

# Парное выравнивание

Алгоритмы в биоинформатике

Дмитрий Мелешко

[meleshko.dmitrii@gmail.com](mailto:meleshko.dmitrii@gmail.com)

# Что было на прошлой лекции?

- Транскрипция и трансляция.
- Свойство локальности ДНК.
- Можно считать расстояние между строками и делать выводы о свойствах организмов.
- Сравнивать участки генома можно достаточно эффективно.
- Расстояние редактирования

# Что будет на этой лекции?

- Определение выравнивания и веса выравнивания.
- Неравнозначные замены. Матрицы замен BLOSUM и PAM.
- Проблема гэпов. Определение аффинных штрафов за гэпы.

# Расстояние редактирования

GATTACA → GATTACA → GATTACA  
GATTACA → GATTCA → GAATTCA

# Попарное выравнивание

GATTACA

GAAATTCA



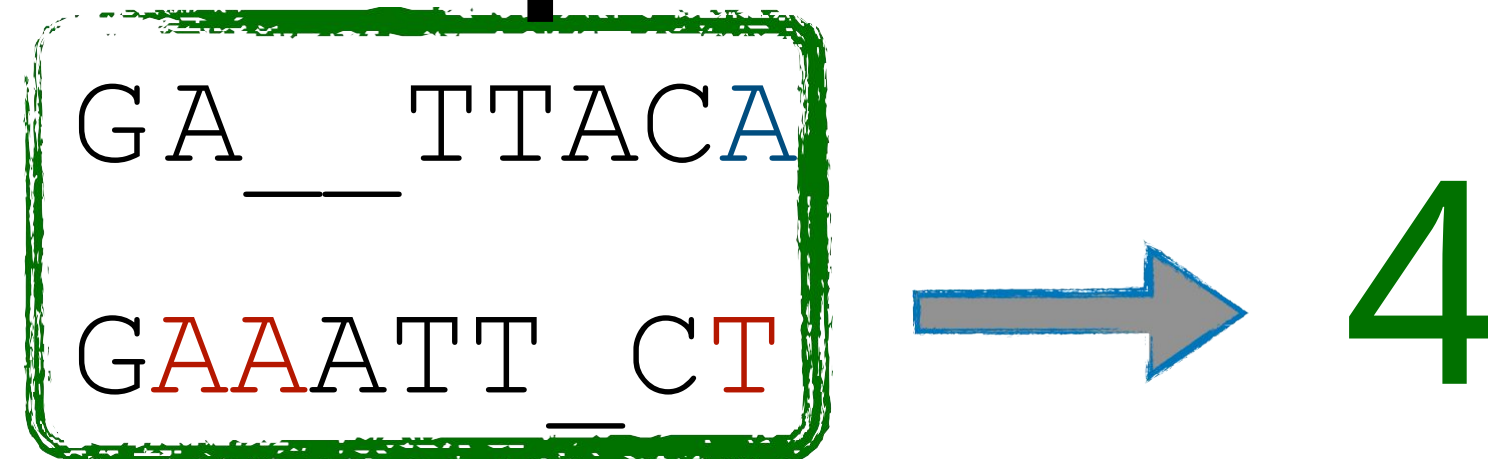
G \_ \_ A T T A C A  
G A A A T T \_ C T

Рассмотрим пару строк  $(a, b)$  где  $a_i, b_i \in \mathbb{A}$

Выравнивание — такая пара строк  $(a^*, b^*)$  где  $a_i^*, b_i^* \in (\mathbb{A} \cup \{\_ \})$ ,  
что

1.  $|a^*| = |b^*|$
2.  $a_i^* \neq \_$  или  $b_i^* \neq \_$
3. При удалении всех гэпов из  $a^*, b^*$  получаем  $a, b$

# Оптимальные выравнивания



Стоимостью (весом) выравнивания будем называть

$$|a^*|$$

$$W(a^*, b^*) = \sum_{i=1}^{|a^*|} w(a_i^*, b_i^*)$$

Где  $w(a_i^*, b_i^*)$  функция  $(\mathbb{A} \cup \{-\})^2 \rightarrow \mathbb{R}$

$\max_{a^*, b^*} W(a^*, b^*)$  - оптимальное выравнивание

# Система весов (Scoring system)

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

# Вес оптимального выравнивания

Чтобы посчитать  $D_w(a, b)$ , где  $|a| = n$ ,  $|b| = m$  построим матрицу  $D$ ,  $\text{Dim}(D) = (n + 1, m + 1)$  по следующим правилам:

- $D_{0,0} = 0$

- $D_{i,0} = D_{i-1,0} + w(a_i, \_)$

GAATCTAA  
G\_ATCTGA

- $D_{0,j} = D_{0,j-1} + w(\_, b_j)$

- $D_{i,j} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \text{ (замена)} \\ D_{i-1,j} + w(a_i, \_) \text{ (удаление)} \\ D_{i,j-1} + w(\_, b_j) \text{ (вставка)} \end{cases}$

GA   ATCTGAA   AA  
G\_   ATC**C**GAA   GA



# Вес выравнивания

$w(a, b) = 2$ , если  $a = b$

$w(a, b) = -1$ , если  $a \neq b$

$w(a, -) = -2$

$w(-, b) = -2$

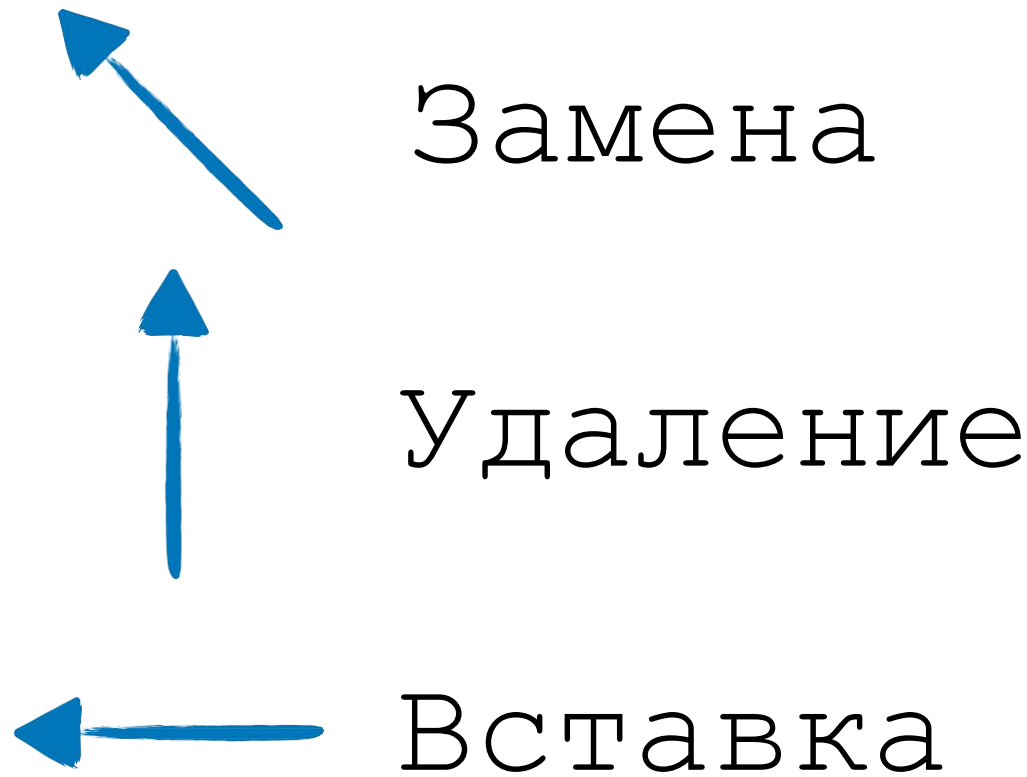


		G	A	T	T	A	C	A
	0	-2	-4	-6	-8	-10	-12	-14
A	-2	-1	0	-2	-4	-6	-8	-10
A	-4	-3	1	-1	-3	-5	-7	-9
G	-6	-2	-1	0	-2	-4	-6	-8
A	-8	-4	0	-2	-1	-3	-5	-7
G	-10	-6	-2	-1	-3	-2	-4	-6
T	-12	-8	-4	0	1	-1	-3	-5
A	-14	-10	-6	-2	-1	3	-1	-1
C	-16	-12	-8	-4	-3	-2	5	3

\_\_GATTACA  
AAGGTAC\_

# Алгоритм Needleman-Wunsch

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$

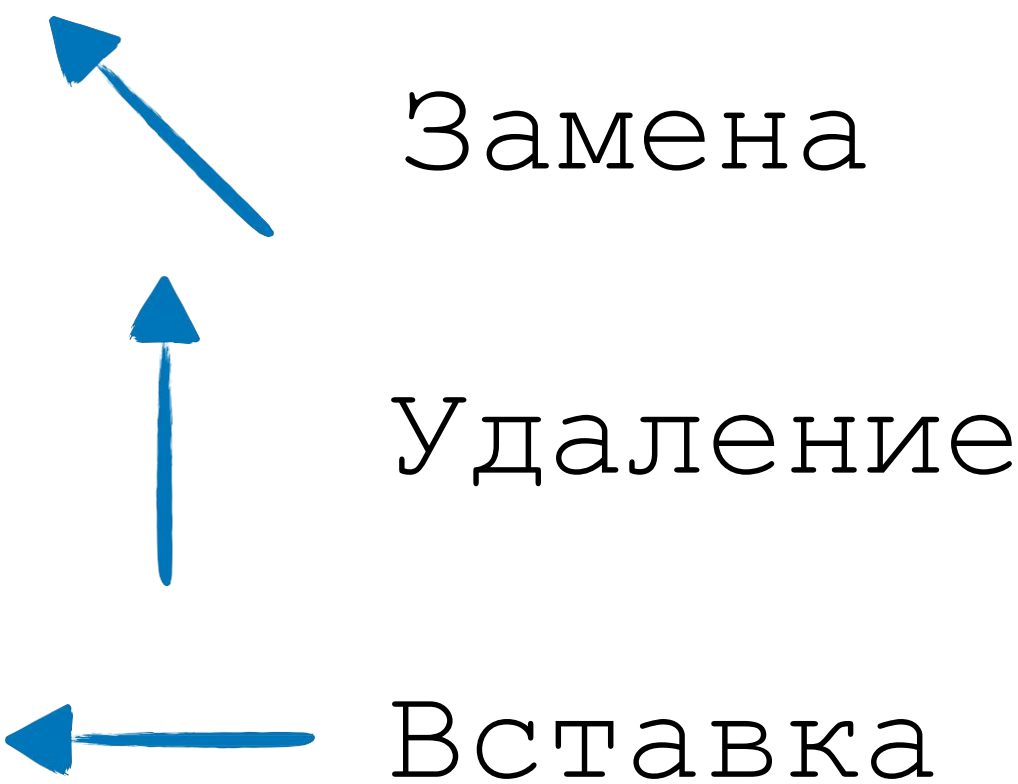


		G	A	T	T	A	C	A
		0	-2	-4	-6	-8	-10	-14
A		-2	-1	0	-2	-4	-6	-10
A		-4	-3	1	-1	-3	-5	-9
G		-6	-2	-1	0	-2	-4	-8
A		-8	-4	0	-2	-1	-3	-7
G		-10	-6	-2	-1	-3	-2	-6
T		-12	-8	-4	0	1	-1	-5
A		-14	-10	-6	-2	-1	3	-1
C		-16	-12	-8	-4	-3	-2	5

\_\_GATTACA  
AAGAGTAC\_

# Алгоритм Needleman-Wunsch (глобальное выравнивание)

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



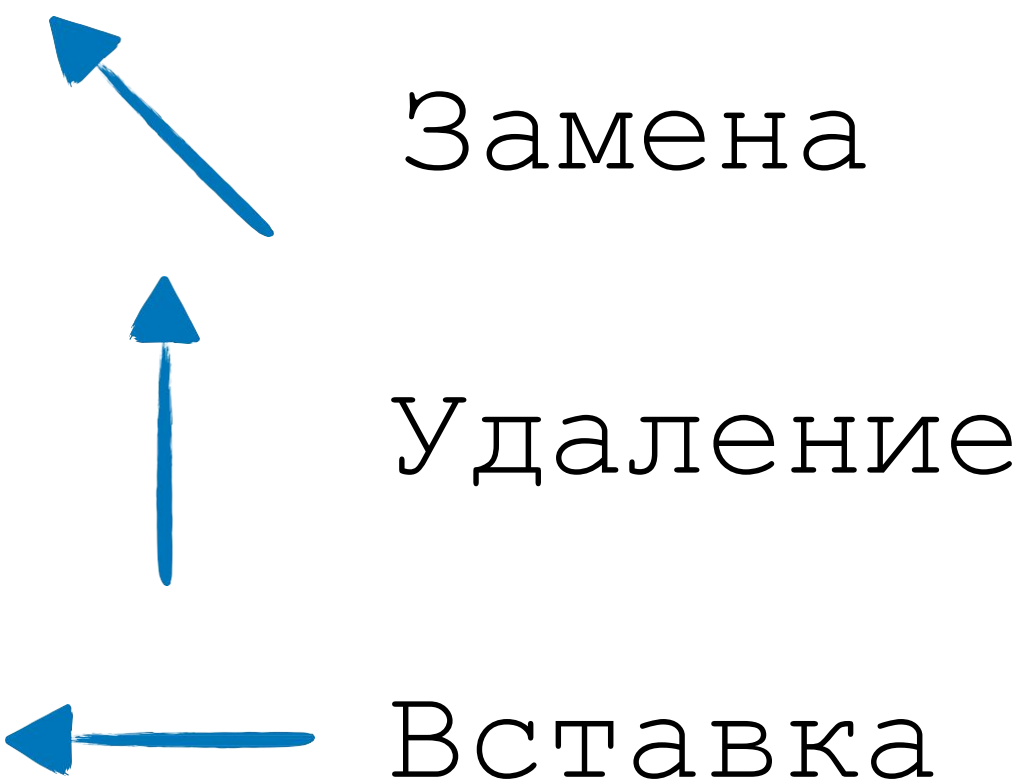
		G	A	T	T	A	C	A
	0	-2	-4	-6	-8	-10	-12	-14
A	-2	-1	0	-2	-4	-6	-8	-10
A	-4	-3	1	-1	-3	-5	-7	-9
G	-6	-2	-1	0	-2	-4	-6	-8
A	-8	-4	0	-2	-1	-3	-5	-7
G	-10	-6	-2	-1	-3	-2	-4	-6
T	-12	-8	-4	0	1	-1	-3	-5
A	-14	-10	-6	-2	-1	3	-1	-1
C	-16	-12	-8	-4	-3	-2	5	3

\_\_GATTAACA

AAGAGTAC\_

# Алгоритм Smith-Waterman (локальное выравнивание)

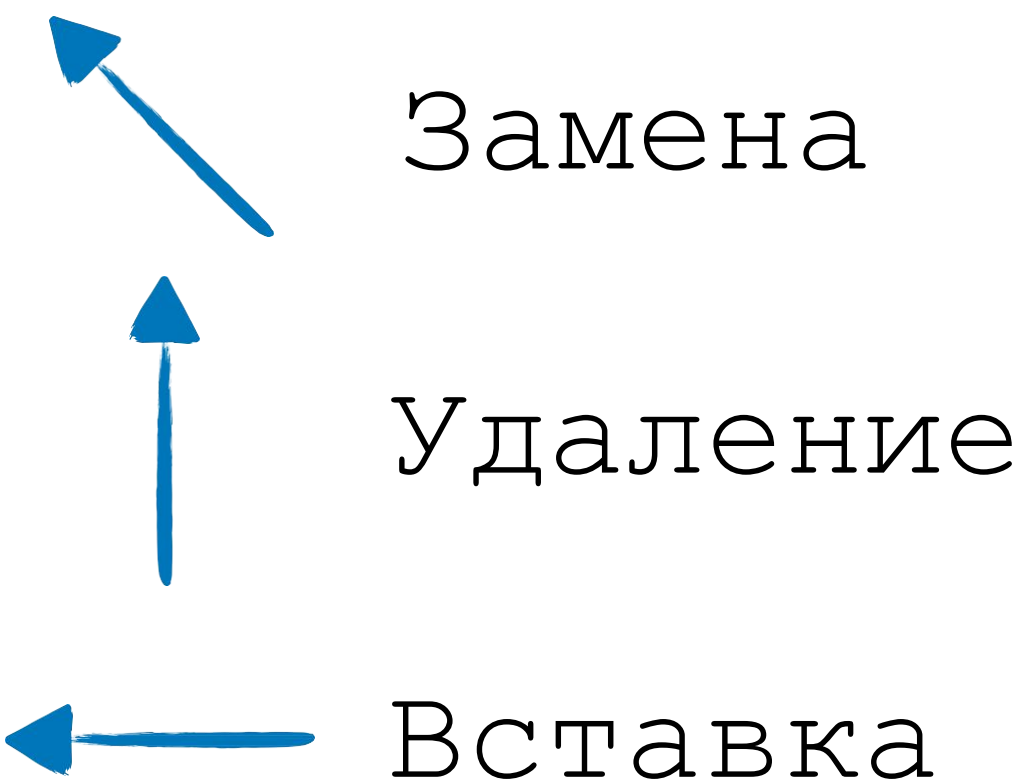
$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



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

# Алгоритм Smith-Waterman (локальное выравнивание)

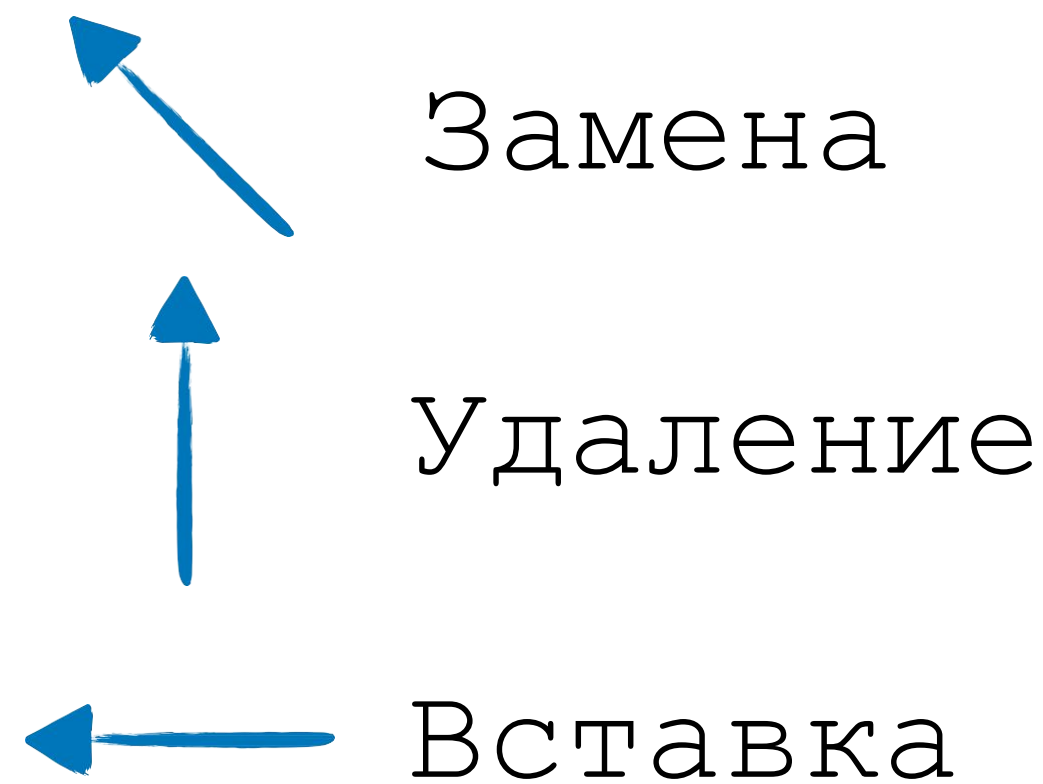
$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



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

# Алгоритм Smith-Waterman (локальное выравнивание)

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3	1	0	1	1
T	0	0	1	4	5	3	1	0
A	0	0	2	2	3	7	5	3
C	0	0	0	1	1	5	9	7

$$D_{0,0} = 0$$

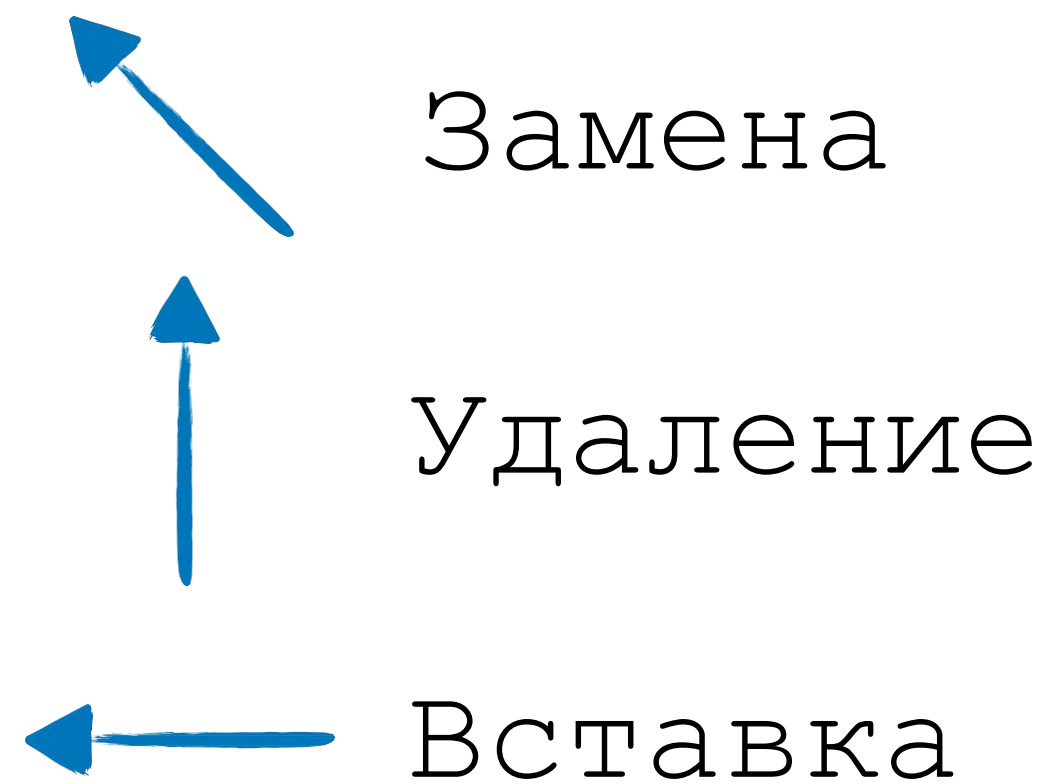
$$D_{i,0} = 0$$

$$D_{0,j} = 0$$

$$D_{ij} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \\ D_{i,j-1} + w(-, b_j) \\ 0 \end{cases}$$

# Алгоритм Smith-Waterman (локальное выравнивание)

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3	1	0	1	1
T	0	0	1	4	5	3	1	0
A	0	0	2	2	3	7	5	3
C	0	0	0	1	1	5	9	7

$$D_{0,0} = 0$$

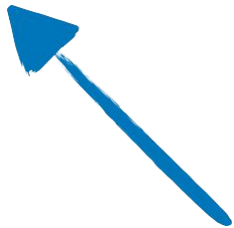


$$D_{i,0} = 0$$

$$D_{0,j} = 0$$

$$D_{ij} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \\ D_{i,j-1} + w(-, b_j) \\ 0 \end{cases}$$

# Алгоритм Smith-Waterman (локальное выравнивание)

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$

 Замена  
 Удаление  
 Вставка

		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3	1	0	1	1
T	0	0	1	4	5	3	1	0
A	0	0	2	2	3	7	5	3
C	0	0	0	1	1	5	9	7

$$D_{0,0} = 0$$

$$D_{i,0} = 0$$

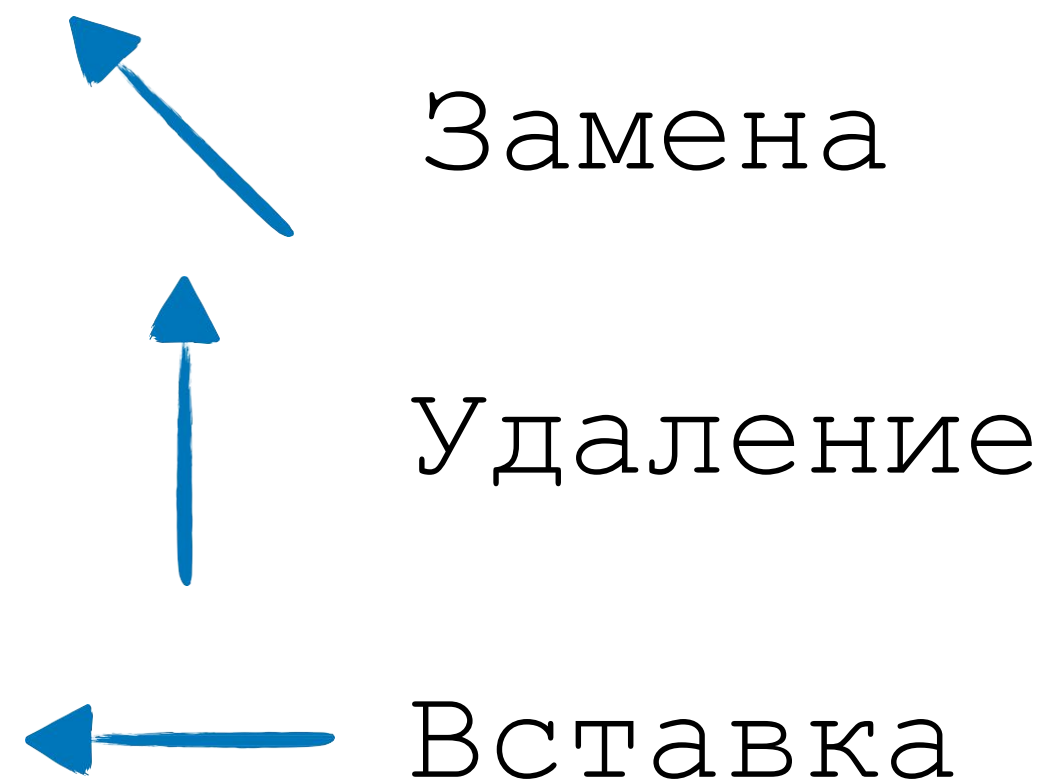
$$D_{0,j} = 0$$

$$D_{ij} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \\ D_{i,j-1} + w(-, b_j) \\ 0 \end{cases}$$



# Алгоритм Smith-Waterman (локальное выравнивание)

$w(a, b) = 2$ , если  $a = b$   
 $w(a, b) = -1$ , если  $a \neq b$   
 $w(a, -) = -2$   
 $w(-, b) = -2$



		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3	1	0	1	1
T	0	0	1	4	5	3	1	0
A	0	0	2	2	3	7	5	3
C	0	0	0	1	1	5	9	7

$$D_{0,0} = 0$$

$$D_{i,0} = 0$$

$$D_{0,j} = 0$$

$$D_{i,j} = \max \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \\ D_{i,j-1} + w(-, b_j) \\ 0 \end{cases}$$

GAGTA  
GATTA

# Замены не равноценны!

Для нуклеотидных последовательностей:

- Silent - не приводят к изменению аминокислотной последовательности
- Missense - приводят к изменению аминокислотной последовательности
- Nonsense - добавляют терминальный кодон, останавливая синтез белка

# Missense замены не равноценны!

Свойства аминокислот:

- Заряд
- Размер
- Гидрофобность
- Химическая реактивность

Valine (гидрофобная) -> Serine  
(гидрофильная) - маловероятно

Tryptophan (ароматическая) -> Glycine  
(неароматическая) - маловероятно

Lysine -> Isolysine (похожие свойства) -  
встречается часто

# Замены не равноценны!

Как быть?

Хотелось бы отличать случайные матчи от вероятных

# РАМ матрицы

- Margaret Dayhoff (1978)
- Основаны на выравнивании 71 различных семейств белков
- Мутации, который видны в выравниваниях не отвергнуты эволюцией (то есть являются Point Accepted)
  - Мутации в кодирующей последовательности
  - Мутация широко распространена
- 1 РАМ - одна мутация на 100 аминокислот (или 1% мутировавших аминокислот)
- Основана на глобальном выравнивании

# РАМ матрицы

... . GDS **F** **H** **Y** **F** V S **H** G... . .  
... . GDS **F** **H** **Y** **Y** V S **F** G... . .  
... . GDS **Y** **H** **Y** **F** V S **F** G... . .  
... . GDS **F** **H** **Y** **F** V S **F** G... . .  
... . GDS **F** **H** **F** **F** V S **F** G... . .

# РАМ матрицы

A	Ala																				
R	Arg	30																			
N	Asn	109	17																		
D	Asp	154	0	532																	
C	Cys	33	10	0	0																
Q	Gln	93	120	50	76	0															
E	Glu	266	0	94	831	0	422														
G	Gly	579	10	156	162	10	30	112													
H	His	21	102	226	43	10	243	23	10												
I	Ile	66	20	36	13	17	8	35	0	3											
L	Leu	95	17	37	0	0	75	15	17	40	253										
K	Lys	57	477	322	85	0	147	104	60	23	43	39									
M	Met	29	17	0	0	0	20	7	7	0	57	207	90								
F	Phe	20	7	0	0	0	0	17	20	90	167	0	17								
P	Pro	345	67	27	10	10	93	40	49	50	7	43	43	4	7						
S	Ser	772	127	432	98	117	47	86	450	26	20	32	168	20	40	269					
T	Thr	590	20	169	57	10	37	31	50	14	129	52	200	28	10	73	696				
W	Trp	0	27	3	0	0	0	0	0	3	0	13	0	0	10	0	17	0			
Y	Tyr	20	1	36	0	30	0	10	0	40	13	23	10	0	260	0	22	23	6		
V	Val	365	20	13	17	33	27	37	97	30	661	303	17	77	10	150	43	186	0	17	
		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	
		Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val



# РАМ матрицы

		ORIGINAL AMINO ACID																			
		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
		Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
A	Ala	9867	2	9	10	3	8	17	21	2	6	4	2	6	2	22	35	32	0	2	18
R	Arg	1	9913	1	0	1	10	0	0	10	3	1	19	4	1	4	6	1	8	0	1
N	Asn	4	1	9822	36	0	4	6	6	21	3	1	13	0	1	2	20	9	1	4	1
D	Asp	6	0	42	9859	0	6	53	6	4	1	0	3	0	0	1	5	3	0	0	1
C	Cys	1	1	0	0	9973	0	0	0	1	1	0	0	0	0	1	5	1	0	3	2
Q	Gln	3	9	4	5	0	9876	27	1	23	1	3	6	4	0	6	2	2	0	0	1
E	Glu	10	0	7	56	0	35	9865	4	2	3	1	4	1	0	3	4	2	0	1	2
G	Gly	21	1	12	11	1	3	7	9935	1	0	1	2	1	1	3	21	3	0	0	5
H	His	1	2	18	3	1	20	1	0	9912	0	1	1	0	2	3	1	1	1	4	1
I	Ile	2	2	3	1	2	1	2	0	0	9872	9	2	12	7	0	1	7	0	1	33
L	Leu	3	1	3	0	0	6	1	1	4	22	9947	2	45	13	3	1	3	4	2	15
K	Lys	2	37	25	6	0	12	7	2	2	4	1	9926	20	0	3	8	11	0	1	1
M	Met	1	1	0	0	0	2	0	0	0	5	8	4	9874	1	0	1	2	0	0	4
F	Phe	1	1	1	0	0	0	0	1	2	8	6	0	4	9946	0	2	1	3	28	0
P	Pro	13	5	2	1	1	8	3	2	5	1	2	2	1	1	9926	12	4	0	0	2
S	Ser	28	11	34	7	11	4	6	16	2	2	1	7	4	3	17	9840	38	5	2	2
T	Thr	22	2	13	4	1	3	2	2	1	11	2	8	6	1	5	32	9871	0	2	9
W	Trp	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	9976	1	0
Y	Tyr	1	0	3	0	3	0	1	0	4	1	1	0	0	21	0	1	1	2	9945	1
V	Val	13	2	1	1	3	2	2	3	3	57	11	1	17	1	3	2	10	0	2	9901

**Mutational probability matrix** for evolutionary distance of **1 PAM**  
(for simplification elements are multiplied by 10000)



# РАМ матрицы

		ORIGINAL AMINO ACID																			
		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
		Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
REPLACEMENT AMINO ACID	A Ala	13	6	9	9	5	8	9	12	5	8	6	7	7	4	11	11	11	2	4	9
	R Arg	3	17	4	3	2	5	3	2	6	3	2	9	4	1	4	4	3	7	2	2
	N Asn	4	4	6	7	2	5	6	4	6	3	2	5	3	2	4	5	4	2	3	3
	D Asp	5	4	8	11	1	7	10	5	6	3	2	5	3	1	4	5	5	1	2	3
	C Cys	2	1	1	1	52	1	1	2	2	2	1	1	1	1	2	3	2	1	4	2
	Q Gln	3	5	5	6	1	10	7	3	7	2	3	5	3	1	4	3	3	1	2	3
	E Glu	5	4	7	11	1	9	12	5	6	3	2	5	3	1	4	5	5	1	2	3
	G Gly	12	5	10	10	4	7	9	27	5	5	4	6	5	3	8	11	9	2	3	7
	H His	2	5	5	4	2	7	4	2	15	2	2	3	2	2	3	3	2	2	3	2
	I Ile	3	2	2	2	2	2	2	2	2	10	6	2	6	5	2	3	4	1	3	9
	L Leu	6	4	4	3	2	6	4	3	5	15	34	4	20	13	5	4	6	6	7	13
	K Lys	6	10	10	8	2	10	8	5	8	5	4	24	9	2	6	8	8	4	3	5
	M Met	1	1	1	1	0	1	1	1	1	2	3	2	6	2	1	1	1	1	1	2
	F Phe	2	1	2	1	1	1	1	1	3	5	6	1	4	32	1	2	2	4	20	3
	P Pro	7	5	5	4	3	5	4	5	5	3	3	4	3	2	20	6	5	1	2	4
	S Ser	9	6	8	7	7	6	7	9	6	5	4	7	5	3	9	10	9	4	4	5
	T Thr	8	5	6	6	4	5	5	6	4	6	4	6	5	3	6	8	11	2	3	6
	W Trp	0	2	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	55	1	0
	Y Tyr	1	1	2	1	3	1	1	1	3	2	2	1	2	15	1	2	2	3	31	2
	V Val	7	4	4	4	4	4	4	5	4	15	10	4	10	5	5	5	7	2	4	17

**Mutational probability matrix** for evolutionary distance of **250 PAM**  
(for simplification elements are multiplied by 100)

# РАМ матрицы

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

If val

>0 →

=0 →

<0 →

If values in matrix,

$>0 \rightarrow$  likely mutation

$=0 \rightarrow$  neutral or random

$<0 \rightarrow$  unlikely mutation

Figure: **The PAM250 log odd matrix.** It is the most used PAM matrix and represents the mutation probabilities of sequences with 20% of equivalence

# BLOSUM

## База выравниваний BLOCS. Белки, разбитые на блоки

[Henikoff, Steven, and Jorja G. Henikoff. "Amino acid substitution matrices from protein blocks." (1992)]

Семейства BLOSUM матриц (BLOSUM30, BLOSUM62, BLOSUM80)

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

# PAM и BLOSUM

- BLOSUM для близких последовательностей, PAM для отдаленных последовательностей.
- Выбор конкретной матрицы зависит от уровня сходства последовательностей.
- BLOSUM часто являются предпочтительными для локального выравнивания, PAM для глобального выравнивания.



# Штрафы за гэпы!



G \_\_\_\_ATTACA  
GAAATT\_CT



G \_A \_TTACA  
GAAATT\_CT

- В примерах выше цена выравнивания одинаковая.
- Но первое “биологически адекватнее”! Два маленьких гэпа происходят менее вероятно чем один, но длинны 2.
- Что делать?

# Штрафы за гэпы!



G \_\_\_\_ ATTACA  
GAAATT\_CT



G \_ A \_ TTACA  
GAAATT\_CT

- В примерах выше цена выравнивания одинаковая.
- Но первое “биологически адекватнее”! Два маленьких гэпа происходят менее вероятно чем один, но длины 2.
- Что делать? Использовать аффинный штраф за гэп!

# Штрафы за гэпы!

Чтобы посчитать  $D_{w,g}(a, b)$ , где  $|a| = n$ ,  $|b| = m$  построим матрицу  $D$ ,  $\text{Dim}(D) = (n + 1, m + 1)$  так:

$$D_{0,0} = 0$$

$$D_{i,0} = g(i)$$

$$D_{0,j} = g(j)$$

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \text{ (замена)} \\ \min_{k=1}^i D_{i-k,j} + g(k) \text{ (удаление k символов)} \\ \min_{k=1}^j D_{i,j-k} + g(k) \text{ (вставка k символов)} \end{cases}$$

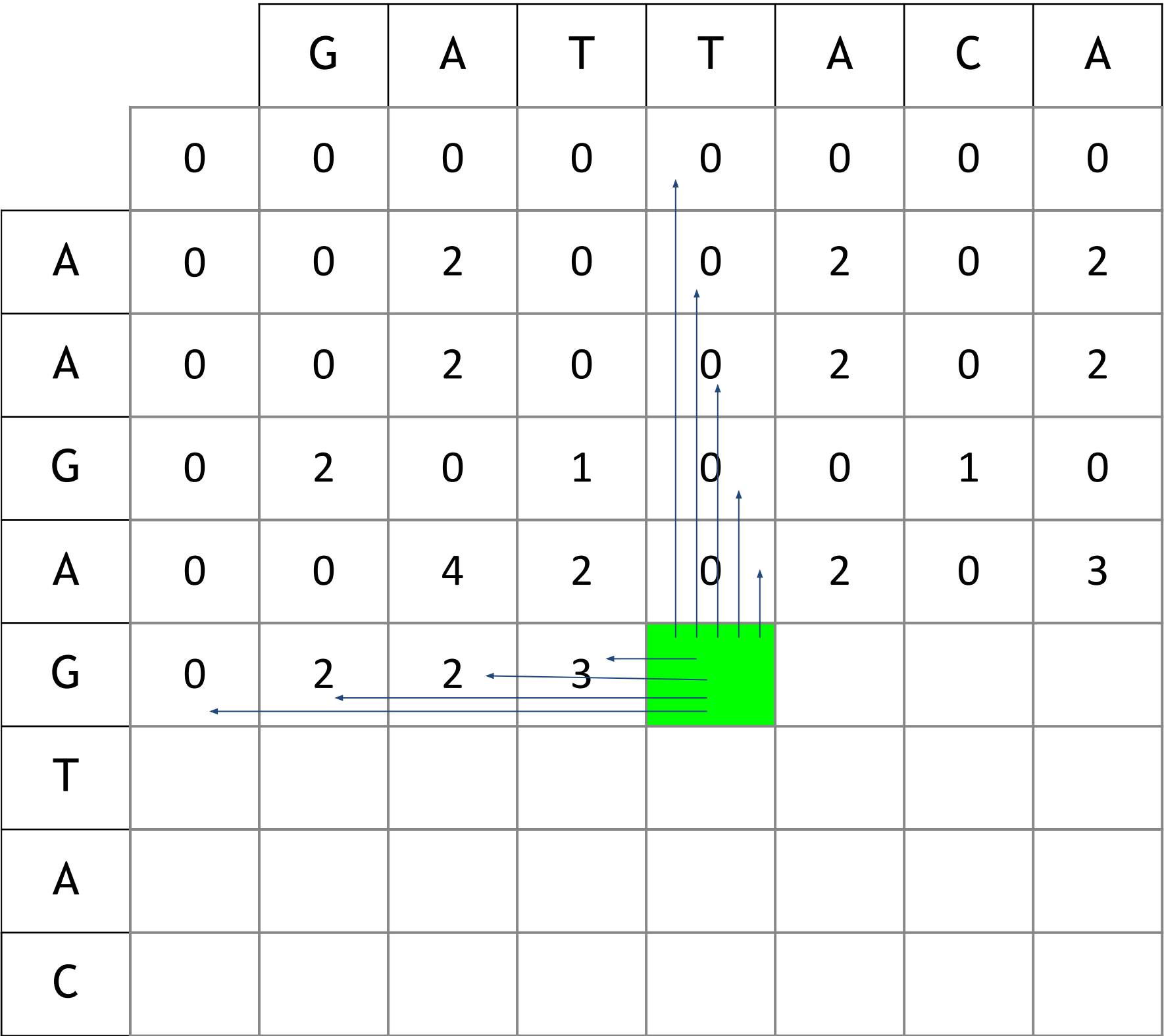
# Штрафы за гэпы!

		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3				
T								
A								
C								



# Штрафы за гэпы!

		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3				
T								
A								
C								



The diagram illustrates the alignment of two DNA sequences: "GATTAACA" (top) and "GATTAACA" (bottom). The alignment is shown in a grid where each cell contains a score. The scores are calculated based on matches (2), mismatches (1), and gaps (0). The alignment is as follows:

		G	A	T	T	A	C	A
	0	0	0	0	0	0	0	0
A	0	0	2	0	0	2	0	2
A	0	0	2	0	0	2	0	2
G	0	2	0	1	0	0	1	0
A	0	0	4	2	0	2	0	3
G	0	2	2	3				
T								
A								
C								

Arrows indicate the alignment path: vertical arrows from the top row to the bottom row, and horizontal arrows from the left to the right. A green box highlights the alignment of "GATTAACA" with "GATTAACA".

# Штрафы за гэпы

- Что хорошего в предыдущем алгоритме?  
Можно использовать вообще для любых функций штрафа за гэпы
- Что плохого?

# Штрафы за гэпы

- Что хорошего в предыдущем алгоритме?  
Можно использовать вообще для любых функций штрафа за гэпы
- Что плохого? Сложность  $O(n^3)$   
:(

# Штрафы за гэпы

- Что хорошего в предыдущем алгоритме?  
Можно использовать вообще для любых функций штрафа за гэпы
- Что плохого?

# Аффинные штрафы за гэпы.

- Используем аффинную функцию

$g(k) = \alpha + \beta k$ , штраф за начало гэпа  $\alpha$ , а за его продолжение

- $\beta$

Существует алгоритм со сложностью  $O(n^2)$


# Интуиция за алгоритмом

## 122. Best Time to Buy and Sell Stock II

Solved 

Medium

 Topics

 Companies

You are given an integer array `prices` where `prices[i]` is the price of a given stock on the  $i^{\text{th}}$  day.

On each day, you may decide to buy and/or sell the stock. You can only hold **at most one** share of the stock at any time. However, you can buy it then immediately sell it on the **same day**.

Find and return *the **maximum** profit you can achieve*.


# Интуиция за алгоритмом

## 122. Best Time to Buy and Sell Stock II

Solved 

Medium

 Topics

 Companies

You are given an integer array `prices` where `prices[i]` is the price of a given stock on the  $i^{\text{th}}$  day.

On each day, you may decide to buy and/or sell the stock. You can only hold **at most one** share of the stock at any time. However, you can buy it then immediately sell it on the **same day**.

Find and return *the **maximum** profit you can achieve*.

# Интуиция за алгоритмом $O(n^2)$

```
1 class Solution {
2 public:
3     int maxProfit(vector<int>& prices) {
4         vector<int> max_profit(prices.size());
5
6         for(int i = 1; i < max_profit.size(); ++i) {
7             for (int j = 0; j < i; ++j) {
8                 max_profit[i] = max(max_profit[i], max_profit[j] + prices[i] - prices[j]);
9             }
10            max_profit[i] = max(max_profit[i], max_profit[i-1]);
11        }
12        return max_profit.back();
13    }
14 };
```



# Интуиция за алгоритмом $O(n)$

# Интуиция за алгоритмом $O(n)$

```
1 class Solution {
2 public:
3     int maxProfit(vector<int>& prices) {
4         vector<int> max_profit_hold(prices.size() + 1, -1000000);
5         vector<int> max_profit_sell(prices.size() + 1);
6
7         for(int i = 1; i < max_profit_sell.size(); ++i) {
8             max_profit_sell[i] = max(max_profit_sell[i - 1], max_profit_hold[i - 1] + prices[i-1]);
9             max_profit_hold[i] = max(max_profit_hold[i - 1], max_profit_sell[i - 1] - prices[i-1]);
10        }
11        return max_profit_sell.back();
12    }
13 };
```

# Интуиция за алгоритмом: лучшее решение

```
class Solution {  
public:  
    int maxProfit(vector<int>& prices) {  
        int ans = 0;  
        for (int i = 1; i < prices.size(); ++i) {  
            if (prices[i-1] < prices[i]) {  
                ans += prices[i] - prices[i-1];  
            }  
        }  
        return ans;  
    }  
};
```

# **Алгоритм выравнивания с афинными гэпами**

# Алгоритм выравнивания с афинными гэпами

- $$D(i, j) = \max \begin{cases} D(i-1, j-1) + \text{substitution score}(i, j) \\ I_x(i, j) \\ I_y(i, j) \end{cases}$$
- $$I_x(i, j) = \max \begin{cases} D(i-1, j) + g \\ I_x(i-1, j) + h \end{cases}$$
- $$I_y(i, j) = \max \begin{cases} D(i, j-1) + g \\ I_y(i, j-1) + h \end{cases}$$

$I_x$  - матрица для выравниваний, заканчивающихся на гэп в  $a$

$I_y$  - матрица для выравниваний, заканчивающихся на гэп в  $b$

# Алгоритм выравнивания с афинными гэпами

Сложность  $O(n^2)$  по времени и памяти

Для глобального выравнивания с использованием  
РАМ250 используется  $g = -10$ , и  $h = -0.1$

# Общие соображения

Пример 1: Геномные последовательности

**Цель:** сравнить геномные последовательности для выявления эволюционных изменений.

**Матрица:** используйте матрицу PAM (например, PAM250).

**Алгоритм:** применить алгоритм Нидлмана-Вунша для глобального выравнивания.

**Результат:** обнаружить консервативные регионы и эволюционные изменения.

# Общие соображения

Пример 2: Функциональная аннотация белков

**Цель:** аннотировать функции белка путем выравнивания последовательностей.

**Матрица:** используйте матрицу BLOSUM (например, BLOSUM62) для оценки замен.

**Алгоритм:** выберите алгоритм Смита-Уотермана для локального выравнивания.

**Результат:** идентифицированы функциональные домены и консервативные мотивы, имеющие важное значение для функции белка.



# На следующих лекциях

Можно ли быстрее чем  $O(n^2)$ ?

Можно ли по памяти лучше чем  $O(n^2)$ ?

# Резюмируем

- Выравнивания последовательностей дают наглядное представление об эволюции.
- Важно то, как именно вычислять стоимость замен.
- Выравнивание с аффинными гэпами вычислять не более трудно, чем с обычными.