## Advanced Algorithms and Models for Computational Biology

-- a machine learning approach

Computational Genomics II:
Sequence Modeling \&
Gene Finding with HMM

Eric Xing


Lecture 4, January 30, 2005
Reading: Chap 3, 5 DEKM book
Chap 9, DTW book

## Probabilities on Sequences

- Let $S$ be the space of DNA or protein sequences of a given length $n$. Here are some simple assumptions for assigning probabilities to sequences:
- Equal frequency assumption: All residues are equally probable at any position; i.e., $P\left(X_{i, r}\right)=P\left(X_{i, q}\right)$ for any two residues $r$ and $q$, for all $i$.
- this implies that $P\left(X_{i, r}\right)=\theta_{r}=1 /|A|$, where $A$ is the residue alphabet ( $1 / 20$ for proteins, $1 / 4$ for DNA)
- Independence assumption: whether or not a residue occurs at a position is independent of what residues are present at other positions.
- probability of a sequence

$$
P\left(X_{1}, X_{2}, \ldots, X_{N}\right)=\theta_{r} \cdot \theta_{r}, \ldots, \cdot \theta_{r}=\theta_{r}^{N}
$$

## Failure of Equal Frequency Assumption for (real) DNA

- For most organisms, the nucleotides composition is significantly different from 0.25 for each nucleotide, e.g.,

| - H. influenza | $.31 \mathrm{~A}, .19 \mathrm{C}, .19 \mathrm{G}, .31 \mathrm{~T}$ |
| :--- | :--- |
| - P. aeruginosa | $.17 \mathrm{~A}, .33 \mathrm{C}, .33 \mathrm{G}, .17 \mathrm{~T}$ |
| - M. janaschii | $34 \mathrm{~A}, .16 \mathrm{C}, .16 \mathrm{G}, .34 \mathrm{~T}$ |
| - S. cerevisiae | $.31 \mathrm{~A}, .19 \mathrm{C}, .19 \mathrm{G}, .31 \mathrm{~T}$ |
| - C. elegans | $.32 \mathrm{~A}, .18 \mathrm{C}, .18 \mathrm{G}, .32 \mathrm{~T}$ |
| - H. sapiens | $.30 \mathrm{~A}, .20 \mathrm{C}, .20 \mathrm{G}, .30 \mathrm{~T}$ |

- Note symmetry: $\mathrm{A} \cong \mathrm{T}, \mathrm{C} \cong \mathrm{G}$, even thought we are counting nucleotides on just one strand. Explanation:
- although individual biological features may have non-symmetric composition, usually features are distributed ~ randomly w.r.t. strand, so get symmetry.



## General Hypothesis Regarding Unequal Frequency

- Neutralist hypothesis: mutation bias (e.g., due to nucleotide pool composition)
- Selectionist hypothesis: natural selection bias


## Models for Homogeneous Sequence Entities

- Probabilities models for long "homogeneous" sequence entities, such as:
- exons (ORFs)
- introns
- inter-genetic background
- protein coiled-coil (other other structural) regions
- Assumptions:
- no consensus, no recurring string patterns
- have distinct but uniform residue-composition (i.e., same for all sites)
- every site in the entity are iid samples from the same model
- The model:
- a single multinomial: $X \sim \operatorname{Mul}(1, \theta)$


## The Multinomial Model for Sequence

- For a site $i$, define its residue identity to be a multinomial random vector:

$$
X_{i}=\left[\begin{array}{l}
X_{i, A} \\
X_{i, C} \\
X_{i, G} \\
X_{i, T}
\end{array}\right], \quad \text { where } \quad \begin{aligned}
& X_{i, j}=[0,1], \quad \text { and } \sum_{j \in A, C, G, T]} X_{i, j}=1 \\
& X_{i, j}=1 \text { w.p. } \theta_{j}, \sum_{j \notin A, C, G, T]}=1 .
\end{aligned}
$$

- The probability of an observation $s_{i}=A$ (i.e, $x_{i, A}=1$ ) at site $i$ :

$$
\begin{aligned}
p\left(x_{i}=(\text { say }, A)\right) & =P\left(\left\{X_{i, j}=1, \text { where } j=A \text { index the observed nucleotide }\right\}\right) \\
& =\theta_{A}=\theta_{A}^{x_{A}} \times \theta_{C}^{x_{C}} \times \theta_{G}^{x_{G}} \times \theta_{T}^{x_{T}}=\prod_{k} \theta_{k}^{x_{k}}=\theta^{x}
\end{aligned}
$$

- The probability of a sequence $\left(x_{1}, x_{2}, \ldots, x_{N}\right)$ :

$$
\begin{aligned}
P\left(x_{1}, x_{2}, \ldots, x_{N}\right) & =\prod_{i=1}^{N} P\left(x_{i}\right)=\prod_{i=1}^{N} \prod_{k} \theta_{k}^{x_{i k}} \\
& =\prod_{k} \theta_{k=1}^{N} \sum_{i, k}^{x_{i, k}}=\prod_{k} \theta_{k}^{n_{k}}
\end{aligned}
$$

## Parameter Estimation

- Maximum likelihood estimation: $\theta=\arg \max _{\theta} P(D \mid \theta)$
- multinomial parameters:

$$
\begin{aligned}
& \left\{\theta_{1}, \theta_{2} \ldots\right\}=\arg \max _{\theta} \prod_{k} \theta_{k}^{n_{k}} \text {, s.t. } \sum_{k} \theta_{k}=1 \\
& \text { It can be shown that: } \theta_{k}^{\mathrm{NL}}=n_{k} / N
\end{aligned}
$$

- Bayesian estimation:

$$
\Gamma\left(\sum \alpha_{k}\right)
$$

- Dirichlet distribution: $\quad P(\theta)=\frac{k}{\prod_{k} \Gamma\left(\alpha_{k}\right)} \prod_{k} \theta_{k}^{\alpha_{k}-1}=C(\alpha) \prod_{k} \theta_{k}^{\alpha_{k}-1}$
- Posterior distribution of $\theta$ under the Dirichlet prior:

$$
P\left(\theta \mid x_{1}, \ldots, x_{N}\right) \propto \prod_{k} \theta_{k}^{\alpha_{k}} \prod_{k} \theta_{k}^{n_{k}}=\prod_{k} \theta_{k}^{\alpha_{k}-1+n_{k}}
$$

- Posterior mean estimation:

$$
\theta_{k}=\int \theta_{k} P(\theta \mid D) d \theta=\int \theta_{k} \prod_{k} \theta_{k}^{\alpha_{k}-1+n_{k}} d \theta=\frac{n_{k}+\alpha_{k}}{N+|\alpha|}
$$

## Models for Homogeneous Sequence Entities, ctd

- Limitations
- non-uniform residue composition (e.g., CG rich regions)
- non-coding structural regions (MAR, centromere, telomere)
- di- or tri- nucleotide couplings
- estimation bias
- evolutionary constrains


## Site Models

- Probabilities models for short sequences, such as:
- splice sites
- translation start sites
- promoter elements
- protein "motifs"
- Assumptions:
- different examples of sites can be aligned without indels (insertions/deletions) such that tend to have similar residues in same positions
- drop equal frequency assumption; instead have position-specific frequencies
- retain independence assumption (for now)


## Site Models ctd.

- Applies to short segments (<30 residues) where precise residue spacing is structurally or functionally important, and certain positions are highly conserved
- DNA/RNA sequence binding sites for a single protein or RNA molecule
- Protein internal regions structurally constrained due to folding requirements; or surface regions functionally constrained because bind certain ligands
- Example: C. elegans splice sites



## Nucleotide Counts for 8192 C. elegans 3' Splice Sites



A $0.4000 .4290 .2820 .0580 .008 \quad 0.0920 .0291 .000 \quad 0.0000 .410 \quad 0.2930 .307$ C $0.1180 .0790 .0810 .0290 .016 \quad 0.1350 .8340 .0000 .0000 .1560 .1870 .225$ G $0.0720 .0700 .0630 .0180 .0050 .0730 .0010 .0001 .0000 .310 \quad 0.1590 .191$ T $0.409 \quad 0.4220 .5740 .896 \quad 0.971 \quad 0.700 \quad 0.135 \quad 0.000 \quad 0.000 \quad 0.124 \quad 0.361 \quad 0.276$

## 3' Splice Site - C. elegans



## 5' Splice Sites - C. elegans



## Limitation of Homogeneous Site Models

- Failure to allow indels means variably spaced subelements are "smeared", e.g.:
- branch site, for 3' splice sites;
- coding sequences, for both $3^{\prime}$ and 5 ' sites
- Independence assumption
- usually OK for protein sequences (after correcting for evolutionary relatedness)
- often fails for nucleotide sequences; examples:
- 5' sites (Burge-Karlin observation);
- background (dinucleotide correlation, e.g., GC repeats).


## Why Correlation?

- Splicing involves pairing of a small RNA with the transcription at the 5' splice site.
- The RNA is complementary to the 5' srRNA consensus sequence.
- A mismatch at position -1 tends to destabilize the pairing, and makes it more important for other positions to be correctly paired.
- Analogy can be easily drew for other DNA and protein motifs.


## Comparing Alternative Probability Models

- We will want to consider more than one model at a time, in the following situations:
- To differentiate between two or more hypothesis about a sequence
- To generate increasingly refined probability models that are progressively more accurate
- First situation arises in testing biological assertion, e.g., "is this a coding sequence?" Would compare two models:

1. one associated with a hypothesis $H_{\text {coding }}$ which attaches to a sequence the probability of observing it under experiment of drawing a random coding sequence from the genome
2. one associate with a hypothesis $H_{\text {noncoding }}$ which attaches to a sequence the probability of observing it under experiment of drawing a random non-coding sequence from the genome.

## Likelihood Ratio Test

- The posterior probability of a model given data is:

$$
P(M \mid D)=P(D \mid M) P(M) / P(D)
$$

- Given that all models are equally probable a priori, the posterior probability ratio of two models given the same data reduce to a likelihood ratio:

$$
L R\left(M_{a}, M_{0} \mid D\right)=\frac{P\left(D \mid M_{a}\right)}{P\left(D \mid M_{0}\right)}
$$

- the numerator and the denominator may both be very small!
- The log likelihood ratio (LLR) is the logarithm of the likelihood ratio:

$$
\operatorname{LLR}\left(M_{a}, M_{0} \mid D\right)=\log P\left(D \mid M_{a}\right)-\log P\left(D \mid M_{0}\right)
$$



## Gene Finding

- Given un-annotated sequences,
- delineate:
- transcription initiation site,
- exon-intron boundaries,
- transcription termination site,
- a variety of other motifs: promoters, polyA sites, branching sites, etc.
- The hidden Markov model (HMM)


Hidden Markov Models

- 00
0.0
- 0
- 0
000

The underlying source: genomic entities,
dice,
The sequence:
Ploy NT,
sequence of rolls,

## Example: The Dishonest Casino

A casino has two dice:

- Fair die

$$
P(1)=P(2)=P(3)=P(5)=P(6)=1 / 6
$$

- Loaded die
$P(1)=P(2)=P(3)=P(5)=1 / 10$
$P(6)=1 / 2$
Casino player switches back-\&-forth between fair and loaded die once every 20 turns


## Game:

1. You bet \$1
2. You roll (always with a fair die)
3. Casino player rolls (maybe with fair die, maybe with loaded die)
4. Highest number wins \$2


## Puzzles Regarding the Dishonest Casino

GIVEN: A sequence of rolls by the casino player
1245526462146146136136661664661636616366163616515615115146123562344

## QUESTION

- How likely is this sequence, given our model of how the casino works?
- This is the EVALUATION problem in HMMs
- What portion of the sequence was generated with the fair die, and what portion with the loaded die?
- This is the DECODING question in HMMs
- How "loaded" is the loaded die? How "fair" is the fair die? How often does the casino player change from fair to loaded, and back?
- This is the LEARNING question in HMMs


## A Stochastic Generative Model

- Observed sequence:


A

B


- Hidden sequence (a parse or segmentation):



## Definition (of HMM)



Graphical model

- Transition probabilities between any two states

$$
\begin{aligned}
& p\left(y_{t}^{j}=1 \mid y_{t-1}^{i}=1\right)=a_{i, j}, \\
\text { or } \quad & p\left(y_{+} \mid y_{t-1}^{i}=1\right) \sim \operatorname{Multinomial}\left(a_{i, 1}, a_{i, 2}, \ldots, a_{i, M}\right), \forall i \in I .
\end{aligned}
$$

- Start probabilities

$$
p\left(y_{1}\right) \sim \operatorname{Multinomial}\left(\pi_{1}, \pi_{2}, \ldots, \pi_{M}\right) .
$$

- Emission probabilities associated with each state

$$
p\left(x_{+} \mid y_{+}^{i}=1\right) \sim \operatorname{Multinomial}\left(b_{i, 1}, b_{i, 2}, \ldots, b_{i, k}\right), \forall i \in \mathrm{I}
$$

or in general:

$p\left(x_{+} \mid y_{+}^{i}=1\right) \sim \mathrm{f}\left(\cdot \mid \theta_{i}\right), \forall i \in \mathbb{I}$.
State automata

## Probability of a Parse

- Given a sequence $\mathbf{x}=x_{1} \ldots \ldots x_{T}$ and a parse $y=y_{1}, \ldots \ldots, y_{\top}$,
- To find how likely is the parse:
(given our HMM and the sequence)


$$
\begin{aligned}
& p(\mathbf{x}, \mathbf{y})=p\left(x_{1} \ldots \ldots x_{\mathrm{T}}, y_{1}, \ldots \ldots, y_{\mathrm{T}}\right) \quad \text { (Joint probability) } \\
& =p\left(y_{1}\right) p\left(x_{1} \mid y_{1}\right) p\left(y_{2} \mid y_{1}\right) p\left(x_{2} \mid y_{2}\right) \ldots p\left(y_{\mathrm{T}} \mid y_{\mathrm{T}-1}\right) p\left(x_{\mathrm{T}} \mid y_{\mathrm{T}}\right) \\
& =p\left(y_{1}\right) \mathrm{P}\left(y_{2} \mid y_{1}\right) \ldots p\left(y_{\mathrm{T}} \mid y_{T-1}\right) \times p\left(x_{1} \mid y_{1}\right) p\left(x_{2} \mid y_{2}\right) \ldots p\left(x_{\mathrm{T}} \mid y_{T}\right) \\
& =p\left(y_{1}, \ldots \ldots, y_{T}\right) p\left(x_{1} \ldots \ldots x_{T} \mid y_{1}, \ldots \ldots, y_{T}\right) \\
& \text { Let } \pi_{y_{1}}=\prod_{i=1}^{\text {def }}\left[\pi_{i}\right]^{y_{1}^{\prime}}, \quad a_{y_{t}, y_{t+1}} \stackrel{\text { def }}{=} \prod_{i, j=1}^{M}\left[a_{i j}\right]^{y_{1}^{\prime} y_{r_{t+1}^{\prime}}}, \quad \text { and } b_{y_{t}, x_{t}} \stackrel{\text { def }}{=} \prod_{i=1}^{M} \prod_{k=1}^{K}\left[b_{i k}\right]^{y_{i}^{\prime} x_{k}^{k}} \text {, } \\
& =\pi_{y_{1}} a_{y_{1}, y_{2}} \cdots a_{y_{T-1}, y_{T}} b_{y_{1}, x_{1}} \cdots b_{y_{T}, x_{T}}
\end{aligned}
$$

- Marginal probability: $p(\mathbf{x})=\sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y})=\sum_{y_{1}} \sum_{y_{2}} \cdots \sum_{y_{N}} \pi_{y_{1}} \prod_{t=2}^{T} a_{y_{t-1}, y_{t}} \prod_{t=1}^{T} p\left(x_{t} \mid y_{t}\right)$
- Posterior probability: $p(\mathbf{y} \mid \mathbf{x})=p(\mathbf{x}, \mathbf{y}) / p(\mathbf{x})$


## The Dishonest Casino Model



## Example: the Dishonest Casino

- Let the sequence of rolls be:
- $\boldsymbol{x}=1,2,1,5,6,2,1,6,2,4$
- Then, what is the likelihood of

- $\boldsymbol{y}=$ Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair?
(say initial probs $\mathrm{a}_{0 \text { Fair }}=1 / 2, \mathrm{a}_{\text {oLoaded }}=1 / 2$ )
$1 / 2 \times P(1 \mid$ Fair $) P($ Fair | Fair $) P(2 \mid$ Fair $) P($ Fair | Fair $) \ldots P(4 \mid$ Fair $)=$
$1 / 2 \times(1 / 6)^{10} \times(0.95)^{9}=.00000000521158647211=5.21 \times 10^{-9}$


## Example: the Dishonest Casino

- So, the likelihood the die is fair in all this run is just $5.21 \times 10^{-9}$
- OK, but what is the likelihood of


$$
\pi=\text { Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, }
$$ Loaded, Loaded, Loaded?

$1 / 2 \times P(1 \mid$ Loaded $) P($ Loaded | Loaded $) \ldots P(4 \mid$ Loaded $)=$
$1 / 2 \times(1 / 10)^{8} \times(1 / 2)^{2}(0.95)^{9}=.00000000078781176215=0.79 \times 10^{-9}$

- Therefore, it is after all 6.59 times more likely that the die is fair all the way, than that it is loaded all the way


## Example: the Dishonest Casino

- Let the sequence of rolls be:
- $x=1,6,6,5,6,2,6,6,3,6$
- Now, what is the likelihood $\pi=\mathrm{F}, \mathrm{F}, \ldots, \mathrm{F}$ F?

- $1 / 2 \times(1 / 6)^{10} \times(0.95)^{9}=0.5 \times 10^{-9}$, same as before
- What is the likelihood $\boldsymbol{y}=\mathrm{L}, \mathrm{L}, \ldots, \mathrm{L}$ ?
$1 / 2 \times(1 / 10)^{4} \times(1 / 2)^{6}(0.95)^{9}=.00000049238235134735=5 \times 10^{-7}$
- So, it is 100 times more likely the die is loaded


## Three Main Questions on HMMs

1. Evaluation

| GIVEN | an $H M M M$, | and a sequence $\boldsymbol{x}$, |
| :--- | :--- | :--- |
| FIND | $\operatorname{Prob}(\boldsymbol{x} \mid \boldsymbol{M})$ |  |
| ALGO. | Forward |  |

2. Decoding

GIVEN an HMM $\boldsymbol{M}$, and a sequence $\boldsymbol{x}$,
FIND the sequence $y$ of states that maximizes, e.g., $\mathrm{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{M})$, or the most probable subsequence of states
ALGO. Viterbi, Forward-backward
3. Learning

GIVEN an HMM $\boldsymbol{M}$, with unspecified transition/emission probs., and a sequence $\boldsymbol{x}$,
FIND parameters $\theta=\left(\pi_{\mathrm{i}}, a_{\mathrm{ij}}, \eta_{\mathrm{ik}}\right)$ that maximize $\mathrm{P}(\boldsymbol{x} \mid \theta)$
ALGO. Baum-Welch (EM)

## Applications of HMMs

- Some early applications of HMMs
- finance, but we never saw them
- speech recognition
- modelling ion channels
- In the mid-late 1980s HMMs entered genetics and molecular biology, and they are now firmly entrenched.
- Some current applications of HMMs to biology

| - | mapping chromosomes |
| :--- | :--- |
| - | aligning biological sequences |
| - | predicting sequence structure |
| - | inferring evolutionary relationships |
| - | finding genes in DNA sequence |

## Typical structure of a gene



GENSCAN (Burge \& Karlin)


## Some Facts About Human Genes

- Comprise about $3 \%$ of the genome
- Average gene length: ~ 8,000 bp
- Average of 5-6 exons/gene
- Average exon length: ~200 bp
- Average intron length: ~2,000 bp
- ~8\% genes have a single exon
- Some exons can be as small as 1 or 3 bp.
- HUMFMR1S is not atypical: 17 exons $40-60$ bp long, comprising $3 \%$ of a 67,000 bp gene


## The Idea Behind a GHMM GeneFinder

- States represent standard gene features: intergenic region, exon, intron, perhaps more (promotor, 5'UTR, 3'UTR, Poly-A,..).
- Observations embody state-dependent base composition, dependence, and signal features.
- In a GHMM, duration must be included as well.
- Finally, reading frames and both strands must be dealt with.



## The HMM Algorithms

## Questions:

- Evaluation: What is the probability of the observed sequence? Forward
- Decoding: What is the probability that the state of the 3rd position is Bk , given the observed sequence? ForwardBackward
- Decoding: What is the most likely die sequence? Viterbi
- Learning: Under what parameterization are the observed sequences most probable? Baum-Welch (EM)


## The Forward Algorithm

- We want to calculate $P(\mathbf{x})$, the likelihood of $\mathbf{x}$, given the HMM
- Sum over all possible ways of generating $\mathbf{x}$ :

$$
p(\mathbf{x})=\sum_{y} p(\mathbf{x}, \mathbf{y})=\sum_{y_{1}} \sum_{y_{2}} \cdots \sum_{y_{N}} \pi_{y_{1}} \prod_{t=2}^{T} a_{y_{t-1}, y_{+}} \prod_{t=1}^{T} p\left(x_{t} \mid y_{+}\right)
$$

- To avoid summing over an exponential number of paths $\mathbf{y}$, define
$\alpha\left(y_{+}^{k}=1\right)=\alpha_{+}^{k} \stackrel{\text { def }}{=} P\left(x_{1}, \ldots, x_{+}, y_{+}^{k}=1\right) \quad$ (the forward probability)
- The recursion:
$\alpha_{+}^{k}=p\left(x_{+} \mid y_{+}^{k}=1\right) \sum_{i} \alpha_{+-1}^{i} a_{i, k}$
$P(\mathbf{x})=\sum_{k} \alpha_{T}^{k}$


## The Forward Algorithm derivation

- Compute the forward probability:

$$
\begin{aligned}
\alpha_{t}^{k} & =P\left(x_{1}, \ldots, x_{t-1}, x_{t}, y_{t}^{k}=1\right) \\
& =\sum_{y_{t-1}} P\left(x_{1}, \ldots, x_{t-1}, x_{t}, y_{t-1}, y_{t}^{k}=1\right)
\end{aligned}
$$


$=\sum_{y_{t-1}} P\left(x_{1}, \ldots, x_{t-1}, y_{t-1}\right) P\left(y_{t}^{k}=1 \mid y_{t-1}, x_{1}, \ldots, x_{t-1}\right) P\left(x_{t} \mid y_{t}^{k}=1, x_{1}, \ldots, x_{t-1}, y_{t-1}\right)$
$=\sum_{y_{t-1}} P\left(x_{1}, \ldots, x_{t-1}, y_{t-1}\right) P\left(y_{t}^{k}=1 \mid y_{t-1}\right) P\left(x_{t} \mid y_{t}^{k}=1\right)$
$=P\left(x_{+} \mid y_{t}^{k}=1\right) \sum_{i} P\left(x_{1}, \ldots, x_{t-1}, y_{t-1}^{i}=1\right) P\left(y_{t}^{k}=1 \mid y_{t-1}^{\prime}=1\right)$
$=P\left(x_{+} \mid y_{t}^{k}=1\right) \sum_{i} \alpha_{t-1}^{i} a_{i, k}$

## The Forward Algorithm

- We can compute $\alpha_{t}^{k}$ for all $k, t$, using dynamic programming!

Initialization:

$$
\alpha_{1}^{k}=P\left(x_{1}, y_{1}^{k}=1\right)
$$

$$
=P\left(x_{1} \mid y_{1}^{k}=1\right) P\left(y_{1}^{k}=1\right)
$$

$$
\alpha_{1}^{k}=P\left(x_{1} \mid y_{1}^{k}=1\right) \pi_{k}
$$

$$
=P\left(x_{1} \mid y_{1}^{k}=1\right) \pi_{k}
$$

Iteration:

$$
\alpha_{t}^{k}=P\left(x_{t} \mid y_{t}^{k}=1\right) \sum_{i} \alpha_{t-1}^{i} a_{i, k}
$$

Termination:

$$
P(\mathbf{x})=\sum_{k} \alpha_{T}^{k}
$$

## The Backward Algorithm

- We want to compute $P\left(y_{+}^{k}=1 \mid \mathbf{x}\right)$, the posterior probability distribution on the $t^{\text {th }}$ position, given $\mathbf{x}$

- We start by computing

$$
\begin{aligned}
P\left(y_{+}^{k}=1, \mathbf{x}\right) & =P\left(x_{1}, \ldots, x_{t}, y_{+}^{k}=1, x_{t+1}, \ldots, x_{T}\right) \\
& =P\left(x_{1}, \ldots, x_{+}, y_{+}^{k}=1\right) P\left(x_{t+1}, \ldots, x_{T} \mid x_{1}, \ldots, x_{+}, y_{+}^{k}=1\right) \\
& =P\left(x_{1} \ldots x_{+}, y_{+}^{k}=1\right) P\left(x_{t+1} \ldots x_{T} \mid y_{+}^{k}=1\right)
\end{aligned}
$$

Forward, $\alpha_{t}^{k}$
Backward, $\beta_{t}^{k}=P\left(x_{t+1}, \ldots, x_{T} \mid y_{t}^{k}=1\right)$

- The recursion:

$$
\beta_{t}^{k}=\sum_{k, i} p\left(x_{t+1} \mid y_{t+1}^{i}=1\right) \beta_{t+1}^{i}
$$

## The Backward Algorithm derivation

- Define the backward probability:

$$
\begin{aligned}
\beta_{t}^{k} & =P\left(x_{t+1}, \ldots, x_{T} \mid y_{t}^{k}=1\right) \\
& =\sum_{y_{t+1}} P\left(x_{t+1}, \ldots, x_{T}, y_{t+1} \mid y_{t}^{k}=1\right)
\end{aligned}
$$


$=\sum_{i} P\left(y_{t+1}^{\prime}=1 \mid y_{t}^{k}=1\right) p\left(x_{t+1} \mid y_{t+1}^{\prime}=1, y_{t}^{k}=1\right) P\left(x_{t+2}, \ldots, x_{\tau} \mid x_{t+1}, y_{t+1}^{\prime}=1, y_{t}^{k}=1\right)$
$=\sum_{i} P\left(y_{t+1}^{i}=1 \mid y_{t}^{k}=1\right) p\left(x_{t+1} \mid y_{t+1}^{i}=1\right) P\left(x_{t+2}, \ldots, x_{T} \mid y_{t+1}^{i}=1\right)$
$=\sum_{i} a_{k, i} p\left(X_{t+1} \mid y_{t+1}^{i}=1\right) \beta_{t+1}^{i}$

## The Backward Algorithm

- We can compute $\beta_{t}^{k}$ for all $k, t$, using dynamic programming!

Initialization:

$$
\beta_{T}^{k}=1, \forall k
$$

Iteration:

$$
\beta_{t}^{k}=\sum_{i} a_{k, i} P\left(x_{t+1} \mid y_{t+1}^{i}=1\right) \beta_{t+1}^{i}
$$

Termination:

$$
P(\mathbf{x})=\sum_{k} \alpha_{1}^{k} \beta_{1}^{k}
$$

## Posterior decoding

- We can now calculate

$$
P\left(y_{t}^{k}=1 \mid \mathbf{x}\right)=\frac{P\left(y_{+}^{k}=1, \mathbf{x}\right)}{P(\mathbf{x})}=\frac{\alpha_{+}^{k} \beta_{t}^{k}}{P(\mathbf{x})}
$$

- Then, we can ask
- What is the most likely state at position $t$ of sequence $\mathbf{x}$ :

$$
k_{t}^{*}=\arg \max _{k} P\left(y_{t}^{k}=1 \mid \mathbf{x}\right)
$$

- Note that this is an MPA of a single hidden state, what if we want to a MPA of a whole hidden state sequence?
- Posterior Decoding: $\left\{y_{t}^{k_{+}^{*}}=1: t=1 \cdots T\right\}$
- This is different from MPA of a whole sequence of hidden states
- This can be understood as bit error rate

Example:
MPA of $X$ ? MPA of $(X, Y)$ ?

| $x$ | $y$ | $P(x, y)$ |
| ---: | ---: | ---: |
| 0 | 0 | 0.35 |
| 0 | 1 | 0.05 |
| 1 | 0 | 0.3 |
| 1 | 1 | 0.3 |

## Viterbi decoding

- GIVEN $\mathbf{x}=x_{1}, \ldots, x_{T}$, we want to find $\mathbf{y}=y_{1}, \ldots, y_{T}$, such that $P(\mathbf{y} \mid \mathbf{x})$ is maximized:

$$
\mathbf{y}^{\star}=\operatorname{argmax}_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x})=\operatorname{argmax}_{\pi} P(\mathbf{y}, \mathbf{x})
$$

- Let

$$
V_{t}^{k}=\max _{\left\{y_{1}, \ldots y_{t-1}\right\}} P\left(x_{1}, \ldots, x_{t-1}, y_{1}, \ldots, y_{t-1}, x_{t}, y_{t}^{k}=1\right)
$$

$$
=\text { Probability of most likely sequence of states ending at state } y_{\mathrm{t}}=k
$$

- The recursion:
$V_{t}^{k}=p\left(x_{t} \mid y_{t}^{k}=1\right) \max _{i} a_{i, k} V_{t-1}^{i}$
- Underflows are a significant problem
 $p\left(x_{1}, \ldots, x_{t}, y_{1}, \ldots, y_{+}\right)=\pi_{y_{1}} a_{y_{1}, y_{2}} \cdots a_{y_{t-1}, y_{t}} b_{y_{1}, x_{1}} \cdots b_{y_{t}, x_{t}}$
- These numbers become extremely small - underflow
- Solution: Take the logs of all values: $V_{+}^{k}=\log p\left(x_{+} \mid y_{+}^{k}=1\right)+\max _{i}\left(\log \left(a_{i, k}\right)+V_{t-1}^{i}\right)$


## The Viterbi Algorithm - derivation

- Define the viterbi probability:

$$
\begin{aligned}
& V_{t+1}^{k}=\max _{\left\langle y_{1}, \ldots y_{t}\right\}} P\left(x_{1}, \ldots, x_{t}, y_{1}, \ldots, y_{t}, x_{t}, y_{t+1}^{k}=1\right) \\
& =\max _{\left(y_{1}, \ldots+\right\}} P\left(x_{t+1}, y_{t+1}^{\kappa}=1 \mid x_{1}, \ldots, x_{t}, y_{1}, \ldots, y_{t}\right) P\left(x_{1}, \ldots, x_{t}, y_{1}, \ldots, y_{t}\right) \\
& =\max _{\left(y_{1}, \ldots, y_{t}\right.} P\left(x_{t+1}, y_{t+1}^{K}=1 \mid y_{t}\right) P\left(x_{1}, \ldots, x_{t-1}, y_{1}, \ldots, y_{t-1}, x_{t}, y_{t}\right) \\
& =\max _{i} P\left(X_{t+1}, y_{t+1}^{k}=1 \mid y_{t}^{\prime}=1\right) \max _{\left(y_{1}, \ldots y_{t-1} \mid\right.} P\left(x_{1}, \ldots, x_{t-1}, y_{1}, \ldots, y_{t-1}, x_{t}, y_{t}^{\prime}=1\right) \\
& =\max _{i} P\left(X_{t+1}, \mid y_{t+1}^{k}=1\right) a_{i, k} V_{t}^{i} \\
& =P\left(X_{t+1}, y_{t+1}^{k}=1\right) \max _{i} a_{i, k} V_{t}^{i}
\end{aligned}
$$

## The Viterbi Algorithm

- Input: $\mathbf{x}=x_{1}, \ldots, x_{T}$,


## Initialization:

$$
V_{1}^{k}=P\left(x_{1} \mid y_{1}^{k}=1\right) \pi_{k}
$$

Iteration:

$$
\begin{aligned}
& V_{t}^{k}=P\left(x_{t} \mid y_{+}^{k}=1\right) \max _{i} a_{i, k} V_{t-1}^{i} \\
& \operatorname{Ptr}(k, t)=\arg \max
\end{aligned} a_{i, k} V_{t-1}^{i} .
$$

Termination:

$$
P\left(\mathbf{x}, \mathbf{y}^{*}\right)=\max _{k} V_{T}^{k}
$$

TraceBack:

$$
\begin{aligned}
& y_{T}^{*}=\arg \max _{k} V_{T}^{k} \\
& y_{t-1}^{*}=\operatorname{Ptr}\left(y_{t}^{*}, t\right)
\end{aligned}
$$

## Computational Complexity and implementation details

- What is the running time, and space required, for Forward, and Backward?

$$
\begin{aligned}
& \alpha_{t}^{k}=p\left(x_{+} \mid y_{+}^{k}=1\right) \sum_{i} \alpha_{t-1}^{i} a_{i, k} \\
& \beta_{t}^{k}=\sum_{i} a_{k, i} p\left(x_{t+1} \mid y_{t+1}^{i}=1\right) \beta_{t+1}^{i} \\
& V_{t}^{k}=p\left(x_{+} \mid y_{t}^{k}=1\right) \max _{i} a_{i, k} V_{t-1}^{i}
\end{aligned}
$$

Time: $O\left(K^{2} \mathcal{M}\right)$; Space: $O(K N)$.

- Useful implementation technique to avoid underflows
- Viterbi: sum of logs
- Forward/Backward: rescaling at each position by multiplying by a constant


## Learning HMM: two scenarios

- Supervised learning: estimation when the "right answer" is known
- Examples:

GIVEN: a genomic region $x=x_{1} \ldots x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands
GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- Unsupervised learning: estimation when the "right answer" is unknown
- Examples:

GIVEN: the porcupine genome; we don't know how frequent are the CpG islands there, neither do we know their composition
GIVEN: $\begin{aligned} & \text { 10,000 rolls of the casino player, but we don't see when he } \\ & \text { changes dice }\end{aligned}$ changes dice

- QUESTION: Update the parameters $\theta$ of the model to maximize $P(x \mid \theta)$--- Maximal likelihood (ML) estimation


## Supervised ML estimation

- Given $x=x_{1} \ldots x_{N}$ for which the true state path $y=y_{1} \ldots y_{N}$ is known,
- Define:

$$
\begin{array}{ll}
A_{i j} & =\# \text { times state transition } i \rightarrow j \text { occurs in } \mathbf{y} \\
B_{i k} & =\# \text { times state } i \text { in } \mathbf{y} \text { emits } k \text { in } \mathbf{x}
\end{array}
$$

- We can show that the maximum likelihood parameters $\theta$ are:

$$
\begin{aligned}
& a_{i j}^{M L}=\frac{\#(i \rightarrow j)}{\#(i \rightarrow \bullet)}=\frac{\sum_{n} \sum_{t=2}^{T} y_{n, t-1}^{i} y_{n, t}^{j}}{\sum_{n} \sum_{t=2}^{T} y_{n, t-1}^{i}}=\frac{A_{i j}}{\sum_{j} A_{i j}} \\
& b_{i k}^{M L}=\frac{\#(i \rightarrow k)}{\#(i \rightarrow \bullet)}=\frac{\sum_{n} \sum_{t=1}^{T} y_{n, t}^{i} x_{n, t}^{k}}{\sum_{n} \sum_{t=1}^{T} y_{n, t}^{i}}=\frac{B_{i k}}{\sum_{k^{\prime}} B_{i k k^{\prime}}}
\end{aligned}
$$

- What if y is continuous? We can treat $\left\{\left(x_{n, t}, y_{n, t}\right): t=1: T, n=1: N\right\}$ as $N_{\times} \boldsymbol{T}$ observations of, e.g., a Gaussian, and apply learning rules for Gaussian ...


## Supervised ML estimation, ctd.

- Intuition:
- When we know the underlying states, the best estimate of $\theta$ is the average frequency of transitions \& emissions that occur in the training data
- Drawback:
- Given little data, there may be overfitting:
- $P(x \mid \theta)$ is maximized, but $\theta$ is unreasonable

0 probabilities - VERY BAD

- Example:
- Given 10 casino rolls, we observe

$$
x=2,1,5,6,1,2,3,6,2,3
$$

$$
y=F, F, F, F, F, F, F, F, F, F
$$

- Then:
$a_{F F}=1 ; \quad a_{F L}=0$
$\mathrm{b}_{\mathrm{F} 1}=\mathrm{b}_{\mathrm{F} 3}=.2$;
$b_{F 2}=.3 ; b_{F 4}=0 ; b_{F 5}=b_{F 6}=.1$


## Pseudocounts

- Solution for small training sets:
- Add pseudocounts
$A_{i j} \quad=\#$ times state transition $i \rightarrow j$ occurs in $\mathbf{y}+R_{i j}$
$B_{i k} \quad=\#$ times state $i$ in $\mathbf{y}$ emits $k$ in $\mathbf{x}+S_{i k}$
- $R_{i j}, S_{i j}$ are pseudocounts representing our prior belief
- Total pseudocounts: $R_{i}=\Sigma_{j} R_{i j}, S_{i}=\Sigma_{k} S_{i k}$,
- --- "strength" of prior belief,
- --- total number of imaginary instances in the prior
- Larger total pseudocounts $\Rightarrow$ strong prior belief
- Small total pseudocounts: just to avoid 0 probabilities --smoothing


## Unsupervised ML estimation

- Given $x=x_{1} \ldots x_{N}$ for which the true state path $y=y_{1} \ldots y_{N}$ is unknown,
- EXPECTATION MAXIMIZATION

0. Starting with our best guess of a model $M$, parameters $\theta$.
1. Estimate $A_{i j}, B_{i k}$ in the training data

- How? $A_{j}=\sum_{n, t}\left\langle Y_{n, t-1}^{i} y_{n, t}^{j}\right\rangle, B_{i k}=\sum_{n, t}\left\langle Y_{n, t}^{i}\right\rangle \chi_{n, t}^{k}$, How? (homework)

2. Update $\theta$ according to $A_{i j}, B_{i k}$

- Now a "supervised learning" problem

3. Repeat 1 \& 2, until convergence

This is called the Baum-Welch Algorithm
We can get to a provably more (or equally) likely parameter set $\theta$ each iteration

## The Baum Welch algorithm

- The complete log likelihood

$$
\ell_{c}(\boldsymbol{\theta} ; \mathbf{x}, \mathbf{y})=\log p(\mathbf{x}, \mathbf{y})=\log \prod_{n}\left(p\left(y_{n, 1}\right) \prod_{t=2}^{T} p\left(y_{n, t} \mid y_{n, t-1}\right) \prod_{t=1}^{T} p\left(x_{n, t} \mid x_{n, t}\right)\right)
$$

- The expected complete log likelihood

- EM
- The E step

$$
\begin{aligned}
& \gamma_{n, t}^{i}=\left\langle y_{n, t}^{i}\right\rangle=p\left(y_{n, t}^{i}=1 \mid \mathbf{x}_{n}\right) \\
& \xi_{n, t}^{i, j}=\left\langle y_{n, t-1}^{i} y_{n, t}^{j}\right\rangle=p\left(y_{n, t-1}^{i}=1, y_{n, t}^{j}=1 \mid \mathbf{x}_{n}\right)
\end{aligned}
$$

- The M step ("symbolically" identical to MLE)

$$
\pi_{i}^{M L}=\frac{\sum_{n} \gamma_{n, 1}^{i}}{N} \quad a_{i j}^{M L}=\frac{\sum_{n} \sum_{t==}^{T} \xi_{n}^{i, j}}{\sum_{n} \sum_{t=1}^{T-1} \gamma_{n, t}^{i}} \quad b_{i k}^{M L}=\frac{\sum_{n} \sum_{t=1}^{T} \gamma_{n, t}^{i} x_{n, t}^{k}}{\sum_{n} \sum_{t=1}^{T-1} \gamma_{n, t}^{\prime}}
$$

## The Baum-Welch algorithm -comments

Time Complexity:
\# iterations $\times \mathrm{O}\left(\mathrm{K}^{2} \mathrm{~N}\right)$

- Guaranteed to increase the log likelihood of the model
- Not guaranteed to find globally best parameters
- Converges to local optimum, depending on initial conditions
- Too many parameters / too large model: Overt-fitting


## Acknowledgments

- Serafim Batzoglou: for some of the slides adapted or modified from his lecture slides at Stanford University
- Terry Speed: for some of the slides modified from his lectures at UC Berkeley

