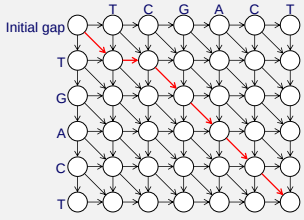
1

BTRY 4840/6840, CS 4775
Computational Genetics and Genomics



TCGACT
T-GACT

September 11, 2018

2

Announcements

- Problem set 1 due tonight at midnight
  - Grace period until 2am
- Problem set 2 out today, due in two weeks
  - Dynamic programming
  - Approximating distributions
  - Sequence alignment
- Make sure you are:
  - Signed up for Piazza
  - On CMS (for submitting problem sets)
- Collaboration:
  - Allowed, but don't study each other's code
  - Can discuss concepts: otherwise you may not learn as much

3

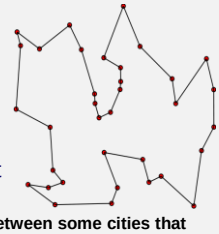
Today's lecture

- Sequence alignment
  - Global alignment via Needleman-Wunsch algorithm
  - Local alignment via Smith-Waterman algorithm
  - Affine gaps
  - Optimizations

4

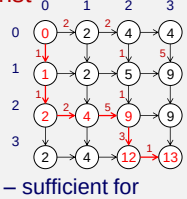
Recall: traveling salesman

- Problem: given a set of places (cities) determine the cycle that visits all places with minimum distance using some metric
- Why fundamentally different than Manhattan Tourist?
  - **May be best to choose path between some cities that is less optimal locally but leads to global optimum**
- Can't solve sub-problem of few cities to build path
  - Breaking into sub-problems essential to dynamic programming



5

Solving sub-problem works for Manhattan Tourist



- What value is stored in each cell in this problem?
  - Maximum weight to arrive at that position
- Why is solving the sub-problem – the optimal path to each node – sufficient for finding the global optimum in this problem?
  - Choices made about the path to a given node have no effect on optimal path thereafter, so best to choose maximum
  - When reasoning about later positions, only need the weight of earlier nodes: later positions independent of earlier path
- Affine gaps (later) somewhat more involved

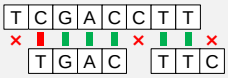
6

Global sequence alignment: Needleman-Wunsch algorithm

7

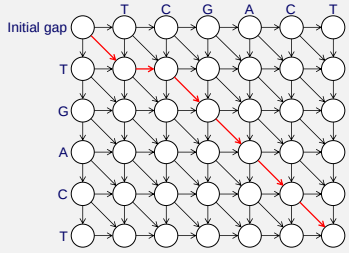
Sequence alignment problem

- Goal: infer mutations, insertion/deletions, and identical positions between two sequences
  - Can decide whether the alignment is biologically sensible using score of alignment
  - Mutations are also called substitutions



8

Can encode alignment using graph

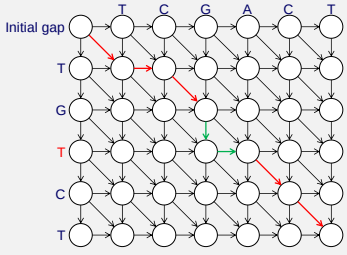



```
TCGACT
T-GACT
```

Diagonal ⇒ bases on row and column aligned
Horizontal ⇒ gap in left side sequence
Vertical ⇒ gap in top sequence

9

Can encode alignment using graph, example 2




```
TCG-ACT
T-GT-CT
```

Diagonal ⇒ bases on row and column aligned
Horizontal ⇒ gap in left side sequence
Vertical ⇒ gap in top sequence

10

Defining scoring metric

- Need way to score potential alignments
  - Want to define score rigorously
  - Convenient for it to be additive
- Given two sequences $\boldsymbol{x}$, $\boldsymbol{y}$, both length n, assume each base (nucleotide) evolves independently
  - Independence not true for all sequences, but OK most of the time

11

Scoring: probabilistic foundations

- Assuming $\boldsymbol{x}$ and $\boldsymbol{y}$ are homologous – i.e., descend from shared ancestral sequence – can consider

$$P(x, y|\text{homologous}) = \prod_{i=1}^{n} P(x_i, y_i) = \prod_{i=1}^{n} \phi_{x_i, y_i}$$

- If $\boldsymbol{x}$ and $\boldsymbol{y}$ are non-homologous, bases are independent, so we have

$$P(x, y|\text{non-homologous}) = \prod_{i=1}^{n} P(x_i) P(y_i) = \prod_{i=1}^{n} \pi_{x_i} \pi_{y_i}$$

- Can estimate above using empirical training data
- Compute likelihood ratio (odds) of homology as

$$\frac{P(x, y|\text{homologous})}{P(x, y|\text{non-homologous})} = \prod_{i=1}^{n} \frac{\phi_{x_i, y_i}}{\pi_{x_i} \pi_{y_i}}$$

12

Scoring: probabilistic foundations, continued

- Move to log space

$$\log\left(\frac{P(x, y|\text{homologous})}{P(x, y|\text{non-homologous})}\right) = \sum_{i=1}^{n} \log \frac{\phi_{x_i, y_i}}{\pi_{x_i} \pi_{y_i}}$$

- Let $s_{a,b} = \log \frac{\phi_{a,b}}{\pi_a \pi_b}$, $\quad S(x, y) = \sum_{i=1}^{n} s_{x_i, y_i}$
- $S(x, y)$: relative likelihood of $\boldsymbol{x}$, $\boldsymbol{y}$ homologous vs not
- The $s_{a,b}$ scores:
  - Are additive across bases
  - Form a score matrix (substitution matrix) of size $|\mathcal{A}|^2$, where $\mathcal{A}$ is the alphabet (4 nucleotides / 20 amino acids)
- More on generating score matrices soon

13

Gap penalties needed

- Gaps: insertions and deletions – indels
  - In practice don't know whether insertion or deletion occurred since we don't have ancestral sequence
- We allow gaps in alignment with a penalty
- If gap penalty is low, will end up with many gaps

```
AC-TT-C-A-GG
-CCT-ACG-C--
```

- If gap penalty is high, many substitutions

```
ACTTCAGG
CCTACGC-
```

- Alignment is about deciding where to place gaps

14

Gap penalty functions

- Let g be the length of a gap
- Have two common gap penalty functions

1. Linear: fixed cost for each gapped base
   - Each base in gap is equally penalized by $-d$

$$\gamma(g) = -gd$$

2. Affine: more elaborate
   - First base in gap penalized by $-d$
   - Subsequent bases penalized by (lesser) extension penalty $-e$

$$\gamma(g) = -d - (g-1)e$$

- More complex / empirical models exist, but these are computationally expensive; less common

15

Edge weights for Needleman-Wunsch

- What is diagonal edge weight from $(i\text{-}1, j\text{-}1)$ to (i, j)? Want log-odds of matching
  - Answer: s_{x_i, y_j}
- Horizontal edge weight from $(i, j\text{-}1)$ to (i, j)?
  - Answer: $-d$
- Vertical edge weight from $(i\text{-}1, j)$ to (i, j)?
  - Answer: $-d$
- How should we define score of cell (i, j)?
  - Recursively: maximum over optimal score to neighboring cells
  - ➢ Defines and solves sub-problem: what is the sub-problem?
    - Optimal path to given cell

16

Needleman-Wunsch: recurrence

- Define $F(i, j)$ as the optimal score at position (i, j), calculated as:

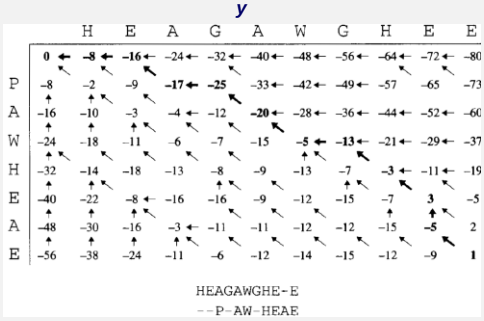
$$F(i,j) = \max \begin{cases} F(i-1, j-1) + s_{x_i, y_j} \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

- Base cases:

$$F(0,0) = 0$$
$$F(i,0) = F(i-1,0) - d$$
$$F(0,j) = F(0,j-1) - d$$

17

Example amino acid alignment



Durbin *et al.* (2006)

18

Pseudo code including trace back pointers

alignNW(x,y,d)
Input: sequences $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_m)$, linear gap penalty d
Output: Best alignment: argmax$_{\boldsymbol{a}}$ $S(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{a})$

$F(0,0) = 0$, $T(0,0) = \text{null};$
for $i = 1$ to n
  $F(i,0) = F(i-1,0) - d$, $T(i,0) = u;$
for $j = 1$ to m
  $F(0,j) = F(0,j-1) - d$, $T(0,j) = l;$
for $i = 1$ to n
  for $j = 1$ to m
    $v_g = F(i-1, j-1) + s_{x_i, y_j};$
    $v_u = F(i-1, j) - d;$
    $v_l = F(i, j-1) - d;$
    $F(i,j) = \max_{z \in \{g,u,l\}} v_z;$
    $T(i,j) = \arg\max_{z \in \{g,u,l\}} v_z;$
$\boldsymbol{a}$ = traceback(x,y,T)

19

How does traceback function work?

- Where do we begin? (i, j) = ?
  $(i, j) = (n, m)$
- Next step?
  - Inspect $T(i, j)$
  - If $T(i, j) = g$:
    - Alignment contains $x[i]$, $y[j]$;
    - Update $i = i - 1$, $j = j - 1$;
  - If $T(i, j) = l$:
    - Alignment contains *gap* $-$, $y[j]$;
    - Update $j = j - 1$
  - If $T(i, j) = u$:
    - Alignment contains $x[i]$, *gap* $-$
    - Update $i = i - 1$

Loop until $i = 0$, $j = 0$

20

Local sequence alignment: Smith-Waterman algorithm

21

Local alignments

- Local alignment is substring in two sequences that is very similar, shows evidence of homology
- Why do we need local alignments?
  - Two sequences may be highly diverged but contain a conserved region we wish to identify
  - May want to look for short sequence in large chromosome

22

Smith-Waterman algorithm

- Expected score of a random match is negative, since it is non-homologous
- Add new option to recurrence – start of local match:

$$F(i,j) = \max \begin{cases} 0 \\ F(i-1, j-1) + s_{x_i, y_j} \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

$$F(i,j) = 0, \quad i = 0 \text{ or } j = 0$$

- Can have start / end positions throughout matrix
  - For best alignment, start traceback from $\max_{i,j} F(i,j)$
  - For all local alignments greater than threshold t, do trace back from all i, j where $F(i,j) > t$

23

Smith-Waterman example

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	**5**	0	0	0	0	0
W	0	0	0	0	2	0	**20** ←	**12** ←	4	0	0
H	0	10 ←	2	0	0	0	12	18	**22** ←	14 ←	6
E	0	2	16 ←	8	0	0	4	10	18	**28**	20
A	0	0	8	21 ←	13	5	0	4	10	20	27
E	0	0	6	13	18	12 ←	4	0	4	16	26

AWGHE
AW-HE

Durbin *et al.* (2006)

24

Summary: global and local sequence alignment

- Global alignment recurrence

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

  - Trace back from $F(n, m)$ – gives alignment of full sequence
- Local alignment recurrence

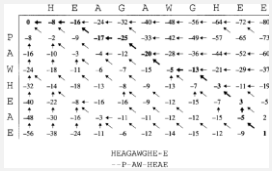
$$F(i,j) = \max \begin{cases} 0 \\ F(i-1, j-1) + s_{x_i, y_j} \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

  - Trace back from either $\max_{i,j} F(i,j)$ or all i, j where $F(i,j) > t$

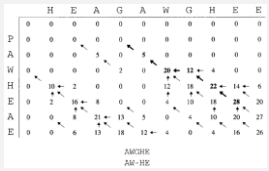
25

Trace back end points

Global alignment



Local alignment



Where should trace back terminate in each case?

- Global: at position $(0,0)$ – aligns full sequence
- Local: at first i, j encountered where $F(i,j) = 0$

Durbin *et al.* (2006)

26

Affine gaps

27

Affine gap penalties

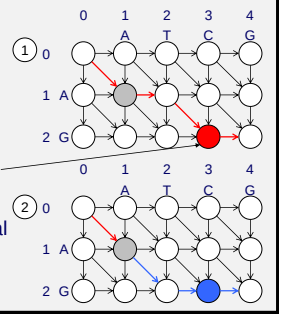
- A bit more difficult than linear $\gamma(g) = -d - (g-1)e$
- Need to know whether a prior site is a gap or not
  - If previous alignment is match, gap opening penalty: $-d$
  - If previous alignment is gap in $\boldsymbol{x}$, gap extension penalty: $-e$
  - If previous alignment is gap in $\boldsymbol{y}$, gap extension penalty: $-e$
- Proposal: inspect trace back pointer of prior site and use e if so, d if not Does not work
  - Only have gap at prior site if gap length ≥1 better than match
  - Need to know best possible path that includes gap at prior site in order to decide on whether to extend or start new gap
- Solution: track best path with gap at given site
  - Instead of F, have M, I_x, I_y – latter two always gapped

28

Example: one matrix insufficient for affine gaps

- Consider two paths
- Score for ① is $F(1,1) - d + s_{G,C} - d$
- Score for ② is $F(1,1) + s_{G,T} - d - e$
- If $s_{G,C} > s_{G,T}$, then $F(2,3) = F(1,1) - d + s_{G,C}$
- But if $s_{G,C} - d < s_{G,T} - e$, this assignment not optimal

➢ Does not "look forward" at $F(2,3)$. Need I_x, I_y



29

Final notes

- Summary – sequence alignment:
  - Several dynamic programming algorithms for optimal global, local alignments
  - Affine gaps possible, more work than linear
  - Optimize runtime via bounded dynamic programming
- Problem set 2 out today
- Readings:
  - Durbin *et al.* chap. 2: sequence alignment
  - Durbin *et al.* chap. 3: hidden Markov models