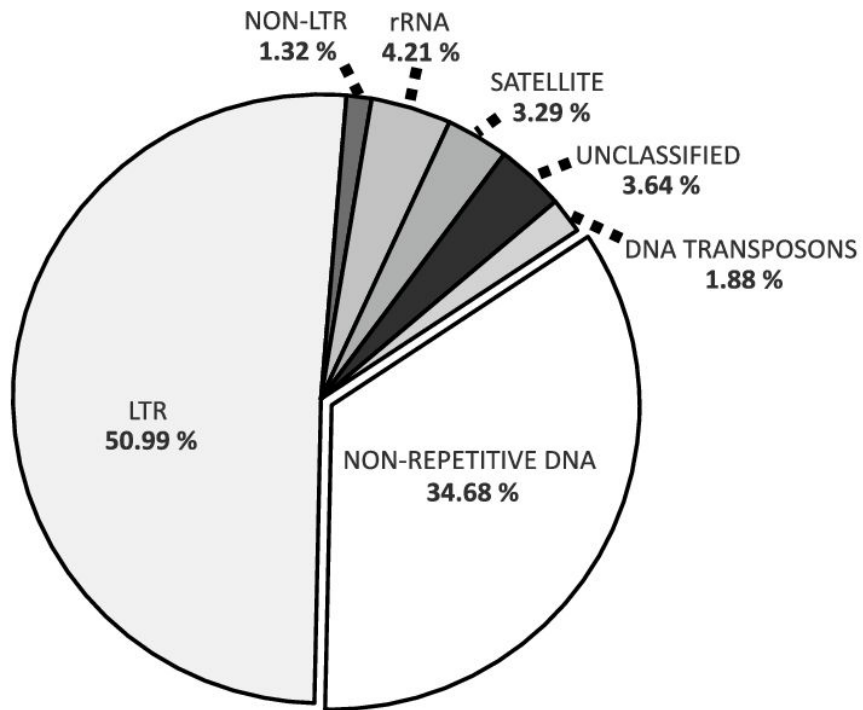




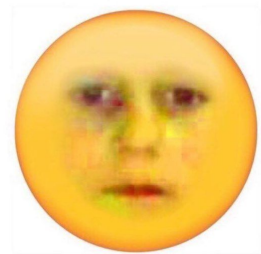
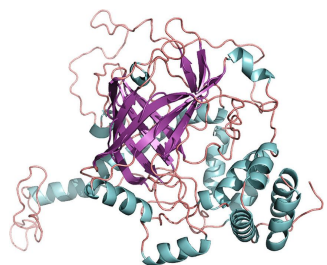
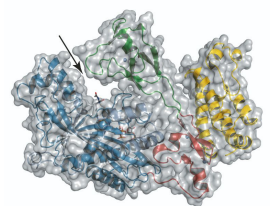
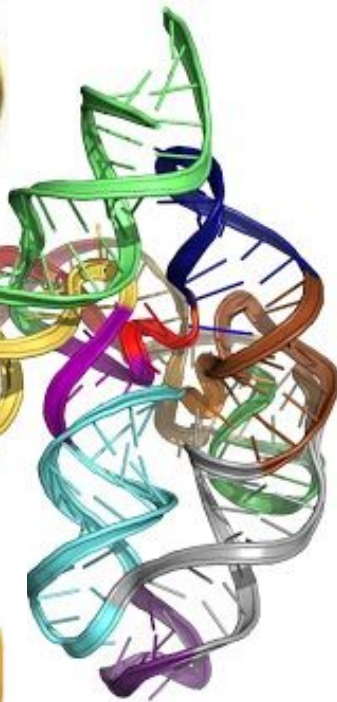
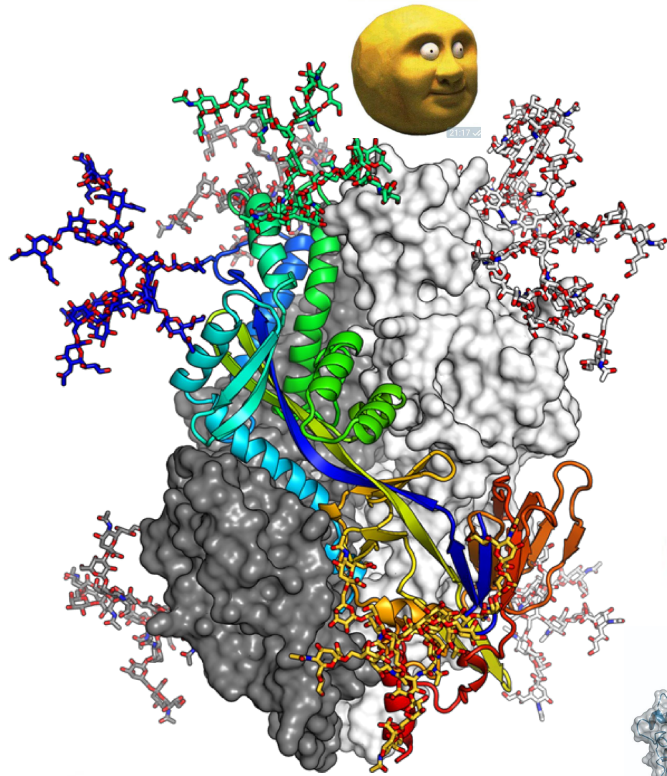
машаеж копирайтед

зачем сравнивать макромолекулы?

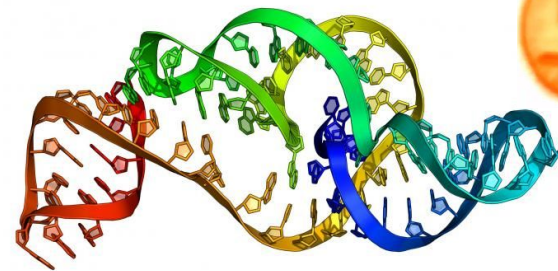
1. макромолекулы близкородственных организмов (чаще всего) больше похожи друг на друга, чем макромолекулы дальних родственников. измеряя их схожесть, можно предположить насколько **далеки друг от друга организмы**
2. если нам попалась новая неизвестная макромолекула, можно сравнить ее с известными. и если среди макромолекул с известной функцией найдется похожая, **можно предположить функцию неизвестной**
3. сравнивать макромолекулы можно не только между организмами, но и внутри одного. например, внутри тандемного повтора куски ДНК одинаковые. так можно **найти новые тандемные повторы или ретротранспозоны (LINEs и SINEs)**
4. *секвенирование и картирование ридов :*



LTR - длинные концевые повторы,
фланкируют гены



НО как сравнивать вот
это?





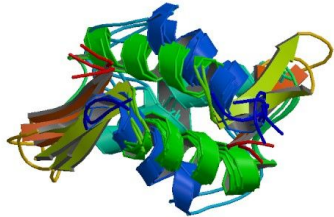
есть ВЫХОД

предположим, что:

похожие последовательности -> похожие 3D структуры с похожими функциями.

тогда сравнивать придется не 3D структуры (непонятно как), а последовательности нуклеотидов или аминокислот

Alanine	A
Arginine	R
Asparagine	N
Aspartic acid	D
Cysteine	C
Glutamic acid	E
Glutamine	Q
Glycine	G
Histidine	H
Isoleucine	I
Leucine	L
Lysine	K
Methionine	M
Phenylalanine	F
Proline	P
Serine	S
Threonine	T
Tryptophan	W
Tyrosine	Y
Valine	V
Selenomethionine*	X



HUMAN	KKASKPKKAASKAPTKKPKATPVKKAKKKLAATPKKAKPKTVKAKPVKASKPKKAKPVK
CHIMP	KKASKPKKAASKAPTKKPKATPVKKAKKKLAATPKKAKPKTVKAKPVKASKPKKAKPVK
MOUSE	KKAAKPKKAASKAPSKKPKATPVKKAKKKPAATPKKAKPKVVKVPVKASKPKKAKTVK
RAT	KKAAKPKKAASKAPSKKPKATPVKKAKKKPAATPKKAKPKIVKVPVKASKPKKAKPVK
COW	KKAAKPKKAASKAPSKKPKATPVKKAKKKPAATPKKTKKPKTVKAKPVKASKPKKTKPVK
	.**.*****.*****.***.***.*****.***.***

нуклеотиды и аминокислоты имеют однобуквенные обозначения - можно сравнить их! так проблема сравнения биологических макромолекул сводится к проблеме **сравнения строк**

как насчет того, чтобы пройтись по двум строчкам и побуквенно их сравнить?

```
counter = 0
```

```
for i, j in range zip(a, b):
```

```
    if i == j:
```

```
        counter +=1
```

```
print(counter)
```

G	A	G	C	C	T	A	C	T	A	A	C	G	G	G	A	T
C	A	T	C	G	T	A	A	T	G	A	C	G	G	C	C	T

расстояние Хэмминга

= количеству однонуклеотидных замен в двух последовательностях, мы же считаем обратную величину

basic решение: берем на вход 2 строки, сравниваем побуквенно, если буквы совпали, повышаем значение “похожести”, если нет, понижаем

как минимум 2 (две) проблемы:

1. как сравнить строки разной длины?

ЗАВТРА и ЗАВТРАК

2. неужели такое выравнивание кажется правильным?

ПРЕДЕЛ

||| |||

ДЕЛАТЬ

шэн :шэшо

**так и в наших биологических последовательностях есть не только
однобуквенные замены, есть еще вставки и делеции, как их обозначить?**

✧ ✧ ГЭПЫ ✧ ✧

добавляем еще один символ - “-”. это одновременно и вставка (в одном слове), и делеция (в другом слове)

тогда выравнивание слов ПРЕДЕЛ и ДЕЛАТЬ будет выглядеть вот так:

ПРЕДЕЛ - - -
| | | | |
- - - ДЕЛАТЬ

более хитрые случаи:

ПИПИДАСТР - - -
- - ПИ - - СТОЛЕТ

МАШИН -А
БА - - НКА

КАКОЕ ВЫРАВНИВАНИЕ ЛУЧШЕ?

1 ГУ -СЕН - ИЦА
 - УТРЕННИК -

2 ГУСЕНИЦА - - - - -
 - - - - - УТРЕННИК

3 ГУ - - СЕ -НИ -ЦА
 - УТР - ЕННИК -

4 ГУСЕНИЦА
 УТРЕННИК

что такое хорошо и что такое плохо

- насколько плохо и хорошо?

матрица замен для аминокислот (blosum62)

матрица замен для нуклеотидов

какой score у этой последовательности?

-2

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

A C T
A A G

✨ гэпы ✨ [2]

гэпы гораздо хуже мутаций => штраф за них больше

линейный штраф

любой гэп оценивается
штрафом в **-d**
(у нас по дефолту $d = 10$)

2 ГУСЕНИЦА -----
----- УТРЕННИК

аффинный штраф

вставка одного длинного участка
лучше вставки нескольких
коротких.

поэтому штраф = **-d - (l - 1)*e**,
где:

l - длина гэпа
 $e \ll d$

ACTTAG
A- T- AG

ACTTAG
A- - TAG

окей может мы уже сравнивать начнем??

какое качество у этого выравнивания? **-8**

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

AGCC
AT - C

гэп = -10

алгоритм Нидлмана-Вунша

моделька

представим наше выравнивание как путь в таблице:

		S	I	M	I	L	A	R	I	T	Y
P		↘									
I			↘	→							
L					↘						
L						↘					
A							↘				
R								↘	→	→	→

выравнивание:

SIMILARITY

PI-LLAR---

выравнивание и путь
взаимно однозначны

начинать будем в левой верхней пустой клетке, конец выравнивания - правая нижняя, а стрелки - путь, по которому мы идем.


добавим в модельку известные числа

		A	T	C	C	G	A	G	T	T
	0	-10	-20	-30	-40	-50	-60	-70	-80	-90
A	-10									
T	-20									
C	-30									
A	-40									
G	-50						$S(5, 6)$			
T	-60									
C	-70									

$S(i, j)$ - качество оптимального пути от клетки (0, 0) до клетки (i, j)

ATCCGAGTT
- - - - -


		A
	0	-10
A	-10	



в этой клетке находится
выравнивание, в котором есть
аденины из двух
последовательностей


- A
A -

		A
	0	-10
A	-10	




A
A

		A
	0	-10
A	-10	



A -
- A

		A
	0	-10
A	-10	



пустые клетки можно заполнять, если значения в 3ех соседних клетках известны:

$S(i-1, j-1)$ 	$S(i, j-1)$ 
$S(i-1, j)$ 	$S(i, j)$

т.е.:

$$S(i, j) = \max \begin{cases} S(i, j-1) - d \\ S(i-1, j-1) + s(x_i, y_j) \\ S(i-1, j) - d \end{cases}$$

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

(i, j) — соответствующая координата в таблице

$S(i, j)$ — значение для этой координаты в матрице замен

ось i

посчитаем первую ячейку (S (1,1)):

		A	T	C	C	G	A	G	T	T
	0	-10	-20	-30	-40	-50	-60	-70	-80	-90
A	-10	2 ↘								
T	-20									
C	-30									
A	-40									
G	-50									
T	-60									
C	-70									

матрица замен

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

$$S(i, j) = \max \begin{cases} S(i, j-1) - d = -10 - 10 = -20 \\ S(i-1, j-1) + s(x_i, y_j) = 0 + 2 = 2 \\ S(i-1, j) - d = -10 - 10 = -20 \end{cases}$$

ось j

$\max(-20, 2, -20) = 2 \Rightarrow$
записываем в ячейку 2
и ставим диагональную
стрелку (потому что
оттуда пришёл лучший
ответ)

посчитаем $S(2,1)$:

		A	T	C	C	G	A	G	T	T
	0	-10	-20	-30	-40	-50	-60	-70	-80	-90
A	-10	2 ↘ -8 →								
T	-20									
C	-30									
A	-40									
G	-50									
T	-60									
C	-70									

матрица замен

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

$$S(i, j) = \max \begin{cases} S(i, j-1) - d = 2 - 10 = -8 \\ S(i-1, j-1) + s(x_i, y_j) = -10 - 2 = -12 \\ S(i-1, j) - d = -20 - 10 = -30 \end{cases}$$

$\max(-8, -12, -30) = -8 \Rightarrow$
записываем в ячейку
-8, горизонтальная
стрелка

по итогу получаем заполненную матрицу:

		A	T	C	C	G	A	G	T	T
	0	→ -10	→ -20	→ -30	→ -40	→ -50	→ -60	→ -70	→ -80	→ -90
A	↓ -10	↘ 2	→ -8	→ -18	→ -28	→ -38	↘ -48	→ -58	→ -68	→ -78
T	↓ -20	↘ -12	↘ 4	→ -6	→ -16	→ -26	→ -36	→ -46	↘ -56	↘ -66
C	↓ -30	↘ -22	↓ -6	↘ 6	↘ -4	→ -14	→ -24	→ -34	→ -44	→ -54
A	↓ -40	↘ -28	↓ -16	↓ -4	↘ 4	↘ -5	↘ -12	→ -22	→ -32	→ -42
G	↓ -50	↓ -38	↓ -26	↓ -14	↘ -6	↘ 6	→ -4	↘ -10	→ -20	→ -30
T	↓ -60	↓ -48	↘ -36	↓ -24	↘ -15	↓ -4	↘ 4	↘ -6	↘ -8	↘ -18
C	↓ -70	↓ -58	↓ -46	↘ -34	↘ -22	↓ -14	↘ -6	↘ 2	↘ -7	↘ -9

как из заполненной матрицы получить оптимальное выравнивание?

traceback

идём назад по стрелкам, попутно выписывая получившееся выравнивание

		A	T	C	C	G	A	G	T	T
	0	→ -10	→ -20	→ -30	→ -40	→ -50	→ -60	→ -70	→ -80	→ -90
A	↓ -10	↘ 2	→ -8	→ -18	→ -28	→ -38	↘ -48	→ -58	→ -68	→ -78
T	↓ -20	↘ -12	↘ 4	→ -6	→ -16	→ -26	→ -36	→ -46	↘ -56	↘ -66
C	↓ -30	↘ -22	↓ -6	↘ 6	↘ -4	→ -14	→ -24	→ -34	→ -44	→ -54
A	↓ -40	↘ -28	↓ -16	↓ -4	↘ 4	↘ -5	↘ -12	→ -22	→ -32	→ -42
G	↓ -50	↓ -38	↓ -26	↓ -14	↘ -6	↘ 6	→ -4	↘ -10	→ -20	→ -30
T	↓ -60	↓ -48	↘ -36	↓ -24	↘ -15	↓ -4	↘ 4	↘ -6	↘ -8	↘ -18
C	↓ -70	↓ -58	↓ -46	↘ -34	↘ -22	↓ -14	↘ -6	↘ 2	↘ -7	↘ -9

наше выравнивание:

ATCCGACTT
AT - C- AGTC

↑
score выравнивания,
грубо говоря, то, насколько
хорошим оно получилось

самостоятельная работа




матрица замен

	C	T	A	G
C	2	-1	-2	-2
T	-1	2	-2	-2
A	-2	-2	2	-1
G	-2	-2	-1	2

$$S(i, j) = \max \begin{cases} S(i, j-1) - d \\ S(i-1, j-1) + s(x_i, y_j) \\ S(i-1, j) - d \end{cases}$$

способы оптимизации: по памяти

часто приходится сравнивать большие последовательности - для них не ок хранить в памяти всю таблицу $m \times n$. тем более, для подсчета качества выравнивания (score) она нам вся и не нужна. сколько строчек нужно для того, чтобы произвести все наши счетные операции?

$S(i-1, j-1)$ 	$S(i, j-1)$ 
$S(i-1, j)$ 	$S(i, j)$

ответ: 

их мы и будем хранить в двух списках!

способы оптимизации: по вычислительной сложности

мы не можем потерять вычисления без потери точности – тогда какие-то пути в таблице не будут пройдены. но точность можно потерять аккуратно:

допущение: если наше выравнивание очень плохое, нам не нужно знать, насколько именно – важен сам факт. тогда мы можем ограничить число гэпов так, чтобы такие возможные выравнивания не учитывались:

- - - - -
ATCAGTC

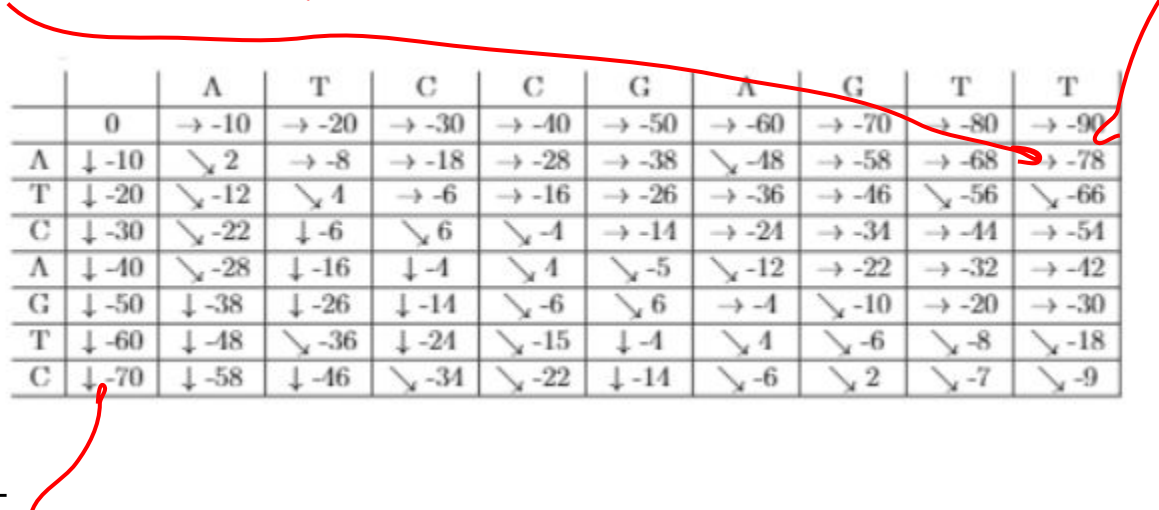
ATCCGAGTT
- - - - -

ATCCGAGTT
- - - - A - - -

заметим, что плохие
выравнивания встречаются в
верхнем правом и нижнем
левом углах таблицы:

ATCCGAGTT
- - - - - A - - - -

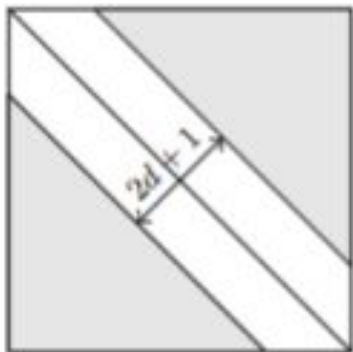
ATCCGAGTT
- - - - - - - - - -



		A	T	C	C	G	A	G	T	T
	0	→ -10	→ -20	→ -30	→ -40	→ -50	→ -60	→ -70	→ -80	→ -90
A	↓ -10	↘ 2	→ -8	→ -18	→ -28	→ -38	↘ -48	→ -58	→ -68	→ -78
T	↓ -20	↘ -12	↘ 4	→ -6	→ -16	→ -26	→ -36	→ -46	↘ -56	↘ -66
C	↓ -30	↘ -22	↓ -6	↘ 6	↘ -4	→ -14	→ -24	→ -34	→ -44	→ -54
A	↓ -40	↘ -28	↓ -16	↓ -4	↘ 4	↘ -5	↘ -12	→ -22	→ -32	→ -42
G	↓ -50	↓ -38	↓ -26	↓ -14	↘ -6	↘ 6	→ -4	↘ -10	→ -20	→ -30
T	↓ -60	↓ -48	↘ -36	↓ -24	↘ -15	↓ -4	↘ 4	↘ -6	↘ -8	↘ -18
C	↓ -70	↓ -58	↓ -46	↘ -34	↘ -22	↓ -14	↘ -6	↘ 2	↘ -7	↘ -9

- - - - -
ATCAGTC

-> можно их просто не учитывать!



все нежеланные значения теперь у нас
находятся в серых треугольниках – в них
мы напишем значения $= -\infty$

алгоритм будет идти в пределах белой
полосы шириной $2d + 1$, где d – количество
допустимых гэпов

он не будет
отфильтровывать все пути с
количеством гэпов $> d$, но
он фильтрует большую
часть и оставляет все пути,
где гэпов $< d$

		A	T	C	C	G
	0	-10	-20	-30	-40	-50
A	-10					
T	-20					
C	-30					
A	-40					
G	-50					

домашнее задание

нарисуйте таблицу для выравнивания последовательностей ATGAGTCTCT и CTGTCTCCTG, запишите score и оптимальное выравнивание

бонус: попробуйте решить ту же задачу с аффинными гэпами ($d = 10$, $e = 0,5$)

(бонус бонусом, но понимать принцип работы аффинного гэпа к следующему семинару хорошо бы всем)



спасибо за внимание!!

