих якісних даних РНК-секвенування, отож величина рівня експресії гена не буде цілим числом. Більше про біоінформатичну складову аналізу RNA-seq даних можна знайти за лінком: https://doi.org/10.1186/s13059-016-0881-8. Чим вища концентрація мРНК певного гена у зразку, що досліджують, тим частіше (більшу кількість разів) цю мРНК секвенують. Наводимо невеликий фрагмент таких даних:

gene_start	gene_stop	gene_id	FPKM
3168543	3170384	XNRR2_00005a	1423.40052258782
3174010	3171596	XNRR2_00015a	127.015371559727
3174354	3175553	XNRR2_00020a	135.298982748405
3175753	3177681	XNRR2_00025a	204.329075987388
3177886	3179202	XNRR2_00030a	1082.39186198724
3179262	3179690	XNRR2_00035a	1851.38710066951
3179743	3180453	XNRR2_00040a	433.508985540809
3181163	3180492	XNRR2_00045a	31011.0790866804
3181317	3182993	XNRR2_00050a	27460.1710904672
3183320	3183090	XNRR2_00055a	429.36717994647
3184408	3183374	XNRR2_00060a	1.38060186477965
3185739	3184420	XNRR2_00065a	95.2615286697956
3186023	3187288	XNRR2_00070a	114.589954776711
3187285	3189147	XNRR2_00075a	28.9926391603726
3189185	3190144	XNRR2_00080a	128.395973424507

Кожен рядок містить інформацію про координати гена, рівень транскрипції якого визначали, назву гена і власне рівень транскрипції (в FPKM). Аналіз транскриптомних даних дає змогу деталізувати чи визначити структуру гена (координати транскрипта), часові рамки його функціонування, тканинну специфічність його експресії, що загалом допомагає краще зрозуміти його функцію. Більше про транскриптоміку див. тут: https://cutt.ly/aZsbBAR.

1.5. Спеціалізовані бази даних. NCBI – первинна база даних, найбільша цінність якої полягає у довготерміновому збереженні інформації про генетичні послідовності, бібліографічні дані тощо. Глибша і ширша анотація певного генома, а також опис суміжних генетичних ресурсів (геномні бібліотеки, колекції мутантів) не є метою первинних баз даних. Цьому присвячені спеціалізовані бази. Вони особливо цінні у випадку модельних об'єктів генетики, на яких

зосереджено роботу багатьох дослідницьких груп. Як приклад, розглянемо три такі бази. Перша база – StrepDB (http://strepdb.streptomyces.org.uk) – присвячена геномам стрептоміцетів, насамперед *S. coelicolor* M145; друга – FlyBase (http://flybase.org/) – база генів і геномів *Drosophila*; третя – MaizeGDB – описує генетику і геноміку кукурудзи; четверта – PATRIC – веб-портал аналізу геномів бактерій.

1.5.1. StrepDB. В основі цієї бази – повна нуклеотидна послідовність генома *Streptomyces coelicolor* M145, а також кількох інших стрептоміцетів – продуцента стрептоміцину *S. griseus* IFO13350, авермектину – *S. avermitilis* ATCC31267, клавуланової кислоти – *S. clavuligerus* ATCC27064, хлорамфеніколу – *S. venezuelae* ATCC10712, і збудника картопляної корости *S. scabiei 87.22.* На рис. 1.17 наводимо початкову сторінку StrepDB, на якій відображено ділянку генома *S. coelicolor.*

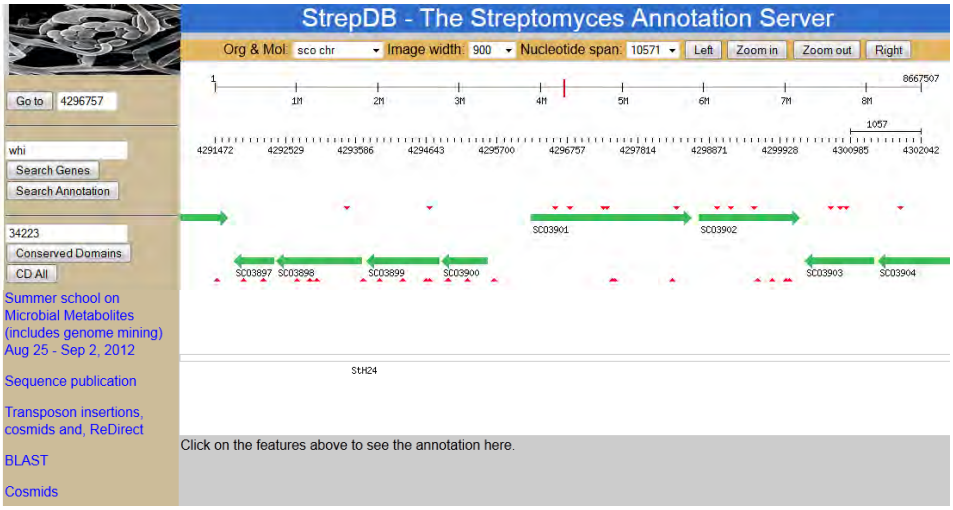


Рис. 1.17. Початкова сторінка StrepDB

Внизу, під заголовком бази даних (StrepDB – The Streptomyces Annotation Server), низка вікон, за допомогою яких можна обрати геном чи плазміду, яку планують досліджувати (Org&Mol), ширину зображення на моніторі комп'ютера (Image Width), проміжок генома, що відображається (Nucleotide Span). Справа від цих вікон розміщені декілька кнопок, які дають змогу рухатися по геному вправо чи вліво, а також збільшувати чи зменшувати певну ділянку генома на моніторі. У цьому прикладі для аналізу обрано хромосому *Streptomyces coelicolor* M145 (sco chr). Зліва, під зображенням повітряного міцелію, бачимо пошукові вікна, які дають змогу шукати певні гени штаму M145 за їхньою назвою чи анотацією, або ж шукати консервативні домени за їхніми номерами (перелік номерів подано у базі Conserved Domain Database, CDD: www.ncbi.nlm.nih.gov/cdd). Нижче розміщені гіперлінки, що дають змогу виконати попарне вирівнювання певної послідовності навпроти обраного генома (BLAST; розглядатимемо у наступному розділі), отримати перелік космід, що містять певний ген чи ділянку генома, та інших біологічних матеріалів. Зокрема, для *S. coelicolor* створена транспозонна бібліотека – колекція космід, що містять в ставки транспозона Tn5062 у геном

цього штаму. Сайти інсерцій транспозона точно визначено за допомогою секвенування. На графічній частині початкової сторінки StrepDB наявність транспозонної вставки у певну ділянку генома позначено червоним трикутником (див. рис. 1.17). Також можна шукати окремі гени чи групи генів за критерієм участі у певному метаболічному шляху (Search the genes by KEGG pathways).

Згідно з прийнятою номенклатурою, усі гени *S. coelicolor* позначені трилітерною абревіатурою SCO з позначенням номера гена. Загалом у хромосомі *S. coelicolor* наразі анотовано 7 846 генів: від *SCO0001* до *SCO7846*. Крім того, низку генів *S. coelicolor* названо ще до початку секвенування генома, і ці назви також можна використовувати у пошуку. Якщо у пошукове вікно Search Genes увести назву гена, наприклад *SCO2792* (або лише номер – 2792), то з'явиться виринаюче вікно з гіперлінком на нову сторінку, що містить інформацію про досліджуваний ген (рис. 1.18). Запис розпочинається у нижній частині вікна, де наведено назву гена *SCO2792* і (якщо такі є) його тривіальні назви – *adpA*, SCC105.23, *bldH*. Далі описують ген – координати початку і кінця, його анотацію,

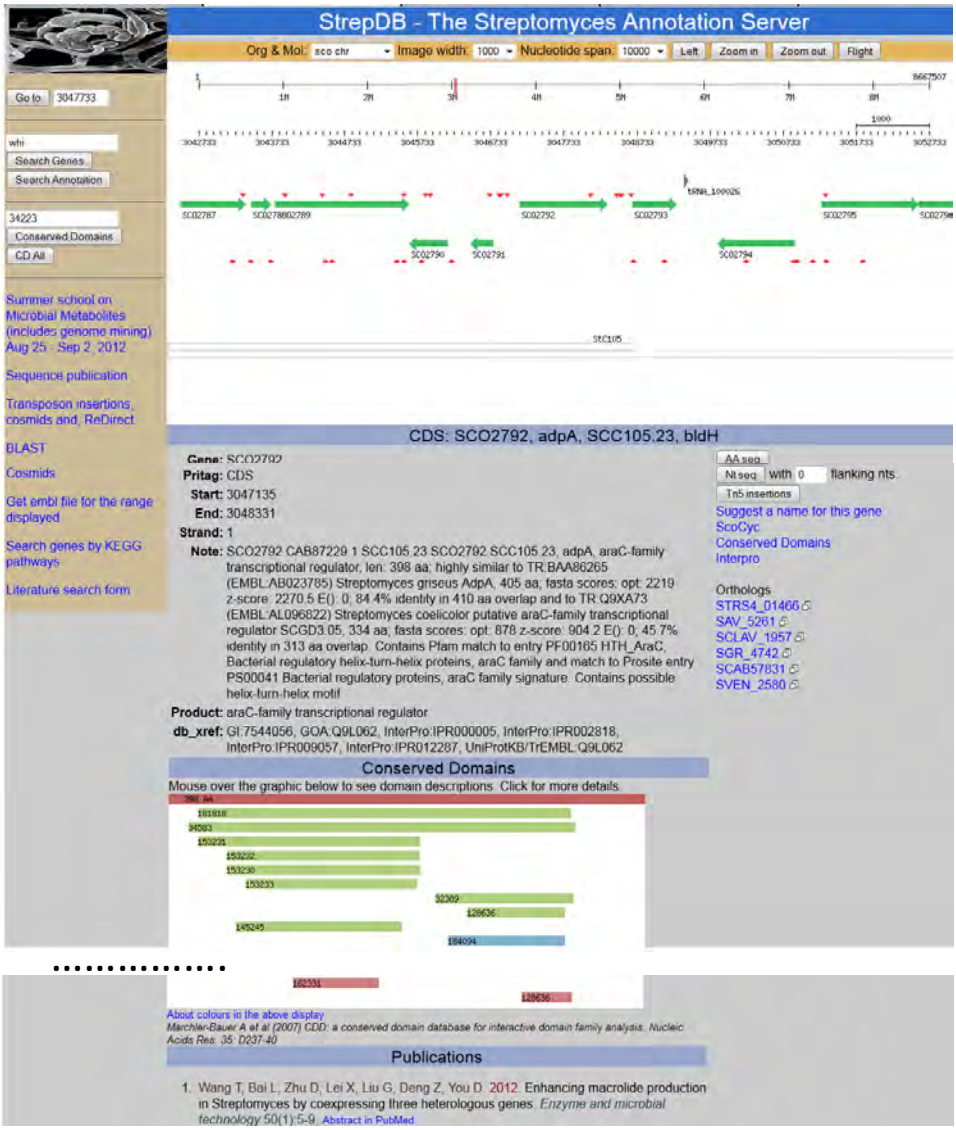


Рис. 1.18. Сторінка з інформацією про ген *SCO2792* (крапками позначено видалення частини рисунка для його спрощення)

розміщення консервативних доменів і публікації, де згадано *SCO2792*. Справа від опису є декілька кнопок-команд, які дають змогу отримати амінокислотну (AA seq) і нуклеотидну (Nt seq) послідовності *SCO2792*. Наприклад, можна отримати послідовність з певною кількістю нуклеотидів з обох кінців гена (with 0 flanking nt). Внизу описані ортологи білка, кодованого геном SCO2792, у базі StrepDB.

У графічній частині вікна програма відображає сегмент генома *S. coelicolor*, де міститься *SCO2792*. Гени позначені зеленими стрілками. Як бачимо, є одна вставка транспозона у межах кодувальної послідовності *SCO2792* (див. рис. 1.18). Доступ до опису відповідної транспозонної космiди можна отримати за допомогою натискання на зображення трикутника, або ж користаючись кнопкою-командою Tn5 insertions, розташованою справа від опису гена (див. рис. 1.18).

Якщо порівняти презентацію генома *S. coelicolor* у межах NCBI (база Genome, див. рис. 1.14–1.16) і StrepDB, то стає зрозумілим: StrepDB пропонує більший набір інформації, що пов'язана з цим організмом і геномом. Це можливість пошуку генів за анотацією, списки публікацій, де згадано певний ген, інформація про космiди, що містить певні ділянки генома тощо. Саме ці додаткові можливості і роблять спеціалізовані бази даних зручним знаряддям пошуку інформації про НАП певного організму.

1.5.2. FlyBase. Початкова сторінка цієї бази даних містить багато гіперлінків до різних підрозділів (рис. 1.19). Частина з них – стандартні знаряддя для попарного порівняння послідовностей, перегляду геномів і філогенії плодової мушки, бібліографічні дані.

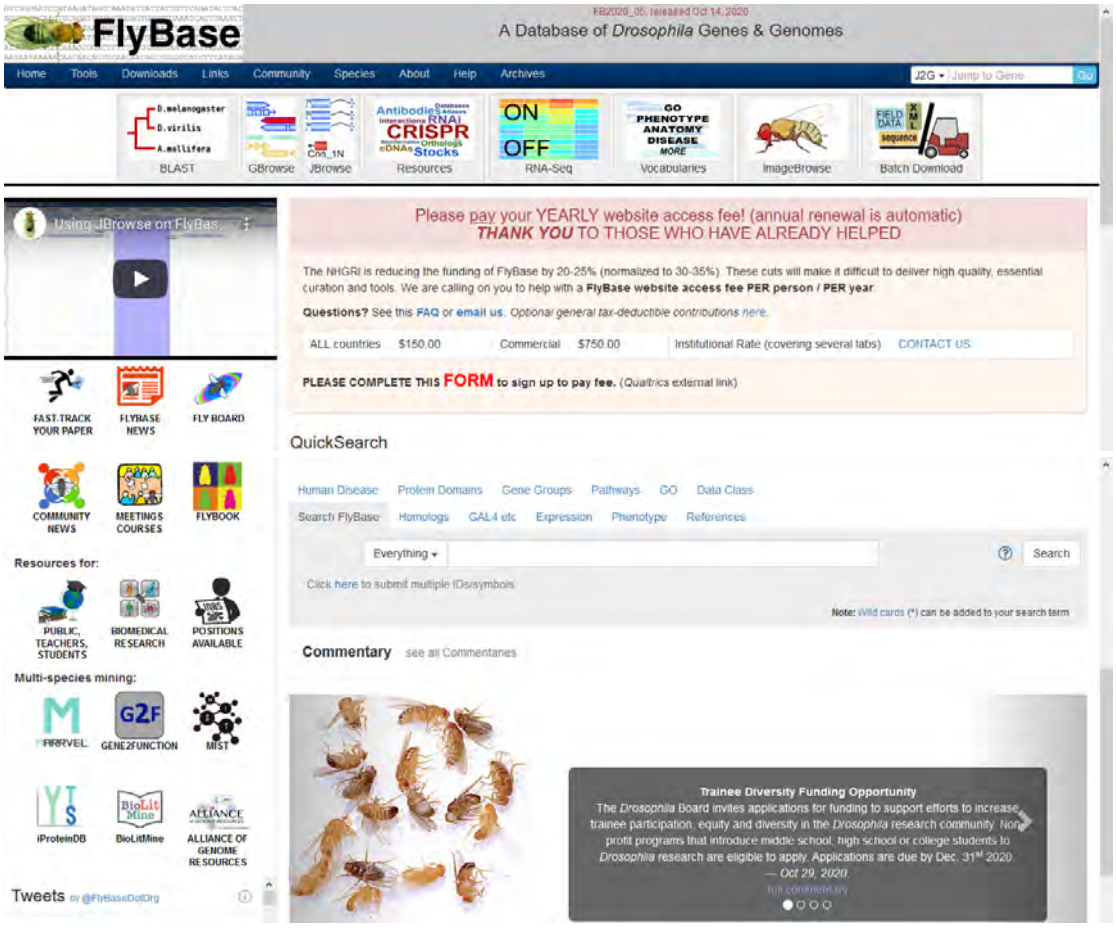


Рис. 1.19. Початкова сторінка бази даних FlyBase

Однак є низка специфічних знарядь. Наприклад, гіперлінк ImageBrowse переадресовує користувача на сторінку, де зібрані зображення різних частин тіла чи різних фізіологічних систем (травна, м'язова, нервова тощо). Іншою корисною рубрикою є Vocabularies. Цей гіперлінк відкриває сторінку, де можна знайти визначення різних термінів у галузі біології *Drosophila* і перелік різних об'єктів бази, що асоційовані з ним. Зверху у правому куті початкової сторінки бази (див. рис. 1.19) розміщене пошукове вікно, в яке можна ввести назву гена. Наприклад, у відповідь на пошук *DYS* програма відображає об'єкт – ген dystrophin (рис. 1.20) разом із загальною інформацією про ген (зокрема, доступ до його нуклеотидної послідовності).

Рис. 1.20. Верхня частина сторінки результатів з пошуку гена *DYS* (червоною стрілкою позначено термін Gene Model, описаний нижче в основному тексті)

У нижній частині сторінки результатів (рис. 1.21) подано інформацію про генетичну структуру цього гена, а саме: унаочнено позицію гена у хромосомі, відомі транскрипти, повтори, дані секвенування транскриптів (RNA-seq; остання частина рис. 1.21) і сайтів інсерцій трансгенів у обраний ген. Важливим елементом сучасної геноміки стосовно анотування генів є концепція *моделі гена* (gene model; див. рис. 1.20). Це сукупність усіх описів гена на рівні послідовності – його зрілих транскриптів (після процесингу або сплайсингу), а для білок-кодувальних генів – кодувальні послідовності і поліпептиди. Таке тлумачення особливо популярне у випадку складних еукаріотичних генів, і дає змогу постійно оновлювати й доповнювати інформацію про певну ділянку генома з появою нових доказів. Більшість цієї інформації є гіперлінком до нових сторінок, що містять додаткові дані – наприклад, анотацію послідовності чи її первинну структуру тощо. Для ілюстрування генетичних послідовностей і усіх асоційованих з ними даних база FlyBase використовує геномний переглядач GBrowse, відповідальний за графічний інтерфейс (рис. 1.21). Цей переглядач – поєднання бази даних та інтерактивних веб-сторінок, що дає змогу маніпулювати анотаціями геномів та описувати їх. GBrowse можна інтегрувати у будь-яку базу даних, покращуючи доступ до наявної у ній геномної інформації. Веб-адреса GBrowse – http://gmod.org/wiki/GBrowse. GBrowse – частина проєкту GMOD (Generic Model Organism Database project – http://gmod.org/wiki/Main_Page), який дає змогу створити колекцію програмних знарядь відкритого доступу для розроблення і керування біологічними базами даних геномного масштабу.

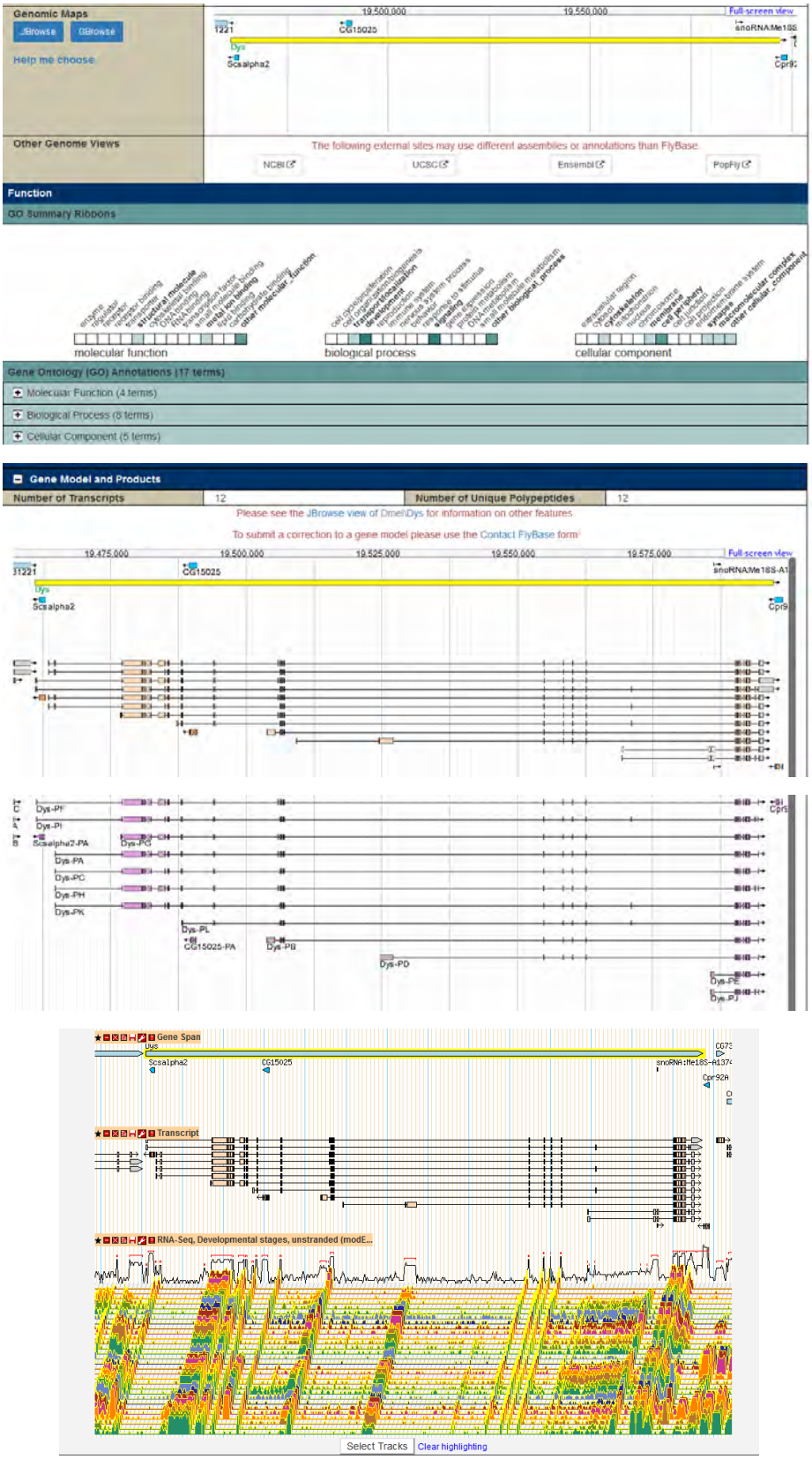


Рис. 1.21. Ген *DYS* і окремі елементи пов'язаної з ним інформації, яку відображає геномний переглядач GBrowse у межах бази даних Fly Base. Верхня панель: розташування гена в хромосомі і біологічна функція. Середня панель: інформація про відомі транскрипти гена *DYS* (за винятком рідкісних). Нижня панель – дані РНК-секвенування в районі локусу *DYS*. Треки різного кольору відповідають транскриптомним даним, отриманим на різних стадіях життєвого циклу плодової мушки

Важливим елементом FlyBase є інформація про наявні лінії плодової мушки і рекомбінантні молекули ДНК (вектори, плазміди, штучні хромосоми тощо), їхній опис і спосіб отримання. Тому цю базу можна вважати інтегрованою базою, місцем, де можна отримати всю інформацію, що стосується біології *Drosophila*.

1.5.3. MaizeGDB (https://www.maizegdb.org/). Біологічний об'єкт цієї бази даних – кукурудза (*Zea mays*), одна із ключових сільськогосподарських культур світу і водночас один із важливих модельних об'єктів генетики. Подібно до FlyBase, MaizeGDB є точкою зібрання всієї інформації, що стосується генетики і геноміки кукурудзи, а також подій у науковій спільноті, що вивчає цей організм. Як і в інших, розглянутих попередньо базах даних, в основі цієї – нуклеотидні послідовності геномів різних сортів кукурудзи. Гіперлінки до нових послідовностей геномів винесені як базові елементи сторінки (рис. 1.22). Важливим елементом усіх сучасних баз даних модельних об'єктів є доступ до транскриптомних даних, які допомагають точніше розуміти функцію гена. Так є і на сторінці MaizeGDB: доступ до даних РНК-секвенування відкрито у розділі qTeller.

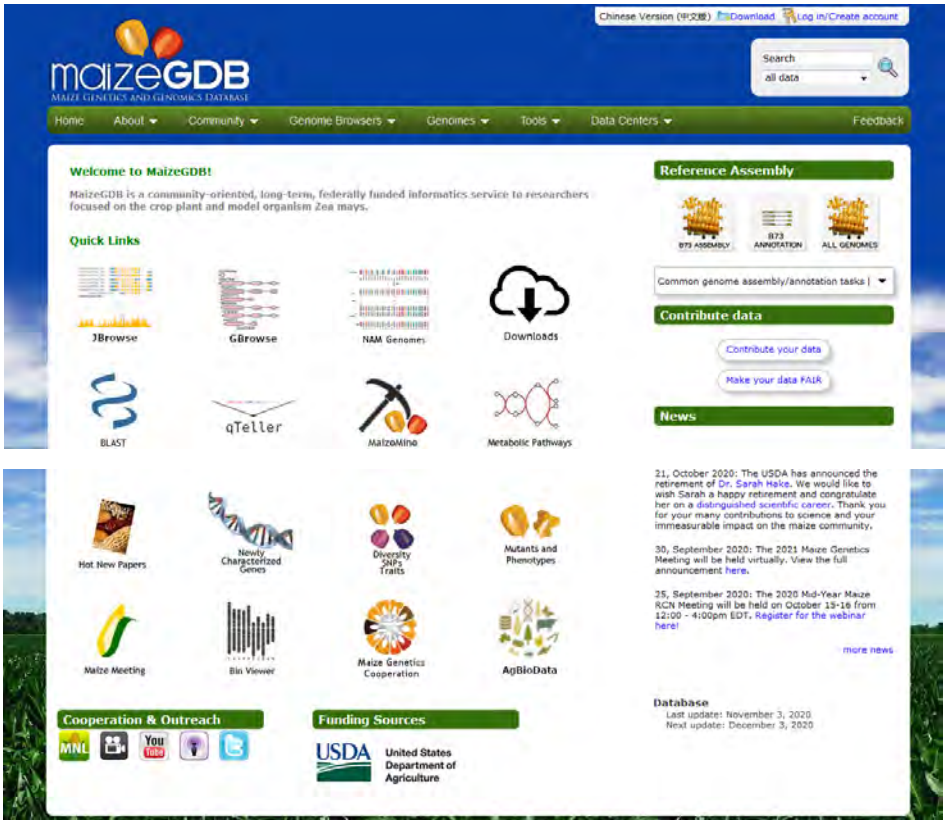


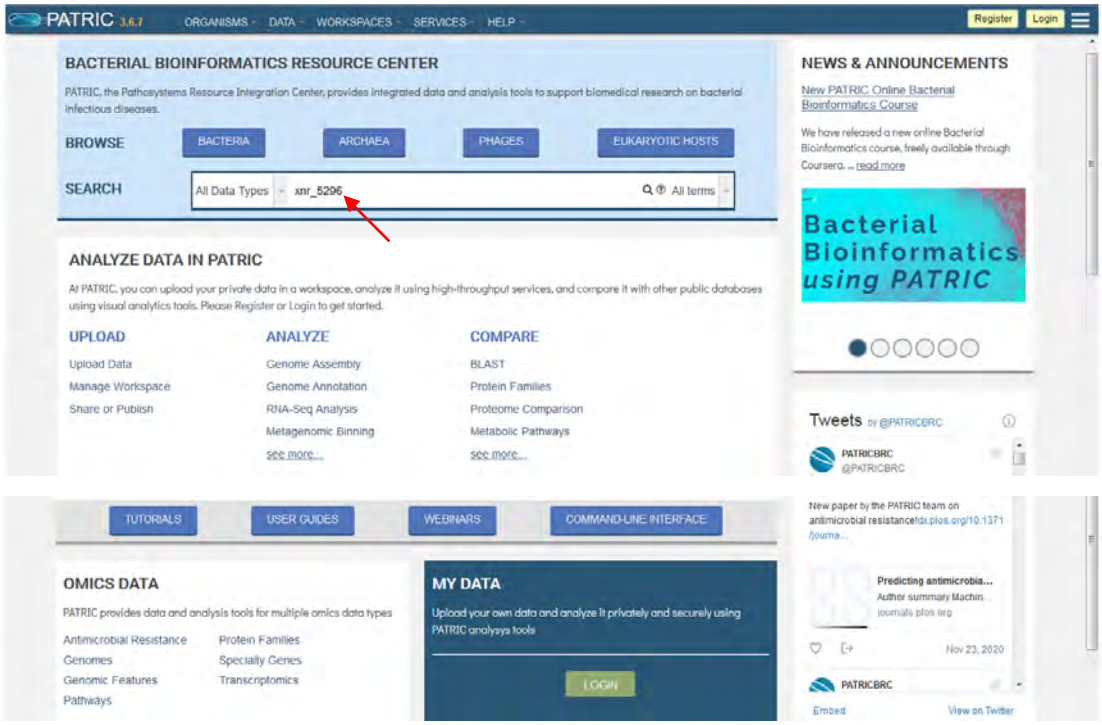
Рис. 1.22. Титульна сторінка бази даних MaizeGDB

На сторінці MaizeGDB можна знайти інформацію про поточні події, конференції у галузі генетики кукурудзи і рослин загалом. Бібліографічні дані курують фахівці бази. Отож база даних є авторитетним джерелом інформації у галузі генетики і геноміки цього модельного об'єкта.

1.5.4. PATRIC **(**PAThosystems Resource Integration Center; https://www.patricbrc.org/). PATRIC є онлайн-порталом для досліджень у галузі геноміки бактерій. Унікальність цього веб-ресурсу полягає в сталому фінансуванні з федерального бюджету США, що робить усі його сервіси (як і NCBI) доступними безкоштовно. Це також сприяє постійному оновленню їхньої бази геномів і доступу до всього спектра знарядь – від складання генома до побудови філогенетичних дерев. Відповідно, не полишаючи межі бази PATRIC, дослідник може: проаналізувати первинні дані секвенування і скласти з них геном; анотувати його (тобто ідентифікувати відкриті рамки зчитування і приписати їм функцію); виявити подібні геноми; дослідити ортологію і функцію генів/білків; створити набори даних (амінокислотних чи нуклеотидних послідовностей) на основі певних геномів. Крім аналізу геномних даних, цей веб-портал дає змогу аналізувати результати РНК-секвенування й сортувати метагеномні дані на окремі геноми (т. зв. “метагеномний біннінг”; metagenome binning). Частина сервісів PATRIC (ті, що потребують великих обчислювальних потужностей) вимагає реєстрації і створення власного кабінету. Користування базою PATRIC полегшує велика кількість допоміжних матеріалів, зокрема й окремий онлайн-курс з біоінформатики (орієнтований саме на PATRIC) на освітній платформі Coursera.

Якщо дослідник не використовує власні геномні дані, то вихідним матеріалом для аналізу на порталі PATRIC слугують геноми, депоновані в базі GenBank. Примітно, що PATRIC використовує свою систему анотації генів, і на сторінці результатів подаватиме як оригінальну анотацію (отриману з NCBI), так і власну. Ініціювати пошук у базі PATRIC можна завдяки введенню в діалогове вікно певного ключового слова. Це може бути назва певного роду бактерій чи окремого виду або ж назва гена. У відповідь база даних надає перелік гіперпосилань, за якими згадується заданий у пошуку об’єкт. Наприклад, задаючи в пошук назву гена *xnr_5296* (рис. 1.23), отримуємо як результат два гіперпосилання (рис. 1.23, *б*). Натискаючи на перше з них, отримуємо набір “веб-закладок”, за якими можна докладніше ознайомитися з різними типами даних, де згадано *xnr_5296* (рис. 1.24). Перша закладка подає загальну інформацію про шуканий ген – імовірний продукт гена, і належність до т.зв. “локальних” і “глобальних” родин білків за класифікацією бази PATRIC. Локальна родина білків об’єднує *гомологів* шуканого гена у геномах інших представників цього ж роду бактерій. Гомологічні гени чи білки – це гени чи білки, що мають спільне еволюційне походження. Наприклад, гемоглобіни людини, коня, крокодила і леггемоглобін люпину – гомологи, оскільки всі вони походять від одного предкового β-глобінового білка (більше про це йтиме мова у розділах, присвячених попарному і множинному вирівнюванням). Фактично локальні родини – це множина *ортологів* і *паралогів* гена у межах роду. Ортологи – це гомологи, споріднені у результаті видоутворення. Отже, якщо в геномі виду *Z1* є ген *z*, і вид *Z1* дав початок новому видові *Z2*, то в геномі *Z2* буде ген *z′*, що є ортологом *z*. Паралоги – це гомологи в одному геномі, що постали в результаті дуплікації. Наприклад, якщо ген *z* у геномі виду *Z1* подвоївся і дав початок генові *z**, то гени *z* і *z** – паралоги. Оскільки паралоги і ортологи мають різний характер еволюції і різне функціональне навантаження, то виявлення ортологів і паралогів має неабияке практичне значення. Використання сервісів на кшталт PATRIC спрощує пошук ортологів і паралогів. Більше про виявлення і застосування ортологів можна дізнатися за гіперпосиланням: https://doi.org/10.1093/molbev/msz150.

а



б

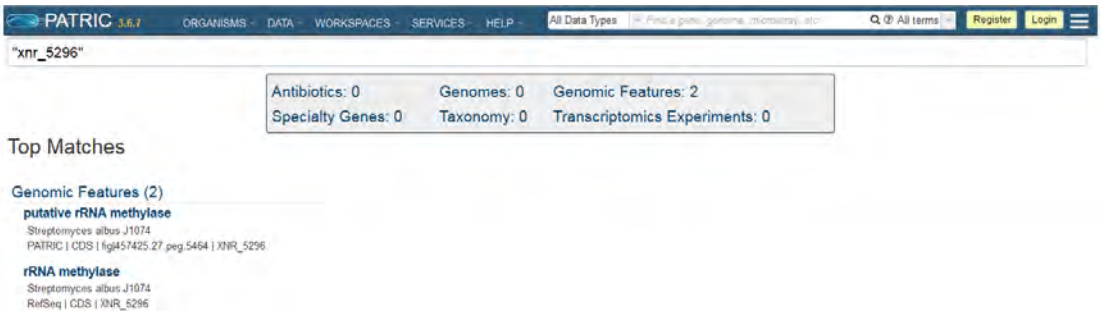


Рис. 1.23. Титульна сторінка бази даних PATRIC (*а*) з червоною стрілкою (позначає назву аналізованого гена) та результат бази PATRIC для гена *xnr_5296* (*б*)

Глобальна родина PATRIC охоплює гомологи, що є в геномах усіх бактерій.

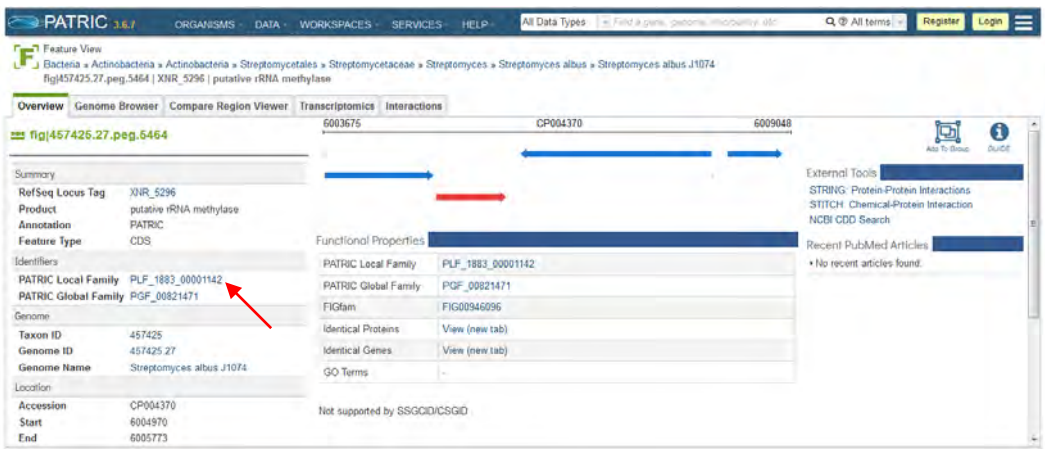


Рис. 1.24. Сторінка бази PATRIC з докладною інформацією про ген *xnr_5296* (червона стрілка позначає назву локальної родини, до якої належить ген *xnr_5296*)

Умовні позначення цих родин є гіперпосиланнями до відповідних списків генів-гомологів, які можна завантажити для подальшого аналізу. Корисну інформацію можна знайти у закладці Compare Region Viewer. Вона візуалізує порядок розташування генів у районі досліджуваного гена для обраної кількості геномів (рис. 1.25).

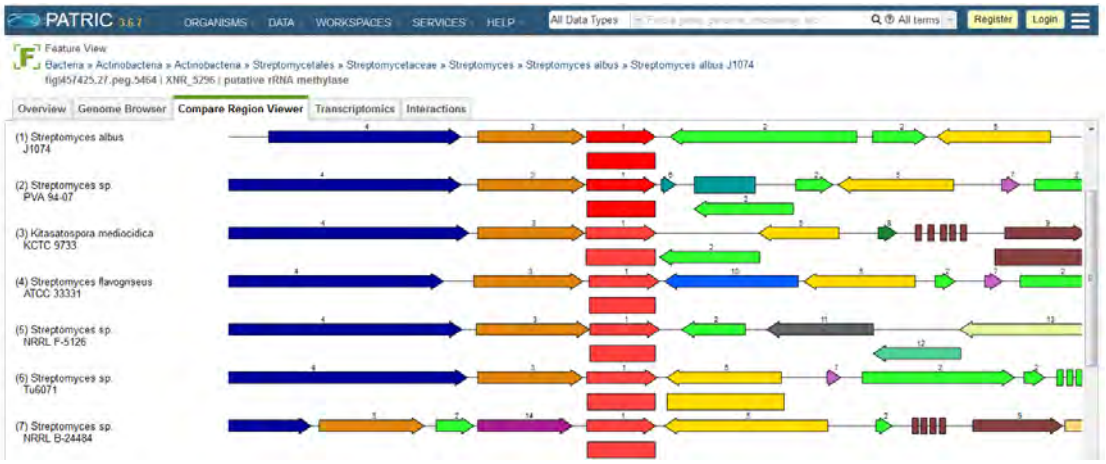


Рис. 1.25. Розташування і функція генів навколо гена *xnr_5296* і його гомологів (червона стрілка) з чотирьох інших стрептоміцетів (однаковим кольором позначено гени, що кодують білки з однаковою функцією)

Якщо функції і розташування (напрям транскрипції) генів, що оточують досліджуваний, збігаються для двох геномів, то кажемо, що ці ділянки геномів *синтенічні*. Синтенія може бути повною або частковою.

Контрольні запитання до розділу 1

1. Типи баз даних нуклеотидних та амінокислотних послідовностей.
2. Що таке інтегрована база даних?
3. Що таке архівна база даних?
4. Який тип інформації містить база даних PubMed?
5. Які цифрові ідентифікатори наукових статей Вам відомі?
6. Що таке цифровий ідентифікатор об’єкта?
7. Які операторні слова вжиті в базі PubMed?
8. Що таке формат FASTA?
9. Структура флет-файла бази GenBank.
10. Які відомості містить заголовок флет-файла?
11. Які типи генетичних послідовностей можна завантажити з бази даних GenBank?
12. Способи депонування генетичних послідовностей.
13. Як можна змінити координати генетичної послідовності, завантаженої зі сторінки GenBank?
14. Компоненти бібліографічного запису у базі PubMed.
15. Що таке геномний переглядач?
16. Який тип даних містить розділ NCBI Geo Datasets?
17. Про який модельний організм містить інформацію база даних FlyBase?
18. Для чого можна використати портал PATRIC?
19. Що таке гомологія?
20. Що таке ортологія?
21. Що таке паралогія?
22. Що таке синтенія?

РОЗДІЛ 2

Математичні моделі організації й еволюції генетичних послідовностей

Генетичні послідовності можна розглядати як мову, що базується на сталій кількості символів (літер), які разом становлять основу мови – абетку. Українська абетка налічує 33 літери, англійська – 26; послідовності ДНК і РНК складаються з п'яти нуклеотидних, а білки – двадцяти амінокислотних залишків. Далі на основі абетки формують слова – певні лінійні комбінації літер, що мають зміст, іншими словами – несуть функціональне навантаження. Спробуємо визначити кількість слів, які можна теоретично утворити з абетки, що налічує N літер. Ця вправа слугуватиме для читача своєрідним тестом на розуміння базових математичних операцій. Якщо розуміння немає або воно поверхове, то потрібно звернутися до спеціалізованих посібників, наведених у списку літератури наприкінці підручника. Які вихідні дані необхідно мати для таких обчислень? Очевидно, треба задати число N і максимальну довжину слова W. Нехай $W = 5$, а $N = 33$. Тоді теоретично можлива кількість 5-літерних слів дорівнюватиме N^W – і за розглянутих умов це приблизно 39 мільйонів (поміркуйте над тим, чому саме така формула описує кількість 5-літерних слів). Чим більше слово і/або чим більший розмір абетки, тим більше різноманітних слів можна утворити. Зауважмо також, що українською мовою слова можуть мати більше й менше 5-ти літер. Тому реалістичніше рахувати сумарну кількість слів (2-, 3-, 4-, 6-літерних і т. д.). Їхня кількість буде астрономічною і сягатиме мільярдів. Однак навіть найповніші словники української мови вміщують не більше 200 тисяч слів! Виконані обчислення є простим прикладом *моделювання* (у цьому випадку – теоретично можливої кількості слів). Описаний приклад засвідчує одну з особливостей будь-яких моделей – вони спрощують дійсний стан (наприклад, обрахунок лише п'ятилітерних слів й ігнорування інших варіантів). Прості моделі завжди можна ускладнювати, дедалі ближче сягаючи дійсності – за рахунок уведення нових параметрів (довжина слова чи розмір абетки) чи обмежень на утворення певних слів (наприклад, на слова, які є повторенням однієї літери). Зведення дійсності до певної спрощеної *моделі* – стандартний прийом у науці (дивіться, зокрема, попередній розділ, де згадано моделі гена у базі FlyBase). Він дає змогу дослідити певне складне явище, на яке впливає багато чинників, за допомогою "конструювання" такої ситуації (моделі), у якій це явище ставлять у залежність від одного чи кількох чинників, унаслідок чого можна зрозуміти їхню роль і значення. Результати описаного моделювання спонукають зробити два важливі висновки. По-перше, наразі тільки незначній частині "теоретично можливих слів" української мови притаманне таке поняття, як зміст. По-друге, будь-яка людська мова – як апарат збереження і передавання інформації – має великий потенціал до утворення нових слів, які, можливо, у майбутньому можна буде використати для позначення нових предметів, явищ, подій тощо. Отримана модель також породжує нові питання, з яких, мабуть, найважливіше таке: чи множина справжніх слів (що мають зміст у сучасній мові) володіє певними ознаками, які відрізняють її від усієї теоретично можливої

множини слів? Виявлення таких ознак дало б змогу, наприклад, передбачити нові слова, поява яких у майбутньому найімовірніша.

Усе вищесказане справедливе і для НАП. “Словами” НАП є гени і білки, і з безлічі комбінацій нуклеотидних чи амінокислотних залишків трапляється в природі лише невелика частка. Це результат молекулярної еволюції і природного добору тих НАП, які забезпечували оптимальну пристосованість їхніх носіїв (клітин, організмів) до навколишнього середовища. Здатність до дарвінівської еволюції – ключова ознака життя, поряд із його здатністю відтворюватися. Усі наявні дані засвідчують, що нові послідовності (гени) виникають унаслідок еволюції вже наявних; існує небагато добре задокументованих випадків виникнення нових генів *ab initio* із некодувальних послідовностей (стислий опис відомих прецедентів виникнення білок-кодувального гена *ab initio* можна відшукати за цим гіперпосиланням: https://doi.org/10.1038/d41586-019-03061-x). Отже, походження всіх сучасних генів і білків теоретично можна відстежити до невеликої кількості предкових послідовностей. Ця гіпотеза – наріжна неявна передумова багатьох методів біоінформатики.

Математичні моделі генетичних послідовностей – основа біоінформатики. Вони ґрунтуються на низці концепцій теорії інформації та імовірності, які спочатку стисло розглянемо. Основним постулатам теорії імовірності присвячений блок 1. У цьому розділі наголошено на розумінні якісної поведінки певної моделі, а не на точному числовому розв’язку.

2.1. Що таке інформація? У побутовому вжитку слово “інформація” має багато значень. Для потреб цієї книги *інформацію* можна визначити як зниження невизначеності. Іншими словами, інформація – це міра несподіванки. Уявімо, що ви опікуєтеся декількома дітьми, і з них одна на всі запитання відповідає ствердно. Вона дуже передбачувана, ви впевнені у кожній відповіді. У цьому разі немає несподіванки, немає інформації й спілкування (комунікації). Інша дитина відповідає по-різному на різні запитання – відтак відбувається певне спілкування, але воно було б обширніше, якби варіантів відповідей було більше. Якщо запитати: “Тобі подобається морозиво?” (яке подобається більшості дітей), то будете поінформовані, незалежно від типу відповіді, але несподіванішими будуть відповіді “ні”. Інакше кажучи, треба очікувати більше інформації при використанні більшого словника і від несподіваніших відповідей. Отже, інформація H, або міра несподіванки від відповіді, обернено пропорційна її ймовірності. Кількісним описувачем попереднього твердження є формула:

$$H(p) = \log_2 \frac{1}{p}; \qquad H(p) = -\log_2 p. \tag{1}$$

Відповідно до визначення, H є логарифмом з основою 2 від оберненої ймовірності. Інформацію H вимірюють у *бітах* (від англ. bit – скорочення слів binary та digit). Якщо ймовірність того, що дитині не подобається морозиво, становить 25 % (0,25), то така (заперечна) відповідь нестиме 2 біти інформації (любов до морозива – 0,41 біта).

БЛОК 1. Основні постулати теорії імовірності й генетичні послідовності

Правило додавання. Якщо дві події E та F взаємовиключні (не можуть трапитися одночасно), то імовірність події E або F є сумою імовірностей цих двох подій:

$$P(E \cup F) = P(E) + P(F), \text{якщо } E \text{ і } F - \text{неспряжені } (disjoint).$$

Наприклад, у певній позиції ДНК може бути або залишок піримідину ($P(E_{pyr})$), або пурину ($P(E_{pur})$). Сума цих двох імовірностей дорівнює 1 (оскільки у певній позиції точно буде або Pur, або Pyr, й інших варіантів немає), і тоді $P(E_{pyr}) = 1 - P(E_{pur})$.

Правило перемноження. Якщо дві події E та F незалежні (можуть трапитися одночасно; *joint*), то імовірність події E й F є добутком імовірностей цих двох подій:

$$P(E \cap F) = P(E) \times P(F), \text{якщо } E \text{ і } F - \text{незалежні.}$$

Наприклад, розглянемо гіпотетичну модель мутування ДНК. Нехай у кожному новому поколінні є 1,5 % імовірності, що у певній позиції послідовності відбудеться трансверзія (Pur⇄Pyr), яку далі позначимо як “зміна”. Отже, є 98,5 % імовірності того, що зміни не буде (або буде транзиція, яку в цій моделі розглядаємо як відсутність зміни). Тоді протягом одного покоління $P(E_{зміна})$ = 0,015, $P(E_{без\ змін})$ = 0,985. Тепер уявімо, що трапиться через два покоління. Є чотири варіанти, які мають таку хронологію: Пок. 1 (зміна/без змін)→ Пок. 2 (зміна/без змін).

Яка імовірність кожного з варіантів? По-перше, ми припускаємо, що події у першому і другому поколіннях незалежні. Це справедливе припущення, якщо розглянути мутації як випадкову помилку, оскільки ДНК не має “пам’яті” про минулі події. На підставі цього скористаємося правилом перемноження для встановлення комбінованої ймовірності незалежних подій: 1. $P(E_{зміна,\ зміна}) = 0{,}015 \times 0{,}015 = 0{,}000225$; 2. $P(E_{зміна,\ без\ змін}) = 0{,}015 \times 0{,}985 = 0{,}014775$; 3. $P(E_{без\ змін,\ зміна}) = 0{,}985 \times 0{,}015 = 0{,}014775$; 4. $P(E_{без\ змін,\ без\ змін}) = 0{,}985 \times 0{,}985 = 0{,}970225$. Поміркуйте, що є сумою цих чотирьох імовірностей? Чому ця сума має бути саме такою?

Тепер розглянемо таке питання: якою є ймовірність не виявити зміну в розглянутій позиції генетичної послідовності починаючи з покоління 0 і до покоління 2? Така подія, фактично, складається з двох подій: у жодному поколінні не було змін (1), або ж зміна відбулася у кожному поколінні (2), спричиняючи в кінцевому результаті відсутність зміни (наприклад, маємо справу з прямою і зворотною мутаціями: A→T→A). Оскільки ці події взаємовиключні, то потрібну імовірність обчислюємо так:

$$P(E_{без\ змін,\ без\ змін}) + P(E_{зміна,\ зміна}) = 0{,}970225 + 0{,}000225 = 0{,}970450.$$

Отже, ймовірність *не виявити зміну* через два покоління ($P(E_{без\ змін,\ без\ змін}) + P(E_{зміна,\ зміна})$) вища за ймовірність того, що через два покоління *фактично змін не відбулося* ($P(E_{без\ змін,\ без\ змін})$). Мутації, що виникли у перших поколіннях, можуть “скасовуватися” зворотними мутаціями у чергових поколіннях, і це впливає на ймовірність кінцевого результату (події), який фактично спостерігають.

Отже, правила додавання і перемноження дають змогу обчислювати ймовірності складних подій за рахунок розгляду елементарних подій, з яких вони складаються, і їхнього поєднання логічними операторами “або” (OR), “і” (AND) та “не” (NOT). OR означає, що потрібно додавати імовірності за умови, що події не поєднуються (disjoint). AND означає перемноження імовірностей (за умови їхньої незалежності). NOT означає обчислення комплементарної події, і її подальше віднімання від 1.

Загальноприйнято описувати інформацію як *символьне повідомлення* (message of symbols), що емітується з певного *джерела*. Наприклад, підкидання монети є джерелом повідомлення з двох символів – аверса (A; гербова сторона монети) і реверса (R; сторона, де викарбуваний номінал монети):

AARARARARRRARAAR

Аналогічно, числа 1, 2, 3, 4, 5, 6 є символами, які емітує шестигранна гральна кістка, ДНК як джерело "випромінює" чотири літери – A, T, G, C. Символи, емітовані джерелом, мають певний *розподіл частот* (frequency distribution) – імовірність трапляння (виникнення) символів у повідомленні. Якщо кількість символів n, а розподіл частот однаковий (для монети ймовірність випадання аверса чи реверса становить 50 %), то ймовірністю будь-якого певного символа є $1/n$. Звідси випливає, що інформація будь-якого символа $\log_2(n)$, і це значення також середнє. Формальна назва середньої інформації на символ – *ентропія*. Наприклад, для випадкової послідовності ДНК (у будь-якій позиції рівноймовірно натрапити на будь-який із чотирьох нуклеотидних залишків) ентропія однієї позиції становитиме $\log_2(4) = 2$, а для гіпотетичної випадкової амінокислотної послідовності – $\log_2(20) = 4{,}47$. Як щодо випадку, коли розподіл частот символів неоднаковий? Тоді для обчислення ентропії треба нормалізувати інформацію кожного символа відповідно до його частоти. Цей підхід кількісно описують за *формулою Шеннона*:

$$H = -\sum_{i=1}^{n} p_i \times \log_2 p_i . \qquad (2)$$

Наприклад, випадкова послідовність ДНК матиме ентропію -((0,25)×(-2) + + (0,25)×(-2) + (0,25)×(-2) + (0,25)×(-2)) = 2 біти.

Однак джерело ДНК, що емітує 90 % AT і 10 % GC, має ентропію -(2(0,45) × (-1,15) + 2(0,05) × (-4,32)) = 1,47 біта. У розглянутих прикладах розподіли частот символів були заздалегідь відомі. Такі параметри, однак, нечасто відомі *a priori*, і їх необхідно обчислювати на основі певного масиву експериментальних даних.

2.2. Ймовірність трапляння генетичної послідовності. Оцінення ймовірності того, що натрапимо на певну НАП у геномі чи протеомі – одне із найзагальніших і практичних завдань біоінформатики. Наприклад, дослідник за допомогою біохімічних експериментів ідентифікував *операторну* послідовність у промоторі певного гена розміром 10 п.н., яку розпізнає транскрипційний фактор родини AraC. Скільки таких операторних послідовностей можна очікувати знайти у геномі *Escherichia coli*? Відповідь на це питання можна шукати на основі двох моделей – *Бернуллі* та *Маркова*. Згідно з першою, поява символів (нуклеотидів, амінокислот) у послідовності не залежить одна від одної. У цій моделі частоти символів зафіксовані *a priori*, і вони не змінюються вздовж послідовності (немає залежності від контексту). Наприклад, GC-склад генома *E. coli* – 50 %, отож частоти всіх чотирьох нуклеотидних залишків – 0,25. У цьому (найпростішому) випадку ймовірність натрапити на певний октамер $P(S)$ – (GCGGATGC) – можна обчислити за формулою:

$$S=pL, \quad (3)$$

де p – частоти символів, L – довжина послідовності. Тут імовірність становитиме $(0{,}25)^8 = 1{,}52\times10^{-5}$. Іншими словами, на кожні 152 т.п.н. генома кишкової палички (або будь-якого генома з GC-вмістом 50 %) можна випадково знайти одну октамерну послідовність визначеної вище структури. Якщо частоти символів неоднакові, то формула (3) набуває такого вигляду:

$$P(S) = \prod_{i=1}^{L} P(r_i). \quad (4)$$

Наприклад, у геномі пекарських дріжджів *Saccharomyces cerevisiae* P(A) = P(T) = = 0,325; P(G) = P(C) = 0,175. Тоді ймовірність натрапити на вищезгаданий октамер (GCGGATGC) у геномі *S. cerevisiae* становитиме: $(0{,}175^6)\times(0{,}325^2) = 3\times10^{-6}$. Помітно, що у геномі дріжджів такий октамер трапляється не так часто, як у геномі кишкової палички.

Другий спосіб обчислення частоти трапляння генетичної послідовності базується на моделі Маркова. *Модель Маркова* – це ймовірнісна (статистична) концепція, яку можна застосовувати у разі опису зміни (переходів) *станів системи*, за умови, що кількість таких станів – скінченна. Найпростішою моделлю є ланцюг Маркова. У ньому ймовірність появи символа (нуклеотидного/амінокислотного залишків) залежить від m попередніх символів. Число m називають *порядком* ланцюга Маркова. Ланцюг Маркова нульового порядку описує систему, в якій імовірність появи символа залежить від 0 попередніх символів, тобто це модель Бернуллі.

Розглянемо ланцюг Маркова докладніше. Нехай маємо систему, що складається з чотирьох станів: T, A, G, C; усі чотири стани рівноймовірні. Біологічний еквівалент такої системи – нуклеотидна послідовність, у якій частоти трапляння всіх чотирьох нуклеотидних залишків становлять 25 %, або 0,25. Ця система генерує лінійну (нерозгалужену) послідовність станів – *ланцюг*, який визначають за ймовірністю переходів від одного стану до іншого. Наприклад, перехід від стану T (S_T) до A (S_A) визначають за ймовірністю переходу P_{TA}, що може залежати від m попередніх станів (рис. 2.1). У найпростішому випадку $m = 0$, тоді ймовірність переходу до нового стану (чи повторення наявного – P_{TT}) не залежить від попередніх. Саме таку систему зображено на рис. 2.1. Її ще називають моделлю Маркова нульового порядку (ймовірність переходу до певного стану залежить від 0 попередніх станів). Параметр m – порядок моделі Маркова – визначає залежність імовірності станових переходів від попередніх станів, але не вказує на ймовірність трапляння (частоти) станів. Наприклад, якщо геном *Escherichia coli* розглянути з позицій моделі Маркова, то його чотири стани (нуклеотидні залишки) трапляються приблизно з однаковою частотою, а стани в геномі пекарських дріжджів *Saccharomyces cerevisiae* – з різною (табл. 2.1).

Оскільки моделі Маркова статистичні, то сума всіх можливих переходів з певного стану в інші має становити 100 %, або 1. У випадку розглянутої нульової моделі можливі чотири переходи зі стану T до станів C, G, A, а також повторення стану T (рис. 2.1). Відповідно до моделі, усі переходи рівноймовірні і мають

становити 100 %, тобто ймовірність будь-якого з чотирьох переходів становитиме 25 %, або 0,25.

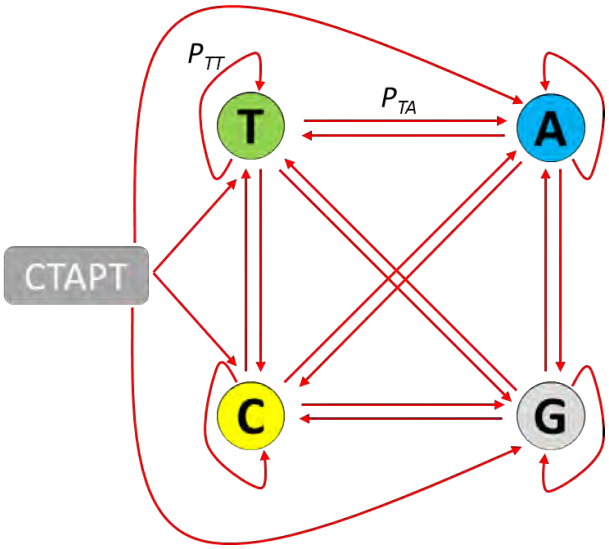


Рис. 2.1. Схематичне зображення системи, що складається з чотирьох станів T, A, G, C і генерує ланцюг Маркова – лінійну послідовність станів (ймовірність усіх зображених переходів (червоні стрілки) однакова, як і ймовірність того, що ланцюг може розпочатися будь-яким із чотирьох станів)

Ця ключова особливість імовірнісних моделей відрізняє їх від інших моделей, де можна змінювати ваги чи рахунки одних станів, не порушуючи інші. В моделях Маркова, змінюючи величину ймовірності одного переходу, треба змінювати інші величини, щоб задовольнити умову сталого значення суми всіх переходів.

Таблиця 2.1

Частоти трапляння нуклеотидів у вибраних геномах

Геном	A	C	G	T
Escherichia coli	0,246	0,254	0,254	0,246
Saccharomyces cerevisiae	0,309	0,191	0,191	0,309

Застосуємо тепер принципи нульової моделі до послідовностей двох описаних геномів. Тоді ймовірність переходу з будь-якого стану однакова: $P(\mathrm{T}) = P(\mathrm{A}) = P(\mathrm{G}) = P(\mathrm{C}) = 0{,}25$. Можна оцінити імовірність генерування певного ланцюга станів (для нуклеотидних послідовностей – це послідовність азотистих основ) за заданих параметрів моделі. Наприклад, імовірність $P(S)$ того, що ця модель генеруватиме ланцюг TCTGCG (наприклад, гексануклеотидну

послідовність у геномі кишкової палички), дорівнюватиме добутку частот трапляння кожного стану, або значенню ймовірності p, в ступені, що дорівнює довжині генерованого ланцюга Маркова L:

$$P(S) = p^L; \quad 0{,}25^6 = 2{,}4^{-4}.$$

Обчислимо тепер імовірність появи такої ж послідовності (TCTGCG) у геномі пекарських дріжджів за умови незалежного розподілу станів. Чотири стани (нуклеотиди) у геномі *S. cerevisiae* нерівноімовірні (див. табл. 2.1). Тоді $P(S)$ становитиме:

$$P(S) = 0{,}309^2 \times 0{,}191^4 = 1{,}3^{-4}.$$

Отже, за нульової моделі імовірність натрапити на цю послідовність у геномі дріжджів менша, ніж у геномі *E. coli*. Цей висновок інтуїтивно правильний, якщо брати до уваги частоти нуклеотидів у цих двох геномах. В *S. cerevisiae* домінують A і T (див. табл. 2.1), тому GC-багата послідовність матиме менше шансів на появу.

Будь-яку модель Маркова можна зобразити графічно (див. рис. 2.1), або у вигляді матриці *ймовірностей переходів* (transition frequencies) станів. У випадку нульової моделі така матриця матиме найпростішу будову, як зображено на рис. 2.2.

	T	A	G	C
T	0,25	0,25	0,25	0,25
A	0,25	0,25	0,25	0,25
G	0,25	0,25	0,25	0,25
C	0,25	0,25	0,25	0,25

Рис. 2.2. Матриця переходів у нульовій моделі Маркова, що налічує чотири стани

Отже, в моделях Маркова нульового порядку ймовірності станів (імовірності появи залишків нуклеотидів у послідовності) зафіксовані *a priori* – це т.зв. *апріорні* (prior) імовірності. У випадку геномних послідовностей, наприклад, їх виводять з нуклеотидного складу цілих геномів і підсумовують у таблицях (див. табл. 2.1). Іншою обов'язковою умовою є незалежність переходів, що не змінюється вздовж усього ланцюга Маркова.

У моделі Маркова вищого (ненульового) порядку ймовірність знайти певну літеру r_i (нуклеотидний/амінокислотний залишок – стан) у позиції ланцюга i залежить від літер, що знаходяться у m попередніх позиціях:

$$P\left(r_i \middle| S_{i-m,i-1}\right). \qquad (5)$$

Іншими словами, у моделі першого порядку ймовірність знайти певний нуклеотидний залишок r_i у позиції i залежить від одного залишку, розміщеного безпосередньо перед r_i. Приклади моделей першого (m = 1) та другого (m = 2) порядків у вигляді матриці ймовірностей переходів наведено на рис. 2.3. У цих матрицях колонки репрезентують суфікс (r_i), ряди – префікс ($S_{i-m,\ i-1}$). Використовуючи ці терміни, можна сказати, що ймовірність натрапити на суфікс у певній позиції послідовності є функцією відповідного префікса.

Модель Маркова ставить появу певного символа у послідовності в залежність від попередніх символів. Цю залежність формально описують *кондиційною ймовірністю* (conditional probability), яку записують так само, як у формулі (5): $P(A|C)$ – ймовірність появи A за умови C. Обчислення різноманітних імовірностей (зокрема, умовних) відіграє важливу роль в аналізі організування й еволюції генетичних послідовностей. Тому теорія ймовірності є неодмінним супутником більшості розділів цієї книги. Основні елементи теореми Баєза (Bayes theorem) із застосуванням прикладу з НАП докладніше розглянуто на двох прикладах у блоці 2. Тут лише зазначимо, що *баєзіанські підходи* мають в основі іншу філософію, ніж класичні *фриквентистські* (frequentist) підходи. “Фриквентист” визначає ймовірність події – наприклад, ймовірність витягти чорну кульку з величезної (бажано нескінченної) купи, що складається з рівних кількостей рівномірно перемішаних чорних і білих кульок – як очікувану *частоту трапляння* такої події після численних спроб. Такий підхід найпростіше зрозуміти, коли альтернативні варіанти розвитку подій рівноймовірні, як у згаданому прикладі, і логіка обчислення ймовірності апелює до симетрії частоти трапляння подій. Отже, якщо в одному людському геномі після одного раунду реплікації виявляють одну мутацію, тоді ймовірність мутації буде $1/(3\times10^9)$ на одне покоління для кожного з трьох мільярдів нуклеотидів генома людини. У фриквентистських підходах фігурують такі поняття, як варіанса, інтервал достовірності і p, і методи максимальної вірогідності. Втім дійсність зрідка буває простою. Наприклад, відомо, що частоти трапляння мутацій варіюють для різних ділянок генома. Часто нас цікавлять ймовірності подій, що не є частиною будь-якого набору симетричних сценаріїв, як-от ймовірність поширення певного хвороботворного алеля гена. Для такого випадку складно (або й неможливо) створити контрольовану вибірку і виконати серію спроб за ідентичних умов. За реальних умов ми змушені вибирати з кількох альтернативних гіпотез на основі одного (і часто малого) набору даних. Баєзіанський підхід дає змогу розглядати не лише ймовірності подій, але й гіпотез. У цьому ключі можна вважати, що ймовірність виражає *міру нашої віри* у певну гіпотезу чи подію. Тоді певний сценарій розвитку подій є параметром, а ймовірність відображає невизначеність цього параметра. В класичному розумінні параметри є невідомими константами до його визначення (наприклад, середній зріст людини на основі вимірювання зросту всіх людей), що не можуть мати розподілу ймовірностей. На думку баєзіанців, оскільки величина параметра невідома, то резонно визначити розподіл ймовірностей для нього, щоб описати його можливі величини. Можна поставити питання про ймовірність виникнення інтронів з транспозонних елементів, і двом альтернативним відповідям присвоїти нашу міру віри (ймовірність) у кожну з них. Це буде т. зв. апріорна ймовірність події (появи інтронів). Теорема Баєза дає змогу обчислити апостеріорну (кондиційну) ймовірність параметра за умови певного набору даних. Усі припущення про параметр далі будують на апостеріорних даних.

БЛОК 2. Теорема Баєза: не зовсім чесне казино і генетичні послідовності

Уявімо собі нечесне казино, в якому 99 % гральних кубиків – чесні (K_{fair}), а 1 % "нечесні"(K_{loaded}) – їхній центр ваги зміщено на одну грань так, що число 3 випадає у половині випадків. Це *апріорні* дані (priors). Кубики для гри вибирають випадково. Нехай кубик тричі поспіль випав трійкою догори. Імовірно, це – нечесний кубик. Як оцінити ймовірність такої події? Те, що потрібно оцінити – це *ймовірність* того, що кубик нечесний P (K_{loaded}) *за умови*, що він випав тричі поспіль трійкою догори – $P(K_{loaded}|3\ \text{threes})$. Тобто обчислюють *апостеріорну ймовірність* (posterior probability) того, що кубик нечесний за наявних даних. Однак прямо можна обчислити лише ймовірність даних за заданої гіпотези $P(3\ \text{threes}|\ K_{loaded}|)$, яку називають *вірогідністю* (likelihood) гіпотези. Апостеріорні ймовірності обчислюють за *теоремою Баєза*:

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)} \quad \text{або} \quad P(H|D) = \frac{P(H \cap D)}{P(D)}.$$

Таке рівняння дає змогу обчислити ймовірність гіпотези H (події), що "кубик нечесний" за умови фактичних даних (D) – тут це "три трійки". Тоді:

$$P(K_{loaded}|3\ threes) = \frac{P(3\ threes|K_{loaded})P(K_{loaded})}{P(3\ threes)}.$$

За умов задачі, ймовірність $P(K_{loaded})$ вибрати нечесний кубик становить 0,01, а ймовірність $P(3\ threes|K_{loaded})$ того, що випадуть три трійки поспіль за умови, що це нечесний кубик – $0{,}5^3 = 0{,}125$. Сумарна ймовірність того, що випадуть три трійки:

$$P(3\ threes) = P(3\ threes|K_{loaded})P(K_{loaded}) + P\big(3\ threes\big|K_{fair}\big)\big(K_{fair}\big),$$

звідси

$$P(K_{loaded}|3\ threes) = \frac{0{,}5^3\ \times 0{,}01}{0{,}5^3\ \times 0{,}01 + \frac{1}{6}^3\ \times 0{,}99} = 0{,}21.$$

Отже, три трійки поспіль не свідчать на користь припущення, що кубик – нечесний.

Тепер припустимо, що *позаклітинні білки* (ПБ) мають інший амінокислотний склад, ніж *внутрішньоклітинні білки* (ВБ), наприклад, цистеїн частіше трапляється у ПБ, ніж у ВБ. За цією інформацією спробуємо оцінити клас (ПБ чи ВБ), до якого належить нова послідовність $x = x_1 \ldots x_n$. Для цього треба мати *тренувальний набір даних* – масив послідовностей, на основі якого можна встановити вихідні параметри моделі. Нехай це будуть білки з бази Swiss-Prot, які можна розділити на ВБ і ПБ. Звідси можна визначити частоти появи усіх амінокислотних залишків q_a^{int} для ВБ і q_a^{ext} для ПБ. Треба також знати ймовірність появи ПБ (p^{ext}) і ймовірність ВБ (p^{int}). Припускаємо, що будь-яка послідовність цілком є ПБ або ВБ, отож $p^{int} = 1 - p^{ext}$. Значення p^{ext} й p^{int} – апріорні ймовірності, оскільки вони є, фактично, припущенням (імовірно, найточнішим) про послідовність *до того*, як ми бачимо її саму. Тепер можна записати:

$$P(x|\text{ПБ}) = \prod_i q_{x_i}^{ext} \ \text{і} \ P(x|\text{ВБ}) = \prod_i q_{x_i}^{int}.$$

Оскільки є лише два варіанти – всі білки можуть бути або ПБ, або ВБ, то:

$$P(x) = p^{ext}P(x|\text{ПБ}) + p^{int}P(x|\text{ВБ}).$$

Тоді, згідно з теоремою Баєза, маємо:

$$P(\text{ПБ}|x) = \frac{\prod_i q_{x_i}^{ext}\ p^{ext}}{p^{ext}\prod_i q_{x_i}^{ext} + p^{int}\prod_i q_{x_i}^{int}},$$

де $P(\text{ПБ}|x)$ – шукане число.

Наведено *апостеріорну ймовірність* того, що послідовність x є ПБ, оскільки це найточніше припущення *після того*, як ураховані актуальні дані ($x_1 \ldots x_n$). Теорема Баєза дає змогу розв'язувати задачі, в яких немає повної інформації про систему.

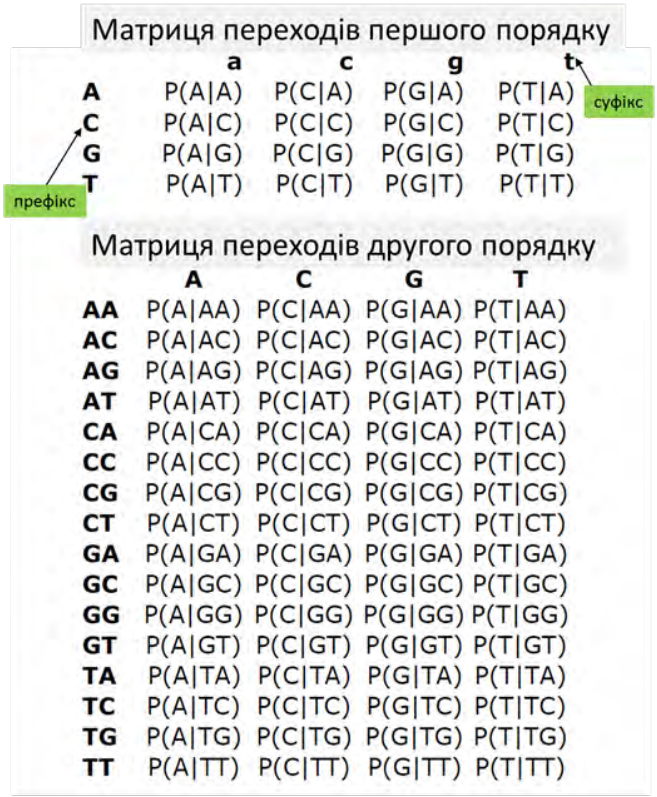


Рис. 2.3. Матриці-схеми, що описують ланцюги Маркова першого і другого порядків

Дані про ймовірність переходів для моделі Маркова *m*-го порядку можна отримати під час аналізу ланцюгів станів довжиною $k = m + 1$. Наприклад, можна визначити частоти появи усіх можливих динуклеотидів ($k = 2$) у міжгенних ділянках генома *S. cerevisiae* (табл. 2.2). Ці дані можна використати для побудови моделі Маркова першого порядку (m=k-1=1). Матрицю переходів для цієї моделі обчислюють за такою формулою:

$$P(r_i|S_{1\ldots m}) = \frac{F_{bg}(r_i|S_{1\ldots m})}{\sum_{j\in A} F_{bg}(r_j|S_{1\ldots m})} = \frac{F_{bg}(S_{1\ldots m}r_i)}{\sum_{j\in A} F_{bg}(S_{1\ldots m}r_j)}. \qquad (6)$$

Частоту переходу від T до G (P(G|T) обчислюють так:

$$P(\mathrm{G|T}) = \frac{F(\mathrm{G|T})}{\sum_{j\in A} F(j|\mathrm{T})} = \frac{F(\mathrm{TG})}{\sum F(\mathrm{T}^*)} = \frac{0{,}060}{0{,}079 + 0{,}060 + 0{,}060 + 0{,}111} = 0{,}194.$$

де T* – усі можливі комбінації двох нуклеотидів, що розпочинаються з T.

Таблиця 2.2

Частоти трапляння динуклеотидів у міжгенних ділянках *S. cerevisiae*

Послідовність S	Кількість *N(S)*	Частота *F(S)*
AA	526 149	0,112
AC	251 377	0,054
AG	275 056	0,059
AT	414 453	0,088
CA	294 423	0,063
CC	178 324	0,038
CG	146 052	0,031
CT	275 859	0,059
GA	277 343	0,059
GC	184 367	0,039
GG	173 404	0,037
GT	239 569	0,051
TA	369 980	0,079
TC	280 475	0,060
TG	279 932	0,060
TT	521 236	0,111

Так само обчислюють усі інші частоти переходів, які узагальнюють у матриці переходів першого порядку (табл. 2.3).

Таблиця 2.3

Матриця переходів першого порядку, міжгенні послідовності *S. cerevisiae*

Pr \ Su	A	C	G	T	P(Pr)	N(Pr)
A	0,359	0,171	0,187	0,283	0,313	1,4e5
C	0,329	0,199	0,163	0,308	0,191	8,9e5
G	0,317	0,211	0,198	0,274	0,187	8,7e5
T	0,255	0,193	0,194	0,359	0,310	1,4e5
P(Su)	0,313	0,191	0,187	0,310		
N(Su)	1,4e6	8,9e5	8,7e5	1,4e6		

Умовні позначення: Pr – префікс; Su – суфікс; P(Pr), P(Su) – імовірність префікса/суфікса; N(Pr), N(Su) – кількість Pr/Su у тренувальному наборі послідовностей.

Зважаючи на проаналізовані принципи, стає зрозуміло, що частоти появи (апріорні ймовірності) окремих нуклеотидних залишків у геномах *E. coli* і *S. cerevisiae*, які наведено у табл. 2.1, фактично, репрезентують матрицю станових переходів для нульової моделі Маркова.

На основі матриці 2-го порядку *B* (табл. 2.4), обчислимо ймовірність появи у геномі *S. cerevisiae* такої 20-п.н. послідовності *S* – CCTACTATATGCCCAGAATT – *P*(*S*|*B*). Модель *B* обчислено на основі даних про частоти тринуклеотидних послідовностей з міжгенних ділянок генома *S. cerevisiae*.

Таблиця 2.4

Матриця переходів другого порядку (*B*)

	A	C	G	T	P(Pr)	N(Pr)
AA	0,388	0,161	0,200	0,251	0,112	5,2e5
AC	0,339	0,198	0,173	0,290	0,054	2,5e5
AG	0,345	0,204	0,196	0,255	0,059	2,7e5
AT	0,311	0,184	0,182	0,323	0,088	4,1e5
CA	0,347	0,178	0,189	0,286	0,063	2,9e5
CC	0,341	0,190	0,161	0,309	0,038	1,8e5
CG	0,293	0,221	0,196	0,290	0,031	1,4e5
CT	0,229	0,195	0,205	0,371	0,059	2,7e5
GA	0,394	0,155	0,187	0,264	0,059	2,7e5
GC	0,330	0,205	0,169	0,297	0,039	1,8e5
GG	0,318	0,217	0,187	0,277	0,037	1,7e5
GT	0,285	0,175	0,204	0,336	0,051	2,4e5
TA	0,300	0,193	0,168	0,339	0,079	3,7e5
TC	0,313	0,203	0,152	0,332	0,060	2,8e5
TG	0,302	0,209	0,208	0,282	0,060	2,8e5
TT	0,210	0,208	0,189	0,392	0,111	5,2e5
P(Su)	0,313	0,191	0,187	0,310		
N(Su)	1,4e6	8,9e5	8,7e5	1,4e6		

За цієї моделі *P*(*S*|*B*) можна обчислити за формулою:

$$P(S|B) = P(S_{1,m}|B) \prod_{i=m+1}^{L} P(r_i|S_{i-m,\,i-1}, B). \qquad (7)$$

За даних умов $P(S|B)$ становить $1{,}29\times10^{-12}$, обчислення цього значення проілюстровано на рис. 2.4.

#	P(R|W)	wR	S	P(S)
1	P(CC)	0,038 cc	CC	3,80e-02
2	P(T|CC)	0,309 ccT	CCT	1,17e-02
3	P(A|CT)	0,229 ctA	CCTA	2,69e-03
4	P(C|TA)	0,193 taC	CCTAC	5,19e-04
5	P(T|AC)	0,290 acT	CCTACT	1,50e-04
6	P(A|CT)	0,229 ctA	CCTACTA	3,45e-05
7	P(T|TA)	0,339 taT	CCTACTAT	1,17e-05
8	P(A|AT)	0,311 atA	CCTACTATA	3,63e-06
9	P(T|TA)	0,339 taT	CCTACTATAT	1,23e-06
10	P(G|AT)	0,182 atG	CCTACTATATG	2,25e-07
11	P(C|TG)	0,209 tgC	CCTACTATATGC	4,69e-08
12	P(C|GC)	0,205 gcC	CCTACTATATGCC	9,61e-09
13	P(C|CC)	0,190 ccC	CCTACTATATGCCC	1,82e-09
14	P(A|CC)	0,341 ccA	CCTACTATATGCCCA	6,21e-10
15	P(G|CA)	0,189 caG	CCTACTATATGCCCAG	1,17e-10
16	P(A|AG)	0,345 agA	CCTACTATATGCCCAGA	4,04e-11
17	P(A|GA)	0,394 gaA	CCTACTATATGCCCAGAA	1,59e-11
18	P(T|AA)	0,251 aaT	CCTACTATATGCCCAGAAT	4,00e-12
19	P(T|AT)	0.323 atT	CCTACTATATGCCCAGAAT	1.29e-12

Рис. 2.4. Обчислення ймовірності натрапити на послідовність *S* (CCTACTATATGCCCAGAATT) за умови застосування моделі Маркова *B* (див. табл. 2.4)

Отже, на основі частот трапляння олігонуклеотидів можна побудувати модель Маркова, що даватиме змогу оцінити імовірність натрапити на певний нуклеотидний залишок (суфікс) після певного олігонуклеотиду (префікса). Важливим моментом у процесі побудови таких моделей є правильний вибір тренувального набору послідовностей, на основі яких визначають частоти префіксів. Таким набором може бути повна послідовність генома. Однак тоді оцінення частот префіксів буде “упереджене” – зсунуте в бік переважного оцінення префіксів, з яких складаються кодувальні послідовності (оскільки в геномах бактерій розмір некодувальних послідовностей доволі малий). Іншим тренувальним набором можуть бути міжгенні ділянки, або тільки 5′-прилеглі

ділянки генів. Другий варіант добре підходить бактеріям, однак його не варто використовувати для вищих організмів.

Моделі Маркова дають змогу ілюструвати особливості вживання (оліго)нуклеотидів у різних геномах. Ці дані можна репрезентувати у вигляді матриць переходів; для покращення сприйняття ці матриці можна зображати у вигляді *теплових карт* (heatmaps), де інтенсивність забарвлення пропорційна величині значення ймовірності переходу (рис. 2.5).

а

	A	C	G	T
A	0,398	0,064	0,052	0,485
C	0,478	0,133	0,067	0,322
G	0,368	0,086	0,124	0,422
T	0,447	0,057	0,077	0,419

б

	A	C	G	T
A	0,232	0,289	0,257	0,224
C	0,246	0,246	0,306	0,203
G	0,217	0,292	0,255	0,235
T	0,188	0,261	0,318	0,233

Рис. 2.5. Матриці ймовірності переходів у вигляді теплових карт для моделей Маркова першого порядку, натренованих на динуклеотидних послідовностях геномів *Plasmodium falciparum* (*а*) і *Mycobacterium leprae* (*б*). По вертикалі сірим виділено префікси. Темніші відтінки червоного рівнозначні вищим частотам (ймовірностям) переходів. Рисунок створено у програмі Excel; використане знаряддя “Умовне форматування → кольорові шкали”

2.3. Ймовірність і вірогідність моделей генетичних послідовностей. Якість будь-яких моделей, передусім ймовірнісних, залежить від розміру та якості вихідного масиву даних, на якому оцінюють (апріорні) параметри системи. Наприклад, імовірність q_a певного амінокислотного залишку можна оцінити як частоту натрапляння на цей залишок у певній базі даних амінокислотних послідовностей. Так можна отримати двадцять частот для 20-ти протеїногенних залишків після аналізу 20-ти мільйонів індивідуальних залишків у базі даних. Якщо розмір бази дуже великий (як у згаданому випадку) і якщо він не зсунутий у бік уживання певного залишку, то можна очікувати, що отримані частоти відображають імовірність натрапляння на залишки у будь-яких послідовностях. Такий спосіб оцінення (побудови) моделей називають *оцінкою максимальної вірогідності* (maximum likelihood (ML) estimation; див. блок 3). Можна показати, що використання частоти натрапляння на амінокислотні залишки у базі даних як

імовірностей q_a максимізує сумарну ймовірність усіх послідовностей за умов обраної моделі (вірогідність). Загалом, зважаючи на модель з параметрами θ і набір даних *D*, очікуване значення максимальної вірогідності – це значення, котре максимізує $P(D|\theta)$.

Поняття вірогідності має важливе значення в оцінюванні різних моделей НАП. Розглянемо такий приклад. Нехай обрані дві моделі частот нуклеотидних залишків ДНК. У моделі M1 C, G, A й T трапляються з частотами 0,35, 0,35, 0,15 і 0,15, відповідно. У моделі M2 кожен нуклеотид трапляється з однаковою частотою (0,25). Частоти появи всіх нуклеотидів незалежні одна від одної (модель Бернуллі). Нехай дано 13-нуклеотидну (13-нт) послідовність *x* = AGTGACGACTCAG. Яка модель (M1 чи M2) точніше описує цю послідовність? Найпростіший спосіб – це порівняти вірогідності обидвох моделей щодо заданої послідовності. Вірогідність моделі – це кондиційна ймовірність даних за умови моделі (див. блок 2). Визначимо вірогідність моделі M1:

$$P(x|M1) = (0{,}35)^6 \times (0{,}15)^7.$$

Аналогічно, вірогідність моделі M2 $P(x|M2)$ становитиме $(0{,}25)^{13}$. Тоді шукаємо *співвідношення вірогідностей* (likelihood ratio):

$$\frac{P(x|M2)}{P(x|M1)} = \frac{(0{,}25)^{13}}{(0{,}35)^6(0{,}15)^7} = 2{,}0333 > 1.$$

Отже, для послідовності *x* модель M2 має більшу вірогідність.

Розглянутий приклад – частина загальнішої проблеми оцінювання вірогідностей різних моделей/гіпотез в аналізі нуклеотидних чи амінокислотних послідовностей. Беручи за основу розглянутий приклад і теорему Баєза (див. блок 2), можна стверджувати, що

$$P(M2|x) = \frac{P(M2 \cap x)}{P(x)} = \frac{P(x|M2) \times P(M2)}{P(x)}.$$

Аналогічно можна записати умовну ймовірність для M1. Їхнє співвідношення становитиме:

$$\frac{P(M2|x)}{P(M1|x)} = \frac{P(x|M2) \times P(M2) \div P(x)}{P(x|M1) \times P(M1) \div P(x)}.$$

Імовірності даних у чисельнику і знаменнику скасовують одна одну, отож рівняння набуде такого вигляду:

$$\frac{P(M2|x)}{P(M1|x)} = \frac{P(x|M2) \times P(M2)}{P(x|M1) \times P(M1)}. \qquad (8)$$

У цьому рівнянні перша частина (ліворуч від знака рівності) – це *співвідношення апостеріорних імовірностей* (posterior odds ratio; див. блок 2), друга частина – співвідношення вірогідностей, третя – *співвідношення апріорних імовірностей* (prior odds ratio). У розглянутому прикладі встановлено другу частину рівняння (співвідношення вірогідностей). Обчислення співвідношення апостеріорних імовірностей тут (і часто на практиці) неможливе, оскільки апріорні ймовірності невідомі.

Загалом відомі два підходи до розв'язання проблеми порівняння імовірностей моделей чи гіпотез за умови певних даних, як це узагальнено рівнянням (8). Перші методи базуються на обчисленні співвідношення вірогідностей (likelihood methods), як проілюстровано попередньо. Другий підхід, який є похідним методів вірогідностей, ґрунтується на концепції Баєзової статистики (Bayesian approaches). Останні намагаються обчислити першу частину рівняння (8) – апостеріорну імовірність за умови даних $P(H|D)$. Замість обчислення імовірності того, що певна гіпотеза (модель) приведе до очікуваних даних, баєзіанські методи шукають значення ймовірності гіпотези за умови даних, присвоюючи певні значення співвідношенню апріорних імовірностей у рівнянні (8).

Значення апріорних імовірностей, або параметри моделі, легко встановити на основі ML-підходів, якщо маємо великий і повний масив даних. За неповного масиву даних використовують алгоритм *максимізації очікування* (expectation maximization, EM; блок 3). Якщо масив даних малий, то виникає ризик *надмірного налаштування* (overfitting) параметрів моделі. Отож модель буде пристосованою до тренувального набору (пояснюватиме його), але недостатньо загальною для того, аби правильно описувати нові дані (послідовності) з відмінними властивостями. Наприклад, якщо встановлювати параметри моделі GC-вмісту генів бактерій на основі геномів роду *Streptomyces* (GC-вміст ≈ 70 %), то гени інших бактерій, з низьким вмістом GC-пар у кодувальних послідовностях, розглядатимемо як невірогідні чи малоймовірні, або вони взагалі не будуть розпізнані як гени. У цих випадках можна використати теорему Баєса для оцінювання невідомих параметрів. Можна обчислити апостеріорну ймовірність будь-якого набору параметрів θ за умови певних даних *D*, як наведено внизу у рівнянні:

$$P(\theta|D) = \frac{P(D|\theta) \times P(\theta)}{P(D)} = \frac{P(D|\theta) \times P(\theta)}{\int_{\theta} P(D|\theta') \times P(\theta')}. \qquad (9)$$

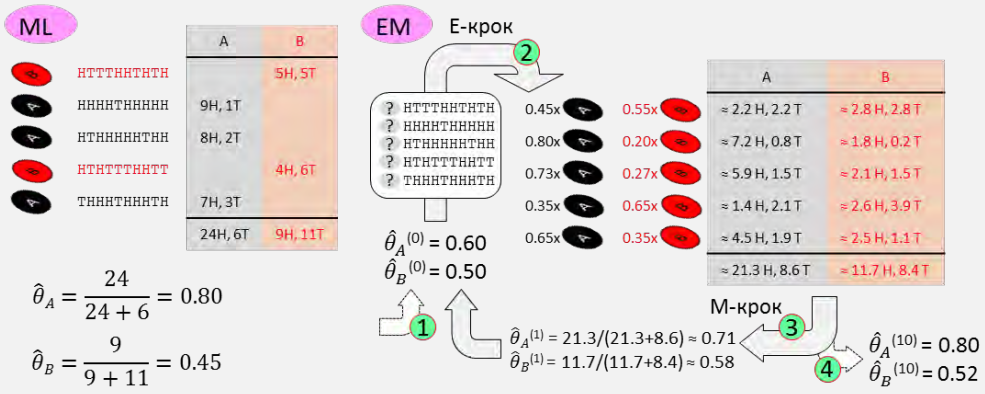
БЛОК 3. Максимізація очікування: визначення параметрів на основі неповних даних

Імовірнісні моделі дуже популярні у біології, оскільки є ефективні методи пошуку їхніх параметрів на основі масивів емпіричних даних. Однак часто такий масив даних неповний. У цих випадках можна скористатися алгоритмом максимізації очікування (expectation maximization, EM) для визначення параметрів. Як він працює?

Уявімо дві монети A і B, з них A при підкиданні випадає аверсом з наразі невідомою імовірністю Θ_A і реверсом – з імовірністю $1 - \Theta_A$; аналогічно – для монети B (Θ_B). Мета – оцінити Θ (Θ_A, Θ_B), повторюючи п'ять разів таку процедуру: вибрати випадково одну з двох монет (з однаковою ймовірністю) і підкинути її 10 разів. Загалом буде 50 підкидань. Під час досліду відстежуємо два вектори $x = (x_1, x_2,\ldots, x_5)$ і $z = (z_1, z_2,\ldots, z_5)$, де $x_i \in \{0,1,\ldots,10\}$ – кількість гербів у i-й серії підкидань, а $z_i \in \{A, B\}$ – тип монети в i-й серії. Маємо *повний масив даних* – усі змінні величини відомі. Тоді $\Theta_A = (\#\text{гербів при підкиданні } A/\text{загальне } \#\text{ підкидань } A)$. Так само – Θ_B. Цей метод має назву "оцінення максимальної вірогідності" (ML; див. також осн. текст і рис. внизу), і дає змогу оцінити якість статистичної моделі на основі ймовірності, яку вона присвоює даним.

Тепер уявімо складніший випадок, де є дані про випадання гербів x, але невідомий тип монети z, використаної у кожній серії підкидань. Тут z – прихована змінна, або латентний фактор, отож маємо неповний масив даних для оцінки параметрів. Якщо б вдалось завершити дані (вгадати монету (A чи B), яку використано у кожній із 5-ти серій підкидань), то оцінювання параметрів можна б звести до ML-методу (див. вище).

Процедуру "вгадування" коректних параметрів можна втілити у вигляді схеми з ітерацією двох кроків. *Е-крок*: вибирають (довільно) вихідні параметри $\hat{\theta}^{(t)} = (\hat{\theta}_A^{(t)}, \hat{\theta}_B^{(t)})$, визначають для кожного з 5-ти масивів, яка з монет (A чи B) найвірогідніше могла генерувати актуальні дані (на підставі вихідних параметрів). *М-крок*: припускаючи, що завершення даних коректне (тип монети вгадано правильно), застосовують ML-підхід і обчислюють нові параметри $\hat{\theta}^{(t+1)}$. Повторюють ці два кроки щоразу на основі нових параметрів (отриманих в М-кроці), доки не досягнуть конвергенції – стану, коли нові параметри вже не покращують опис масиву даних. Тобто досягнуто локальний максимум вірогідності того, що виявлено правильні параметри Θ (Θ_A, Θ_B). На практиці замість вибору одного можливого завершення даних (вгадування типу монет для 5-ти серій підкидань) у кожному раунді, EM-алгоритм обчислює імовірності для кожного можливого завершення даних на основі $\hat{\theta}^{(t)}$. Ці ймовірності дають змогу створити тренувальний набір, що складається з усіх можливих завершень. EM-алгоритм чергується між двома кроками: вгадування розподілу імовірностей різних можливих завершень за заданої моделі; переоцінки параметрів моделі на основі цих завершень.



Пояснення ЕМ. 1 – вихідне припущення про параметри; 2 – Е-крок, обчислюють розподіл імовірностей усіх можливих завершень даних на основі поточних параметрів; числа у таблиці – очікувана кількість аверсів (H) та реверсів (T) відповідно до цього розподілу; 3 – М-крок, на основі отриманих завершень даних визначають нові параметри, спрямовують їх в Е-крок; 4 – після кількох повторів Е- і М-кроків досягають конвергенції.

Оскільки розподіл параметрів переважно неперервний і не складається із дискретних величин, то знаменник тепер у вигляді інтеграла, а не суми, як подано у блоці 2. В цьому й полягає особливість баєзіанського підходу: якщо необхідно ввести невизначений параметр у проблему, то це відбувається не за рахунок простого (точкового) вибору певної величини. Натомість невизначеність знаменникової частини (рівняння (9)) вирішується систематичніше за допомогою інтегрування всіх можливих значень, яких може набути певний параметр.

Баєзове висновування є прямим, формальним способом маніпуляцій невизначеністю у системі, але він має кілька складнощів. По-перше, часто незрозуміло, як саме використати рівняння (9) для визначення оптимальних параметрів. Один можливий підхід – це підібрати такі значення параметрів для θ, які максимізують значення $P(\theta|D)$. Такий метод названо "оцінення максимального апостеріорного значення" (maximum *a posteriori* (MAP) estimation). Якщо у рівнянні використані неінформативні апріорні значення, то MAP-оцінка тоді є, фактично, оцінкою максимальної вірогідності (ML). В інших підходах оцінки параметрів використовують ті значення, які ведуть не до його максимального $P(\theta|D)$, а до середнього значення. Відомі інші підходи, усі вони доволі місткі в обчислювальному сенсі, оскільки неминуче потребують інтегрування невизначених параметрів. Таке інтегрування часто не має аналітичного розв'язку, отож потребує числового інтегрування (наприклад, MCMC (Markov Chain Monte Carlo)-симулювання).

По-друге, баєзіанські методи потребують визначення розподілу ймовірностей апріорних величин ($P(\theta)$), які також невідомі. Часто у цьому випадку немає раціональної основи для вибору значень апріорних даних, і тоді їм присвоюють т. зв. "неінформативні", або "пласкі" (часто однорідні) значення. В інших випадках можна вводити інформативні $P(\theta)$. Наприклад, відомо *a priori*, що амінокислоти фенілаланін, тирозин й триптофан структурно подібні і часто взаємозамінні в еволюційному значенні. Тому раціонально використати такі $P(\theta)$, які надають перевагу (приписують більшу ймовірність) тим наборам параметрів, що приписують цим трьом амінокислотам однакові ймовірності. Використання суб'єктивних припущень у статистичному оціненні є часто неприйнятним для частини фахівців у галузі теорії ймовірностей. Контраргументом є те, що небаєзіанські методи також дають підставу робити подібні припущення, отож, мабуть, краще робити це відкрито (як у баєзіанських методах).

По-третє, хоча теорема Баєза тривіально справджується щодо випадкових змінних *A* і *B*, не зовсім зрозуміло, чи параметри/ гіпотези (як це є у галузі біоінформатики) можна розглядати як випадкові змінні. Прийнятно говорити про ймовірність даних за умови певної моделі, де мають на увазі частоту, з якою можна було б отримати ці дані у разі нескінченної кількості спроб. Але якщо мова йде про одноразову, неповторювану подію, то вона або відбудеться, або ж ні. Тут немає частотного тлумачення імовірності події: ймовірність тут витлумачується у сенсі міри переконання чи ступеня віри в те, що подія відбудеться (припущення чи гіпотеза справдиться). Таке тлумачення змісту баєзіанського аналізу засновано на здоровому глузді, але воно залишається суперечливим серед фахівців у галузі статистики. Тому використання ймовірності як мірила віри є ще однією відмітною ознакою баєзіанських підходів.

Проілюструємо роботу баєзіанського методу. Пригадаймо приклад з гральними кубиками, який згадано у блоці 2. Нехай дано кубик φ і ми

припускаємо, що він “нечесний” – центр ваги зміщено на одну грань, але невідомо, на яку саме. Дозволено кинути φ 10 разів, і на основі отриманих даних якомога точніше оцінити параметри p_i. Отримано послідовність символів: 1, 5, 2, 2, 4, 1, 2, 6, 4, 2. Оцінення максимальної вірогідності для p_3, на основі фактичної частоти три (0/10), становить 0. Якщо використати цей параметр у моделі, тоді одна трійка у новому (і, можливо, більшому) масиві даних зробить неможливим припущення, що цей масив можна отримати за допомогою кубика φ. Однак така модель виглядає надто жорсткою. Очевидно, що вихідний масив даних (на основі якого визначали параметри моделі) недостатньо великий, аби бути впевненим, що кубик φ ніколи не випадає трійкою догори. Коригування фактичних частот, які використали для встановлення ймовірностей подій (випадання різних граней кубика) – відомий спосіб вирішення цієї проблеми. Такого коригування досягають за допомогою додавання до фактичних частот кожного можливого класу подій (випадання кожної із шести граней кубика) додаткових, “несправжніх” значень (наприклад, числа 1). Так, очікувана ймовірність випадання трійки p_3 буде тепер не 0, а 1. Додаткове значення для кожного класу називають *псевдорахунком* (pseudocount). Різні набори псевдорахунків відображають різні попередні (апріорні) припущення про те, який розподіл імовірностей випадання різних граней матиме гральний кубик. Якщо попередній досвід засвідчив, що кубики, здебільшого, були “чесними”, то можна додати великі псевдорахунки до фактичних значень. Якщо у минулому спостерігали багато “нечесних” кубиків у певному казино, то буде більше довіри до вже описаного масиву даних, і значення псевдорахунків мають бути невеликі. За будь-яких умов та наявності достатньо великого масиву даних, величина справжніх рахунків завжди переважатиме таку ж для псевдорахунків. Концепція псевдорахунків має широке застосування у різних розділах біоінформатики, зокрема, у побудові моделей НАП на основі множинних вирівнювань, де часто недостатньо даних для встановлення справжніх частот нуклеотидних чи амінокислотних залишків у тій чи іншій позиції вирівнювання.

Біоінформатичний аналіз НАП та їхніх властивостей потребує розуміння фундаментальних розподілів імовірності. Наприклад, геометричний розподіл дає змогу описувати відкриті рамки зчитування в ДНК і розподіл довжин фрагментів ДНК після оброблення ендонуклеазою рестрикції; розподіл Пуасона є наближенням до частоти появи олігонуклеотидів у ДНК. Основні розподіли наведено у блоці 4. Їхнє використання проілюстровано двома прикладами, а також низкою задач наприкінці цього розділу.

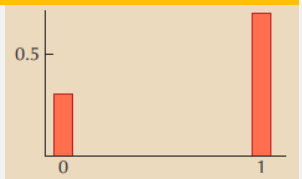
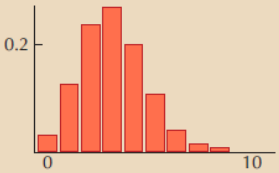
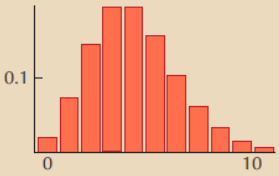
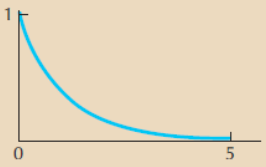
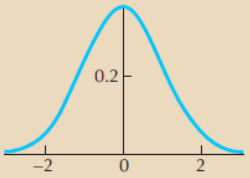
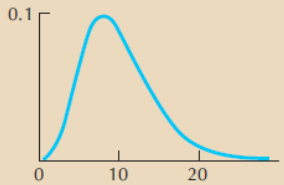
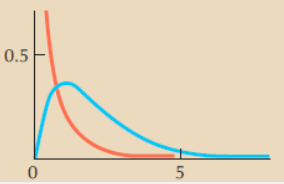
Приклад 1. Ендонуклеаза рестрикції розпізнає певну паліндромну гексануклеотидну послідовність і гідролізує (розрізає) ДНК у її межах. Потрібно визначити ймовірність того, що двониткову ДНК розміром 96 000 п.н. буде гідролізовано цією ендонуклеазою саме на 15 фрагментів. Відомо, що нуклеотиди у цій ДНК трапляються випадково (незалежно один від одного) і з однаковою частотою. По-перше, можна визначити ймовірність, з якою ендонуклеаза гідролізує ДНК в одній ділянці. Ця ймовірність становить $(1/4)^6 = 0{,}0002441$. Задля спрощення ситуації можна проігнорувати взаємну залежність появи сайтів розпізнавання для ендонуклеаз рестрикції у позиціях i та j, коли $|i\text{-}j| \leq 6$. Тоді число сайтів рестрикції X у ДНК можна розглядати як кількість успішних спроб (з імовірністю p) у загальній послідовності спроб L за моделі Бернуллі (всі спроби

БЛОК 4. Випадковий процес у біології, описаний розподілом імовірностей

Розподіл імовірностей описує ймовірність кожної можливої події у системі. Розподіли можуть бути дискретними (напр., число мутацій), або неперервними (частоти алелів, що коливаються у діапазоні 0÷1). Для дискретного розподілу сума подій дорівнює одиниці. Для неперервного розподілу $f(x)$ імовірність певної події (появи алеля із певною частотою) нескінченно мала, але площа під кривою розподілу визначає імовірність, що ця подія лежатиме у певному діапазоні. Або, використовуючи інтегральне числення, кажемо, що ймовірність події x лежить у межах від a до b:

$$\int_a^b f(x)dx.$$

Основні розподіли, які стосуються біологічних систем, зображені внизу.

Розподіл	Середнє	Варіанса	
ДИСКРЕТНІ			
Двох значень $f_0 = p$, $f_1 = q$ (на графіку $p = 0{,}7$)	p	pq	
Біноміальний $\frac{n!}{i!(n-i)!}q^{n-i}p^i$ ($n = 10$, $p = 0{,}3$)	np	npq	
Пуасона $e^{-\lambda}\frac{\lambda^i}{i!}$ ($\lambda = 4$)	λ	λ	
НЕПЕРЕРВНІ			
Експоненційний $\lambda e^{-\lambda x}$ ($\lambda = 1$)	$1/\lambda$	$1/\lambda^2$	
Гауса (нормальний) $\frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{(x-\bar{x})^2}{2\sigma^2})$ ($\bar{x} = 0, \sigma = 1$)	$\bar{x}$	σ^2	
Хі-квадрат $\frac{1}{2\Gamma(\frac{n}{2})}(\frac{x}{2})^{(\frac{n}{2}-1)}e^{-x/2}$ (n = 10)	n	$2n$	
Гамма $\frac{1}{\beta\Gamma(\alpha)}(\frac{x}{\beta})^{(\alpha-1)}e^{-x/\beta}$ (β = 2, α = 0,5 (червона лінія)). Гамма-розподіл є окремим випадком розподілу хі-квадрат.	αβ	$\alpha\beta^2$	

незалежні одна від одної). Отже, X матиме біноміальний розподіл з параметрами p та L. Оскільки L велике, а p маленьке, то, як наближення біноміального, можна використати розподіл Пуасона (див. блок 4) з параметром $\lambda = pL = 23{,}4$. Тоді:

$$P(X = 15) = 2{,}718^{-\lambda}\frac{\lambda^{15}}{15!} \cong 0{,}018\ (1{,}8\ \%).$$

Зазначимо, що імовірність розрізати ДНК на будь-яку заздалегідь задану кількість фрагментів k ($P(X = k)$) збільшується, коли k зростає від 0 до λ, і зменшується, коли k зростає від λ до L, сягаючи свого максимуму у точці $k = \lambda$. У цьому випадку розрізання ДНК на 23 фрагменти матиме найбільшу ймовірність.

Приклад 2. Відомо, що CG-острівці у геномах вищих евкаріотів збагачені на CG-динуклеотиди, тоді як ці динуклеотиди трапляються зрідка у решті хромосоми (“неострівці”). Припускають, що частоти появи CG можливо описати розподілом Пуасона, де в середньому на один острівець і неострівець завдовжки 250 п.н. може бути 25 і 10 CG-динуклеотидів, відповідно. До якого класу ДНК – острівець чи неострівець – належатиме 250-п.н. фрагмент із 19-ма CG-динуклеотидами?

В умові припускають, що кількість CG-динуклеотидів уCG-острівцях і не-острівцях можливо описати розподілом Пуасона з параметрами $\lambda_1 = 25$ і $\lambda_2 = 10$, відповідно. Тоді можна перефразувати питання: якщо дано 250-п.н. фрагмент ДНК, то яка ймовірність, що він належатиме до GC-острівця? Тут потрібно порівняти дві взаємовиключні апостеріорні ймовірності $P_1 = P$(острівець | n CG-динуклеотидів) і $P_2 = P$ (неострівець | n CG-динуклеотидів). Обидві альтернативні гіпотези *a priori* рівноімовірні (50 % або 0,5). Необхідно застосувати теорему Баєза для обчислення P_1 і P_2. Тоді:

$$P_1 = P(\text{ДНК описують за розподілом Пуасона з } \lambda_1 = 25 \mid n \text{ CG}) =$$

$$= \frac{P(n\ CG|\lambda_1 = 25) \times 0{,}5}{P(n\ CG|\lambda_1 = 25) \times 0{,}5 + P(n\ CG|\lambda_2 = 10) \times 0{,}5} = \frac{25^n \times e^{-25}}{25^n \times e^{-25} + 10^n \times e^{-10}}.$$

У цих розрахунках використано формулу розподілу Пуасона (див. блок 4), де спільний знаменник *n!* скорочено, оскільки він був наявний у верхній і нижній частинах рівняння. Аналогічно:

$$P_2 = P(\text{ДНК описують за розподілом Пуасона з } \lambda_2 = 10 \mid n \text{ CG}) =$$

$$= \frac{P(n\ CG|\lambda_2=10)\times 0{,}5}{P(n\ CG|\lambda_1 = 25)\times 0{,}5 + P(n\ CG|\lambda_2=10)\times 0{,}5} = \frac{10^n \times e^{-10}}{25^n \times e^{-25} + 10^n \times e^{-10}},$$

або $P_2 = 1 - P_1$.

Тоді простий алгоритм визначення CG-острівців працюватиме так. Для заданого 250-п.н. фрагмента ДНК з n CG-динуклеотидами обчислюють значення P_1; якщо $P_1 > 0{,}5$ ($P_1 > P_2$), то такий фрагмент ДНК ідентифікують як частину CG-острівця; інакше це частина неострівця. Для $n = 19$:

$$P_1 = \frac{(25)^{19} \times e^{-25}}{(25)^{19} \times e^{-25} + (10)^{19} \times e^{-10}} = 0{,}92;\ P_2 = 1 - P_1 = 0{,}08.$$

Отже, 250-п.н. фрагмент ДНК із 19-ма CG-динуклеотидами належить, найімовірніше, до CG-острівця; ймовірність того, що ділянка завдовжки 250 п.н. із 19-ма динуклеотидами буде неострівцем, становить 8 % (0,08).

Тепер, зважаючи на розглянуте завдання, сформулюємо таке правило. Якщо на 250 п.н. трапляється більше 18-ти CG-динуклеотидів, то такий фрагмент позначають як острівець. Необхідно визначити частоту виникнення псевдопозитивних і псевдонегативних помилок за таких умов.

Частоту псевдопозитивних помилок (false positive rate, FPR), або *помилок першого типу*, визначають як імовірність того, що неострівець, відповідно до заданого правила, ідентифікований як острівець. Оскільки число CG-динуклеотидів, X, у неострівці описується розподілом Пуасона з параметром $\lambda = 10$, то звідси:

$$FPR = P(\text{більше ніж 18 CG на 250 п.н.} \mid \text{неострівець}) =$$

$$= P(X > 18|\lambda = 10) =$$

$$= \sum_{n=19}^{+\infty} P(X = n) = 1 - \sum_{n=0}^{18} P(X = n) = 1 - \sum_{n=0}^{18} e^{-10}\frac{10^n}{n!} \approx 0{,}007\ .$$

Частота псевдонегативних помилок, або *помилок другого типу* (false negative rate, FNR) – це ймовірність того, що острівець буде ідентифікований як неострівець. Оскільки кількість CG-динуклеотидів Y у острівці підлягає закономірностям розподілу Пуасона з параметром $\lambda = 25$, то FNR становитиме:

$$FNR = P(\text{менше/рівне 18 CG на 250 п.н.} \mid \text{острівець}) =$$

$$= P(Y \leq 18|\lambda = 25) = \sum_{n=0}^{18} P(Y = n) = \sum_{n=0}^{18} e^{-25}\frac{25^n}{n!} \approx 0{,}09.$$

Помітно, що FNR>FPR, тобто задане правило класифікації швидше позначить острівець як неострівець, аніж навпаки.

2.4. Моделі еволюції нуклеотидних послідовностей. Розглядаючи дві генетичні послідовності, природним є питання про міру їхньої спорідненості чи віддаленості. Наприклад, цікаво знати, чи гени людини більше подібні до генів горили – чи генів шимпанзе; як давно розійшлися еволюційні шляхи генів людини та інших приматів; з якою швидкістю виникають мутації у геномах ссавців тощо. Простий приклад можливих варіантів еволюції окремих позицій генетичної послідовності наведено на рис. 2.6. Який із них притаманний для заданої пари послідовностей? Відповідь на це питання потребує кількісної моделі еволюції послідовностей, яку, очевидно, потрібно ґрунтувати на певних властивостях експериментального масиву даних.

Найпростіший спосіб порівняння двох послідовностей полягає в їхньому записі одної під другою так, щоб однакові літери знаходилися в одному стовпчику – як зображено на рис. 2.6, *а* на прикладі декануклеотидних послідовностей. Такий спосіб називають *попарним вирівнюванням* (pairwise alignment). Попарні вирівнювання можна виконувати вручну лише у випадку простих (коротких) послідовностей, на кшталт тих, що на рис. 2.6, *а*; концепції та алгоритми попарного вирівнювання генетичних послідовностей докладніше розглянуто у наступному розділі.

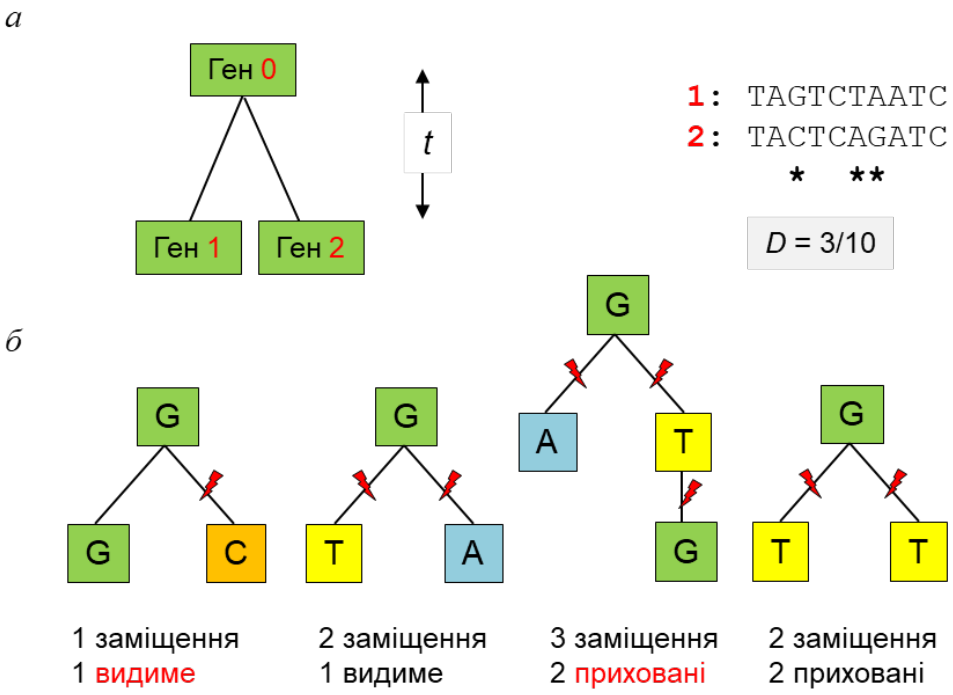


Рис. 2.6. Еволюція генетичних послідовностей. Ген 0 дав початок двом новим генам 1 і 2, наприклад, як результат дуплікації, розмноження чи видоутворення. У результаті дуплікації ген 0 подвоюється в геномі і спочатку зберігає ідентичність з генами 1 і 2, фактично він є геном 1 або 2. У результаті розмноження ген 0 матері/батька (предок) опиняється в геномах своїх дітей (популяційний процес). Аналогічно виглядає й процес видоутворення, лише в цьому випадку предок буде віддаленіший у часі від генів-дітей (філогенетичний процес). Нуклеотидна послідовність гена 0 (предок) невідома. Гени 1 і 2 відрізняються за нуклеотидною послідовністю, імовірно, тому, що вони мутували: незалежно один від одного накопичили певні випадкові нуклеотидні заміщення. Ці заміщення позначено зірочками у попарному вирівнюванні (див. рис. 2.6, *а*). Різні літери в одній позиції (стовпчику) попарного вирівнювання можуть бути наслідком різних еволюційних подій, як зображено в частині *б*

З аналізу попарного вирівнювання (див. рис. 2.6, *а*) бачимо, що три позиції містять різні літери. Отже, відмінність (відстань) між досліджуваними генетичними послідовностями $D = 3/10$. Логічно очікувати, що зі збільшенням часу від моменту утворення двох різних послідовностей (*дивергенції*) збільшуватиметься і відстань між ними. Якщо час дивергенції короткий, то послідовності міститимуть небагато заміщень. Імовірно, що кожне заміщення у такому випадку зачіпатиме іншу позицію послідовності. Тоді при вирівнюванні таких послідовностей усі заміщення – видимі (див. рис. 2.6, *б*). Припускають, що заміщення виникають випадково зі сталою швидкістю. Тоді середнє число заміщень, які виникають протягом часу t ,будуть пропорційні t, і D має зростати лінійно, якщо t набуває малих значень. Якщо t зростає далі, то можливий сценарій, коли в одній позиції виникає більше, ніж одне заміщення (див. рис. 2.6, *б*). При вирівнюванні двох наявних послідовностей помітно, що в одній позиції дві різні літери, але найпростіше та єдине, що можна припустити – це виникнення одного заміщення. Можливий також варіант, коли в двох різних послідовностях в одній позиції відбулося заміщення G на T, але наявні послідовності у цій позиції тепер однакові, отож заміщення *приховані* (див. блок 1). Кількість видимих заміщень між двома послідовностями завжди менша або дорівнює кількості заміщень, що фактично відбулися. Тому на великих часових проміжках D зростає повільніше, ніж на коротких, де спостерігають лінійну залежність. Якщо з моменту дивергенції минуло дуже багато часу, то заміщення відбудуться у кожній позиції. Послідовності будуть повністю випадковими (рандомізованими) щодо їхнього спільного предка. Якщо вирівняти дві випадкові нуклеотидні послідовності, що мають однакові частоти чотирьох основ, то в середньому ¾ позицій міститимуть різні літери. Тобто D наближається до ¾, коли час еволюції тривалий.

Відстань D як кількісна міра еволюції має низку недоліків. По-перше, як уже зазначено вище, вона не зростає лінійно з часом еволюції. По-друге, відстані *неадитивні*. Нехай відомо нуклеотидну послідовність спільного предка 0 (див. рис. 2.6, *а*). Тоді можна виміряти відстані D_{01} й D_{02}: від предка до сучасних послідовностей. Якщо б відстані були адитивними, то відстань від послідовності 1 до 2 була б сумою відстаней від послідовності 0: $D_{12} = D_{01} + D_{02}$. Якщо ж D – частка позицій, за якими послідовності відрізняються, то відстані не будуть адитивними. Тому корисно увести таку міру, яка була б лінійною з часом і *адитивною*. Зазвичай для цього використовують відстань d – середню кількість заміщень у перерахунку на одну позицію вирівняної пари послідовностей. Якщо заміщення виникають випадково, зі сталою швидкістю, то їхня кількість прямо пропорційна часові. Також кількість заміщень між послідовностями 1 і 2 (d_{12}) буде сумою d_{01} і d_{02}. Тому d – корисна міра еволюційної відстані між послідовностями. Однак, на відміну від D, d не можна визначити безпосередньо з послідовностей. Необхідно мати еволюційну модель, щоб обчислити d.

Найпростішу модель еволюції нуклеотидних послідовностей, відому під скороченням JC69, запропоновано 1969 року Томасом Джаксом (T. Jukes) та Чарльзом Кентором (C. Cantor). Модель описує одну позицію у попарному вирівнюванні ДНК, у якій може бути одна із 4-х можливих основ (*станів*) – A, C, G або T; основи трапляються в ДНК з однаковою частотою ($\pi_A = \pi_C = \pi_G = \pi_T = 1/4$); є єдина швидкість заміщення ($\alpha/3$) будь-якої з чотирьох основ на будь-яку з трьох інших. Тоді сумарна швидкість зміни однієї основи до будь-якої іншої становить α, оскільки є три інші основи. Для визначення середньої

кількості замін треба перемножити швидкість заміщення на часовий проміжок. Часовий проміжок (див. рис. 2.6, *а*) на гілці, що веде до кожної з двох послідовностей (1 і 2), становить t. Отже, сумарний час, протягом якого відбуваються зміни – $2t$. Звідси, середня кількість заміщень на одну позицію становить

$$d = \alpha \times 2t = 2\alpha t. \quad (10)$$

Однак насправді складно дізнатися справжню швидкість заміщень α. Тому обчислити d безпосередньо з рівняння (10) не вдасться. Цю проблему можна обійти, якщо вважати відстань D функцією швидкості заміщень і часу еволюції (виведення цього рівняння описано у блоці 5):

$$D = \frac{3}{4} - \frac{3}{4}e^{-8\alpha t}. \quad (11)$$

Якщо час невеликий, то $D \approx 2\alpha t$. Якщо час еволюції дуже великий, то D прямуватиме до ¾. Оскільки d і D є функціями αt, то цей добуток можна вилучити з рівнянь. Отож потрібно переписати вищенаведене рівняння так:

$$\ln\left(1 - \frac{4}{3}D\right) = -8\alpha t. \quad (12)$$

Підставляючи ліву частину отриманої рівності у рівняння (10) отримаємо:

$$d = -\frac{3}{4}\ln\left(1 - \frac{4}{3}D\right). \quad (13)$$

Рівняння (13) – найважливіше у цьому підрозділі: його можна використати для визначення еволюційної відстані d на основі даних про частоту видимих заміщень D. Параметр d відомий також як відстань Джакса-Кентора (JC) між послідовностями. Якщо значення D невеликі, то $D \approx d$; якщо $D >$ ½, то d значно перевищує D. Відстань стає нескінченною, якщо D наближається до ¾.

Моделі еволюції генетичних послідовностей можна репрезентувати у вигляді *матриці швидкостей заміщень* (rate matrix), недіагональні елементи якої r_{ij} описують швидкість (імовірність) переходу зі стану i (рядок) до стану j (стовпчик). Діагональні елементи – r_{ii} – відображають імовірність того, що нуклеотидна основа залишиться незмінною у заданій позиції двох послідовностей (імовірність збереження, або консервації). За такими моделями побудовані методи філогенетики. Моделі нуклеотидних заміщень – це матриці 4×4, що містять 16 станів. Наприклад, модель Джакса-Кентора JC69 відображено на рис. 2.7.

Аналіз великого масиву даних засвідчив, що у справжніх нуклеотидних послідовностях транзиції (Pu↔Pu, Py↔Py) виникають частіше, ніж трансверсії (Pu ↔ Py). Отож 1983 року Мотоо Кімура запропонував складнішу модель еволюції з двома параметрами: β для швидкості транзицій і γ – для трансверсій. Це *двопараметрична модель Кімури* (K2P) і, зазвичай, у ній β > γ. Для заданої пари вирівняних послідовностей можна вирахувати частку позицій, які відрізняються внаслідок транзиції (S), і частку тих, що відрізняються внаслідок трансверсій (V).

БЛОК 5. JC69: обчислення *d*, якщо значення α невідоме

Розглянемо одну позицію (сайт) послідовності. Нехай $P_{ij(t)}$ – імовірність того, що сайт у стані j у момент часу t, за умови того, що у предківській послідовності він був у стані i у момент часу 0. Стани i, j позначають основи ДНК (A, G, C, T). Нехай сайт перебуває у стані A, через інтервал t – у стані k, а потім, через короткий проміжок δt, знову в A. Імовірність такої події: $P_{\mathrm{AA}}(t+\delta t)$. Це можна виразити як суму усіх можливих основ, що можуть перебувати у сайті у момент t. Наприклад, якщо у стані k була основа C, то є імовірність $P_{\mathrm{AC}(t)}$, що A перейде в C у час t, яку далі перемножують на імовірність $\alpha\delta t$ – що відбудеться заміщення C на A у проміжок δt. Те саме справедливе, якщо у стані k будуть G або T. Сумарна швидкість заміщення A та будь-який інший стан (із трьох можливих) – 3α. Тоді імовірність, що заміщення не буде: $1 - 3\alpha\delta t$. Разом:

$$P_{AA}(t+\delta t) = \alpha\delta t\big(P_{AC}(t) + P_{AG}(t) + P_{AT}(t)\big) + (1 - 3\alpha\delta t)P_{AA}(t). \qquad (5.1)$$

Для малих значень δt : $P_{AA}(t+\delta t) = P_{AA}(t) + \delta t\, dP_{AA}/dt$. Підставляючи це рівняння у (5.1), отримаємо таке диференціальне рівняння:

$$\frac{dP_{AA}}{dt} = \alpha(P_{AC} + P_{AG} + P_{AT}) - 3\alpha P_{AA}. \qquad (5.2)$$

Відомо, що $P_{\mathrm{AC}} + P_{\mathrm{AG}} + P_{\mathrm{AT}} = 1 - P_{\mathrm{AA}}$. Тобто сума імовірностей того, що у стані k буде C, G або T дорівнює ймовірності того, що у стані k не буде A. Підставляючи цей вираз у (5.2), отримаємо:

$$\frac{dP_{AA}}{dt} = -4\alpha P_{AA} + \alpha. \qquad (5.3)$$

У цьому рівнянні одна невідома функція – P_{AA}. (5.3) – типове диференціальне рівняння. Спосіб їхнього розв'язку описано у вступних підручниках з математики для студентів університетів. Звідси припускаємо, що значення невідомої функції можна обчислити так:

$$P_{AA}(t) = Ae^{-4\alpha t} + B, \qquad (5.4)$$

де A і B – константи, які маємо обчислити.

Підставляючи рівняння (5.4) у (5.3) і розв'язуючи його згідно з описаними математичними підходами, можна визначити, що рівність задовольняється, якщо B = ¼. За вихідної умови, коли $P_{\mathrm{AA}}(0) = 1$, можна знайти, що A = ¾. Тоді розв'язок:

$$P_{AA}(t) = \frac{3}{4}e^{-4\alpha t} + \frac{1}{4}. \qquad (5.5)$$

Оскільки модель симетрична, то $\mathrm{P_{CC}}(t)$, $\mathrm{P_{GG}}(t)$ і $\mathrm{P_{TT}}(t)$ рівні $\mathrm{P_{AA}}(t)$. Також $\mathrm{P_{AC}}(t)$, $\mathrm{P_{AG}}(t)$, $\mathrm{P_{AT}}(t)$ рівні між собою, звідси:

$$P_{AC}(t) = \frac{1}{3}\big(1 - P_{AA}(t)\big) = \frac{1}{4} - \frac{1}{4}e^{-4\alpha t}. \qquad (5.6)$$

Якщо побудувати графік, де $P_{\mathrm{AA}}(t)$ і $P_{\mathrm{AC}}(t)$ є функцією часу t, то на великих часових відстанях обидві ймовірності прямують до ¼. Очевидно, що після довгого часу еволюції у сайті послідовності з однаковою ймовірністю (тобто 25 %, або 1/4) може знаходитися будь-яка основа, незалежно від того, яка основа була у точці відліку (t = 0).

Імовірність того, що у час t в сайті знаходиться основа, відмінна від основи у t = 0, рівна $P_{\mathrm{AC}}(t) + P_{\mathrm{AG}}(t) + P_{\mathrm{AT}}(t) = 3\,P_{\mathrm{AC}}(t)$. Коли порівнюють дві послідовності, які від спільного предка відділяє час t, то сумарний час, що розділяє ці послідовності – $2t$. Отже, ймовірність того, що ці дві послідовності матимуть різні основи в одній (вирівняній) позиції, буде $D = 3P_{\mathrm{AC}}(2t)$. У результаті отримуємо $8\alpha t$ у ступені рівняння (11) основного тексту. Таким чином отримуємо рівняння (11) в основному тексті.

а *б*

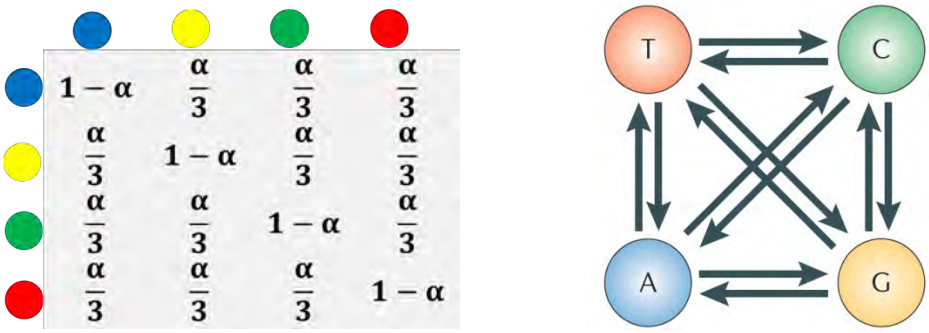


Рис. 2.7. Презентація моделі еволюції нуклеотидних послідовностей JC69 у вигляді: *а* – матриці швидкостей заміщення; *б* – графічної схеми (в обидвох варіантах виконання кола різного кольору означають чотири нуклеотидні основи; всі кола і стрілки у схемах однакові, що засвідчує однакову частоту нуклеотидів і швидкість їхнього заміщення)

Сумарна частка (D) позицій, за якими відрізняються послідовності $D = S + V$. Очікувана еволюційна відстань у перерахунку на одну позицію (d) для моделі K2P становить:

$$d = -\frac{1}{2}\ln(1 - 2S - V) - \frac{1}{4}\ln(1 - 2V). \qquad (14)$$

Модель K2P зображено у вигляді матриці і графічної схеми на рис. 2.8. В обидвох варіантах виконання кола різного кольору означають чотири нуклеотиди ДНК (основи). Зірочками у матриці позначені швидкість (імовірність) збереження основи, яка визначається як $1 - \beta - 2\gamma$. Всі кола у схемах однакові, що позначає однакову частоту нуклеотидів. Стрілки різної товщини відображають різну швидкість транзицій і трансверсій.

а *б*

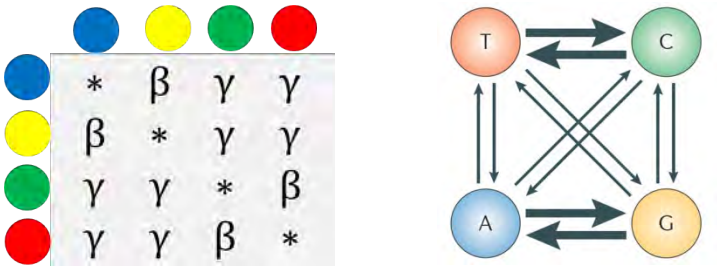


Рис. 2.8. Репрезентація моделі еволюції нуклеотидних послідовностей K2P у вигляді матриці швидкостей заміщення (*а*) і графічної схеми (*б*)

Порівняймо можливість різних моделей оцінювати відстань між двома послідовностями. Нехай для пари послідовностей S = 0,2, V = 0,1 і D = 0,3. Відповідно до моделі K2P (рівняння (14)), відстань між такими послідовностями становить 0,402. Згідно з JC69, відстань становить 0,383. Отже, оцінювання кількості заміщень у перерахунку на один сайт послідовності залежить від обраної моделі еволюції. Точне знання еволюційних відстаней важливе в галузі молекулярної філогенетики. Це дисципліна, метою якої є встановлення походження й еволюції видів, родинних зв'язків між різними НАП. Тому треба очікувати на точніші результати у таких дослідженнях за умови вибору відповідної моделі еволюції.

Обидві описані моделі не дають змоги врахувати ще один аспект справжніх нуклеотидних послідовностей – різну частоту основ. Наприклад, GC-вміст геномів деяких мікоплазм становить 20 %, а актиноміцетів – понад 65 %. Вперше враховано цей факт у моделі Гасеґави-Кішіно-Янґа 1985 року (модель HKY85), коли у матрицю швидкостей заміщення ввели чотири частоти основ, π_A, π_G, π_C, π_T (рис. 2.9), які набували різних значень.

а *б*

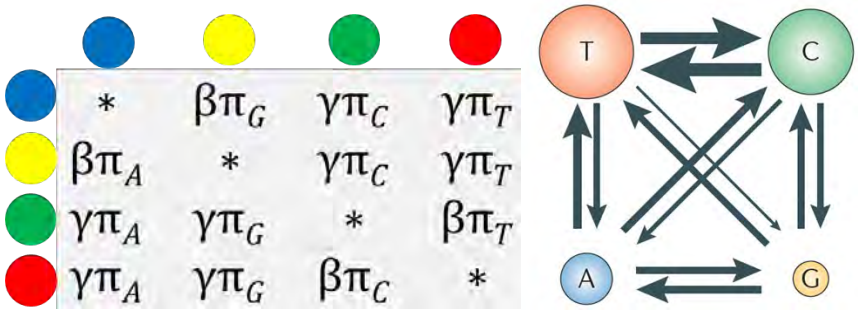


Рис. 2.9. Репрезентація моделі еволюції нуклеотидних послідовностей HKY85 у вигляді матриці швидкостей заміщення (*а*) і графічної схеми (*б*). В обидвох варіантах виконання кола різного кольору означають чотири нуклеотидні основи. Зірочки у матриці позначають швидкість (імовірність) збереження основи, яку визначають як $1 - \beta\pi_N - \gamma(\pi_{N+}\pi_N)$. Кола у графічній схемі і товщина стрілок різні, що означає різну частоту нуклеотидів і різну швидкість транзицій і трансверсій

Елементи матриці JC69 (й інших розглянутих вище моделей еволюції) – це швидкості заміщення, що мають марковську природу, тобто імовірність поточного стану не залежить від минулих – див. вище цей розділ. Відповідно, їх можна виразити як імовірності заміщення за одиницю часу *t*. Тоді матриця швидкостей заміщення буде перетворена у *матрицю імовірностей переходів* (transition probability matrix). Зокрема, для JC69 імовірності переходів описуватимуть системою з двох рівнянь: одне – для імовірності збереження залишку через час $t(p_0(t)$, для діагональних елементів); друге – для імовірності заміщення залишку через час $t(p_1(t)$, для недіагональних елементів). Ці рівняння набудуть такого вигляду:

$$p_0(t) = \frac{1}{4} + \frac{3}{4}e^{-4\lambda t}; \quad (15)$$

$$p_1(t) = \frac{1}{4} - \frac{3}{4}e^{-4\lambda t}, \quad (16)$$

де λ – швидкість мутацій, α/З з рис. 2.7.

У цій системі рівнянь дві змінні (швидкість і час), тому вона має безліч розв'язків. Використовуючи експериментальні методи біології, можна визначити λ для різних видів чи генів, що дає змогу визначити імовірність того чи іншого типу мутації через заданий проміжок часу. Наприклад, якщо для певного РНК-вірусу λ = 0,015 заміщень/(сайт × день), то, згідно з моделлю JC69, через 100 днів імовірність збереження залишку у певній позиції вірусного генома становитиме лише 25 %, а ймовірність мутації до будь-якого іншого нуклеотиду становитиме 75 % (рис. 2.10). На практиці це означає, що для пари послідовностей, які відрізняються на понад 75 %, встановити еволюційну відстань неможливо.

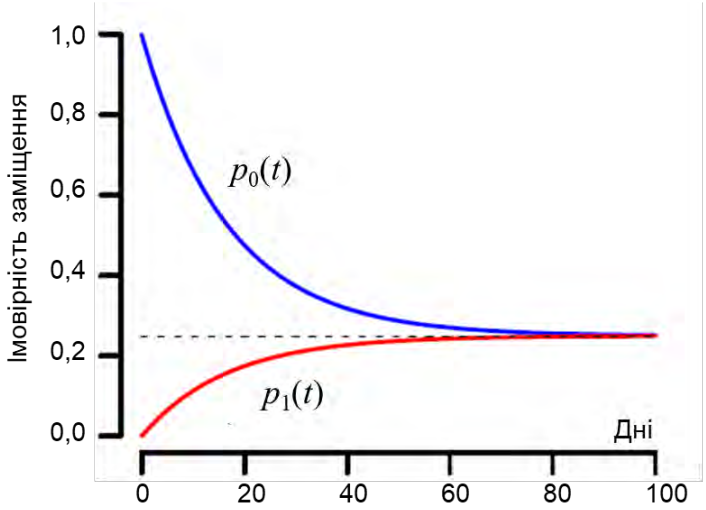


Рис. 2.10. Імовірність збереження чи заміщення нуклеотидного залишку як функція часу: модель JC69. Починаючи з 100-го дня імовірність збереження/заміщення нуклеотидного залишку не залежить від часу (стаціонарний стан). Модель JC69 корисна доти, доки вона не наблизилась до стаціонарного стану. Це можливо за умови невисокої швидкості мутацій і/або невеликих часових відстаней, для яких встановлюють імовірності заміщення (див. також текст нижче)

Через доволі тривалий проміжок часу будь-яка послідовність (нехай TTTTTTT), що еволюціонує згідно з моделлю JC69, складатиметься з однакової кількості усіх чотирьох можливих нуклеотидних залишків. Це так званий *стаціонарний стан* еволюційної системи. Після його досягнення надійне визначення часу її еволюції неможливе. Наприклад, нехай дано дві послідовності: M1 = TTTTTTT і M2 = ACGCATG. Хоча ці послідовності загалом не подібні одна на одну, у рамках JC69 можливо, що M2 є нащадком M1 за умови тривалого часу еволюції, достатнього для заміщення T на інші нуклеотидні залишки. З тих самих

міркувань послідовність M1 може походити з M2 – еволюційні моделі на кшталт JC69 *зворотні у часі*. Тобто у моделях марковської природи неможливо відрізнити нащадка і предка лише на основі наявних параметрів, таких як α. Чим більша α, тим швидше система досягає стаціонарного стану.

Сказане вище засвідчує необхідність належного розуміння можливостей і обмежень еволюційних моделей для правильного тлумачення отриманих результатів. Це доволі важливо у тих випадках, де точність визначення часу появи чи швидкості еволюції певної генетичної послідовності впливає на життя чи безпеку людини. Наприклад, у молекулярній епідеміології застосування адекватних моделей еволюції вірусів дає змогу передбачити час і джерела виникнення вогнищ інфекції, у криміналістиці допомагає встановити факт передачі вірусів від одних осіб іншим тощо. Приклад застосування моделей еволюції і їхнє практичне значення можна знайти за посиланням: https://evolution.berkeley.edu/evolibrary/article/0_0_0/%3C?%20echo%20$baseURL;%20?%3E/evotrees_treesmatter_07. Часто необхідно виражати швидкість еволюції в одиницях реального часу (дні, роки тощо) або в кількості нуклеотидних заміщень за одиницю часу. Таке можливе за умови, що λ/α має часову розмірність. Розглянемо такий приклад. Нехай в гаплоїдному людському геномі (3×10^9 п.н.) трапляється 2,2 заміщення на рік (тобто $\lambda = 2{,}2/(3\times10^9)$ заміщень/(сайт×рік)). З рівнянь (10) і (15) випливає, що еволюційна відстань d пропорційна добутку часу на α. Тоді час становитиме:

$$t = \frac{d}{\alpha} = \frac{d}{3\lambda}.$$

Отже, одне заміщення в одному сайті генома людини можна очікувати кожні ~ 450 млн років (рис. 2.11).

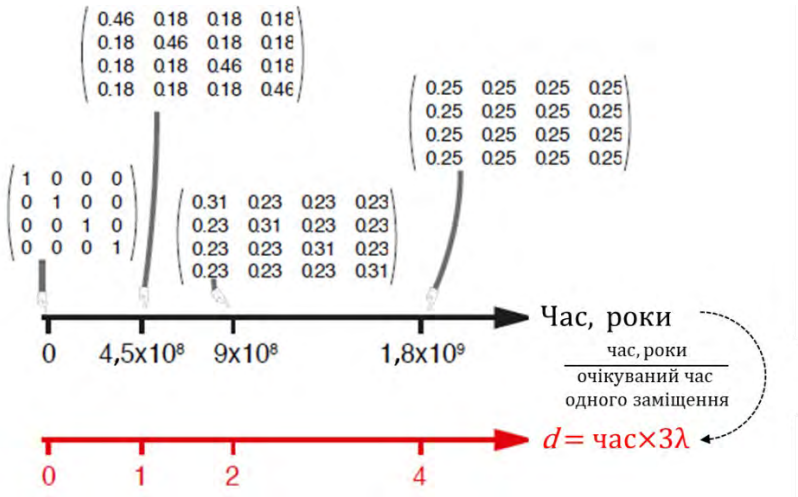


Рис. 2.11. Швидкість еволюційних подій можна виражати в різних одиницях. У початковому стані (точка 0, вісь часу, чорного кольору) обраний сайт генома описано одиничною матрицею. З плином часу імовірність збереження вихідного стану (нуклеотидного залишку) знижується (див. значення матриць над віссю часу), натомість зростає імовірність заміщення. В моделі JC69 відбувається одне нуклеотидне заміщення (вісь d, червоного кольору) за 450 млн років еволюції. Через 1,8 млрд років еволюційна система досягає стаціонарного стану

Згадану вище швидкість накопичення точкових мутацій у генах людини слід сприймати як орієнтовну, яка сильною мірою залежить від обраного методу

визначення, а також гена чи генів, на основі яких цю величину визначали. Сьогодні еволюційні швидкості для генома людини обчислено на основі різноманітних генів (див., наприклад https://pubmed.ncbi.nlm.nih.gov/22965354/), що відкриває нові можливості для вивчення еволюції та походження роду *Homo*. Завдяки доступу до нуклеотидних послідовностей генів з різних популяцій людини і викопних решток гомінід, можна змоделювати час їхньої дивергенції, шляхи міграції та розселення на різних материках Землі (більше про це – див. статтю за посиланням вище).

Описані моделі заміщення нуклеотидних залишків – приклади *механістичних*, або *параметризованих* моделей. У них еволюцію НАП моделюють на основі використання одного чи декількох параметрів, таких як швидкість транзиції, трансверсії тощо. Сьогодні описано величезну кількість (близько 1 600) механістичних моделей заміщення нуклеотидів (див. напр., doi: 10.3389/fgene.2015.00319). Однією із найскладніших і найефективніших є модель "Загальна зворотна у часі" (General Time Reversible) – GTR+Г+I. В абревіатуру цієї моделі входять два параметри: +Г означає, що частоти заміщення нуклеотидів уздовж послідовності можуть варіювати і вони підлягають гамма-розподілу (див. блок 4); +I означає, що у послідовності є певна кількість інваріантних залишків (вони не змінюються). *Гамма-розподіл* дає змогу розглянути швидкість зміни різних позицій (сайтів) нуклеотидної послідовності як випадкову змінну, яку вибирають із неперервного розподілу. Форму та масштаб цього розподілу визначають два параметри: α і β, відповідно. Вибір їхніх значень впливає на розподіл швидкостей, що проілюстровано на рис. 2.12 на прикладі параметра форми α.

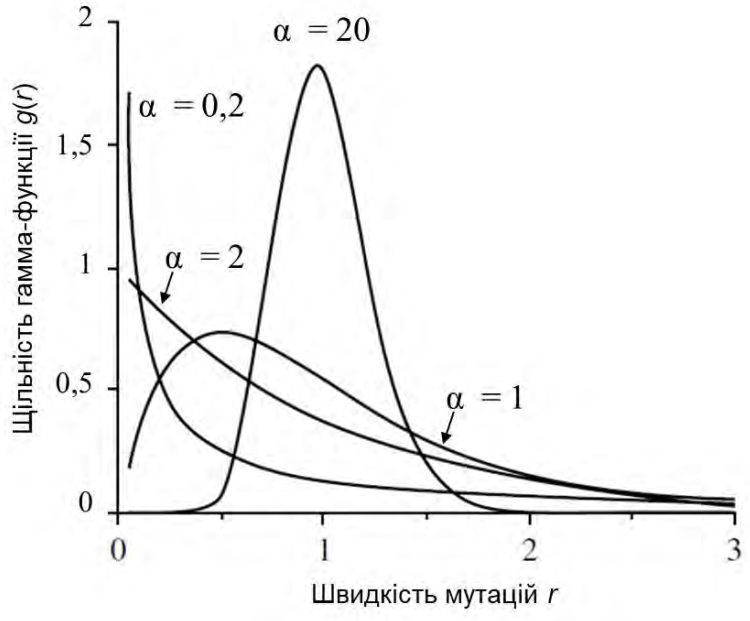


Рис. 2.12. Щільність імовірності функції гамма-розподілу варіабельної швидкості мутацій для різних сайтів нуклеотидної послідовності. Параметр масштабу розподілів (β) зафіксований так, що його середнє значення становить одиницю. Тоді щільність імовірності є функцією лише параметра форми α. Вісь абсцис – швидкість заміщення (мутацій), ординат – пропорційна кількості сайтів з такою швидкістю

Наприклад, якщо $\alpha > 1$, то розподіл дзвоноподібний, більшість сайтів має r в районі одиниці, а декілька з них мають або дуже високу, або дуже низьку швидкість. Коли $\alpha \to \infty$, то розподіл зводиться до моделі з однією швидкістю для всіх сайтів. Якщо $\alpha \leq 1$, то розподіл має L-подібну форму – більшість сайтів має дуже низьку швидкість, або ж вони є майже інваріабельними, але є і "гарячі точки" – сайти з високою r. Визначення α практично неможливе на основі двох послідовностей. Точні числові значення +Г й +І дослідник може налаштувати відповідно до своїх потреб і/або властивостей масиву послідовностей, які аналізують (про це йтиме мова далі). Створено низку веб-сервісів, які для заданого дослідником масиву даних пропонують адекватні моделі заміщень (див. наприклад, http://iqtree.cibiv.univie.ac.at/).

Окрім механістичних моделей, є також *емпіричні*: у них швидкості чи ймовірності переходів від одного стану до іншого визначають із експериментального масиву даних. Емпіричні моделі зрідка використовують при аналізі ДНК/РНК, але вони домінують у вивченні еволюції білків. Річ у тім, що білки складаються з 20-ти різних "літер", і моделювання їхньої еволюції кількома простими параметрами не давало ефективних результатів.

Якщо нуклеотидна послідовність кодує білок, то вона має кодонну структуру. Еволюцію таких послідовностей можна аналізувати на основі вже розглянутих нуклеотидних моделей, однак вони не враховують низки особливостей заміщення кодонів. Наприклад, для нуклеотидної моделі (JC69) припускають, що частоти різних нуклеотидних заміщень однакові, водночас відомо, що швидкості заміщення трьох різних позицій кодону різні. Зокрема, внаслідок дегенерованості генетичного коду зміни в третій позиції кодону часто мають синонімічний характер (не змінюють типу кодованого амінокислотного залишку білка) і трапляються частіше, ніж у першій і другій позиціях. Отже, порівняно з нуклеотидними моделями, перевагою *кодонних моделей* стає можливість приписати своє (унікальне) значення кожному можливому переходу від одного кодону до іншого. Перевагою кодонних моделей над моделями амінокислотних заміщень є можливість відрізнити несинонімічні (змінюють тип амінокислотного залишку) й синонімічні (не спричиняють заміну амінокислоти) зміни. Синонімічні, фактично, залишаються "невидимими" під час аналізу послідовностей білків, однак містять надзвичайно цінну інформацію про еволюційні сили, до діють на білок. Наприклад, селективний тиск на кодувальну послідовність можна виміряти як співвідношення несинонімічних до синонімічних швидкостей заміщень: d_N та d_S, відповідно. Ці співвідношення обчислюють на основі великого масиву вирівняних кодонних послідовностей (про вирівнювання послідовностей йтиме мова у наступному розділі), які кодують споріднені білки. Якщо зміни амінокислот у білку корисні, то швидкість несинонімічних замін вища, ніж швидкість синонімічних заміщень, і тоді їхнє співвідношення $\omega = d_N/d_S > 1$. Це є свідченням рушійного добору (positive selection). На противагу його, стабілізуючий добір (stabilizing або purifying selection) забезпечує незмінний амінокислотний склад, і тоді переважає швикість синонімічних заміщень – $\omega < 1$. Послідовності, що еволюціонують нейтрально, мають приблизно однакові швидкості несинонімічних й синонімічних замін – $\omega \approx 1$. Сьогодні розроблено низку програм, зокрема й у вигляді веб-сервісів (наприклад, DataMonkey https://www.datamonkey.org/), які дають змогу оцінити тип добору, що діє на певний (заданий дослідником) масив кодонних послідовностей.

Емпіричні моделі кодонних заміщень не надто поширені, оскільки є дуже трудомісткими (необхідно обрахувати 1 830 значень швидкості переходів у матриці 61×61 кодон; стоп-кодони ігнорують), і відображають особливості еволюції передусім того масиву даних, з якого обчислювали швидкості заміщень (тому отримані швидкості складно поширити на нові масиви послідовностей). Перевагою емпіричної кодонної моделі є те, що вона відображає усі складні і часто незрозумілі (відтак недоступні до моделювання) аспекти біологічної дійсності використаного масиву даних. Наприклад, створено кодонну модель на основі кодувальних послідовностей хребетних (doi:10.1186/1471-2105-6-134). Імовірно, зі зростанням швидкодії комп'ютерів привабливість створення емпіричних моделей зростатиме, незважаючи на їхнє обмежене застосування.

Механістичні кодонні моделі – найпоширеніші, вони ґрунтуються на низці параметрів, які обмежують можливі варіанти кодонних заміщень до невеликого набору параметрів. Розглянемо модель M0, яка є частиною програмного пакета PAML і є основою складніших моделей. M0 описує заміщення між кодонами як марковський процес, тобто ймовірність – швидкість – заміщення залежать від поточного стану системи, а не від минулих станів (див. попередній розділ). Станами в M0 є 61 змістовний кодон; стоп-кодони не включають; не дозволені мутації, що ведуть до стоп-кодонів. Модель описують матрицею Q розміром 61×61, значення якої Q_{ij} відображають швидкості заміщення кодону i на кодон j. Діагональний елемент матриці Q_{ii} (імовірність консервації – збереження певного кодону) підлягає такій вимозі: сума значень у ряді матриці становить нуль. Тоді: $Q_{ii} = -\sum_{i \neq j} Q_{ij}$. Імовірності заміщення кодону i на кодон j за проміжок часу t можна обчислити під час розв'язання диференціального рівняння $dP(t)/dt = QP(t)$ за вихідної умови $P(0) = I$, якщо $P(t) = \exp\{Q(t)\}$. Після визначення Q та стаціонарних частот кодонів обчислюють імовірності їхнього заміщення, які згодом використають для моделювання процесу мутації кодувальних послідовностей.

Нехай кодон i позначимо як $i_1 i_2 i_3$, а кодон j – як $j_1 j_2 j_3$, де $i_k, j_k \in \{A, C, G, T\}$, а позиція кодону k набуває значень 1, 2 або 3. Модель M0 визначає Q так:

$$Q_{ij} = \begin{cases} 0, & \text{якщо дві чи три з пар } (i_1, j_1), (i_2, j_2), (i_3, j_3) \text{різні} \\ \mu\kappa\pi_j, & \text{якщо тільки одна з пар } (i_1, j_1), (i_2, j_2), (i_3, j_3) \text{ різна, і це синонімічна транзиція} \\ \mu\kappa\omega\pi_j, & \text{якщо тільки одна з пар } (i_1, j_1), (i_2, j_2), (i_3, j_3) \text{ різна, і це несинонімічна транзиція} \\ \mu\pi_j, & \text{якщо тільки одна з пар } (i_1, j_1), (i_2, j_2), (i_3, j_3) \text{ різна, і це синонімічна трансверсія} \\ \mu\omega\pi_j, & \text{якщо тільки одна з пар } (i_1, j_1), (i_2, j_2), (i_3, j_3) \text{ різна, і це несинонімічна трансверсія} \end{cases}$$

У наведеній моделі не дозволені подвійні чи потрійні заміщення у межах кодону. Параметр κ – співвідношення швидкостей транзицій і трансверсій. Параметр ω визначає різну швидкість синонімічних та несинонімічних заміщень (див. попередній підрозділ). Параметри π_j задають частоти 61 кодону у стані

рівноваги. Вони можуть набувати різних значень, які визначають за частотами нуклеотидів, або безпосередньо (емпірично) визначають із певного масиву кодонних послідовностей. Матрицю заміщень масштабовано за допомогою параметра μ так, щоб у стані рівноваги середня швидкість заміщення дорівнювала одиниці: $-\sum_{i=1}^{61} \pi_i Q_{ii} = 1$. Тому час уздовж гілок філогенетичного дерева вимірюють в одиницях [кількість очікуваних нуклеотидних заміщень / кодон]. M0 задовольняє умову *зворотності часу* (time reversibility), тобто $\pi_i Q_{ij} = \pi_j Q_{ji}$ для всіх i та j (добуток частот нуклеотидів на швидкості прямих і зворотних мутацій однаковий).

Подальше ускладнення параметризованих моделей можливе за рахунок уведення додаткових параметрів, що, зокрема, дають змогу моделювати подвійні і потрійні заміщення в кодоні. Наявні дані засвідчують, що швидкості таких заміщень на два-три порядки нижчі, ніж швидкості одиничних мутацій у кодоні (див., наприклад, працю: doi:10.1093/molbev/msu196).

Описані комбіновані *емпірично-механістичні* моделі, в яких простий набір параметрів доповнюють експериментальними даними. До таких належить модель ECM, яка поєднує параметри M0 з даними про швидкості заміщень амінокислотних залишків на основі їхніх фізико-хімічних властивостей (їх беруть із бази даних Pandit). Більше про моделі кодонних заміщень можна знайти у праці: doi:10.1093/molbev/msn232.

2.5. Повтори. Послідовності низької складності. Природні генетичні послідовності суттєво відмінні від випадкових вервечок амінокислотних чи нуклеотидних залишків. Багато ділянок і сегментів цих полімерів має зміщений склад: наприклад, вони є *простішими* за амінокислотним (нуклеотидним) складом (із 20-ти можливих літер використано лише 5), ніж це очікувалося для статистично випадкових полімерів. Термін “проста послідовність” описує лише частину різноманітності й багатства варіацій природних послідовностей, які можуть бути вельми тонкими. Ці варіації мають глибоке значення для розуміння молекулярної структури, функції та еволюції живого.

Прості послідовності ДНК виникають унаслідок таких мутаційних процесів, як нерівний кросинговер, помилки реплікації, заміщення основ і транспозиція. Приклади простих послідовностей – мікросателіти, тандемні повтори зі змінною кількістю копій (VNTR), теломерні ділянки, GC-острівці. Кодувальні послідовності також виявляють тринуклеотидну квазіперіодичність унаслідок нерівномірного вживання літер у трьох позиціях кодону. Вважають, що не менше 25 % усіх амінокислотних залишків у білках, анотованих у базах даних, містять у своєму складі сегменти зі зміщеним складом. Зміщені послідовності часто є майже гомополімерними, входять до складу неглобулярних доменів фібрилярних чи структурних білків.

Складність, *паттерн* і *періодичність* – три різні абстрактні властивості генетичних послідовностей, які треба чітко розрізняти. Наприклад, опишемо три прості послідовності, що мають ідентичний ступінь складності (низька складність) унаслідок ідентичності (G_8A_8):

GAAGGAAAGGGAGAGA – послідовність не має ані паттерну, ані періодичності;

GGAGGAAAAGGAAGGA – послідовність має характерний шаблон (паттерн) вживання триплета GGA і квадруплета AGGA. Іншими словами, послідовність має *k-грамний* паттерн, де *k* = 3 (GGA) або 4 (AGGA), але ці *k*-грами розміщені нерегулярно і вони не виявляють періодичності;

GAGAGAGAGAGAGAGA – має *періодичність* (або *модуль*) 2 (мінімальна повторювана послідовність – дуплет GA) і, отже, низку *k*-грамних паттернів.

Складність (однією з граней якої є простота) може бути означена по-різному, на підставі концепцій теорії інформації, кодування, лінгвістики тощо. В узагальненому значенні можна говорити про *композиційну складність*, яка може бути локальною/глобальною властивістю НАП. Її обчислюють з векторів стану складності (рис. 2.13) і вона є загальною репрезентацією композиційного зсуву, що не залежить від паттерну чи періодичності. Також може бути *лінгвістична складність*, яка прив'язана до "багатства словника" – співвідношення кількості унікальних підпослідовностей (слів), які можна утворити з заданої НАП

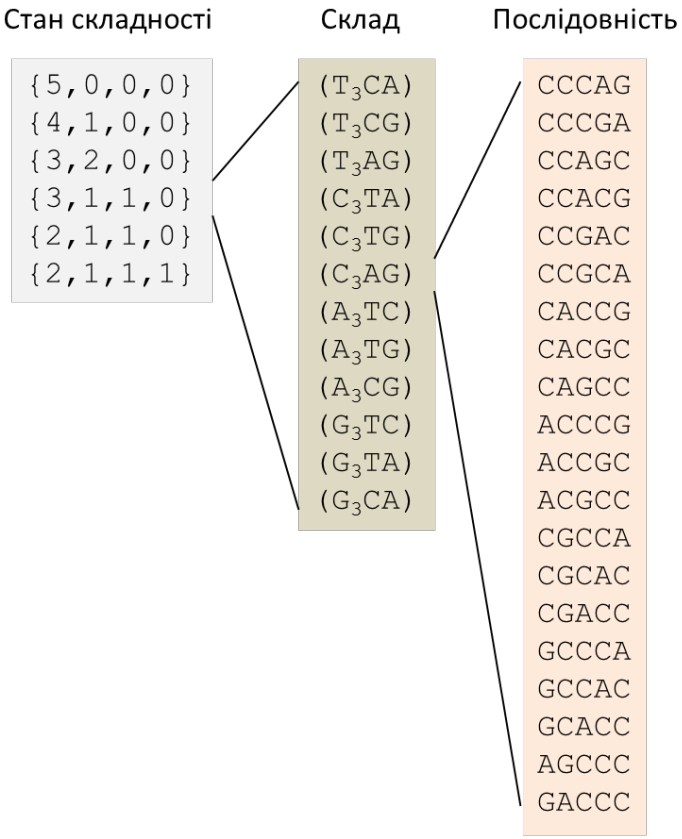


Рис. 2.13. Вектори стану складності, склад і послідовності. Чотирилітерна абетка ДНК, у разі довжини послідовності L = 5, генерує шість векторів складності, 56 складів і 4^5 = 1 024 послідовності. Для кожного стану складності всі склади описують однакову кількість можливих послідовностей, і ця кількість – основа для обчислення складності. Наприклад, один зі станів, {3,1,1,0}, має 12 складів, де різним нуклеотидам приписані чотири числа у векторі, і кожний такий склад описує 20 різних послідовностей. Можливість утворити 20 різних послідовностей робить стан {3,1,1,0} складнішим, ніж, наприклад, {3,2,0,0}, який описує лише 10 послідовностей на склад

до максимального числа слів, які можна теоретично утворити з послідовності такої ж довжини за умови використання тієї ж абетки. Порахуємо, наприклад, скільки слів можна утворити з послідовності ДНК S1: ACGGGAAGCTGATTCCA. S1 – 17 нуклеотидів (17 нт) завдовжки. Вона містить 15 із 16-ти можливих динуклеотидів, 15 із 15-ти можливих тринуклеотидів у нитці розміром 17 нт, максимально можливе число тетрануклеотиів (14) і так далі. В сумі S1 містить 4+15+15+14+…..+1 = 119 слів. Лінгвістична складність S1 = 119/120 = 0,992. Інша послідовність розміром 17 нт, S2 = ACACACACACACACACA, містить лише два різні мононуклеотиди, два різні динуклеотиди (AC і CA), два різні тринуклеотиди і т. д. S2 містить 2+2+2++2+……+1 = 33 різні слова. Тоді лінгвістична складність S2 буде 33/120 = 0,275. Складність найпростішого полімера – поліА$_{17}$ – буде 17/120 = 0,142. Це мінімальна складність для послідовностей завдовжки 17 нт.

Паттерни – прості послідовності (включно з нерегулярними повторами). Їх аналізують за вмістом та взаємним розміщенням. До паттернів застосовують поняття *k*-грамів (наприклад, ATTG є 4-грамом).

Періодичність – це повторення *k*-грамів з певним сталим інтервалом (періодом, модулем). Для ДНК, зазвичай, вирізняють справжню періодичність (наприклад, тандемні повтори) і квазіперіодичність, де повторення виникає як вторинний наслідок різних нерівномірностей у складі послідовності в різних фазах (наприклад, у різних позиціях кодонів).

Для білків висока композиційна складність є нормою. Наприклад, глобулярні білки за частотою і вживанням амінокислотних залишків нагадують випадкові послідовності за ступенем складності. Прості амінокислотні послідовності можна розглядати як відхилення від випадкової моделі вживання амінокислотних залишків у білках. На противагу білкам, ДНК настільки варіює за складом, що непросто визначити вигляд "випадкової" нуклеотидної послідовності. Незрозуміло, до якої міри мали б відхилятися частоти *k*-грамів, щоб вважати їх неочікуваними з точки зору теорії імовірності. Отож, сучасні бази даних білків, і передусім ДНК, радять спочатку розглядати як гетерогенну суміш з невідомими статистичними властивостями, які необхідно визначити. Знаючи ці властивості, можна класифікувати генетичні послідовності за ступенем складності. Практичною цінністю такої роботи є те, що сучасні методи аналізу генетичних послідовностей базуються на моделях випадкових послідовностей. Такі моделі незадовільно описують прості послідовності, саме тому їх варто вилучити з аналізу. У багатьох сучасних програмах цього досягають за допомогою процедури *маскування* простих послідовностей. Суть процедури полягає у її віртуальному вилученні з процесу порівняння, щоб вони не спотворювали результатів аналізу. Наприклад, у популярному пакеті програм попарного вирівнювання BLAST (його розглянемо згодом) використано програму SEG, яка маскує (*фільтрує*) ділянки низької складності. Якщо цього не робити, то список послідовностей, подібних до заданої (з низькою складністю), буде доволі великим і переобтяженим випадковими збігами.

Серед повторюваних ділянок найчастіше трапляються *тандемні повтори* (ТП) – послідовне виникнення одного нуклеотидного чи амінокислотного мотиву. ТП описують за довжиною мінімального повторюваного мотиву, кількістю мотивів і ступенем подібності мотивів. ТП виявляють у міжгенних, кодувальних і некодувальних частинах генома. ТП відіграють важливу роль у біології. Наприклад, багато таких структурних білків, як колаген, білок нуклеації

льоду, білки із тріозо-ізомеразним (TIM) доменом тощо, мають періодичну структуру завдяки ТП. Поширення ТП у складі білків часто спричиняє патологічні стани (наприклад, поліглутамінові тракти у білках асоційовані зі смертю нейронів), а в складі промоторів – зміни в експресії гена. Подібність у вихідному стані ідентичних ТП з часом втрачається внаслідок накопичення точкових мутацій, інсерцій і делецій (*інделів*). ТП, що мають змінену первинну структуру (внаслідок мутацій), можуть зберігати подібність на рівні тривимірних структур протягом тривалих еволюційних періодів. Точні механізми появи ТП недостатньо зрозумілі, але можуть ґрунтуватися на дуплікації чи втраті ТП, рекомбінації і генній конверсії. ТП можуть мутувати внаслідок помилок реплікації, де неправильне спаровування ниток ДНК веде до втрати чи набуття мотивів, коли петлі ТП формують шпилькові структури.

Простежується зацікавлення у виявленні і вивченні ТП. Сьогодні розроблено низку підходів для детекції ТП, які мають певні недоліки і переваги. Важливою проблемою є саме визначення ТП, оскільки їх декілька. Перше запозичене із комп'ютерних наук, де ТП визначено як повторюваний *регулярний вираз* (expression), в якому дозволено фіксовану кількість незбіжностей між ТП. На такому визначенні ТП засновані прості і вичерпні алгоритми (наприклад, http://tandem.bu.edu/trf/trf.html), де вирівнювання двох копій ТП розглядають як просту послідовність Бернуллі однієї з двох незалежних подій – збіг (успіх, Н) або незбіг (невдача, Т) нуклеотидних чи амінокислотних залишків (рис. 2.14). Однак такі методи виявлення ТП не завжди мають біологічний зміст. Зі структурно-біологічної точки зору, білкові ТП можна визначити як структурні повтори, тобто елементи, що на рівні третинної будови повторюються. Таке визначення дає змогу виявляти ТП з низьким ступенем подібності первинної послідовності (див., наприклад базу Repeats DB; http://repeatsdb.bio.unipd.it/). Насамкінець, в еволюційному визначенні стверджують, що ТП постають унаслідок дуплікацій предківського мотиву. Така точка зору має пряме біологічне тлумачення, відображаючи механізм утворення ТП. Чимало ТП виявлено сьогодні емпіричними методами, на основі визначення подібності мотивів, що не дає змоги оцінити статистичні властивості методів. Отож розроблення підходів до виявлення ТП на основі математичних моделей, що піддаються статистичному оцінюванню – головний напрям у галузі пошуку ТП. Наприклад, статистичний підхід втілений у пакеті програм TRAL (http://elkeschaper.github.io/tral/).

```
AGCTCACTAGTACACACACTTACACCAGA
CGCTCACTGGT--ACACACTCACACCAG-
THHHHHHHTHHTTHHHHHHHTHHHHHHHT
```

Рис. 2.14. Вирівнювання двох сусідніх копій ТП з гена рецептора β-Т-клітин людини: Н – збіг літер у позиції вирівнювання; Т – незбіг, інсерція чи делеція. Завдання алгоритму пошуку – виявити усі ТП, що задовільняють певній моделі, тобто мають певний статистичний розподіл властивостей. Наприклад, припускаємо, що у ТП має бути 80 % збігів ($P_M = 0{,}8$) і не більше 10 % незбігів ($P_I \leq 0{,}1$). Здебільшого точні значення цих параметрів невідомі, і в алгоритмі пошуку використовують певні статистичні критерії, щоб їх знайти (див., наприклад, https://tandem.bu.edu/trf/trfdesc.html)

Відштовхуючись від описаних визначень складності, локальну *композиційну складність ДНК* можна визначити як кількість усіх можливих нових послідовностей, які можна утворити внаслідок переставляння залишків у заданій послідовності розміром L:

$$\frac{L!}{\#\{A!\}\#\{C!\}\#\{G!\}\#\{T!\}}\ . \tag{17}$$

Отримане число – не нормалізоване щодо довжини НАП, отож довші послідовності матимуть більшу складність. Наведена нижче формула веде до значення складності ДНК нормалізованого щодо L, яке буде у межах від 0 до 1:

$$K_l = \frac{1}{L}\log_4\left(\frac{L!}{\#\{A!\}\#\{C!\}\#\{G!\}\#\{T!\}}\right). \tag{18}$$

За цими рівняннями можна вивести загальну формулу обчислення складності для будь-яких послідовностей символів, які ґрунтуються на різних абетках. K_l можна визначити як інформацію, яку необхідно мати, щоб за заданого складу послідовності (“вікна”) описати всі можливі послідовності, які можна утворити із залишків (літер) заданої. Тоді для абетки з N літер (чотири для ДНК і 20 для білків) і для довжини вікна (послідовності) L:

$$K_l = \frac{1}{L}log_N\left(\frac{L!}{\prod_{i=1}^{N} n_i!}\right), \tag{19}$$

де n_i – кількість літер N у векторі складності (див. рис. 2.13). Іншою мірою локальної композиційної складності є K_2, що, зазвичай, виражають у бітах:

$$K_2 = -\sum_{i=1}^{N}\frac{n_i}{L}\left(log_2\frac{n_i}{L}\right), \tag{20}$$

причому K_2 наближається за значенням до K_l при великих довжинах L.

За умови однакової ймовірності появи літер у НАП імовірність P_0 появи будь-якого стану складності обчислюють за формулою:

$$P_0 = \frac{1}{N^L}\left(\frac{L!}{\prod_{i=1}^{N} n_i!}\right)\left(\frac{N!}{\prod_{k=0}^{L} r_k!}\right), \tag{21}$$

де r_k – кількість разів появи числа k серед значень n_i вектора стану складності.

Обчислення P_0 є основою багатьох програм виявлення простих послідовностей у базах даних, наприклад, SEG (див.: http://smart.embl-heidelberg.de/smart/set_mode.cgi?NORMAL =1).

Типові задачі до розділу 2

Задача 2.1. В геномі віруса герпеса нуклеотиди C, G, A й T виникають з частотами 35/100, 35/100, 15/100 і 15/100. За незалежного вживання нуклеотидів у геномі визначити ймовірність того, що випадково обраний фрагмент ДНК завдовжки 15 нт міститиме вісім C або G і сім A або T.

Розв'язок. Частота натрапляння C + G й A + T у такій ДНК становитиме 0,7 і 0,3. Тоді ймовірність восьми C або G й семи A або T становитиме $0{,}7^8 \times 0{,}3^7 = 0{,}0000126$. Отриману величину ймовірності потрібно перемножити на кількість усіх комбінацій із восьми C або G й семи A або T, які можуть сформувати 15-нт послідовність. Це число – 15!/8!7! У результаті перемноження отримаємо шукану ймовірність – 0,08 (8 %).

Задача 2.2. Унаслідок надлишковості генетичного коду одну амінокислотну послідовність можна закодувати низкою різних нуклеотидних послідовностей. Нехай задано певну декапептидну послідовність. Яка мінімальна і максимальна кількість можливих послідовностей ДНК, що кодують цей декапептид?

Розв'язок. Нижньої межі можна досягти у тому разі, коли декапептид складатиметься лише з залишків метіоніну і/або триптофану, оскільки ці амінокислоти кодуються одним кодоном. Така амінокислотна послідовність кодується однією нуклеотидною. Верхньої межі можна досягти у випадку пептидів, що складаються з залишків лейцину, серину чи аргініну – для цих амінокислот є максимальна кількість (шість) кодонів. Декапептид, що складатиметься із будь-якого поєднання Leu, Ser чи Arg, можна закодувати 6^{10} – 60 466 176 – нуклеотидними послідовностями.

Задача 2.3. Прокаріотичний ген, що кодує білок, зазвичай складається із неперервної послідовності триплетів – кодонів. Ця послідовність починається спеціальним старт-кодоном (ATG – найпоширеніший) і закінчується одним із трьох стоп-кодонів – TAA, TAG, TGA. Послідовність із такою структурою має назву "відкрита рамка зчитування" (open reading frame, ORF). Однак не кожна ORF, яку можна знайти у прокаріотичній геномній ДНК, є функціональним геном. Якщо припустити, що ATG – єдино можливий старт-кодон, то яким є розподіл довжин випадкових ORF? Обчислення побудувати на моделі, де ймовірності появи всіх чотирьох нуклеотидів однакові й незалежні один від одного.

Розв'язок. У послідовності з однаковою ймовірністю можна натрапити на 64 триплети (кодони). Три з 64-х – стоп-кодони. Отже, ймовірність натрапити на стоп-кодон під час аналізу послідовності, триплет за триплетом, буде 3/64 = 0,047. Тоді ймовірність появи ORF довжиною L (в кодонах) запишемо так:

$$P\ (ORF \text{ завдовжки } L, \text{що починається у заданій позиції}) =$$
$$= P(ATG) \times P(\text{не стоп кодон})^{L-2} \times P(\text{стоп кодон}) =$$
$$= \frac{1}{64} \times (1 - \frac{3}{64})^{L-2} \times \frac{3}{64} = \frac{3}{4096}(\frac{61}{64})^{L-2}.$$

Для визначення розподілу довжин ORF скористаємося кондиційною ймовірністю:

$$P\ (\text{довжина } ORF \text{ становитиме } L) =$$
$$= \frac{P\ (ORF \text{ довжиною } L \text{ починається у заданій позиції})}{P\ \ (\text{будь-яка } ORF \text{ починається у заданій позиції})} =$$
$$= \frac{P\ (ORF \text{ довжиною } L \text{ починається у заданій позиції})}{\sum_{L=2}^{+\infty} P\ (ORF \text{ довжиною } L \text{ починається у заданій позиції})} =$$
$$= \frac{(\frac{3}{4096})(\frac{61}{64})^{L-2}}{1/64} = \frac{3}{64}\ (\frac{61}{64})^{L-2}.$$

Так виведено геометричний розподіл довжин випадкових ORF і параметри розподілу.

Задача 2.4. У районах бактерійної хромосоми, що кодують білок, на олігонуклеотид TTTTAAAA можна натрапити з частотою 0,008. Такий олігонуклеотид у некодувальних ділянках генома трапляється з частотою 0,003. У просеквенованому фрагменті ДНК завдовжки 400 нт TTTTAAAA знайдено чотири рази. Обчисліть апостеріорну ймовірність того, що цей фрагмент – частина гена, що кодує білок. Апріорна ймовірність того, що випадковий 400-нт фрагмент ДНК є частиною гена, який кодує білок, становить 0,8.

Розв'язок. Визначаємо такі події:

$B1$ = {400-нт фрагмент знаходиться у гені}

$B2$ = {400-нт фрагмент знаходиться у некодувальній ділянці}

C = {TTTTAAAA знайдено чотири рази в 400-нт фрагменті}

Відомо, що $P(B1) = 0{,}8$, а $P(B2) = 1 - P(B1) = 0{,}2$. Апостеріорну ймовірність $P(B1|C)$ можна обчислити за теоремою Баєза:

$$P(B1|C) = \frac{P(C|B1)P(B1)}{P(C|B1)P(B1)+P(C|B2)P(B2)}.$$

З цією метою застосовують розподіл Пуасона, параметр якого $\lambda = pN$, як наближення біноміального розподілу кількості виникнення олігонуклеотиду

TTTTAAAA (“успіхів”), оскільки ймовірність “успіху” p невелика, а число спроб N велике. Тоді отримуємо λ1 = 3,2 (0,8×400/100), а λ2 = 0,8 (0,2×400/100) для кодувальних і некодувальних послідовностей. Імовірності знаходження чотирьох олігонуклеотидів TTTTAAAA обчислюємо так:

$$P(C|B1) = P(\text{чотири } TTTTAAAA|B1) = e^{-3,2}\frac{3,2^4}{4!} = 0,1781,\ P(C|B2) =$$
$$= P(\text{чотири } TTTTAAAA|B2) = e^{-0,8}\frac{0,8^4}{4!} = 0,0332.$$

Тоді апостеріорна ймовірність того, що заданий 400-нуклеотидний фрагмент ДНК є частиною гена, становитиме:

$$P(B1|C) = \frac{0,1781 \times 0,8}{0,1781 \times 0,8 + 0,0332 \times 0,2} = 0,955.$$

Контрольні запитання до розділу 2

1. Як тлумачити поняття моделі у біоінформатиці?
2. Що описує рівняння Шеннона?
3. Що таке ентропія з точки зору теорії інформації?
4. Що описує модель Бернуллі?
5. Що таке ланцюг Маркова?
6. Що таке порядок ланцюга Маркова?
7. Для чого застосовують теорему Баєза?
8. Що таке кондиційна ймовірність?
9. Що описує еволюційна відстань *D*?
10. Що описує еволюційна модель JC69?
11. У чому відмінності моделей JC69 та K2P?
12. У чому відмінності моделей JC69 та HKY85?
13. Що таке стаціонарний стан у еволюційних моделях?
14. Що описує модель кодонних заміщень M0?
15. Яка роль гамма-функції у моделях нуклеотидних заміщень?
16. Який біологічний зміст зворотності в часі моделей заміщення?
17. Що таке еволюційна відстань *d*?
18. Що таке швидкість мутацій α?
19. У чому полягає відмінність між емпіричними та механістичними моделями?
20. Що таке апріорні та апостеріорні ймовірності в теорії Баєза?
21. Що таке ділянки низької складності?
22. У чому полягає принцип максимізації вірогідності?
23. Сформулюйте визначення композиційної складності НАП.

РОЗДІЛ 3

Попарне вирівнювання послідовностей

У результаті секвенування окремих ділянок або цілих геномів (чи пошуку у базах даних) дослідник отримує НАП, яка потребує відповіді на два взаємопов'язані питання:

1. Що це таке, що кодує ця послідовність?
2. Чи вже описано таку саму або подібну послідовність?

Найпростіший спосіб відповісти на них – почергово порівняти отриману НАП із усіма відомими послідовностями, описаними у базах даних (наприклад, GenBank), та оцінити ступінь їхньої подібності (якщо є). Таке порівняння найлегше виконати методом запису двох послідовностей одна під одною. Це називають методом *попарного вирівнювання* (pairwise alignment). Ураховуючи обсяги сучасних баз даних, ручний (візуальний) процес попарного вирівнювання – шлях нереальний і потенційно помилковий, насамперед для складних послідовностей. Складність передусім полягає у кількості різних елементів (літер), з яких послідовності складаються. Для ДНК таких літер чотири – A, T, G і C; для природних білків – 20. Другим компонентом складності генетичних послідовностей є їхній розмір – тисячі пар нуклеотидів для генів, сотні, а інколи тисячі амінокислотних залишків для білків, мільйони п.н. для геномів. Візуальне порівняння можливе для простих випадків, таких як зображений на рис. 3.1, але не для генів і білків зі "справжньої біології". Тому важливо мати алгоритми, які даватимуть змогу швидко порівнювати великі масиви послідовностей, оцінювати ступінь їхньої подібності і статистичну значущість отриманих результатів. Тут потрібно розрізняти поняття алгоритму та програми. Перше – це набір кроків, які визначають певний процес обчислення на абстрактному рівні, а друге – це робоче втілення алгоритму. Можуть бути різні програми, що втілюють один алгоритм, але всі вони мають вести до ідентичного результату, за умови суворого дотримання розробниками програм усіх приписів алгоритму. Цей розділ присвячуємо розгляду біологічних принципів, алгоритмів і онлайн-сервісів попарного вирівнювання, які є найважливішим інструментом аналізу НАП.

3.1. Концептуальні засади попарного вирівнювання. Процедура попарного вирівнювання полягає у записі двох генетичних послідовностей (заданої дослідником (query) та знайденої, наприклад, у базі даних (subject)) одна під одною так, щоб між ними було якомога більше *збігів* (match) і менше *незбігів* (mismatch). Тобто, у *позиціях вирівнювання* (стовпчик з двох символів) якомога частіше мають бути ідентичні амінокислотні чи нуклеотидні залишки. Найпростіший варіант вирівнювання – без уведення *розривів* (gaps). Однак їхнє уведення часто покращує *рахунок вирівнювання* (alignment score) – суму значень усіх позицій (збіги, незбіги і розриви) двох вирівняних послідовностей (рис. 3.1).

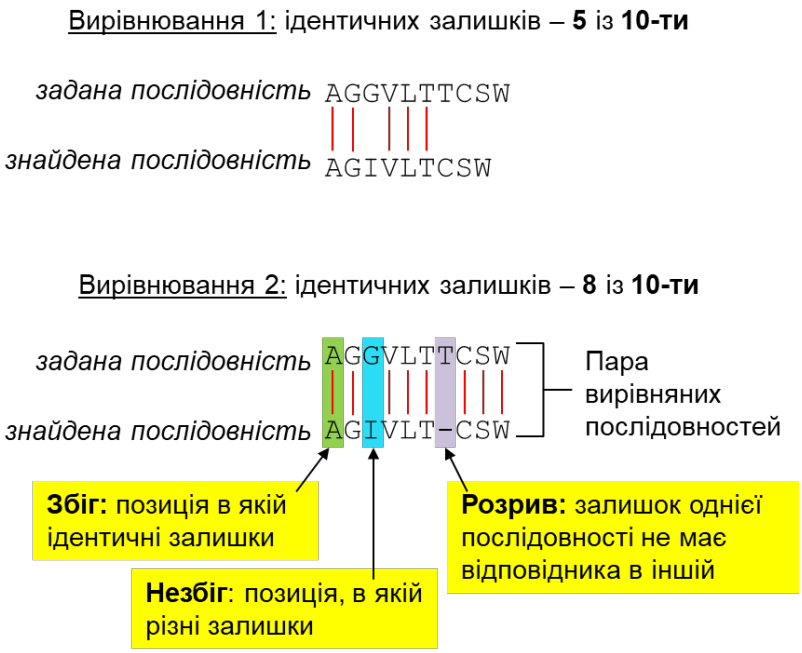


Рис. 3.1. Принцип попарного порівняння НАП і його основні елементи. Дві послідовності (*задана* дослідником і *знайдена* ним, наприклад, під час пошуку у певній базі даних) можна вирівняти у різний спосіб, два з яких проілюстровано. Перший випадок – "наївне" вирівнювання послідовностей від початку і без уведення розривів. У другому випадку введено розрив у знайдену послідовність, унаслідок чого значно зросла кількість ідентичних амінокислотних залишків у вирівнюванні (8 з 10-ти проти 5-ти з 10-ти у першому). Зауважте, що усі стовпчики попарного вирівнювання – і ті, де однакові символи (збіг), і різні (незбіг чи розриви) – враховують як позиції попарного вирівнювання

Об'єктом вирівнювання можуть бути три типи НАП (рис. 3.2). Це, по-перше, нуклеотидні послідовності. Кодувальні частини генів можна вирівнювати як кодонні чи амінокислотні послідовності. Ці три типи мають різне інформаційне навантаження. Найпростіше – вирівнювання нуклеотидних послідовностей, які ґрунтуються на дещо обмеженій абетці ДНК/РНК (чотири літери). Складнішими у сенсі виконання і тлумачення (див. далі у розділі) є вирівнювання амінокислотних послідовностей, основаних на значно ширшій абетці (20 літер). Вирівнювання кодувальних послідовностей генів (*кодонні вирівнювання*), де одиницею порівняння є кодон, мабуть, найскладніші технічно, оскільки дають змогу класифікувати різні типи генетичних змін за їхнім впливом на амінокислотну послідовність продукту гена. Таким чином можна відстежити характер селективних сил, які діють на послідовність гена. Наприклад, якщо у певному вирівнюванні спостерігають, здебільшого, заміщення, що не змінюють значення кодону (синонімічні заміщення, виділені зеленим кольором на рис. 3.2, *б*), то це може засвідчувати дію стабілізуючого добору на зазначені послідовності. Переважання ж несинонімічних заміщень (виділені сірим кольором на рис. 3.2, *б*) засвідчує дію рушійного добору (див. підрозділ 2.4). Вирівнювання нуклеотидних і амінокислотних послідовностей – найуживаніші в сучасній біології, отож саме їх розглядатимемо. Окремі аспекти кодонних вирівнювань розглядатимемо в інших розділах, присвячених множинним вирівнюванням і філогенетиці.

Рис. 3.2. Попарне вирівнювання двох нуклеотидних послідовностей (*а*), що кодують білки, можна також репрезентувати у вигляді кодонного (*б*) або амінокислотного (*в*) вирівнювання. Дві тринуклеотидні позиції, позначені пунктирним овалом, відрізняються за однією позицією, але в другому випадку це веде до несинонімічного заміщення на рівні білка

Дві послідовності, обрані для порівняння, можуть бути або неспорідненими (випадковими), або ж еволюційно спорідненими. Під еволюційною спорідненістю маємо на увазі, що в минулому існувала єдина й однорідна популяція послідовностей, з якої постають дві різні популяції – унаслідок розходження і накопичення різних генетичних змін (мутації, рекомбінації, інверсії, злиття генів тощо). Попарне вирівнювання двох послідовностей дає змогу визначити правильну гіпотезу з двох взаємовиключних. Фактично всі методи порівняння генетичних послідовностей базуються на пошуку подібності між ними, з якої і роблять висновок про гомологію. Ці два терміни – подібність і гомологія (гомологічність) – фундаментально відмінні. *Подібність* – це певна кількість, яку можна обчислити й описати, наприклад, у вигляді співвідношення (частки, відсотків) ідентичних літер до всіх літер у певному вирівнюванні двох послідовностей (див. рис. 3.1). *Ідентичність* – один із екстремальних випадків подібності (100 % подібності). *Гомологія* – якісний термін, який означає, що два об'єкти (генетичні послідовності, органи тощо) мають спільне еволюційне походження. Тобто два гени можуть бути або гомологічні, або негомологічні, і вислови на кшталт "50 % гомології" чи "…менш (більш) гомологічний ніж…." – позбавлені змісту, бо порівняльних ступенів гомології немає. Водночас два гени можуть бути більше подібні чи менше подібні, порівняно з певним стандартом, і цю міру подібності можна описати кількісно – як відсоток подібних/ідентичних нуклеотидів у вирівнюванні. Тому процес порівняння первинної структури генів і білків у своїй основі є пошуком еволюційних зв'язків між ними. Процес видоутворення полягає у розходженні (дивергенції) колись єдиної популяції організмів, і це відображається на молекулярному рівні. У процесі дивергенції предковий ген у новоутворених видах зазнає різних еволюційних змін, серед яких – заміщення, делеції та інсерції. В ідеальному випадку, якщо у певній позиції

попарного вирівнювання розміщені два різні нуклеотидні чи амінокислотні залишки, то це засвідчує заміщення (див. підрозділ 2.5). Якщо у вирівнюванні залишкові однієї послідовності відповідає розрив у другій (див. рис. 3.1), то це можна тлумачити як делецію в одній послідовності або ж як інсерцію в іншій. Гомологічність двох генів чи білків випливає із попарного вирівнювання, в якому значна частина збігів (позицій з ідентичними залишками) і незбігів, які вважають прийнятними для споріднених білків згідно з певними критеріями (про це докладніше у наступних підрозділах). Тобто справедливим буде твердження, що частина позицій у попарному вирівнюванні містить гомологічні нуклеотидні (амінокислотні) залишки, а частина – негомологічні (незбіги, розриви). Наприклад, залишок гліцину (див. рис. 3.1) у заданій послідовності (затінений блакитним; вирівнювання 2) не має гомолога у знайденій послідовності.

Зауважимо, що дані попарного вирівнювання можуть слугувати на користь припущення про гомологічність генетичних послідовностей, але не можуть бути фактом гомології. Гомологія двох послідовностей – це завжди припущення, адже спостерігати за процесом появи гомологічних послідовностей у природі (розходження колись однорідної популяції організмів/геномів) неможливо; для цього довелося б здійснити мандрівку в минуле. На сучасному етапі розвитку геноміки варіантом таких "подорожей у часі" стали довготермінові еволюційні експерименти на бактеріях, які можна культивувати протягом тисяч поколінь і через певні проміжки часу відбирати зразки для довготермінового зберігання при -80 °C. За потреби такі зразки можна відживлювати і секвенувати їхні геноми, і в такий спосіб прямо виявляти процес поступового накопичення змін у генетичному матеріалі (див., наприклад, doi: 10.1126/science.342.6160.790; doi:10.1038/463864a; doi: 10.1371/journal.pbio.1002185). Однак тривалість таких експериментів наразі недостатня, аби екстраполювати їх на процес еволюції біосфери; неочевидно, чи дослідження бактерій даватимуть відповідь на природні процеси еволюції вищих організмів. На противагу гомології, ступінь подібності двох послідовностей – чітко встановлений, і кількісний показник – факт, який слугує підставою для припущення про гомологію. Зауважимо також, що визначення гомології стосується лише еволюційних зв'язків між НАП, і не засвідчує, що гени- чи білки-гомологи мають однакову/подібну функцію. Справді, еволюційна спорідненість не завжди відображає функціональну, хоча очевидно, що дуже подібні білки (на рівні первинної структури) мали б бути гомологічними, і треба очікувати, що вони матимуть подібні функції.

Декади біохімічних досліджень (підсумованих у вигляді догми Анфінсена) дають підстави стверджувати, що тривимірну структуру більшості білків визначають за їхньою первинною структурою (амінокислотною послідовністю). Отже, якщо дві амінокислотні послідовності достатньо подібні на рівні первинної структури, то можна очікувати, що і їхні третинні структури теж подібні. Це надзвичайно важливе припущення, яке можна практично використати для передбачення будови і функції протеїнів. Постає наріжне питання: наскільки подібними мають бути дві амінокислотні послідовності, аби вважати їх гомологічними? Очевидно, що 100 % подібність (ідентичність) є переконливим аргументом на користь гомологічності двох білків, відтак – однакової функції (проте, знову ж таки, гомологія і тут не є фактом, адже не можна викреслити імовірність конвергентної еволюції двох послідовностей). Вважають, що достатньо довгі послідовності, які мають не менше 35 % подібності, можуть бути

гомологічними. На межі 20–35 % подібності знаходиться “зона сутінків” – рівень подібності недостатньо високий, аби стверджувати про гомологію; водночас такий рівень недостатньо низький, щоб засвідчити негомологічність (рис. 3.3). Розроблення чутливих методів виявлення гомології серед послідовностей, які виявляють малу подібність, – один із важливих напрямів біоінформатики. Деякі з цих методів розглядатимемо згодом.

Низький рівень подібності двох амінокислотних послідовностей не може слугувати доказом їхньої негомологічності так само, як високу подібність первинних послідовностей білків неможливо розглядати як факт гомології. Причина полягає у тому, що функцію білка визначає його третинна структура, а вона еволюціонує значно повільніше, ніж первинна. Тобто кодувальна частина гена може накопичувати мутації, які приводять до амінокислотних замін, але ці заміни, в абсолютній більшості, не змінюють характер *згортання* (folding) білка. Структурна біологія рясніє прикладами білків, послідовності яких ідентичні не більше ніж на 12 %, але які вражаюче подібні на рівні тривимірної будови. Тому порівняння третинних структур білків вважають “золотим стандартом” виявлення гомології, до якого прагнуть наблизитися за допомогою методів порівняння НАП. Деякі з методів порівняння структур розглянемо згодом. Все ж саме методи порівняння НАП, а не третинних структур, залишаються найуживанішим інструментом у сучасній молекулярній біології. Вони дають змогу швидко вирівняти нову послідовність із відомими і сформулювати гіпотезу про її функцію. Це створює основу для подальших біологічних досліджень *in vivo, in vitro* чи *in silico*, адже абсолютна більшість анотацій генів і білків у базах даних – це біоінформатичне передбачення, результат порівняння нових послідовностей з уже описаними. Зважаючи на сказане, складно переоцінити значення методів пошуку подібних послідовностей. Якщо ген чи білок розглянути як слово у певній мові, то без цих методів великі масиви даних були б не наборами слів, а лише стосом літер, у послідовності яких зашифровані невідомі слова.

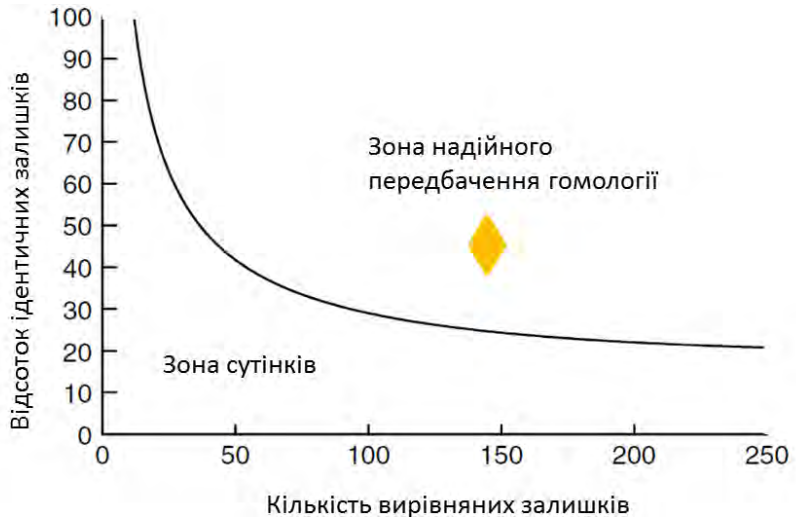


Рис. 3.3. Дві зони вирівнювання білків. Для двох послідовностей практично гарантовано, що вони матимуть однакову третинну структуру, якщо їхня довжина і відсоток ідентичності перебувають у межах зони, яку позначено як “надійну”. Приклад двох послідовностей завдовжки 150 ак, в яких 50 % ідентичних позицій у вирівнюванні, позначено ромбом

3.2. Механістичні підходи до кількісного оцінювання попарних вирівнювань. Розглянемо дві послідовності, К і Т, перша з яких має розмір 200 залишків, друга – 350. Якщо послідовність К повністю ідентична до будь-якої частини послідовності Т, то К – *підпослідовність* Т. Необхідно ввести розриви з одного чи обидвох боків послідовності К для формального вирівнювання К і Т (рис. 3.4, *а*). Якщо ж К′ містить два райони, що проявляють ідентичність до Т, то треба вирівняти відповідні ділянки К і Т, а між ними увести розриви для завершення вирівнювання (рис. 3.4, *б*).

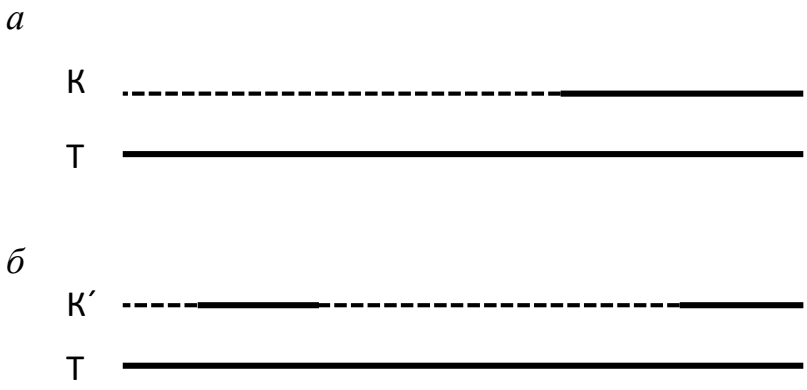


Рис. 3.4. Попарне вирівнювання послідовностей К і Т у випадку, коли К є підпослідовністю Т (*а*), або ж у випадку, коли окремі ділянки К′ ідентичні з двома неприлеглими ділянками Т (*б*)

Описані дії – найпростіший евристичний алгоритм вирівнювання послідовностей, що мають яскраво виражені ділянки ідентичності. Ці приклади вирівнювання, однак, не відображають реальної картини біоінформатичного пошуку, де досліджувані послідовності складніші, ніж розглянуті попередньо. Часто дослідник переслідує мету – вирівнювання якомога більшої кількості позицій двох послідовностей (бажано, від їхнього початку і до кінця) – *глобальне вирівнювання* (global alignment). Відповідні алгоритми мають давати змогу оцінювати всі позиції попарного вирівнювання. Іншим різновидом є *локальне вирівнювання* (local alignment): пошук спарованих підпослідовностей (“острівців”) з максимально високим рівнем подібності (часто за рахунок ігнорування ділянок, які знижують подібність), оточених неідентичними ділянками. Алгоритми локального вирівнювання дають найбільше інформації під час аналізу геномних баз даних, і їх розглядатимемо далі. Потрібно пам’ятати, що застосування методів глобального вирівнювання може приховувати ділянки локальної подібності, отож дослідникові необхідно вибрати методи вирівнювання відповідно до мети експерименту.

Існує дуже багато варіантів вирівнювання двох достатньо довгих послідовностей, і мета дослідника – отримати й проаналізувати “найкраще” з них. Тут слово “найкраще” може мати кілька значень, залежно від мети експерименту. Для початку його можна сформулювати так: найкраще вирівнювання матиме максимальну кількість ідентичних позицій. Отже, повертаючись до рис. 3.1, з двох описаних вирівнювань кращим є друге, бо в ньому більша частка ідентичних амінокислотних залишків. Цю кількісну величину, або міру подібності двох послідовностей, назвали *рахунком вирівнювання* (alignment score). Зауважимо, що

“краще” вирівнювання у цьому прикладі досягнуте за рахунок уведення *розриву* в одну із послідовностей (див. рис. 3.1). Як у глобальному, так і в локальному вирівнюваннях у багатьох позиціях будуть незбіги, і вирішення проблеми уведення розривів для досягнення максимального рахунку ставатиме дедалі складнішим зі зростанням розміру послідовностей. Як зазначено, уведення розривів ґрунтується на припущенні про спільне еволюційне минуле порівнюваних послідовностей. Хоча суто механічне, неконтрольоване введення розривів у дві послідовності, що їх вирівнюють, дійсно може забезпечити високий рахунок, у результаті вирівнювання може втратити біологічний зміст. Тому кінцева мета всіх сучасних алгоритмів пошуку оптимального локального вирівнювання – досягнення максимального рахунку вирівнювання, причому якнайменше вдаючись до введення розривів. Для цього в алгоритмах вирівнювання прописані правила накладання *штрафів* (scoring penalties або gap scores) за розриви. Штрафи більшого розміру (10–15 одиниць) накладають за введення (відкриття; *G* scores) нового розриву, і меншого (1–2 одиниці) – за продовження наявного розриву (*L* – extension penalties). Таким є принцип *афінних штрафів* (affine penalties), який має мінімізувати частоту відкриття нових розривів (більше про афінні штрафи далі). Отже, сучасні алгоритми вирівнювання генетичних послідовностей – це система правил, метою яких є збалансування процесу пошуку ідентичних і подібних позицій та уведення в них розривів.

Найпростішим підходом до обчислення рахунку вирівнювання є встановлення кількості пар ідентичних літер у вирівняних позиціях (див. рис. 3.1). У цьому випадку всім позиціям, у яких знайдені ідентичні літери, приписують одне значення (наприклад, 1), усім відмінним – інше (0). Такі двійкові системи оцінювання названі *одиничними матрицями рахунків* (unitary scoring matrices; рис. 3.5).

а

	A	T	G	C
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

б

	L	V	K	R	...
L	1	0	0	0	0
V	0	1	0	0	0
K	0	0	1	0	0
R	0	0	0	1	0
...	0	0	0	0	1

в

Білок 1	K	G	G	V	L	T	T	R	S	W
Білок 2	K	G	G	I	L	T	I	R	S	W
Рахунок - 8	1	1	1	0	1	1	0	1	1	1

Рис. 3.5. Одиничні матриці рахунків вирівнювання: *а* – для нуклеотидних послідовностей; *б* – для амінокислотних послідовностей; *в* – приклад вирівнювання двох амінокислотних послідовностей. В частині *б* зображений лише фрагмент матриці (4 амінокислоти) задля спрощення рисунка. Кольором виділені позиції з відмінними амінокислотними залишками

Одиничні матриці ще називають *виродженими* (sparse), адже нуль становить більшість їхніх елементів. Одинична система оцінення є механістичною за змістом (див. вище підрозділ 2.4, с. 73), вона зручна і загалом виправдана у разі аналізу ДНК, яка складається лише з чотирьох літер і яку безпосередньо не задіяно до виконання біохімічних функцій. Як буде докладніше проаналізовано далі, одинична система рахунків основоположна для графічного порівняння двох послідовностей – дотплот-аналізу. Така система оцінювання, однак, має мало біологічного змісту при вирівнюванні білкових послідовностей. Багато амінокислотних заміщень у білках практично не впливає на їхню функцію, а частина змінюватиме заряд чи ступінь гідрофільності. Тому, з точки зору фізичної хімії та біохімії, амінокислоти поділяють на групи подібності. Один із прикладів класифікації амінокислот наведено на рис. 3.6.

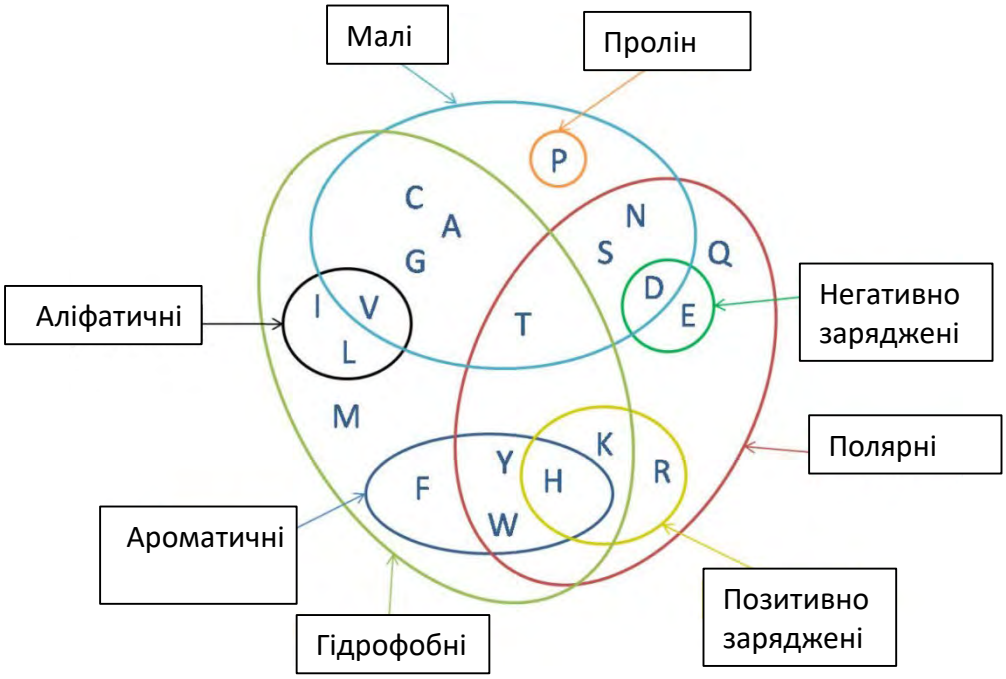


Рис. 3.6. Класифікація амінокислот за фізико-хімічними ознаками: заряд, розмір, гідрофобність, полярність тощо. Амінокислоти позначені однолітерним кодом IUPAC

Отже, якщо у певній позиції вирівнювання знаходяться подібні амінокислоти (в тому сенсі, що описано попередньо), то таким позиціям варто приписати особливе значення – не один і не нуль. В одиничних системах заміщення також не враховані ролі розривів, яким треба присвоювати певну величину (див. попередній текст). Очевидно, що реалістичний підхід до вирівнювання амінокислотних послідовностей і його оцінювання має враховувати фізико-хімічні властивості амінокислот та всі типи розривів неперервної нитки літер-амінокислот, до яких програма вдалася для загального покращення вирівнювання. Наприклад, заміщенню амінокислоти валін (Val; V) на ізолейцин (Ile; I) у вирівняній позиції двох послідовностей потрібно приписати певне значення більше нуля, оскільки обидві амінокислоти малі й аліфатичні (див. рис. 3.6).

3.3. Теорія еволюції амінокислотних послідовностей М. Дейгоф. Зважаючи на всі міркування щодо властивостей амінокислот, можемо диференційовано оцінити незбіги у попарних вирівнюваннях амінокислотних послідовностей. Деякі заміщення можна розглядати як *консервативні*, оскільки вони не змінюють (консервують) структуру чи функцію білка. Введення додаткових значень для консервативних заміщень збільшує чутливість і гнучкість методу вирівнювання, тобто дає змогу виявити гомологію для послідовностей, які доволі складно або й неможливо вирівняти, застосовуючи одиничні (ідентичні/неідентичні) матриці рахунків. Надання диференційованого значення різним типам вирівняних позицій – ідентичні, позиції з консервативним заміщенням, неконсервативні заміщення і розриви – названі *зваженими рахунками* (weighted scores). Крім того, заміщенню, наприклад, Val → Ile, варто приписувати різні значення у випадку вирівнювання споріднених послідовностей (скажімо, людина і миша) і віддалених (людина і дріжджі). Тому для різних типів вирівнювання необхідно мати набір різних *матриць заміщення* (substitution matrices), які даватимуть змогу враховувати тип генетичної зміни й еволюційну відстань між вирівнюваними послідовностями.

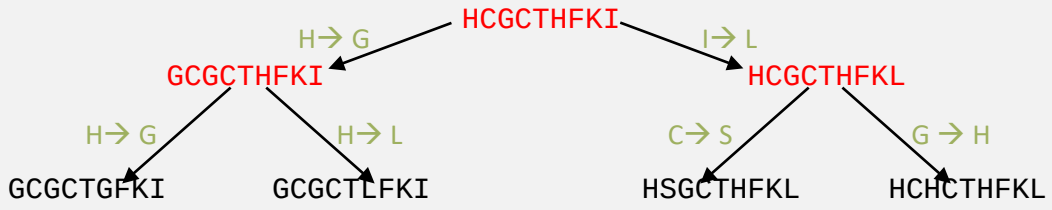
Першу матрицю заміщення розробила 1978 р. Марґарет Дейгоф із Джорджтаунського університетського медичного центру (США). Її можна вважати засновницею біоінформатики. Матрицю обчислено на моделі еволюції під назвою *точкові прийнятні мутації* (Point Accepted Mutation, PAM). Один PAM – це еволюційна відстань, протягом якої змінився один відсоток амінокислотних залишків, тобто на вирівняних 100 залишків можна виявити одну змінену позицію. Це не означає, що через 100 PAM усі амінокислотні залишки зміняться: деякі позиції зазнають багаторазових змін, інші повернуться до вихідного стану, ще інші – залишаться незмінними. Без тиску природного добору на амінокислотні послідовності частоти заміщення (наприклад, амінокислотного залишку *i* на залишок *j*) мали б залежати, передусім, від частот різних амінокислот у протеомі організму – це *фонові частоти* (background frequencies) f_i, f_j. Однак у споріднених білках частоти заміщення, що фактично спостерігають – це *цільові частоти* (target frequencies) заміщення амінокислоти *i* амінокислотою *j* (M_{ij}). Такі заміщення, здебільшого, не порушують (або порушують незначно) функціонування білка. Іншими словами, це точкові мутації, "прийняті" еволюцією. Для визначення цільових частот Дейгоф та співробітники будували *множинні вирівнювання* дуже подібних за амінокислотною послідовністю білків, які можна отримати вручну. Отримані вирівнювання використали для збирання даних про частоти мутації на відстані в один PAM, і ці дані екстраполювали для визначення частот на відстані, наприклад, 250 PAM (шляхом перемножування матриці 1 PAM 250 разів). Іншими словами, теорія еволюції білків М. Дейгоф має *емпіричну* основу: вона не ґрунтується на певній механістичній моделі, описаній певними параметрами (одиничні матриці заміщень), а на експериментальних даних. Докладніше про підхід М. Дейгоф можна дізнатися у блоці 6. Для порівняння послідовностей корисно використовувати матрицю значень, що є відношенням імовірності заміни амінокислоти (цільова частота) до ймовірності про те, що дві амінокислоти у вирівняній позиції знаходяться випадково (фонова частота) – *співвідношення спорідненості* R_{ij} (relatedness odds). Одну з найпопулярніших матриць з цієї серії, PAM250, наведено на рис. 3.7.

БЛОК 6. Механізм обчислення рахунків PAM-матриці

Створюють множинне вирівнювання низки гомологічних амінокислотних послідовностей, які відрізняються не більше ніж на 15 % (тобто на 100 залишків – не більше 15 відмінностей). Такі вирівнювання М. Дейгоф виконувала вручну. Розглянемо побудову PAM-матриці на прикладі такого вирівнювання:

```
GCGCTGFKI
GCGCTLFKI
HSGCTHFKL
HCHCTHFKL
```

Для цього вирівнювання будують філогенетичне дерево. Філогенетичну реконструкцію розглянемо у наступних розділах. Наразі потрібно лише знати, що у цьому випадку дендрограма (дерево) показує порядок виникнення амінокислотних заміщень, які описують шлях появи послідовностей-нащадків вихідної послідовності:



У цьому дереві в основі розміщено 4 послідовності, з яких складалося вихідне множинне вирівнювання; їх згруповано по два, згідно з найощаднішим сценарієм еволюції, який включає три предківські послідовності (червоний колір) і низку амінокислотних заміщень (сині написи на стрілках). Зауважимо, що предківські послідовності – суто гіпотетичні, їхній вибір ґрунтується на певному уявленні (моделі) про характер еволюції заданого масиву послідовностей.

Для кожного амінокислотного залишку визначають частоту його заміщення на всі інші можливі залишки (A_{ij}). Припускають, що усі заміщення рівноймовірні в обидвох напрямах, тобто H→G будемо рахувати як G→H. Тому для обчислення A_{HG} беремо до уваги гілки із заміщеннями H→G й G→H. Для цього дерева: A_{HG} = 3, A_{HL} = 1.

Далі обчислюють відносну мутабільність амінокислотного залишку (m_j). Це співвідношення кількості разів його заміщення на будь-який інший залишок у філогенетичному дереві до загальної кількості заміщень, які могли б виникнути. Цей знаменник обчислюють як добуток загальної кількості заміщень у дереві на два, на фонову частоту залишку f_j і на коефіцієнт 100. Наприклад, в описаному дереві є чотири заміщення H. Цю кількість розділяємо на добуток подвоєної кількості всіх можливих мутацій (6×2=12), на фонову частоту зустрічності H у масиві даних (10 із 63-х залишків (див. дерево = 0,159)) і на 100. Так, m_H = 4/(12×0,159×100) = 0,0209.

Обчислюють цільову частоту заміщення $M_{ij} = m_j \times A_{ij}/(\Sigma A_{ij})$. Наприклад, $M_{G,H}$= 0,0209×3/4 = 0,0156. Знаменник у формулі (ΣA_{ij}) – сума всіх заміщень за участю H.

Обчислюють PAM-рахунки як log-odds-величини співвідношення M_{ij}/f_i (R_{ij}), де f_i – фонова частота i (0,159 для H, див. вище). Тоді $R_{G,H}$ = 0,0156/0,1587, його lg = -1,01. R_{ij} використовують у ф-лі (22) (див. осн. текст нижче) для обчислення PAM Score. Для діагональних значень: $M_{jj} = 1 - m_j$, і R_{jj} (відсутність мутації) обчислюють за описом вище.

C	12																			
S	0	2																		
T	-2	1	3																	
P	-3	1	0	6																
A	-2	1	1	1	2															
G	-3	1	0	-1	1	5														
N	-4	1	0	-1	0	0	2													
D	-5	0	0	-1	0	1	2	4												
E	-5	0	0	-1	0	0	1	3	4											
Q	-5	-1	-1	0	0	-1	1	2	2	4										
H	-3	-1	-1	0	-1	-2	2	1	1	3	6									
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6								
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5							
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6						
I	-2	-1	-	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5					
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6				
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4			
F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9		
Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10	
W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W

Рис. 3.7. Емпірична матриця PAM250 М. Дейгоф, побудована на основі мутаційних даних. Амінокислоти зібрано у групи з урахуванням їхніх фізико-хімічних даних (різні групи розділено лініями). Додатні значення позначають еволюційно консервативні заміни. Зауважте часову зворотність (time reversibility, див. підрозділ 2.4 вище) матриці, тобто рахунки заміщення $i \rightarrow j$ = = $j \rightarrow i$; саме тому подана лише половина значень (нижче діагоналі)

Остаточна формула для обчислення значень (*рахунків*) PAM-матриць така:

$$PAM\ Score = 10 \times \lg R_{ij} = 10 \times \lg \frac{M_{ij}}{f_i}, \qquad (22)$$

де R_{ij} – співвідношення спорідненості – цільової частоти заміщення амінокислоти i на амінокислоту j, до фонової частоти i у певному протеомі, як докладніше пояснено у блоці 6. Коефіцієнт 10 у формулі необхідний для перетворення значень логарифма цього співвідношення у ціле число.

Коли одну амінокислотну послідовність вирівнюють з іншою, то значення часток співвідношень усіх позицій перемножують для обчислення загального рахунку вирівнювання. Однак арифметично простіше оперувати логарифмами часток, які можна додавати. Тому значення, зібрані у сучасних матрицях заміщення – це логарифм (log) відношення цільових частот до фонових частот (*log odds*, або *відносна вірогідність*). У PAM250 додатні значення відповідають заміщенням, що в межах теорії PAM прийнятні як мутації, тобто ймовірність знайти відповідні залишки в одній позиції вирівнювання вища за випадкову; від'ємні значення приписують парам залишків, імовірність знаходження яких в одній позиції вирівнювання менша за випадкову (неприйнятні мутації). Діагональні елементи (виділені червоним) матриці мають найбільше значення, тобто амінокислотний залишок з найвищою ймовірністю (яка вища за випадкову) може вирівнюватися із самим собою. Також у матриці наявні нульові значення – їх приписують парам амінокислотних залишків, імовірність натрапити на які в одній позиції вирівнювання на рівні випадковості (нейтральні мутації).

Для розуміння біологічного змісту різних значень PAM250 опишемо співвідношення спорідненостей (вірогідність) R_{ij} у рівнянні (22) як функцію PAM-рахунку (PAM Score) S:

$$R_{ij} = \frac{M_{ij}}{f_i} = 10^{\frac{S}{10}}. \qquad (23)$$

Найбільше значення S у PAM250 відповідає збереженню залишку триптофану (W) – 17. З рівняння (23) очевидно, що значення $R_{WW} = 50{,}11$. Тобто на заданій еволюційній відстані (250 PAM) імовірність натрапити на пару залишків триптофану в одній позиції вирівнювання у 50,11 раза вища за випадкову. Іншими словами, ймовірність двох залишків триптофану в одній позиції вирівнювання у 50 разів вища для гомологічних послідовностей, ніж для випадкових. Такі високі значення рахунку і R_{ij} відображають дуже низьку мутабільність триптофану, що, своєю чергою, частково пояснюють фізико-хімічною унікальністю триптофану й рідкісністю цієї амінокислоти в протеомі (див. блок 6). Амінокислоти з високою мутабільністю, такі як серин, мають невелике діагональне значення. Це означає, що серин трапляється у вирівнюваннях гомологічних послідовностей трохи частіше, порівняно з випадковими послідовностями. Додатне значення (3) має заміщення D↔E (див. рис. 3.7). Відповідно до рівняння (23), імовірність натрапити на вирівняну пару залишків D-E вдвічі більша для гомологічних послідовностей, ніж для випадкових. Це відображає близькість фізико-хімічних параметрів аспартату й глутамату (обидві – негативно заряджені амінокислоти, що містять кислотну карбоксильну групу) і їхню досить-таки високу поширеність у протеомі. Аналогічно, високі рахунки мають такі пари амінокислотних залишків: Y↔F; K↔R; V↔I. У всіх випадках це пов'язують із подібністю фізико-хімічних властивостей. Парам амінокислот, що мають відмінні властивості, притаманні від'ємні значення. Триптофан (W) і цистеїн (C) – дві амінокислоти, що еволюціонують найповільніше. Триптофан – найбільша амінокислота, з дуже громіздкою бічною групою. Заміщення W іншою бічною групою, ймовірно, спричинить суттєву зміну у просторовій будові білка. Тоді це шкідлива мутація,

яку буде усунуто добором. Залишки цистеїну здатні утворювати дисульфідні містки, C-S-S-C, між різними частинами поліпептидних ланцюгів. Руйнування цих містків також спричинятиме суттєві зміни у білковій структурі. Як результат, фактичні частоти заміщень за участю W і C – невеликі. Ці амінокислоти мають великі додатні діагональні рахунки (консервації) і від'ємні рахунки вирівнювання майже з усіма іншими амінокислотами.

Наведені результати і значення частот заміщення справедливі для заданої еволюційної відстані. Розглянемо рисунок 3.8, де відображені частини двох PAM-матриць, в яких наведені частоти амінокислотних заміщень на різних еволюційних відстанях. Так, в PAM40 наведені частоти для послідовностей, що містять 40 прийнятних мутацій на 100 залишків, а PAM250 підібрана для послідовностей, які мають 250 прийнятних мутацій на 100 залишків (такі послідовності залишаються на 20 % подібними, накопичують багато прямих і зворотних мутацій – звідси і дещо парадоксальне твердження про 250 мутацій на 100 залишків). PAM40 і PAM250 дуже відрізняються відносними рахунками консервації (діагональні значення) і заміщення залишків. Наприклад, заміщення, які в PAM40 розглядають як невірогідні (наприклад, R→N; $S_{N,R}$ = -7), вважають нейтральними ($S_{N,R}$ = 0) у PAM250. Аналогічно, частоти заміщень, які простежуються рідше за випадок на відстані 40 PAM – 40 % мутаційних змін (наприклад, $S_{I,L}$ = -1), вже перевищуватимуть випадкову частоту ($S_{I,L} = 2$) на відстані 250 PAM (250 % змін).

а

	A	R	N	D	E	I	L
A	8						
R	-9	12					
N	-4	-7	11				
D	-4	-13	3	11			
E	-3	-11	-2	4	11		
I	-6	-7	-7	-10	-7	12	
L	-8	-11	-9	-16	-12	-1	10

б

	A	R	N	D	E	I	L
A	2						
R	-2	6					
N	0	0	2				
D	0	-1	2	4			
E	0	-1	1	3	4		
I	-1	-2	-2	-2	-2	5	
L	-2	-3	-3	-4	-3	2	6

Рис. 3.8. Рахунки збереження (діагональ) і заміщення (недіагональні значення) амінокислотних залишків у матрицях PAM40 (*а*) та PAM250 (*б*)

На величинах PAM-рахунків позначається і структура генетичного коду. Наприклад, одні амінокислотні заміщення відбуваються за рахунок заміщення одного нуклеотиду в кодоні, тоді як інші потребують більше однієї зміни. Можна очікувати, що амінокислотні заміщення, які ґрунтуються на однонуклеотидних змінах у кодоні, траплятимуться частіше, ніж ті, котрі потребують двох чи трьох нуклеотидних замін. Дійсно, немає таких заміщень, які часто трапляються і яких не можна досягти за рахунок однонуклеотидної зміни (рис. 3.9). Також є амінокислотні заміщення, які трапляються нечасто, хоча вони можливі за рахунок однонуклеотидної зміни. Отже, ефект генетичного коду на частоти амінокислотних заміщень урівноважується природним добором.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	2	247	216	386	106	208	600	1183	46	173	257	200	100	51	901	2413	2440	11	41	1766
R	–1	5	116	48	125	750	119	614	446	76	205	2348	61	16	217	413	230	109	46	69
N	0	0	3	1433	32	159	180	291	466	130	63	758	39	15	31	1738	693	2	114	55
D	0	–1	2	5	13	130	2914	577	144	37	34	102	27	8	39	244	151	5	89	127
C	–1	–1	–1	–3	11	9	8	98	40	19	36	7	23	66	15	353	66	38	164	99
Q	–1	2	0	1	–3	5	1027	84	635	20	314	858	52	9	395	182	149	12	40	58
E	–1	0	1	4	–4	2	5	610	41	43	65	754	30	13	71	156	142	12	15	226
G	1	0	0	1	–1	–1	0	5	41	25	56	142	27	18	93	1131	164	69	15	276
H	–2	2	1	0	0	2	0	–2	6	26	134	85	21	50	157	138	76	5	514	22
I	0	–3	–2	–3	–2	–3	–3	–3	–3	4	1324	75	704	196	31	172	930	12	61	3938
L	–1	–3	–3	–4	–3	–2	–4	–4	–2	2	5	94	974	1093	578	436	172	82	84	1261
K	–1	4	1	0	–3	2	1	–1	1	–3	–3	5	103	7	77	228	398	9	20	58
M	–1	–2	–2	–3	–2	–2	–3	–3	–2	3	3	–2	6	49	23	54	343	8	17	559
F	–3	–4	–3	–5	0	–4	–5	–5	0	0	2	–5	0	8	36	309	39	37	850	189
P	1	–1	–1	–2	–2	0	–2	–1	0	–2	0	–2	–2	–3	6	1138	412	6	22	84
S	1	–1	1	0	1	–1	–1	1	–1	–1	–2	–1	–1	–2	1	2	2258	36	164	219
T	2	–1	1	–1	–1	–1	–1	–1	–1	1	–1	–1	0	–2	1	1	2	8	45	526
W	–4	0	–5	–5	1	–3	–5	–2	–3	–4	–2	–3	–3	–1	–4	–3	–4	15	41	27
Y	–3	–2	–1	–2	2	–2	–4	–4	4	–2	–1	–3	–2	5	–3	–1	–3	0	9	42
V	1	–3	–2	–2	–2	–3	–2	–2	–3	4	2	–3	2	0	–1	–1	0	–3	–3	4

Рис. 3.9. Матриця амінокислотних заміщень Джонса-Тейлора-Торнтона (JTT) у вигляді рахунків log odds (S_{ij}; нижче діагоналі) та кількість заміщень A_{ij}, які фактично спостерігали у масиві даних. Рахунки обчислювали за формулою S_{ij};= $10\log_{10}R_{ij}$ й округлювали до найближчого цілого значення. Клітинки з сірим фоном мають додатне значення. Це означає, що амінокислоти, на перетині яких розміщена клітинка, взаємозамінні з вищою за випадковість імовірністю. Значення, написані білим на чорному фоні, відповідають амінокислотним заміщенням, можливим унаслідок однонуклеотидного заміщення в одній позиції кодону

Використання PAM-матриць на практиці (наприклад, у складі певних комп'ютерних програм) означає, що програма уникатиме вирівнювання позицій з великими від'ємними значеннями, які зменшують рахунок вирівнювання. Матриця PAM250 дає змогу вирівнювати послідовності, в яких залишилося 20 % подібних позицій. PAM-матриці з нижчим значенням (PAM1, PAM10, PAM100) можна використовувати для вирівнювання більш подібних послідовностей.

З часу опублікування PAM-матриць минуло понад 30 років, і за цей час неодноразово намагалися оновити їхні значення з урахуванням значно більшого масиву амінокислотних послідовностей. Втім, новіші PAM-матриці, такі як JTT (див. рис. 3.9), незначно відрізняються від вихідних.

Матриці, засновані на моделі М. Дейгоф, обраховані на основі порівняння послідовностей, що виявляють щонайменше 85 % ідентичності. Очевидно, що це гомологічні послідовності, які легко вирівнювати вручну. Для такого типу послідовностей інтуїтивно виправдане застосування найпростішого принципу еволюції – максимальної ощадності (див. блок 6). Тому PAM-матриці добре описують процес заміщення у подібних білках. Еволюційний принцип, побудований за методом М. Дейгоф – велика перевага над суто механістичними підходами, яка забезпечила тривку популярність PAM-матриць. Однак існує низка особливостей PAM-матриць, що обмежують їхнє використання. Нині вирівнювання виконують, здебільшого, над суттєво відмінними послідовностями, для яких частоти заміщень непрямо випливають із матриць мутаційних даних М. Дейгоф (їх визначають за допомогою перемноження

PAM1). Отже, припущення теорії PAM можуть не справджуватися для філогенетично віддалених білків. Кожна матриця амінокислотних заміщень містить 200 значень. Однак в оригінальному масиві даних, на основі якого М. Дейгоф обчислювала цільові частоти, не було майже 40 типів заміщень, тому відповідні частоти екстраполювали з подібних заміщень. Потрібно розуміти, що свої обмеження має кожна математична модель. Фактично будь-яка матриця найкраще описує процес еволюції лише тих послідовностей, які використали для її побудови.

3.4. BLOSUM та інші матриці амінокислотних заміщень. Мабуть найпопулярнішою серією матриць амінокислотних заміщень стали BLOSUM (BLOcks SUbstitution Matrix). Вони позбавлені низки обмежень, притаманних PAM-матрицям. Їх створили Стівен Генікоф і Хорхія Генікоф з Центру ракових досліджень Фреда Гатчінсона (Сіетл, США) 1992 року. Як і PAM, BLOSUM базуються на логарифмі співвідношення вірогідностей. Матрицю рахунків виводять безпосередньо із вирівнювань, не посилаючись на філогенетичне дерево чи певну еволюційну модель.

Для побудови BLOSUM-матриць використано базу Blocks – множинні вирівнювання фрагментів білків. Єдиним критерієм відбору білків до BLOCKS стала їхня здатність до досконалого вирівнювання з іншими білками, незалежно від їхньої філогенетичної спорідненості. Отримані вирівнювання аналізують, що описано згодом і підсумовано на рис. 3.10.

а

```
    αβγΔλπφω
s1  TLKKVQKT
s2  TLKKVQKT
s3  TLKKIQKQ
s4  IITKLQKQ
s5  TITKLQKQ
s6  TLTKIQKQ
s7  TLTQIQKQ
```

$A_{ij} = N_i \times N_j$

$\gamma_{KK} = 3 \times 2 = 6$

$\Delta_{KK} = 6 \times 5 = 30$

$\varphi_{KK} = 7 \times 6 = 42$

$A_{KK} = \gamma_K + \Delta_K + \varphi_K = 78$

б

	I	K	L	Q	T	V
I	8	-	16	-	6	6
K	-	78	-	6	12	-
L	16	-	22	-	-	4
Q	-	6	-	62	10	-
T	6	12	-	10	44	-
V	6	-	4	-	-	2

Рис. 3.10. Отримання вихідних даних для побудови матриць BLOSUM. Вихідне множинне вирівнювання (*а*) містить сім послідовностей (s1-s7) і вісім позицій вирівнювання, α – ω. Принцип обчислення A_{KK} поданий справа від вирівнювання. Матрицю значень A_{ij} подано внизу (*б*)

Спочатку визначають, скільки разів кожна амінокислота трапляється у вирівнюванні. Отримують частоту трапляння π_i кожної амінокислоти у масиві даних. Далі визначають кількість вирівнювань певної амінокислоти з кожною іншою. На рис. 3.10, *а* (див. вище) вирівняно сім послідовностей; є 42 способи вибрати пару амінокислотних залишків з однієї колонки вирівнювання (7×6=42). Сьома колонка вирівнювання містить лише залишки лізину (К). Вона матиме значення 42 для A_{KK}. У першій колонці вирівнювання маємо шість Т і один І. Ця колонка матиме таке значення: 30 для A_{TT} , шість для A_{TI} і шість для A_{IT}. У результаті отримаємо матрицю значень A_{ij} (див. рис. 3.10, *б*).

Розглянутий підхід до обчислення матриці A_{ij} для BLOSUM відрізняється від способу отримання PAM-матриць тим, що для першої визначають кількість пар вирівняних амінокислот, а не кількість заміщень (див. блок 6). З огляду на сказане діагоналі матимуть ненульові значення. Також деякі вирівняні пари залишків, які в теорії PAM (відповідно до принципу максимальної ощадності) не були б прямими заміщеннями, вважають як такі в BLOSUM. Отже, одне множинне вирівнювання методи PAM і BLOSUM тлумачать по-різному, і це відобразиться у різних матрицях рахунків.

Далі встановлюємо частку вирівняних пар – цільову частоту – залишків типу ij:

$$q_{ij} = \frac{A_{ij}}{A_{tot}}, \tag{24}$$

де A_{tot} – сума всіх елементів у матриці A_{ij} (див. рис. 3.10, *б*).

Звідси можна безпосередньо встановити відношення цільової частоти пар порівняно з тим, чого очікують у вирівнюваннях випадкових послідовностей з ідентичною частотою амінокислотних залишків (такі послідовності найпростіше отримати за допомогою переставляння залишків):

$$R_{ij} = \frac{q_{ij}}{\pi_i \pi_j}. \tag{25}$$

Значення R_{ij} – *відносна вірогідність*, або odds – однакове у формулах (24), (25) і у блоці 6. Ці значення можна використати для обчислення log odds (текст далі щодо формули обчислення), які потраплять у матрицю рахунків (рис. 3.11). Наведемо формулу обчислення рахунків:

$$S_{ij} = 2\log_2 R_{ij}. \tag{26}$$

Математика обчислень у цьому методі значно простіша, ніж у PAM, тому що тут ідуть від даних прямо до матриці рахунків, оминаючи формулювання еволюційної моделі (M_{ij}, див. блок 6). Водночас BLOSUM має низку обмежень. Тут на рахунки матриці впливає наявність груп дуже споріднених послідовностей

у множинному вирівнюванні. Ці послідовності будуть майже однакові і вони робитимуть великий внесок до діагональних рахунків матриці A_{ij}. На противагу, групи майже однакових послідовностей не впливають на метод PAM, оскільки вони опиняються дуже близько одна до одної на філогенетичному дереві (див. блок 6) і, за принципом ощадності, їх описують незначною кількістю заміщень (вони потрапляють у матриці PAM). Реальні біологічні дані досі отримують доволі фрагментарно: є значна кількість послідовностей з кількох добре вивчених груп організмів, і незначна – з решти таксонів. Тому необхідно опрацювати систематичний підхід, який ураховуватиме нерівномірність розподілу еволюційних відстаней між вирівнюваннями, що є основою концепції побудови матриць родини BLOSUM.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Рис. 3.11. Матриця BLOSUM62 з наведеними рахунками S_{ij}, обчисленими за формулою (26)

Як уже зазначено, С. Генікоф та Х. Генікоф використовували вирівнювання доменів білків з бази Blocks: надійно вирівняні ділянки (часто домени) білків без розривів. У межах кожного вирівнювання послідовності згруповані в “пучки”

(*кластери*), що мали відсоток ідентичності, більший від певного порогового значення (наприклад, 80 % чи 62 %). Під час підрахунку пар вирівняних амінокислотних залишків для матриці A_{ij} послідовності з одного кластера не враховували. У разі підрахунку пар між кластерами послідовності з одного кластера *зважували* – рахували як одну послідовність. Якщо у кластері є дві послідовності, то кожна матиме половину ваги однієї відмінної послідовності (рис. 3.12). Матрицю значень log odds, яка походить з кластера послідовностей, що мають відсоток ідентичності не більше за заданий, позначать цим відсотком. Відповідно, BLOSUM80 побудовано на основі обрахунку заміщень між послідовностями, що мали не більше 80 % подібності, а BLOSUM62 – на основі послідовностей, що мали менше 62 % подібності. Менше число BLOSUM відображає нижчу подібність, а менше число PAM – більшу подібність.

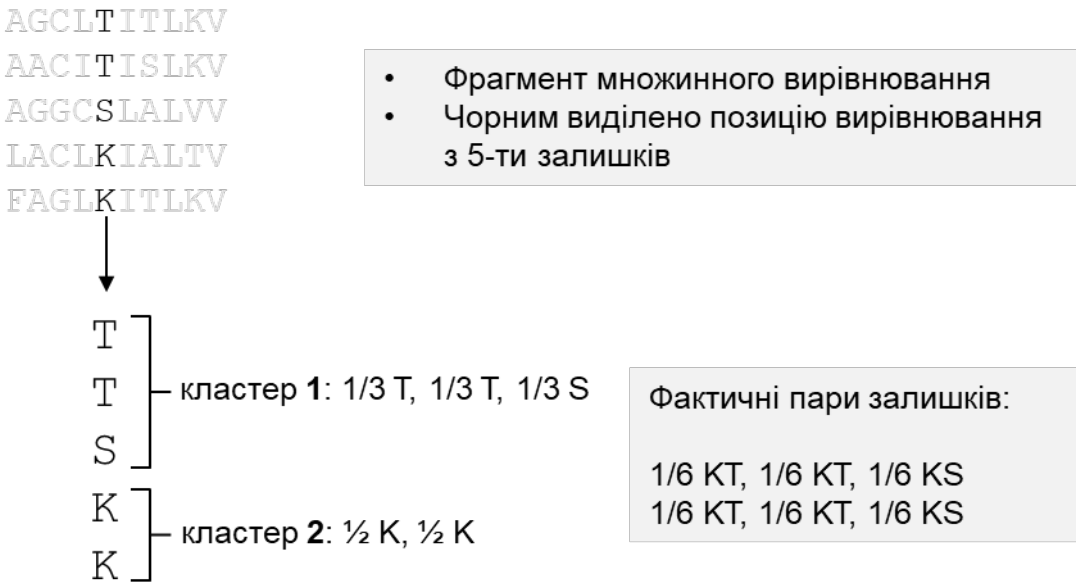


Рис. 3.12. Принцип зважування послідовностей під час побудови BLOSUM-матриць. Послідовності кластеризують у блоки, щоб зменшити надмірний внесок послідовностей із найспорідненішої родини до частот амінокислотних пар. Кластеризування виконують за відсотком ідентичності, тобто всі послідовності в межах блока матимуть, наприклад, понад 80 %. Амінокислотні залишки у кожному кластері зважують за розміром кластера – 1/*c*, де *c* – кількість послідовностей у кластері

Коефіцієнт унормування рахунків до цілого числа (scaling factor; число зліва від логарифма), що його використовують в BLOSUM62 – $S(i,j) = 2\log_2 R_{ij}$. Тому рахунки BLOSUM62 у "півбітних" одиницях. Отже, одиниця ($S(i,j) = 1$) в описаній матриці містить 0,5 біта інформації. Якщо рахунок $S(i,j)$ дорівнює одиниці, то відносна вірогідність $R_{ij} = 2^{1/2} = \sqrt{2}$. Найбільший рахунок у BLOSUM62 – S(W,W) = 11. Звідси $R_{WW} = 2^{11/2} = 45{,}2$ ($R_{ij} = 2^{Sij/2}$). Це не надто відрізняється від результату, який отримано для матриці PAM250.

Зазначимо також, що в різних матрицях коефіцієнт унормування різний, як і їхній інформаційний вміст. Наприклад, значення рахунків у матриці PAM250 подані в 1/3 біта (адже $10\log_{10} \approx 3\log_2$); такий самий вміст матриці BLOSUM50.

Розуміння інформаційного вмісту важливе, коли дослідник хоче порівняти ступінь подібності певних послідовностей, використовуючи різні матриці. У цьому випадку значення рахунків вирівнювання треба перерахувати на один біт інформації, тобто отримати *біт-рахунки* (bit scores), які можна прямо порівнювати. Здебільшого сучасні програми вирівнювання (такі як BLAST, далі про це у розділі) враховують відмінності в інформаційному вмісті різних матриць і подають біт-рахунок за замовчуванням.

Зі сказаного очевидно, що кожна матриця рахунків матиме власний *інформаційний вміст* (відносну ентропію, див. підрозділ 1.1) у перерахунку на одну вирівняну позицію – або "*біт-на-позицію*" (БНП). За допомогою БНП можна передбачити кількість вирівняних залишків, яка необхідна для досягнення статистично значущого рахунку вирівнювання (про статистику попарних вирівнювань далі у розділі). "Мілкі" матриці (PAM10/VTML10, PAM20/VTML 20, PAM40/VTML40 – ті, що побудовані на основі білків, розділених невеликою еволюційною відстанню) мають вищий інформаційний зміст, ніж "глибокі" матриці (BLOSUM62, PAM250). Це означає, що коротше вирівнювання (від 10 до 50 залишків) на основі VTML20 може генерувати статистично значний рахунок. "Мілкі" матриці підходять до пошуку коротких доменів (екзонів); вони мають тенденцію продукувати вирівнювання із вищою ідентичністю, бо приписують вищі додатні рахунки ідентичним парам і більші від'ємні – заміщенням (рис. 3.13). Глибокі матриці – чутливий метод пошуку подібності, але потребують довших вирівнювань, й інколи ведуть до надлишкового видовження вирівнювань (overextension).

VTML 20

	A	**R**	**N**	**D**	**C**	**Q**	**E**
A	7						
R	-7	8					
N	-6	-5	8				
D	-6	-12	-1	8			
C	-3	-7	-8	-14	12		
Q	-5	-2	-4	-4	-13	9	
E	-5	-10	-5	-1	-14	-1	7

BLOSUM 62

	A	**R**	**N**	**D**	**C**	**Q**	**E**
A	4						
R	-1	5					
N	-2	0	6				
D	-2	-2	1	6			
C	0	-3	-3	-3	9		
Q	-1	1	0	0	-3	5	
E	-1	0	0	2	-4	2	5

Рис. 3.13. Фрагменти "мілкої" (VTML20) та "глибокої" (BLOSUM62) матриць у півбітному масштабі. VTML20 продукує рахунок ідентичності (збігу) в середньому 2,80 півбіта на позицію, а рахунок заміщення (незбігу) – -0,59 півбіта. BLOSUM62 дає 1,86 за збіг, і лише -0,06 – за заміщення. Отже, оптимальною мішенню для VTML20 є коротші вирівнювання з великою кількістю збігів, оскільки ця матриця надто штрафує незбіги

Можна підібрати BLOSUM-/PAM-матрицю, яку налаштовано на аналіз амінокислотних послідовностей різного ступеня ідентичності (рис. 3.14). Так, PAM10÷PAM40 підійдуть для аналізу високоподібних послідовностей, а PAM200÷PAM250 – для послідовностей, що мають 20÷25 % ідентичності. Аналогічно, серія BLOSUM-матриць містить такі, що приписують високі значення ідентичним парам залишків (BLOSUM80), і такі, що орієнтовані на віддалені послідовності, як-от BLOSUM45÷50.

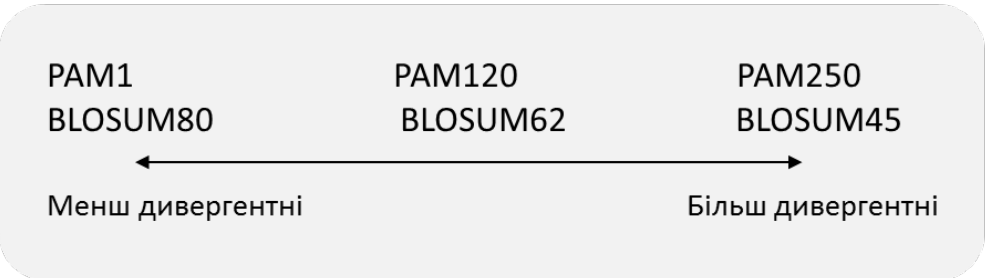


Рис. 3.14. Взаємовідносини між PAM- і BLOSUM-матрицями

Попри подібність, на перший погляд, PAM і BLOSUM, біологічне значення "мілкої" матриці PAM20 суттєво відрізняється від дуже консервативних значень заміщення у матриці BLOSUM80. PAM20 – це набір рахунків для послідовностей, що змінилися лише на 20 % – рівень зміни, очікуваний між білками людини і миші, наприклад. Матрицю BLOSUM80 налаштовано на найконсервативніші райони білків, які залишаються на 80 % ідентичні у межах двох послідовностей, що загалом можуть мати не більше 30 % ідентичності. Тому саме "мілкі" PAM-матриці, а не BLOSUM80, підходять для аналізу білків на коротких еволюційних відстанях. Таблиця 3.1 у порівняльний спосіб підсумовує основні риси матриць PAM і BLOSUM.

Таблиця 3.1

Порівняльний опис PAM- і BLOSUM-матриць

• PAM зорієнтовано на глобальні вирівнювання споріднених білків	• BLOSUM зорієнтовано на локальні вирівнювання
• PAM1 – матриця, вирахувана з порівнянь послідовностей, що мали не більше 1 % дивергенції	• BLOSUM62 – матриця, вирахувана з порівнянь послідовностей, що мали не менше 62 % дивергенції
• Усі PAM-матриці екстраполюють з PAM1	• Усі BLOSUM засновані на експериментальних даних
• PAM-матриці засновані на експліцитній (явно описаній, застосованій) еволюційній моделі, їх можна застосувати у філогенетиці	• BLOSUM не мають еволюційної основи; їх, зазвичай, не використовують у філогенетиці

Отже, розглянуті серії матриць дають початкове уявлення про принципи оцінювання амінокислотних заміщень у вирівнюваннях. Сьогодні відомо декілька сотень різноманітних рахункових матриць заміщень, які спеціально підібрані на певних масивах даних, наприклад, лише на білках вірусного

походження (див. doi: 10.3389/fgene.2015.00319). Очікують, що такі матриці дуже точно описуватимуть особливості амінокислотних заміщень (еволюцію) певної групи білків (наприклад, вірусних). Підсумуємо деякі загальні ознаки всіх підходів до опису амінокислотних заміщень; термінологію, яку використано далі, докладніше описано вище, в підрозділі 2.4. Матриці, що базуються на аналізі великих масивів експериментальних даних (на первинних структурах – послідовностях – справжніх білків), називають емпіричними (такими є PAM і BLOSUM). Інший підхід до опису амінокислотних заміщень полягає у використанні певних параметрів генетичного коду чи фізико-хімічних властивостей (наприклад, гідрофобності, розміру, полярності тощо) амінокислотних залишків, за допомогою якого різним парам залишків можна було б приписувати відмінні рахунки. Це параметризовані, або механістичні, моделі заміщень амінокислот. Приклад параметризованої матриці амінокислотних заміщень наведено на рисунку 3.15.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	0																			
R	2	0																		
N	2	2	0																	
D	1	2	1	0																
C	2	1	2	2	0															
Q	2	1	2	2	2	0														
E	1	2	2	1	2	1	0													
G	1	1	2	1	1	2	1	0												
H	2	1	1	1	2	1	2	2	0											
I	2	1	1	2	2	2	2	2	2	0										
L	2	1	2	2	2	1	2	2	1	1	0									
K	2	1	1	2	2	1	1	2	2	1	2	0								
M	2	1	2	2	2	2	2	2	2	1	1	1	0							
F	2	2	2	2	1	2	2	2	2	1	1	2	2	0						
P	1	1	2	2	2	1	2	2	1	2	1	2	2	2	0					
S	1	1	1	2	1	2	2	1	2	1	1	2	2	1	1	0				
T	1	1	1	2	2	2	2	2	2	1	2	1	1	2	1	1	0			
W	2	1	2	2	1	2	2	1	2	2	1	2	2	2	2	1	2	0		
Y	2	2	1	1	1	2	2	2	1	2	2	2	3	1	2	1	2	2	0	
V	1	2	2	1	2	2	1	1	2	1	1	2	1	1	2	2	2	2	2	0

Рис. 3.15. Параметризована матриця заміщень. Значення матриці – мінімальна кількість нуклеотидних заміщень, необхідна для переходу від однієї амінокислоти (її кодону) до іншої

Теорема Альтшуля постулює, що принцип обчислення PAM- і BLOSUM-матриць – через співвідношення цільових до фонових частот – є загальним. А саме: будь-яку матрицю, що підходить для локальних вирівнювань, або прямо обчислюють із цільових частот q_{ij} і фонових частот (f_i, f_j), або ж ці частоти неявно закладені у матриці і їх можна вирахувати із рахунків заміщень s_{ij}. Останній варіант стосується параметризованих моделей заміщення. Тоді:

$$\lambda s_{ij} = \log\left(\frac{q_{ij}}{f_i f_j}\right), \qquad (27)$$

де λ – коефіцієнт унормування (приведення рахунків попарних вирівнювань s до цілого числа).

На рис. 3.16 проілюстровані попарні вирівнювання, виконані на основі двох різних матриць заміщення – BLOSUM62 і PAM250. Для вирівнювання використані два подібні білки – LanI (позначено як Query) та LndI (Sbjct; 63 % ідентичних амінокислот). Очевидно, що еволюційні відносини між різними амінокислотними залишками різні з точки зору різних матриць. Наприклад, у п'ятій позиції вирівнювання в LanI розміщений залишок треоніну (T), в LndI – аланіну (A). З точки зору BLOSUM62, T/A – неконсервативне заміщення, яке вважають неприйнятною мутацією, тоді як PAM250 розцінює T/A як консервативну заміну, і позначає цю позицію знаком + (подібні амінокислоти). Водночас обидві матриці дають змогу знайти однакову кількість ідентичних позицій. Відмінності у вирівнюванні на основі двох описаних матриць виглядають тривіальними тому, що для вирівнювання вибрані дві високоподібні послідовності. У разі вирівнювання дуже відмінних послідовностей навіть незначні відмінності у стратегії оцінювання консервативних і неконсервативних заміщень можуть допомогти виявити найслабші сигнали подібності, що надзвичайно важливо під час аналізу геномних баз даних.

Матриці заміщення створені для аналізу амінокислотних послідовностей, аналогічні еволюційні моделі можна застосувати і до нуклеотидних послідовностей. Зауважимо: якщо певна нуклеотидна послідовність кодує білок, то майже завжди доречніше порівняти продукти трансляції генів, ніж гени. Причина полягає в тому, що навіть після незначних еволюційних змін послідовності ДНК (складаються лише з чотирьох літер) під час порівняння на основі простих матриць заміщення нуклеотидів містять менше інформації, ніж їхні продукти трансляції (20 літер). Якщо ж мета дослідника – порівняння некодувальних послідовностей (регуляторні РНК, промотори, енхансери, оператори, інтрони тощо), то тут можна застосовувати підходи, розглянуті на прикладах PAM250 і BLOSUM62. Відповідно до певної моделі еволюції, можна обчислити частоти різних типів нуклеотидних замін, і далі застосувати отримані матриці у вирівнюваннях.

а

```
Identities – 146/231 (63 %); similarities – 172/231 (74 %); gaps – 1/231 (0%)

Query  31   VPQRTEQQGQGVLAILAVGSEESGMSELAQGLRRHGYRVEGAATGTKALQVHRNADLVLL  90
            PQ+  +    VL +L V S+      L Q LRRHGYR    ATG KALQ HR+ADLVLL
Sbjct  30   TPQQASRAQNDVLNVLVVESDACSADSLVQKLRRHGYRAASVATGAKALQAHRSADLVLL  89

Query  91   DLDLPDLDGLEVCRNIRASSQTPVIAVTARESELDRVLVLQAGADDCMTKPYGIRELMAR  150
            DLDLPDLDGLEVCR IRA + TPVIA+TAR+SELDRVL LQAGADD M KPYG RELMAR
Sbjct  90   DLDLPDLDGLEVCRAIRADADTPVIAITARDSELDRVLGLQAGADDYMAKPYGFRELMAR  149

Query  151  MDAIMRRIQPQW-PLEQVIEHWPLRIDVNSREVRLDGRLIEVTRKEFDLLHLLASWPDTV  209
            ++A+MRR +PQ   L+QVI H PL ID   RE+RL    ++VTRKEF   +LLAS P ++
Sbjct  150  IEAVMRRFRPQQRALQQVIVHGPLHIDAGKREIRLHQVPVDVTRKEFGPSYLLASHPGSI  209

Query  210  IPRQQLTAQVWGDSVPLRGRTIDTHVSSLRSKLGSSSWIVTVRGVGFRLGN  260
            + R+QL  QVW D     GRTIDTHVSSLR+KLGSS+WI++VRGVGFRLG+
Sbjct  210  VSRKQLMTQVWEDPGSRPGRTIDTHVSSLRNKLGSSNWIISVRGVGFRLGS  260
```

б

```
Identities – 146/231 (63 %); similarities – 191/231 (82 %); gaps – 1/231 (0%)

Query  31   VPQRTEQQGQGVLAILAVGSEESGMSELAQGLRRHGYRVEGAATGTKALQVHRNADLVLL  90
             PQ++ +  ++VL +L V S+  +   L Q LRRHGYR  + ATG+KALQ HR+ADLVLL
Sbjct  30   TPQQASRAQNDVLNVLVVESDACSADSLVQKLRRHGYRAASVATGAKALQAHRSADLVLL  89

Query  91   DLDLPDLDGLEVCRNIRASSQTPVIAVTARESELDRVLVLQAGADDCMTKPYGIRELMAR  150
            DLDLPDLDGLEVCR IRA ++TPVIA+TAR+SELDRVL LQAGADD M+KPYG+RELMAR
Sbjct  90   DLDLPDLDGLEVCRAIRADADTPVIAITARDSELDRVLGLQAGADDYMAKPYGFRELMAR  149

Query  151  MDAIMRRIQP-QWPLEQVIEHWPLRIDVNSREVRLDGRLIEVTRKEFDLLHLLASWPDTV  209
            ++A+MRR++P  Q++L+QVI H PL+ID   RE+RL+   ++VTRKEF+   LLAS P+++
Sbjct  150  IEAVMRRFRPQQRALQQVIVHGPLHIDAGKREIRLHQVPVDVTRKEFGPSYLLASHPGSI  209

Query  210  IPRQQLTAQVWGDSVPLRGRTIDTHVSSLRSKLGSSSWIVTVRGVGFRLGN  260
            ++R+QL +QVW D+ +  GRTIDTHVSSLR+KLGSS+WI++VRGVGFRLG+
Sbjct  210  VSRKQLMTQVWEDPGSRPGRTIDTHVSSLRNKLGSSNWIISVRGVGFRLGS  260
```

Рис. 3.16. Попарне вирівнювання амінокислотних послідовностей (запит – Query; знайдена у базі даних послідовність – Sbjct) з використанням двох різниць матриць заміщення: *а* – BLOSUM62; *б* – PAM250. Загальний результат подібний, але є відмінності у подробицях, а саме: в оціненні того, які амінокислотні заміщення вважати неконсервативними. Консенсусний рядок (між послідовностями, на зеленому тлі в частині *а*) містить інформацію про всі незбіги й збіги. Наприклад, у позиції № 2 в обидвох послідовностях знаходиться пролін (P), тоді в консенсусному рядку програма теж наводить літеру P. Прийнятні (консервативні) заміщення позначені у консенсусному рядку знаком +. Розриви позначені знаком тире (–). Приклади розривів і неконсервативних заміщень зображено на жовтому тлі в частині *б*. Таким позиціям у консенсусному рядку відповідає незаповнена комірка. Identities – позначає кількість (і відсоток) вирівняних позицій з однаковими амінокислотними залишками. Similarities – кількість вирівняних позицій з однаковими і подібними (ті, що позначені знаком +, тобто консервативні заміщення) амінокислотними залишками. Gaps – кількість розривів, уведених програмою у процесі вирівнювання. У цьому випадку програма увела одну прогалину на 231 вирівняну позицію.

3.5. Основні алгоритми попарного вирівнювання

3.5.1. Дотплот-аналіз – графічна репрезентація попарних вирівнювань генетичних послідовностей. *Дотплот* – найпростіший метод попарного вирівнювання, що репрезентує результат вирівнювання у графічному форматі. Розглянемо дві послідовності, А і Б, які можуть мати різну довжину, але, в ідеальному випадку, є доволі подібними за розміром. Створюють прямокутну матрицю, в якій за віссю абсцис відкладають послідовність А, за віссю ординат – Б. У вихідному стані всім коміркам матриці $x_i y_j$ (де i набуває значень від одиниці до значення довжини А (кількості амінокислотних/нуклеотидних залишків), а j – від 1 до значення довжини Б) приписують одну величину, наприклад, 0. Розглядають кожну комірку і приписують їй певне значення, що відображає рівень подібності двох залишків, A_i й B_j, на перетині яких і розміщена комірка $x_i y_j$. У найпростішому випадку всі комірки матимуть значення 0, за винятком тих, де $A_i = B_j$; тоді комірці приписують значення одиниці. Таку матрицю легко візуалізувати у випадку вирівнювання коротких послідовностей, наприклад, замінюючи одиниці певним графічним символом, як зображено на рис. 3.17. Графік, зображений на рисунку, характеризується низкою випадкових символів (“фон” або “шум”) і центральною діагональною лінією (червоні хрестики, затінені сірим кольором), де висока щільність прилеглих символів позначає райони найбільшої подібності між двома послідовностями.

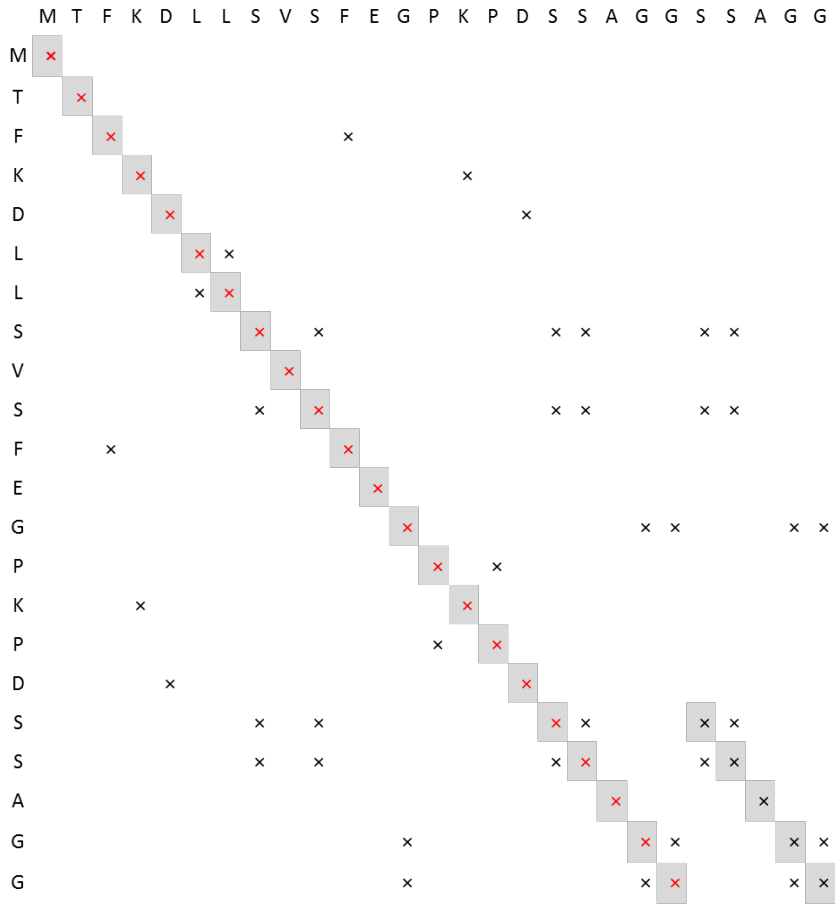


Рис. 3.17. Дотплот-аналіз двох амінокислотних послідовностей. Довга діагональна лінія (червоні хрестики) позначає ідентичні ділянки послідовностей. Чорні хрестики затінені сірим кольором (утворюють відрізок паралельний до основної діагоналі) – дуплікація кінцевої ділянки у послідовності, що на осі абсцис

На графіку “дотплот” двом ідентичним послідовностям притаманна одна неперервна діагональна лінія. Для відображення довгих (50 нт і більше) послідовностей графік необхідно зменшити в розмірах для того, щоб відобразити його повністю. Тоді графічні символи, зображені на рис. 3.17, будуть зменшені до точок (звідси й англійська назва методу – dot-plot), які при достатньому ущільненні графіка зіллються у лінії.

Яку інформацію може здобути дослідник, застосовуючи метод дот-плоту? Розглянемо це на прикладах аналізу декількох нуклеотидних послідовностей за допомогою відкритого веб-сервісу DNAdot (http://www.vivo.colostate.edu/molkit/dnadot/). Для аналізу використана 200-нт послідовність (П1), розміщена у 5′-напрямку від старт-кодону гена *SCO0794 Streptomyces coelicolor* M145. На основі П1 створена низка похідних, які зображено на рис. 3.18. А саме: отримано повністю інвертовану послідовність П1 з внутрішньою інверсією чи дуплікацією, термінальною дуплікацією. П1 попарно вирівняно із кожною з попередньо описаних послідовностей за допомогою DNAdot.

1. Промотор sco0974p, 200 п.н. (**П1**)

AGTGCTCTCACGGGAGCCTCCCCTGGGTCAAAGGAACCTAGGATCTCTAGGTTTCCGCCTAGCGT
AGATAGCCTACGGGTGAAATCGCCAGCGGATTTCGGGGAGGCGAGTTCATAGCATGAATTAAGAG
CAAAGGGAAGTCGCGTGCGGCCGTCCGGCGGAGGTATCTTGCAGACCGGGCACGGTACGGGAGGA
GCCAC

2. Інвертована послідовність П1 (**П2**)

CACCGAGGAGGGCATGGCACGGGCCAGACGTTCTATGGAGGCGGCCTGCCGGCGTGCGCTGAAGG
GAAACGAGAATTAAGTACGATACTTGAGCGGAGGGGCTTTAGGCGACCGCTAAAGTGGGCATCCG
ATAGATGCGATCCGCCTTTGGATCTCTAGGATCCAAGGAAACTGGGTCCCCTCCGAGGGCACTCT
CGTGA

3. П1 з внутрішньою інверсією (**П3**)

AGTGCTCTCACGGGAGCCTCCCCTGGGTCAAAGGAACCTAGGATCTCTAGGTTTCCGCCTTACGA
TACTTGAGCGGAGGGGCTTTAGGCGACCGCTAAAGTGGGCATCCGATAGATGCGAGAATTAAGAG
CAAAGGGAAGTCGCGTGCGGCCGTCCGGCGGAGGTATCTTGCAGACCGGGCACGGTACGGGAGGA
GCCAC

4. П1 з дуплікацією термінальної ділянки (**П4**)

AGTGCTCTCACGGGAGCCTCCCCTGGGTCAAAGGAACCTAGGATCTCTAGGTTTCCGCCTAGCGT
AGATAGCCTACGGGTGAAATCGCCAGCGGATTTCGGGGAGGCGAGTTCATAGCATGAATTAAGAG
CAAAGGGAAGTCGCGTGCGGCCGTCCGGCGGAGGTATCTTGCAGACCGGGCACGGTACGGGAGGA
GCCACGGAGGTATCTTGCAGACCGGGCACGGTACGGGAGGAGCCAC

5. П1 з дуплікацією внутрішньої ділянки (**П5**)

AGTGCTCTCACGGGAGCCTCCCCTGGGTCAAAGGAACCTAGGATCTCTAGGTTTCCGCCTAGCGT
AGATAGCCTACGGGTGAAATCGCCAGCGGATTTCGGGGAGGCGAGTTCATAGCATAGCGTAGATA
GCCTACGGGTGAAATCGCCAGCGGATTTCGGGGAGGCGAGTTCATAGCATGAATTAAGAGCAAAG
GGAAGTCGCGTGCGGCCGTCCGGCGGAGGTATCTTGCAGACCGGGCACGGTACGGGAGGAGCCAC

Рис. 3.18. Нуклеотидні послідовності, використані для дотплот-аналізу (П1 – промоторна ділянка гена *SCO0794*; інверсії і дуплікації позначено червоним кольором)

Важливими параметрами цієї програми є розмір *вікна вирівнювання* (window size; обов'язково непарне число у DNAdot) та *межа незбігів*, або *жорсткість* (mismatch limit, stringency). Перший параметр – це кількість нуклеотидів, яку порівнюють за один раз (іншими словами, за один крок аналізу порівнюють не по одній літері послідовностей, а більше – т. зв. "вікна", а це пришвидшує аналіз у випадку порівняння дуже довгих послідовностей). Другий – це кількість відмінних нуклеотидів у вікні, коли два сегменти, що їх вирівнюють, все ще класифікуватимуть як збіг. Наприклад, якщо розмір вікна 9 і межа незбігів 2, то при двох відмінних позиціях вікно з 9 нт треба вважати збігом. Налаштування параметрів DNAdot за замовчуванням такі: розмір вікна – 9, межа незбігів – 0.

Як уже зазначено вище, вирівнювання ідентичних послідовностей (П1 – П1) на дотплоті відображатиметься як неперервна діагональна лінія з лівого верхнього кута (точка відліку у координатному полі, в якому знаходиться початок послідовностей, що порівнюють) у правий нижній (рис. 3.19, *а*). Якщо довжина вікна становить 9, то дотплот-графік практично не міститиме фону, оскільки випадкові (недіагональні) збіги 9-нт ділянок (вікон) уздовж 200-нт послідовностей вкрай малоімовірні. Однак, якщо зменшити розмір вікна до одного (як мінімальну одиницю збігу на графіку відображати 1 нт), то одразу ж виникне значний фон (рис. 3.19, *б*). Це неминуче при вирівнюванні нуклеотидних послідовностей, де чотири літери-нуклеотиди багаторазово повторюються і розпізнаються програмою як збіги за заданих параметрів. Порівняння білків, які ґрунтуються на 20-ти літерах, завжди матиме менший фон. Важливо правильно налаштувати параметри вирівнювання для отримання інформативного результату.

Порівняння П1 з інвертованою послідовністю П2 матиме вигляд діагональної лінії з правого верхнього у лівий нижній кут (рис. 3.19, *в*). Якщо одна з двох послідовностей, які порівнюють, містить внутрішню інверсію, то на графіку це матиме вигляд розірваної лінії, де інвертована ділянка формуватиме хрестоподібну структуру (рис. 3.19, *г*). Дотплот-графіки, що є результатом порівняння послідовностей із дуплікаціями, зображені на рис. 3.19, *д-е*. Отже, за характером розміщення ліній на дотплот-графіку можна робити висновки про наявність у досліджуваних послідовностях інверсій, повторів, дуплікацій тощо. Цей метод особливо ілюстративний, коли необхідно візуалізувати зміни в нуклеотидній послідовності (як-от інверсії чи дуплікації) у геномних масштабах (десятки чи сотні т.п.н.).

Для порівняння у всіх розглянутих прикладах використано ідентичні послідовності, тому лінії загалом неперервні (крім ділянок з інверсіями/повторами). Вирівнювання двох подібних послідовностей матиме вигляд діагональної лінії з розривами (рис. 3.20). Лінії відображають ідентичні ділянки, перервані лінії – сегменти послідовностей, за якими вони відрізняються (відсутність однакових символів у відповідних позиціях).

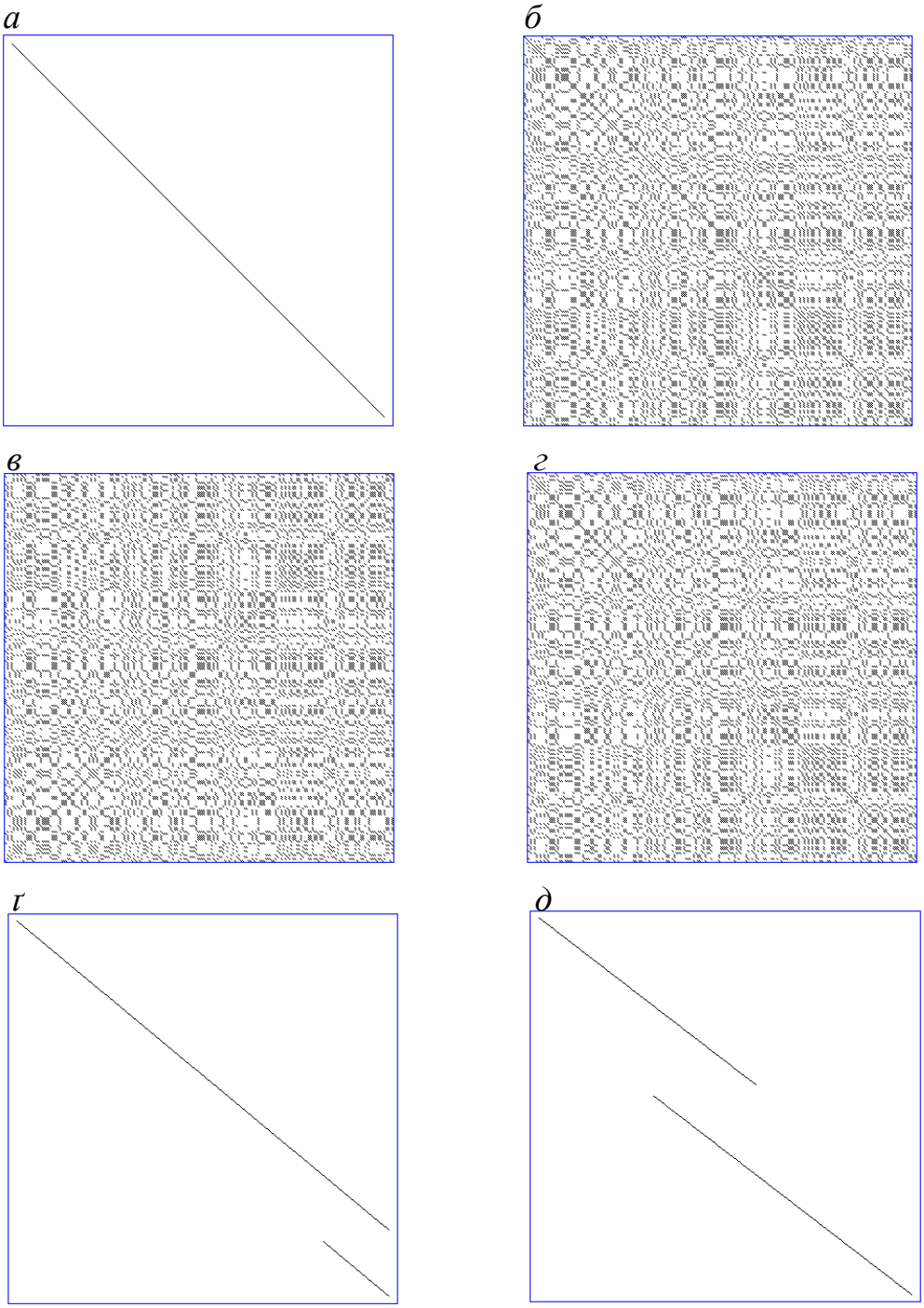


Рис. 3.19. Дотплот-графіки попарного вирівнювання послідовностей П1–П5 (див. рис. 3.18): вирівнювання ідентичних послідовностей П1–П1 (*а* і *б*), інвертованих П1–П2 (*в*) і П1–П3 (*г*), П1 і її похідної з термінальною дуплікацією (*ґ*) та внутрішньою дуплікацією (*д*)

Отже, для графічного відображення вирівнювання двох послідовностей використані одиничні матриці, де точки, що позначають ідентичні літери, за умови високої щільності, зливаються у лінії. До дотплот-графіків, однак, можна застосувати і складніше оцінювання на кшталт матриць рахунків заміщень.

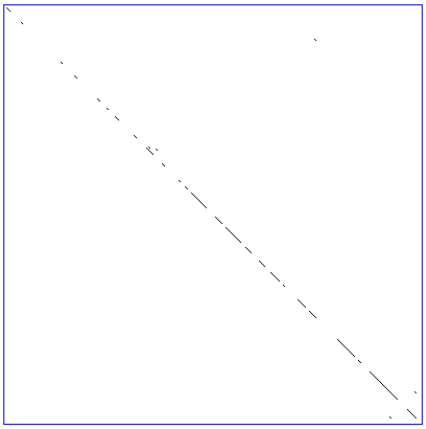


Рис. 3.20. Дотплот-графік вирівнювання послідовностей гена *lndI* (номер доступу в GenBank – AY659998) та його гомолога (HQ828984), виявленого у метагеномному зразку ДНК (ідентичність нуклеотидних послідовностей становить 87 %)

3.5.2. Алгоритми динамічного програмування для локального і глобального попарного вирівнювання послідовностей. Термін "динамічне програмування" означає, що алгоритм (програма) вирішує певне завдання шляхом розбиття його на простіші підзавдання, їхнє розв'язання і подальше поєднання окремих розв'язків в одну відповідь. *Алгоритм Нідельмана–Ванча* (АНВ) – це історично перший метод динамічного програмування, створеного для виявлення найвищої глобальної (від початку і до кінця) подібності між двома послідовностями. АНВ функціонує добре при вирівнюванні послідовностей, які подібні одна до одної вздовж усієї довжини. Однак такі випадки становлять меншість усіх можливих варіантів вирівнювання. Доменна структура білків диктує особливий характер їхньої еволюції, де сегменти первинної послідовності, що формують активні центри, залишаються (практично) незмінними внаслідок стабілізуючого добору, а функціонально менш важливі ділянки накопичують значну кількість замін. З огляду на сказане, часто є складно знайти задовільне глобальне вирівнювання двох споріднених багатодоменних білків. У цих випадках доцільніше шукати райони локальної подібності між однотипними доменами двох білків, не намагаючись їх повністю вирівняти. Такий підхід має біологічний зміст і, як засвідчує досвід, дає змогу порівнювати філогенетично віддалені генетичні послідовності. Локальне вирівнювання послідовностей вперше описано в *алгоритмі Сміта–Уотермана* (АСУ). АСУ шукає ділянку(-и) найбільшої подібності між ними, не намагаючись вирівняти їх від початку і до кінця. Ці два алгоритми подібні за принципом функціонування, тому далі розглядатимемо АСУ, який застосовують частіше як основний у сучасних методах попарного вирівнювання.

За своєю суттю АСУ нагадує дотплот-аналіз, який інтерпретують не графічно, а математично. Головне завдання АСУ – шукати ділянки послідовності, які виявляють локально максимальний рахунок вирівнювання. Як і в дотплот-аналізі, послідовності порівнюють за допомогою побудови двовимірної матриці й обчислення значень кожної комірки. Для обчислення цих значень можна використовувати різні матриці заміщень і різні системи штрафів за введення прогалин. Тут використаємо такі значення: збіг – 1, незбіг – 0, розриви – певне

від'ємне значення, залежне від позиції вирівнювання (опишемо докладніше далі). На першому етапі матрицю наповнюють значеннями 1 і 0 – відбувається ініціалізація матриці. Далі поступово додають значення комірок, щоб визначити рахунки матриці. Під час цього етапу переглядають кожну комірку матриці і до її значення додають максимальне значення, вибране з трьох сусідніх комірок – тих, що розміщені зліва, зверху і зверху по діагоналі. Такий процес додавання продовжується для всіх інших комірок. Після закінчення етапу додавання значень слідує етап *виявлення шляху максимальних значень* (traceback), що відображає оптимальний спосіб вирівнювання. Отже, вирівнювання генерується за рахунок прокладання шляху через матрицю, починаючи з елемента на С-кінці амінокислотної послідовності, і далі, рухаючись комірками максимальних значень, до N-кінця. Недіагональні переміщення (по горизонталі чи вертикалі) матрицею вестимуть до прогалин у текстовому записі попарного вирівнювання.

Розглянемо функціонування АСУ на прикладі вирівнювання двох послідовностей, ADLGAVFALCDRYFQ і ADLGRTQNCDRYYQ. Ключовою особливістю АСУ є те, що кожна комірка може бути кінцем потенційного вирівнювання, і значення (рахунок) цієї комірки є функцією рахунків попередніх комірок. Перший крок АСУ – це побудова матриці, де перший рядок і перша колонка матимуть значення дедалі від'ємніші через віддалення від початку вирівнювання (рис. 3.21). У такий спосіб у межах матриці накладається штраф за переміщення по горизонталі чи вертикалі, що відображає розриви у послідовності. Чим далі відбувається переміщення такими лініями – тим більший розмір штрафу.

	x	A	D	L	G	A	V	F	A	L	C	D	R	Y	F	Q
x	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15
A	-1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0
D	-2	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
L	-3	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
G	-4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
R	-5	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
T	-6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Q	-7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
N	-8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	-9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
D	-10	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
R	-11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Y	-12	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Y	-13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Q	-14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Рис. 3.21. Перший крок АСУ – ініціалізація матриці вирівнювання (сірим кольором виділений перший рядок і перша колонка – осі координатного поля; їх позначено як "x" і їм приписують щоразу від'ємніше значення, накладаючи штраф на переміщення по горизонталі чи вертикалі)

Далі визначають максимальний рахунок кожної комірки, починаючи з верхнього лівого кута матриці. Під час виконання цієї операції необхідно знати значення комірок зверху, зліва і зверху по діагоналі від комірки, що її розглядають. Наприклад, перша комірка на перетині амінокислотних залишків A і A (AA) у вихідному стані має значення 1, а три сусідні – -1, 0, -1. Отже, максимальний рахунок комірки AA: 1 + 0 = 1 (рис. 3.22).

	x	A	D	L	G	A	V	W	A	L	C	D	R	Y	F	Q
x	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15
A	-1	1	1	1	0	1	0	0	1	0	0	0	0	0	0	0
D	-2	1	2	0	0	0	0	0	0	0	0	1	0	0	0	0
L	-3	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0
G	-4	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
R	-5	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
T	-6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Q	-7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
N	-8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	-9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
D	-10	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
R	-11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Y	-12	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Y	-13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Q	-14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Рис. 3.22. Другий крок АСУ – визначення рахунку матриці вирівнювання (кольорами виокремлено декілька перших комірок, згаданих у тексті)

Тепер розглянемо комірку AD, справа від AA. Її вихідне значення – 0 (A і D – не збігаються), а три сусідні комірки (AA, *x*A, *x*D) мають такі значення (за годинниковою стрілкою): 1, -1, -2. Отже, до значення комірки AD – 0 – потрібно додати максимальне значення, знайдене в одній із сусідніх комірок – одиницю. Тому максимальний рахунок комірки AD – 1. Так само обчислюють значення комірок DA, DD і так, порядково, і згори донизу – до кінця матриці. Процес визначення рахунку кожної комірки продовжується, як уже описано, до кінця (правого нижнього кута) матриці. Повністю заповнену матрицю рахунків відображено на рис. 3.23.

Графічно найоптимальнішим (найкоротшим) шляхом у двовимірній матриці буде діагональна лінія. Якщо ж послідовності відрізняються за довжиною чи своїм змістом, то діагональний шлях неможливий. Відстеження оптимального шляху в розглянутій матриці розпочинається з кінця, у максимальному значенні (тут це комірка QQ, рахунок 10). З цієї комірки потрібно

	x	A	D	L	G	A	V	W	A	L	C	D	R	Y	F	Q
x	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15
A	-1	1	1	1	1	2	2	2	3	3	3	3	3	3	3	3
D	-2	1	2	2	2	2	2	2	2	2	2	4	4	4	4	4
L	-3	1	2	3	3	3	3	3	3	4	4	4	4	4	4	4
G	-4	1	2	3	4	4	4	4	4	4	4	4	4	4	4	4
R	-5	1	2	3	4	4	4	4	4	4	4	4	5	5	5	5
T	-6	1	2	3	4	4	4	4	4	4	4	4	5	5	5	5
Q	-7	1	2	3	4	4	4	4	4	4	4	4	5	5	5	6
N	-8	1	2	3	4	4	4	4	4	4	4	4	5	5	5	6
C	-9	1	2	3	4	4	4	4	4	4	5	5	5	5	5	6
D	-10	1	3	3	4	4	4	4	4	4	5	6	6	6	6	6
R	-11	1	3	3	4	4	4	4	4	4	5	6	7	7	7	7
Y	-12	1	3	3	4	4	4	4	4	4	5	6	7	8	8	8
Y	-13	1	3	3	4	4	4	4	4	4	5	6	7	9	9	9
Q	-14	1	3	3	4	4	4	4	4	4	5	6	7	8	9	10

Рис. 3.23. Заповнена матриця рахунків

рухатися догори у правий верхній кут (до початку вирівнювання), щоразу переходячи до сусідньої комірки з максимально можливим значенням. Якщо всі сусідні комірки мають однакове значення, то рух продовжують по діагоналі. Керуючись цими правилами, можна знайти найкоротший шлях з одного кінця в інший, як відображено на рис. 3.24.

а

	x	A	D	L	G	A	V	F	A	L	C	D	R	Y	F	Q
x	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15
A	-1	1	1	1	1	2	2	2	3	3	3	3	3	3	3	3
D	-2	1	2	2	2	2	2	2	3	3	3	4	4	4	4	4
L	-3	1	2	3	3	3	3	3	3	4	4	4	4	4	4	4
G	-4	1	2	3	4	4	4	4	4	4	4	4	4	4	4	4
R	-5	1	2	3	4	4	4	4	4	4	4	4	5	5	5	5
T	-6	1	2	3	4	4	4	4	4	4	4	4	5	5	5	5
Q	-7	1	2	3	4	4	4	4	4	4	4	4	5	5	5	6
N	-8	1	2	3	4	4	4	4	4	4	4	4	5	5	5	6
C	-9	1	2	3	4	4	4	4	4	4	5	5	5	5	5	6
D	-10	1	3	3	4	4	4	4	4	4	5	6	6	6	6	6
R	-11	1	3	3	4	4	4	4	4	4	5	6	7	7	7	7
Y	-12	1	3	3	4	4	4	4	4	4	5	6	7	8	8	8
Y	-13	1	3	3	4	4	4	4	4	4	5	6	7	9	9	9
Q	-14	1	3	3	4	4	4	4	4	4	5	6	7	8	9	10

б

A D L G A V F A L C D R Y F Q

A D L G - R T Q N C D R Y Y Q

Рис. 3.24. Шлях вирівнювання, що має максимальний рахунок (*а*), виділено кольором; текстова репрезентація відповідного попарного вирівнювання (*б*)

У цьому прикладі дві послідовності достатньо подібні одна до одної, і їх можна вирівняти з кінця в кінець. Насправді виконано глобальне вирівнювання, однак головне завдання АСУ – це пошук локальної подібності у неспоріднених білках, як відображено на рис. 3.25.

а

	x	K	T	S	G	A	V	F	A	L	W	M	T	N	F	Q
x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
D	0	0	0	0	0	1	1	1	2	2	2	2	2	2	2	2
L	0	0	0	0	0	1	1	1	1	3	3	3	3	3	3	3
G	0	0	0	0	1	1	1	1	1	3	3	3	3	3	3	3
A	0	0	0	0	1	2	2	2	3	3	3	3	3	3	3	3
V	0	0	0	0	1	2	3	3	3	3	3	3	3	3	3	3
F	0	0	0	0	1	2	3	4	4	4	4	4	4	4	4	4
A	0	0	0	0	1	3	3	4	5	5	5	5	5	5	5	5
L	0	0	0	0	1	3	3	4	5	6	6	6	6	6	6	6
W	0	0	0	0	1	3	3	4	5	6	7	7	7	7	7	7
R	0	0	0	0	1	3	3	4	5	6	7	7	7	7	7	7
Y	0	0	0	0	1	3	3	4	5	6	7	7	7	7	7	7
Y	0	0	0	0	1	3	3	4	5	6	7	7	7	7	7	7
R	0	0	0	0	1	3	3	4	5	6	7	7	7	7	7	7

б

K T S G A V F A L W M T N F Q

A D L G A V F A L W R Y Y R -

Рис. 3.25. Ділянка локальної подібності між двома послідовностями (*а*), виділена кольором; текстова репрезентація відповідного попарного вирівнювання (*б*)

Отже, максимальні значення перших комірок матриці (див. рис. 3.25) мають нульові значення, отож їх не можна вважати вирівняними. Так само значення останніх комірок (правий нижній кут) не зростають, що також засвідчує недостатню кількість збігів. Тому АСУ відкидає ці райони матриці і повертає вирівнювання лише центральної частини послідовностей. Ще одна відмінність матриці, що відображена на рис. 3.25, від попередніх – це інші значення першої колонки і рядка. Тут вони мають одне значення – 0. Нулем в АСУ позначають кінець вирівнювання послідовностей довжиною 0. Відомі й інші підходи до позначення крайніх позицій матриці, які, однак, не змінюють самого принципу функціонування АСУ.

Алгоритм глобального вирівнювання Нідельмана-Ванча (АНВ) за принципом функціонування нагадує АСУ. АНВ відрізняється від АСУ тим, що він дає змогу враховувати всі позиції заданих послідовностей. На відміну від вирівнювання, зображеного на рис. 3.25, вихідною точкою відстеження шляху максимальних значень в АНВ завжди є N-кінець, і його обчислюють у процесі додавання значень, починаючи з С-кінця. З цих причин АНВ називають *методом глобального вирівнювання*.

Задля кращого розуміння принципів динамічного програмування у всіх розглянутих прикладах використано прості (одиничкові, унітарні) матриці заміщення. Сьогодні, однак, практично всі вирівнювання амінокислотних послідовностей на основі АСУ або АНВ використовують PAM- чи BLOSUM-

матриці. Як уже зазначено вище, це дає змогу точніше вирівняти послідовності і виявити гомологію у групах віддалених білків. Ілюстративно порівняємо рахунок попарного вирівнювання трьох простих амінокислотних послідовностей на основі унітарної матриці і BLOSUM62. Результат висвітлено на рис. 3.26. Нехай задано три послідовності, LLACIVDEWL (1), LLACIVDEPL (2) і IIGCMIDEWL (3). Перед дослідником постало питання – котра з двох останніх послідовностей (2 і 3) подібніша до 1. У межах одиничної матриці рахунків приймемо, що збіг має рахунок 1, незбіг – 0. Як емпіричну матрицю заміщень використано BLOSUM62 (див. рис. 3.11). Згідно з одиничною моделлю (див. рис. 3.26), білок 2 подібніший до білка 1 (рахунок вирівнювання білків 1 і 2 – 9, 1 і 3 – 5). Водночас обчислення рахунку цих вирівнювань, згідно з BLOSUM62, засвідчує, що білок 3 подібніший до білка 1, ніж білок 2.

```
Білок 1: LLACIVDEWL
Білок 2: LLACIVDEPL
Білок 3: IIGCMIDEWL

            Одиничкова модель (1/0)

Білок 1: LLACIVDEWL рах.:1+1+1+1+1+1+1+1+0+1=9
Білок 2: LLACIVDEPL

Білок 1: LLACIVDEWL рах.:0+0+0+1+0+0+1+1+1+1=5
Білок 3: IIGCMIDEWL

ВИСНОВОК: 9>5; 1 подібніший до 2

                    BLOSUM62

Білок 1: LLACIVDEWL рах.:4+4+4+9+4+4+6+5-4+4=40
Білок 2: LLACIVDEPL

Білок 1: LLACIVDEWL рах.:2+2+0+9+1+3+6+5+11+4=43
Білок 3: IIGCMIDEWL

ВИСНОВОК: 43>40; 1 подібніший до 3
```

Рис. 3.26. Обчислення рахунків вирівнювання на основі унітарної та емпіричної моделей вирівнювань (жовтим кольором виокремлено позиції вирівнювання з відмінними амінокислотними залишками)

Високий рахунок вирівнювання білків 1 і 3 передусім пов'язаний зі збереженням триптофану (W) у дев'ятій позиції обох послідовностей. Триптофан – рідкісна амінокислота, яка має унікальну структуру і кодується лише одним

кодоном. Тому збереження залишку триптофану у послідовностях двох білків має велике еволюційне значення. Це виражається у високій частці спорідненості, яку приписують цій амінокислоті у всіх матрицях заміщення – вона найвища серед усіх амінокислот (див. рис. 3.7; 3.11). Збереження триптофану у білках 1 і 3 переважає наявність п'яти незбігів між ними, які зачіпають відносно прості й, очевидно, функціонально замінні амінокислотні залишки.

3.5.3. Штрафи за розриви. Одночасно з використанням реалістичніших, неодиничних матриць заміщення, пошук оптимальної стратегії оцінювання розривів у попарних вирівнюваннях "справжніх" генетичних послідовностей – важливий аспект біоінформатичних алгоритмів. Про види штрафів за введення розривів уже зазначено. Однак, з огляду на важливість цього питання, підходи до накладання штрафів тут розглянемо розгорнуто.

Попарне вирівнювання може базуватися на трьох різних підходах щодо розривів, тобто можна: *а*) заборонити їхнє уведення; *б*) дозволити уводити довільно без штрафів; *в)* розробити певну систему стягнення штрафів за різні типи розривів. Як проілюстровано на рис. 3.1, уведення розривів дає змогу отримати краще і біологічно змістовне вирівнювання послідовностей, тому перший варіант (заборонити розриви у вирівнюваннях) практично не використовують у сучасних алгоритмах. Довільне введення розривів може підвищити рахунок вирівнювання, як відображено на рис. 3.27, *а*, але саме вирівнювання втрачатиме біологічний зміст унаслідок їхнього надмірного накопичення. Отож виникає потреба у розробленні певного набору правил, за якими визначають величини штрафів для розривів різних типів. Запропоновано багато різних типів штрафувань, однак найуживанішими залишаються *афінні штрафи*. Вони складаються з двох параметрів – штрафу G за відкриття розриву і додаткового штрафу L за його продовження. Для розриву довжиною n нуклеотидів чи амінокислот сумарний афінний штраф γ становитиме $G+Ln$ або $G+L(n-1)$. Вибір величин G та L досі залишається емпіричним; зазвичай перша достатньо велика, а друга – мала. Біологічний зміст полягає в тому, що вставки чи делеції – рідкісні події, але коли вони трапляються, то часто зачіпають декілька сусідніх залишків. Надання різної ваги параметрам G та L дає змогу оцінити їхній вплив на вирівнювання, як це зображено на рис. 3.27, *б-в*, і відібрати оптимальну стратегію штрафувань для певного типу послідовностей.

3.5.4. Статистичне оцінювання попарних вирівнювань. Застосовуючи певну систему обчислення "вартості" кожної позиції попарного вирівнювання (збігів, відмінностей і розривів), алгоритм визначить рахунок вирівнювання послідовності S як суму рахунків s усіх вирівняних пар. Для двох достатньо складних послідовностей може бути достатньо багато варіантів вирівнювання, і програма обирає одне – з *оптимальним* значенням S. Наголосимо, що оптимальне значення – не обов'язково найвище, як це бачимо з аналізу рис. 3.27, *а*, де найвищий рахунок має біологічно малозмістовне вирівнювання з довільним уведенням розривів. Визначаючи оптимальне значення, потрібно враховувати систему штрафів, застосовану для вирівнювань. Наскільки великим має бути значення S, аби слугувати доказом гомології двох вирівняних послідовностей?

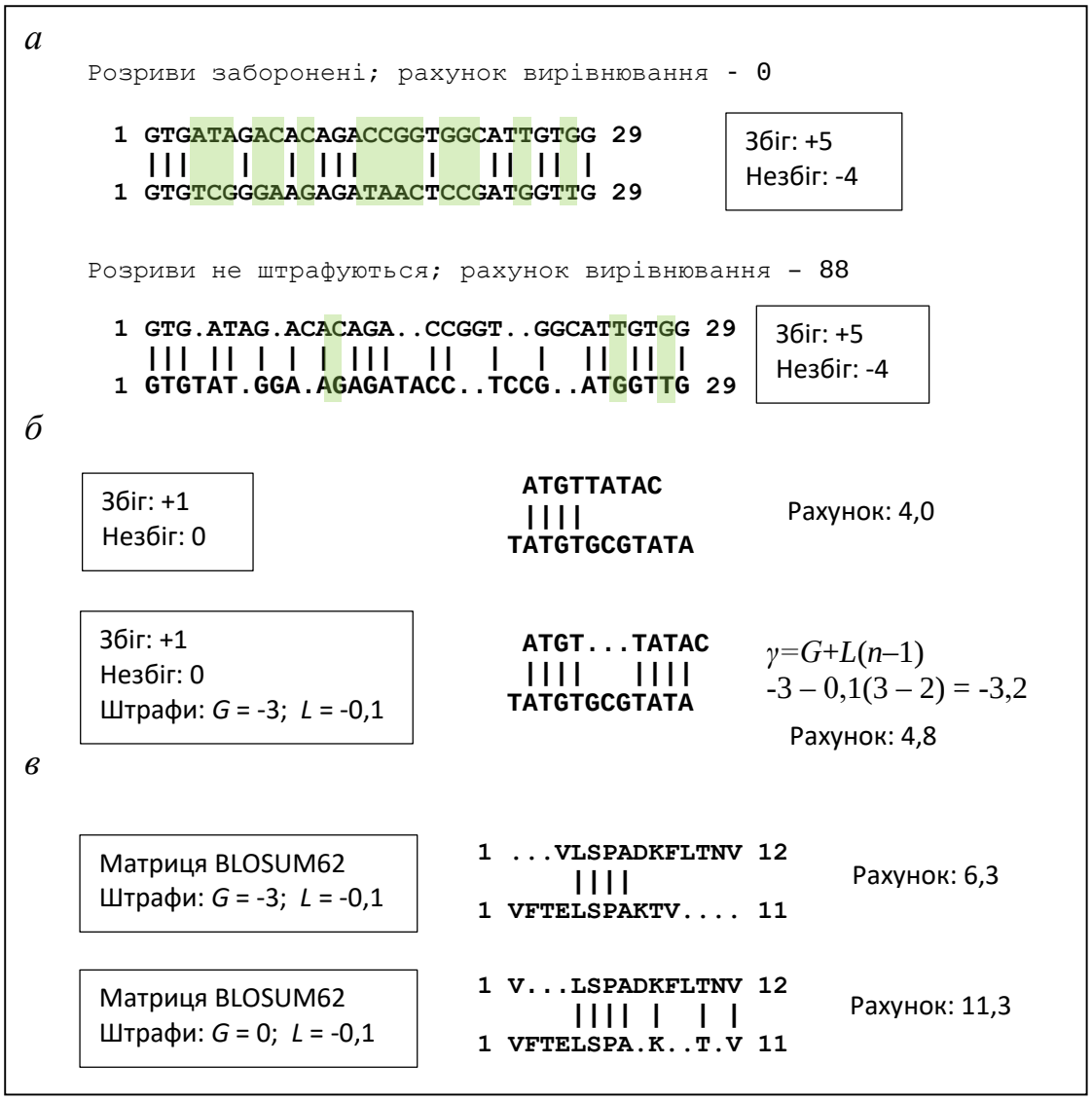


Рис. 3.27. Штрафи у попарних вирівнюваннях генетичних послідовностей. Довільне введення розривів збільшує рахунок вирівнювання (*а*). Зеленим кольором виділені позиції з відмінними залишками – помітне їхнє суттєве зменшення за умови довільного введення розривів. Система афінних штрафів підвищує рахунок вирівнювання (*б*). Використання різних систем штрафів приводить до різних вирівнювань, що матимуть різні рахунки; брак штрафу за відкриття розриву (*G*) приводить до появи багатьох незалежних розривів (*в*)

Відповіддю на це питання може стати оцінювання імовірності протилежного пояснення – того, що отримана величина *S* випадкова (тобто її можна отримати внаслідок вирівнювання заданої послідовності із випадковими). Сьогодні немає математичної теорії для визначення такої ймовірності для глобальних вирівнювань. Тут можна застосувати підхід, який ґрунтується на порівнянні *S* певних двох послідовностей із рахунками вирівнювань випадкових послідовностей, що мають однакову довжину із експериментальною. Випадкові послідовності для таких статистичних оцінювань можна отримати декількома способами. По-перше, задану послідовність можна вирівнювати із набором справжніх, але не гомологічних послідовностей. По-друге, набір випадкових

послідовностей можна отримати у процесі цілком випадкового генерування послідовностей, або ж у процесі переставляння позицій однієї з двох вирівняних послідовностей А і Б. Спосіб переставляння проілюстровано на рис. 3.28. Наприклад, у послідовності Б (LITTLEREDFOX) переставляють окремі блоки чи окремі літери, внаслідок чого отримують LITTLEFOXRED, LLTTEEIDFXR, TELLITREDFOX тощо. Далі рекомбіновані послідовності вирівнюють із послідовністю А, до якої LITTLEREDFOX виявляла гомологію, й отримують великий набір значень *S*, з яких можна побудувати розподіл. У цьому розподілі за віссю ординат відкладена кількість вирівнювань, що мають певне значення *S*. Величина *S* вирівнювання вихідних двох послідовностей різко відрізняється від середнього значення *S* вирівнювань на основі рекомбінованих послідовностей. Далі, застосовуючи стандартні методи статистичного аналізу, можна обчислити кількість стандартних відхилень *Z*, яка відділяє середнє значення *S* від $S_{АБ}$ (рис. 3.28). Наприклад, у програмі fasta (розглянемо у наступному пункті 3.5.5) прийнято, що статистично значущим рахунком є той, який лежить на відстані, більшій за шість стандартних відхилень від середнього рахунку випадкових послідовностей ($Z > 6$). Це твердження математично не обґрунтоване. Оскільки "хвіст" розглянутого розподілу (у напрямі зростання *S*) нескінченно довгий, то такий розподіл не можна вважати нормальним, отож неможливо перетворити *Z* у загальновідому величину ймовірності *P*. Отже, якщо 100 випадкових вирівнювань мають *S* менше ніж $S_{АБ}$, то найбільше, що може сказати дослідник – це те, що значення *P* для $S_{АБ}$, імовірно, менше за 0,01 (1/100).

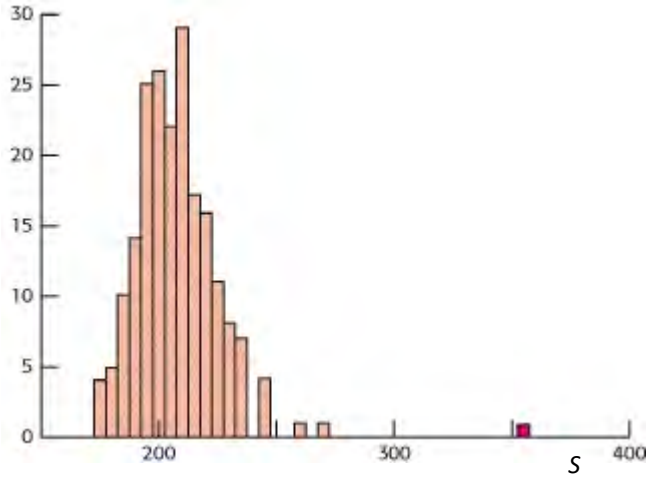


Рис. 3.28. Розподіл значень *S* для вирівнювання послідовності А з рекомбінованими послідовностями (стовпчики кольору цвіту сакури) і з послідовністю LITTLEREDFOX (стовпчик вишневого кольору; $S_{АБ}$). Вісь ординат – кількість вирівнювань з певним значенням *S*

Для локальних вирівнювань без прогалин у 1990-х роках запропонована перша статистична модель розподілу рахунків. Вона відома як *теорія Карліна-Альтшуля* (ТКА), названа в честь Стівена Альтшуля й Семюела Карліна – двох науковців, що найбільше працювали над її розробленням. Основоположним для ТКА є спостереження, що рахунки локальних вирівнювань послідовностей описують згідно з *розподілом екстремального значення* (extreme value distribution, EVD). Цей розподіл відображено на рис. 3.29.

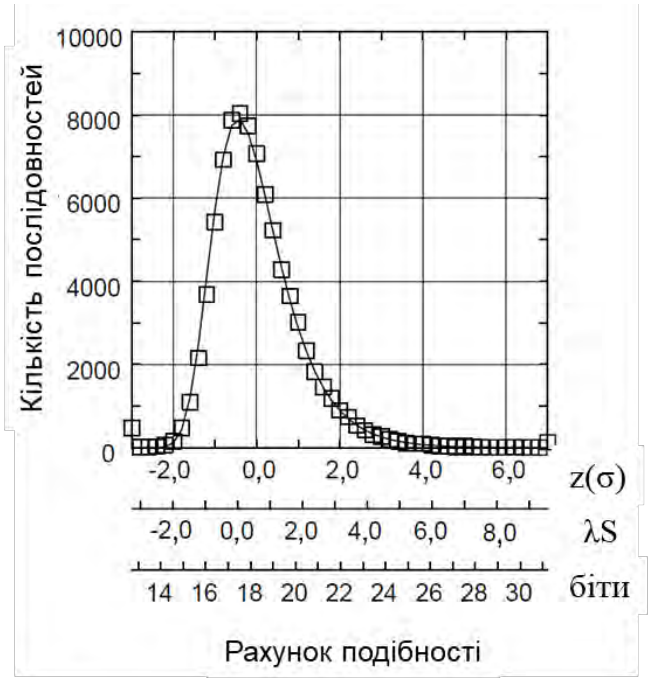


Рис. 3.29. Розподіл екстремальних значень (EVD). Фактичний розподіл (квадрати) рахунків подібності у результаті порівняння білка-транспортера глюкози в людини gtr1_human з 84 000 білками у базі Swiss-Prot, й очікувана крива (неперервна лінія) розподілу рахунків на основі EVD. Рахунки обчислені за допомогою АСУ і матриці BLOSUM62 (*G*:–12; *l*: –1). Вісь ординат – кількість послідовностей у Swiss-Prot, що отримали рахунок вирівнювання з gtr1_human. Вісь абсцис – три шкали оцінення міри подібності (рахунок подібності): *z*(σ) – це значення у кількостях стандартних відхилень від середнього; λ*S* – рахунок *S*, нормалізований поправкою на тип матриці і частоти залишків; біт-рахунок. Біологічне значення константи λ і біт-рахунку обговоримо далі. EVD нагадує нормальний розподіл, в якому праве плече спадає невизначено довго, бо максимальне значення прямує до екстремуму

ТКА ґрунтується на п’яти умовах:

1. Для заданої пари подібних послідовностей має бути вирівнювання із додатним значенням (тобто, для заданої пари послідовностей необхідна пара сегментів з високим рахунком – high scoring segment pairs, HSPs).
2. Очікуваний рахунок вирівнювання (випадкових послідовностей) має бути від’ємним.
3. Літери (залишки) генетичних послідовностей, що вирівнюють – незалежні та рівномірно розподілені (***i***ndependent ***i***dentically ***d***istributed, *iid*).
4. Послідовності нескінченно довгі.
5. Вирівнювання не містять розривів.

Перші дві умови справедливі для будь-якої матриці заміщень, яка ґрунтується на реальних даних. Три останні – проблематичні, оскільки: а) у генетичних послідовностях (білках і ДНК/РНК) часто наявна позиційна залежність залишків (наприклад, у амінокислотній послідовності можуть бути три неприлеглі залишки гістидину, які при формуванні глобули білка стають каталітичним центром, і частоти мутацій цих залишків суттєво відрізнятимуться від інших залишків; інший приклад – це повтори у білках); послідовності

скінченні і вони часто вирівнюються за рахунок уведення розривів. Згодом опишемо способи, як можна уникнути цих складнощів. Наразі розглянемо головне рівняння ТКА:

$$E = Kmne^{-\lambda S}. \quad (28)$$

Рівняння стверджує, що *очікувана кількість* E випадкових вирівнювань з рахунком рівним або більшим S є функцією розміру простору пошуку (добуток довжин заданої послідовності й бази даних, до якої вирівнювали першу; $m \times n$), нормалізованого рахунку λS і поправки K. Математичні засади рівняння (28) деталізовано у блоці 7, а значення параметрів (констант) λ і K описано далі.

Поправка K дає змогу врахувати той факт, що рахунки оптимальних локальних вирівнювань, які починаються в різних позиціях, можуть значно корелювати. Наприклад, якщо є HSP з початком на залишках 1, 1, то також буде HSP, що розпочинається залишками 2, 2. Значення K – в районі 0,1, і його вплив на статистичне оцінювання вирівнювань незначний. Між значенням E і простором пошуку (mn) існує лінійна залежність. Зокрема, якщо розмір простору пошуку подвоюється, то подвоюється й очікуване число випадкових вирівнювань з рахунком S. Між E й S є експоненційна обернена залежність – малі зміни в рахунку S спричиняють значні відмінності в E. Наприклад, невелике зростання рахунку (скажімо від 38 до 40) вестиме до значного зменшення (на порядки) числа E.

Параметр λ – константа унормування, на яку треба перемножити рахунок S, щоб обчислити його ймовірність. Цей параметр у рівнянні (28) обернено пропорційний до фактора внормування, який входить до складу рівняння (27), про що йшлося в тексті вище. Але точне значення λ може відрізнятися внаслідок помилок заокруглення до цілого числа. Для вирівнювань без розривів λ є унікальним розв’язком такого рівняння:

$$\sum_{i,j} p_i p_j e^{\lambda s_{i,j}} = 1. \quad (29)$$

Помітно, що λ залежить від типу матриці заміщень і від амінокислотного складу послідовностей, які вирівнюють. У певному значенні λ можна тлумачити як коефіцієнт, що перетворює рахунки попарного збігу в імовірності; $e^{-\lambda S}$ подібний до p^l у прикладі з підкиданням монет (блок 7).

Отже, один раз визначену величину λ можна використовувати в обчисленнях E усіх HSP. Втім частоти амінокислотних залишків у різних білках дуже варіюють від тих частот, на яких базуються матриці рахунків. Сьогодні програми попарного вирівнювання використовують задану послідовність та подібні до неї (з бази даних), аби обрахувати параметр λ відповідно до актуальних частот залишків. Це *хіт-специфічний* параметр λ, який підвищує чутливість методів попарного вирівнювання.

БЛОК 7. Дуже стисло про статистичні засади теорії Карліна-Альтшуля

Підкидаючи монету n разів, імовірність E випадання герба l разів поспіль можна описати такою формулою (за Erdös та Renyi, як цитує Waterman, 1995):

$$E \cong np^l, \qquad (7.1)$$

де p – імовірність герба

Це рівняння є результатом міркування, що очікуване число гербів – добуток числа підкидань на ймовірність герба під час кожного підкидання. Якщо найдовша серія гербів R_l очікується один раз, то $1 = np^{Rl}$, тоді $R_l = \log_{1/p}(n)$. Найдовша серія гербів еквівалентна пошуку району з найвищим рахунком (напр., гідрофобний сегмент) в одній амінокислотній послідовності, при використанні рахункової матриці, що приписує позитивне значення деяким залишкам, і $-\infty$ – усім іншим. Тоді ймовірність додатного рахунку (відповідає випаданню серії гербів) буде $\sum p_i$ для кожного залишку p_i, що отримає додатний рахунок S_i.

Приклад з монетою (пошуком ділянки білка зі штрафом $-\infty$ за незбіг) засвідчує, що рахунки локальної подібності для окремих НАП мають зростати як логарифм довжини n послідовностей. У порівнянні послідовностей розглядають можливі вирівнювання двох послідовностей, $a_{1\ldots m}$ й $b_{1\ldots n}$, однак обчислення ймовірності доволі подібне. Тут замість обчислення ймовірності випадання гербів, де $p_k = pp_{k\text{-}1}$, розглянемо випадок збігу в m позиціях, що еквівалентно приписуванню рахунку (ймовірності випадання) герба, якщо $a_i = b_i$. Якщо розмістити дві послідовності по осях двовимірної матриці (як на рис. 3.21–3.24, основний текст), то проблема пошуку найдовшої серії гербів зводиться до виявлення найдовшої серії збігів уздовж будь-якої з діагоналей. Якщо літери (залишки) у двох послідовностях мають однакові частоти зустрічності (ймовірності, p), тоді ймовірність збігу залишку a_i з b_j буде p, а ймовірність збігу довжиною l з a_i, b_j до $a_{i+l\text{-}1}$, $b_{j+l\text{-}1}$ знову-таки буде p^l. У цьому випадку, однак, є $(m\text{-}l{+}1){\times}(n\text{-}l{+}1)$ місць (позицій) з яких може початися збіг, тому $E(l) \cong mnp^l$. Звідси, очікувана довжина найдовшої ділянки ідентичності між двома випадковими послідовностями завдовжки m та n, коли збіг матиме додатний рахунок, а незбіг $-\infty$ – буде:

$$M_{mn} = \log_{\frac{1}{p}}(mn) \text{ або } 2\log_{\frac{1}{p}}(n), \text{якщо } m = n. \qquad (7.2)$$

Перехід від $\log_{1/p}(n)$ до $\log_{1/p}(n^2)$ для двох послідовностей завдовжки n відображає більшу кількість позицій, де може розпочатися ділянка збігу M_l з імовірністю $P(M_l) = p^{Ml}$. Як і у випадку однієї послідовності, можна перейти від пошуку ймовірності найдовшої ідентичної ділянки до пошуку ймовірності рахунку $S_l \geq x$. Тобто розглядають $P(S>x)$, коли пара залишків a_ib_j збігається і має додатний рахунок $S_{i,j}$, а всі незбіги мають від'ємний рахунок $-\infty$. Для локального попарного вирівнювання з рахунком незбігу $-\infty$ і без розривів очікуване число ділянок з рахунком $S \geq x$ матиме загальну форму:

$$E(S > x) \alpha mnp^x, \text{ що еквівалентне } E(S > x) \alpha mne^{lnp}, \text{ або: } mne^{-\lambda x}, \qquad (7.3)$$

де $\lambda = -\ln p$, константа, що враховує частоти залишків і тип матриці рахунків.

Карлін та Альтшуль поширили цей підхід на загальний випадок локальних збігів послідовностей, або локальних рахунків подібності, для вирівнювань, що не перекриваються і не мають розривів. Щоб найвищий рахунок був локальним, потрібно дотриматися умови, що рахунок випадкового вирівнювання має бути від'ємним: $E(s_{i,j}) = \sum_{i,j} p_i p_j s_{i,j} < 0$. Тоді очікуване число випадкових (негомологічних) вирівнювань з рахунком S буде:

$$E(S \geq x) = Kmne^{-\lambda x}, \text{або } \; E = Kmne^{-\lambda S}. \qquad (7.4)$$

Отримане рівняння (7.4) тотожне рівнянню (28) з основного тексту.

Зважаючи на зазначене вище, для локальних вирівнювань ТКА описує очікуваний розподіл величин S щодо випадкових попарних вирівнювань. Дві вирівняні послідовності у такому випадку міститимуть одну чи декілька пар сегментів однакової довжини, по одному сегменту з кожної послідовності. АСУ знаходить у вирівнюваннях такі пари, значення S яких не зростає при продовженні чи вкороченні з кінців. Їх назвали *парою сегментів з високим рахунком* (high scoring segment pairs; HSPs). Далі застосовують прийом, описаний попередньо, тобто оцінюють імовірність того, що обчислене значення S можна отримати унаслідок вирівнювання випадкових послідовностей. Тут випадкові послідовності генеруються *ab initio* відповідно до певної моделі частот вживання амінокислот чи нуклеотидів. Це аналітичний підхід, який не залежить від певного набору вже відомих генетичних послідовностей. Важливо, що очікуваному рахунку вирівнювання випадкових послідовностей необхідне від'ємне значення. Інакше довгі послідовності матимуть велике значення S незалежно від ступеня подібності, і ТКА не буде справджуватися.

Як уже зазначено, порівняння різних значень S має мало змісту без розуміння способу його обчислення, типу матриці заміщення і системи штрафів за розриви – це те саме, що описувати відстань без визначення одиниці її вимірювання: в метрах, милях чи світлових роках. Наприклад, три різні вирівнювання двох послідовностей (рис. 3.30) мають різне значення S. Ці значення неможливо порівняти, чи обрати з них "найкраще", оскільки їх отримано завдяки застосуванню різної системи штрафів. Тому прийнято унормовувати S за допомогою формули:

$$S' = (\lambda S - \ln K) / \ln 2. \qquad (30)$$

Так отримують ***біт-рахунок*** (bit score) S', виражений у стандартному наборі одиниць. S' дає змогу оцінити кількість випадкових вирівнювань, яку треба проаналізувати перед тим, як очікують знайти принаймні одне з рахунком не менше S'. Наприклад, якщо S' = 30, то слід проаналізувати не менше 2^{30}, тобто приблизно 10^9 вирівняних пар випадкових послідовностей, щоб знайти одну пару з рахунком $\geq S'$. Кожна біт-одиниця подвоює кількість випадкових вирівнювань, яку слід перевірити аби знайти одне з рахунком $\geq S'$. Біт-рахунок – унормована версія S, яка не залежить від розміру НАП і бази, які аналізували (простір пошуку - mn), і системи визначення рахунків (розтлумачимо далі).

Значення E пов'язане з біт-рахунком S' таким рівнянням:

$$E = mn2^{-S'}. \qquad (31)$$

Біт-рахунки дають змогу підсумувати природу системи обчислення рахунків, отож тепер, для розрахунку статистичної значущості E, потрібно володіти лише інформацією про розмір простору пошуку.

γ[1]	Вирівнювання	ID/SI, %[2]	*G*[3]	*S*[4]
0	1 GTC-ATGCTA-GTCGT---GG---GTAGCATTTA-GCT-ATG-TGGG-GT 38 1 –TCGATGCT-GGTCG-CAAGGCAAGTAG---TTATG-TCATGCT---AG- 39	27/50 (54,0)	23/50	135
-5	1 GTC-ATGCTAGTCG--TGGGTAGCATTTA-GCT-ATG-TGGGGT 38 1 –TCGATGCTGGTCGCAAGGCAAGTAGTTATG-TCATGCTAG--- 39	26/44 (59,1)	11/44	67
-10	1 ----------GTCATGCTAGTCGTGGGTAGCATT 38 1 TCGATGCTGGTCGCAAGGCAAGTAGTTATGTCATGCTAG---------- 39 TAGCTATGTGGGGT	10/67 (14,9)	57/67	50

Рис. 3.30. Три різні способи вирівнювання пари послідовностей, що дають змогу отримати різні значення подібності і *S*: 1 – розмір штрафу; 2 – відсоток позицій вирівнювання, в яких є ідентичні (ID) або подібні (SI) пари нуклеотидних залишків; 3 – кількість розривів; 4 – рахунок вирівнювання

Рівняння (28) і (31) дають змогу обчислити кількість випадкових послідовностей у базі даних розміру *m*, що при вирівнюванні з заданою послідовністю *n* матимуть рахунок ≥*S*. Як оцінити статистичну достовірність вирівнювання білка довжиною *n* до бази даних, що містить багато білків різної довжини? Іншими словами, за яким принципом визначати кількість випадкових вирівнювань (*E*) зі значенням *S* не під час аналізу пари послідовностей, а під час скринінгу великої бази даних? Є декілька варіантів дій. Наприклад, нульовою гіпотезою можна вважати те, що всі білки в базі даних *a priori* з однаковою ймовірністю споріднені з заданим білком. Тоді низькі величини очікування *E* для вирівнювання заданої послідовності як з короткими, так і з довгими послідовностями матимуть однакову статистичну вагу. У цьому випадку для обчислення "*E* бази даних" необхідно перемножити значення *E* попарного вирівнювання (*mn*) на кількість послідовностей у базі даних, звідки походить *n*. Альтернативною нульовою гіпотезою може бути те, що ймовірність спорідненості *a priori* пропорційна довжині послідовності. Тоді значення *E* попарного вирівнювання потрібно перемножити на *j*, де *j* – це сума довжин усіх послідовностей з застосованої бази даних. В обчисленнях "*E* бази даних" значення *n* у рівняннях (28) і (31) дорівнюватиме *j*. Тобто базу даних з багатьох послідовностей (*n*) розглядають як одну довгу послідовність завдовжки *j* амінокислотних залишків. Цей підхід застосовують у програмах родини BLAST.

Ще одним статистичним показником HSP у попарних вирівнюваннях є величина ймовірності *P*. Якщо *E* вказує на очікувану кількість випадкових вирівнювань із рахунком рівним/більше *S* за заданих умов, то *P* (асоційована з певним значенням *S*) вказує на ймовірну частоту натрапляння на випадкове

вирівнювання з таким (або більшим) рахунком. Між E та P є така залежність, що має характер розподілу Пуасона:

$$P = 1 - e^{-} \; ; \tag{32}$$

$$E = -\ln(1 - P). \tag{33}$$

Більшість сучасних програм вирівнювання (такі як BLAST, їх опишемо далі) не подає значення P, оскільки простіше зрозуміти різницю між значеннями E 5 і 10, ніж між значеннями P 0,993 і 0,99995. Коли значення $E < 0{,}001$, то E і P практично однакові.

У контексті попарного вирівнювання двох послідовностей (не порівняння проти великого масиву послідовностей), формула обчислення P набуває такого вигляду:

$$P_S = Ke^{-\lambda S}; \tag{34}$$

$$P_{S'} = 2^{-S'}. \tag{35}$$

Статистична теорія, описана попередньо, має вичерпне аналітичне доведення тільки для попарних вирівнювань без розривів. Математичне моделювання засвідчує, що розподіл рахунків попарних вирівнювань з розривами також описується розподілом EVD (див. рис. 3.29). Цей емпіричний доказ слугує підставою для застосування ТКА до всіх типів попарних вирівнювань. Для вирівнювань без розривів статистичні параметри (K, λ) обчислюють за аналітичними формулами із матриць заміщень і фонових частот залишків у послідовностях, що їх вирівнюють. Для вирівнювань з розривами ці параметри оцінюють на основі масштабних порівнянь випадкових послідовностей, генерованих на основі певного набору природних, чи на основі певної моделі частот вживання амінокислот/нуклеотидів. За цим принципом працює BLAST.

Описана статистика дає змогу передбачати, що оптимальне локальне вирівнювання може починатися з будь-якої пари вирівняних залишків амінокислот/нуклеотидів. Вирівнювання має мати певний розмір (бути достатньо довгим), щоб отримати велике значення S. За цих обставин вирівнювання, що починаються поблизу краю однієї з двох послідовностей, не будуть достатньо довгими, щоб стати HSP – лише тому, що вони будуть надто короткими для набрання “ваги” – великого рахунку попарного вирівнювання. Це *крайовий ефект* (edge effect), відображений на рис. 3.31. Для послідовностей більше 200 нуклеотидних чи амінокислотних залишків крайовий ефект, зазвичай, мізерний і його не коригують. Для коротших послідовностей цей ефект коригують за допомогою обчислення т.зв. “ефективної довжини” послідовностей.

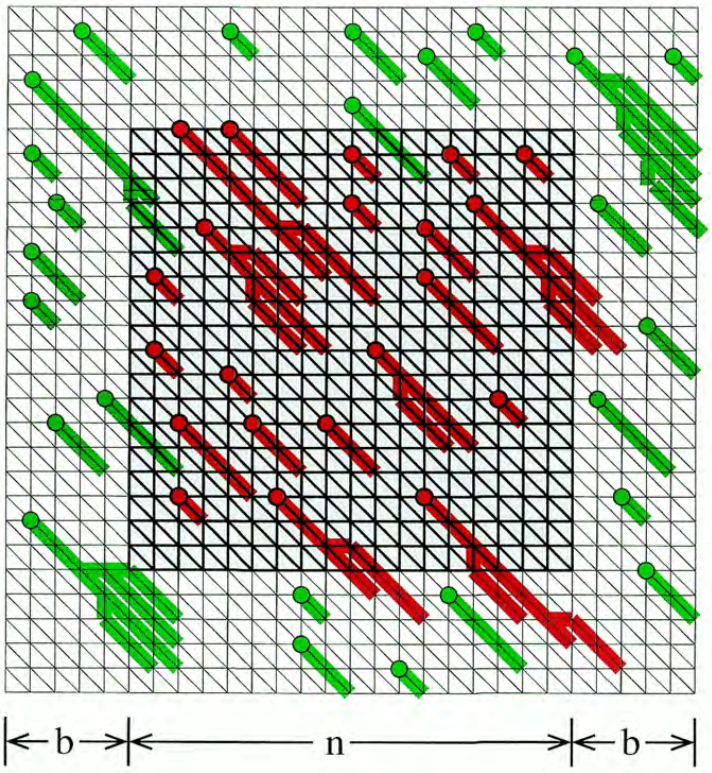


Рис. 3.31. Графічне відображення попарного вирівнювання і способу уникнення крайового ефекту. Площину обчислень рахунків вирівнювань розміром $n \times n$ оточено краєм розміру b. Обчислюють рахунки вирівнювань лише в межах району $n \times n$ (позначено червоним кольором). Інші вирівнювання не беруть до ваги (зелений колір), оскільки вони знаходяться на краю площини вирівнювання і, ймовірно, будуть надто короткими, щоб набрати великого рахунку і, відповідно, достатньої статистичної ваги, що зазначено у тексті. Зауважимо, що деякі прийняті вирівнювання заходять у крайовий простір, а деякі з тих, що ігнорують, – у зону $n \times n$, – оскільки класифікація вирівнювань залежить від точки початку вирівнювання (якір – позначено колом на зелених і червоних лініях). Рисунок адаптований з: doi: 10.1093/nar/29.2.351

Фактично коригування полягає у вкороченні послідовностей довжиною m і n до ефективних довжин m' і n':

$$m' = m - \ln Kmn/H \; ; \tag{36}$$

$$n' = n - lnKmn/H \; , \tag{37}$$

де H – *відносна ентропія* використаної матриці рахунків, або середній очікуваний рахунок на одну вирівняну пару залишків у вирівнюванні двох випадкових послідовностей; H – середня кількість бітів на позицію вирівнювання, завжди має додатне значення:

$$H = -\sum_{i=1}^{20} \sum_{j=1}^{i} q_{ij} \lambda S_{ij} \tag{38}$$

У разі використання матриці BLOSUM62 типові значення λ, K і H становлять 0,318, 0,13 і 0,40, відповідно. Ці коефіцієнти можна використовувати для приблизного коригування крайового ефекту і визначення статистичної значущості отриманих значень S.

Приблизно ¼ всіх амінокислотних залишків у послідовностях розташовано в районах низької складності (low complexity regions; див. попередній розділ). Такі райони, зазвичай, характеризуються використанням лише декількох з 20-ти можливих амінокислот; наприклад, описані пролін- чи лейцин-багаті сегменти білків тощо. Вирівнювання двох районів низької складності може мати дуже високе значення S, яке, однак, не має нічого спільного з порядком вживання залишків у послідовності, а пов'язане лише з особливим амінокислотним складом усього сегмента. Райони низької складності виникають, найімовірніше, внаслідок помилок реплікації, тому їхнє вирівнювання малозмістовне. Зважаючи на зазначене, сучасні програми попарного вирівнювання спочатку аналізують амінокислотні послідовності, маючи змогу виключити з вирівнювання всі райони низької складності. У вирівнюваннях ці райони позначені символом X або малими літерами абетки (у випадку амінокислотних послідовностей) або N (у випадку ДНК).

Рівняння (28) і (31), що описують залежність очікуваного числа випадкових вирівнювань E від розміру простору пошуку, K, λ і рахунку S – ключові у цьому пункті. Розглянемо на конкретних прикладах, що означає зменшення/збільшення числового значення різних складових цих рівнянь. Величина E зменшується експоненційно зі зростанням S. Фактично E описує випадковий фоновий шум у збігах двох послідовностей. Нехай білок φ вирівнювали з базою даних, що складається з 10 тисяч послідовностей. Отримали вирівнювання з рахунком $S = 20$, якому приписане значення $E = 1$. Це означає, що у використаній базі даних можна очікувати на одне вирівнювання φ з випадковою (негомологічною) послідовністю, що матиме рахунок $S = 20$. Отже, статистично значущими (невипадковими) будуть вирівнювання, що матимуть $E << 1$. Наскільки великим має бути величина рахунку S, щоб досягнути таких значень E? Станом на липень 2019 р. білок завдовжки 400 амінокислотних залишків можна порівняти проти бази даних NCBI Protein розміром 81×10^9 залишків. Для високої надійності результату вважатимемо, що *мінімально прийнятне значення* E дорівнює 0,05 (апелюючи до усталеного рівня p у статистичному аналізі). Тоді нормалізований (біт-)рахунок S' має бути рівним або більшим 49:

$$S' = \log_2\left(\frac{N}{E}\right) = \log_2\left(\frac{400 \times 81\,000\,000\,000}{0{,}05}\right) \approx 49.$$

Описаний спосіб знаходження S' походить з рівняння (31), з якого S' виразили як функцію простору пошуку N і числа очікування E.

Сьогодні розмір баз даних дуже збільшився, порівняно з тим часом, коли розробляли ТКА. Тому білки, здебільшого, мають гомологи у базах, які при вирівнюванні матимуть дуже високі рахунки, а, отже, дуже низькі значення E. Наприклад, уявімо, що під час аналізу заданої послідовності і бази даних, що

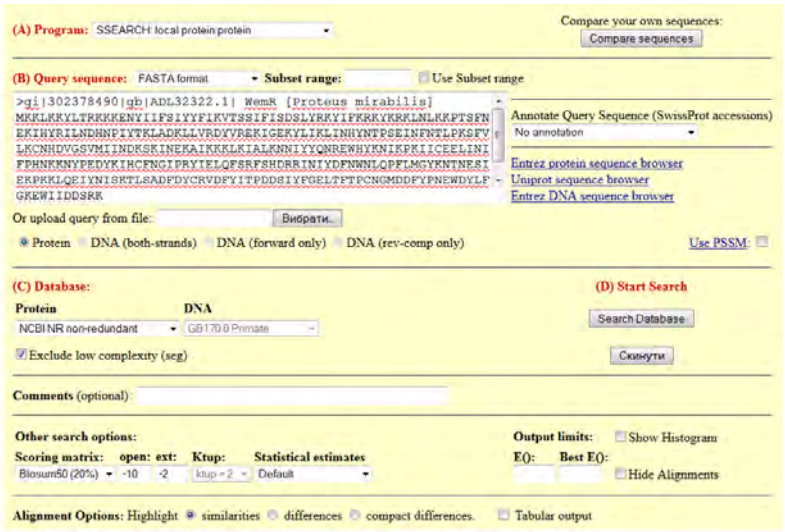
формують простір пошуку розміром N залишків, отримали попарне вирівнювання з рахунком S = 200 й $E = 10^{-51}$. Це означає, що в такій базі можна знайти 10^{-51} випадкових послідовностей, що вирівнюються із заданою з рахунком 200. Число це менше найближчого цілого числа – одиниці. Отож у використаній базі даних фактично немає жодної випадкової послідовності, яка б вирівнялася із заданою з рахунком S = 200. Суто теоретично можна очікувати, що зі збільшенням розміру простору пошуку в 10^{51} разів вдасться натрапити на одне випадкове вирівнювання з S = 200.

3.5.5. Аналіз баз даних за допомогою оптимальних методів вирівнювання. Попередньо розглянуті способи вирівнювання двох заздалегідь визначених послідовностей, однак для щойно встановленої послідовності нуклеотидів чи амінокислот неможливо *a priori* вибрати одну чи декілька, що виявлятимуть гомологію, а, отже, підлягатимуть оптимальному вирівнюванню. Для пошуку гомологів необхідно почергово вирівняти задану послідовність (query) з усіма послідовностями (subject) певної бази даних. Сучасні бази даних, такі як GenBank, містять десятки мільйонів послідовностей, і практично кожна задана послідовність матиме в них високоподібного відповідника. Протилежний результат (брак високоподібної послідовності) стає дедалі рідкіснішим зі зростанням обсягів баз даних. Результати порівняння заданої послідовності з базою даних репрезентують у вигляді *списку хітів* (hit list) – переліку n найподібніших послідовностей, після чого подають попарні вирівнювання послідовностей разом зі статистичною оцінкою. Результат вирівнювання визначає вибір параметрів – тип матриці заміщення, штрафування і спосіб статистичного оцінювання. Відома достатньо велика кількість онлайн-сервісів, що дають змогу виконувати попарне вирівнювання на основі АСУ і АНВ. Наприклад, можна скористатися біоінформатичним сервером університету Вірджинії (http://fasta.bioch.virginia.edu/), що містить низку програм вирівнювання, зокрема і програмне втілення АСУ для аналізу амінокислотних послідовностей (рис. 3.32). Після введення у вікно певної послідовності (WemR (ADL32322) у цьому випадку), вибору бази даних для аналізу (NCBI NR non redundant), застосування параметрів пошуку за замовчуванням та запуску програми отримуємо результат, який складається з опису параметрів вирівнювання, списку хітів з описом статистичних параметрів кожного вирівнювання (рис. 3.32, *б*) і власне вирівнювань (рис. 3.32, *в*). Список хітів поданий у низхідному порядку, тобто спочатку йдуть найподібніші послідовності, і вниз – дедалі менш подібні:

```
The best scores are: s-w bits E(23641837) %_id %_sim alen
gb|ADL32322.1| WemR [Proteus mirabilis] ( 311) 1758 376.5  1e-101 1.000 1.000  311
ref|YP_002152804.1| glycosyltransf [Proteus mir ( 291)  896 195.8 2.5e-47 0.468 0.734
282
```

Розглянемо організацію списку хітів. Перший рядок – це вирівнювання послідовності білка WemR, що задана дослідником, з білком WemR із бази NCBI. Очікувано тут спостерігаємо стовідсоткову ідентичність. Програма подає низку статистичних параметрів для послідовності (311 амінокислотних залишків). Далі

а



б

```
# ssearch36 -p -q -w 80 -m 9i -m 6 -f -10 -V "!ann_feats2ipr.pl --neg" -S -g -2 TMP.q N

SSEARCH performs a Smith-Waterman search
 version 36.3.6 May, 2013(preload9)
Please cite:
 T. F. Smith and M. S. Waterman, (1981) J. Mol. Biol. 147:195-197;
 W.R. Pearson (1991) Genomics 11:635-650

Query: TMP.q
  1>>>QUERY - 311 aa
Library: NCBI NR non-redundant
  8123359852 residues in 23641837 sequences

Statistics:  Expectation_n fit: rho(ln(x))= 7.0904+/-0.000186; mu= 4.6664+/- 0.010
 mean_var=77.8523+/-15.676, 0's: 22 Z-trim(131.3): 31  B-trim: 10 in 1/65
 Lambda= 0.145358
 statistics sampled from 60000 (418533) to 23638887 sequences
Algorithm: Smith-Waterman (SSE2, Michael Farrar 2006) (7.2 Nov 2010)
Parameters: BL50 matrix (15:-5)xS, open/ext: -10/-2
 Scan time: 797.630

Annotation symbols:
 * : phosphorylation
 = : active site
 @ : site
 ^ : binding
 ! : metal binding

The best scores are:                                      s-w bits E(23641837) %_id  %_sim  alen
gb|ADL32322.1| WemR [Proteus mirabilis]              ( 311) 1758 376.5  1e-101 1.000 1.000  311 align
ref|YP_002152804.1| glycosyltransferase [Proteus mirabi ( 291)  896 195.8 2.5e-47 0.468 0.734  282 align
ref|YP_006215255.1| glycosyltransferase [Providencia st ( 313)  890 194.5 6.4e-47 0.460 0.705  285 align
ref|ZP_16360088.1| glycosyltransferase [Providencia bur ( 314)  889 194.3 7.4e-47 0.460 0.716  285 align
```

в

```
>>ref|YP_002152804.1| glycosyltransferase [Proteus mirabilis HI              (291 aa)
 s-w opt: 896  Z-score: 1014.6  bits: 195.8 E(23641837): 2.5e-47
Smith-Waterman score: 896; 46.8% identity (73.4% similar) in 282 aa overlap (23-304:6-286)
Entrez Lookup  Re-search database  General re-search
```


```
[alignment]

               10        20        30        40        50        60        70        80
QUERY  MKKLKKYLTRKKKENYIIFSIYYFIKVTSSIFISDSLYRKYIFKRKYKRKLNLKKPTSFNEKIHYRILNDHNPIYTKLAD
                           :..:   ..   : .: :  : :.   . :...: :.:::. :.: :..:::.:::
ref|YP                   MKILEYYFKQLRTLAQPDRVYLKNKFIRNLGYQPNFRHPMSLNEKINARMLFDRDPIYTRLAD
                                10        20        30        40        50        60

               90       100       110       120       130       140       150       160
QUERY  KLLVRDYVREKIGEKYLIKLINHYNTPSEINFNTLPKSFVLKCNHDVGSVMIINDKSKINEKAIKKKLKIALKNNIYYQN
       :. ::.::.::::::::.:..: :  :.::..:::. :::::::: ::..:  :::... .  :::.. :: :.:. .
ref|YP KISVREYVKEKIGEKYLVKILNTYRHPNEIELNTLPNRFVLKCNHDSGSTIICLDKSQFDLEMAKKKLNFHLKRNLYHVT
            70        80        90       100       110       120       130       140
```

Рис. 3.32. Аналіз бази даних NCBI за допомогою АСУ: загальний вигляд діалогового вікна (*а*), сторінки результатів і початку списку хітів (*б*) і одного з вирівнювань – № 2 зі списку хітів (*в*)

іде рахунок вирівнювання *S*, який обчислив АСУ (у випадку першого вирівнювання – це 1 758). Біт-рахунок S′ (bits) поданий насамкінець (376,5 у цьому випадку). Його можна використовувати для порівняння результатів вирівнювання, виконаних на основі різних матриць заміщення і системи штрафів (див. пункт 3.5.4). Найістотнішою статистичною оцінкою є величина *E*. Тут її значення становить 1e-101, або 10^{-101}. Це означає, що у проаналізованій базі NCBI є 10^{-101} випадкових послідовностей, яка б мала з білком WemR рахунок вирівнювання $S \geq 1\,758$. У вирівнюваннях оперують дискретними величинами – кількостями послідовностей, які набувають значень лише цілих чисел. Оскільки обчислене число значно менше одиниці, то фактично це означає, що в застосованій базі немає жодної випадкової послідовності, яка б вирівнялась із заданою і мала рахунок ≥ 1 758. Тому можна стверджувати, що отриманий хіт – невипадковий. Загалом дві амінокислотні послідовності можна вважати гомологічними, якщо їхнє значення $E \leq 0{,}001$ або, принаймні, ≤ 0,05 (за умови правильного оцінення статистичних параметрів і розміру бази даних – щонайменше 10 000 послідовностей). Тут необхідно наголосити, що малі (статистично достовірні) значення *E* можуть слугувати доказом гомології білків, але протилежне твердження (великі значення *E* слугують доказом браку гомології) – некоректне. Часто можна отримати значення *E* більше 0,05 при вирівнюванні білків, які є філогенетично споріднені і для яких біохімічно доведені однакові функції. Тут недостовірні статистичні показники є лише характеристикою конкретного вирівнювання, де вирішальне значення має те, що саме дослідник обрав як задану послідовність (query). Також у списку статистичних показників подають відсоток ідентичності (%_id) та подібності (%_sim) і кількість вирівняних амінокислотних залишків (alen). Хоча відсоток ідентичності/подібності часто використовують як основний показник якості вирівнювання і доказ гомології, варто ще раз зазначити, що значення *E* – набагато надійніша міра якості і гомології.

3.5.6. FastA – перший евристичний підхід. АСУ й АНВ порівнюють кожну літеру однієї послідовності з кожною літерою іншої, і репрезентують клас оптимальних методів аналізу, де гарантоване виявлення найкращого способу вирівнювання двох послідовностей. Однак цей процес займає багато часу та обчислювальної потужності. Якщо база геномних даних, з якою порівнюють певну послідовність, велика, то більшість часу алгоритм зайнятий, фактично, непотрібною роботою, тобто обчисленням значень рахунків у секторах матриць, що напевне не містять оптимального шляху вирівнювання (див., наприклад, правий верхній кут матриці на рис. 3.23–3.25). Зважаючи на сказане, розробили *евристичні* методи, які використовують певний набір правил, що спрощують і пришвидшують вирівнювання послідовностей за рахунок незначного зниження точності аналізу. Один із найпоширеніших евристичних підходів полягає у розбитті заданої послідовності на короткі сегменти – слова (words) – і подальшому пошуку слів, спільних для двох аналізованих послідовностей. Тобто із заданої послідовності створюється перелік (*індекс*) слів (рис. 3.33), які використовують для порівняння з базою даних. Оскільки слова коротші, ніж задана послідовність, то процес вирівнювання пришвидшується.

Перший евристичний алгоритм – FastA – описаний Девідом Ліпманом та Уільямом Пірсоном 1985 р. Спочатку FastA шукає однакові слова у двох обраних послідовностях (рис. 3.34, *а*). Розмір слова *k* (*k* - тупль) визначає швидкість і

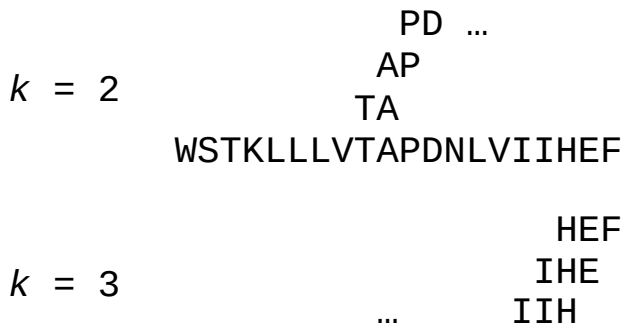


Рис. 3.33. Індекс слів, що походять із заданої послідовності. Довжина слова (*k*-тупль) *k* – дві й три літери

чутливість методу. Чим довше слово, тим менший час аналізу (довших слів менше, ніж коротших), але нижча чутливість (зростає ймовірність пропустити ідентичні ділянки розміром <*k*).

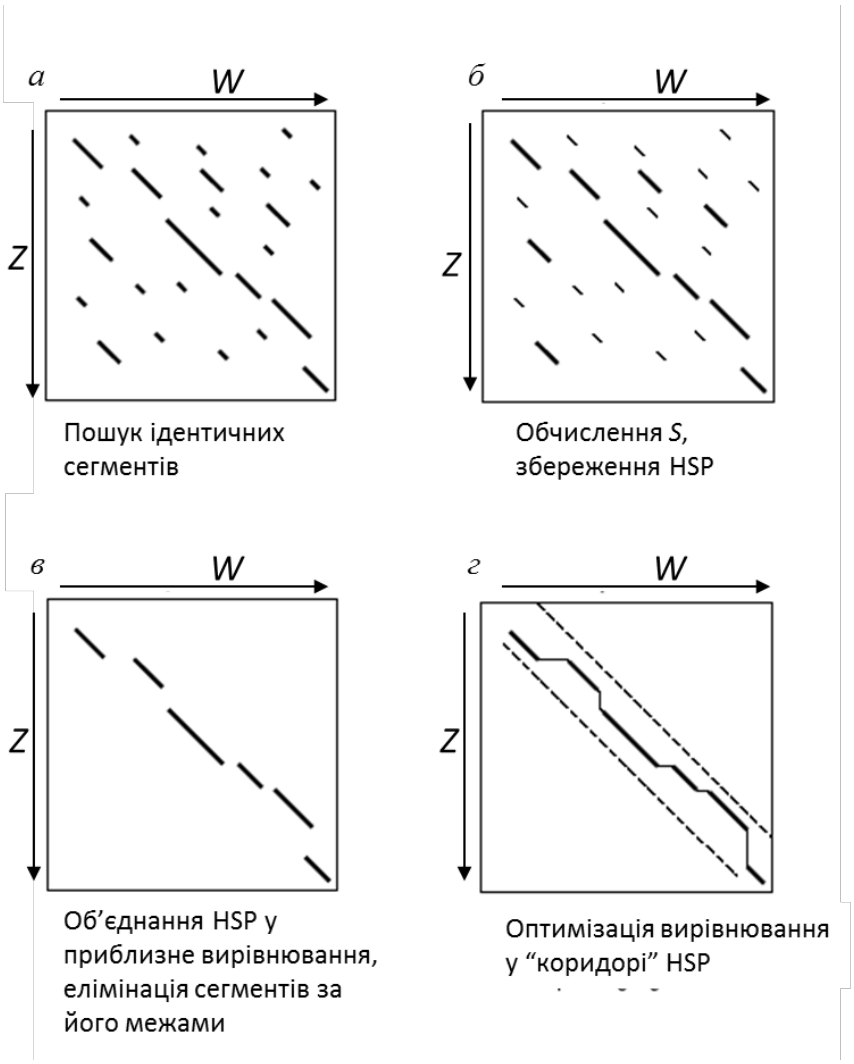


Рис. 3.34. Принцип роботи алгоритму FastA на прикладі вирівнювання двох послідовностей, *Z* і *W* (подробиці алгоритму наведено в основному тексті вище)

Для вирівнювання амінокислотних послідовностей значення k, зазвичай, становить два, для нуклеотидних – шість. Порівняння k-туплів з двох послідовностей, фактично, веде до звуження простору вирівнювання до діагональної лінії. Далі FastA не аналізує усі збіги слів, а лише ті, які розташовані поряд. Зазвичай програма відбирає десять локальних сегментів ідентичності (HSP), розміщених уздовж діагоналі, і обчислює для них рахунок вирівнювання S (див. рис. 3.34, *б*) згідно з певною матрицею заміщення (напр., BLOSUM50). Найвище значення S подають на сторінці результатів як init1. Якщо виявлені HSP мають S вищі певного порогового значення T, то далі програма прораховує можливі способи з'єднання окремих HSP в одне приблизне локальне вирівнювання (див. рис. 3.34, *в*). Комбіноване значення S для цих вирівнювань подане на сторінці результатів як initn. Далі програма виконує вичерпний пошук (за допомогою АСУ) оптимального локального вирівнювання (з розривами), у межах вузького коридора приблизного вирівнювання (див. рис. 3.34, *г*). Рахунок S оптимального попарного вирівнювання поданий на сторінці результатів як opt_score.

Зрештою для послідовностей, що потраплять на сторінку результатів (їхню кількість визначає користувач), програма виконує повний пошук оптимального вирівнювання, не обмежуючись коридором вихідного (приблизного) вирівнювання. Подають тільки одне оптимальне вирівнювання для кожної послідовності з бази даних, отож у разі багатодоменних білків є ризик втрати цікавої інформації. Такі білки потрібно додатково проаналізувати за допомогою програм локального вирівнювання (наприклад, дотплот чи lalign: http://fasta.bioch.virginia.edu). Сторінку результатів і вирівнювання наведено на рис. 3.35. Шапка сторінки результатів містить інформацію про параметри пошуку (тип матриці, систему штрафів за розриви, розмір слова k тощо).

Як і оптимальні методи аналізу, FastA обчислює для кожного вирівнювання (рис. 3.35, *б*) декілька статистичних параметрів, що дають змогу оцінити значущість отриманих результатів – це рахунок Z (див. пункт 3.5.4) та величину очікування E для заданого значення Z і розміру бази даних. Подають також відсотки ідентичності і подібності, рахунок вирівнювання, обчислений АСУ. Найінформативнішим для виявлення спорідненості є значення числа очікування E.

3.6. Родина програм BLAST (Basic Local Alignment Search Tool). BLAST – це збірна назва найпопулярнішого сьогодні пакета онлайн-програм для локального попарного вирівнювання послідовностей, що дає змогу швидко ідентифікувати у великих базах даних послідовності, високоподібні до заданої. Програми BLAST доступні через веб-сайт NCBI: www.ncbi.nlm.nih.gov/blast. Історично першою була BLASTP (або Protein BLAST) – програма для попарного вирівнювання амінокислотних послідовностей. BLASTP залишається одним із найкорисніших і найуживаніших знарядь біоінформатики, яке постійно оновлюють та доповнюють. На її основі буде розглянутий принцип функціонування BLAST.

а

```
Algorithm: FASTA (3.8 Nov 2011) [optimized]
Parameters: BL50 matrix (15:-5)xS, open/ext: -10/-2
 ktup: 2, E-join: 1 (0.355), E-opt: 0.2 (0.0939), width:  16
 Scan time: 316.440

Annotation symbols:
 * : phosphorylation
 = : active site
 @ : site
 ^ : binding
 ! : metal binding
```

The best scores are:		opt	bits	E(23641837)	%_id	%_sim	alen	
gb|ADL32322.1| WemR [Proteus mirabilis]	(311)	1758	410.4	6.3e-112	1.000	1.000	311	align
ref|YP_002152804.1| glycosyltransferase [Proteus mirabi	(291)	896	213.8	9.7e-53	0.468	0.734	282	align
ref|YP_006215255.1| glycosyltransferase [Providencia st	(313)	890	212.4	2.7e-52	0.460	0.705	285	align
ref|ZP_16360088.1| glycosyltransferase [Providencia bur	(314)	889	212.1	3.2e-52	0.460	0.716	285	align
ref|ZP_02961307.1| hypothetical protein PROSTU_03325 [P	(313)	888	211.9	3.7e-52	0.460	0.705	285	align
ref|ZP_16368890.1| glycosyltransferase [Providencia sne	(310)	877	209.4	2.1e-51	0.446	0.723	285	align
ref|ZP_06125904.2| glycosyltransferase [Providencia ret	(319)	869	207.6	7.7e-51	0.431	0.721	283	align
ref|ZP_16356458.1| glycosyltransferase [Providencia ret	(314)	860	205.5	3.1e-50	0.414	0.719	292	align
ref|ZP_05973215.1| glycosyltransferase [Providencia rus	(313)	857	204.8	5e-50	0.445	0.721	283	align
ref|ZP_06715054.1| glycosyltransferase [Edwardsiella ta	(312)	855	204.4	6.9e-50	0.405	0.722	299	align
ref|YP_003296034.1| hypothetical protein ETAE_1986 [Edw	(305)	813	194.8	5.1e-47	0.378	0.712	299	align

[show/hide all alignments]

б

```
>>ref|YP_002152804.1| glycosyltransferase [Proteus mirabilis HI              (291 aa)
 initn: 817 init1: 741 opt: 896  Z-score: 1111.7  bits: 213.8 E(23641837): 9.7e-53
Smith-Waterman score: 896; 46.8% identity (73.4% similar) in 282 aa overlap (23-304:6-286)
```

Entrez Lookup Re-search database General re-search

[alignment]

```
               10        20        30        40        50        60        70        80
QUERY  MKKLKKYLTRKKKENYIIFSIYYFIKVTSSIFISDSLYRKYIFKRKYKRKLNLKKPTSFNEKIHYRILNDHNPIYTKLAD
                           :..:   ..   : .: :  : :.   . :...: :.::::. :.: :..::::.:::
ref|YP                  MKILEYYFKQLRTLAQPDRVYLKNKFIRNLGYQPNFRHPMSLNEKINARMLFDRDPIYTRLAD
                                10        20        30        40        50        60

               90       100       110       120       130       140       150       160
QUERY  KLLVRDYVREKIGEKYLIKLINHYNTPSEINFNTLPKSFVLKCNHDVGSVMIINDKSKINEKAIKKKLKIALKNNIYYQN
       :. ::.::.:::::::::.:..: :  :.:::..::::. :::::::: ::..:  :::... .  :::::.. :: :.:. .
ref|YP KISVREYVKEKIGEKYLVKILNTYRHPNEIELNTLPNRFVLKCNHDSGSTIICLDKSQFDLEMAKKKLNFHLKRNLYHVT
            70        80        90       100       110       120       130       140

               170       180       190       200       210       220       230       240
QUERY  REWHYKNIKPKIICEELINIFPHNKKNYPEDYKIHCFNGIPRYIELQFSRFSHDRRINIYDFNWNLQPFLMGYKNTNESI
       ::::::::. .::::: :..:  .    :   .::::.: :.: :...:  .  . ::.:: .:.:::: .   :: : :
ref|YP REWHYKNISAQIICEEYIDLFDTQCPMIPTTCRIHCFHGQPHYAEIDISDTKGCEYINVYDTQWQLQPFTLEDPNTPEYI
           150       160       170       180       190       200       210       220

               250       260       270       280       290       300       310
QUERY  EKPKKLQEIYNISKTLSADFDYCRVDFYITPDDSIYFGELTFTPCNGMDDFYPNEWDYLFGKEWIIDDSRK
       ::: ..... .... :.  : :::::: .. . .. :.:.::::  :   . :.:::: .:. :
ref|YP EKPLQFDNMLTLANKLANGFAYCRVDFLMS-ETTLLFSEMTFTPNAGRIPIKPKEWDYKLGELWTPIAS
           230       240       250       260       270       280       290
```

Рис. 3.35. Список хітів, статистичні параметри FastA (*а*) і локальне вирівнювання (*б*). Задана послідовність – WemR

Своїй популярності BLAST завдячує передусім швидкості і точності аналізу, що можливі завдяки евристичному підходу до вирівнювання послідовностей. У функціонуванні BLAST можна умовно виділити три основні фази: пошук спільних слів, сканування бази даних, нарощування хітів (hit/word extension).

Як і FastA, BLAST аналізує не цілі послідовності, а *слова*, що походять з них. Розмір слова W у версіях BLASTP до 2015 року становив три, а зараз, з огляду на безпрецедентне зростання розмірів баз даних – шість; для ДНК W становить 6÷11. Важливою новацією BLASTP стало застосування принципу *сусідніх слів* (neighborhood words). А саме: програма шукає не ідентичні слова у заданій послідовності і базі даних, а слова, що мають рахунок S, більший від певної порогової величини T. Значення S обчислюють на основі матриць заміщення; BLAST за замовчуванням використовує BLOSUM62. Пошук сусідніх слів проілюстровано на рис. 3.36 для послідовності RCPHHERCSD. Першим

потенційного хіта припиняється, коли зниження кумулятивного рахунку (щодо його максимально досягнутого значення S_{max}) перевищує певну *величину падіння* (dropoff level) X (рис. 3.37, *б*).

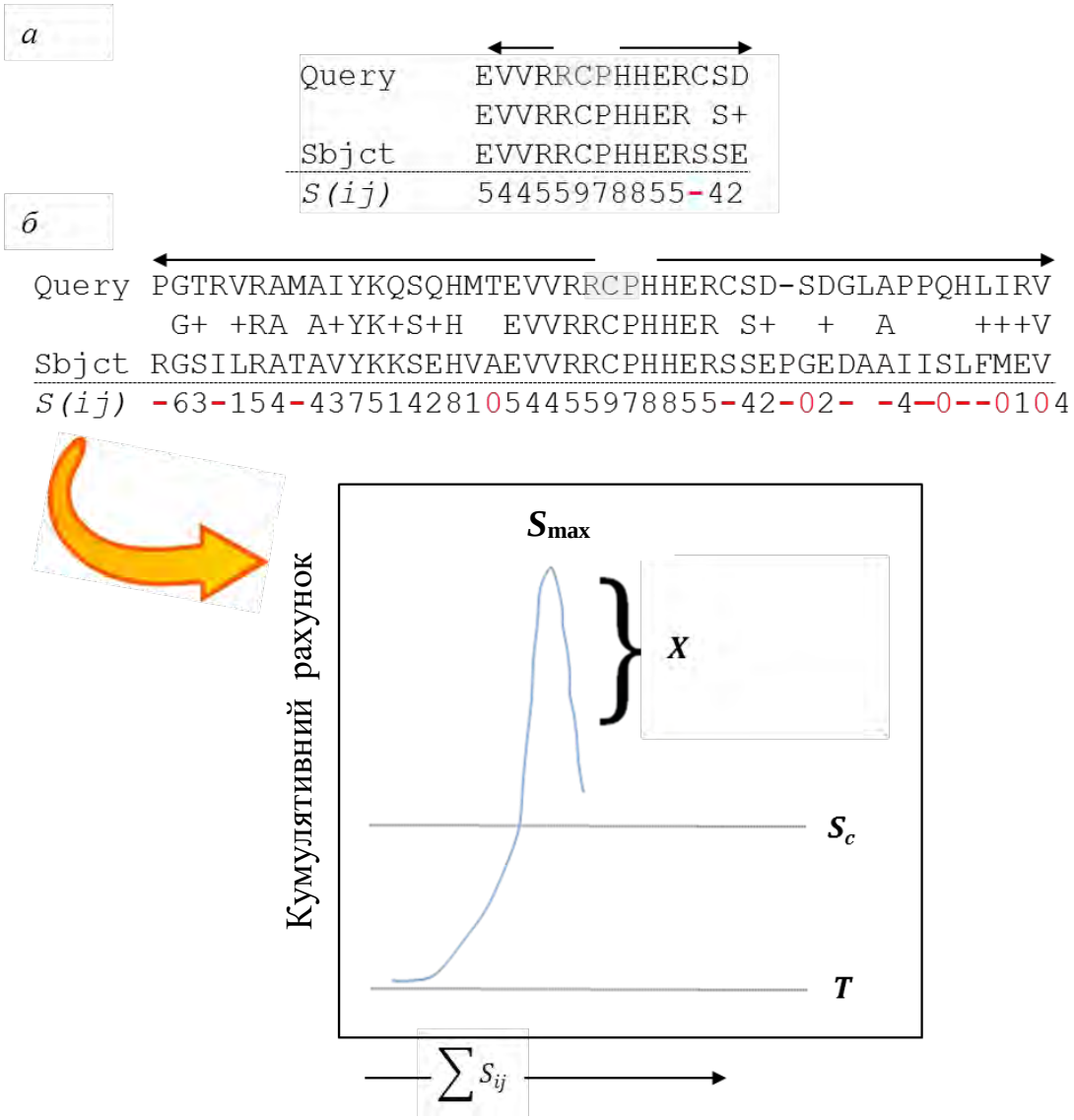


Рис. 3.37. Нарощування спільного слова в BLAST. Спільне слово RCP, виділене сірим кольором, нарощується у два боки по одній літері (*а*). Подальше нарощування слова RCP натрапляє на зону низької подібності (*б*), що веде до падіння кумулятивного рахунку (графік внизу). Під вирівнюваннями подане значення S_{ij}, червоним виділені від'ємні (дефіс) і нульові значення. На графіку проілюстровано параметри S_c, T та X, описані вище в основному тексті

Використання малих значень X покращує роботу BLAST за рахунок скорочення часу, який програма марнує на нарощування безперспективних слів (звісно, є незначний ризик загубити біологічно змістовні вирівнювання).

Усі описані етапи вирівнювання проілюстровано на рис. 3.38, *а-в*. Отже,

трилітерним словом у цій послідовності буде RCP. BLAST створює список усіх трилітерних слів, що мають рахунок *S*, близький до максимального (точний збіг – RCP – 21), але не нижче *T* (наприклад, 17). BLAST порівнює RCP з усіма можливими трилітерними словами (яких, у разі амінокислотних послідовностей, є 20^3 – 8 000). Так BLAST з усіх 8 000 варіантів створить множину сусідніх з RCP слів, тобто тих, що мають $T \geq 17$. Це, зокрема, будуть слова KCP ($S = 18$), QCP ($S = 17$), але не ECP ($S = 16$). Така стратегія пошуку дає змогу утримувати розмір слова *W* великим (що прискорює аналіз), не жертвуючи чутливістю пошуку. Параметр *T* стає критичним у BLAST, що визначає швидкість і чутливість аналізу, водночас *W* практично не змінюють. Збільшення *T* зменшує кількість фонових збігів слів і пришвидшує пошук. Зменшення *T* дає змогу виявити віддаленіших "родичів" заданої послідовності.

Слово	BLOSUM62	*S*
RCP	5 + 9 + 7	21
KCP	2 + 9 + 7	18
QCP	1 + 9 + 7	17
ECP	0 + 9 + 7	16

$W = 3$
$T = 17$

Рис. 3.36. Створення списку сусідніх слів для RCP. Суцільною жирною лінією відділено сусідні слова від тих, що мають *S* нижче порогового значення *T*

Далі BLAST сканує базу даних з метою пошуку у ній слів, що подібні (тобто мають $S \geq T$) до множини сусідніх слів, знайдених на попередньому етапі у заданій послідовності. Усі збіги – потенційні хіти, однак вони дуже короткі. Далі BLAST намагається наростити потенційні хіти, додаючи поступово по одній літері справа і зліва (рис. 3.37, *а*) і постійно обчислюючи рахунок розширеного хіта (*кумулятивний рахунок*). Якщо кумулятивний рахунок вирівнювання постійно зростає і сягає певної *величини відсікання* S_c (cutoff level), то вдається отримати локально оптимальне вирівнювання – HSP. Величину S_c встановлюють емпірично, за допомогою аналізу розподілу рахунків вирівнювань випадкових послідовностей. Ця величина має бути достатньо високою, щоб гарантувати статистичну значущість HSP. Якщо ж у процесі розширення слова накопичуються переважно низькі значення внаслідок незбігів літер і штрафів за розриви (рис. 3.37, *б*), то кумулятивний рахунок швидко знижується, як і ймовірність того, що він згодом зросте до значення S_c. Це спостереження покладено в основу другого елементу евристики в BLAST: нарощування

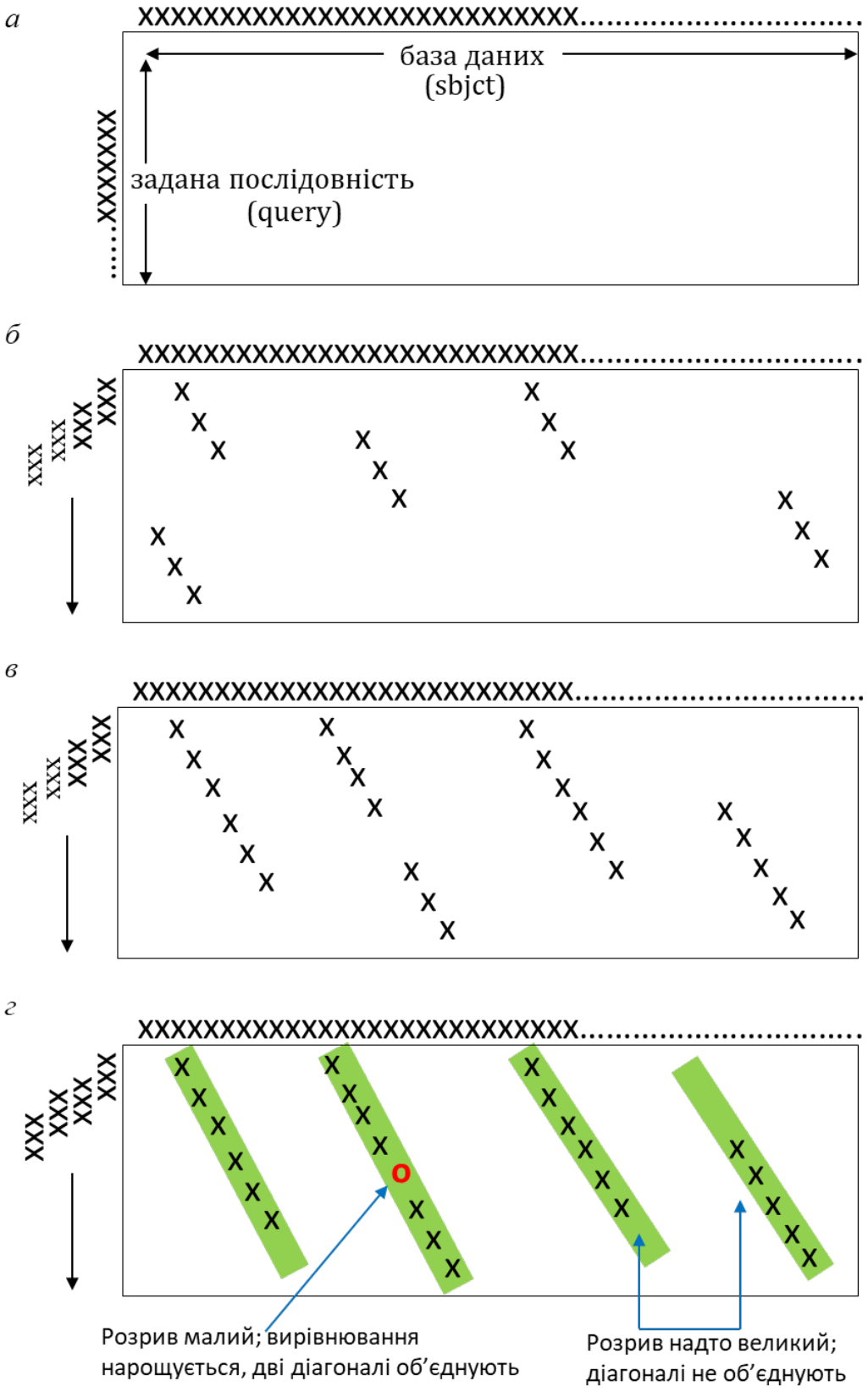


Рис. 3.38. Графічна інтерпретація алгоритму BLAST: *а* – ініціалізація пошуку; *б* – створення списку сусідніх слів і їхній пошук у базі даних; *в* – нарощування і термінація слів; *г* – об'єднання HSP

програма генерує низку HSP уздовж аналізованих послідовностей. Далі програма "намагається" з'єднати окремі HSP у довші послідовності (рис. 3.38, *г*). Як кандидати на об'єднання насамперед розглядають ті HSP, що лежать на одній діагоналі у спільному "коридорі (див. пункт, присвячений FastA) і розрив між якими – невеликого розміру (тоді розмір штрафу буде невеликий і його компенсує значне зростання кумулятивного рахунку об'єднаних HSP). Якщо на одній діагоналі існує декілька варіантів об'єднання HSP, то програма обирає найоптимальніший. Для вибору оптимального варіанта об'єднання можна використати два методи: Пуасона і суми рахунків. Наприклад, нехай є дві пари HSP з парою рахунків (65, 40) та (52, 45). За методом Пуасона буде обрана пара з максимально нижчим рахунком (45>40), метод суми рахунків – першій парі, оскільки 65+40 (105) більше, ніж 52+45 (97). У перших версіях BLAST використовували метод Пуасона, тоді як сучасні версії застосовують другий метод. На сторінці результатів BLAST подає вирівнювання з розривами, що може включати всі початково знайдені HSP. Наводять лише ті вирівнювання, величина *E* яких нижча за порогове значення *E* – значення *E* бази даних для заданих значень *S* (див. вище).

У таблиці 3.1 підсумовані всі кроки алгоритму BLASTP, використовуваного сьогодні для аналізу баз даних.

Таблиця 3.2

Етапи алгоритму BLASTP

Крок	Операція	Примітка
1	Видалити райони низької складності і повтори із заданої послідовності	Програма SEG
2	На основі заданої послідовності створити список слів W	W=3
3	Створити перелік "сусідніх слів" до кожного слова, що знайдено у заданій послідовності	$S \geq T$
4	Пошук у базі даних (sbjct) співпадінь (hits) зі списком сусідніх слів, створеним на основі заданої послідовності (query)	
5	Нарощування слів з утворенням HSP (термінація – якщо кумулятивний рахунок знижується на величину X)	
6	Збереження HSP з високим рахунком	
7	Статистична оцінка відібраних HSP	E, P
8	Об'єднання HSP	
9	Генерування оптимального локального вирівнювання заданої послідовності з подібними у базі даних	
10	Формування сторінки результатів	

Розглянемо функціонування BLASTP, розміщеного на сервері NCBI. Нехай задана послідовність (query) – білок MoeH5 *Streptomyces ghananesis* ATCC14672

(його розглянуто у розділі 1). Дослідникові треба знайти гомологи цього білка. Якщо заданої послідовності немає у GenBank, то її копіюють у діалогове вікно BLASTP, що опишемо згодом. Послідовність білка MoeH5 вже наявна в базах даних. Тому доступ до неї можна отримати з початкової сторінки NCBI чи GenBank. На сторінці зі списком білків є лінк до BLAST (рис. 3.39). Він переадресовує користувача на початкову сторінку BLASTP (рис. 3.40), що складається з чотирьох основних розділів – вікна для введення заданої послідовності (Enter query sequence), меню вибору бази даних для аналізу (Choose search set), вибору програми вирівнювання (Program selection) і параметрів алгоритму (Algorithm parameters).

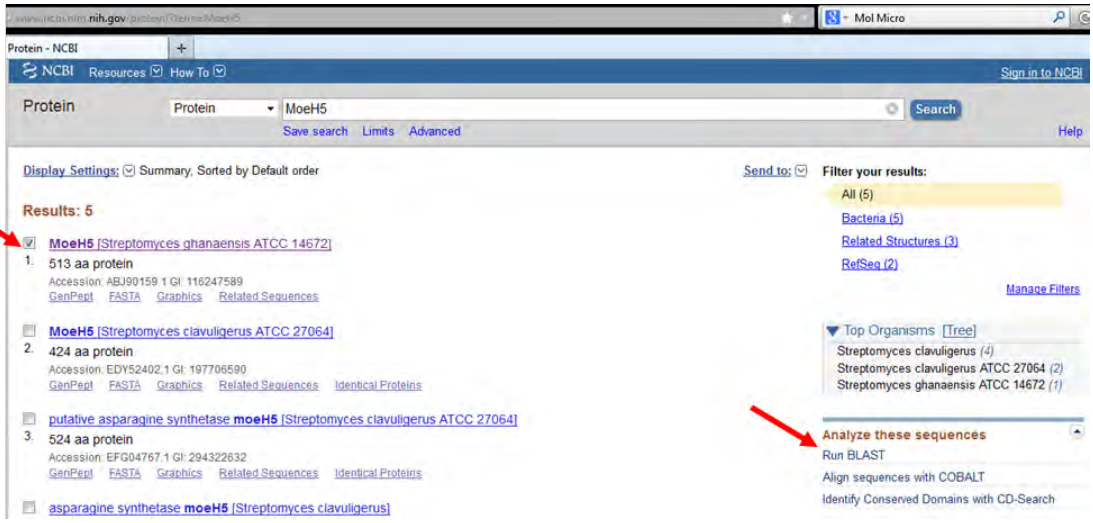


Рис. 3.39. Сторінка NCBI, що містить список білків MoeH5. Для аналізу обрано перший білок зі списку (червона стрілка зліва). Для порівняння обраного білка з базою даних можна скористатися лінком Run BLAST (червона стрілка справа); можна завантажити амінокислотну послідовність з цієї сторінки і скопіювати її у діалогове вікно BLASTP (див. основний текст)

У розглянутому випадку діалогове вікно містить номер доступу (accession) до амінокислотної послідовності MoeH5 – результат автоматичного переходу зі списку MoeH5-білків на сторінку BLAST при натисканні на команду Run BLAST (див. рис. 3.39). У діалогове вікно так само можна ввести номер gi, або ж власне амінокислотну послідовність у FASTA-форматі. У наступному розділі з меню потрібно обрати базу даних (вікно Database), проти якої дослідник хоче порівняти MoeH5. За замовчуванням програма використовує базу даних nr – усі неповторювані (nonredundant) білкові послідовності, що наявні в GenBank на момент аналізу. Дослідник може звузити коло пошуку до певної групи організмів чи одного виду (штаму), користуючись вікном Organism. У наступному розділі сторінки дослідник обирає програму аналізу. За замовчуванням застосовують blastp – вирівнювання "білок–білок". Альтернативними варіантами є PSI-, PHI- та DELTA-BLAST, про які йтиме мова далі. Нарешті, останнім розділом початкової сторінки BLASTP є налаштування параметрів алгоритму. За замовчуванням програма відображає не більше 500 хітів на сторінці результатів, розмір слова *W* становить шість; для вирівнювань застосовують матрицю заміщення

BLOSUM62, за відкриття розриву – штраф –11, за її розширення –1. Ці параметри оптимізовані з метою аналізу бази даних NCBI передусім для потреб ідентифікації подібних послідовностей, і їх радять застосовувати для початкового аналізу заданої послідовності. Змінюючи параметри програми, дослідник може досягти більшої селективності пошуку. Наприклад, можна змусити програму шукати віддаленіші послідовності за рахунок обрання інших матриць заміщення, побудованих на основі вирівнювання філогенетично віддаленіших послідовностей (BLOSUM40, BLSOUM50, PAM250).

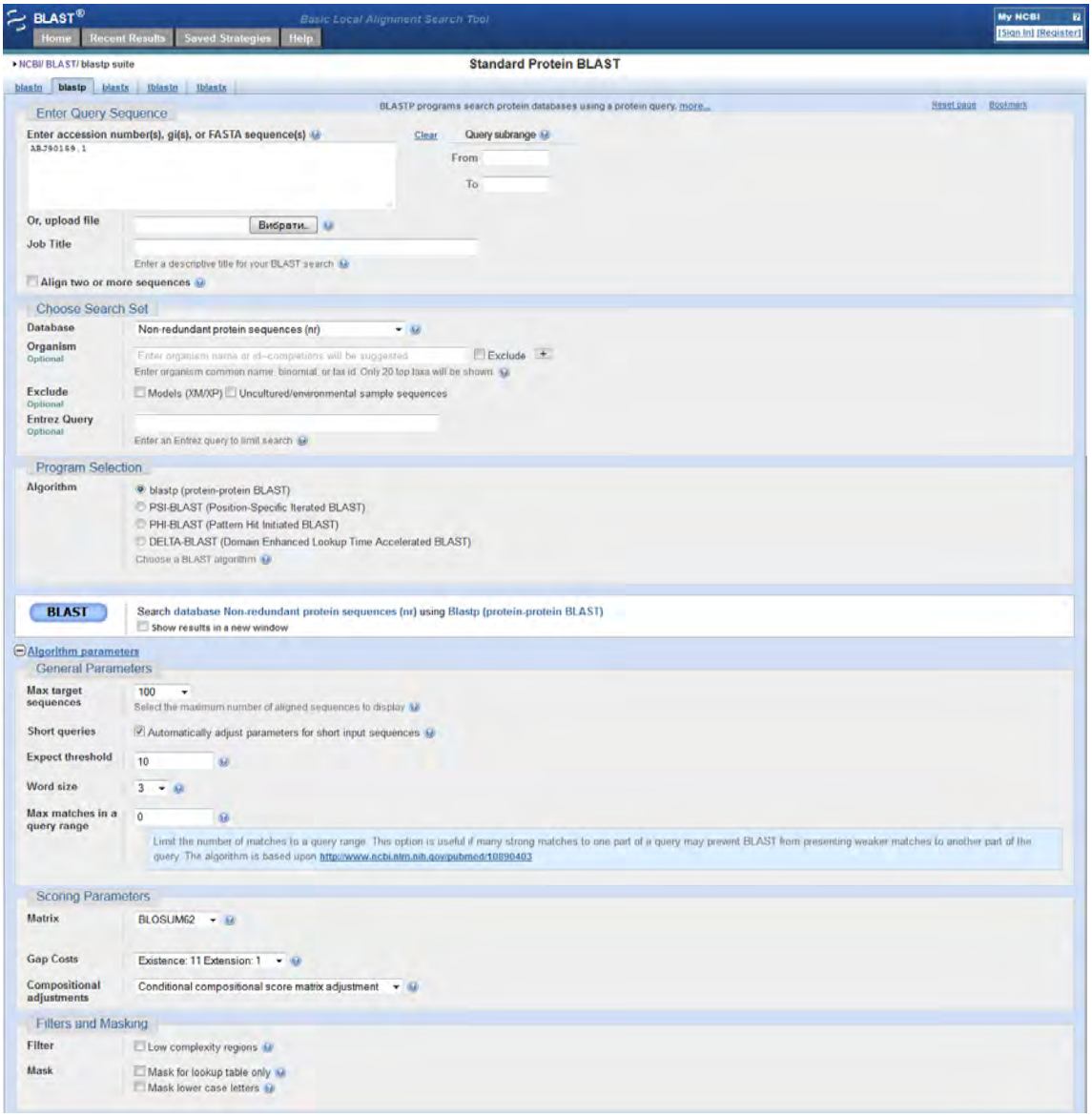


Рис. 3.40. Початкова сторінка BLASTP

Після налаштування відповідним чином усіх параметрів пошуку і запуску програми вирівнювання (команда BLAST над розділом “Параметри алгоритму”) дослідник переходить до вікна статусу пошуку (рис. 3.41). Воно сповіщає про прийняття запиту до виконання сервером NCBI і про те, скільки часу спливло з

моменту початку аналізу. Вікно статусу періодично оновлюється до моменту отримання результатів порівняння заданої послідовності із базою даних.

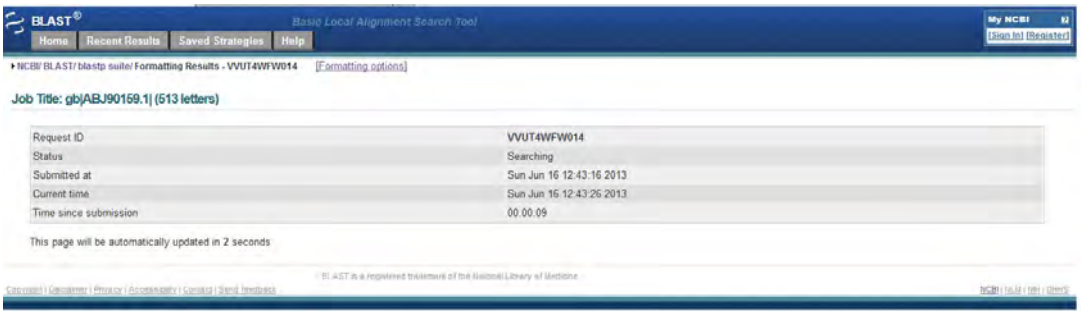


Рис. 3.41. Вікно статусу поданого запиту

Програма подає результат порівняння (рис. 3.42). Як і вихідна сторінка

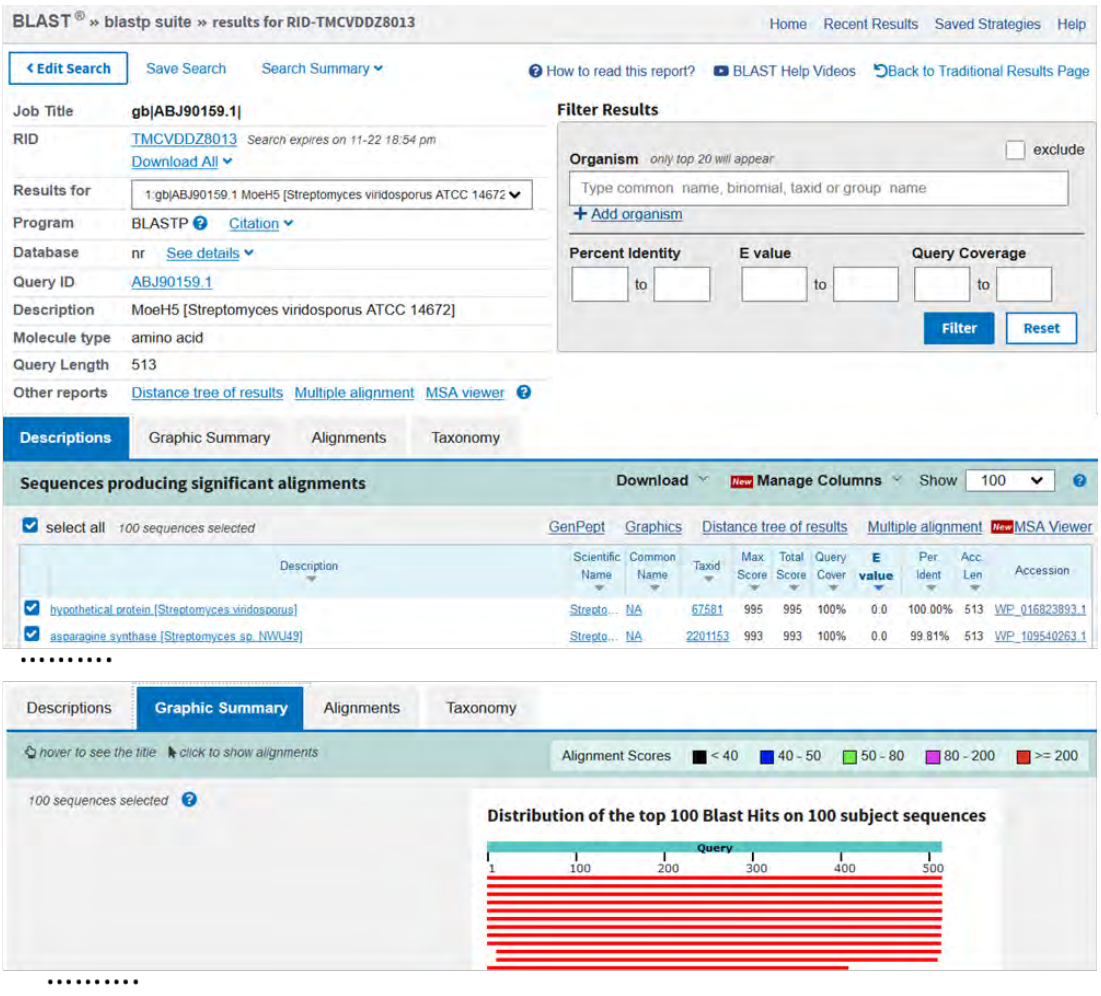


Рис. 3.42. Типова сторінка результатів BLASTP. Пунктирною лінією позначені однотипні елементи сторінки результатів, видалені задля спрощення. Окремі елементи сторінки (напр., закладка Alignments) розглянемо й опишемо згодом

BLAST, результат має вигляд інтерактивної інтернет-сторінки, на якій можна виділити дві основні частини: інформацію про вихідні параметри аналізу (задана послідовність, база даних, програма вирівнювання) та результати аналізу у вигляді чотирьох закладок. Перша з них – загальний опис знайдених послідовностей (Descriptions), друга – графічний підсумок вирівнювання (Graphical Summary). Третя закладка (рис. 3.43) – попарні вирівнювання разом із їхніми статистичними параметрами (Alignments). Четверта – таксономічний опис видів (Taxonomy), з яких походять знайдені послідовності (ця закладка активується при виборі окремих видів у закладці загального опису послідовностей).

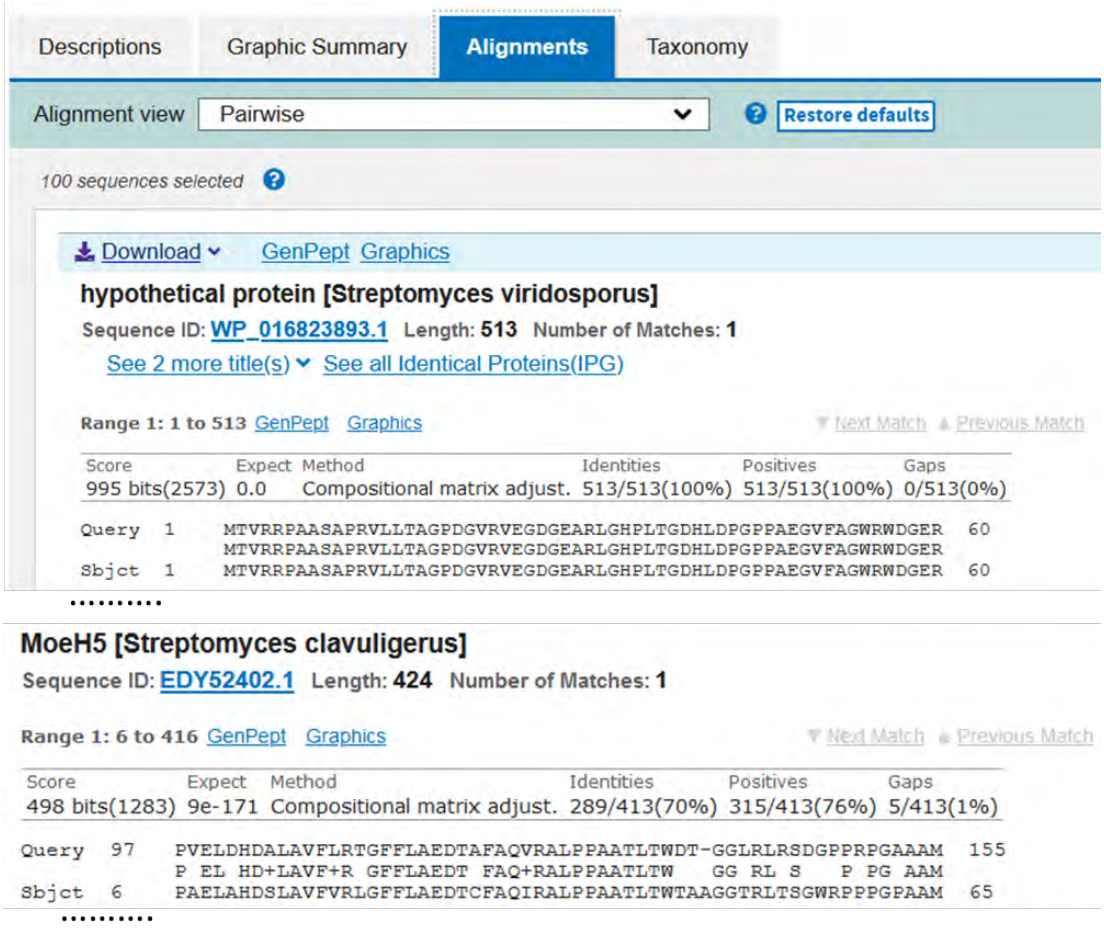
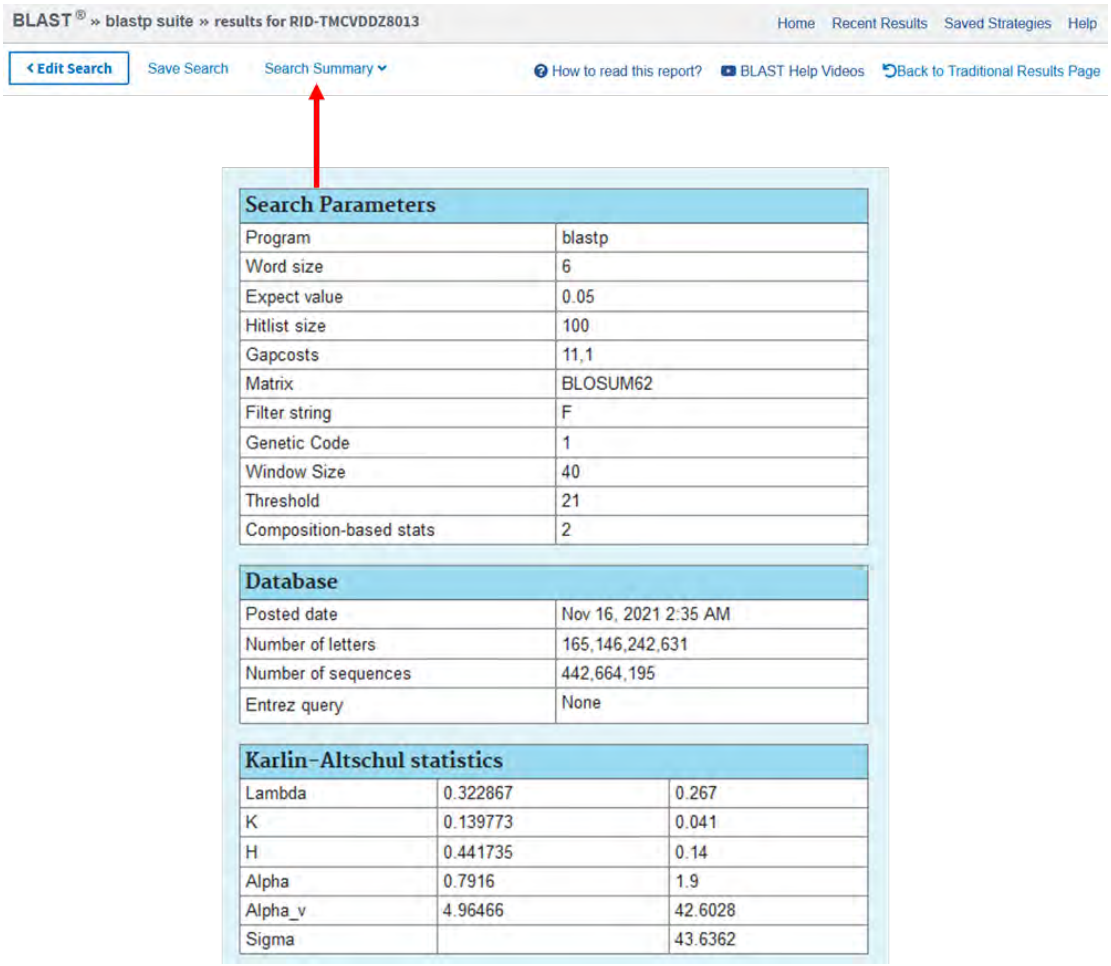


Рис. 3.43. Закладка Alignments на сторінці результатів BLASTP (пояснення – див.: осн. текст). Пунктирною лінією позначені однотипні елементи сторінки результатів, видалені задля спрощення. Кожне попарне вирівнювання супроводжується статистичними параметрами Score, Expect (E), рівнем ідентичності послідовностей (Identities), подібності (Positives), кількістю розривів (Gaps), що увела програма BLASTP у ході попарного вирівнювання

Головний елемент першої частини сторінки результатів – номер доступу до заданої послідовності в GenBank. Над ним розміщені декілька лінків, один з яких (Search Summary) дає змогу дізнатися усі параметри процесу попарного вирівнювання (*W, G, T, k, S, m, n,* λ тощо), та властивості бази даних, з якою порівнювали задану послідовність (рис. 3.44).



Search Parameters	
Program	blastp
Word size	6
Expect value	0.05
Hitlist size	100
Gapcosts	11,1
Matrix	BLOSUM62
Filter string	F
Genetic Code	1
Window Size	40
Threshold	21
Composition-based stats	2

Database	
Posted date	Nov 16, 2021 2:35 AM
Number of letters	165,146,242,631
Number of sequences	442,664,195
Entrez query	None

Karlin-Altschul statistics		
Lambda	0.322867	0.267
K	0.139773	0.041
H	0.441735	0.14
Alpha	0.7916	1.9
Alpha_v	4.96466	42.6028
Sigma		43.6362

Рис. 3.44. Розгорнутий вигляд закладки Search Summary у першій частині сторінки результатів BLAST. Подано основні параметри алгоритму пошуку, розмір бази даних, в якій шукали подібні послідовності, та значення констант (Karlin-Altschul statistics), які використовували для статистичного оцінювання попарних вирівнювань

Закладка загального опису містить перелік перших 100 найподібніших послідовностей, або *ximie* (див. рис. 3.42). Цей опис містить 11 типів інформації, які на рис. 3.45 позначено цифрами 1–11.

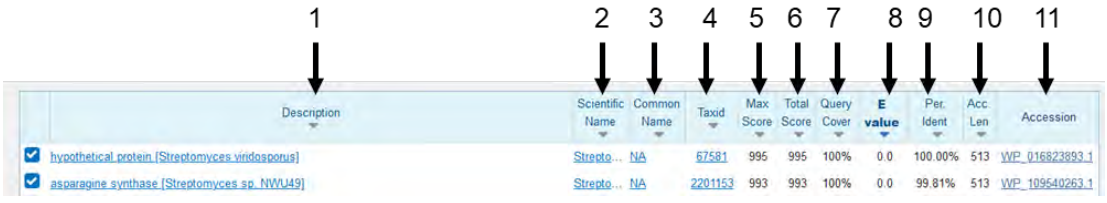


Рис. 3.45. Фрагмент сторінки результатів з рис. 3.43. Наведено перші дві знайдені в базі даних послідовності. Стрілками і цифрами 1–11 позначено усі дані, які програма BLASTP наводить для кожного попарного вирівнювання заданої послідовності зі знайденими. Ці рубрики даних описано далі в тексті

Опис 1 починається із назви знайденого білка (див. рис. 3.45). Рубрики 2 і 3 подають наукову та загальновживану (якщо така є) назви видів, з яких походить знайдена послідовність. Рубрика 4 – таксономічний ідентифікатор виду у базі Taxonomy NCBI (https://www.ncbi.nlm.nih.gov/taxonomy). Рубрика 5 – максимальний рахунок вирівнювання (max score) – найвищий рахунок вирівнювання (біт-рахунок S') між заданою послідовністю і послідовністю із бази даних. Рубрика 6 – сумарний рахунок (total score) – сума S' усіх сегментів, що походять з однієї послідовності і збігаються з заданою. Сумарний рахунок відрізнятиметься від максимального рахунку, якщо декілька сегментів однієї послідовності програма вирівняє з різними ділянками заданої послідовності. Рубрика 7 – покриття заданої послідовності (query cover) – відсоток заданої послідовності (кількість амінокислотних залишків), який увійшов до вирівнювань. Причому 100 % означає, що послідовність повністю (від першого до останнього амінокислотного залишку) вирівняно. Покриття обчислюють на основі всіх сегментів, що ввійшли до вирівнювання (див. сумарний рахунок). Рубрика 8 – величина очікування E. За цією величиною програма розміщує всі хіти у список, починаючи із послідовностей з найменшим E (найбільш значущі вирівнювання/хіти). Значення максимального рахунку та E обернено пропорційні. Рубрика 9 – максимальний відсоток ідентичних амінокислотних позицій у вирівнюванні виявленої (sbjct) і заданої (query) послідовностей (max ident). Рубрика 10 (Accession length) подає інформацію про розмір (тут – кількість амінокислотних залишків) знайденої послідовності. Рубрика 11 наводить номер доступу до знайденої послідовності у базі GenBank.

Друга закладка сторінки результатів – графічний підсумок пошуку (див. рис. 3.42). Ступінь подібності "хітів" до заданої послідовності можна визначити за кольоровим кодом, а саме: послідовності з рахунком вирівнювання $S \geq 200$ до заданої позначено червоним кольором, у межах 80÷200 – рожевим і т. д.

Кожне вирівнювання (див. рис. 3.43) супроводжується описом, в якому вказані назва виявленої послідовності, номер доступу до неї в GenBank, а також статистичні параметри. Зокрема, це значення ненормалізованого рахунку S (вказують у дужках поблизу біт-рахунка); число очікування E; кількість подібних амінокислотних залишків (positives) у вирівняній парі послідовностей (до таких належать ідентичні залишки і ті, що мають додатне значення відповідно до використаної матриці заміщення); кількість розривів у вирівняній парі послідовностей (gaps). Цифри з країв попарного вирівнювання в BLASTP – це номери амінокислотних залишків у білках.

Аналіз виконано за допомогою BLASTP (за замовчуванням), без використання фільтра, який маскує райони білка з низькою складністю. Цей фільтр може бути корисним під час аналізу білків з повторами чи послідовностей, до складу яких входить лише декілька амінокислот. Розглянемо, наприклад, серин–аргінін-багатий білок регулювання сплайсингу RZP23_ORYSJ, ген якого виявлений у геномі рису Oryza sativa. Наведемо FASTA-файл амінокислотної послідовності цього білка:

```
>gi|75324099|sp|Q6K9C3.1|RZP23_ORYSJ RecName: Full=Serine/arginine-rich
splicing factor RSZ23; AltName: Full=RS-containing zinc finger protein 23;
Short=Os-RSZ23; Short=Os-RSZp23
MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDGKNGWRVELST
KAGSGRGRDRSGGSDMKCYECGEPGHFARECRLRIGSGGLGSGRRRSRSRSRSPRYRGRSRSRSPRYRRS
PSYGRSPRDRSPKRRSYSRSPPPARARSYSRSPPPPRERSYSRSPAQPANREESPYANNA
```

Помітно, що С-кінцева ділянка білка, як і засвідчує назва, дуже збагачена на залишки аргініну і серину. Під час BLASTP-аналізу з параметрами за замовчуванням для цього білка вдається отримати низку хітів, серед яких білок OsJ_07486 цього ж культивару:

```
hypothetical protein OsJ_07486 [Oryza sativa Japonica Group]

 Score        Expect Method                  Identities    Positives     Gaps
 301 bits(770)  9e-101 Compositional matrix adjust.  200/203(99%)  200/203(98%)  3/203(1%)
Query  1    MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDG  60
            MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDG
Sbjct  1    MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDG  60

Query  61   KNGWRVELSTKAGSGRGRDRSGGSDMKCYECGEPGHFARECRLRIGSGGLGSGRRRSRSR 120
            KNGWRVELSTKAGSGRGRDRSGGSDMKCYECGEPGHFARECRLRIGSGGLGSGRRRSRSR
Sbjct  61   KNGWRVELSTKAGSGRGRDRSGGSDMKCYECGEPGHFARECRLRIGSGGLGSGRRRSRSR 120

Query  121  SRSPRYRGRSRSRSPRYRRSPSYGR---SPRDRSPKRRSYSRSPPPARARSYSRSPPPPR 177
            SRSPRYRGRSRSRSPRYRRSPSYGR   SPRDRSPKRRSYSRSPPPARARSYSRSPPPPR
Sbjct  121  SRSPRYRGRSRSRSPRYRRSPSYGRRSYSPRDRSPKRRSYSRSPPPARARSYSRSPPPPR 180

Query  178  ERSYSRSPAQPANREESPYANNA  200
            ERSYSRSPAQPANREESPYANNA
Sbjct  181  ERSYSRSPAQPANREESPYANNA  203
```

При увімкненні фільтра районів низької складності (на початковій сторінці BLAST) вирівнювання RZP23_ORYSJ і OsJ_07486 матимуть інший вигляд:

```
hypothetical protein OsJ_07486 [Oryza sativa Japonica Group]

 Score        Expect Method                  Identities    Positives     Gaps
 134 bits(337)  9e-36 Compositional matrix adjust.  101/101(100%)  101/101(100%)  0/101(0%)
Query  1    MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGfafidfddrrdaedairdldG  60
            MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDG
Sbjct  1    MARVYVGNLDPRVTAREIEDEFRVFGVLRSVWVARKPPGFAFIDFDDRRDAEDAIRDLDG  60

Query  61   KNGWRVELSTKAgsgrgrdrsggsdMKCYECGEPGHFAREC  101
            KNGWRVELSTKAGSGRGRDRSGGSDMKCYECGEPGHFAREC
Sbjct  61   KNGWRVELSTKAGSGRGRDRSGGSDMKCYECGEPGHFAREC  101
```

Останнє вирівнювання засвідчує, як функціонує фільтр ділянок низької складності: він відсік від заданої послідовності С-кінцеву частину, що містить повтори, а два ділянки з низькою складністю у межах N-кінцевої частини білка позначив малими літерами (виділено зеленим кольором).

Крім BLASTP, на сервері NCBI розміщено ще декілька програм, які використовують алгоритм BLAST:

BLASTN – порівнює задану нуклеотидну послідовність проти баз даних нуклеотидних послідовностей;

BLASTX – задану нуклеотидну послідовність транслює в усіх шести можливих рамках зчитування і порівнює продукти трансляції з базами даних амінокислотних послідовностей;

TBLASTN – задану амінокислотну послідовність порівнює з базою даних нуклеотидних послідовностей, трансльованою в усіх шести можливих рамках зчитування;

TBLASTX – задану нуклеотидну послідовність, трансльовану в усіх шести можливих рамках зчитування, порівнюють з базою даних нуклеотидних послідовностей, трансльованою в усіх шести можливих рамках зчитування. TBLASTX не може використовувати неповторювану базу даних амінокислотних послідовностей (nr), що є середовищем пошуку для BLASTP.

Принцип функціонування цих програм не відрізняється від BLASTP, а сторінка результатів відрізнятиметься окремими подробицями. Наприклад, якщо за допомогою програми BLASTX порівняти ген *moeH5* з кластера генів біосинтезу моеноміцину *S. ghanaensis* проти бази даних амінокислотних послідовностей, то одним із перших хітів буде білок MoeH5 *S. clavuligerus* , який також виявляє BLASTP (див. рис. 3.43). Порівнюючи результат BLASTP (див. рис. 3.43) і BLASTX (рис. 3.46), помітно, що координати заданої послідовності – це початок і кінець гена *moeH5* у межах більшого фрагмента ДНК, що було подано в GenBank, тоді як координати виявленої послідовності – це номери амінокислотних залишків білка MoeH5. Рядок статистичних параметрів містить додатковий показник – “Рамка” (Frame). Він позначує одну з шести можливих рамок зчитування (+1, +2, +3, -1, -2, -3), в якій отримано амінокислотну послідовність (query), вирівняну до виявленої (sbjct).

Розроблено також низку більш спеціалізованих BLAST-програм, які дають змогу підібрати праймери для ПЛР, виявити залишки послідовностей векторів клонування у поданих нуклеотидних даних тощо. З них варто згадати програму bl2seq, яка дає змогу вирівняти дві чи більше заданих НАП, а також програму для глобального вирівнювання на основі АНВ. Усі програми пакета BLAST доступні за адресою https://blast.ncbi.nlm.nih.gov/Blast.cgi .

Програми з пакета BLAST є своєрідним “золотим стандартом” якості попарного вирівнювання, сьогодні вони є найпопулярнішим біоінформатичним знаряддям. Втім зростання розміру баз генетичних послідовностей стає викликом для BLAST у сенсі обчислювального часу для виконання завдань. Наприклад, у нещодавньому метагеномному дослідженні мікробіоти, що присутня у зразках пермафросту (льоду, добутого із зони вічної мерзлоти), BLASTX-опосередковане порівняння 176 мільйонів якісно просеквенованих фрагментів ДНК проти бази KEGG потребувало 800 тисяч CPU-годин роботи суперкомп’ютера (https://www.nature.com/articles/ ismej201793). Обсяги даних, аналогічні до згаданих вище, стають типовими у біологічних дослідженнях, що спонукало до впровадження алгоритмів попарного вирівнювання, які б функціонували набагато швидше, ніж BLAST, без суттєвої втрати точності. Такі алгоритми сьогодні існують, більше про них можна дізнатися нижче (блок 8).

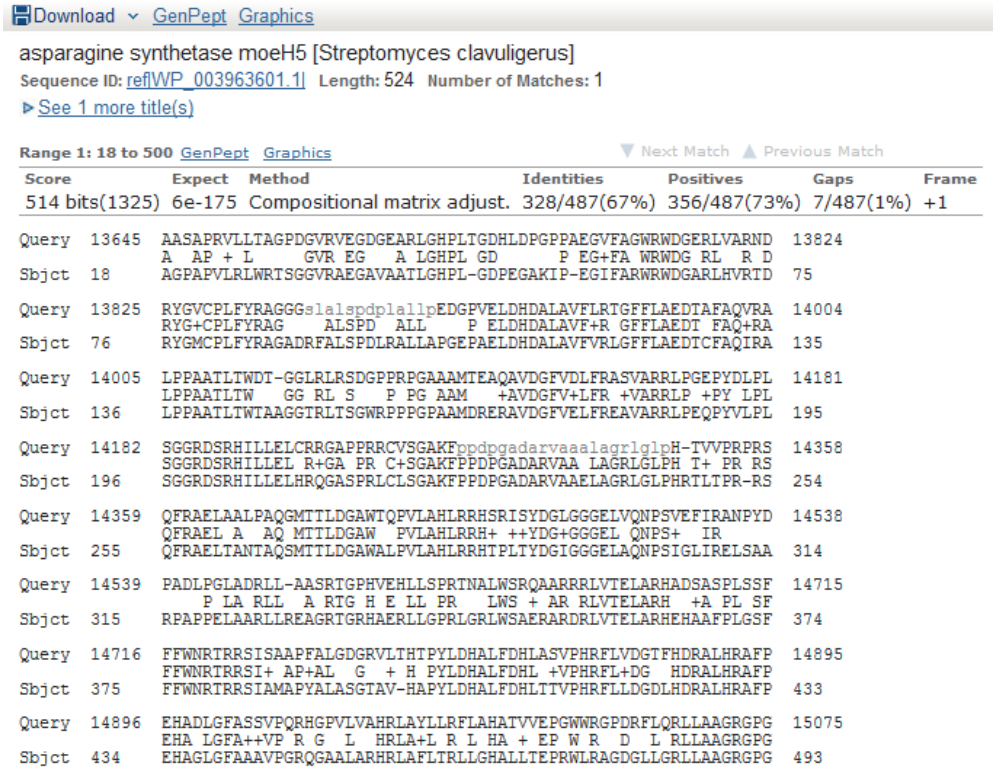
БЛОК 8. Швидше і точніше: нові алгоритми попарного вирівнювання

Базовий алгоритм BLAST полягає в генеруванні набору засівних слів певного розміру, див. с. 132 цієї книги. Далі цим набором слів сканують певну базу даних з метою виявлення збігів. Тривалість такого сканування (аналізу бази) зростає лінійно з розміром бази. Алгоритм BLAT (https://kentinformatics.com/#BLAT) відізняється від BLAST тим, що індексуванню підлягає база даних чи цільовий геном, і далі утвореним набором слів сканують задану послідовність. Для нуклеотидних послідовностей розмір слів становить 8–16 нуклеотидів, для амінокислотних – 3–7 амінокислотних залишків. Такий обернений підхід до попарного вирівнювання має зміст у випадку, коли предметом дослідження є великий геном, з яким постійно порівнюють щоразу нові набори даних. Типовий приклад такого завдання – референтний геном людини, проти якого можуть вирівнювати індивідуальні геном людей чи геноми інших хребетних. Алгоритм BLAT є в основі переглядача (браузера) генома людини Університету Каліфорнії Санта Круз (http://genome.ucsc.edu/). Порівняно з BLAST, використання BLAT дає змогу вирівнювати нуклеотидні послідовності у 500 разів швидше, амінокислотні – у 50 разів швидше без суттєвої втрати точності аналізу. Оскільки геном людини був вихідним мотивом створення BLAT, то в ньому є також особливості щодо виявлення екзонів, які роблять цей алгоритм особливо зручним для аналізу геномів хребетних тварин. Більше про BLAT можна дізнатися тут: https://genome.cshlp.org/content/12/4/656.long.

Алгоритм USEARCH (https://www.drive5.com/usearch/manual/usearch_algo.html) шукає лише один або кілька хітів у базі даних для заданої послідовності. Це пришвидшує аналіз порівняно з BLAST, який генерує великий список хітів. Індексуванню підлягає як задана послідовність, так і база даних. Базовим припущенням, на якому ґрунтується алгоритм, є те, що в базі даних є високоподібна послідовність, яка збігатиметься із заданою зразу за декількома словами. Такі послідовності відбирають для подальшого аналізу як потенційні кандидати на хіт. Для білків і ДНК цей алгоритм добре працює, якщо в базі даних є послідовності на 50 % і 75 % ідентичні щодо заданої.

Алгоритм RAPSearch (https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-12-159) створений для виявлення подібних амінокислотних послідовностей на основі скороченої амінокислотної абетки. Тобто 20 природних амінокислот згрупували у 10 груп за подібністю фізико-хімічних властивостей. Наприклад Lys та Arg утворили одну групу, яку позначили одним символом. Відтак швидкість виявлення і довжина збігів (засівних слів) між заданою послідовністю та базою даних значно зросли, що в сумі дало змогу пришвидшити аналіз майже на два порядки. Спрощену амінокислотну абетку використовують лише на етапі первинного виявлення збігів між базою та заданою послідовністю; відібрані кандидати далі аналізують як у програмі BLAST на основі звичайної (20-літерної) амінокислотної абетки.

Алгоритм DIAMOND (https://www.wsi.uni-tuebingen.de/lehrstuehle/algorithms-in-bioinformatics/software/diamond/) ґрунтується на стратегії подвійного індексування. Її зміст можна зрозуміти за порівняння з BLAST. Ця програма утворює набір засівних слів на основі заданої послідовності і далі індексує розташування ідентичних слів (чи сусідніх слів, див. с. 134) у послідовностях із бази даних. Тоді засівні слова заданої послідовності у випадковому порядку співставляють зі словами у базі. DIAMOND індексує засівні слова як у базі даних, так і в заданій послідовності. Тоді два списки сортують лексикографічно й аналізують разом для виявлення як збігів, так і їхнього розташування у послідовностях (про лексикографічне сортування див. https://rosalind.info/problems/lexf/). Засівні слова – 15–24 літери завдовжки, частина позицій може бути будь-яка. Для амінокислотних послідовностей використовують спрощену амінокислотну абетку на основі 11 символів, подібно до RAPSearch. Такий підхід сприяє пришвидшенню аналізу у 20 тисяч разів.



Download GenPept Graphics

asparagine synthetase moeH5 [Streptomyces clavuligerus]

Sequence ID: ref|WP_003963601.1| Length: 524 Number of Matches: 1

See 1 more title(s)

Range 1: 18 to 500 GenPept Graphics Next Match Previous Match

Score	Expect	Method	Identities	Positives	Gaps	Frame
514 bits(1325)	6e-175	Compositional matrix adjust.	328/487(67%)	356/487(73%)	7/487(1%)	+1

```
Query  13645  AASAPRVLLTAGPDGVRVEGDGEARLGHPLTGDHLDPGPPAEGVFAGWRWDGERLVARND  13824
              A  AP + L      GVR EG    A LGHPL GD       P EG+FA WRWDG RL   R D
Sbjct  18     AGPAPVLRLWRTSGGVRAEGAVAATLGHPL-GDPEGAKIP-EGIFARWRWDGARLHVRTD  75

Query  13825  RYGVCPLFYRAGGGslalspdplallpEDGPVELDHDALAVFLRTGFFLAEDTAFAQVRA  14004
              RYG+CPLFYRAG     ALSPD  ALL     P ELDHDALAVF+R GFFLAEDT FAQ+RA
Sbjct  76     RYGMCPLFYRAGADRFALSPDLRALLAPGEPAELDHDALAVFVRLGFFLAEDTCFAQIRA  135

Query  14005  LPPAATLTWDT-GGLRLRSDGPPRPGAAAMTEAQAVDGFVDLFRASVARRLPGEPYDLPL  14181
              LPPAATLTW   GG RL S   P PG AAM   +AVDGFV+LFR +VARRLP +PY LPL
Sbjct  136    LPPAATLTWIAAGGTRLTSGWRPPPGPAAMDRERAVDGFVELFREAVARRLPEQPYVLPL  195

Query  14182  SGGRDSRHILLELCRRGAPPRRCVSGAKFppdpgadarvaaalagrlglpH-TVVPRPRS  14358
              SGGRDSRHILLEL R+GA PR C+SGAKFPPDPGADARVAA LAGRLGLPH T+ PR RS
Sbjct  196    SGGRDSRHILLELHRQGASPRLCLSGAKFPPDPGADARVAAELAGRLGLPHRTLTPR-RS  254

Query  14359  QFRAELAALPAQGMTTLDGAWTQPVLAHLRRHSRISYDGLGGGELVQNPSVEFIRANPYD  14538
              QFRAEL A  AQ MTTLDGAW  PVLAHLRRH+ ++YDG+GGGEL QNPS+  IR
Sbjct  255    QFRAELTANTAQSMTTLDGAWALPVLAHLRRHTPLTYDGIGGGELAQNPSIGLIRELSAA  314

Query  14539  PADLPGLADRLL-AASRTGPHVEHLLSPRTNALWSRQAARRRLVTELARHADSASPLSSF  14715
                  P LA RLL   A RTG H E LL PR   LWS + AR RLVTELARH  +A PL SF
Sbjct  315    RPAPPELAARLLREAGRTGRHAERLLGPRLGRLWSAERARDRLVTELARHEHAAFPLGSF  374

Query  14716  FFWNRTRRSISAAPFALGDGRVLTHTPYLDHALFDHLASVPHRFLVDGTFHDRALHRAFP  14895
              FFWNRTRRSI+ AP+AL  G  + H PYLDHALFDHL +VPHRFL+DG  HDRALHRAFP
Sbjct  375    FFWNRTRRSIAMAPYALASGTAV-HAPYLDHALFDHLTTVPHRFLLDGDLHDRALHRAFP  433

Query  14896  EHADLGFASSVPQRHGPVLVAHRLAYLLRFLAHATVVEPGWWRGPDRFLQRLLAAGRGPG  15075
              EHA LGFA++VP R G  L  HRLA+L R L HA + EP W R  D  L RLLAAGRGPG
Sbjct  434    EHAGLGFAAAVPGRQGAALARHRLAFLTRLLGHALLTEPRWLRAGDGLLGRLLAAGRGPG  493
```

Рис. 3.46. Елемент сторінки результатів програми BLASTX–опосередкованого вирівнювання продукту трансляції гена *moeH5 S. ghanaensis* у рамці +1 з білком MoeH5 *S. clavuligerus* (задана послідовність пронумерована в нуклеотидних, виявлена – в амінокислотних залишках)

Типові задачі до розділу 3

Задача 3.1. Програма blastn2.0 для попарного вирівнювання ДНК використовує таку схему оцінювання вирівнювань: +1 за збіг; – 3 за незбіг; λ = 1,374. На аналіз яких послідовностей налаштована програма?

Розв'язання. Будь-яку систему рахунків можна записати у формі рівняння (27) (див. попередній текст):

$$\lambda s_{ij} = \log(\frac{q_{ij}}{f_i f_j}).$$

Частоти нуклеотидів у задачі не наведено, тому припускаємо найпростіший сценарій – однакове натрапляння (25 %, або 0,25). За рівнянням можна виразити цільову частоту:

$$q_{ij} = f_i f_j e^{\lambda s}.$$

Цільова частота збігу будь-яких нуклеотидів (з урахуванням натурального числа $e \approx 2{,}718$):

$$q_{AA} = q_{TT} = q_{GG} = q_{CC} = (0{,}25 \times 0{,}25) \times 2{,}718^{1{,}347\times(+1)} = 0{,}240.$$

Тоді *загальна цільова ідентичність* (ідентичність, на яку налаштована система рахунків) послідовності становитиме:

$$\sum\nolimits_{b=A,G,C,T} p_{b,b} = 0{,}960.$$

Отже, blastn2.0 оптимально ефективна для виявлення гомологічних послідовностей, що є на 96 % ідентичними.

Задача 3.2. Обчисліть рахунки збігу та незбігу для програми попарного вирівнювання нуклеотидних послідовностей, яку оптимізовано для послідовностей, що мають 64 % ідентичності (λ = 0,1915).

Розв'язання. В умові задачі не наведено частот нуклеотидів, тому припускаємо найпростіший варіант – рівноймовірність усіх нуклеотидів (1/4, або 25 %; 0,25). Якщо програму оптимізовано для послідовностей, що мають 64 % ідентичності, то цільова частота збігу для будь-яких двох нуклеотидів буде 0,16 (0,64/4 можливі пари ідентичних нуклеотидів). Фонові частоти – 0,25. Отже, на підставі рівняння (27) (див.: попередню задачу):

$$0{,}1915 s_{ij} = \log\left(\frac{0{,}16}{0{,}25 \times 0{,}25}\right) = \log(2{,}56) = 0{,}940;$$

$$0{,}1915 s_{ij} = 0{,}940;\ s_{ij} = 4{,}908 \cong +5.$$

На незбіги припадає 36 % (100 % – 64 %). Загалом у матриці заміщень розміром 4×4 можливі 12 варіантів незбігів (див. підрозділ 2.4 про моделі еволюції), унаслідок чого на цільову частоту будь-якого з них припадатиме 3 %, або 0,03 (36 %/12 = 3 %). Звідси обчислюємо рахунок незбігів:

$$0{,}1915 s_{ij} = \log\left(\frac{0{,}03}{0{,}25 \times 0{,}25}\right) = \log(0{,}48) = -0{,}7339;$$

$$0{,}1915 s_{ij} = -0{,}7339;\ s_{ij} = -3{,}83 \cong -4.$$

Отже, за заданого значення λ програма, оптимізована для послідовностей з 64 % ідентичності, приписуватиме збігові рахунок +5, а незбігові –4.

Задача 3.3. Порівнюють дві амінокислотні послідовності завдовжки 400 ак. Яке очікуване число випадкових вирівнювань послідовностей такого ж розміру можна отримати з біт-рахунком 10? Який мав би бути мінімальний біт-рахунок S' для такої пари послідовностей, щоб гарантувати відсутність випадкових вирівнювань з рахунком, рівним або більшим S'?

Розв'язання. Для відповіді на перше питання можна скористатися формулою (31):

$$E = mn2^{-S'} = 400 \times 400 \times 2^{-10} \approx 156.$$

Тоді при вирівнюванні двох послідовностей розміром 400 ак на сегменти з біт-рахунком не більше 10 можна очікувати натрапити принаймні 156 разів. Очікуване число E, значно менше одиниці (0÷0,05), може гарантувати відсутність випадкових вирівнювань з рахунком, рівним або більшим S'. Обчислимо S', за якого E становитиме, наприклад, 0,05. Для цього виразимо S' з рівняння:

$$S' = \log 2\left(\frac{mn}{E}\right);$$

тоді

$$S' = \log 2\left(\frac{400 \times 400}{0{,}05}\right) = \log 6\,400\,000 \cong 16.$$

Отже, якщо дві послідовності заданого розміру сформують вирівнювання з біт-рахунком принаймні 16, то таке вирівнювання вже не буде випадковим (E = 0,05).

Задача 3.4. Гомолог заданої послідовності міститься у двох базах даних – UniProt та PDB. Після виконання BLAST-пошуку бази UniProt вирівнювання заданої послідовності з гомологом отримало значення E = 1. Таке саме вирівнювання при BLAST-аналізі бази PDB мало значення E = 0,0625. Яке співвідношення розмірів цих баз даних?

Розв'язання. З умови задачі випливає, що для аналізу баз використали одну програму з однаковими параметрами. Тому різні числа очікування можуть бути лише наслідком різного розміру баз даних. Розміри баз UniProt і PDB співвідносяться як 1:0,0625. База UniProt – більша за розміром, тому в ній можна знайти одну випадкову послідовність, що вирівнюватиметься з певним рахунком S із заданою послідовністю. База PDB містить меншу кількість послідовностей. Тому вирівнювання заданої послідовності із знайденою в базі у результаті буде невипадковим (E < 1) відносно PDB, хоча матиме такий самий рахунок S.

Задача 3.5. Використовуючи матрицю BLOSUM62 (див. рис. 3.11) і таку систему афінних штрафів – G = -11, l = -1, обчисліть рахунки двох вирівнювань:

а) α-глобіну людини і леггемоглобіну жовтого люпину:

```
HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHV---D--DMPNALSALSDLHAHK
           ++ ++++H+ KV   + +A  ++          +L +L+++H+ K
LGB2_LUPLU NNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSK;
```

б) той самий район α-глобіну людини і фрагмент глутатіон-S-трансферази нематоди, позначену як F11G11.2:

```
HBA_HUMAN  GSAQVKGHGKKVADALTNAV    DMPNALSALSD----LHAHK
           GS+ + G +   +D L  ++ H+ D+  A +AL D    ++AH+
F11G11.2   GSGYLVGDSLTFVDLL--VAQHTADLLAANAALLDEFPQFKAHQ.
```

Розв’язання. Рахунок вирівнювання – це сума рахунків заміщення вирівняних амінокислотних залишків і штрафів за розриви, що визначають за формулою γ = -11-(g-1), де g – довжина розриву. Для першого вирівнювання обчислюємо:

$$\begin{aligned} S_1 = s(G,N) + s(S,N) + \cdots + s(K,K) \\ = 0 + 1 - 1 + 2 + 1 + 1 + 0 + 8 + 0 - 2 + 5 + 4 - 2 - 1 - 1 - 1 \\ - 2 + 0 + 4 + 0 - 1 + 0 - 1 + 0 + 1 - 13(gap) - 1 - 12(gap) \\ - 3 - 1 - 1 - 2 + 0 + 4 + 0 - 2 + 4 + 0 + 0 + 1 + 8 + 0 - 1 + 5 \\ = \mathbf{1}. \end{aligned}$$

Друге вирівнювання:

$$\begin{aligned} S_2 = s(G,G) + s(G,G) + \cdots + s(K,Q) \\ = 6 + 4 + 0 - 1 + 1 - 2 + 6 - 1 + 0 - 2 - 1 - 1 + 0 + 6 - 1 + 4 \\ - 12(gap) + 0 + 0 - 1 + 8 + 0 - 2 + 6 + 2 - 3 - 2 + 4 - 3 + 1 \\ + 4 + 4 - 2 + 6 - 14(gap) + 0 - 1 + 4 + 8 + 1 = \mathbf{26}. \end{aligned}$$

Цікаво, що структурно й еволюційно вірогідне вирівнювання α-глобіну людини і леггемоглобіну (вирівнювання 1) отримує нижчий рахунок, ніж вирівнювання α-глобіну людини і неспорідненого білка нематоди (вирівнювання 2). Цей приклад засвідчує важливість фахового оцінення висновків про гомологію на основі даних комп’ютерного аналізу.

Задача 3.6. Дослідника цікавить, чи ймовірна сульфуртрансфераза Sco5854 (283 ак залишки) Streptomyces coelicolor має гомолога у Mycobacterium tuberculosis. Користуючись програмою BLASTP, дослідник виявив, що у базі GenBank (сумарний розмір бази 41 144 954 208 ак залишків на момент пошуку) є амінокислотна послідовність SseA, що походить із M. tuberculosis, яка вирівнюється із заданою (Sco5854) й має такі параметри попарного вирівнювання: S′ = 279; E = 8e-93 (тобто 8×10-93); identitites = 57 %, positives = = 66 %. Який із розглянутих вище параметрів слугує найпереконливішим доказом гомологічності Sco5854 та SseA?

Відповідь. З усіх розглянутих параметрів вирівнювання найважливішим є статистичний параметр – число очікування *E*. Воно дає відповідь на питання про те, чи знайдена у результаті попарного вирівнювання послідовність є насправді випадковою (негомологічною). У цьому разі *E* засвідчує, що у використаній базі даних розміром ~41,1 млрд ак залишків можливо знайти 8e-93 (або 8×10^{-93}) випадкових (= негомологічних) послідовностей, які вирівняються із заданою послідовністю та матимуть біт-рахунок не менше 279. Число *E* – це власне кількість випадкових послідовностей – не ймовірність їхньої появи. Отож *E* мало б набувати лише цілих значень – 1, 2, 3 тощо. Насправді тут число *E* значно менше одиниці – отже, у базі даних немає жодної випадкової послідовності, яка б вирівнялася із Sc05854 і водночас мала $S' \geq 279$. Фактично базу даних треба збільшити в астрономічну (10^{92}) кількість разів (див. рівняння (31)), щоб натрапити там на одну випадкову послідовність, яка б, за умови вирівнювання з Sco5854 з біт-рахунком 279, мала $E = 1$. Отже, SseA є не випадковою послідовністю, а гомологом Sco5854. Спільність еволюційного походження дає змогу припустити спільність (подібність) функцій цих білків.

Контрольні запитання до розділу 3

1. Розтлумачте основні терміни попарного вирівнювання: позиція вирівнювання; задана і знайдена послідовності; позиції збігу і незбігу; розриви.
2. Дайте визначення попарного вирівнювання.
3. На які питання дослідник шукає відповідь, виконуючи попарне вирівнювання послідовностей?
4. У чому полягає принцип штрафів у попарних вирівнюваннях?
5. Що відображає процес уведення розривів у попарне вирівнювання?
6. Що є підставою для віри в те, що попарне вирівнювання відображає біологічну дійсність?
7. Чому необхідно обмежувати кількість розривів у попарних вирівнюваннях?
8. Які дві послідовності можна назвати гомологічними?
9. Що таке локальне вирівнювання?
10. Що таке глобальне вирівнювання?
11. Що таке підпослідовність?
12. Що таке афінні штрафи?
13. Поясніть принцип дотплот-аналізу.
14. Що таке вікно та жорсткість дотплот-аналізу?
15. Чому унітарні (параметризовані) підходи до оцінювання попарних вирівнювань малопридатні для аналізу амінокислотних послідовностей?
16. У чому полягає суть змісту теорії точкових прийнятних мутацій М. Дейгоф?
17. Що таке еволюційна відстань 1 PAM?
18. Поясніть зміст термінів: фонові частоти амінокислотних залишків; цільові частоти заміщення амінокислотних залишків.
19. Як обчислюють PAM-рахунок заміщення одного амінокислотного залишку на інший?
20. Що означають PAM-рахунки: більше одиниці; менше одиниці?
21. Скільки заміщень відбулося на відстані 250 PAM?
22. На основі якого масиву амінокислотних послідовностей М. Дейгоф будувала PAM-матриці?
23. Що таке “мілкі” та “глибокі” матриці заміщення?
24. Опишіть недоліки PAM-матриць.
25. Що таке база BLOCKS?
26. Принцип побудови BLOSUM-матриць.
27. Потрактуйте такі терміни: подібність послідовностей; ідентичність послідовностей; гомологічність послідовностей.
28. Які амінокислотні заміщення можна вважати консервативними, а які – неконсервативними?
29. Що постулює теорема Альтшуля?
30. Чому різні матриці амінокислотних заміщень мають різний інформаційний зміст? Що роблять, щоб уніфікувати рахунки вирівнювання на основі різних підходів?
31. Алгоритми динамічного програмування.
32. Етапи заповнення матриць попарного вирівнювання в алгоритмах динамічного програмування.
33. Які є евристичні алгоритми попарного вирівнювання?
34. Принцип роботи алгоритму FastA.

35. Принцип роботи алгоритму BLAST.
36. Які елементи евристики використані в алгоритмі BLAST?
37. Що таке “слова сусідства” і “поріг відсікання” в алгоритмі BLAST?
38. Етапи попарного вирівнювання за алгоритмом BLAST.
39. Які є підходи до статистичної оцінки попарних вирівнювань?
40. В чому полягає зміст теорії Карліна-Альтшуля?
41. Що таке число очікування *E*?
42. Який біологічний зміст поправок (констант) λ та *K* у рівнянні Карліна-Альтшуля?
43. Які є основні елементи сторінки уведення даних веб-сервера BLAST?
44. За допомогою BLASTP-аналізу, з використанням параметрів за замовчуванням, вирівняно амінокислотні послідовності бета-субодиниці РНК-полімерази з архебактерії *Pyrococcus furiosus* та *Homo sapiens*. Програма не виявила суттєвої подібності між цими послідовностями. Яка може бути причина? Які зміни у налаштуваннях програми BLASTP можуть привести до виявлення подібності між аналізованими послідовностями?
45. Які є основні елементи сторінки результатів BLAST?
46. Перерахуйте основні різновиди програм родини BLAST, яке їхнє призначення?
47. Кожен результат попарного вирівнювання послідовностей у програмах BLAST супроводжується низкою параметрів: біт-рахунок, число *E*, відсоток подібності, відсоток ідентичності, кількість розривів. Котрий з них найважливіший для виявлення гомологічності двох аналізованих послідовностей? Чому?
48. Якщо число *E* для пари вирівняних амінокислотних послідовностей становить 0,0001, то що може дослідник припустити про ці дві послідовності? Чому?
49. Якщо число *E* для пари вирівняних амінокислотних послідовностей становить 0,5, то який висновок може дослідник зробити про ці дві послідовності? Чому?
50. Якщо число *E* для пари вирівняних амінокислотних послідовностей становить 2,1, то який висновок може дослідник зробити про ці дві послідовності? Чому?
51. Що таке маскування ділянок зі зниженою складністю у попарних вирівнюваннях?
52. Які обмеження притаманні методам попарного вирівнювання як підходу до виявлення гомології білків?

РОЗДІЛ 4

Множинне вирівнювання послідовностей

Якщо за допомогою попарного вирівнювання виявлено більше однієї послідовності, подібної до заданої, то черговим важливим кроком стає порівняння всієї множини (>2) гомологічних послідовностей – *множинне вирівнювання*. Таке вирівнювання дає змогу отримати ключову вихідну інформацію про вторинну структуру й еволюцію послідовностей, функцію окремих нуклеотидних чи амінокислотних залишків. Множинні вирівнювання основоположні в найчутливіших методах пошуку гомологічних послідовностей, про що докладніше опишемо далі. Хоча множинне вирівнювання можна застосовувати як до нуклеїнових кислот, так і до білків, тут, здебільшого, увагу звертатимемо на вирівнювання білків. Саме для їхнього аналізу розроблено найпотужніші та найінформативніші біоінформатичні методи, що ґрунтуються на множинному вирівнюванні.

Множинні вирівнювання можна виконувати двома способами. Суть першого підходу полягає в порівнянні первинної послідовності амінокислотних залишків, аналогічно до попарного вирівнювання. Другий підхід заснований на порівнянні вторинних і третинних структур генетичних послідовностей (структурне вирівнювання). Вирівнювання амінокислотних послідовностей за допомогою цих двох різних підходів часто має значні відмінності, спонукаючи до роздумів про те, яке ж із них коректне. Загалом структурне вирівнювання двох і більше білків вважають “золотим стандартом”, з яким порівнюють алгоритми множинного вирівнювання первинних послідовностей. Структурне вирівнювання дає змогу найточніше знаходити функціонально еквівалентні амінокислотні залишки з урахуванням усіх особливостей згортання білка і посттрансляційних модифікацій. Наприклад, розглянемо два білки: пептидоглікан-глікозилтрансферазний домен білка Pbp1A гіпертермофільної бактерії *Aquifex aeolicus* та глікозидазний домен лізоциму фага λ (λR). Визначену просторову структуру (синонім – кристалічну будову) обидвох білків і тривимірні координати всіх атомів, з яких вони складаються, можна отримати з бази даних Protein Data Bank (PDB). Первинні амінокислотні послідовності білків, які кристалізували, можна отримати через NCBI за номерами доступу в PDB – pdb2OQO (Pbp1A) і pdb1D9U (лізоцим):

```
>gi|145580129|pdb|2OQO|A Chain A, Pbp1a Aquifex aeolicus
GPGYQDPKGRLYGTIGIQKRFYVSIDKIPEHVINAFVATEDRNFWHHFGIDPVAIVRAAIVN
YRAGRIVQGGSTITQQLAKNLFLTRERTLERKIKEALLAIKIERTFDKKKIMELYLNQIYLG
SGAYGVEAAAQVYFGKHVWELSLDEAALLAALPKAPAKYNPFYHPERALQRRNLVLKRMLEE
GYITPEQYEEAVNK

>gi|14488738|pdb|1D9U|B Chain B, Bacteriophage Lambda Lysozyme
MVEINNQRKAFLDMLAWSEGTDNGRQKTRNHGYDVIVGGELFTDYSDHPRKLVTLNPKLKST
GAGRYQLLSRWWDAYRKQLGLKDFSPKSQDAVALQQIKERGALPMIDRGDIRQAIDRCSNIW
ASLPGAGYGQFEHKADSLIAKFKEAGGTVR
```

Спроба попарного вирівнювання цих послідовностей за допомогою програми bl2seq (див. вище у тексті) з параметрами за замовчуванням (матриця BLOSUM62) не дає жодного результату. Якщо застосувати для аналізу матрицю для вирівнювання віддаленіших послідовностей (BLOSUM45), то програмі вдається знайти невеликий спільний сегмент для заданих білків (24-х амінокислот з понад 150-ти) з високим значенням *E*:

Score	Expect	Method	Identities	Positives	Gaps
15.2 bits(36)	0.50	Compositional matrix adjust.	8/24(33%)	12/24(50%)	2/24(8%)

```
Query  150  AALLAALPKAPAKYNPFYHPERAL  173
            + + A+LP A   Y  F H   +L
Sbjct  121  SNIWASLPGAG--YGQFEHKADSL  142
```

Отже, методи вирівнювання первинних послідовностей засвідчують неподібність між двома досліджуваними білками. Водночас за структурного порівняння очевидно, що пептидоглікан-глікозилтрансферазний і глікозидазний домени мають низку ділянок з майже однаковою просторовою будовою (рис. 4.1). У ретроспективі отриманий результат не видається абсолютно неочікуваним, ураховуючи, що ці білки фактично каталізують зворотні реакції на одному субстраті, а саме: Pbp1A полімеризує дисахаридні залишки з утворенням пептидоглікану (основний елемент клітинної стінки бактерій), а λR – розщеплює глікозидні зв’язки у пептидоглікані. Однак тільки порівняння структур двох вищеописаних білків дало змогу виявити подібність між ними!

Структурні множинні вирівнювання можливі тільки тоді, коли встановлено тривимірні структури всіх білків, які вирівнюють, а це наразі поодинокі випадки. Експоненційний ріст геномних баз даних спричиняє накопичення величезного масиву інформації про первинні амінокислотні послідовності, отож очевидно, що ці дані кількісно переважатимуть розмір баз даних про просторову будову білків. Тому завданням біоінформатики стає розроблення алгоритмів і методів, що за браком інформації про просторову будову білків продукують вирівнювання первинних амінокислотних послідовностей, які за точністю наближаються до структурних вирівнювань.

Хоча структурні вирівнювання є найважливішим інструментом передбачення функції, вони не обов’язково тотожні з еволюційним вирівнюванням, тобто не дають змоги реконструювати процес розходження (*дивергенції*) амінокислотних послідовностей від спільного предкового білка. Еволюційне вирівнювання потребувало б інформації про точну послідовність заміщень, інсерцій та делецій, що привели від вихідної форми до сучасних білків, спільність моделі еволюції досліджуваних білків.

Рис. 4.1. Накладання тривимірних структур пептидоглікан-глікозилтрансферазного домену білка Pbp1A *Aquifex aeolicus* і глікозидазного домену лізоциму фага λ. Чотири α-спіралі білків, що мають однакове просторове розміщення, зображені у вигляді кольорових стрічок (Pbp1A – світлішого кольору). Складчасті структури, що сполучають спіралі α5 і α6 Pbp1A, показані зеленим кольором. У нитковому форматі зображено райони білків, що не накладаються. Glu-19 в λR і Glu-83 в Pbp1A (активний центр білків) подані у вигляді атомної моделі (червоні кульки – кисень, зелені – вуглець; джерело рисунка – стаття з PubMed-ідентифікатором PMID17360321)

4.1. Множинне вирівнювання: основні терміни і концепції. Множинне вирівнювання – це двовимірна таблиця, в якій ряди репрезентують окремі послідовності, а колонки – позиції вирівнювання. Окремі послідовності укладаються в таблицю так, що: а) відносні позиції амінокислотних/нуклеотидних залишків у межах кожної послідовності зберігаються; б) подібні залишки в усіх послідовностях (рис. 4.2) займатимуть одну колонку (синонім – вертикальний регістр). Позиція амінокислотного/нуклеотидного залишку у невирівняній послідовності матиме назву “абсолютна позиція”. Наприклад, якщо залишок 3 у послідовності Z1 – гліцин (G), то абсолютна позиція I3 завжди буде G. На противагу цьому, позиція залишку у вирівняній послідовності матиме назву “відносна позиція”.

Наприклад, у позиції 8 послідовності Z1 множинного вирівнювання на рис. 4.2 розміщений залишок глутамінової кислоти (E), однак він потрапив у 8-му позицію внаслідок уведення розриву в послідовність Z1, тому його абсолютна позиція – 7. Отже, відносна позиція залишків – це властивість вирівнювання, тоді як абсолютна позиція – властивість окремої послідовності.

Z1	Y	D	G	G	A	V	-	E	A	L
Z2	Y	D	G	G	-	-	-	E	A	L
Z3	F	E	G	G	I	L	V	E	A	L
Z4	F	D	-	G	I	L	V	Q	A	V
Z5	Y	E	G	G	A	V	V	Q	A	L

Рис. 4.2. Множинне вирівнювання п’яти послідовностей Z1–Z5. Послідовності розміщені так, щоб ідентичні і подібні залишки ввійшли в одну колонку. Для цього в послідовності Z1, Z2, Z4 уведені розриви, їх позначено короткою рискою

Множинне вирівнювання можна підсумувати одним рядком – *псевдопослідовністю* (консенсус). Його, зазвичай, подають унизу (рис. 4.3). Псевдопослідовність складається з символів, які узагальнюють природу вирівнювання у кожній колонці. Консенсусна послідовність не обов’язково має бути у вигляді одного рядка. Відома низка методів, які дають змогу створювати матриці різного типу на основі вирівнювань, і використовують їх для аналізу баз даних білків. Ці методи розглянемо згодом.

Z1	Y	D	G	G	A	V	-	E	A	L
Z2	Y	D	G	G	-	-	-	E	A	L
Z3	F	E	G	G	I	L	V	E	A	L
Z4	F	D	-	G	I	L	V	Q	A	V
Z5	Y	E	G	G	A	V	V	Q	A	L
	y	d	G	G	A/I	V/L	V	e	A	l

Рис. 4.3. Множинне вирівнювання п’яти послідовностей Z1–Z5 (внизу виділено зеленим кольором консенсусну послідовність)

Множинне вирівнювання – дуже інтенсивний процес у сенсі часу обчислення, а саме: час, необхідний для вирівнювання n послідовностей довжиною m, дорівнює $O(m^n)$, де O – коефіцієнт аналізу. Тому час аналізу зростає експоненційно відповідно до того, як збільшується кількість послідовностей. Станом на сьогодні описано вирівнювання десятків тисяч послідовностей, і це стало можливим на основі евристичних підходів до множинних вирівнювань, які дають змогу пришвидшити аналіз.

Незважаючи на велику кількість евристичних прийомів, алгоритмічно відмінних підходів до множинного вирівнювання небагато. В основу більшості з них, зокрема і в найпопулярніших, які розглядатимемо у цьому підрозділі, покладено використання певної *об'єктивної функції*, що вказує на найоптимальніший варіант з міріад можливих варіантів множинних вирівнювань. Наприклад, об'єктивною функцією у попарних вирівнюваннях є рахунок вирівнювання – сума рахунків усіх позицій вирівнювання і штрафів за розриви. Найкоректнішим (у математичному значенні) вважають вирівнювання, що максимізує суму рахунків, яку отримує кожна пара залишків. Цей підхід можна використати і для множинних вирівнювань: тут алгоритм мав би шукати вирівнювання, що максимізує суму подібностей усіх пар послідовностей (sum-of-pairs score, або *SP-рахунок*). Для попарних вирівнювань обчислення рахунку займає кілька секунд. Однак обчислення SP-рахунку – складне завдання навіть для декількох десятків послідовностей і залишається практично нерозв'язним завданням для сотень унаслідок астрономічного числа варіантів для обчислення (див. попередній текст). Тому необхідне використання евристичних підходів, які б спростили процес пошуку найкоректнішого множинного вирівнювання. Саме розгляду цих евристичних підходів і присвячено решту цього підрозділу.

Незалежно від програми (алгоритму), у процесі побудови множинного вирівнювання можна виокремити такі основні етапи:

1. Відібрати послідовності для вирівнювання (за допомогою пошуку баз даних, аналізу літератури тощо).
2. Локалізувати ділянки кожної послідовності, які потрібно проаналізувати. Не треба намагатися вирівняти послідовності, що значно відрізняються за довжиною. Тому варто відредагувати послідовності – скоротити їх до тих сегментів, що виявляють подібність один до одного.
3. В ідеалі необхідно оцінити міру подібності множини послідовностей за допомогою попарного вирівнювання їхніх рандомізованих варіантів. Насамперед вирівнюють послідовності, що формують кластери вище 6σ. Як альтернативу до тесту з рандомізації треба взяти за правило вирівнювати тільки ті послідовності, величина E яких < 1.
4. Виконати множинне вирівнювання за допомогою комп'ютерної програми.
5. Виконати візуальну інспекцію множинного вирівнювання для виявлення багатих на розриви ділянок.
6. Видалити послідовності, що значно порушують вирівнювання, і повторити множинне вирівнювання – вже без проблемної послідовності.
7. Після визначення ключових амінокислотних залишків у легко вирівнюваній вибірці треба спробувати додати до неї проблемні послідовності.

4.2. Прогресивні методи множинного вирівнювання – на прикладі CLUSTAL W2. Ці методи – одні з найпопулярніших на сьогодні, швидкі та акуратні. Типовими зразками цієї групи є програми CLUSTAL X і CLUSTAL W (веб-адреси цих і інших програм множинного вирівнювання, про які згадано у тексті наприкінці цього підрозділу). Прогресивні методи складаються з трьох етапів: обчислення *матриці відстаней* між усіма можливими парами послідовностей; побудови *дерева-провідника* (guide tree) на основі матриці відстаней, на основі цього дерева; застосування дерева-провідника для *прогресивного вирівнювання* послідовностей. Отже, елементом евристики у прогресивних методах є не пошук оптимального вирівнювання всіх послідовностей зразу, а пошук спочатку найподібнішої пари, до якої поступово будуть вирівнювати решту, на основі певної філогенетичної моделі (дерево-провідник). Це значно спрощує процес множинного вирівнювання. Метод проілюстровано на рис. 4.4 на прикладі множинного вирівнювання глобінових білків тварин (гемоглобінів людини, коня тощо) та рослини (леггемоглобін люпину). Спочатку всі послідовності попарно вирівнюють за допомогою розглянутих методів (FastA, BLAST тощо), що дає змогу отримати рахунки вирівнювання S.

Для множинних вирівнювань можна прямо використовувати значення S як мірило відстані між досліджуваними послідовностями (чим більше S – тим ближчі послідовності). На практиці на основі S обчислюють, здебільшого, складніші показники. Наприклад, дві послідовності рандомізують (випадково змінюють послідовність амінокислотних залишків), зазвичай, 100 разів. Визначають рахунки попарного вирівнювання 100 рандомізованих пар, їхнє середнє значення x і стандартне відхилення σ. Потім обчислюють значення SD:

$$\mathrm{SD} = (S - x) / \sigma \, . \qquad (39)$$

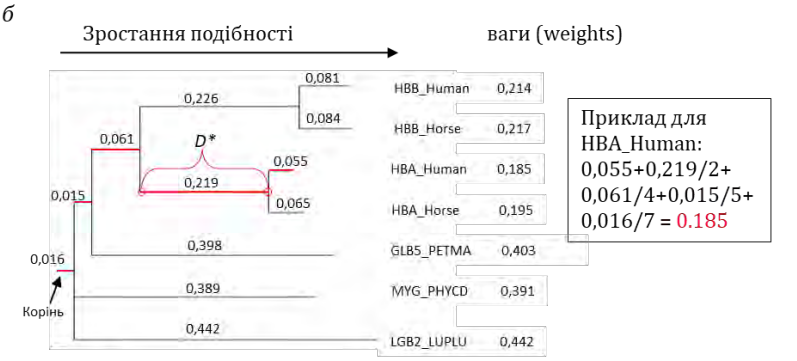
Значення SD дають змогу врахувати довжину і склад послідовностей, отож їм надають перевагу в ієрархічних методах, порівняно з S чи відсотками подібності послідовностей, вирівняних без розривів.

Обчислену матрицю значень далі скеровують для побудови дерева-провідника за допомогою певного алгоритму. Процес побудови дерева полягає в *кластеризації* – об'єднанні пари найближчих послідовностей, а потім поступовому приєднанні до цієї пари інших послідовностей. Сьогодні описано багато таких алгоритмів кластеризації, як об'єднання сусідів (neighbor-joining), UPGMA, K-середніх, розгляд яких виходить за межі цього підрозділу (їх розглядатимемо далі під час опису методів філогенетичної реконструкції). Кінцевим результатом кластеризації стає дендрограма, що відображає ступінь подібності між послідовностями (рис. 4.4, *б*). Множинне вирівнювання розпочинається з попарного вирівнювання найближчих послідовностей на дереві-провіднику, далі – у вирівнюванні чергової найближчої пари, керуючись порядком галуження дерева. Вирівнювання виконують за допомогою динамічних алгоритмів, на основі матриць PAM або BLOSUM. Для вирівнювання можуть слугувати не тільки окремі послідовності, а вирівнювання, які отримані під час аналізу. Якщо вирівнювання порівнюють з окремою послідовністю або іншим вирівнюванням, то розриви, наявні у вирівнюванні, будуть зберігатися.

а

	1	2	3	4	5	6	7
1: HBB_HUMAN	100,00						
2: HBB_Horse	83,67	100,00					
3: HBA_HUMAN	45,00	44,29	100,00				
4: HBA_Horse	43,57	45,00	87,32	100,00			
5: GLB5_PETMA	25,55	24,09	35,61	34,09	100,00		
6: MYG_PHYCD	24,66	24,66	27,46	26,76	23,19	100,00	
7: LGB2_LUPLU	18,06	19,44	16,31	17,02	16,55	17,57	100,00

б



в

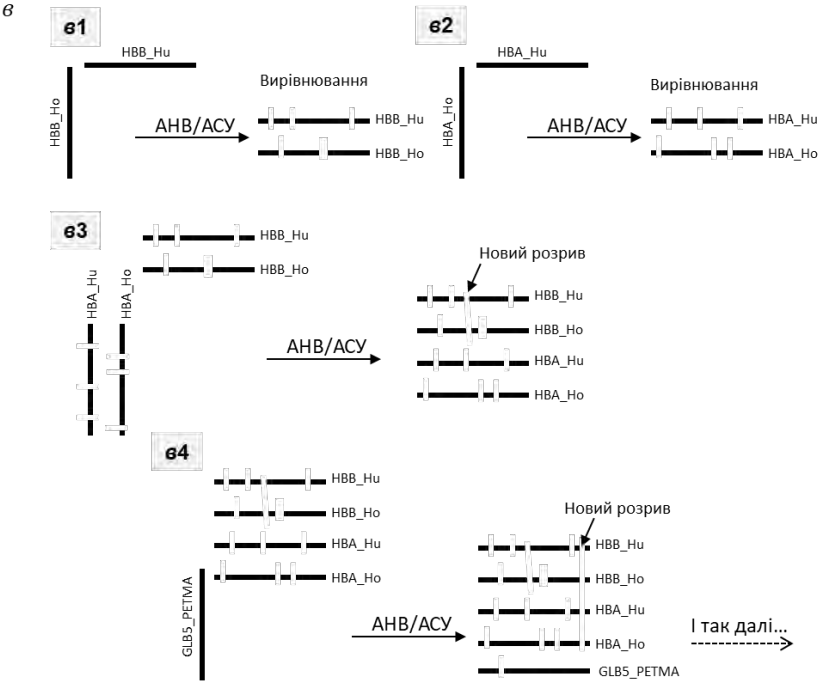


Рис. 4.4. Множинне вирівнювання амінокислотних послідовностей семи глобінових білків (NP_000509, NP_000509, NP_001078900, NP_001157490, P02185, P02208.2, P02240.2) ієрархічним методом CLUSTAL W2 (http://www.ebi.ac.uk/Tools/msa/clustalw2/): *а* – матриця відстаней між усіма парами послідовностей, виражена у відсотках ідентичності: зеленим кольором виділені найвищі відсотки ідентичності – між β-ланцюгами гемоглобіну людини (HBB_Human) і коня (HBB_Horse), а також α-ланцюгами; *б* – дерево-провідник, побудоване на основі матриці відстаней: довжини гілок дерева (вказані над гілками) пропорційні мірі дивергенції послідовностей (наприклад, відрізок *D** (позначений на дереві) має довжину 0,219 – це кількість амінокислотних замін на 100 залишків, яка відділяє гіпотетичного спільного предка послідовностей HBA_Human, HBA_Horse, зі спільним предком HBA_Human, HBA_Horse, HBB_Human, HBB_Horse); справа від дерева вказана вага кожної послідовності: відмінніші послідовності мають більшу вагу. Приклад обчислення ваги послідовності HBA_Human подано у рамці: це сума довжин гілок (позначені червоним кольором), які сполучають цю послідовність з коренем дерева, поділених на кількість послідовностей, які об'єднує відповідна гілка; *в* – ваги послідовностей використовують для множинного вирівнювання (докладніше пояснення – далі у тексті). Під час множинного вирівнювання фіксують перші та вводять нові розриви

Помилки вирівнювання, що виникли на ранніх етапах, можуть закріпитися і поширюватися на всю множину послідовностей. Для подолання цієї проблеми вживають низку заходів. *Ітеративно-прогресивні* методи (iterative progressive methods) – наприклад, MUSCLE – передбачають ревізію вихідного вирівнювання, а саме: після того, як усі послідовності вирівняні, програма повертається до найближчих послідовностей у дереві-провіднику і намагається їх повторно (краще) вирівняти (про цей підхід йтиме мова наприкінці підрозділу). Інший метод усунення помилок – використання різних матриць частот заміщень на різних стадіях вирівнювання. Наприклад, перші етапи вирівнювань ґрунтуються на матрицях заміщень для високоподібних послідовностей (наприклад, BLOSUM62), а для вирівнювання віддаленіших послідовностей (останні гілки дерева-провідника) програма може застосувати "глибші" матриці (BLOSUM50, BLOSUM45 тощо).

Ще один елемент евристики у множинних вирівнюваннях – це спосіб оцінювання розривів. У величину штрафу за відкриття і продовження розриву може бути закладена залежність від: *а*) типу використаної матриці заміщення; *б*) ступеня подібності послідовностей; *в*) їхньої довжини; *г*) відмінностей в довжинах; *д*) розміщення розриву; *е*) природи амінокислотних залишків навколо розриву. Наприклад, щодо п. *в* і *г*, величина штрафів більша для коротших послідовностей, щоб попередити накопичення в них надмірної кількості розривів. Установлення залежності штрафу від параметрів, описаних в п. *д* і *е*, базується на аналізі дуже якісних вирівнювань гомологічних білків зі встановленою вторинною структурою. Виявлено, що частоти появи розривів поблизу кожного з 20-ти амінокислотних залишків неоднакові, і цей факт використовують для локального коригування величини штрафів за відкриття розривів після кожного залишку. Короткі сегменти гідрофільних залишків (5 і більше), зазвичай, свідчать про петлі, що сполучають стабільні елементи вторинної структури (α-спіралі і β-шари). Тому штрафи за відкриття розривів на ділянках білка з петлевою структурою малі, а на інших ділянках – великі. Також мале значення мають величини штрафів за відкриття розривів на ділянках, де під час вирівнювання вже накопичилося багато розривів. Розриви, зазвичай, розміщуються неподалік, але не ближче, ніж за вісім амінокислотних залишків. Штраф зростає за відкриття нового розриву, якщо відстань між новим і наявним менша або рівна вісьмом залишкам.

Послідовності у множинному вирівнюванні "зважують" для коригування нерівномірності вибірки послідовностей у розрізі еволюційних відстаней між ними. Іншими словами, більшість послідовностей у вибірці для вирівнювання, зазвичай, походить з певної таксономічної групи, тоді як послідовності з інших груп мало репрезентовані. Процес *зважування* полягає у приписуванні меншої ваги дуже подібним послідовностям, і більшої ваги – найбільш відмінним. "Ваги" послідовностей обчислюють з довжин гілок вихідного дерева-провідника (див. рис. 4.4). Вирівнювання вирівнювань – здебільшого глобальні або напівглобальні; матриці, що використовують для обчислення *S*, містять тільки додатні величини; вирівнювання амінокислотного залишку проти розриву матиме нульове значення. Сумарний рахунок розділяють на число пар, що брали участь у вирівнюванні. Приклад обчислення рахунків множинного вирівнювання без і з урахуванням ваг послідовностей наведено на рис. 4.5.

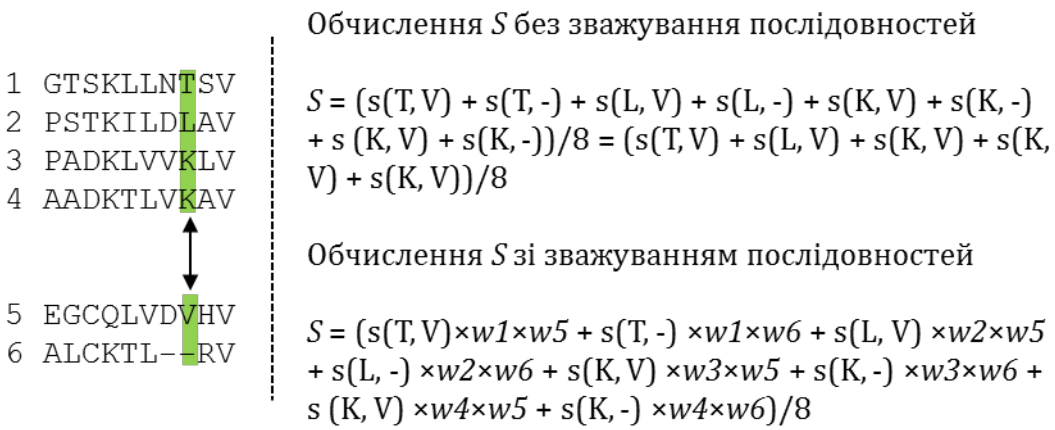


Рис. 4.5. Обчислення *S* обраної позиції (виділено зеленим) вирівнювання чотирьох і двох вирівнювань. У множинному вирівнюванні цієї позиції бере участь вісім пар послідовностей (4 × 2). Сумарний рахунок множинного вирівнювання цієї позиції – це сума восьми окремих *S* попарних вирівнювань, поділених на кількість пар. Вирівнювання можна виконувати без урахування ваг послідовностей і з їхнім урахуванням. У випадку врахування, значення *S* попарних вирівнювань перемножують на ваги w_i послідовностей

Результат процесу множинного вирівнювання – двовимірна таблиця, яка, у разі застосування алгоритму CLUSTAL W2 до вже описаних глобінових білків (див. рис. 4.4) виглядатиме так, як відображено на рис. 4.6.

```
HBB_HUMAN       --------MVHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDLS 50
HBB_Horse       --------MVQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDLS 50
HBA_HUMAN       ---------MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF-DLS 50
HBA_Horse       ---------MVLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHF-DLS 50
GLB5_PETMA      MPIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLT 60
MYG_PHYCD       ---------MVLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLK 51
LGB2_LUPLU      --------MGALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTS 52
                         *:  :   :   *  .          :  .:   * :   *  :   .

HBB_HUMAN       TPDAVMGNPKVKAHGKKVLGAFSDGLAHLDN-----LKGTFATLSELHCDKLHVDPENFR .05
HBB_Horse       NPGAVMGNPKVKAHGKKVLHSFGEGVHHLDN-----LKGTFAALSELHCDKLHVDPENFR .05
HBA_HUMAN       -----HGSAQVKGHGKKVADALTNAVAHVDD-----MPNALSALSDLHAHKLRVDPVNFK .00
HBA_Horse       -----HGSAQVKAHGQKVGDALTLAVGHLDD-----LPGALSNLSDLHAHKLRVDPVNFK .00
GLB5_PETMA      TADQLKKSADVRWHAERIINAVNDAVASMDDT--EKMSMKLRDLSGKHAKSFQVDPQYFK .18
MYG_PHYCD       TEAEMKASEDLKKHGVTVLTALGAILKKKGH-----HEAELKPLAQSHATKHKIPIKYLE .06
LGB2_LUPLU      EVP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-VADAHFP .09
                       . .:: *.  :   .                :  *.  *   .  :    :

HBB_HUMAN       LLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------ 147
HBB_Horse       LLGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH------ 147
HBA_HUMAN       LLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------ 142
HBA_Horse       LLSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSKYR------ 142
GLB5_PETMA      VLAAVIADTVAAG---------DAGFEKLMSMICILLRSAY------- 150
MYG_PHYCD       FISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG 154
LGB2_LUPLU      VVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA--- 154
                .:   :  .:           ...       .   :
```

Рис. 4.6. Множинне вирівнювання семи глобінових білків. Абсолютно консервативні позиції позначені під вирівнюванням зірочками; позиції, де розміщені подібні залишки – двокрапкою, позиції, де розміщені переважно подібні залишки – однією крапкою. Абсолютно консервативні залишки гістидину у двох позиціях показано червоним шрифтом. Прямокутниками виділено сім α-спіральних районів вирівняних послідовностей; помітно, що майже всі розриви зосереджені між α-спіральними сегментами. Вирівнювання отримано за допомогою програми CLUSTAL W2 (http://www.ebi.ac.uk/Tools/msa/clustalw2/), параметри за замовчуванням

Прогресивний принцип покладено в основу практично всіх сучасних методів множинного вирівнювання. Перелік (неповний) сучасних програм множинного вирівнювання можна знайти за адресою: http://toolkit.tuebingen.mpg.de/sections/alignment. Відмінності різних підходів полягають, передусім, у способах покращення вихідного множинного прогресивного вирівнювання – його ревізії, застосування особливої системи обчислення рахунків, поєднання локального і глобального попарного вирівнювань, залучення інформації про просторову будову вирівнюваних білків тощо.

4.3. T-COFFEE та MUSCLE. Обґрунтуємо потребу у вдосконаленні прогресивного принципу множинного вирівнювання. Після того, як знайдено найподібнішу пару послідовностей, процес вирівнювання "заморожується" – його перебудова не дозволена. Отже, всі недоліки найподібнішої пари – характерний набір збігів і розривів – накладаються на вирівнювання решти послідовностей, а остаточне множинне вирівнювання може відображати локальний, а не глобальний максимум. Тобто таке вирівнювання, ймовірно, є одним з найкращих, але не найкращим. Прогресивний підхід до множинного вирівнювання, як описано попередньо, неявно використовує логіку попарного вирівнювання, намагаючись максимізувати збіги (= SP-рахунок) у множині послідовностей, відштовхуючись від найподібнішої пари. Прогресивний підхід не дає змоги використовувати всю наявну інформацію у множині послідовностей, що їх вирівнюють, адже не вирівнює водночас їх усіх. Для повнішого використання інформації, що прихована у масиві послідовностей, потрібно мати певне загальне уявлення про цілий масив – передусім про кількість і розташування ділянок низької подібності, що будуть перешкодою на шляху оптимального множинного вирівнювання. У цьому випадку можна скористатися таким евристичним підходом: на основі множини послідовностей (з яких планують утворити множинне вирівнювання) створити бібліотеку всіх можливих попарних вирівнювань і шукати між ними *узгодженість* (consistency). Тоді оптимальне множинне вирівнювання можна визначити як найузгодженіше. Перший докладний опис алгоритму множинного вирівнювання на основі принципу узгодженості виконав Седрік Нотрдам 1998 року (PMID9682054), увівши нову об'єктивну функцію (COFFEE; **C**onsistency based **O**bjective **F**unction **F**or alignm**E**nt **E**valution). Ним створений онлайн-сервіс, T-COFFEE (http://tcoffee.crg.cat/), на якому можна скористатися низкою програм множинного вирівнювання, що працюють на основі принципу узгодженості. У функціонуванні всіх COFFEE-програм можна виокремити три ключові складові: (1) множинне вирівнювання, яке можна створити будь-яким алгоритмом; (2) референтну бібліотеку попарних вирівнювань усіх послідовностей, з яких побудовано множинне вирівнювання між цими вирівнюваннями; (3) оцінення об'єктивної функції узгодженості між множинним вирівнюванням і попарними вирівнюваннями, що містяться у бібліотеці. Нехай маємо N вирівняних послідовностей, від S_1 до S_N, A_{ij} – попарна проєкція (отримана з множинного вирівнювання) послідовностей S_i та S_j, LEN(A_{ij}) – довжина цього вирівнювання, SCORE(A_{ij}) – загальна узгодженість (рівень ідентичності) між A_{ij} і попарним вирівнюванням S_i та S_j з референтної бібліотеки, W_{ij} – вага, асоційована з цим попарним вирівнюванням. Рівні ідентичності між попарно вирівняними послідовностями (чи їхніми проєкціями) можна подати як у відсотках збігів, так

і в частках від одиниці. На підставі цих визначень рахунок COFFEE можна визначити як:

$$COFFEE\ score = \frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} \times SCORE(A_{ij})}{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} W_{ij} \times LEN(A_{ij})}, \quad (40)$$

де $SCORE(A_{ij})$ – кількість вирівняних пар залишків, що є спільними для A_{ij} та референтної бібліотеки.

Принцип визначення рахунку COFFEE підсумовано на рис. 4.7. Фактично рахунок узгодженості, який використовується в T-COFFEE, є мірою того, як часто два залишки, що формують одну позицію попарного вирівнювання, опиняються разом в остаточному множинному вирівнюванні. Отже, після обчислення матриці відстаней самі послідовності не використовуються у формуванні множинного вирівнювання методом CLUSTAL W2, водночас T-COFFEE використовує інформацію про частоти пар залишків як спосіб обмеження можливих варіантів множинного вирівнювання до тих, які узгоджуються з результатами попарного вирівнювання і, вірогідно, є біологічно коректними. Більше про принципи множинного вирівнювання методом T-COFFEE можна знайти тут: https://www.tcoffee.org/Publications/Pdf/tcoffee.pdf

Складність всієї процедури множинного вирівнювання методом T-COFFEE визначається таким рівнянням:

$$O(N^2L^2) + O(N^3L) + O(N^3) + O(NL^2),$$

де N – кількість послідовностей, з яких будують множинне вирівнювання; L – довжина послідовності; $O(N^2L^2)$ – складність, що пов'язана з обчисленням бібліотеки попарних вирівнювань; $O(N^3L)$ – складність, що пов'язана: з формуванням масиву для множинного вирівнювання; $O(N^3)$ – з побудовою дерева провідника методом з'єднання сусідів; $O(NL^2)$ – з прогресивним алгоритмом побудови множинного вирівнювання.

Емпіричний аналіз ефективності алгоритму на основі масиву послідовностей приблизно однакової довжини засвідчив, що його загальна складність описується квадратичною функцією. Щодо швидкодії T-COFFEE, то програма приблизно вдвічі повільніша за алгоритм прогресивного вирівнювання ClustalW.

Типовий результат множинного вирівнювання з використанням веб-сервера T-COFFEE, а також нормалізовані рахунки COFFEE для залишків, стовпчиків та всього вирівнювання (на основі яких сформоване множинне вирівнювання) відображено на рис. 4.8. У цьому випадку множинне вирівнювання виконане за допомогою прогресивного алгоритму, а його ревізія – на основі підходу узгодженості, як описано попередньо і відображено на рис. 4.7.

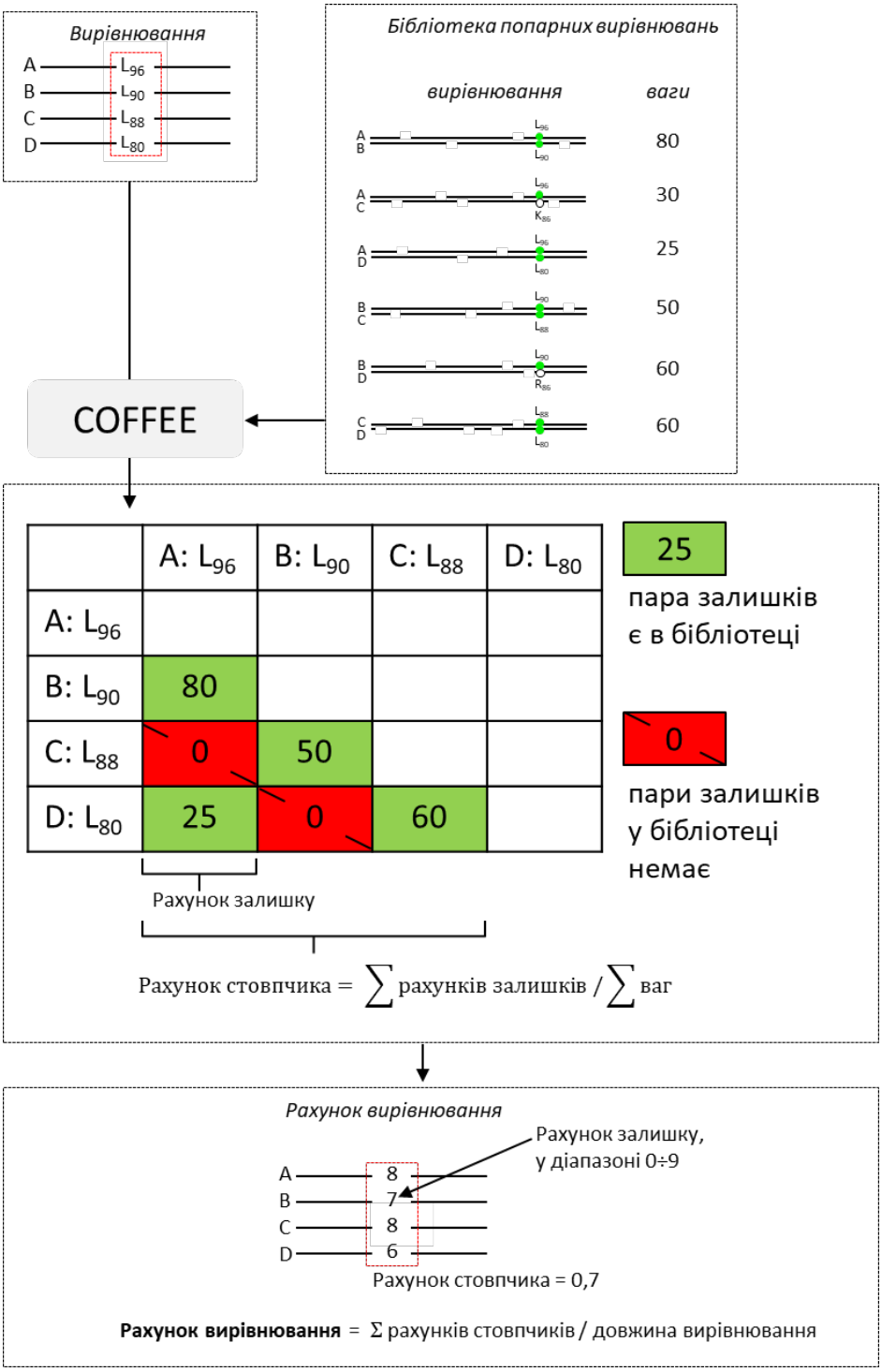
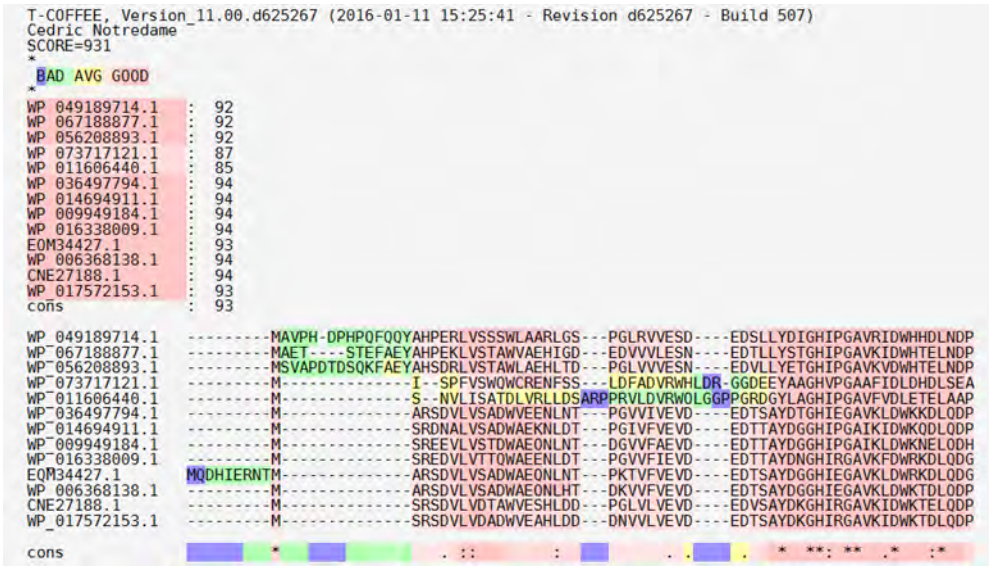


Рис. 4.7. Схема обчислення рахунку COFFEE для множинного вирівнювання. Кожен стовпчик вирівнювання оцінюють за функцією COFFEE, використовуючи референтну бібліотеку (РБ) попарних вирівнювань. Кожна пара залишків у вирівнюванні отримує рахунок відповідно до обраної матриці заміщення. У матриці пара отримує нульовий рахунок, якщо її немає у РБ, або рахунок, що дорівнює вазі пари послідовностей, у якій ця пара наявна. Оскільки матриця симетрична, то рахунок стовпчика рівний сумі половини комірок матриці, за винятком основної діагоналі. Отримане значення розділяють на максимальний рахунок кожної комірки (сума ваг, які приписані певному залишку у РБ). Рахунок залишку рівний сумі комірок одного рядка матриці, розділених на суму максимальних рахунків цих комірок

На основі принципу узгодженості створено низку програм, які дають змогу аналізувати послідовності РНК, ДНК, амінокислот, а також дво- та тривимірні структури на їхній основі. Крім того, розроблено метод (Transitive Consitency Score), який дає змогу оцінити якість будь-якого вирівнювання (doi:10.1093/nar/gkv310).

а



б

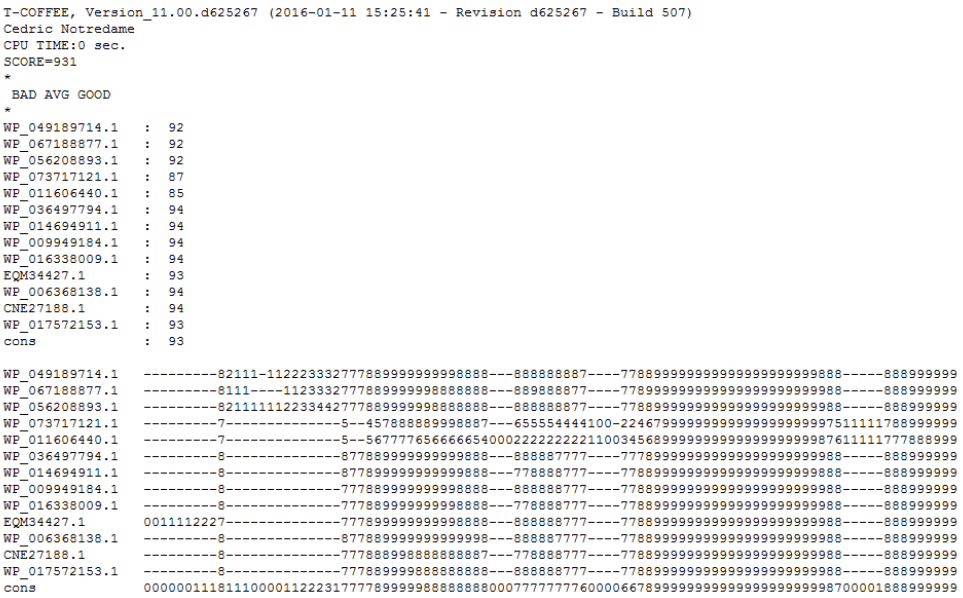


Рис. 4.8. Множинне вирівнювання, виконане на веб-сервері T-COFFEE (*а*) та рахунки COFFEE (*б*), на яких ґрунтується це множинне вирівнювання. Кольоровий код (BAD→AVG→GOOD) множинного вирівнювання позначає ділянки з різною якістю вирівнювання. Кожному залишкові послідовності і всьому вирівнюванню приписаний рахунок від 0 до 10

Однак принцип узгодженості та програма T-COFFEE не дають змоги отримувати оптимальні вирівнювання в усіх випадках. Ця програма часто зумовлює хибні вирівнювання у N- та С-кінцевих ділянках. Як приклад розглянемо вирівнювання п’яти SH2-доменів двома програмами – T-COFFEE та MUSCLE (рис. 4.9).

а

```
YES_XIPHE    MGCvrSKEaKgPAlKYqpdNsnvvPvSahlgHYGpeptimg
YES_AVISY    --------------dKgPAmKYrtdNtpePiSshvsHYGsd
YES_CHICK    ------MGCikSKEdKgPAmKYrtdNtpePiSshvsHYGsd
YES_HUMAN    ------MGCikSKEnKsPAiKYrpeNtpePvStsvsHYGae
YES_MOUSE    ------MGCikSKEnKsPAiKYtpeNlteP--vSpsasHYG
```

б

```
YES_XIPHE    MGCvrSKEaKgPAlKYqpdNsnvvPvSahlgHYGpeptimg
YES_AVISY    --------dKgPAmKYrtdNtp-ePiSshvsHYGsdssqat
YES_CHICK    MGCikSKEdKgPAmKYrtdNtp-ePiSshvsHYGsdssqat
YES_HUMAN    MGCikSKEnKsPAiKYrpeNtp-ePvStsvsHYGaepttvs
YES_MOUSE    MGCikSKEnKsPAiKYtpeNlt-ePvSpsasHYGvehatva
```

Рис. 4.9. Вирівнювання послідовностей SH2-доменів за допомогою програм T-COFFEE (*а*) та MUSCLE (*б*). Зображені N-кінцеві ділянки п’яти послідовностей. Консервативні трипептиди виділені кольором, їх не вирівнює T-COFFEE

Видно, що прогресивний алгоритм (T-COFFEE) нездатний коректно вирівняти консервативні послідовності у цих білках, а MUSCLE – здатний. Особливістю останньої програми є циклічний (*ітеративний*) алгоритм покращення вихідного множинного вирівнювання. Перший крок – обчислення дерева-провідника на основі *k*-мерів – двох підпослідовностей, що є спільними для пари послідовностей з масиву, що підлягає множинному вирівнюванню. Порівняно з попарним вирівнюванням цілих послідовностей (CLUSTALW), вирівнювання *k*-мерів у 3 000 (приблизно) разів швидше, хоча й менш акуратне. Цей крок називають кластеризацією *k*-мерів, він веде до дерева-провідника.

Другий крок – прогресивне вирівнювання на основі дерева. У ноді (ділянці розгалуження) дерева створюється попарне вирівнювання у напрямі від термінальних ділянок дерева (листків) до кореня (рис. 4.10). Перше вирівнювання виконують між двома послідовностями. Далі вирівнювання буде одного із трьох типів: послідовність–послідовність, профіль–послідовність, профіль–профіль. Тут під терміном “профіль” мають на увазі множинне вирівнювання послідовностей, які описують певну внутрішню ноду дерева. Фактично цей крок нагадує процедуру, що виконує CLUSTALW на основі дерева-провідника. Так отримують первинне множинне вирівнювання, якість якого доволі висока, але все ж потребує вдосконалення.

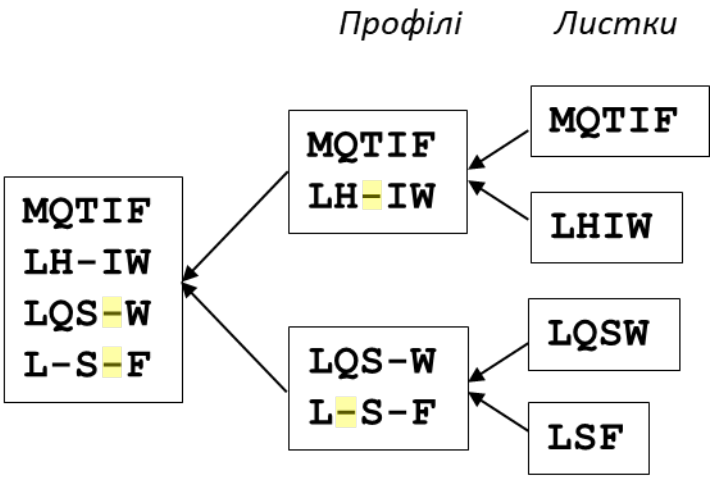


Рис. 4.10. Вирівнювання за методом MUSCLE. Профілі вирівнюють так, що їхні колонки зберігаються. Колонки з інделами (жовтий фон) уводять за потреби, коли необхідно вирівняти колонки між собою. Рахунок вирівнювання визначають за функцією профілю, яка має присвоїти високий рахунок парам колонок, що містять подібні амінокислоти

На основі отриманого множинного вирівнювання обчислюють ступінь ідентичності кожної пари послідовностей. Це веде до нової матриці відстаней, з якої визначають нове дерево. Порівнюють старе і нові дерева: якщо вони відмінні, то перевирівнюють послідовності. Всю процедуру визначення матриці ідентичності, нового дерева і повторного вирівнювання повторюють доти, доки не отримають стабільне дерево (яке не відрізняється від попереднього), або певне стале число раундів ітерації.

Далі масив послідовностей розділяють на дві підмножини, на основі кожної конструюють профіль. Профілі перевирівнюють з метою отримання нового вирівнювання, що матиме вищий SP-рахунок згідно з певною об'єктивною функцією. У MUSCLE нею є рахунок log-очікування (LE score). Нехай i та j – амінокислотні залишки певного типу, p_i – фонова ймовірність (частота) i, p_{ij} – спряжена ймовірність того, що i та j буде вирівняно (вони сформують пару у вирівнюванні), f^x_i – фактична частота i у стовпчику x першого профілю, f^x_G – фактична частота розривів у тому стовпчику у позиції x в родині (подібно – і для позиції y в другому профілі). Очікувана ймовірність α^x_i натрапити на амінокислоту i у позиції x можна вивести з f^x, зазвичай за допомогою додавання евристичних псевдорахунків або використовуючи баєзіанські методи, такі як апріорні значення з сумішей Діріхле. Про них докладніше описано у блоці 9 (підрозділ розділ 5.2). Звідси рахунок log-очікування:

$$LE^{xy} = \frac{(1-f_G^x)(1-f_G^y)log\sum_i\sum_j f_i^x f_j^y p_{ij}}{p_i p_j}. \qquad (41)$$

MUSCLE бере фонові частоти p_i та p_{ij} з матриць PAM240. Програма повторює формування нових профілів та обчислення LE-рахунку до моменту настання

конвергенції – стану, коли подальша перебудова дерева не покращує його об'єктивну функцію. Програма MUSCLE доступна тут: https://toolkit.tuebingen.mpg.de/tools/muscle.

4.4. Нові алгоритми та супровідні сервіси. Оскільки множинне вирівнювання є вихідним матеріалом для багатьох інших методів біоінформатики, то зберігається значний інтерес до розроблення нових, точніших чи потужніших методів множинного вирівнювання. Ще одним мотивом є постійне зростання кількості даних – для вирівнювання стають доступними десятки тисяч і мільйони послідовностей. З сучасних розробок варто згадати регресивний алгоритм, в якому вирівнювання розпочинається не з пари найподібніших послідовностей, а з попарного вирівнювання найвіддаленішої пари послідовностей (https://pubmed.ncbi.nlm.nih.gov/31792410/). Це попереджає надмірне накопичення розривів, яке спостерігають за використання прогресивного алгоритму на пізніх етапах вирівнювання дуже багатьох послідовностей (що, як гадають, є головною причиною низької якості таких множинних вирівнювань). Іншим елементом евристики запропонованого підходу є принцип "розділяй і владарюй" – великий масив розбивають на підмасиви (кожен з яких перетворюється на множинне вирівнювання), які далі зливають в одне.

Для об'єктивного порівняння різних методів множинного вирівнювання створена база BALIBASE (http://www.lbgi.fr/balibase/) – колекція референтних множинних вирівнювань, виконаних вручну, на основі білків, для яких відома тривимірна структура (за винятком трансмембранних білків). Різні референтні набори дають змогу тестувати вплив різних чинників (відмінності у подібності, довжинах послідовностей, інсерції, філогенетичні відмінності тощо) на остаточне множинне вирівнювання, отримане за допомогою різних алгоритмів. BALIBASE також містить програму для обчислення рахунків за принципом SP-рахунку (див. с. 158) різних множинних вирівнювань (bali_score_src), які можна потім порівнювати з рахунком референтного (найкращого, на експертну думку творців бази) множинного вирівнювання у базі. Більше про базу та її застосування можна знайти тут: https://www.ncbi.nlm.nih.gov/ pmc/articles/PMC29792/. BALIBASE є першою, але не єдиною базою референтних множинних вирівнювань. З нових розробок варто згадати базу референтних множинних вирівнювань нуклеотидних послідовностей mdsas, побудовану на основі кодувальних послідовностей білків з відомою третинною структурою (http://dna.cs.byu.edu/mdsas/download.shtml). В той час як інші сервіси зосереджені на аналізі білків, mdsas вирішує проблему "калібрування" послідовностей ДНК. Більше про інші сервіси для тестування методів множинного вирівнювання можна дізнатися тут: https://cutt.ly/2JKKyf1.

Візуалізація множинних вирівнювань є ще одним важливим елементом, який варто згадати. Наприклад, остаточним результатом часто є множинне вирівнювання, зображене внизу на рис. 4.11. У такому вигляді множинне вирівнювання малоінформативне (зокрема, важко відрізнити абсолютно консервативні, консервативні та варіативні позиції). До такого множинного вирівнювання можна застосувати певну стратегію зафарбування амінокислотних залишків з однаковими чи подібними фізико-хімічними властивостями, що полегшить сприйняття результату.

Рис. 4.11. Вихідний результат множинного вирівнювання набору білків методом T-COFFEE на сайті https://toolkit.tuebingen.mpg.de/

Так, зображене вище множинне вирівнювання отримане методом T-COFFEE на сайті Bioinformatic Toolkit Тюбінгенського університету. Цей сайт дає змогу застосувати до вирівнювання два способи розфарбування (рис. 4.12).

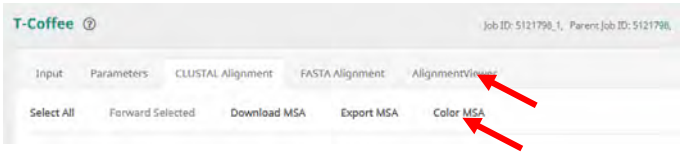


Рис. 4.12. Панель команд програми T-COFFEE на сайті https://toolkit.tuebingen.mpg.de/ (червоними стрілками позначено команди, що дають змогу візуалізувати множинне вирівнювання)

При виборі команди Color MSA амінокислотні залишки з подібними фізико-хімічними властивостями зафарбовують одним кольором (рис. 4.13).

Рис. 4.13. Результат виконання команди Color MSA на множинному вирівнюванні, зображеному на рис. 4.11

Багатше меню можливостей пропонує закладка Alignment Viewer. За переходу на неї дослідник має багато варіантів графічного репрезентування множинного вирівнювання (зокрема, долучення метаданих: консенсусний рядок, відсоток прогалин, логотип вирівнювання) та його розфарбування. Щодо останнього можна застосовувати різні типи візуалізації, наприклад за ступенем гідрофобності амінокислотних залишків (рис. 4.14, *а*) або за класифікацією амінокислотних залишків за Тейлором (рис. 4.14, *б*) та інші. Більше про типи і значення різних схем розфарбування нуклеотидних й амінокислотних послідовностей можна отримати тут: https://doc.ugene.net/wiki/display/UM/Coloring+Schemes.

а

б

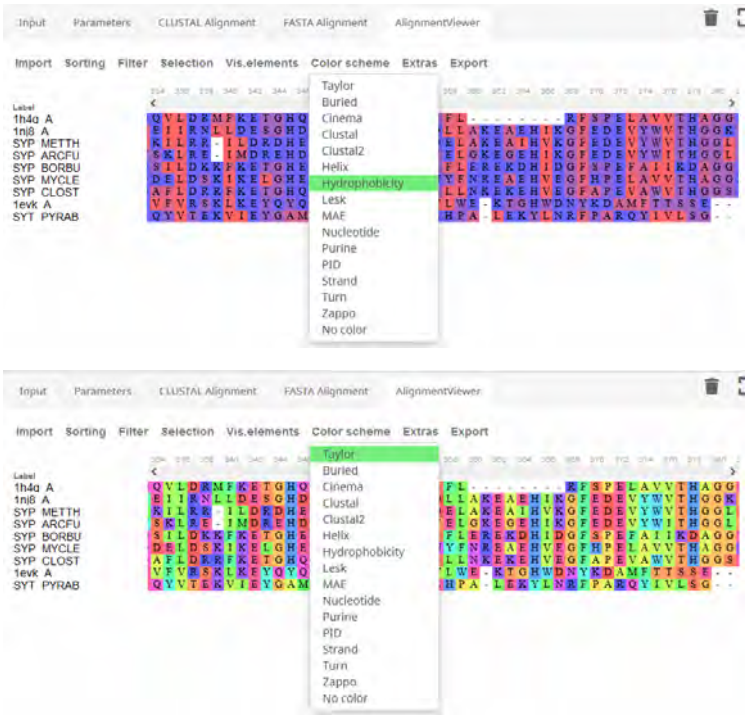


Рис. 4.14. Результат застосування різних способів розфарбування множинного вирівнювання на закладці AlignmentViewer на сайті https://toolkit.tuebingen.mpg.de/. Вибір можливих кольорових схем подано на спадному меню: зображено розфарбування за ступенем гідрофобності амінокислотних залишків (*а*) та розфарбування за Тейлором (*б*)

Незважаючи на складність і витонченість сучасних методів вирівнювання, потрібно пам’ятати: жоден алгоритмічний прийом не виправить наслідків неправильного чи невдалого обрання масиву даних для аналізу. Це правило влучно і стисло описує англійська максима: “garbage in – garbage out” (сміття в комп’ютер – сміття з комп’ютера). Бажано, щоб послідовності мали приблизно однакову довжину і виявляли подібність одна до одної від початку і до кінця; масив даних має містити кілька споріднених послідовностей, щодо яких можна буде розміщувати віддаленіші.

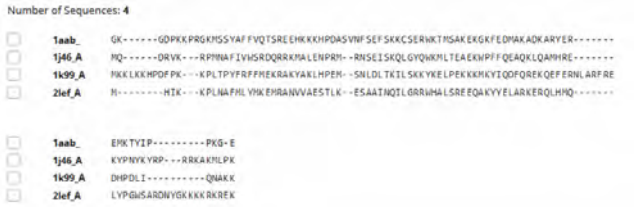
Типові задачі до розділу 4

Задача 4.1. Задано масив послідовностей у форматі multifasta:

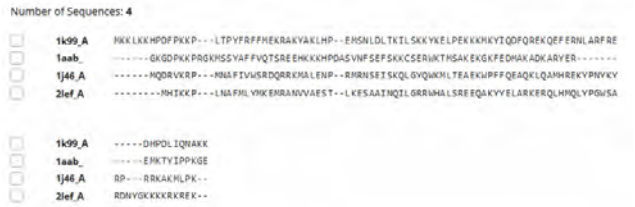
```
>1aab_
GKGDPKKPRGKMSSYAFFVQTSREEHKKKHPDASVNFSEFSKKCSERWKTMSAKEKGKFEDMAKADKAR
YEREMKTYIPPKGE
>1j46_A
MQDRVKRPMNAFIVWSRDQRRKMALENPRMRNSEISKQLGYQWKMLTEAEKWPFFQEAQKLQAMHREKY
PNYKYRPRRKAKMLPK
>1k99_A
MKKLKKHPDFPKKPLTPYFRFFMEKRAKYAKLHPEMSNLDLTKILSKKYKELPEKKKMKYIQDFQREKQ
EFERNLARFREDHPDLIQNAKK
>2lef_A
MHIKKPLNAFMLYMKEMRANVVAESTLKESAAINQILGRRWHALSREEQAKYYELARKERQLHMQLYPG
WSARDNYGKKKKRKREK
```

Виконайте їхнє множинне вирівнювання за допомогою алгоритмів T-COFFEE та MUSCLE. Який алгоритм, на Вашу думку, приводить до точнішого вирівнювання? Відповідь обґрунтуйте.

Розв'язання. На сайті https://toolkit.tuebingen.mpg.de/ знаходимо закладки програм (→ Alignment→). Завантажуємо у діалогове вікно програми T-COFFEE заданий масив. Виконуємо вирівнювання з параметрами за замовчуванням (нічого не міняємо). Отримуємо результат (внизу).



Аналогічно виконуємо аналіз за допомогою алгоритму MUSCLE, результат внизу.



Візуальне порівняння отриманих вирівнювань засвідчує, що T-COFFEE обрала глобальний тип вирівнювання – програма намагається вирівняти усі послідовності від початку до кінця. Водночас у результаті за алгоритмом

MUSCLE більше розривів на кінцях. Для множинних вирівнювань немає об'єктивної мірки точності. Тут можна скористатися порівнянням кількості розривів, які ввів кожен алгоритм у процесі побудови вирівнювання. Підрахунок виявив, що більшу кількість розривів увів алгоритм T-COFFEE – 78, тоді як MUSCLE увів 59. Користуючись закладкою AlignmentViewer (див. основний текст, рис. 4.12, 4.14), можна оцінити відсоток ідентичності послідовностей у множинному вирівнюванні відносно найменш подібної. Результат такого аналізу зображено внизу. Бачимо, що за допомогою алгоритму T-COFFEE досягли більшого відсотка ідентичності.

Отже, два різні підходи до оцінювання якості множинного вирівнювання дають різну відповідь щодо того, котрий саме алгоритм приводить до біологічно коректнішого вирівнювання. Використаний масив послідовностей походить з референтної бази BALIBASE, для нього існує референтне (коректне, таке, що ґрунтується на знанні тривимірної будови білків) множинне вирівнювання (http://www.lbgi.fr/balibase/BalibaseDownload/; папка RV1, файл BB11001, формат .msf). Воно має такий вигляд:

```
1aab_       ...GKGDPKK PRGKMSSYAF FVQTSREEHK KKHPDASVNF SEFSKKCSER
1j46_A      ......MQDR VKRPMNAFIV WSRDQRRKMA LENP..RMRN SEISKQLGYQ
1k99_A      MKKLKKHPDF PKKPLTPYFR FFMEKRAKYA KLHP..EMSN LDLTKILSKK
2lef_A      ........MH IKKPLNAFML YMKEMRANVV AEST..LKES AAINQILGRR

1aab_       WKTMSAKEKG KFEDMAKADK ARYEREMKTY IPPKGE.... ......
1j46_A      WKMLTEAEKW PFFQEAQKLQ AMHREKYPNY KYRPRRKAKM LPK...
1k99_A      YKELPEKKKM KYIQDFQREK QEFERNLARF REDHPDLIQN AKK...
2lef_A      WHALSREEQA KYYELARKER QLHMQLYPGW SARDNYGKKK KRKREK
```

Помітно, що множинне вирівнювання алгоритмом MUSCLE більше нагадує референтне – і те, й інше за своїм виглядом аналогічні локальному вирівнюванню з мінімізацією кількості уведених розривів. Отже, вважаємо, що саме алгоритм MUSCLE приводить до коректнішого множинного вирівнювання.

Контрольні запитання до розділу 4

1. Що таке множинне вирівнювання?
2. Що таке консенсусний рядок?
3. Опишіть основні етапи множинного вирівнювання.
4. У чому полягає прогресивний принцип множинного вирівнювання?
5. Чи відомі об’єктивні функції оцінювання якості множинних вирівнювань?Відповідь поясніть.
6. Розкажіть про основні етапи множинного вирівнювання за допомогою алгоритму CLUSTAL W2.
7. Що таке матриця відстаней і дерево-провідник?
8. У чому полягає суть принципу зважування послідовностей?
9. Що таке кластеризація послідовностей?
10. За якими правилами вводять розриви в множинне вирівнювання за методом CLUSTAL W2?
11. Які недоліки прогресивних методів множинного вирівнювання?
12. Принцип функціонування алгоритму T-COFFEE, його відмінності від CLUSTAL W2.
13. Що таке рахунок узгодженості?
14. Принцип функціонування алгоритму MUSCLE, його відмінності від CLUSTAL W2.
15. Чому для множинних вирівнювань немає статистичних параметрів, які обчислюють для попарних вирівнювань, як-от *E*, подібність, рахунок вирівнювання?
16. Нехай задано чотири нуклеотидні послідовності:

```
>Ex_1
CTATCCG
>Ex_2
TCCG
>Ex_3
CTGTACTG
>Ex_4
CTGTCTG
```

Виконайте множинне вирівнювання цього масиву послідовностей так, аби воно мало максимальний рахунок. Множинне вирівнювання виконуємо шляхом додавання символу розриву “ – ” до послідовностей, щоб в остаточному результаті усі послідовності мали однаковий розмір. Рахунок множинного вирівнювання обчислюємо як суму рахунків усіх можливих попарних вирівнювань послідовностей після того, як вони стали частиною множинного вирівнювання, тобто з розривами. Система рахунків наступна: збіг символів (чи збіг розривів): 0; незбіг: -1.

РОЗДІЛ 5

Моделі й бази даних на основі множинних вирівнювань

У процесі вирівнювання множини структурно чи функціонально споріднених білків зазвичай виявляють, що різні позиції (стовпчики вирівнювання) мають різний ступінь консервативності, як зображено на рис. 5.1. Деякі позиції абсолютно консервативні (репрезентовані тільки одним амінокислотним чи нуклеотидним залишком), на протилежному полюсі – високоваріабельні позиції, в яких (за умови, що вибірка послідовностей достатньо велика) трапляються всі можливі літери.

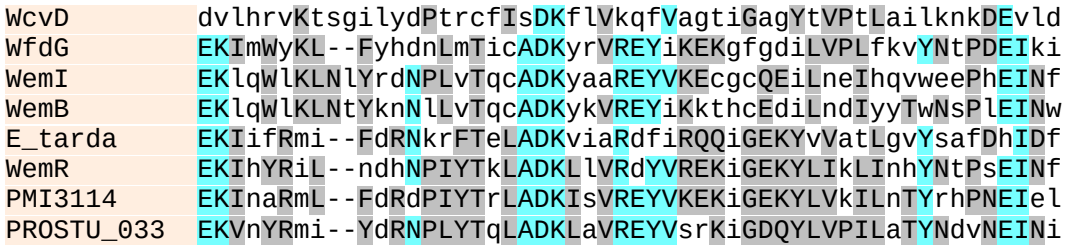


Рис. 5.1. Фрагмент множинного вирівнювання ймовірних глікозилтрансфераз, задіяних у синтезі К- та О-антигенів Грам-негативних бактерій. Кольором виділені найконсервативніші позиції (стовпчики, де значно переважає один амінокислотний залишок). Кольоровий код – це середня величина *S* за матрицею BLOSUM62: позиції з максимальним значенням виділені блакитним, із низьким – сірим. Вирівнювання і розфарбування виконане за допомогою програми MUSCLE на сайті: http://www.phylogeny.fr/version2_cgi/ one_task.cgi?task_type=muscle; параметри за замовчуванням

Цю інформацію, недоступну під час аналізу попарних вирівнювань, можна застосувати для створення певних моделей, які узагальнюють дані про структуру і функцію групи споріднених НАП. Такі моделі є знаряддям пошуку гомологічних НАП. По-перше, вони можуть давати уявлення про висококонсервативні ділянки білка – мотиви, які задіяні у виконання ним біологічної функції; по-друге, вони дають змогу передбачити просторову будову білка; по-третє, вони дають інформацію, що стосується філогенії досліджуваної групи білків. Найпростіша узагальнююча модель множинного вирівнювання НАП – його *консенсусна послідовність*, про яку йшла мова на початку розділу 4 (див. рис. 4.3). Вона дає інформацію про локалізацію найконсервативніших позицій, що, ймовірно, важливі для виконання білком чи нуклеїновою кислотою біологічної функції. Консенсусні послідовності будують за правилом більшості і на основі ділянок НАП, де не спостерігають значну варіабельність – інакше інформативність такої моделі буде дуже низькою. На рисунку 5.2 наведено приклад побудови короткої консенсусної послідовності.

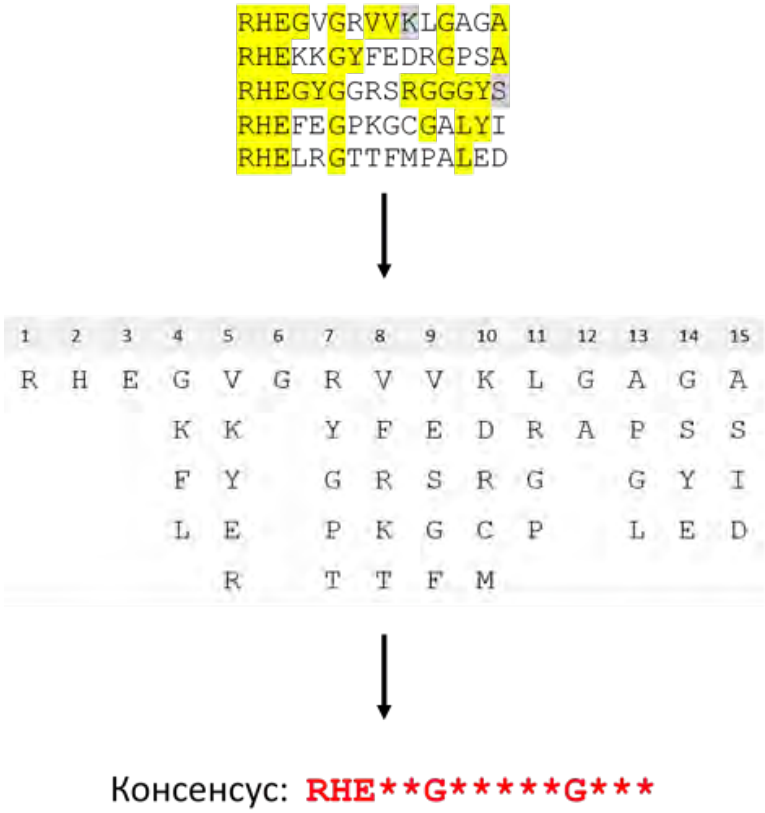


Рис. 5.2. Консенсусна послідовність, виведена у результаті вирівнювання п’яти послідовностей (зірочками (*) позначені невизначені позиції – ті, в яких трапляється більше, ніж один залишок)

Консенсусні послідовності легко будувати, їх можна застосовувати для простих завдань, таких як пошук висококонсервативних ділянок, наприклад, сайтів розпізнавання для ендонуклеаз рестрикції. Консенсусні послідовності не є зручним і потужним інструментом аналізу НАП: вони не містять інформації про варіабельність позицій, їхня якість дуже залежить від розміру вихідного набору даних (кількості послідовностей, з яких складалося вирівнювання), це своєрідна бінарна система подання даних: у кожній позиції або є певний залишок, або ж повна невизначеність (див. рис. 5.2). Це спонукало до розроблення інших моделей множинних вирівнювань, яким присвячені окремі підрозділи нашого розділу. А саме: у першому підрозділі описані паттерни (від англ. patterns) – якісний підхід до опису мотивів білка. Інші підрозділи дають уявлення про позиційно-специфічні вагові матриці, або профілі, які можна створювати на основі множинних вирівнювань, і деякі способи їхнього застосування.

5.1. Паттерни. Збереження тільки певного амінокислотного залишку (чи незначна варіативність) у деяких ділянках багатьох функціонально подібних білків засвідчує роль цього залишку чи сегмента білка. Це біологічно важливі ділянки – *мотиви*. Зазвичай мотиви невеликі за розміром – приблизно вони становлять 10–20 амінокислотних залишків. Вони необхідні для: *а*) каталізу ферментативної реакції; *б*) посттрансляційних модифікацій (ацетилування, глікозилування тощо, приєднання простетичних груп – гем, біотин,

фосфопантотенат); *в*) координування йонів металів, високомолекулярних кофакторів і лігандів (напр., вітаміни, ATP, 2Fe-2S-комплексів) чи інших білків; *г*) формування дисульфідних містків (залишки цистеїну); *д*) зв'язування з ДНК. Множинне вирівнювання мотивів можна звести до консенсусного виразу, який ще називають *регулярним виразом* (regular expression), або підписом білка (*сигнатура*, від англ. signature), або ж *паттерном* (pattern). Мабуть, найуживанішим у науковій літературі є термін "паттерн". У кожній позиції такого паттерна може міститися будь-який амінокислотний залишок із заздалегідь визначеного списку дозволених залишків, і кожну позицію може бути повторено певну кількість разів. В абсолютно консервативних позиціях дозволена наявність лише одного амінокислотного залишку, у більш варіабельних – амінокислоти, що подібні за фізико-хімічними властивостями. Також можливо визначити, які амінокислотні залишки несумісні з певною позицією паттерна, і консервативні позиції можуть бути розділені розривами різної довжини. Наприклад, множинне вирівнювання метилтрансфераз N6-аденіну ДНК (N6-MTASE) дало змогу виявити мотив розміром сім амінокислотних залишків, які описують паттерном PS00092 (рис. 5.3). Згаданий номер паттерна взяли з бази даних PROSITE (http://prosite.expasy.org), яку створено фахівцями Швейцарського інституту біоінформатики і яка є сайтом, де розробляють, зберігають та оновлюють паттерни і супровідну документацію.

а

```
AATR1_SCHPO/118-124     1 IIHNPPY
ACTL9_BOVIN/147-153     1 LFSDPPF
ACTL9_HUMAN/147-153     1 LFSDPPF
ACTL9_MOUSE/146-152     1 LFSDPPF
ACTL9_RAT/142-148       1 LFTDPPF
AML1_YEAST/157-163      1 LLIDPPF
AOFH_MYCBO/6-12         1 AVTNPPW
AOFH_MYCTU/6-12         1 AVTNPPW
APEA_THEMA/330-336      1 AAVNPPY
APEA_THENE/331-337      1 AAVNPPY
AROB_MYCBO/12-18        1 VAVDPPY
AROB_MYCLE/12-18        1 VAVDPPY
AROB_MYCPA/12-18        1 VAVDPPY
AROB_MYCTU/12-18        1 VAVDPPY
ARPC5_CAEEL/52-58       1 VLLNPPF
AUR3_ARATH/211-217      1 LYGNPPF
AURKB_BOVIN/267-273     1 LVGNPPF
AURKB_DANRE/243-249     1 LVGNPPF
AURKB_PIG/267-273       1 LVGNPPF
AURKB_RAT/270-276       1 MVGNPPF
BGAL2_KLEP7/108-114     1 IAVNPPY
BGAL2_KLEPN/108-114     1 IAVNPPY
BOC_HUMAN/1044-1050     1 AVWDPPF
CAND2_MOUSE/963-969     1 VFVNPPY
CAND2_RAT/1001-1007     1 VFVNPPF
```

б

```
[LIVMAC]-[LIVFYA]-{DYP}-[DN]-P-P-[FYW]
```

Рис. 5.3. Множинне вирівнювання мотивів N6-MTASE (*а*) і паттерн, що з нього виведено (*б*). Джерело вирівнювання – PROSITE; розфарбування – BOXSHADE (http://www.ch.embnet.org/software/BOX_form.html)

Паттерн, зображений на рис. 5.3, *б*, містить низку умовних позначень – *синтаксис*, які дають змогу узагальнити множинне вирівнювання. Амінокислотні залишки позначені однолітерним кодом IUPAC. Кожен елемент (позиція) паттерна відокремлений від інших дефісом (-). Символ "x" використовують у позиціях, де може бути розташована будь-яка амінокислота. Неоднозначні позиції позначені квадратними дужками. У прикладі, наведеному нижче, квадратні дужки позначають першу позицію паттерна, в якій може знаходитись один із шести амінокислотних залишків: лейцин (L), ізолейцин (I), валін (V), метіонін (M), аланін (A) або ж цистеїн (C); друга позиція також доволі варіабельна. В третій позиції фігурними дужками позначені несумісні амінокислоти, тобто в цій позиції можуть бути будь-які амінокислоти, за винятком D, Y, P. Повтори елементів зазначено у круглих дужках. Наприклад, [K, T](2,4) означає, що позиція, в якій є лізин (K) або треонін (T), може повторюватися два або чотири рази, x(2) – будь-яка амінокислота двічі. Паттерн можна "заякорити" до N- чи C-кінцевої ділянки білка за допомогою символів "<" чи ">".

Наведемо приклад довшого паттерна (PS00375), знайденого при вирівнюванні множини UDP-глікозилтрансфераз:

[FW]-x(2)-[QL]-x(2)-[LIVMYA]-[LIMV]-x(4,6)-------[LVGAC]-[LVFYAHM]-[LIVMF]-[STAGCM]-[HNQ]---------[STAGC]-G-x(2)-[STAG]-x(3)-[STAGL]-[LIVMFA]--------x(4,5)-[PQR]-[LIVMTA]-x(3)-[PA]-x(2,3)-[DES]-[QEHNR]

Розглянемо початок цього паттерна. Можна сказати, що між позиціями [FW] і [QL] міститься район з двох випадкових амінокислотних залишків – x(2); x(4,6) позначає район з чотирьох або шести випадкових амінокислотних залишків.

Паттерн – це якісний описувач (*дескриптор*; descriptor) певної амінокислотної послідовності. Він або збігається з певним мотивом послідовності – якщо виявлено 100 % ідентичність із нею, або ж ні, навіть якщо розбіжність виявлено в одній позиції. Немає певної порогової величини, вище якої збіг можна вважати статистично достовірним. Однак точність паттерна можна оцінити, якщо визначити кількість послідовностей (з певної бази даних), які його містять. Зазвичай коректніше виведені паттерни матимуть більшу кількість хітів. Паттерни повніше описують білкові мотиви. Для цього потрібно порівняти рис. 5.4 і 5.2, щоб у цьому переконатися. В обидвох випадках проаналізоване одне множинне вирівнювання: на рис. 5.2 подано його консенсусну послідовність, на рис. 5.4 – паттерн.

База даних PROSITE містить пошукове вікно; увівши в нього номер паттерна або ключове слово (назва білка, напр., glycosyltransferase або zinc finger), можна отримати інформацію про паттерни, що асоційовані з номером чи заданим словом. На сервері PROSITE можна задати свою послідовність для пошуку в ній описаних паттернів. Також PROSITE дає змогу генерувати паттерни з набору невирівняних послідовностей (програма PRATT: http://web.expasy.org/pratt/).

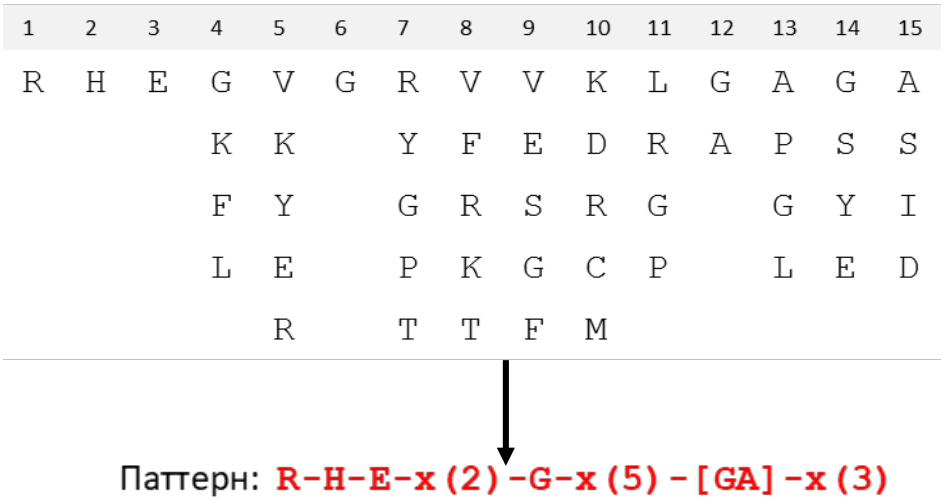


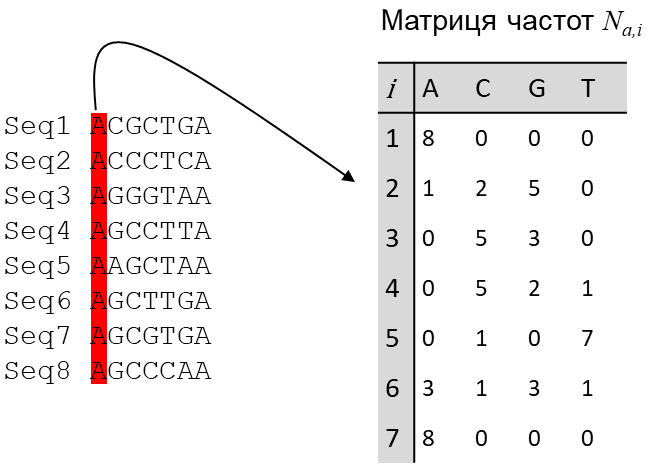
Рис. 5.4. Паттерн, виведений на основі множинного вирівнювання п’яти послідовностей, які зображено на рис. 5.2

Як модель множинного вирівнювання паттерни легко порівнювати з базами даних послідовностей: вони зрозумілі всім, хто має елементарні знання з біохімії. Однак, як і консенсусні послідовності, паттерни непридатні для опису розривів у множинних вирівнюваннях. Короткі паттерни часто дають багато несправжніх позитивних результатів (збігів, які насправді випадкові, не мають біологічного значення), а довгі дуже складно створювати. Тому паттерни найчастіше використовують для пошуку коротких мотивів, що входять до складу активних центрів білків, і для презентації даних широкому загалу науковців.

5.2. Позиційно-специфічні вагові матриці (PSWM): принципи і застосування (WebLogo; скринінг операторів; база консервативних доменів CDD). Матриці амінокислотних заміщень (наприклад, PAM250 чи BLOSUM62) ґрунтуються на припущенні, що частоти заміщень певного амінокислотного залишку (нехай Cys) на інші не залежать від позиції Cys. Іншими словами, є однакова ймовірність заміщення залишку цистеїну на гліцин як на N- так і на С-кінцевій ділянці білка. Тому PAM- і BLOSUM-матриці – *позиційно незалежні*. Хоча припущення про позиційну незалежність справедливе у загальному випадку щодо попарних вирівнювань, воно втрачає зміст у множинних вирівнюваннях споріднених груп послідовностей, де частоти заміщень певних (однотипних) залишків у різних позиціях значно відрізняються. Це спонукало дослідників до розроблення нового класу моделей множинних вирівнювань, які враховують залежність частоти натрапляння на різні залишки від їхньої позиції у вирівнюванні. Ці моделі отримали назву *позиційно специфічні рахункові* (*вагові*) *матриці* (від англ. position-specific scoring (weight) matrices – PSSM/PSWM).

Розглянемо схему побудови PSSM на прикладі вирівнювання восьми олігонуклеотидних послідовностей (seq1 – seq8), що відображено на рис. 5.5. Перший крок – створення таблиці, де підсумовано частоту трапляння ($N_{a,i}$) кожного з чотирьох нуклеотидних залишків у кожній позиції (колонці)

вирівнювання. Далі в отриманій матриці необхідно позбутися нульових значень. Нульове значення засвідчує відсутність певної літери у певній позиції матриці. Таке трапляється за малої кількості послідовностей, які аналізують (скажімо, у розглянутому прикладі їх лише вісім, а якби для вирівнювання було використано 80 послідовностей, то, можливо, в одній з них першу позицію займала б G).

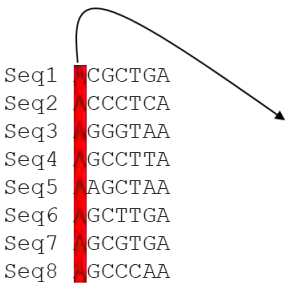


Матриця частот $N_{a,i}$

i	A	C	G	T
1	8	0	0	0
2	1	2	5	0
3	0	5	3	0
4	0	5	2	1
5	0	1	0	7
6	3	1	3	1
7	8	0	0	0

Рис. 5.5. Перший крок побудови PSSM – створення таблиці частот трапляння залишків ($N_{a,i}$) у кожній *i*-й позиції вирівнювання

Крім того, на останньому етапі побудови вагової матриці значення переводять у логарифмічні величини, тому нульові значення даватимуть величину -∞, якою складно далі оперувати. Зважаючи на сказане, в позиційно специфічних матрицях застосовують принцип *псевдорахунків*, який полягає у додаванні до всіх значень частот у матриці невеликого числа, величину якого визначають емпірично (див. розділ 2). Так отримують відкориговані значення частот ($N'_{a,i}$), що фактично є поправкою на малий розмір вибірки послідовностей, які слугують основою для конструювання матриці. Нехай у розглянутому прикладі величина псевдорахунку становитиме 0,25. У результаті додавання цього значення до всіх $N_{a,i}$ буде отримано матрицю $N'_{a,i}$ (рис. 5.6).



Матриця частот $N'_{a,i}$

i	A	C	G	T
1	8,25	0,25	0,25	0,25
2	1,25	2,25	5,25	0,25
3	0,25	5,25	3,25	0,25
4	0,25	5,25	2,25	1,25
5	0,25	1,25	0,25	7,25
6	3,25	1,25	3,25	1,25
7	8,25	0,25	0,25	0,25

Рис. 5.6. Другий крок побудови PSSM – коригування псевдорахунками – створення матриці $N'_{a,i}$

На наступному етапі матрицю частот $N'_{a,i}$ перетворюють у матрицю ймовірностей $P_{a,i}$ – розділяють значення $N'_{a,i}$ на кількість послідовностей у вирівнюванні (у цьому випадку – 8, рис. 5.7, *а*). Значення $p_{a,i}$ у цій матриці відображають імовірності того, що певна послідовність матиме нуклеотид a у позиції i, а сама матриця – вихідний матеріал для створення ймовірнісної моделі множинного вирівнювання, що описує, наприклад, сайт зв'язування певного транскрипційного фактора. Зауважимо, що в описаній далі матриці всі літери мають більшу від нуля ймовірність внаслідок уведення псевдорахунків. Потім іде черговий етап, коли матимемо змогу врахувати різну фонову частоту трапляння кожного залишку. Нехай у розглянутому випадку всі вісім послідовностей seq1-seq8 походять з геному, в якому усі чотири нуклеотиди A, T, G, C трапляються з однаковою частотою (q_a – 25 %, або 0,25). Тоді значення частот розділяють на 0,25 й отримують таблицю співвідношень (часток; odds) фактичної позиційної частоти залишку до його фонової частоти у геномі (рис. 5.7, *б*). Як і у випадку матриць PAM і BLOSUM, доцільно перетворити отримані частки у логарифмічні величини (рис. 5.7, *в*), якими простіше оперувати.

а

Матриця імовірностей $P_{a,i}$

i	A	C	G	T
1	1,03	0,03	0,03	0,03
2	0,16	0,28	0,66	0,03
3	0,03	0,66	0,40	0,03
4	0,03	0,66	0,28	0,16
5	0,03	0,16	0,03	0,91
6	0,40	0,16	0,40	0,16
7	1,03	0,03	0,03	0,03

$\frac{P_{a,i}}{q_a}$

б

Матриця часток (*odds*)

i	A	C	G	T
1	4,12	0,12	0,12	0,12
2	0,64	1,12	2,64	0,12
3	0,12	2,64	1,60	0,12
4	0,12	2,64	1,12	0,64
5	0,12	0,64	0,12	3,64
6	1,60	0,64	1,60	0,64
7	4,12	0,12	0,12	0,12

$s_{a,i} = ln\frac{P_{a,i}}{q_a}$

в

Матриця log-часток - PSSM

i	A	C	G	T
1	1,41	-2,12	-2,12	-2,12
2	-0,44	0,11	0,97	-2,12
3	-2,12	0,97	0,47	-2,12
4	-2,12	0,97	0,11	-0,44
5	-2,12	-0,44	-2,12	1,29
6	0,47	-0,44	0,47	-0,44
7	1,41	-2,12	-2,12	-2,12

Рис. 5.7. Третій, четвертий і п'ятий кроки побудови PSSM – перетворення значень $N'_{a,i}$ у $P_{a,i}$ (*а*), внормування щодо фонових частот (*б*) і перетворення часток у логарифмічні значення (*в*)

За допомогою PSSM у кожній позиції вирівнювання можна передбачити наявність більше одного залишку (нуклеотидного чи амінокислотного), що робить позиційно-специфічні матриці гнучкішим знаряддям пошуку подібних послідовностей у базах даних. Це головна перевага PSSM над консенсусним рядком чи паттерном як спосіб опису множини споріднених послідовностей. Побудова PSSM і/або їхнє вирівнювання проти певного масиву послідовностей є основоположними для великої кількості біоінформатичних підходів та веб-сервісів.

Стосовно нуклеотидних PSSM необхідно розглянути декілька прикладів. Доволі часто дослідникам доводиться вирішувати проблему простого графічного

відображення множини споріднених нуклеотидних послідовностей, наприклад, операторів для певного транскрипційного фактора, нуклеотидної послідовності навколо рибосом-зв'язувального сайта тощо.

Сьогодні послуговуються веб-сервісом під назвою WebLogo (http://weblogo.berkeley.edu/logo.cgi), який перетворює множинне нуклеотидне чи амінокислотне вирівнювання у *логотип* – графічне відображення частоти трапляння чотирьох літер нуклеотидної "абетки" (A, T, G і C) у кожній позиції вирівнювання. Наприклад, якщо завантажити вісім послідовностей, зображених на рис. 5.5, у діалогове вікно WebLogo, то в результаті буде отримано ДНК-логотип цього вирівнювання (рис. 5.8). Зауважимо, що у позиції 6 логотипу висока різноманітність літер, яка спричинює практично нульовий інформаційний вміст цієї позиції.

Рис. 5.8. ДНК-логотип, який відповідає множинному вирівнюванню, зображеному на рис. 5.5

Для побудови логотипу немає потреби в обчислюванні PSSM, а лише у встановленні частот трапляння кожної літери у кожній позиції вирівнювання (перший крок – див. рис. 5.5). У цій моделі (див. рис. 5.8) висоту літер визначають за частотою їхнього трапляння у вихідному вирівнюванні. Зокрема, найчастіше репрезентована літера буде найвищою і знаходитиметься зверху. Верхні літери логотипу, фактично, репрезентують консенсусну послідовність множинного вирівнювання. Вісь ординат (висота літер) у логотипі визначають у бітах інформації (див. рис. 5.8). Зміст цієї системи виміру можна зрозуміти на такому прикладі: щоб обрати один символ чи стан із двох можливих, потрібен один біт інформації (одне питання). А саме: щоб дізнатися, якою стороною випала при підкиданні монета, треба поставити лише одне питання – "це герб?" (відповідь "ні" автоматично означатиме, що монета випала номіналом догори). Якщо послідовність ДНК у певній позиції містить виключно G, то необхідні два біти інформації, оскільки необхідно поставити два питання: "це A чи G?" (чи це пурин) і "чи це A або C?" (якщо відповіді на обидва питання – "ні", то у цій позиції має бути T). Далі, якщо позиція вирівнювання містить дві різні літери з чотирьох можливих (інколи A й інколи G), то достатньо одного питання, щоб дізнатися які це, оскільки вибір двох літер (станів) із чотирьох можливих еквівалентний вибору одного з двох (приклад з монетою). Так, один біт необхідний, щоб описати позицію, що містить пурини, і два – позицію, що містить тільки аденін. Якщо частоти літер не рівноймовірні, то застосовують складніші обчислювання середнього інформаційного вмісту кожної позиції згідно з рівнянням Шеннона (2), що описане у розділі 2. Сумарна інформація в

кожній позиції репрезентується зниженням невизначеності відповідно до того, як локалізується (чи вирівнюється) шукана послідовність (напр., операторна послідовність для транскрипційного фактора):

$$R_{\text{послідовність}}(l) = 2 - \big(H(l) + e(n)\big) \quad \frac{\text{біти}}{\text{позиція}}, \qquad (41)$$

де $R_{\text{послідовність}}(l)$ – інформаційний вміст позиції l; 2 – максимальний рівень невизначеності будь-якої позиції нуклеотидної послідовності; $e(n)$ – поправка, що необхідна тоді, коли вирівнювання складається з малої кількості (n) послідовностей.

Множина значень $R_{\text{послідовність}}(l)$ формує криву, що репрезентує важливість різних позицій у вирівнюванні (послідовності). Висота кривої – це висота логотипу в цій позиції. Висоту літери у надрукованому логотипі визначають у результаті множення частоти цієї літери на сумарний інформаційний вміст позиції, в якій вона є:

$$\text{висота основи } \boldsymbol{b} \text{ у позиції } \boldsymbol{l} = f(b, l) \times R_{\text{послідовність}}(l). \qquad (42)$$

Програма WebLogo аналізує й амінокислотні послідовності. У цьому разі інформаційний вміст позиції амінокислотного вирівнювання обчислюють аналогічно:

$$R_{\text{послідовність}}(l) = \log_2 20 - \big(H(l) + e(n)\big) \quad \frac{\text{біти}}{\text{позиція}}. \qquad (43)$$

Отже, матриця частот – вихідна точка конструювання PSSM – може слугувати для стислого опису певної послідовності, як це бачимо з прикладу WebLogo; матрицю частот $P_{a,i}$ (див. рис. 5.7, *а*) можна використовувати для визначення ймовірності виявлення певного мотиву у випадковій послідовності.

Сьогодні наявні інші веб-сервіси та програми для побудови логотипів, які удосконалюють графічну репрезентацію результату, чи додають до логотипу нові елементи (наприклад, планки похибок до частот трапляння символів тощо). Перелік деяких із цих сервісів можна знайти за цим посиланням: http://users.fred.net/tds/lab/paper/logopaper/.

Однак найважливішою галуззю застосування PSSM є пошук мотивів, подібних до тих, що описують цією PSSM у заданих послідовностях. Принципову схему використання PSSM з цією метою стисло описано нижче. Нехай задано певну 9-нуклеотидну (нт) послідовність (рис. 5.9, *а*), у межах якої потрібно знайти 7-нт підпослідовність, яка найбільше нагадує ті, з яких побудована PSSM (див. рис. 5.8).

а

T	A	G	C	C	T	A	A	G	?

б

	1	2	3	4	5	6	7
A	1.41	-0.44	-2.12	-2.12	-2.12	0.47	1.41
C	-2.12	0.11	0.97	0.97	-0.44	-0.44	-2.12
G	-2.12	0.97	0.47	0.11	-2.12	0.47	-2.12
T	-2.12	-2.12	-2.12	-0.44	1.29	-0.44	-2.12

T	A	G	C	C	T	A	A	G	S = -0.59

в

	1	2	3	4	5	6	7
A	1.41	-0.44	-2.12	-2.12	-2.12	0.47	1.41
C	-2.12	0.11	0.97	0.97	-0.44	-0.44	-2.12
G	-2.12	0.97	0.47	0.11	-2.12	0.47	-2.12
T	-2.12	-2.12	-2.12	-0.44	1.29	-0.44	-2.12

T	A	G	C	C	T	A	A	G	S = 7.49

г

	1	2	3	4	5	6	7
A	1.41	-0.44	-2.12	-2.12	-2.12	0.47	1.41
C	-2.12	0.11	0.97	0.97	-0.44	-0.44	-2.12
G	-2.12	0.97	0.47	0.11	-2.12	0.47	-2.12
T	-2.12	-2.12	-2.12	-0.44	1.29	-0.44	-2.12

T	A	G	C	C	T	A	A	G	S = -5.25

Рис. 5.9. PSSM як знаряддя пошуку операторних сайтів у заданих послідовностях (*а*). Матрицю вирівнюють у всіх можливих 7-нт вікнах із заданою послідовністю і кожного разу визначають рахунок вирівнювання (суму рахунків/ваг усіх позицій матриці виділили зеленим кольором; жовтим виділені ті значення матриці, що збігаються з літерою послідовності)

Для цього матрицю послідовно вирівнюють проти усіх можливих 7-нт послідовностей (“вікон”) у межах заданої, тобто пересуваються по одній літері від початку до кінця (див. рис. 5.9, *б-г*). Кожного разу визначають рахунок вирівнювання матриці із послідовністю. Тут використано матрицю, що й на рис. 5.8, однак її для зручності розвернули на 90° проти годинникової стрілки. Помітно, що 7-нт підпослідовність, яка розпочинається другою позицією, має найбільший рахунок, тобто вона найподібніша до послідовності, яку описує використана модель – PSSM. Аналогічно скринінгують геноми для виявлення всього набору послідовностей, описаного певними PSSM. На відміну від модельного випадку, у якому одна із підпослідовностей має значно вищий рахунок, ніж інші (див. рис. 5.9), скринінг реальних геномних послідовностей повертає великий список підпослідовностей, що матимуть доволі наближені величини *S*. Наприклад, операторний сайт транскрипційного фактора AdpA описують доволі простою консенсусною послідовністю 5′-TGGCSNGWWY-3′ (неоднозначні позиції подані згідно з номенклатурою IUBMB: S – G/C; W – A/T; Y – C/T; N – A/C/G/T). PSSM, створена на основі експериментально перевірених операторних сайтів AdpA, дає змогу виявити у геномі продуцента стрептоміцину

Streptomyces griseus близько 40 тисяч подібних сайтів, але очевидно, що не всі вони насправді розпізнаються AdpA. Дослідникові необхідно встановлювати певне порогове значення S_c для того, щоб звузити коло пошуку до сайтів, які матимуть $S > S_c$. Вибір величини значення S_c – емпіричний, тут потрібно керуватися наявною інформацією про біологічну роль досліджуваного транскрипційного фактора, ймовірну кількість підконтрольних генів тощо. Як і у попарних вирівнюваннях, тут також можна обчислювати величину очікування *E*.

PSSM також використовують для опису множини споріднених амінокислотних послідовностей. Принцип побудови амінокислотних PSSM не відрізняється від такого для нуклеотидних – необхідно знати фактичні частоти трапляння 20 літер амінокислотної абетки у кожній позиції множинного вирівнювання, а також значення фонових частот цих літер – зазвичай беруть величини з матриць BLOSUM, PAM тощо (рис. 5.10).

а

G H E G V G K V V K L G A G A
G H E K K G Y F E D R G P S A
G H E G Y G G R S R G G G Y S
G H E F E G P K G C G A L Y I
G H E L R G T T F MP A L E C

б

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	0	0	0	0	0	0	0	0	0	0	0	2	1	0	2
C	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
D	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	5	0	1	0	0	0	1	0	0	0	0	1	0
F	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
G	5	0	0	2	0	5	1	0	1	0	2	3	1	1	0
H	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
K	0	0	0	1	1	0	1	1	0	1	0	0	0	0	0
L	0	0	0	1	0	0	0	0	0	0	1	0	2	0	0
M	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0
S	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
T	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
V	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0

Рис. 5.10. Матриця частот (*б*), що описує множинне вирівнювання амінокислотних послідовностей (*а*). Червоним позначено позицію матриці G1. Аналогічну матрицю частот для нуклеотидних послідовностей наведено на рис. 5.5

В амінокислотні PSSM також уводять псевдорахунки. Тут вони навіть важливіші, ніж у нуклеотидних, зважаючи на значно більший обсяг білкової абетки. Псевдорахунки, у разі амінокислотних послідовностей, можна обчислити так. Значення ймовірностей $f_{a,j}$ є відношенням частоти трапляння певної літери у позиції до кількості послідовностей у вирівнюванні ($f_{G,1} = 5/5 = 1$); тоді значення $f_{G,1}$, відкориговане псевдорахунком ($f'_{G,1}$), матиме вигляд:

$$f'_{G,1} = (5+1)/(5+20) = 0{,}24\ , \qquad (44)$$

де число 1 у чисельнику – величина псевдорахунку, доданого до частоти трапляння, а число 20 у знаменнику – це обсяг амінокислотної абетки білків, оскільки вони збудовані на основі 20-ти біогенних амінокислот. Зазначимо, що також застосовують складніші методи обчислення псевдорахунків, що базуються на матрицях заміщення або баєзіанських підходах, наприклад, сумішах розподілів Діріхле (Dirichlet mixtures, див. блок 9); один приклад розглянемо, аналізуючи програму PSI-BLAST.

Білкові PSSM є основою низки баз даних, які описують групи споріднених білків чи консервативних доменів. Як приклад розглянемо Conserved Domains Database (CDD: http://www.ncbi.nlm.nih.gov/Structure/cdd/cdd.shtml), що є підрозділом NCBI. CDD – це колекція множинних вирівнювань і PSSM, що виводяться з них. Для множинних вирівнювань у цій базі даних використовують, зазвичай, добре вивчені амінокислотні послідовності, що становлять структурно і функціонально автономні одиниці – домени, для яких часто відома просторова структура. PSSM обчислені на основі цих вирівнювань, отож є моделлю певного домену. На відміну від багатьох інших баз даних подібного спрямування, CDD класифікує отримані моделі у вигляді певних ієрархічних систем, щоб відобразити еволюційні та функціональні зв'язки між різними доменами. Крім того, CDD відображає посилання на просторову структуру доменів, якщо така відома.

Базу CDD можна використовувати по-різному. Наприклад, у наукових статтях, присвячених описові нових білків, часто можна натрапити на твердження, що якийсь білок містить консервативний домен cd00778. Цей номер – код доступу до моделі певного консервативного домену у CDD (рис. 5.11). Увівши номер у діалоговое вікно стартової сторінки CDD (рис. 5.11, *а*), можна отримати вичерпну інформацію про його структуру і функцію (рис. 5.11, *б*).

Інший варіант використання CDD полягає у можливості пошуку консервативних доменів у межах заданої амінокислотної послідовності. Принцип пошуку полягає у попарному вирівнюванні заданої послідовності з усією колекцією PSSM, що становлять базу CDD. Таке вирівнювання виконує програма CD-Search (рис. 5.12), у якій для цього є алгоритм BLAST. Як BLAST, CD-Search обчислює статистичні параметри (S, E) для вирівнювань і сортує їх від найбільш до найменш значущих.

Сторінку результатів CD-Search відображено на рис. 5.13 (підрозділ 5.3). Вона складається з двох панелей – графічного підсумку пошуку й таблиці, що описує знайдені у послідовності консервативні домени, які збігаються з певною моделлю в CDD (matches). Задану послідовність зображено як сірий прямокутник. Головним елементом графічного підсумку є "специфічні хіти" (specific hits) – ділянки заданої дослідником послідовності, рахунки вирівнювання яких з певними, перевіреними в NCBI, моделями (PSSM) перевищують встановлене порогове значення (специфічне для кожної моделі). За порогові значення беруть мінімальні біт-рахунки, які спостерігали для послідовностей, з яких, власне, і побудована модель. Іншими словами, якщо домен, виявлений у заданій послідовності, позначено як специфічний хіт, то це означає, що задана послідовність має рахунок вирівнювання з моделлю такий, як одна чи декілька послідовностей, з яких побудована сама модель. Це жорсткий критерій відбору, що дає змогу з високою вірогідністю виявляти консервативні домени у межах заданої послідовності.

БЛОК 9. Суміш Діріхле (більше на https://compbio.soe.ucsc.edu/dirichlets/).
Нехай у позиції φ вирівнювання трьох амінокислотних (*ак*) послідовностей – три залишки Leu. На основі такої малої вибірки не можна виключити ймовірність появи у позиції φ (при додаванні нових гомологічних послідовностей) інших залишків, наприклад, Ile і Val, що часто заміщують Leu у районах β-складок. Отже, оцінювання очікуваного розподілу ймовірностей появи різних *ак* залишків у цій позиції має включати як Leu (дані), так і Ile й Val (передбачення), і, можливо, інші залишки. Тепер розглянемо вирівнювання 100 послідовностей і позицію φ, в якій тепер 100 залишків Leu. Це вагомий доказ того, що залишок Leu дійсно консервативний; буде хибно узагальнювати цю позицію у вигляді розподілу ймовірностей, який включатиме інші залишки. Тут треба надати менше значення попереднім знанням, більше – актуальним даним. Отже, за нестачі даних, оптимальна статистична модель мала б більше спиратись на попередні знання про частоти трапляння *ак*, а за наявності великого обсягу даних – більше зважати на дані. *Суміші Діріхле* (DM) володіють усіма описаними властивостями.

Щоб зрозуміти DM, розглянемо спочатку *щільність Діріхле* ρ. Це щільність імовірності всіх векторів імовірності $\vec{p}$ ($p_i \geq 0$ і $\sum_i p_i = 1$) у межах певного мультиноміального простору. Тут $p_i = P(ак_i)$. Щодо білків, які містять 20 різних *ак*, маємо справу з 20-компонентним вектором у 19-вимірному просторі (з L параметрів лише L-1 незалежні). Кожен вектор $\vec{p}$ – це можливий розподіл імовірностей 20 *ак*. Щільність Діріхле має параметри $\vec{\alpha} = \alpha_1 \ldots \alpha_{20}$ для $\alpha_i > 0$. Тоді величина щільності вектора $\vec{p}$ – це

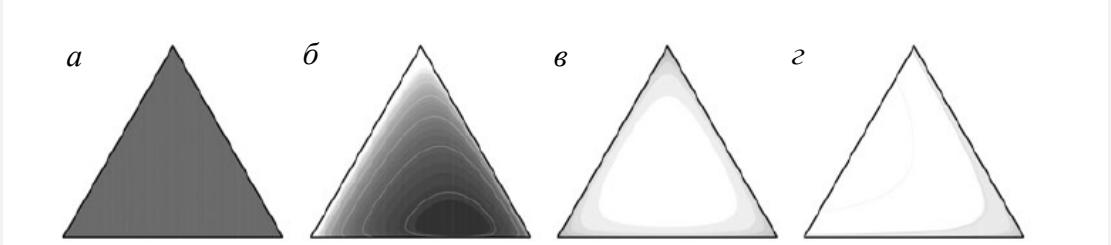
$$\rho(\vec{p}) = Z \prod_{i=1}^{20} p_i^{\alpha_i - 1}, \quad \text{де} \quad Z = \Gamma(\sum \alpha_i) / \prod \Gamma(\alpha_i). \qquad (9.1)$$

Тут Z – нормалізувальна константа, необхідна, аби щільність за всіма векторами зводилася до одиниці. Іншими словами, це функція гамма-розподілів (див. блок 4) параметрів α_i. Фактично щільності Діріхле – це апріорний розподіл імовірностей у мультиноміальному просторі; у межах теорії Баєза (див. блок 2 і розділ 2) ці величини, після отримання даних, перетворюються в апостеріорний розподіл. Якщо всі параметри $\alpha_i = 1$, то він має однорідну щільність по всьому простору (рис. *а* внизу); для нього можна визначити концентраційний параметр $\alpha = \sum \alpha_i$. і центр маси розподілу $\vec{p} = \vec{\alpha}/\alpha$.

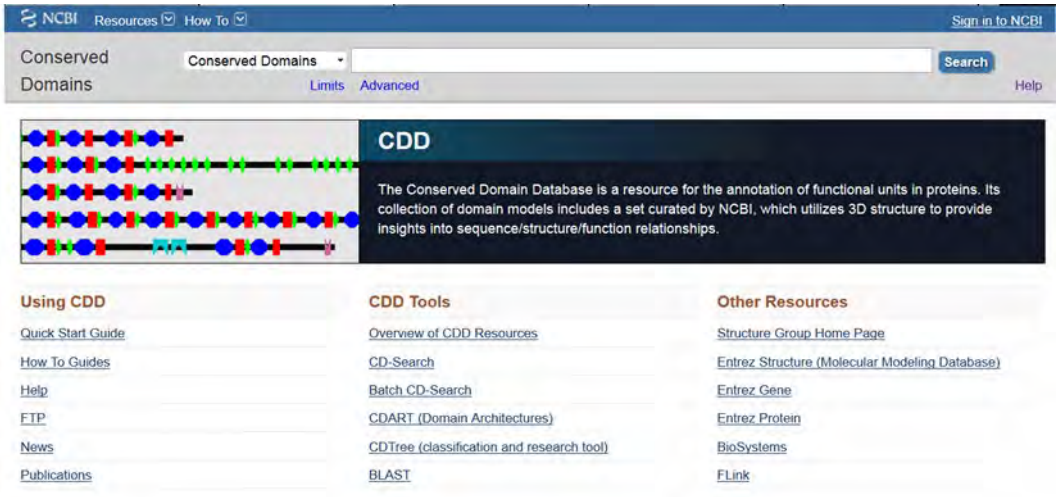
Розподіл Діріхле має один пік щільності ймовірності у просторі. Це не підходить до оцінювання множинних вирівнювань, де є ймовірність заміщення певної *ак* не на один, а на декілька різних *ак* залишків. Тут застосовують DM – набір індивідуальних щільностей Діріхле, які разом приписують розподілам імовірності. Кожну окрему щільність у суміші називають *компонентом*. Для будь-якого розподілу *ак* DM загалом приписує ймовірність розподілу, використовуючи зважену комбінацію ймовірностей за умови заданого розподілу кожного компонента суміші. Ці ваги називають *коефіцієнтами суміші*. Тоді щільність DM ρ з l компонентів буде:

$$\rho = q_1\rho_1 + \cdots + q_l\rho_l \ , \qquad (9.2)$$

де ρ_j – щільність Діріхле, визначена параметрами $\vec{\alpha}_j = (\alpha_{i,1} \ldots \alpha_{ij,20})$, $q_1 \ldots q_l$ – коефіцієнти суміші, щоб звести суму до одиниці. Параметри і коефіцієнти DM обчислюють із високоякісних вирівнювань. Приклади тривимірних DM подано нижче. Параметри α такі: 1, 1, 1 (*а*); 1000, 100, 500 (*б*); 0,1, 0,1, 0,1 (*в*); 0,1, 0,001, 0,001 (*г*). Якщо $\alpha > 1$, то щільність імовірності концентрується навколо середнього значення q. Якщо $\alpha < 1$, то щільність концентрується коло країв і кутів простору.



а



б

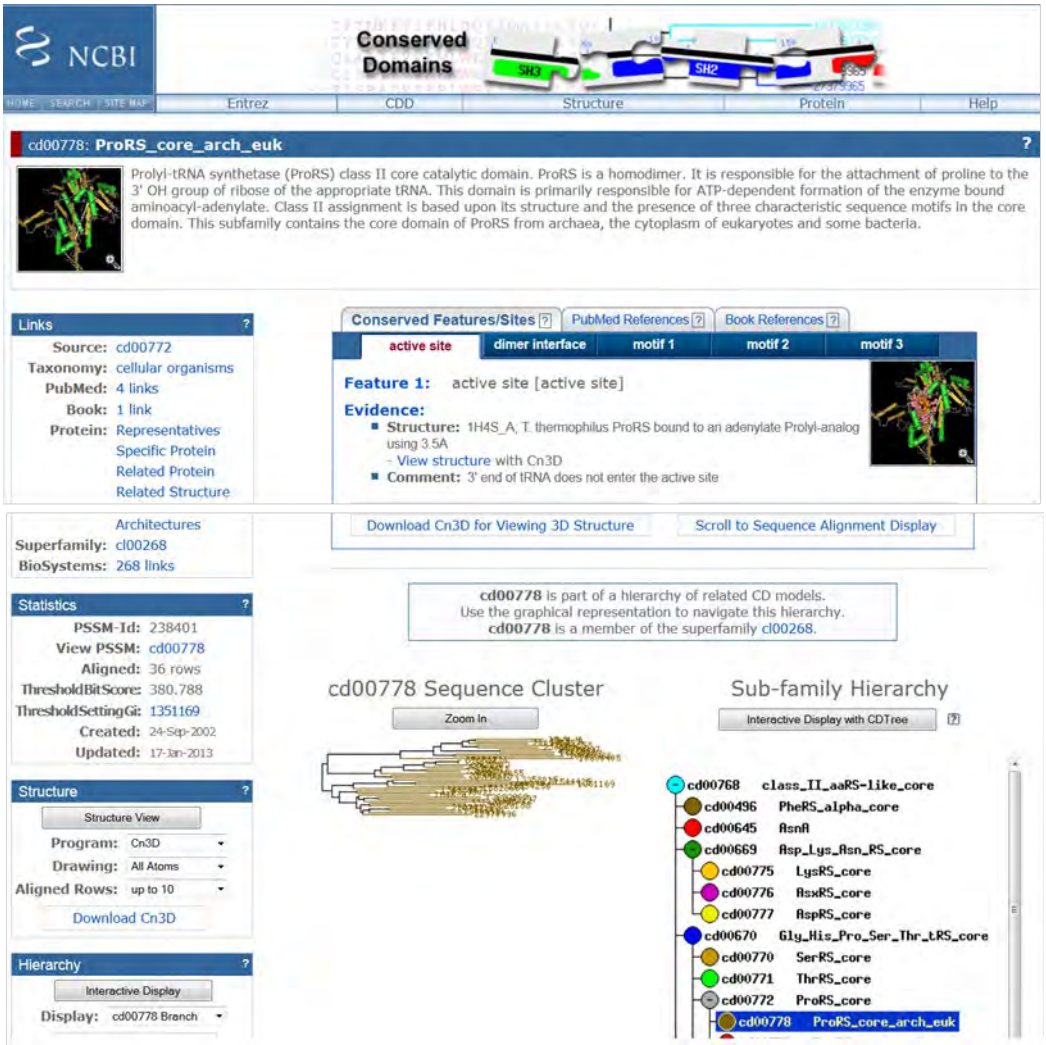


Рис. 5.11. Стартова сторінка бази CDD (*а*) і результат пошуку cd00778 (*б*)

5.3. PSI-BLAST. Попарне вирівнювання PSSM з НАП майже повністю аналогічне вирівнюванню двох послідовностей. Початкова сторінка такого вирівнювання – на рис. 5.12, результат – рис. 5.13. Відмінність полягає у тому, що рахунок вирівнювання літери послідовності до комірки матриці визначається саме PSSM, її позиційно специфічними значеннями (див. рис. 5.9), а не заздалегідь заданою матрицею заміщення, такою як PAM чи BLOSUM.

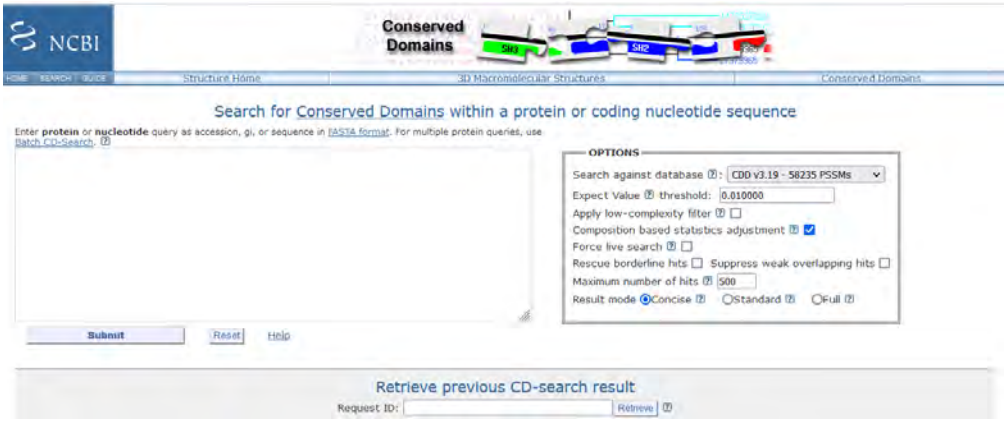


Рис. 5.12. Стартова сторінка CD-Search (білкову послідовність завантажують у діалогове вікно)

Для білків довжиною L матриця заміщень розміром 20×20 літер замінюється на PSSM розміром L×20. У таких матрицях також можна закласти значення штрафів за розриви. Як і в попарному вирівнюванні послідовностей, можна виконувати глобальні та локальні вирівнювання послідовностей до PSSM.

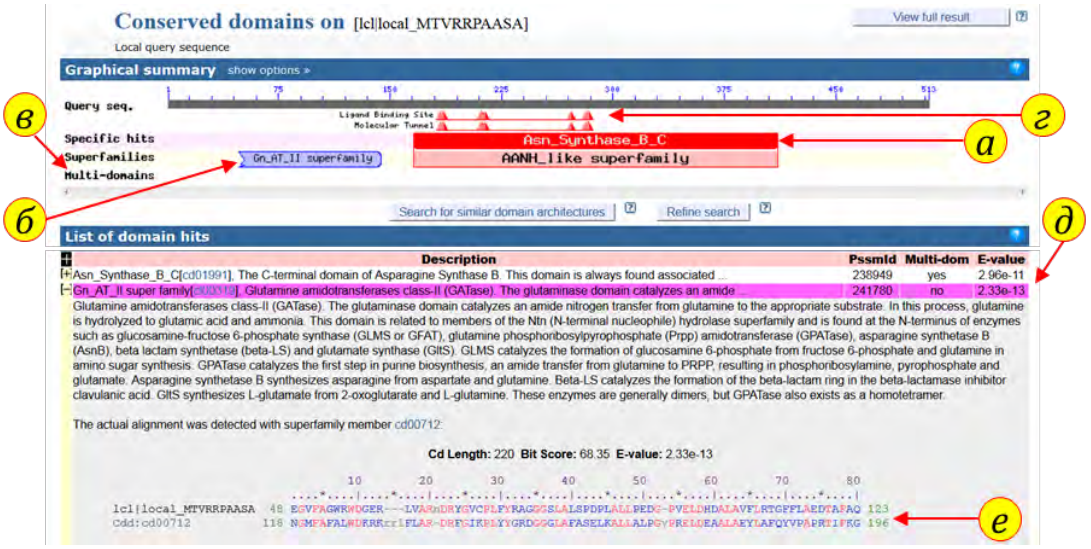


Рис. 5.13. Сторінка CD-Search (стислий формат), що з'являється у відповідь на завантаження у діалогове вікно (див. рис. 5.12) амінокислотної послідовності MoeH5. Специфічні хіти (*а*) позначені прямокутниками яскравого кольору. Якщо немає специфічних хітів до заданої послідовності, то далі наведені інші класифікатори – передусім надродини доменів (superfamilies), до складу яких входять специфічні хіти. Надродини позначені пастельними кольорами. Якщо збіг між заданою послідовністю і доменом (моделлю) неповний, як у випадку (*б*), то край чи краї такого домену зрізані. Далі йдуть мультидоменні класифікатори (*в*), яких немає у цьому разі. Функціональні сайти у межах доменів зображені трикутниками під заданою послідовністю (*г*), вони походять із найподібнішої моделі, у цьому випадку – специфічного хіта. У таблиці результатів подані величина очікування (*д*) та попарне вирівнювання заданої послідовності із моделлю (*е*)

Є дві переваги вирівнювання PSSM до послідовності. Перша полягає в покращеному оцінюванні імовірностей, з якою амінокислоти репрезентовані у різних позиціях послідовності, що веде до вдосконаленої системи рахунків. Друга – відносно точне визначення меж важливих мотивів, які намагаються вирівняти, що значно зменшує простір пошуку і кількість випадкових вирівнювань. Ці переваги втілені в алгоритмі PSI-BLAST (Position-Specific Iterative BLAST), що доступний на сторінці NCBI.

Концептуальну схему PSI-BLAST наведено на рис. 5.14. Задану амінокислотну послідовність попарно вирівнюють до всієї бази GenBank, послуговуючись алгоритмом blastp. Список хітів з $E < 0{,}01$ використовують для побудови PSSM. Отриману PSSM повторно вирівнюють проти бази GenBank (чи іншої обраної бази даних), у результаті чого отримують нові хіти, які не виявлені у першому раунді BLASTP-опосередкованого пошуку (із застосуванням матриці заміщень BLOSUM62). Нові хіти залучають до побудови нової PSSM, яку використовують у черговому раунді PSI-BLAST. На певному етапі спостерігаємо *конвергенцію* – списки хітів останнього і передостаннього раундів PSI-BLAST не відрізняються. Це засвідчує вичерпання пошукових можливостей алгоритму і зупиняє роботу програми.

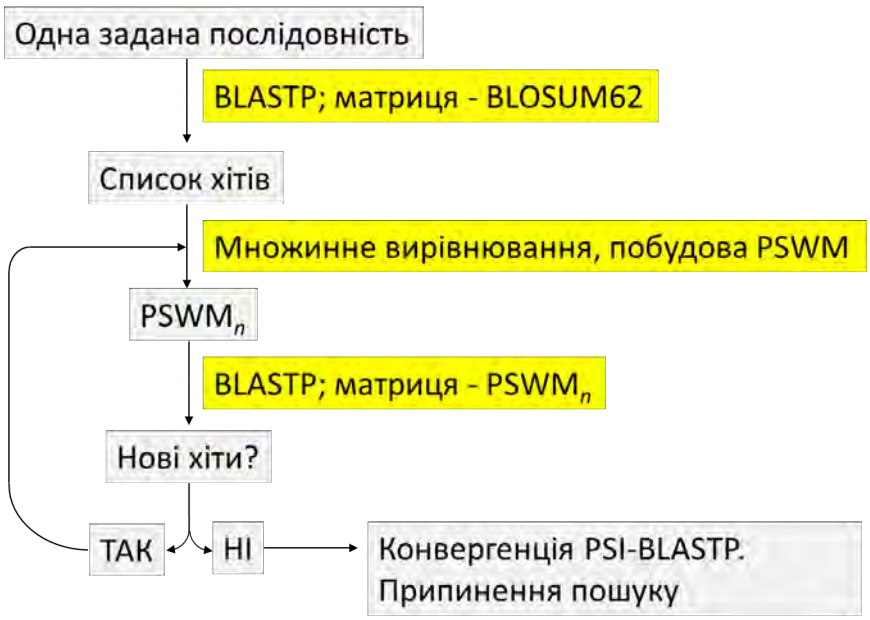


Рис. 5.14. Принцип функціонування алгоритму PSI-BLAST

При побудові PSSM із множинного вирівнювання хітів було б помилково надавати однакову вагу всім послідовностям. Очевидно, що хіти, здебільшого, будуть близькоспоріднені і нестимуть не більше інформації, ніж одна віддалена послідовність; однак “близькоспоріднена” більшість, унаслідок свого розміру, переважить невелику кількість більш дивергентних послідовностей. Для уникнення цього близьким послідовностям приписують меншу вагу, віддаленим – більшу. Процедуру зважування послідовностей згадували під час розгляду алгоритмів множинного вирівнювання (див. рис. 4.4 і 4.5). Для потреб

PSI-BLAST використовують процедуру зважування, яку 1994 р. вперше описали С. Генікоф та Х. Генікоф. Розгляньмо її на прикладі такого множинного вирівнювання:

GYVGS
GFDGF
GYDGF
GYQGG

Простим методом репрезентації ступеня різноманітності амінокислотних залишків у позиції вирівнювання може стати присвоєння однакової частки ваги кожному відмінному залишкові, а потім розділення цієї ваги однаково між усіма послідовностями, що мають цей амінокислотний залишок. Тобто: якщо у певній позиції множинного вирівнювання є r різних залишків, то залишок, що міститься тільки в одній послідовності, матиме внесок $1/r$ до ваги цієї послідовності. Водночас залишок, який міститься у s послідовностях, матиме внесок $1/r \times s$ до ваги кожної з s послідовностей. Внески всіх позицій вирівнювання додають і в такий спосіб отримують вагу послідовності (рис. 5.15).

а

Позиційно-специфічні ваги залишків

Залишок	Позиція				
	1	2	3	4	5
G	1/(1×4)			1/(1×4)	1/(3×1)
Y		1/(2×3)			
F		1/(2×1)			1/(3×2)
V			1/(3×1)		
D			1/(3×2)		
Q			1/(3×1)		
K					1/(3×1)

$1/r \times s$

б

Позиційно-специфічні ваги послідовностей

Посл.	Позиція						
	1	2	3	4	5	Сума	Норм.
GYVGK	1/(1×4)	1/(2×3)	1/(3×1)	1/(1×4)	1/(3×1)	4/3	0.267
GFDGF	1/(1×4)	1/(2×1)	1/(3×2)	1/(1×4)	1/(3×2)	4/3	0.267
GYDGF	1/(1×4)	1/(2×3)	1/(3×2)	1/(1×4)	1/(3×2)	3/3	0.200
GYQGG	1/(1×4)	1/(2×3)	1/(3×1)	1/(1×4)	1/(3×1)	4/3	0.267
Разом	1	1	1	1	1	5	1,001

Рис. 5.15. Приклад підрахунку ваг послідовностей за Генікоф та Генікоф (1994). Вагу окремих амінокислотних залишків у різних позиціях (*а*) обчислюють за формулою $1/r \times s$, де r – кількість різних залишків у позиції, s – кількість послідовностей, що містить певний залишок. Вагу послідовностей (*б*) обчислюють як суму ваги залишків у позиціях. Нормалізовану вагу послідовності (виділено жовтим для першої послідовності) визначають як співвідношення кількості позиції до ваги послідовності (ці дві величини виділені зеленим кольором)

У PSI-BLAST застосовують модифікований підхід, а саме: розриви у вирівнюванні розцінюють як 21-шу літеру амінокислотної абетки, а вираховуючи вагу послідовності, не залучають позиції вирівнювання, що містять лише один амінокислотний залишок. Для побудови позиційно-специфічної матриці для потреб PSI-BLAST важливо знати не тільки цільові частоти залишків у позиції, а й число незалежних подій (залишків) N_c, що вона містить. Як N_c у PSI-BLAST використовують середню кількість різних амінокислотних залишків, яку спостерігають у різних колонках (позиціях) множинного вирівнювання. Це не ідеальний спосіб, оскільки у великих вирівнюваннях значення N_c досягатиме стану насичення – максимальної величини для білків (21 у цьому випадку – 20 амінокислот + 1 символ, що позначає розрив). Але, як засвідчує практика, у нормі N_c набуває менших значень; крім того, важливі не абсолютні значення N_c, а те, як вони змінюються від позиції до позиції. Тому вищеописаний підхід залишається практично корисним.

Завдяки побудові множинного вирівнювання, отримавши матриці частот трапляння літер, перетворюють їх у матрицю зважених рахунків (вагову матрицю). Як уже зазначено (див. рис. 5.7 і супровідний текст), це можна зробити за допомогою визначення логарифма співвідношення цільової й очікуваної частот трапляння літери – $\log(Q_i/P_i)$. У вирівнюваннях, що складаються з великої кількості незалежних послідовностей, значення Q_i певної позиції – це функція цільової частоти літери i у цій позиції. Однак необхідно врахувати вплив ще кількох чинників на величину Q_i – це вага послідовності, малий розмір вибірки і невипадковість трапляння різних амінокислотних залишків у білках. Фактор трапляння різних амінокислотних залишків можна врахувати за допомогою введення псевдорахунків (див. рис. 5.6 і супровідний текст). У PSI-BLAST їх обчислюють на основі певної матриці заміщень (наприклад, чим менший рахунок заміщення певної амінокислоти у BLOSUM62, тим частіше ця амінокислота трапляється). Псевдорахунок g_i для певної позиції C обчислюють за формулою:

$$g_i = \sum_j \frac{f_j}{P_j} q_{ij} \,, \tag{45}$$

де q_{ij} – цільова частота заміщення літери i літерою j, що неявно очевидно з матриці рахунків (PAM, BLOSUM тощо):

$$q_{ij} = R_i R_j e^{\lambda_u s_{ij}} \,, \tag{46}$$

де f_j – частота натрапляння на j; P_j – очікувана (фонова) частота натрапляння на j у вирівнюванні; R_i і R_j – цільові частоти залишків i та j, що неявно очевидно з матриці заміщення; s_{ij} – рахунок з матриці заміщення, застосованої до вирівнювання i та j; λ_u – поправка до матриці заміщення для вирівнювань без розривів. Очікуємо, що ті амінокислотні залишки, які вважають “популярними” у використаній матриці (мають малі додатні значення) і які вирівнюються із залишками, що фактично спостерігають у вирівнюванні, матимуть високі значення псевдорахунків. Тоді Q_i можна обчислити за формулою:

$$Q_i = \frac{\alpha f_i + \beta g_i}{\alpha + \beta}\ , \qquad (47)$$

де f_i – зважена частота трапляння літери i, α і β – відносні ваги, які приписують фактично отриманій і фоновій частотам натрапляння на цю літеру. Значення α дорівнює $N_c - 1$; значення β – емпіричне і в PSI-BLAST його вважають рівним 10 (за умови, що α визначають як описано попередньо). Чим більша величина β, тим більшу вагу надають попереднім даним (матриці заміщень) про взаємозв'язки між амінокислотними залишками. Отож рахунок для залишку i, розглянутий у PSSM, обчислюють за формулою, аналогічною тій, яку використовували для конструювання нуклеотидних PSSM (див. рис. 5.7 і супровідний текст):

$$\frac{1}{\lambda_u} \ln \frac{Q_i}{P_i}. \qquad (48)$$

Отже, PSSM (PSWM) відкривають ширші можливості для аналізу генетичних послідовностей – як нуклеотидних, так і амінокислотних. PSSM, як концентрована модель множинного вирівнювання послідовностей, які репрезентують певний домен, є у центрі бази даних і веб-сервісу CDD, а також пошукового алгоритму PSI-BLAST. PSSM також широко застосовують для ідентифікації операторних ділянок транскрипційних факторів, тобто основне застосування PSSM – це ідентифікація високоваріабельних послідовностей, що мають сталий розмір. Результати такого пошуку нескладно тлумачити, оскільки для вирівнювань PSSM з генетичними послідовностями можна застосовувати статистичну теорію, розроблену для попарних вирівнювань послідовностей. PSSM як моделі генетичних послідовностей мають два обмеження. По-перше, принципи конструювання і використання PSSM не передбачають уведення інсерцій і делецій; по-друге, відносно довгі послідовності складно або неможливо репрезентувати у вигляді PSSM.

Для конструювання PSSM на основі амінокислотних послідовностей можна скористатися низкою веб-сервісів. Зокрема, на біоінформатичному веб-сервері Mobyle Pasteur розміщена програма Prophecy (http://mobyle.pasteur.fr/cgi-bin/portal.py#forms::prophecy), що дає змогу з набору послідовностей у FASTA-форматі побудувати PSSM. Так само можна скористатися сервісом CHAPS (http://fasta.bioch.virginia.edu/fasta_www2/chaps.cgi) на сервері університету Вірджинії. У цьому випадку вихідним матеріалом для побудови PSSM слугує множинне вирівнювання. На момент написання цього розділу не було веб-сервісів для конструювання нуклеотидних PSSM. Наявна програма motifBS, написана на програмній мові Perl, її потрібно інсталювати на комп'ютер (http://compbio.berkeley.edu/people/ed/motifBS.html). Побудову PSSM можна виконати на базі спеціалізованих пакетів біоінформатичних програм, зокрема UGENE (вільнодоступна; треба інсталювати на комп'ютер).

5.4. Прості і генералізовані профілі. Генералізовані профілі можна означити як позиційно-специфічні вагові матриці, що дають змогу вводити

розриви для відображення таких подій, як інсерції та делеції нуклеотидних чи амінокислотних залишків. Як і PSSM, генералізовані профілі можна вирівнювати між собою і з генетичними послідовностями. Кожна позиція генералізованого профілю (як і PSSM, будують на основі множинного вирівнювання) має назву стану збігу (M; match state). Стан M можна оминути у вирівнюванні, і це називають станом делеції (D; deletion state), який матиме свою позиційно-специфічну величину штрафу. Між двома сусідніми станами M–M чи D–D може виникнути новий стан інсерції (**I**; insertion state), що також матиме свою величину штрафу. Перехід між будь-якими двома сусідніми станами матиме своє значення (рахунок). Кілька додаткових параметрів визначають поведінку кінцевих ділянок вирівнювання, залежно від того, який характер має вирівнювання: локальний чи глобальний. Загальну схему конструювання генералізованого профілю, за якою функціонує веб-сервер PROSITE, відображено на рис. 5.16. В основу методу покладено множинне вирівнювання консервативних доменів, що походять зі споріднених білків.

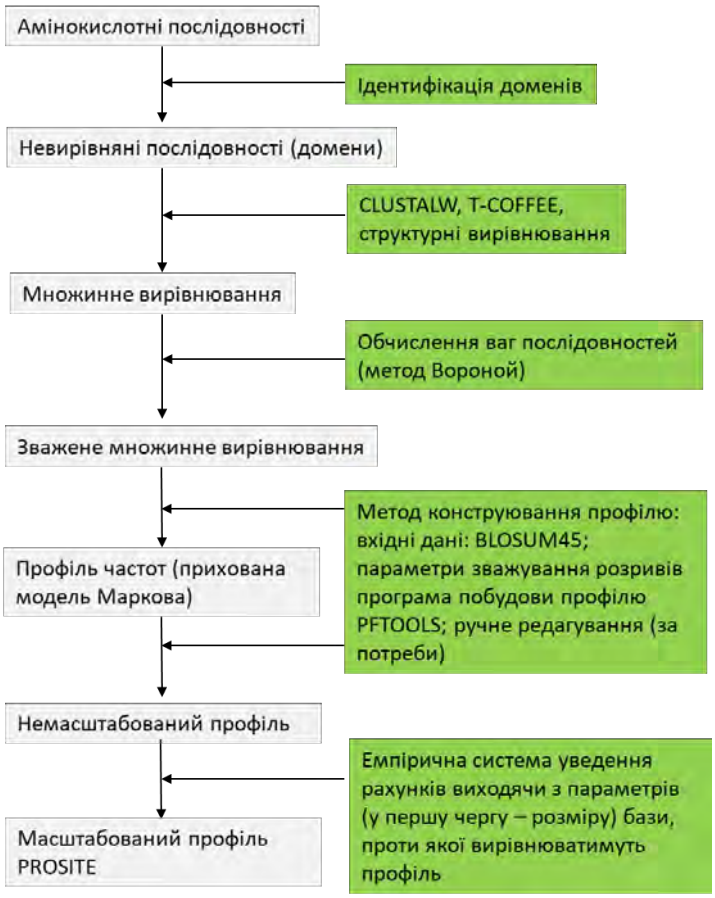


Рис. 5.16. Принцип конструювання генералізованого профілю

Кожній послідовності у цьому вирівнюванні далі приписують певну вагу з метою уникнення надмірної репрезентації окремих підродин білків. Зважене множинне вирівнювання далі перетворюють у “немасштабований профіль”. Цей крок відбувається поетапно. На першому етапі створюють профіль частот, в якому кожну колонку множинного вирівнювання картують до певної позиції

профілю – стану M або I. За замовчуванням, стан M приписують колонкам, які містять менш 50 % розривів, стан I приписують усім іншим колонкам. На другому етапі профіль частот трапляння амінокислотних залишків у кожному стані M перетворюють у профіль рахунків, або вагу залишку j у позиції профілю i – M_{ij}. Рахунки M_{ij} обчислюють за формулою Ґрібскова:

$$M_{ij} = \sum_{j'} s_{jj'} f_{ij} \,, \tag{49}$$

де $s_{jj'}$ – рахунок заміщення пари амінокислотних залишків jj' (відповідно до значень використаної матриці заміщень, найчастіше це BLOSUM45); f_{ij} – зважена частота залишку j у позиції профілю i.

Рахунки (вагу) переходів зі стану M до станів I чи D обчислюють аналогічно. Штраф за розрив у послідовності чи профілі дуже великий у тих позиціях профілю, де розриви досі не спостерігали. Він зменшується у тих позиціях, де розриви спостерігають у відповідних позиціях вирівнювання способом, який залежить від розміру розривів, а не частоти їхньої зустрічності.

Отриманий у результаті немасштабований профіль містить “сирі” рахунки, що описують ступінь очікування того, що амінокислотний залишок j може бути у позиції профілю i. Ці рахунки побудовані на довільних одиницях вимірювань, і їхній біологічний зміст складно зрозуміти. Однак PROSITE-формат генералізованого профілю дає змогу перетворити “сирі” рахунки в зрозумілі (з точки зору біологічної функції досліджуваних послідовностей) нормалізовані рахунки. Як і в попарному порівнянні двох послідовностей, наріжною проблемою під час вирівнювання профілю до послідовності є здатність розрізнити випадкові збіги і вирівнювання, що відображають спільне еволюційне минуле (гомологію). Висновки можна робити на основі порівняння рахунків вирівнювання, до яких застосовують розглянуту у попередніх підрозділах статистичну теорію Карліна-Альтшуля. “Сирі” дані перетворюють у десятковий логарифм величини очікування E у перерахунку на один амінокислотний залишок. На основі цих нормалізованих (масштабованих) рахунків можна визначити кількість у базі певного розміру очікуваних збігів з профілем, що матимуть рахунок, рівний або більший S. Наприклад, якщо рахунок вирівнювання профілю до послідовності становить 9,0, то у базі даних із одного мільярда залишків очікують на один випадковий збіг. Нормалізований і “сирий” рахунки пов'язані лінійною залежністю, чиї параметри еквівалентні λ і K з теорії Карліна-Альтшуля (див. розділ 3, пункт 3.5.4).

Для масштабування профілю використовують емпіричну процедуру. На першому етапі профіль вирівнюють щодо рандомізованої бази амінокислотних послідовностей, у результаті відбирають 2 000 збігів (хітів) із найвищими рахунками. Кумулятивний розподіл значень хітів далі підлаштовують до розподілу екстремальних значень, щоб визначити параметри масштабування. Графічну репрезентацію генералізованого профілю наведено на рис. 5.17.

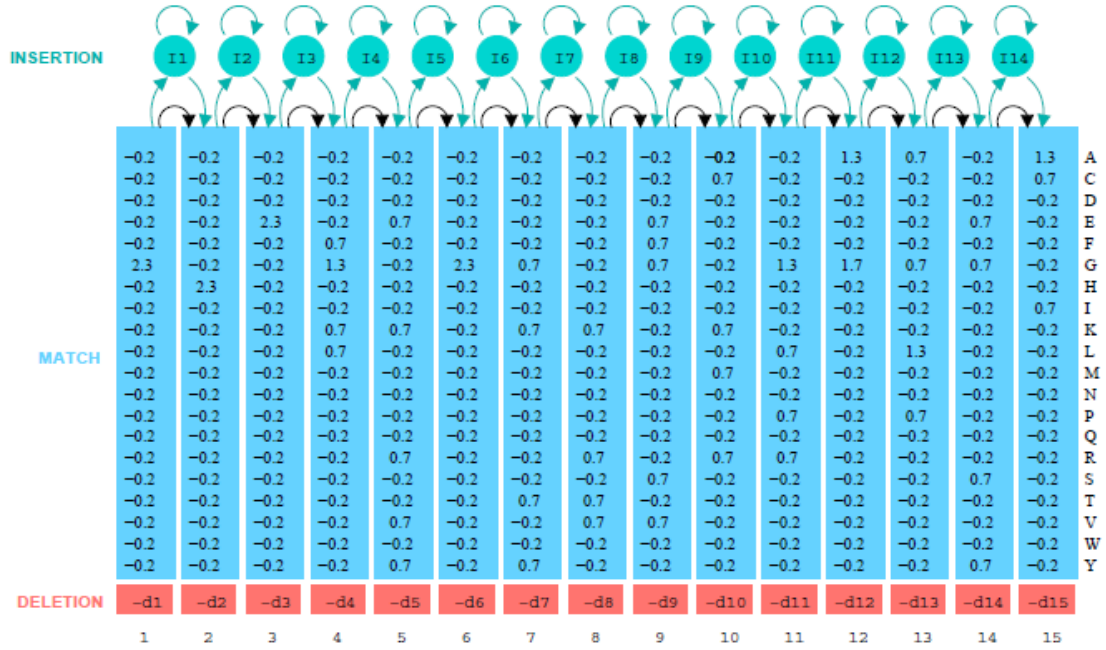


Рис. 5.17. Приклад генералізованого профілю

Текстовий варіант генералізованого профілю наведено на рис. 5.18.

```
ID   THIOREDOXIN_2; MATRIX.
AC   PS50223;
DT         ? (CREATED); MAY-1999 (DATA UPDATE);         ? (INFO UPDATE).
DE   Thioredoxin-domain (does not find all).
MA   /GENERAL_SPEC: ALPHABET='ABCDEFGHIKLMNPQRSTVWYZ'; LENGTH=103;
MA   /DISJOINT: DEFINITION=PROTECT; N1=6; N2=98;
MA   /NORMALIZATION: MODE=1; FUNCTION=LINEAR; R1=1.9370; R2=0.01816483; TEXT='-LogE';
MA   /CUT_OFF: LEVEL=0; SCORE=361; N_SCORE=8.5; MODE=1; TEXT='!';
MA   /DEFAULT: D=-20; I=-20; B1=-100; E1=-100; MM=1; MI=-105; MD=-105; IM=-105; DM=-105; M0=-6;
MA   /I: B1=0; BI=-105; BD=-105;

                                        ... many lines deleted ...

MA   /M: SY='K'; M=-8,0,-25,1,8,-24,-14,-9,-22,19,-20,-11,0,-9,5,13,-3,-4,-16,-24,-13,6; D=-3;
MA   /I: I=-3; DM=-16;
MA   /M: SY='P'; M=-6,-13,-26,-12,-9,-12,-19,-14,-5,-11,-5,-4,-12,8,-11,-13,-9,-6,-6,-25,-11,-12;
MA   /M: SY='V'; M=-4,-22,-19,-24,-20,-2,-25,-21,11,-15,2,3,-20,-23,-17,-14,-9,-1,19,-11,-4,-19;
MA   /M: SY='A'; M=28,-7,-15,-13,-6,-20,-2,-15,-15,-6,-14,-11,-5,-12,-6,-11,9,1,-6,-21,-17,-6;
MA   /M: SY='P'; M=-6,-3,-27,2,2,-22,-14,-11,-20,-6,-24,-17,-5,25,-4,-11,3,1,-19,-29,-17,-3;
MA   /M: SY='W'; M=-16,-27,-41,-28,-21,2,-13,-20,-20,-16,-19,-17,-26,-25,-15,-15,-26,-20,-26,93,19,-15;
MA   /M: SY='C'; M=-9,-17,106,-26,-27,-20,-27,-28,-29,-28,-20,-20,-17,-37,-28,-28,-8,-9,-10,-48,-29,-27;
MA   /M: SY='G'; M=-4,-12,-31,-9,-9,-27,24,-18,-27,-13,-25,-17,-7,14,-13,-17,-3,-13,-24,-24,-26,-13;
MA   /M: SY='H'; M=-12,-10,-30,-8,-4,-14,-18,18,-17,-10,-18,-8,-7,16,-5,-11,-8,-10,-20,-22,-1,-8;
MA   /M: SY='C'; M=-9,-19,111,-28,-28,-20,-29,-29,-28,-29,-20,-19,-18,-38,-28,-29,-8,-8,-9,-49,-29,-28;
MA   /M: SY='R'; M=-12,-4,-27,-4,3,-22,-20,-2,-21,22,-19,-6,-2,-13,9,23,-9,-8,-16,-20,-6,4;

                                        ... many lines deleted ...

//
```

Рис. 5.18. Текстовий варіант генералізованого профілю

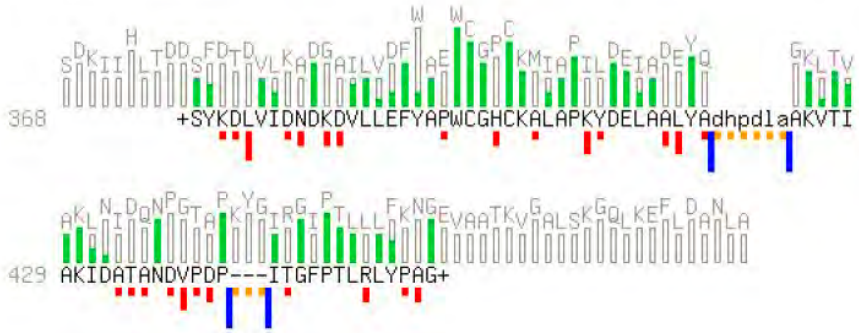
Вирівнювання профілів до послідовностей не відрізняється від вирівнювання послідовностей між собою чи з PSSM (див. рис. 5.9). Вебсервер

PROSITE (www.prosite.expasy.org) містить велику колекцію генералізованих профілів, які описують різноманітні домени, до яких можна вирівнювати амінокислотні послідовності. Позиційна специфічність рахунків генералізованих профілів дає змогу ідентифікувати віддалені послідовності як гомолог певної родини чи домену. Наприклад, у попарному вирівнюванні двох тіоредоксинових доменів рахунки за збіг усіх цистеїнів, пролінів тощо матимуть однакове значення, величина якого залежить від вибору матриці заміщення (рис. 5.19, *а*). Всупереч цьому, у вирівнюванні профілю тіоредоксинового домену до послідовності одного з тіоредоксинів помітно, що деякі зі збігів мають значно вищі значення, ніж інші (рис. 5.19, *б*).

а

```
THIO_ECOLI SFDTDVLKADGAILVDFWAEWCGPCKMIAPILDEIADEYQ------GKLTVAKLNIDQNP
           :.   :.  :  .:..:.: ::: :: .::  ::.:  :       .:.:.::..   :
PDI_ASPNG  SYKDLVIDNDKDVLLEFYAPWCGHCKALAPKYDELAALYADHPDLAAKVTIAKIDATAND

THIO_ECOLI GTAPKYGIRGIPTLLLFKNG
              :   : :.::: :.  :
PDI_ASPNG  VPDP---ITGFPTLRLYPAG
```



б

```
consensus    1 XVXVLSDENFDEXVXDSDKPVLVDFYAPWCGHCRALAPVFEELAEEYK----DBVKFVKV  -48
               : :          : : :  ::  : : :::: : :    : :  :        :
PDI_ASPNG  360 PVTVVVAHSYKDLVIDNDKDVLLEFYAPWCGHCKALAPKYDELAALYAdhpdLAAKVTIA  -97

consensus   57 DVDENXELAEEYGVRGFPTIMFF--KBGEXVERYSGARBKEDLXEFIEK              -1
                 :              : ::              : : :  : :  :
PDI_ASPNG  420 KID-ATANDVPDPITGFPTLRLYpaGAKDSPIEYSGSRTVEDLANFVKE              -49
```

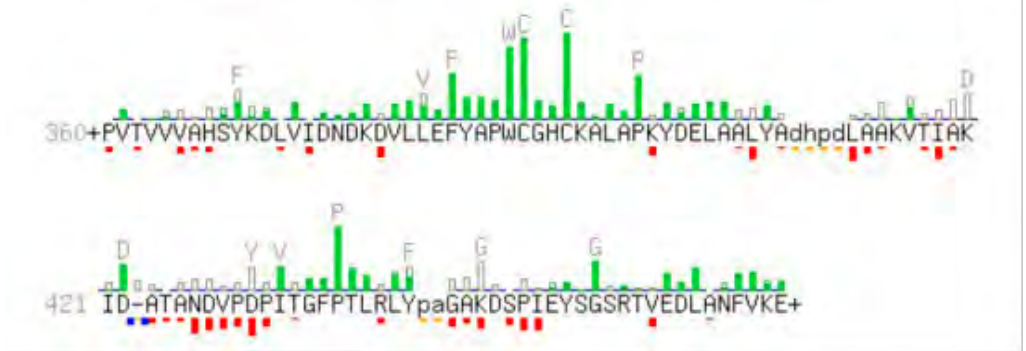


Рис. 5.19. Вирівнювання послідовності тіоредоксину PDI_ASPNG з тіоредоксином THIO_ECOLI (*а*) і з профілем тіоредоксинового домену consensus (*б*). Зверніть увагу на рахунки вирівнювання залишків цистеїну й проліну у послідовностях і профілі

Крім PROSITE, різновиди генералізованих профілів є основою кількох веб-сервісів пошуку гомології між віддаленими білками за рахунок вирівнювання профілів. Таким, зокрема, є сервер COMPASS (http://prodata.swmed.edu/compass). На основі заданої послідовності чи множинного вирівнювання COMPASS конструює позиційно-специфічну матрицю, що допускає розриви (отже, є профілем), яку можна використати як запит для порівняння проти наявних профілів у таких базах даних, як SCOP, Pfam, COG, KOG, PDB (рис. 5.20). Опис баз даних, з якими порівнюють задану послідовність чи множинне вирівнювання, є на сторінці вебсервера. Перші чотири бази – це колекції білків, що структурно подібні, або подібні за первинною стурктурою чи доменною організацією. Наприклад, база SCOP (https://scop.mrc-lmb.cam.ac.uk/) забезпечує докладний і актуальний опис структурних та еволюційних зв'язків усіх білків, для яких відома просторова структура. Остання база (PDB; https://www.rcsb.org/) – це сховище інформації про атомні координати усіх білків, для яких експериментально встановлена тривимірна структура. Амінокислотні послідовності з вищеперелічених баз згрупували на основі відсотка подібності первинної структури, утворені множинні вирівнювання та моделі використовують для порівняння із заданою послідовністю. Вебсервер використовує алгоритм локального вирівнювання Сміта-Уотермана. Статистичне оцінювання вирівнювань базується загалом на теорії, що використана у програмі PSI-BLAST. COMPASS дає змогу у межах одного робочого вікна вирівнювати та візуалізувати тривимірні структури хітів з відомими атомними координатами. Вебсервер також автоматизує картування залишків послідовності білка з невідомою структурою на послідовність гомолога з описаною тривимірною будовою.

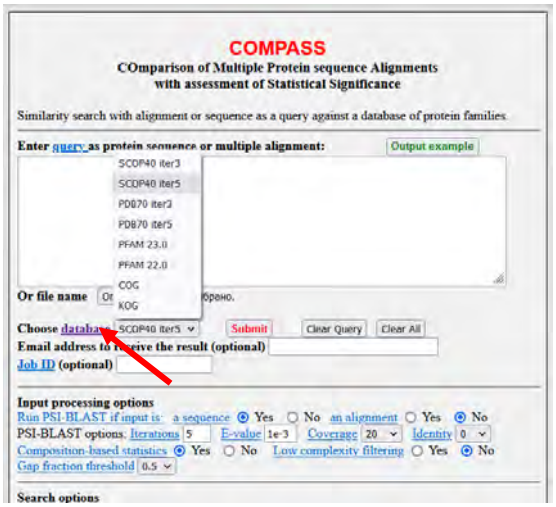


Рис. 5.20. Стартова сторінка вебсервера COMPASS. Розгорнуте спадне меню з переліком баз даних, з якими порівнюватимуть задану послідовність чи множинне вирівнювання (при натисканні на лінк database (червона стрілка) користувач отримає інформацію про ці бази даних)

5.5. Приховані моделі Маркова (HMMs). Як і розглянуті у розділі 2 ланцюги Маркова, *приховані моделі Маркова* (Hidden Markov Models; HMMs) є системами зі скінченним числом *станів*, що з'єднані *переходами*. Ключовою

відмінністю HMMs від ланцюгів Маркова є те, що стани в HMMs є не символами, а *розподілом символів*. Відповідно до усталеної термінології, кажуть, що кожен стан *емітує* певний символ з імовірністю, визначеною розподілом. Концептуальну схему HMM, що генерує GC-багату послідовність, відображено на рис. 5.21. У кінцевому результаті видимою є власне послідовність (ланцюг) нуклеотидів (символів), а сама *послідовність станів*, що привела до ланцюга нуклеотидів – невідома (прихована); її намагаються визначити з послідовності нуклеотидів. Послідовність станів ще називають “прихованим ланцюгом Маркова”.

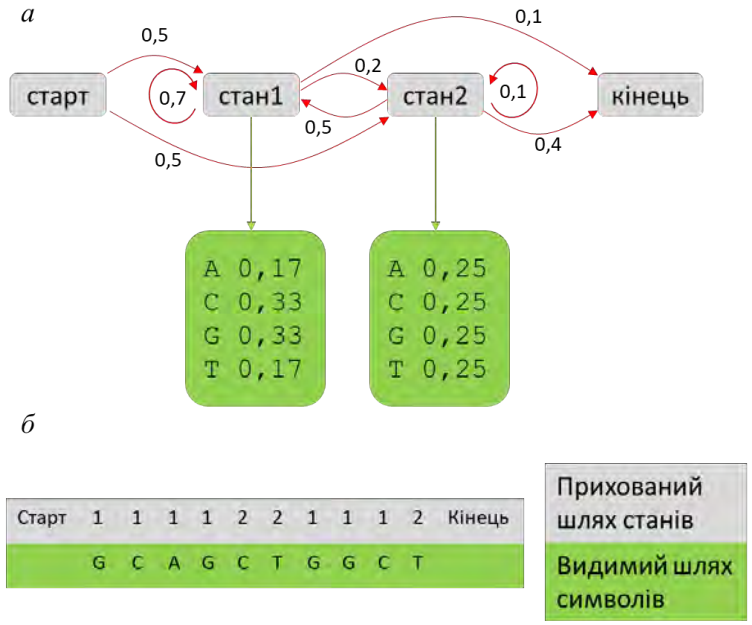


Рис. 5.21. Схема прихованої моделі Маркова (*а*), що складається з двох станів (розподіли символів, емітованих кожним станом, відрізняються). Послідовність, яку генерує ця HMM (*б*). Послідовність станів, що привела до такої послідовності – прихована

Розглянемо тепер декілька прикладів, до вирішення яких можна застосовувати концепцію HMMs. Найпоширенішим завданням біоінформатики є визначення функціонального значення певної генетичної послідовності (одного залишку), тобто для просеквенованого фрагмента ДНК необхідно визначити, де знаходяться екзони, інтрони та міжгенні ділянки. У множинних вирівнюваннях необхідно вирівняти амінокислотні залишки заданої послідовності (запиту) із гомологічними залишками послідовностей з бази даних. Це – різні стани, в яких може перебувати НАП, вони приховані і їх ми можемо висновувати лише за видимою послідовністю символів (нуклеотидів, амінокислот). HMMs вирішують проблему ідентифікації стану на підставі аналізу послідовності символів.

Краще зрозуміти, як працює HMM, нам допоможе проста графічна схема. Нехай задано нуклеотидну послідовність, що починається в екзоні, містить один 5′-сайт сплайсингу і закінчується в інтроні. Завдання – ідентифікувати місце переходу екзона в інтрон – 5′-сайт сплайсингу (5SS). Для цього необхідно, щоб

послідовності екзона, інтрона та 5SS мали різні статистичні властивості. Припустимо, що екзони мають однорідний нуклеотидний склад (кожен залишок – 25 %), інтрони АТ-багаті (А і Т – по 40 %, G і С – по 10 %), а консенсусний нуклеотид 5SS майже завжди G (наприклад, 95 % G, 5 % А). Ці дані отримують із *тренувального набору* послідовностей. У цьому разі ними можуть бути експериментально перевірені послідовності навколо 5SS. Множинне вирівнювання цих послідовностей дає змогу отримати дані про нуклеотидний склад станів E, 5SS й I.

Отже, можна зобразити HMM (рис. 5.22).

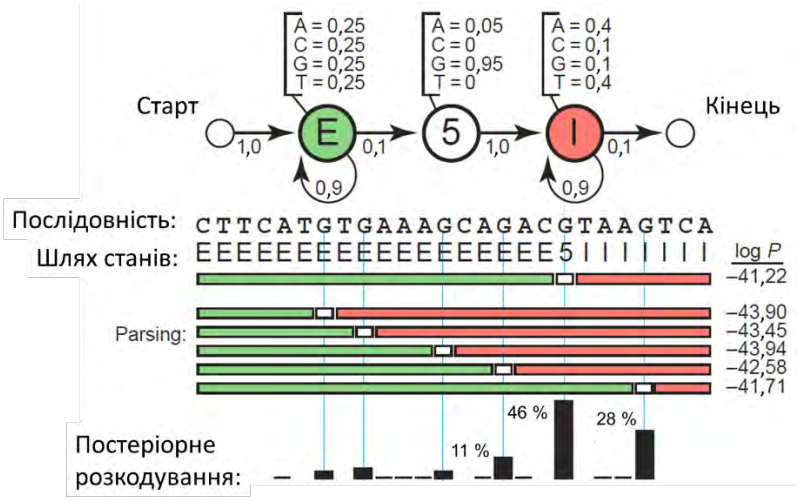


Рис. 5.22. HMM для розпізнавання 5′ сайту спласингу (див також основний текст). Рисунок адаптовано зі статті Nat. Biotechnol. – 2004. – Vol. 22, №10. – P. 1315–1316

Ця HMM матиме три стани, по одному на кожне з трьох можливих позначень, які можна приписати нуклеотидному залишку: *E* (екзон), 5 (5SS) й *I* (інтрон). Кожен стан має власні ймовірності емісії символів (A, T, G, C), які відображають нуклеотидний склад екзонів, 5SS та інтронів. Кожен стан також має імовірності переходу (стрілки) з одного стану в інші. Розміщення станів та переходів між ними визначають лінійний порядок очікуваної появи станів: один або більше *E* → один *5* → один або більше *I*.

Як уже зазначено (див. рис. 5.21 та супровідний текст), HMM тут – це генератор послідовності. Іншими словами, будь-яку послідовність можна репрезентувати як шлях через HMM. Наприклад, стан *E* емітує символ (нуклеотидний залишок) відповідно до розподілу ймовірностей емісії всіх можливих у цьому стані символів. Потім відбувається перехід до нового стану, відповідно до розподілу ймовірностей переходу з цього стану до всіх можливих інших. Розглянута вище модель Маркова (див. рис. 5.22) – першого порядку, оскільки ймовірність переходу до нового стану залежить лише від попереднього (того, в якому система зараз перебуває). Ця HMM генерує два ряди інформації. Першим рядом є *шлях станів* (state path; позначення символів – *E*, 5, I), другим – *шлях символів* – наявна нуклеотидна послідовність (observed sequence). У

послідовності кожен нуклеотидний залишок емітується одним станом зі шляху станів. Завдання дослідника – визначити прихований шлях станів (визначити, де *E*, 5, *I*) за наявною послідовністю.

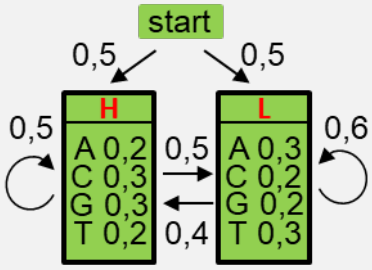
Імовірність $P(S,\pi|\text{HMM},\theta)$ того, що HMM з параметрами θ (розподіл імовірностей емісії і переходів) генеруватиме шлях станів π і наявну послідовність *S*, є добутком усіх складових обраного шляху – використаних імовірностей емісії та переходів. Наприклад, розглянемо 26-нт послідовність і шлях станів у центрі (див. рис. 5.22), де потрібно підсумувати 27 переходів та 26 емісій. З цією метою перемножимо усі 53 ймовірності і візьмемо їхній логарифм (для зручності оперування, оскільки це дуже малі числа), щоб визначити, що ймовірність (або "рахунок") $P(S,\pi|\text{HMM},\theta) = -41{,}22$.

HMM – повністю ймовірнісна модель, адже всі параметри моделі і значення рахунків послідовностей є ймовірностями. Тому до обчислених величин можна застосовувати теорію ймовірності Баєза з метою оптимізації параметрів та тлумачення статистичної значущості рахунків.

Отже, проблема "маркування" послідовності зведена до пошуку найкращого шляху станів через HMM – шляху з найвищою ймовірністю. Наприклад, якщо розглянути HMM та 26-нт послідовність (див. рис. 5.22), то тут є 14 можливих шляхів з ненульовою ймовірністю, оскільки 5SS (стан 5) має припасти на один із 14-ти внутрішніх A або G. На рис. 5.22 зображені шість шляхів із найвищими рахунками (ймовірностями) – ті, що припадають на G. Найвищий рахунок має логарифм імовірності – 41,22; звідси очевидно, що найімовірніша позиція 5SS припадає на п'ятий G. Для більшості проблем кількість можливих шляхів станів – астрономічна, і "ручне" обчислення їхніх імовірностей не має практичного змісту. Для цього розроблений ефективний алгоритм (*алгоритм Вітербі* (Viterbi)), який гарантує виявлення найімовірнішого шляху станів за умов певної послідовності й заданих параметрів HMM. Алгоритм Вітербі – це метод динамічного програмування, подібний до тих, що їх застосовують у вирівнюваннях послідовностей (див. попередні підрозділи, присвячені алгоритмам Сміта–Уотермана й Нідельмана–Ванча). Більше про цей алгоритм можна прочитати у наведеному нижче блоці 10.

На рисунку 5.22 також бачили, що ще один шлях станів за величиною рахунку незначно відрізняється від найкращого, який приписує п'ятому залишку G стан 5 (логарифм ймовірностей – 41,71 проти – 41,22). Наскільки можна бути впевненим, що саме п'ятий G є 5SS? Оскільки HMM – імовірнісна модель, можна обчислити достовірність (confidence) кожного обраного шляху. Ймовірність того, що символ *i* (нуклеотидний чи амінокислотний залишок) буде емітований станом *k*, є сумою ймовірностей усіх шляхів станів, які використовують *k*, щоб емітувати *i* ($\pi_i = k$ у шляху станів π), яку розділяють на суму всіх можливих шляхів станів (унормування даних). У моделі, зображеній на рис. 5.21, у чисельнику буде один шлях, а у знаменнику – 14 можливих шляхів. Так можна визначити, що вірогідність визначення п'ятого залишку G як 5SS (стан 5) дорівнює 46 %, а вірогідність для шостого G – 28 % (див. нижню частину рис. 5.21). Це *апостеріорне декодування* (posterior decoding). Для складніших моделей проблему апостеріорного декодування розв'язують за допомогою алгоритмів Forward і Backward, які за принципом функціонування нагадують алгоритм Вітербі.

БЛОК 10. Алгоритм Вітербі



Розглянемо HMM (праворуч), що складається з двох станів, H та L. Нехай стан H відповідає кодувальній ДНК, L – некодувальній. Модель можна використати, щоб передбачити ділянки кодувальної ДНК у заданій послідовності.

Розглянемо послідовність *S* = GGACTGAAC.

Є низка шляхів (*P*) прихованих станів, що приводитимуть до заданої послідовності, кожен з яких матиме свою ймовірність. Наприклад, *P* = LLHHHHLLL. Ймовірність того, що HMM генеруватиме задану послідовність *S* через шлях *P*, буде $p = p_L(0)\times p_L(G)\times p_{LL}\times p_L(G)\times \times p_{LH}\times p_H(A)\times p_{HH}\times p_H(C)\times p_{HH}$….=0,5×0,2×0,6×0,2×0,4×0,2×0,5×0,3×0,5….=…., де $p_L(0)$ – ймовірність, що *S* розпочинається у некодувальному районі (стан L); $p_L(G)$ – ймовірність трапляння G у некодувальному районі; p_{LL} – ймовірність того, що система залишиться в стані L (перехід L→L); $p_L(G)$ – ймовірність трапляння G у стані L (цього разу – це залишок G у другій позиції послідовності); p_{LH} – ймовірність того, що система перейде зі стану L в стан H (перехід L→H, на третьому кроці шляху станів L→L→H→H….) і так далі. Алгоритм Вітербі дає змогу обчислити найімовірніший шлях станів, що відповідає заданій *S*.

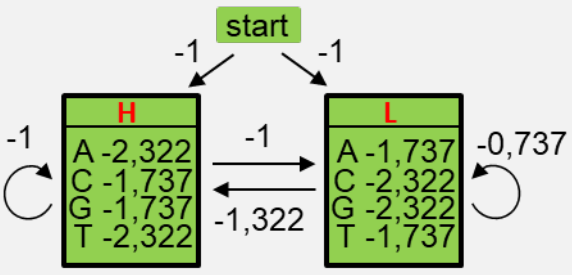
Ймовірність того, що найімовірніший шлях закінчиться станом k і в тому стані буде символ i, описує рівняння (10.1).

$$p_k(i,x) = e_k(i)\max(p_k(j,x-1)p_{kk}, p_l(j,x-1)p_{lk})\,. \quad (10.1)$$

$$p_H(C,4) = e_H(C)\max(p_L(A,3)p_{LH}, p_H(A,3)p_{HH})\,. \quad (10.2)$$

У рівнянні три множники. Це ймовірність трапляння символу i в стані k ($e_{k(i)}$), ймовірність найімовірнішого шляху символів, що закінчується в попередній позиції, що є добутком ймовірності символу j в позиції x-1 (тут маємо вибір між p_l, p_k; вибирається максимальне значення – функція max) та ймовірність відповідного переходу (p_{kk}, p_{lk}). Для нашого випадку рівняння (10.2) описує імовірність закінчення найімовірнішого шляху, що закінчується в стані H і символом C у 4-й позиції. Тут функція max дає змогу обрати той добуток, який максимізує ймовірність: $p_L(A,3)p_{LH}$ або $p_H(A,3)p_{HH}$.

Виражаємо значення ймовірностей p у HMM як їхні двійкові логарифми $\log_2 p$ (праворуч). Це значно спрощує дальші обчислення, оскільки логарифми ймовірностей можна додавати, а "сирі" значення p треба множити. Тоді:



ймовірність, що G в позиції 1, емітує стан H: $p_H(G,1) = p_H(0) + p_H(G) = -1-1{,}737=-2{,}737$;
ймовірність, що G в позиції 1, емітує стан L: $p_L(G,1) = p_L(0) + p_L(G) = -1-2{,}322=-3{,}322$;
ймовірність, що G в позиції 2, емітує стан H: $p_H(G,2) = p_H(G) + \max(p_H(G,1)p_{HH}, p_L(G,1)p_{LH})$ = = -1,737+max(-2,737-1 або -3,322-1,322) = -1,737-2,737-1=-5,474 (обираємо макс. значення); ймовірність, що G в позиції 2, емітує стан L: $p_L(G,2) = p_L(G) + \max(p_H(G,1)p_{HL}, p_L(G,1)p_{LL})$ = -2,232+max(-2,737-1 або -3,322-0,737) = -2,322-2,737-1=-6,059; і так далі. Усі значення вносимо у таблицю, відстежуємо оптимальний шлях за максимальними значеннями двійкового логарифму ймовірності (червоні стрілки). Отже, найімовірніший шлях станів такий: HHLHLLLLL, його ймовірність = $2^{-24{,}751} \approx 3{,}54\times10^{-8}$.

	G	G	A	C	T	G	A	A	C
H	-2,737	-5,474	-8,796	-10,948	-14,270	-16,744	-20,066	-22,862	-25,021
L	-3,322	-6,059	-8,211	-11,270	-13,685	-16,744	-19,218	-21,692	-24,751

Конструювання HMM потребує визначення чотирьох параметрів: абетки символів, K різних символів (для ДНК $K = 4$ (A, C, G, T), для білків $K = 20$); числа станів у моделі, M; імовірності емісії $e_i(x)$ для кожного стану i, що в сумі становлять одиницю для K символів, які трапляються в цьому стані, x: $\sum_x e_i(x)=1$; імовірності переходів $t_i(j)$ стану i у будь-який інший стан j (включно з повторенням стану i), що в сумі становлять 1 для M станів, j: $\sum_j t_i(j)=1$. Будь-яка модель, що має такі властивості, є HMM. Графічна простота побудови HMM дає змогу зосередитися на біологічному змісті проблеми, а також видозмінювати чи нарощувати певну модель у разі потреби. Наприклад, вже описана HMM (див. рис. 5.22) може бути недостатньо реалістичною. Тоді дослідник може відобразити повну консенсусну послідовність 5SS, GTRAGT у вигляді шести HMM-станів поспіль, які моделюють цей шестинуклеотидний безпрогалинний мотив. Так само можна змоделювати цілий інтрон, з 5′- і 3′-сайтами сплайсингу, чи цілий ген тощо.

Крім проблеми ідентифікації біологічної функції генетичних послідовностей (як описано на прикладі локалізації екзон-інтронних меж), HMM можна використовувати для репрезентації множинних вирівнювань амінокислотних послідовностей. У цій ролі HMM аналогічні таким методам узагальнення вирівнювань, як PSSM чи генералізовані профілі; часто у спеціалізованій літературі трапляється термін “HMM-профілі”. Є й програми, які дають змогу конвертувати генералізовані профілі у HMM-профілі, й навпаки. Як і в генералізованих профілях, кожній колонці множинного вирівнювання у моделі відповідає стан M (збіг; match), що визначає розподіл імовірностей натрапити у цій позиції на набір певних амінокислотних залишків. Стан I (insert) поблизу кожної колонки дає змогу вставити один або декілька залишків неподалік цієї колонки, а стан D (delete) – оминути її. Як і всі розглянуті приклади, HMM-профілі – це суворо лінійні моделі. При переході від стану до стану (зліва направо; відповідно до ймовірностей переходу) генерується амінокислотна послідовність унаслідок емісії станами амінокислотних залишків (відповідно до імовірностей емісії). Приклад HMM-профілю, що моделює просте множинне вирівнювання п’яти п’ятиамінокислотних послідовностей, можливо побачити на рис. 5.23. Як помітно з рисунка 5.23, *в*, п’ять станів M і шість I емітують амінокислотні залишки з імовірністю, що відображає частоти їхньої зустрічності у вихідному вирівнюванні. Зважаючи на невеликий обсяг вибірки (і, можливо, філогенетичну спорідненість організмів, з яких вирівняні послідовності походять), іншим залишкам також приписують незначну ймовірність емісії; ця процедура аналогічна приписуванню псевдорахунків у PSSM. Зауважимо, що відомі процедури та алгоритми для побудови HMM-профілів на основі невирівняних послідовностей. Такі підходи, що ґрунтуються на локальній оптимізації HMM зі заздалегідь заданою топологією станів, не завжди дають змогу отримати найкращу можливу конфігурацію моделі. Тому для практичних потреб завжди краще скористатися наявними попередньо вирівняними послідовностями.

HMM-профілі – ефективне знаряддя виявлення гомологічних послідовностей, оскільки вони містять більше інформації про певну білкову родину, ніж одна послідовність. Є низка програм вирівнювання HMM-профілів до послідовностей (і навпаки) чи між собою; деякі з них розглянемо згодом. Вирівнювання HMM-профілів до послідовностей аналогічне до попарного вирівнювання послідовностей (BLAST), чи вирівнювання послідовності до генералізованого профілю. Тут необхідно зазначити, що спосіб визначення і

біологічний зміст рахунків (імовірностей емісії) під час вирівнювання станів *M* аналогічний до рахунків *S* у попарних вирівнюваннях. Інші рахунки в HMM-профілях мають інший принцип визначення, ніж у попарних вирівнюваннях. Штрафи за розриви в HMM не є довільним заданим числом (як у BLAST), оскільки в HMM сума усіх можливих переходів з певного стану рівна одиниці (100 %). Отже, величини ймовірностей (рахунків) делеції, інсерції та переходи між станами потрібно ретельно збалансовувати, керуючись даними тренувальних наборів даних (множинних вирівнювань).

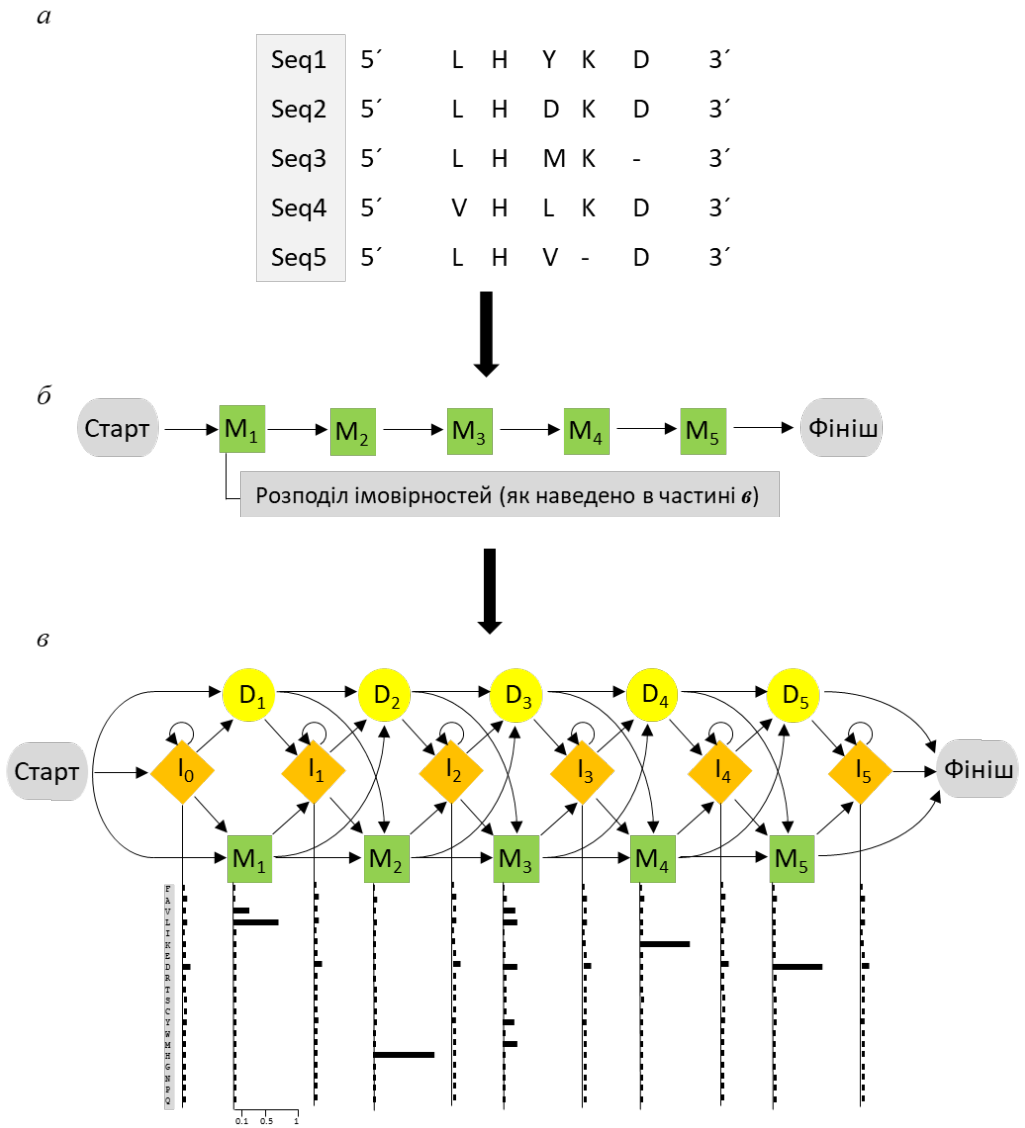
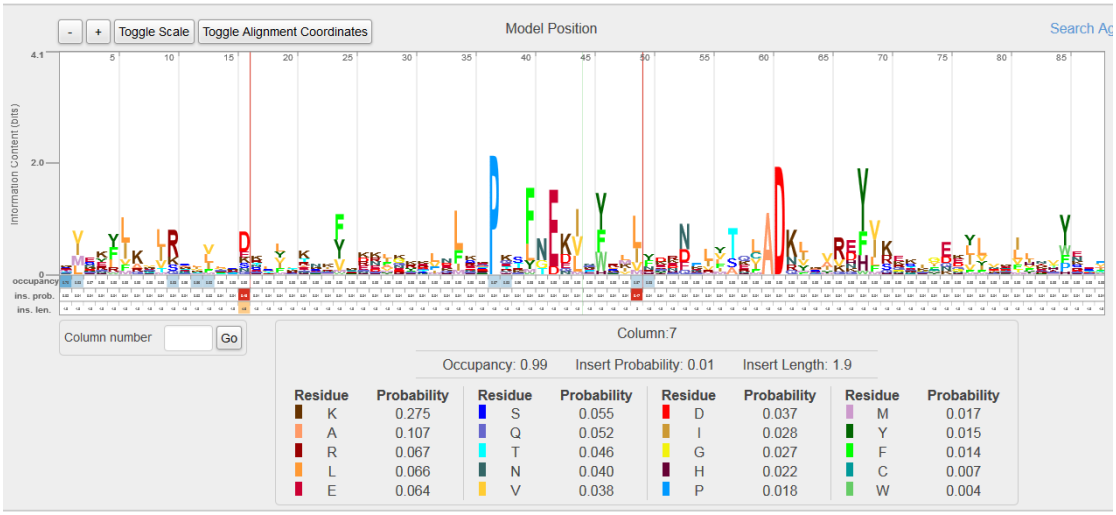


Рис. 5.23. Архітектура HMM-профілю. Множинне вирівнювання (*а*), на основі якого будують безрозривну HMM (*б*), містить тільки стани збігу (M_k); модель далі стає основою для побудови HMM, що містить також стани делеції (D_k) та інсерції (I_k). Остаточна модель (*в*) містить два порожні стани (старт і фініш), що не емітують символи. Стан *D* – мовчазний, не має ймовірностей емісії. Стани *M* та *I* мають свої ймовірності емісії амінокислотних залишків

З цією метою використовують декілька онлайн-знарядь для побудови HMM-профілів на основі набору амінокислотних послідовностей, зокрема, на

біоінформатичних серверах Pasteur Mobyle (www.mobyle.pasteur.fr/), FASTA (http://fasta.bioch.virginia. edu/fasta_www2/chaps.cgi). Результат такої побудови – текстовий файл, який містить розподіл імовірностей емісії амінокислотних залишків у станах *M* та *I* та ймовірності переходу до й від стану *D*. Зручним онлайн-знаряддям побудови HMM із множинних вирівнювань та його візуалізації у вигляді амінокислотного логотипу є вебсервіс Skylign – http://skylign.org/ . Фрагмент HMM-логотипу та вихідного текстового файла наведено на рис. 5.24.

а



б

```
HMMER3/f [3.1dev | April 2012] NAME  Query LENG  289 ALPH  amino RF    no MM    no CONS
yes CS    no MAP    yes DATE  Sun Apr  6 07:02:46 2014 NSEQ  14 EFFN  1.350098 CKSUM
1127768853 STATS LOCAL MSV      -10.8057  0.70160 STATS LOCAL VITERBI  -11.6883  0.70160
STATS LOCAL FORWARD   -5.4595  0.70160 HMM A C D E F G H I K L M N P Q R S T V W Y
            m->m     m->i     m->d     i->m     i->i     d->m     d->d
  COMPO   2.69328  4.18337  2.89004  2.63247  3.05500  3.25235  3.68119  2.75609  2.52980
2.43902  3.65171  2.94328  3.34577  3.09792  2.96647  2.75051  2.89124  2.73734  4.42117
3.18217
          2.68653  4.42274  2.77569  2.73151  3.46358  2.40552  3.72544  3.29327  2.67536
2.69321  4.24244  2.90345  2.73789  3.18196  2.89807  2.37936  2.77546  2.98549  4.58388
3.61447
          0.70916  1.22485  1.54111  1.92022  0.15850  0.00000        *
      1   2.63792  4.60675  3.21854  2.64346  3.80319  3.51065  3.70425  2.88934  2.10181
2.83437  2.66181  3.11462  3.89445  2.91169  2.52896  2.44178  2.86504  2.91033  5.13730
3.86495      20 k  - . -
```

Рис. 5.24. Фрагмент HMM-логотипу та HMM-профілю у текстовому форматі, створений на основі множинного вирівнювання WemB-подібних білків програмою Skylign; червоним овалом виділена 20-та позиція логотипу (*а*), яку також позначено у текстовому файлі внизу червоним тлом (*б*). Конструювання HMM виконане за допомогою програми HHMER3 (див. основн. текст). Логотип і HMM-файл можна завантажити на комп'ютер у низці форматів

Отже, HMM-профілі – це ймовірнісні моделі родини споріднених генетичних послідовностей (доменів), передусім амінокислотних (хоча відомі HMM-профілі і для нуклеотидних послідовностей). Бібліотеки таких моделей, як докладніше розглядатимемо далі на конкретних прикладах, можна використовувати для виявлення гомології. Аналогічно до BLAST, задану

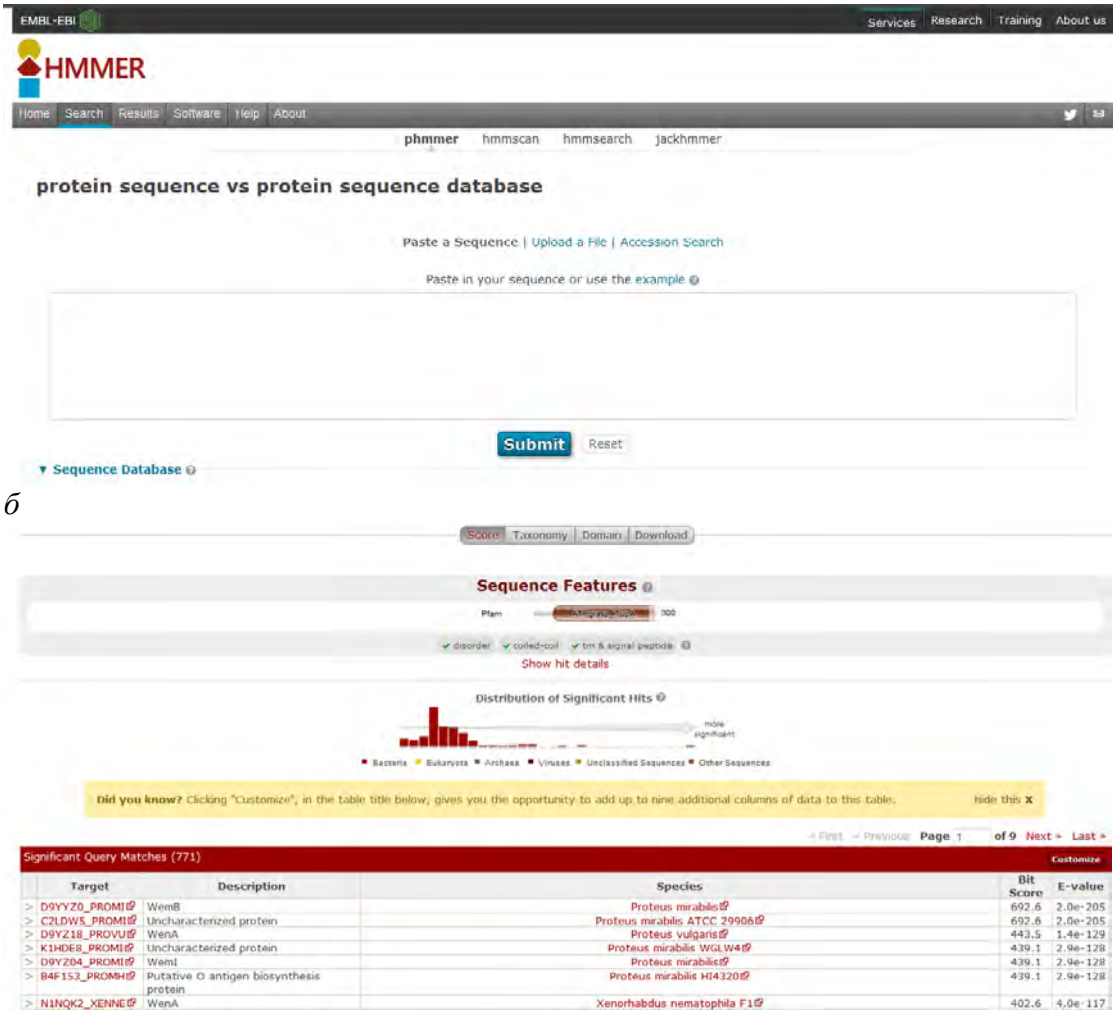
послідовність можна порівнювати з бібліотекою HMM-профілів; заданий профіль можна вирівнювати з бібліотекою послідовностей чи HMM-профілів. HMM застосовують для функціонального аналізу (анотації) генетичних послідовностей (пошук генів, екзонів-інтронів, трансмембранних доменів). Також HMM застосовують для розпізнавання елементів вторинної і третинної структури (fold recognition), яких набуватимуть генетичні послідовності – α-спіралі, β-складчасті шари тощо. Останній аспект використання HMM-профілів, на перший погляд, суперечить їхньому визначенню як моделей першого порядку, де поява певного стану залежить лише від попереднього. Як HMM може моделювати взаємодії між віддаленими амінокислотними залишками, які є основою просторової будови білка? Принцип позиційної незалежності означає тільки те, що у момент, коли стан HMM емітує певний символ у послідовності (приписує їй певний рахунок), він робить це незалежно від решти вирівнювання. Ніщо не перешкоджає тому, щоб розподіл імовірності емісії певного стану було визначено з урахуванням інформації про складну тривимірну структуру білка. Якщо відомо, що амінокислотний залишок є в складі сегмента гідрофобних амінокислот і що цей сегмент виявлений у багатьох структурах однієї родини білків, то цю інформацію можна вбудувати в HMM. Справжнім обмеженням HMM є нездатність моделювати взаємодії (чи кореляції) між віддаленими залишками, як це є у випадку вторинних структур РНК. Тут комплементарність пари віддалених нуклеотидних залишків важливіша, ніж ідентичність кожної з позицій, а HMM не "пам'ятає", що саме генерував віддалений стан. Водночас HMM можна успішно застосовувати для моделювання кореляцій на невеликих відстанях. Наприклад, у програмах пошуку генів стан HMM емітує не один залишок, а кодон (послідовність трьох прилеглих нуклеотидних залишків), або ж дикодон.

5.6. Вибрані онлайн-ресурси, що базуються на PSWM і HMM

5.6.1. HHMER3 – пошук подібності між генетичними послідовностями. Веб-адреса цього сервісу – http://www.ebi.ac.uk/Tools/hmmer/. HHMER3.1 присвячений аналізу амінокислотних послідовностей, а nhmmer – нуклеотидних. HHMER3.1 можна розглядати як аналог BLASTP, що ґрунтується на ймовірнісній моделі Маркова. Станом на 2020 рік пакет HHMER у своїй онлайн-версії містив чотири програми (для аналізу амінокислотних послідовностей): phhmer, hmmscan, hmmsearch та jackhmmer. Програма phhmer аналогічна BLASTP (рис. 5.25). Задану амінокислотну послідовність (у форматі FASTA) вирівнюють з базою даних. Для цього необхідно задану послідовність перетворити в HMM. Традиційно HMM-профілі розглядають як позиційно-специфічні статистичні моделі. В останній версії HHMER єдина задана послідовність перетворюється в HMM без пошуку гомологів і множинного вирівнювання. Ймовірності емісії залишків виводять зі стандартної матриці рахунків, такої як BLOSUM62, що має неявну ймовірнісну основу. Параметри делецій та інсерції встановлюють аналогічно до штрафів за відкриття і продовження розривів у BLAST. Отже, HMM, застосовуваний у HMMER, є особливим випадком теорії Маркова. На сторінці результатів HMMER наводять як біт-рахунки, так і величини очікування *E*. Біт-рахунок є логарифмом (основа 2) співвідношення достовірності HMM-профілю до достовірності нульової гіпотези (модель, в якій послідовності незалежні, і частоту їхнього трапляння описують одним розподілом імовірностей). Величина *E* має значення, ідентичне до такого в BLAST. Програма hmmscan порівнює задану послідовність з бібліотекою HMM-профілів Pfam (її

опис наведемо нижче). Вхідним матеріалом для програми hmmsearch слугує множинне вирівнювання, на основі якого побудований HMM-профіль, який порівнюють із заданою базою. Сьогодні HHMER дає змогу застосувати низку евристичних прийомів, які ми не розглядатимемо. Ці прийоми роблять процес аналізу послідовностей на сервері настільки ж швидким, як і на сервері BLAST.

а



б

Рис. 5.25. Вебсервіс HHMER: *а* – початкова сторінка; *б* – сторінка результатів. Набір даних, що відображається на сторінці результатів, можна змінювати (кнопка Customize у лівому куті)

5.6.2. HHSearch – пошук подібності між HMM-профілями. Сервіс доступний за адресою: http://toolkit.tuebingen.mpg.de/hhblits. Функціонує алгоритм HHSearch на припущенні, що шлях через дві HMMs відображає послідовність символів (амінокислотних залишків), яка коемітується обидвома HMMs (рис. 5.26). Завдання програми – пошук шляху через дві HMMs з максимальним рахунком вирівнювання станів двох моделей, тобто алгоритм HHSearch описує метод попарного вирівнювання HMM-профілів. HHSearch містить елементи евристики та дає змогу враховувати у процесі пошуку інформацію про структуру білків. Зокрема, в алгоритмі застосовують зважування послідовностей та псевдорахунки, за аналогією з PSI-BLAST.

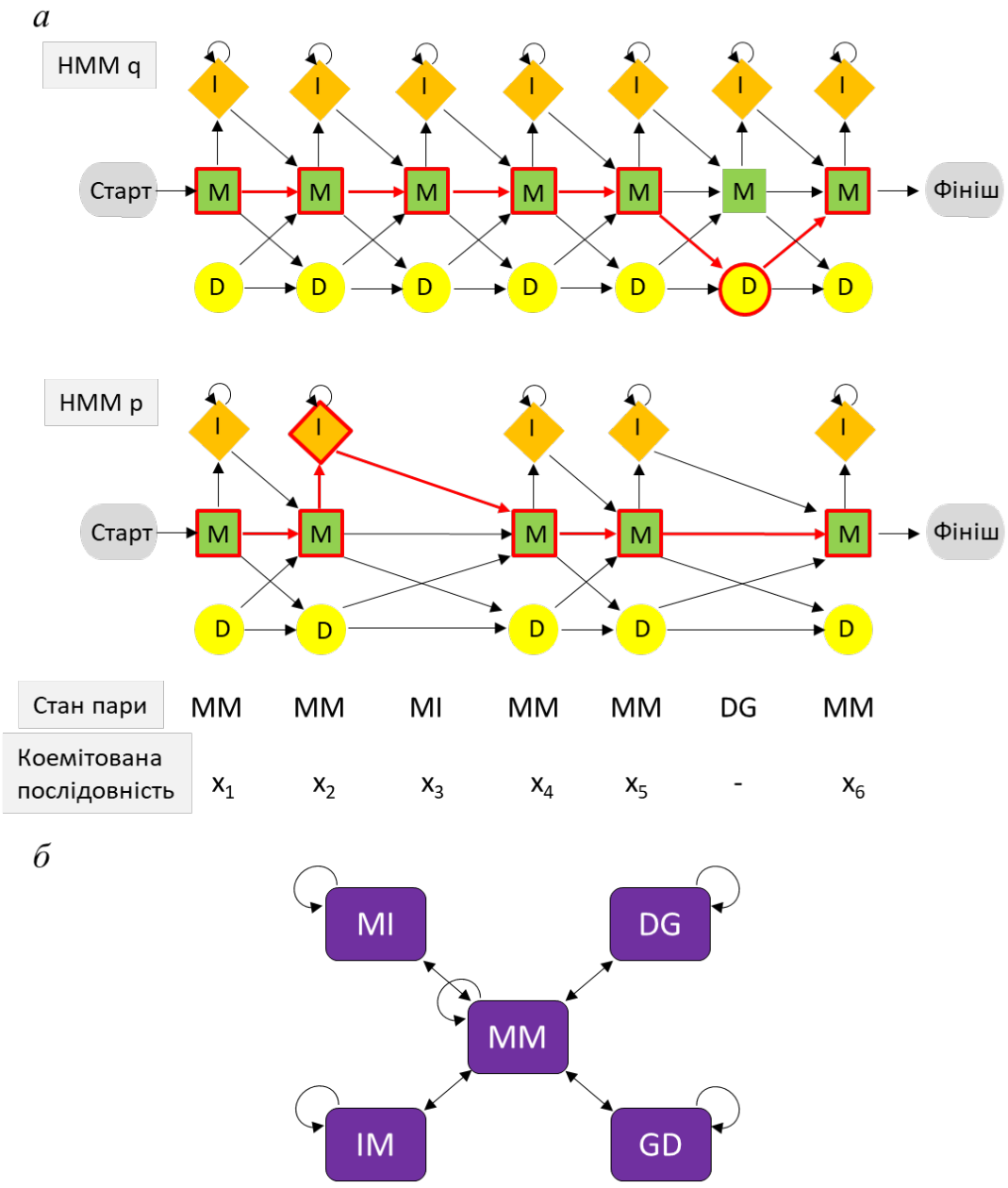


Рис. 5.26. Вирівнювання двох HMMs, *p* та *q* (*а*) – це пошук спільного шляху з найвищим сумарним рахунком (log-sum-of-odds score). Шлях через дві HMMs відображає послідовність, що коемітується обома профілями; оптимальний шлях через *p* та *q* позначений червоним. Стани *M* та *I* емітують літери, стан *D* – ні. Дозволені станові переходи у вирівнюванні двох HMMs (*б*). Адаптовано з Soding 2004; doi:10.1093/bioinformatics/bti125

У процесі вирівнювання використовують автокореляційну функцію, яка враховує той факт, що у вирівнюваннях гомологічних послідовностей законсервовані колонки часто розміщені поряд (кластерне упорядкування), а в негомологічних такого групування не спостерігають. Далі, якщо в процесі пошуку виявлені гомологи з відомою або передбаченою просторовою структурою, то HHSearch враховуватиме розміщення елементів вторинної структури. Для цього застосовують низку спеціальних матриць заміщення, які на

підставі первинних амінокислотних послідовностей описують усі відомі на сьогодні елементи вторинної структури. Стани *M* та *I* можна вирівнювати зі станами *M* та *I*, стан *D* – лише зі станом *D* або розривом G. Алгоритм HHSearch використовують у двох онлайн-програмах: HHBlits та HHPred. Розглянемо функціонування першої з них. HHBlits (HMM-HMM-based lighting-fast iterative sequence search; http://toolkit.tuebingen.mpg.de/hhblits/) є програмою циклічного пошуку гомологічних послідовностей, аналогічно до PSI-BLAST. Як задана послідовність, так і база даних мають бути репрезентовані у вигляді HMM. Для цього послідовності з таких найпопулярніших баз даних, як GenBank чи Uniprot, попередньо групують за подібністю, а ці групи далі перетворюють у HMM. Наприклад, 15 млн послідовностей Uniprot зібрані у 2,6 млн HMM (у середньому 5,5 послідовності на HMM). Задану послідовність чи множинне вирівнювання також перетворюють у HMM. Потім HHBlits шукає базу HMM і додає нові гомологічні послідовності до вихідного множинного вирівнювання. З такого доповненого вирівнювання будують новий HMM для чергового раунду пошуку. Для збільшення швидкості й чутливості пошуку застосовують евристичний підхід, що полягає у зведенні вирівнювання типу "HMM–HMM" до вирівнювання типу "послідовність–HMM". Для цього колонку HMM-профілю, що містить стани *D*, *I*, *M* (*I*, *M* мають характерний розподіл імовірностей емісії амінокислотних залишків), наближено репрезентують (апроксимують) у вигляді дискретного символу-літери. Кожна така літера відображає типову колонку HMM-профілю. Наприклад, якщо розглянути рис. 5.26, то у першій колонці (D_1-I_1-M_1) стан M_1 матиме розподіл залишків із переважанням амінокислот L та V. Таку колонку, наприклад, позначимо літерою φ. У другій колонці в M_2 повністю домінує гістидин. Такий роподіл позначимо літерою Δ. Автори програми фактично "розбили" все різноманіття колонок профілів (розподілів емісії) на 219 типів – починаючи від "чистих" станів, де лише один амінокислотний залишок, до тих, де цілком випадково (рівноімовірно) трапляються всі залишки. Так уся база HMM-профілів апроксимується псевдопослідовністю літер із 219-символьної абетки. Іншими словами, база профілів перетворюється у своєрідну базу послідовностей. Далі обчислюють вирівнювання кожної колонки заданого HMM-профіля із кожною із 219-ти літер, досягаючи побудови 219-символьного розширеного профілю на основі заданої HMM. Потім на першому етапі відбувається безрозривне вирівнювання, а на другому – вирівнювання з розривами розширеного профілю й бази HMM, "переписаної" 219-символьною абеткою. Цей своєрідний двоетапний "фільтр подібності" загалом проходить 2÷10 % HMM закодованих у символи. Далі вихідні HMM бази (що містять уже згадані відфільтровані 2÷10 % послідовностей) вирівнюють із заданою HMM і встановлюють статистичні показники достовірності. Для хітів виконують повторне максимально акуратне вирівнювання. Відповідно до тверджень авторів програми, HHBlits швидша і чутливіша, ніж PSI-BLAST і HHMER3.

Алгоритм HHSearch покладено в основу програми виявлення віддаленої структурної гомології – HHPred (https://toolkit.tuebingen.mpg.de/tools/hhpred). Розглянемо один приклад, щоб зрозуміти функціонування і переваги HMM-опосередкованого пошуку гомологів. Геном *Proteus mirabilis* кодує білок WemB, що в базі GenBank знаходиться під номером доступу ADL32277. Цей білок анотовано як глікозилтрансферазу. Повторна BLASTP-опосередкована перевірка цього білка наприкінці 2021 р. засвідчила, що усі хіти до нього анотовані як глікозилтрансферази. При перевірці амінокислотної послідовності WemB на

сервері HHPred найближчими до цього білка виявилися лігази, що приєднують амінокислотні залишки до різноманітних субстратів (рис. 5.27). Відмінний результат викликаний принциповими відмінностями у функціонуванні алгоритмів BLAST та HHSEARCH. BLAST вирівнює задану первинну структуру (амінокислотну послідовність) із базою, і виявить подібні послідовності у базі, якщо ті мають хоча б 30 % подібності до заданої (на рівні первинної структури).

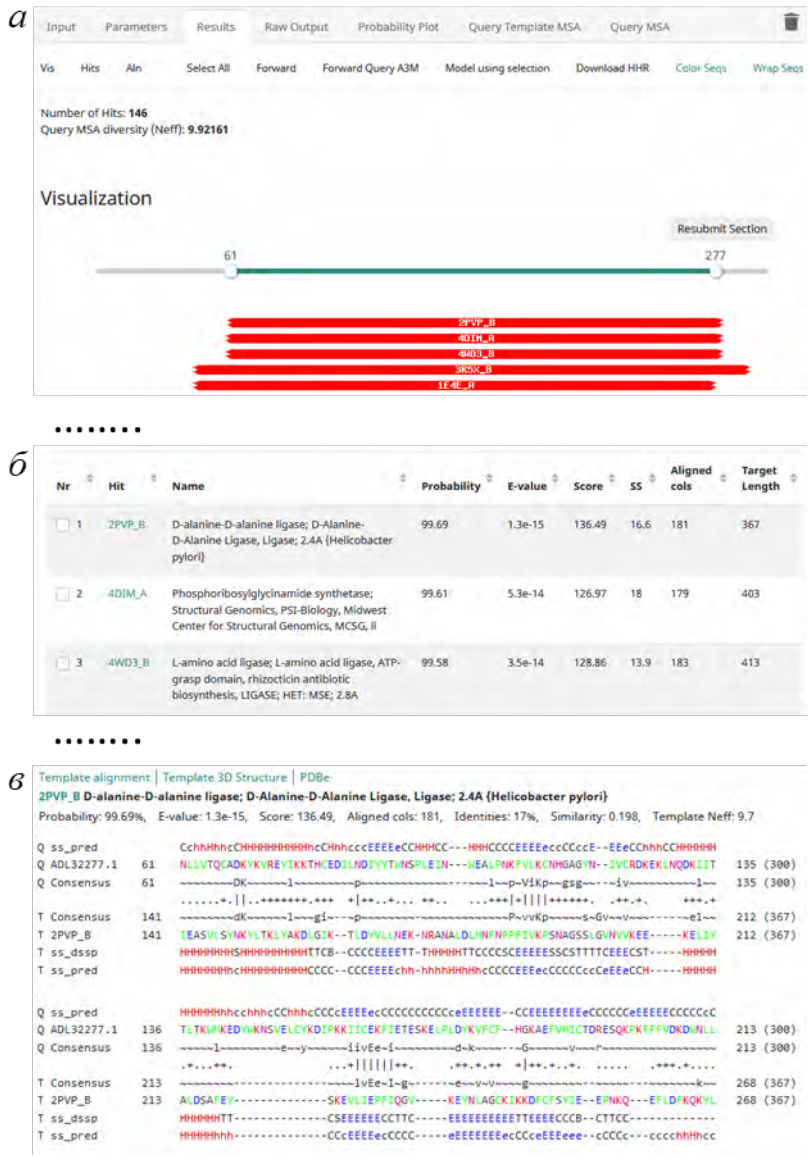


Рис. 5.27. Результат HHPred-опосередкованого пошуку гомологів білка WemB: однотипні елементи сторінки видалені для спрощення і замінені на рядок з крапок; *а* – сегмент заданого білка виявив гомологію до тих чи інших хітів; *б* – список хітів; *в* – попарні вирівнювання заданого білка (рядки зверху донизу: у вигляді послідовності станів вторинної структури; амінокислотної послідовності; консенсусу) та знайденого гомолога (рядки знизу догори: у вигляді послідовності станів вторинної структури за двома класифікаціями – ss_pred, ss_dssp; амінокислотної послідовності; консенсусу). Кожне вирівнювання супроводжується рахунком і числом очікування *E*, які за змістом аналогічні до таких у програмі BLAST

Отже, BLAST здатний виявити гомологи, якщо вони мають хоча б помірну подібність на рівні первинної структури. Натомість HHSEARCH використовує задану послідовність для виявлення подібних (на рівні первинної структури)

білків, для яких відома просторова будова. Таку колекцію білків разом із заданою використовують для побудови HMM-профілю, який включатиме вже інформацію про вторинну структуру, і цей HMM використають для попарного вирівнювання з базою HMM-профілів біліків з відомою структурою. Такий метод дає змогу виявити гомологію білків, що втратили подібність на рівні первинної структури.

5.6.3. Pfam – база родин білків. База доступна за адресою http://pfam.sanger.ac.uk/. Родина білків у Pfam – це набори ділянок білків (доменів), які мають значну подібність, засвідчуючи їхню гомологію. Подібність визначають за допомогою програми HHMER3, яку розглянуто (див. пункт 5.6.1). Pfam містить родини білків двох типів. Це родини Pfam-A, високоякісні набори даних, що куруються фахівцями вручну, і Pfam-B – генеруються автоматично. Родини Pfam-A конструюють поетапно. Спочатку (1) будують високоякісне множинне вирівнювання – "засівне вирівнювання" (seed alignment). На основі цього вирівнювання будують HMM (2). Цю HMM використовують для скринінгу бази UniProtKB (3). Усі послідовності цієї бази даних, що мають рахунок вирівнювання з HMM вище певної порогової величини (встановлюють для кожної родини окремо, залежно від довжини і композиційної складності послідовності), додають до повного вирівнювання представників родини (4). На листопад 2021 року база містить 20 тисяч родин Pfam-A, з яких чверть становлять домени з невивченою функцією –(domains of uncharacterized function, DUFs).

5.6.4. Phyre – веб-сервер для передбачення структури білка. Сервер доступний за адресою: http://www.sbg.bio.ic.ac.uk/phyre2/html/ page.cgi?id=index. За інформацією про описані структури білків він "звертається" до баз SCOP (Structural Classification of Proteins) та PDB (Protein Data Bank, NCBI). Первинні амінокислотні послідовності цих структур вирівнюють з базою NCBI(nr). На основі виявлених подібних послідовностей конструюють HMM-профіль. Для цього Phyre використовує програму HHSearch. Так створюється *бібліотека фолдів* (fold library): термін (від англ. fold – складка, згортка) позначає певний тип розміщення елементів вторинної структури у просторі, що часто притаманний функціонально подібним білкам. Наприклад, давно описаний фолд helix-turn-helix (HTH), що є у безлічі ДНК-зв'язувальних білків, які належать до різних родин, як от: TetR, GntR тощо; фолди типу β-діжки (β-barrel), пропелера, крильчастої спіралі, TIM-діжки і міріади інших. До певної міри терміни "фолд" і "домен" мають однакове значення. У бібліотеці фолдів також зберігаються відомі чи передбачені вторинні структури білків. Задану дослідником послідовність із невідомою структурою використовують як запит для скринінгу NCBI(nr) за допомогою PSI-BLAST (п'ять раундів) і побудови профілю на основі виявлених подібних послідовностей. Далі програма на основі отриманого профілю передбачає елементи вторинної структури білка. Для цього Phyre "послуговується" трьома програмами: PsiPred, SSPro, JNet. На основі трьох різних передбачень формується консенсусна вторинна структура. Далі вихідний профіль і вторинну структуру порівнюють з бібліотекою фолдів, використовуючи методи профіль–профільного вирівнювання. Десять найподібніших фолдів згодом використовують для передбачення тривимірної структури білка. Phyre регулярно займає високі місця за якістю передбачення структури білка у

міжнародному змаганні CASP (Critical Assessment of Structure Prediction), що відбувається кожні два роки.

5.6.5. GeneMark – програма пошуку генів ab initio. Проблема коректного ідентифікування генів у новосеквенованих геномах має два шари складності. Перший – простіший – стосується ідентифікації генів, які мають описані гомологи в інших геномах. Тут можна скористатися низкою програм попарного вирівнювання, які, зазвичай, легко виявляють гени, що подібні на рівні нуклеотидної чи ймовірної амінокислотної послідовності до вже описаних генів. Другий шар – це нові гени, що не мають гомологів у наявних базах даних: вони становлять значний відсоток кожного геному, і їхнє виявлення методами вирівнювання неможливе. Для виявлення таких генів потрібно використовувати підходи до пошуку генів *ab initio*. Це підходи, які використовують для пошуку гена тільки послідовність ДНК, без прямого залучення сторонніх нуклеотидних послідовностей. До таких програм належить GeneMark. Адреса веб-версії пакета програм GeneMark: http://opal.biology.gatech.edu/ GeneMark/. До пакета входять дві основні програми: GeneMark і GeneMark.hmm. Обидві програми використовують негомогенну (з періодом три) модель Маркова для опису триплетної структури кодувальних послідовностей, і гомогенну модель – для опису некодувальної ДНК. GeneMark використовує статистичну теорію Баєза, щоб обчислити апостеріорну ймовірність наявності генетичного коду (принаймні в одній із шести можливих рамок зчитування) у коротких послідовностях ДНК. GeneMark.hmm використовує логіку HMM для виявлення найімовірнішої послідовності прихованих станів (наприклад, кодувальна/некодувальна ДНК) на основі аналізу всієї наявної ДНК. Отже, в GeneMark моделі Маркова є *моделями гена* – вони описують імовірність появи RBS, промоторів з певною послідовністю, GC- і кодонного складу тощо. Ці ознаки гена є спільними для певного таксона і можуть бути використані для виявлення нових генів у вибірці геномів, що входять до цього таксона чи філогенетично близькі до нього. Наприклад, сьогодні користувачі сервера GeneMark можуть обрати одну із 256-ти попередньо сконструйованих моделей гена, які підібрані на основі аналізу 256-ти повністю секвенованих прокаріотичних геномів, що репрезентують усі основні таксономічні групи. Звідки беруть параметри моделі гена (частоти кодонів, послідовність RBS, промотора тощо), якщо у вихідному стані вони невідомі? У GeneMark моделі гена ґрунтуються на наближеному визначенні цих параметрів із GC-складу ДНК і частот появи гексамерів у всіх можливих рамках зчитування ДНК. Ключове припущення (обґрунтоване 1992 р. Фікетом і Тангом емпірично на великому масиві послідовностей) – за наявного відсотка GC-основ у ДНК кодувальна послідовність має містити лише певні гексамери. Наприклад, у GC-багатій ДНК кодуюча послідовність неодмінно складатиметься із кодонів, що міститимуть у третій позиції частіше GC й зрідка – AT. Водночас гексамери поза рамкою зчитування такої закономірності не демонструють. Короткий фрагмент ДНК (~400 нт) є достатнім для встановлення параметрів HMM, використовуваних GeneMark. Це евристичний підхід; розробники GeneMark вважають, що він засвідчує ключову роль мутаційного тиску на GC-склад геному як чинника, що формує характер вживання кодонів. Після визначення базових статистичних параметрів і побудови моделі гена програми пакета GeneMark починають сканувати задану послідовність для виявлення кодувальних послідовностей. Сканування відбувається за принципом “вікна, що ковзає” (sliding window) – програма за замовчуванням аналізує 96 нт поспіль – визначає

апостеріорну ймовірність кодування цим сегментом білкової послідовності, далі переміщається на наступні 96 нт і т. д. Так сканується повна послідовність у всіх шести можливих рамках зчитування. Типовий графічний результат ідентифікації кодуючих послідовностей у ДНК відображено на рис. 5.28 (для пошуку використані параметри за замовчуванням; модель гена, яку відібрали для пошуку, обчислено на основі хромосоми *Streptomyces coelicolor*).

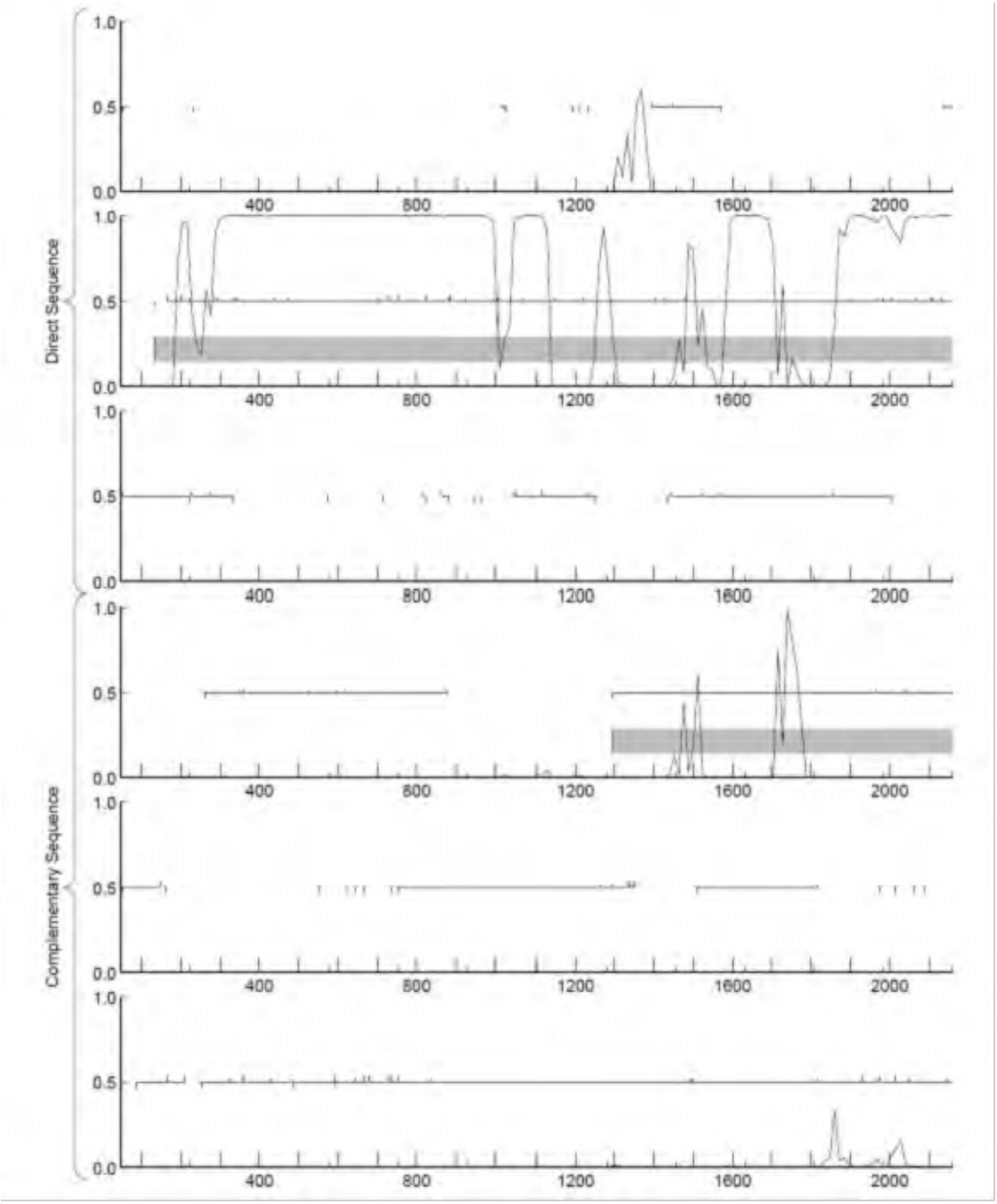


Рис. 5.28. Пошук кодувальних послідовностей програмою GeneMark у 2 200-нуклеотидному сегменті ДНК (GI: 300492603), що походить зі *Streptomyces globisporus* 1912. Програма аналізує задану (direct sequence) і комплементарну (complementary sequence) нитки ДНК у всіх рамках зчитування. Вісь ординат – апостеріорні ймовірності; сірими прямокутниками позначені передбачені кодувальні послідовності

GeneMark й низка інших концептуально подібних програм (GeneScan) досягає високої точності передбачення кодуючих послідовностей у про- та еукаріотичних геномах. Однак невирішеною проблемою цього пакета програм (і всіх наявних сьогодні підходів до пошуку генів) є точне визначення тРНК, старт-кодону і RBS, оскільки ділянки ініціації транскрипції і трансляції мають відносно слабкі паттерни (статистичні особливості їхньої послідовності), які були б суттєво відмінними від статистичних властивостей кодуючих послідовностей. Накопичення експериментальних даних щодо точок ініціації транскрипції і трансляції у різних організмах стане ефективним тренувальним набором, що сприятиме вирішенню цієї проблеми.

5.6.6. TMHMM – програма передбачення трансмембранних доменів білків. Адреса вебсервісу: http://www.cbs.dtu.dk/services/TMHMM-2.0/. Білки, що контактують із біологічними мембранами, або вбудовані у них (рис. 5.29), становлять значний відсоток протеому і виконують життєво важливі функції, такі як: транспортування низькомолекулярних сполук, експортування інших білків, сигнальна трансдукція тощо. Наявність трансмембранного (ТМ) домену (чи кількох ТМ-доменів) – важлива ознака білка, яка дає змогу його точніше класифікувати, передбачити локалізацію і функцію.

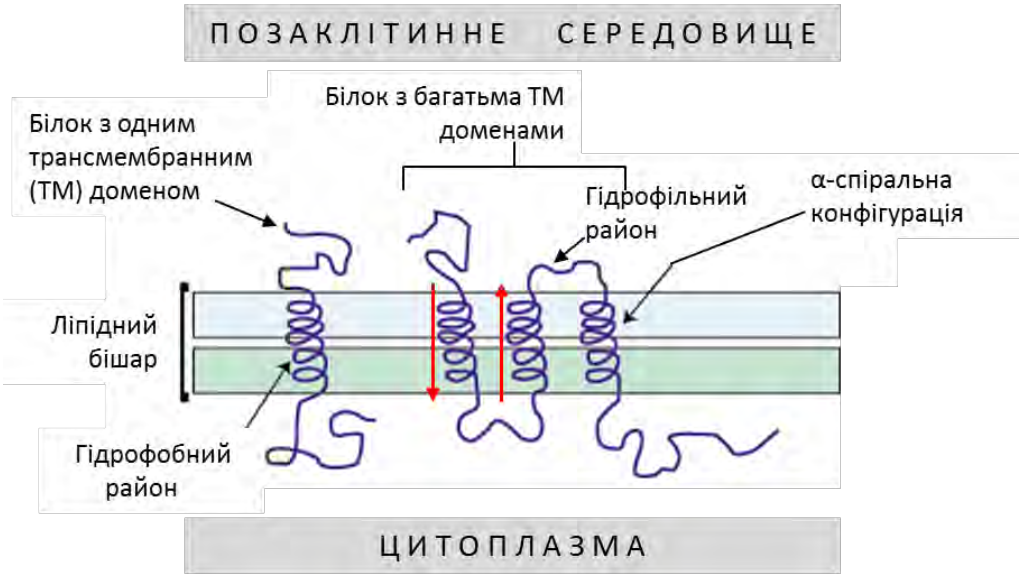


Рис. 5.29. Білки з одним або декількома трансмембранними (ТМ) доменами. Високий ступінь гідрофобності ТМ-домена – визначальна риса їхнього організування. Переважна вторинна структура ТМ-доменів – це α-спіралі. Червоні стрілки позначають дві спіралі, що мають протилежну орієнтацію

Переважно ТМ-райони мають α-спіральну конформацію. ТМ α-спіралі значно простіше відрізнити від α-спіралей глобулярних (цитозольних) білків, оскільки перші містять непропорційно велику кількість гідрофобних амінокислотних залишків (рис. 5.30). Такий композиційний зсув у вживанні амінокислот у межах ТМ доменів пов’язаний з обмеженнями, які накладають ліпідні мембрани на взаємодії з білком. Цей “гідрофобний сигнал” настільки виразний, що дає змогу з високою надійністю передбачати трансмембранні ділянки білків із первинної послідовності. Важливим фактом є також орієнтація

ТМ домена, тобто чи він скерований назовні клітини, чи всередину (див. рис. 5.29). Орієнтація ТМ спіралей визначає загальну топологію білка. Відомо, що позитивно заряджені залишки аргініну та лізину відіграють важливу роль у визначенні орієнтації, оскільки вони, здебільшого, розміщені у нетрансмембранних сегментах білка (петлях) його цитоплазматичної частини.

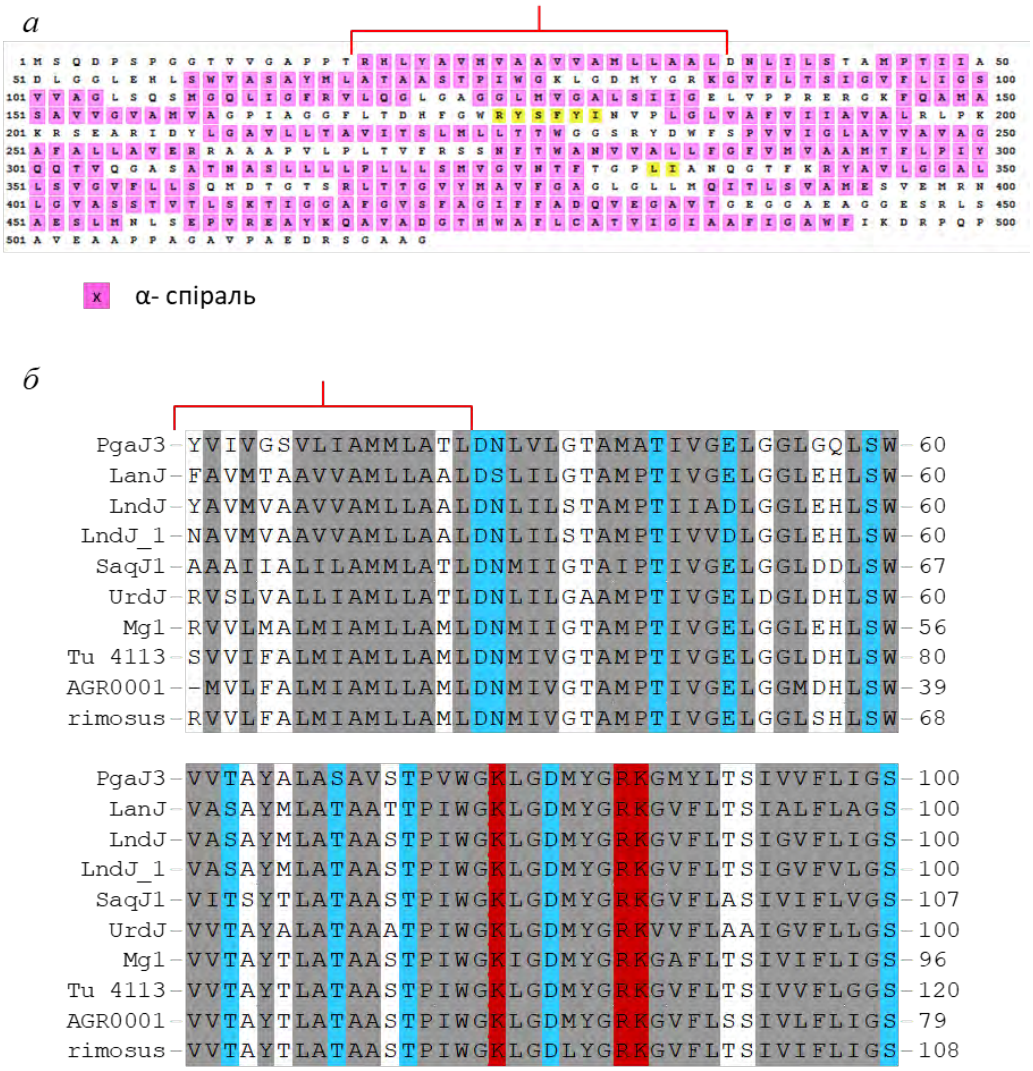


Рис. 5.30. Вторинна структура (*а*) протон-залежного антипортера LndJ (GI:82571027) згідно з PsiPred; α-спіральні ділянки позначені рожевим. Як видно з множинного вирівнювання LndJ-подібних білків (*б*), ці ділянки збагачені на гідрофобні залишки (виділені сірим). Одну з α-спіралей позначено червоною дужкою в *а* і *б*. Вирівнювання: програма MUSCLE на www.ebi.ac.uk; кольорове редагування: http://www.bioinformatics.org/SMS/multi_align.html

Цей та інші відомі факти про особливості амінокислотного складу різних частин ТМ білків використані для побудови HMM, що є в основі TMHMM. Автори програми припускають, що амінокислотні залишки ТМ білка можуть бути розташовані у: 1) ТМ спіралях; 2) “кепах” (сегменти на межі виходу спіралі з мембрани); 3) петлях. Оскільки розподіл частот різних амінокислотних залишків у цитоплазматичній і позаклітинній частинах білка різний, то HMM для ТМ білків ґрунтується на семи станах: один – для корової частини α-спіралі, два

– для кепів з боків спіралі, один – для петель з цитоплазматичного боку, по одному – для коротких і довгих петель на нецитоплазматичному боці, по одному – для глобулярних доменів посередині кожного типу петлі. Ймовірності емісії амінокислотних залишків усіх станів одного типу спряжені одна з одною, тобто їх оцінюють разом.

ТМ ділянку в TMHMM моделюють як послідовність α-спіралі (5–25 залишків), оточену двома кепами по п’ять залишків кожен (рис. 5.31, *а*). Очікують, що ТМ ділянка становитиме 15÷35 залишків завдовжки. Кепи, що обрамлюють спіраль, мають свої розподіли емісії амінокислот (один розподіл для цитоплазматичного боку, інший – для нецитоплазматичного), але вони і спіральна ділянка позначені однаково. Модель містить два набори ТМ станів для спіралей, що ідуть всередину і назовні клітини; всі їхні параметри дзеркально відображені і пов’язані між собою.

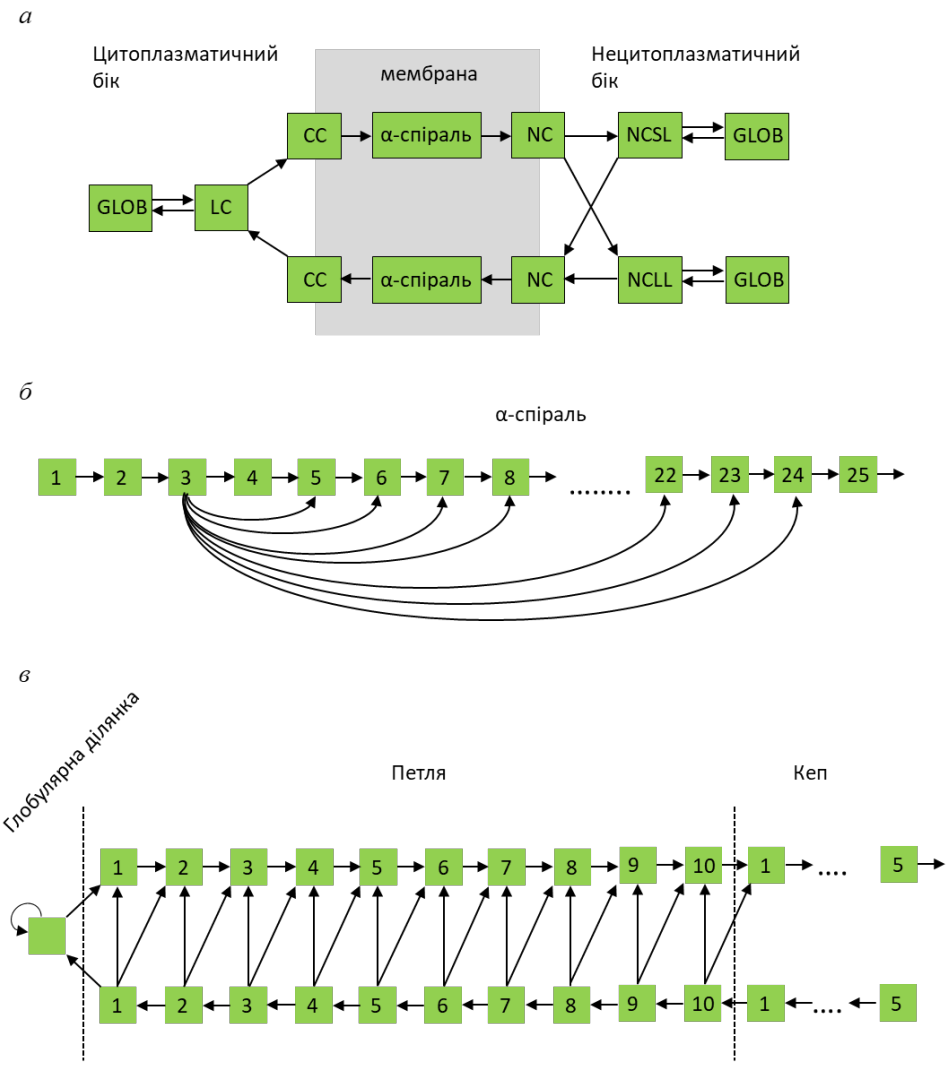


Рис. 5.31. Архітектура моделі, використана в TMHMM: *а* – загальний вигляд моделі (GLOB – глобулярна ділянка; LC – цитоплазматична петля; CC – цитоплазматичний кеп; NC – нецитоплазматичний (НЦ) кеп; NCSL –НЦ коротка петля; NCLL – НЦ довга петля); *б* – діаграма станів гідрофобного району, що моделює α-спіраль від п’яти до двадцяти п’яти залишків; *в* – діаграма станів глобулярного, петлевого і кепового районів (ділянок)

У моделі ТМ району – 21 змінна ймовірність переходу (з третього стану в наступні; див. рис. 5.31, *б*), що відображає варіабельність довжини α-спіралі. Оскільки сума всіх переходів має становити одиницю, то ці переходи формують 20 вільних параметрів, які потрібно оцінювати у процесі "тренування" НММ. Петлі між спіралями моделюють як модулі, що містять 2×10 станів у конфігурації драбини, і один стан замкнутий на себе (див. рис. 5.31, *в*). Основа такої архітектури моделі побудована на припущенні, що перші десять станів (амінокислотні залишки) мають містити основний топогенний сигнал (зміна частот вживання амінокислот). Водночас великі глобулярні домени моделюють простіше, у вигляді одного замкнутого на себе стану, що має рівномірніший розподіл емісії амінокислотних залишків. Довгі петлі з нецитоплазматичного боку мають властивості, відмінні від таких коротких петель. По всій довжині нецитоплазматичних коротких петель поодинці розкидані позитивно заряджені амінокислотні залишки, а в довгих петлях, що містять глобулярні домени, такого не спостерігається. Тому модель нецитоплазматичних петель має паралельну архітектуру двох різних шляхів (див. рис. 5.31, *в*). Кожен з цих шляхів (модулів) має 21 змінну ймовірність переходу. Сумарна кількість вільних параметрів усієї моделі – 216 (7×19 + 20 + 3×21). Сконструйовану модель тренують на наборах відомих ТМ білків для оцінення вільних параметрів (ймовірностей станових переходів та емісії амінокислотних залишків станами). Сьогодні ТМНММ – один із найнадійніших методів передбачення ТМ доменів, що ґрунтується на аналізі однієї заданої дослідником амінокислотної послідовності. Типовий графічний результат передбачення наведено на рис. 5.32 для білка LndJ, (його використано як приклад особливостей структурного організування ТМ білків; див. рис. 5.30).

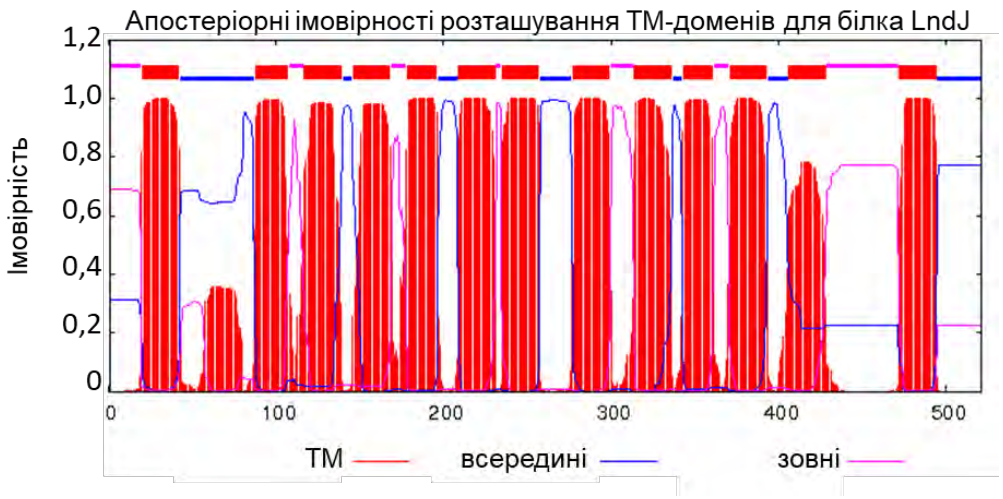


Рис. 5.32. Топологія ТМ доменів білка LndJ. ТМ домени позначені червоним, ділянки білка на цитоплазматичному боці – синім, на нецитоплазматичному – рожевим кольорами. Вісь абсцис – довжина білка, у амінокислотних залишках, вісь ординат – достовірність передбачення. Загалом програма з високою надійністю дає змогу передбачити в LndJ наявність 13 ТМ районів

5.6.7. MSAProbs – множинне вирівнювання послідовностей. Адреса веб-сервісу: http://toolkit.tuebingen.mpg.de/msaprobs. Метод базується на прогресивному принципі вирівнювання, що описано попередньо на прикладі програми ClustalW. Усі послідовності спочатку попарно вирівнюють, виявляють найподібнішу пару, до якої вирівнюють наступну найподібнішу, і т. д. Ключовий

етап множинного вирівнювання програмою MSAProbs – обчислення матриці відстаней між усіма парами послідовностей. У випадку ClustalW така матриця ґрунтується на рахунках попарного BLAST-подібного вирівнювання, а в MSAProbs – на апостеріорних ймовірностях вирівнювання, що обчислюють для всіх можливих пар послідовностей. Ці ймовірності виводять на основі двох моделей – функції розподілення (partition function) і *HMM-пари* (pair HMM). HMM-пара – окремий тип моделей Маркова, який застосовують у багатьох біоінформатичних сервісах, отож його розглянемо детальніше. Отже, на відміну від традиційних HMMs, які генерують (емітують) одну послідовність, HMM-пара генерує вирівняну пару послідовностей. Просту HMM-пару, що репрезентує вирівнювання двох послідовностей x та z, зображено на рис. 5.33. HMM-пара ґрунтується на трьох станах – I_x, I_z та M. Стани I_x та I_z емітують єдиний невирівняний символ у послідовностях x та z, а стан M генерує вирівняну пару символів. Наприклад, розглянемо вирівнювання між послідовностями $x=x_1x_2x_3x_4x_5x_6$ = TTCGG і z = $z_1z_2z_3z_4z_5z_6$ = ACCGAG. Припускаємо, що прихований шлях станів $y = MI_xMMMI_zI_z$. Тоді x_1 та z_1 спільно емітуються станом M, залишок x_2 емітується окремо станом I_x (тому не має пари у послідовності z), і т. д. Отже, HMM-пару можна використати для оцінювання достовірності попарних вирівнювань і відбору оптимальних пар.

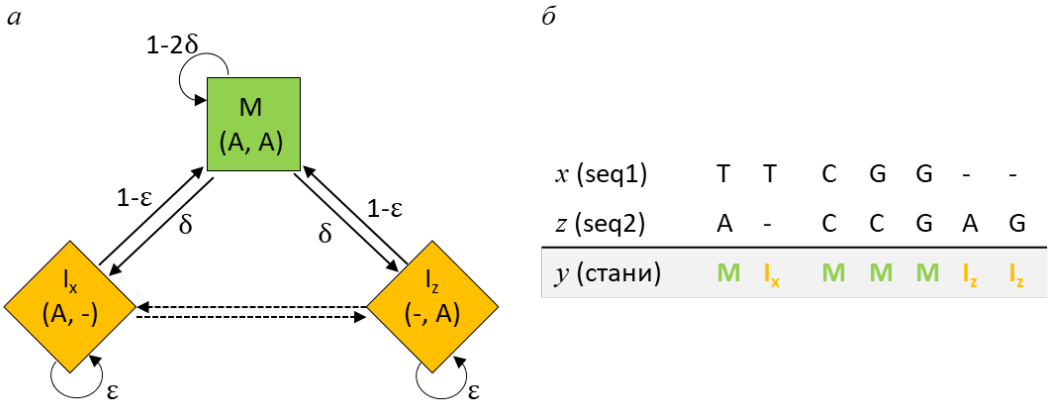


Рис. 5.33. Концептуальна схема HMM-пари (*а*) і послідовність станів y, що вона емітує (*б*). Кожен стан визначає пару символів у послідовностях x та z, зумовлюючи їхнє попарне вирівнювання. Стрілки між станами HMM-пари вказують на можливі переходи. Зауважте, що перехід між двома станами I_x та I_z – необов'язковий (його позначено пунктирною стрілкою), причому в деяких моделях його немає

Потім ці попарні вирівнювання за прогресивним принципом укладають у множинне вирівнювання й обчислюють за допомогою HMM-пари апостеріорну ймовірність для кожної колонки вирівнювання. Керуючись цими значеннями, можна відкинути колонки, адже їхнє вирівнювання ненадійне (для них, зазвичай, характерні низькі величини ймовірності), і виконати повторне вирівнювання, використовуючи тепер лише "надійні" сегменти послідовностей.

Перевагою HMM-опосередкованих методів множинного вирівнювання над традиційними підходами є те, що перші дають змогу обчислити ймовірність вирівнювання пари послідовностей. Це найдоречніше тоді, коли послідовності не виявляють сильної подібності, тому складно знайти коректне вирівнювання, що

має біологічний зміст. У таких випадках корисніше обчислити ймовірність того, що дві послідовності споріднені, замість пошуку єдиного найкращого їхнього вирівнювання.

Після обчислення матриці відстаней, MSAProbs будує дерево-провідник і обчислює вагу кожної послідовності; трансформує матриці апостеріорних імовірностей попарних вирівнювань згідно з особливою статистичною процедурою; будує прогресивне множинне вирівнювання відповідно до топології дерева-провідника, керуючись трансформованими матрицями. Для подальшого вдосконалення точності вирівнювання програма повторно вирівнює послідовності, точніше HMM-профілі, що походять з них. З цією метою увесь масив послідовностей випадково розбивають на два підмасиви, що не перекриваються, і далі виконують вирівнювання двох профілів, які побудовані на підмасивах.

5.6.8. Програма виявлення кластерів генів вторинного метаболізму у геномах – antiSMASH. Адреса сервісу: http://antismash.secondarymetabolites.org/. Вихідним матеріалом для аналізу слугує анотована послідовність геному у форматі GenBank, або амінокислотні послідовності. Тут геном (сукупність продуктів його транслювання) розглядають як систему, що складається з двох станів: перший – білки первинного метаболізму; другий – білки вторинного, або спеціалізованого. Веб-сервіс, аналізуючи амінокислотні послідовності, приписує кожен білок до одного з двох станів. Розглянемо сценарій пошуку кластерів генів вторинного метаболізму докладніше. Дослідник задає номер доступу до послідовності геному, що його цікавить, а програма шукає у ньому продукти трансляції генів, що є *маркерними* для кожного типу вторинного метаболізму. Наприклад, для біосинтезу всіх відновлених полікетидних антибіотиків (макроліди, поліефіри тощо) необхідні гени модульних (I типу) полікетидсинтаз. Ці гени притаманні виключно біосинтезу відновлених полікетидів. Авторами обрано з бази Pfam (див. пункт 5.6.3) вже наявні у ній HMM-профілі, що моделюють низку маркерних білків. Також на основі множинних вирівнювань створено HMM-профілі для тих описаних і схарактеризованих білків вторинного метаболізму, що досі не репрезентовані у Pfam. Так створили бібліотеку HMM-профілів, які охоплюють усі відомі сьогодні типи вторинного метаболізму. Програма HMMER3 (див. пункт 5.6.1) у заданому геномі здійснює пошук послідовностей, які подібні до тих, що становлять бібліотеку маркерних генів. Виявлення маркерних послідовностей далі супроводжується аналізом їхнього кластерного організування, порівнянням із описаними кластерами генів вторинного метаболізму і передбаченням хімічної структури. Цей вебсервіс швидко і вичерпно виявляє у геномах кластери генів біосинтезу полікетидів і нерибосомних пептидів, оскільки вони мають особливі генетичні детермінанти, які не характерні основному (первинному) метаболізму. Водночас виявлення деяких класів кластерів генів синтезу антибіотиків, передусім тих, що базуються на вуглеводних каркасах, меж генних кластерів – досі залишається складним завданням. Подальша розбудова сервісу сприятиме ліквідації цих недоліків і з часом має настати “плато” – момент, коли абсолютна більшість типів метаболізму буде репрезентована у бібліотеці маркерних послідовностей цієї програми.

Описані вище вебсервіси не вичерпують усієї різноманітності HMM- опосередкованих підходів до вивчення біологічних проблем. Наприклад, HMMs знайшли своє застосування у моделюванні процесу синтезу білка (http://dx.doi.org/10.1016/j.cell.2013.05.049)

Типові задачі до розділу 5

Задача 5.1. Наведено множинне вирівнювання 21 амінокислотної послідовності. Перетворіть його у паттерн. Зверніть увагу, що дві позиції множинного вирівнювання можуть містити будь-які амінокислотні залишки, і їх у вирівнюванні позначено як “x’’.

```
APL_ARATH/31-91          WTVELHERFVAAxxVAQL
APRR2_ARATH/293-352      WTPELHKKFVAAxxVEQL
ARR10_ARATH/180-239      WTHELHNKFLAAxxVDHL
ARR12_ARATH/192-251      WTVELHKKFVAAxxVNQL
ARR14_ARATH/197-256      WSIELHQQFVAAxxVIKL
ARR18_ARATH/191-250      WSQELHQKFVAAxxVQQL
ARR1_ARATH/234-293       WSVELHQQFVAAxxVNQL
ARR20_ARATH/207-272      WTPELHHKFEAAxxVFKM
ARR2_ARATH/213-272       WSVELHQQFVAAxxVLQL
AS1_ARATH/1-53           WSGEEDALLRAAxxVRQF
AS1_ARATH/54-108         KLTEEEQRLVAAxxQTKH
AZPA6_ASPTN/84-138       WTEEEDERLGAAxxVYHH
BAS1_YEAST/111-165       WTQEEDEQLLAAxxYMEH
BAS1_YEAST/166-218       WTLEEDLNLIAAxxVKAY
BFD1_TOXGM/969-1023      WSAEEDAMILAAxxQCEL
CEF1_ASPFU/57-106        WSREEDEKLLAAxxAPLM
CEF1_CANAL/1-55          WTNEEDEILKAAxxIGKY
CEF1_CANAL/56-109        WTKEEDEKLLAAxxHKIF
CEF1_CANGA/1-59          WTNEEDQILKAAxxVHKY
CEF1_CANGA/62-109        FTKEEDKKLLAAxxVVTL
CEF1_CRYNB/2-57          WRNEEDEILKAAxxISKY
```

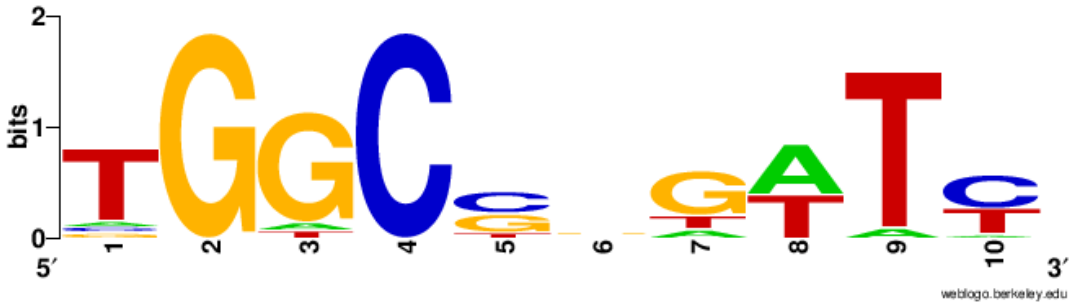
Розв’язання. Згідно з усталеним синтаксисом паттернів (див. основний текст), це множинне вирівнювання можна репрезентувати так:

[W,K,F]-[T,S,L,R]-{W,Y,D,E,G,F,S,C}-E-[L,E]-[H,D,E]-[E,K,N,Q,H,A,L]-[R,K,Q,L,N,M,I]-[F,L,I]-[V,L,E,R,G,K,I]-A×2-x2-[V,Q,Y,A,I, H]-{W}-[Q,H,K,E,A,L,I,T]-[L,M,F,H,Y]

Задача 5.2. Задано масив десятинуклеотидних послідовностей, які розпізнає транскрипційний фактор AdpA. Побудуйте ДНК-логотип на основі цього масиву. Поясніть, чому максимальне значення осі ординат сягає 2 біти. Які позиції логотипу найконсервативніші? Які найменш консервативні? Відповідь обґрунтуйте.

```
>adpA1
TGGCGGGTTC
>adpA2
TGACGCTTTT
>adpA3
TGGCGCGATC
>adpA4
TGGCCATTTC
>adpA5
TGGCCCTATT
>adpA6
TGGCCAGATT
>adpA7
TGGCGTGATT
>adpA8
TGGCCTGTTC
>adpA9
TGGCGGGATC
>adpA10
GGGCCGAATA
>adpA11
CGGCTGATTC
>adpA12
TGGCGCGATC
>adpA13
TGGCTGGTTC
>adpA14
TGTCCGGTAT
>adpA15
AGGCCGGATT
```

Розв'язання. Завантажуємо масив послідовностей у діалогове вікно вебсайту генерування логотипів WebLogo (https://weblogo.berkeley.edu/logo.cgi). Результат наведений внизу:



Графік за віссю ординат відображає інформаційний вміст однієї позиції (у бітах) досліджуваної послідовності, тут це ДНК. ДНК можна вважати текстом, побудованим на основі чотирилітерної абетки. Згідно з рівнянням Шеннона, одна позиція такого тексту може мати інформаційний вміст не більше 2 бітів. Найконсервативніші позиції – 2 і 4, тут лише одна літера трапляється. Найменш консервативна – позиція 6. Тут трапляються усі можливі 4 літери більш-менш рівномірно, тому сумарний інформаційний вміст дуже низький (див. логотип). Іншими словами, немає жодної закономірності у траплянні нуклеотидів у цій позиції. На противагу, наприклад, в позиції 1 – також трапляються усі 4 нуклеотиди, але, здебільшого, натрапляємо на тимін та аденін (вони зверху у стосі літер в цій позиції), отож загальна висота стоса літер вища, ніж у позиції 6.

Контрольні запитання до розділу 5

1. Що узагальнює амінокислотний паттерн?
2. Які етапи побудови позиційно-специфічної рахункової матриці?
3. Що таке псевдорахунок?
4. Поясніть значення основних елементів синтаксису паттернів.
5. Який принцип пошуку ДНК-мотивів за допомогою PSSM?
6. В яких базах даних використовують моделі на основі PSSM?
7. Які переваги має PSSM порівняно з паттерном?
8. Які стани налічує профіль?
9. Що таке прихована модель Маркова?
10. Що таке прихований шлях станів і видимий шлях символів у прихованих моделях Маркова?
11. Чи можна вирівнювати приховані моделі Маркова між собою?
12. Чи можна вирівнювати приховані моделі Маркова з послідовностями?
13. Чи можна вирівнювати приховані моделі Маркова з PSSM?
14. Які бази даних ґрунтуються на прихованих моделях Маркова?
15. Для яких завдань можна використовувати вебсервіс Weblogo?
16. Які елементи вторинної структури білків передбачає програма TMHMM?
17. Для чого використовують програму HHPred?
18. Для чого використовують програму GeneMark?
19. Для чого використовують програму antiSMASH?

Список літератури

1. Higgs P. G. Bionformatics and Molecular Evolution / P. G. Higgs, T. K. Attwood. – Oxford : Blackwell Publishing, 2005. – 398 p.
2. Durbin R. Biological Sequence Analysis. Probabilistic Models of Proteins and Nucleic Acids / R. Durbin, S. Eddy, A. Krogh, G. Mitchison. – Cambridge : Cambridge University Press, 1998. – 371 p.
3. Borodovsky M. Problems and Solutions in Biological Sequence Analysis / M. Borodovsky, S. Ekisheva. – Cambridge : Cambridge University Press, 2006. – 362 p.
4. Allman E. S. Mathematical Models in Biology. An Introduction / E. S. Allman, J.A. Rhodes. – Cambridge : Cambridge University Press, 2003. – 386 p.
5. Baxevanis A. D. Bioinformatics: a practical guide to the analysis of genes and proteins, 2nd Ed / A. D. Baxevanis, B. F. F. Ouellette. – New York : John Wiley & Sons, 2001. – 455 p.
6. Attwood T. K. Introduction to bioinformatics / T. K. Attwood, D. J. Parry-Smith. – London : Prentice Hall, 1999. – 218 p.
7. Bishop M. J. DNA and protein sequence analysis: a practical approach / M. J. Bishop, C. J. Rawlings. – Oxford : IRL Press, 1997. 352 p.
8. Anisimova M. Investigating protein-coding sequence evolution with probabilistic codon substitution models / M. Anisimova, C. Kosiol // Mol. Biol. Evol. – 2009. – Vol. 26. – P. 255–271.
9. Eddy S. R. What is a hidden Markov model? / S. R. Eddy // Nat. Biotechnol. – 2004. – Vol. 22. – P. 1315–1316.
10. Mrázek J. Finding sequence motifs in prokaryotic genomes – a brief practical guide for a microbiologist / J. Mrazek // Brief. Bioinform. – 2009. – Vol. 10. – P. 525–536.
11. Schneider A. Empirical codon substitution matrix / A. Schneider, G. M. Cannarozzi, G. H. Gonnet // BMC Bioinformatics. – 2005. – Vol. 6. – P. 134.
12. Schneider T. D. Sequence logos: a new way to display concensus sequences / T. D. Schneider, R. M. Stephens // Nucleic Acids Res. – 1990. – Vol. 18. – P. 6097–6100.
13. Korf I. An essential guide to the basic local alignment search tool (BLAST) / I. Korf, M. Yandell, J. Bedell. – Cambridge, MA : O'Reilly, 2003. – 22 p.

Предметний покажчик

А

Б

В

Г

Д

Е

Ж

З

І

К

Л

М

Н

О

П

Р

С

Т

У

Ф

Ц

Ч

Ш

Навчальне видання

ОСТАШ Богдан Омелянович

Біоінформатика: аналіз генетичних послідовностей

Електронний підручник

Редактори: *Людмила Макітринська, Ірина Лоїк*

Комп'ютерна верстка та обкладинка: *Богдан Осташ*

Формат 60x84/8.Ум.друк.арк. 26,97

Львівський національний університет імені Івана Франка,
вул. Університетська, 1, м. Львів, 79000
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру видавців, виготівників
і розповсюджувачів видавничої справи.
Серія ДК № 3059 від 13.12.2007 р.

Осташ Богдан

О 76 Біоінформатика: аналіз генетичних послідовностей : електронний підручник / Богдан Осташ. – Львів : ЛНУ ім. Івана Франка, 2022. – 232 с.
ISBN 978-617-10-0729-1

Електронний підручник ознайомлює з практичними знаряддями комп'ютерного аналізу нуклеотидних та амінокислотних послідовностей, а також подає стислий опис математичних та статистичних концепцій, що є теоретичним підґрунтям цих знарядь. Розглянуто сучасні бази біомедичних даних та основні математичні моделі, що застосовують в аналізі генетичних послідовностей. У наступних трьох розділах викладено методи біоінформатики: попарні й множинні вирівнювання, моделі на основі вирівнювань (зокрема, позиційно-вагові матриці й приховані моделі Маркова).

Для студентів та науковців природничого та медико-біологічного профілів.

УДК [575.112:004](075.8)

Про малюнок на палітурці підручника

Ще 1871 року Ч. Дарвін дійшов висновку, що людина як вид еволюційно ближча до африканських мавп, ніж до будь-якого іншого виду, що існує на Землі. Секвенування ДНК горили, бонобо та шимпанзе підтверджує це припущення і змальовує чітку картину "родинних" зв'язків (їх часто зображують у вигляді філогенетичного дерева – на кшталт фонового малюнка палітурки). Зокрема, шимпанзе і бонобо є нашими найближчими родичами: геноми цих приматів і *Homo sapiens* ідентичні на 99 %. Геном горили на 98 % ідентичний людському. Ці, на перший погляд, мізерні вкраплення неспорідненої ДНК є критичними для людини як виду, даючи нам змогу, зокрема, ходити на двох кінцівках, писати вірші чи планувати міжпланетні подорожі. Ми досі дуже мало розуміємо, як та частка генома, що притаманна лише людям, робить нас людьми. Але науковці розробили низку знарядь, які ґрунтуються на методах математики, теорії ймовірності й інформаційних технологій, що дають змогу порівнювати й аналізувати гени та цілі геноми. Зокрема, порівнюючи одну третю усіх білок-кодувальних послідовностей у геномах людини і шимпанзе (кольоровий квадрат з точок на палітурці), можна побачити, що відмінності ці незначні, але вони "пронизують" весь геном, впливаючи у різний спосіб на кожну нашу хромосому.

У квадраті кожна точка репрезентує попарне вирівнювання послідовностей розміром 500 тисяч пар нуклеотидів, що кодують білки у геномах людини і шимпанзе. Колір точки ілюструє міру подібності послідовності цього сегмента генома людини до послідовності з шимпанзе: чим темніший колір, тим менше подібності. Для компактної репрезентації великого масиву даних обрано криву Гільбертона. Зображення відтворено з дозволу автора, д-ра М. Кшивінскі (©Martin Krzywinski; martink@bcgsc.ca), Центр досліджень раку в Британській Колумбії, Канада. Повну версію зображення і його опис подано у вересневому випуску журналу Scientific American за 2014 р.

ISBN 978-617-10-0729-1