

CS395T
Computational Statistics with
Application to Bioinformatics

Prof. William H. Press
Spring Term, 2009
The University of Texas at Austin

Unit 6: Mixture Models

Mixture Models



this one is a 3-component mixture

Suppose we have N independent events, $i=1\dots N$.
Each might be from distribution 0 or distribution 1, but
we don't know which (2-component mixture)

But we do know the respective probabilities for each i ,

$$p_{0i} \equiv P(\mathbf{x}_i|0) \quad p_{1i} \equiv P(\mathbf{x}_i|1)$$

We want a (probabilistic) assignment of each event to 0 or 1.

Suppose $\mathbf{v} = (v_1, v_2, v_3, \dots, v_N)$ is an assignment of each event to a distribution

e.g., $\mathbf{v} = (1, 1, 0, 1, 0, 0, 0, 1, \dots, 1)$

Suppose s is the fraction of events in distribution 1, $s = P(v_i = 1)$

That is everything we need to know to write down a “forward” model for the
probability of the data, given the (known and unknown) quantities:

$$P(\text{data}|\mathbf{v}, s) = \prod_{v_i=1} p_{1i} \times \prod_{v_i=0} p_{0i}$$

Now do the Bayes thing!

$$\begin{aligned}
 P(\mathbf{v}, s | \text{data}) &\propto P(\text{data} | \mathbf{v}, s) P(\mathbf{v}, s) \\
 &= P(\text{data} | \mathbf{v}, s) P(\mathbf{v} | s) P(s) \\
 &= \prod_{v_i=1} p_{1i} \times \prod_{v_i=0} p_{0i} \times s^{\#(v_i=1)} (1-s)^{\#(v_i=0)} P(s) \\
 &= \prod_{v_i=1} p_{1i} s \times \prod_{v_i=0} p_{0i} (1-s) \times P(s)
 \end{aligned}$$

That is the complete model, usually too much to comprehend directly. Instead, we are usually interested in various marginalizations. For example:

$$\begin{aligned}
 P(s | \text{data}) &\propto \sum_{\mathbf{v} \in \mathbf{2}^N} \left[\prod_{v_i=1} p_{1i} s \times \prod_{v_i=0} p_{0i} (1-s) \times P(s) \right] \\
 &= \prod_i \underbrace{[p_{1i} s + p_{0i} (1-s)]}_{\text{prob of } i \text{ in the mixture distribution}} P(s) \leftarrow \text{prior on the mixture}
 \end{aligned}$$

remember to normalize over s if you want a real probability distribution!

Even more interesting is the marginalization that gives the assignment of each data point to one distribution or the other:

$$\begin{aligned}
 P(v_j = 1 | \text{data}) &\propto \int \sum_{v \in 2^{N-1}} p_{1j}^s \prod_{v_i=1, i \neq j} p_{1i}^s \prod_{v_i=0, i \neq j} p_{0i}(1-s) P(s) ds \\
 &= \int p_{1j}^s \frac{P(s | \text{data})}{p_{1j}^s + p_{0j}(1-s)} ds \\
 &= \int \frac{p_{1j}^s}{p_{1j}^s + p_{0j}(1-s)} P(s | \text{data}) ds
 \end{aligned}$$

and similarly

$$P(v_j = 0 | \text{data}) \propto \int \frac{p_{0j}(1-s)}{p_{1j}^s + p_{0j}(1-s)} P(s | \text{data}) ds$$

it's just that data point's relative probabilities in the two distributions, weighted by the mix probability

and then averaged over the mix probabilities

This is a very general idea, which can occur with many useful variations. Let's apply to Towne...



Bayes and Bar Sinister

We can now understand that the Towne family problem is really a mixture model problem: Each VLSTR sample is either from a descendent of William Towne or from the descendent of a “non-paternal event”. We are given an unknown mixture of such samples.

Our model will have 3 unknown parameters:

- r mutation probability per locus per generation
- c non-paternal probability per generation
- L if non-paternal, number of generations back to LCA

Arms of Sir Charles Beauclerk, 1st Duke of St Albans, bastard son of King Charles II by Nell Gwynn



Modeling L as a constant is rather crude, but will illustrate the point. If this really mattered, we'd need to do a better job here.

The model is:

$$\begin{aligned}
 \text{pmi } x &= @(\text{k}, \text{n}, \text{loci}, \text{r}, \text{c}, \text{lca}) (1-\text{c}).^{\text{n}} * \text{bin}(\text{k}, \text{n} * \text{loci}, \text{r}) \dots \\
 &\quad + (1-(1-\text{c}).^{\text{n}}) * \text{bin}(\text{k}, (\text{n} + \text{lca}) * \text{loci}, \text{r}); \\
 \text{model } 2 &= @(\text{r}, \text{c}, \text{lca}) \text{pmi } x(23, 10, 37, \text{r}, \text{c}, \text{lca}) * \text{pmi } x(5, 9, 37, \text{r}, \text{c}, \text{lca}) \dots \\
 &\quad * \text{pmi } x(0, 3, 37, \text{r}, \text{c}, \text{lca}) * \text{pmi } x(0, 3, 37, \text{r}, \text{c}, \text{lca}) \dots \\
 &\quad * \text{pmi } x(1, 5, 37, \text{r}, \text{c}, \text{lca}) * \text{pmi } x(0, 5, 37, \text{r}, \text{c}, \text{lca}) \dots \\
 &\quad * \text{pmi } x(0, 6, 37, \text{r}, \text{c}, \text{lca}) * \text{pmi } x(1, 11, 37, \text{r}, \text{c}, \text{lca}) \dots \\
 &\quad * \text{pmi } x(3, 10, 37, \text{r}, \text{c}, \text{lca}) * \text{pmi } x(4, 10, 12, \text{r}, \text{c}, \text{lca}) ./ \text{r};
 \end{aligned}$$

Notice that we now include all the data, especially clearly non-paternal T2.

We evaluate the model over a 3-dimensional grid of parameters, and then normalize it.

```

rvals = del : del : 0.02;
cvals = [.002 .005 .01 .02 .03 .06 .1 .2];
lvals = [25 50 100 200];
[rgrid cgrid lgrid] = ndgrid(rvals, cvals, lvals);
f2vals = arrayfun(model2, rgrid, cgrid, lgrid);
f2vals = f2vals ./ squeeze(sum(sum(sum(f2vals, 3), 2), 1))

```

priors are implicit in the spacing of the grids, here approximately logarithmic; each grid cell is taken as equiprobable

MATLAB madness!

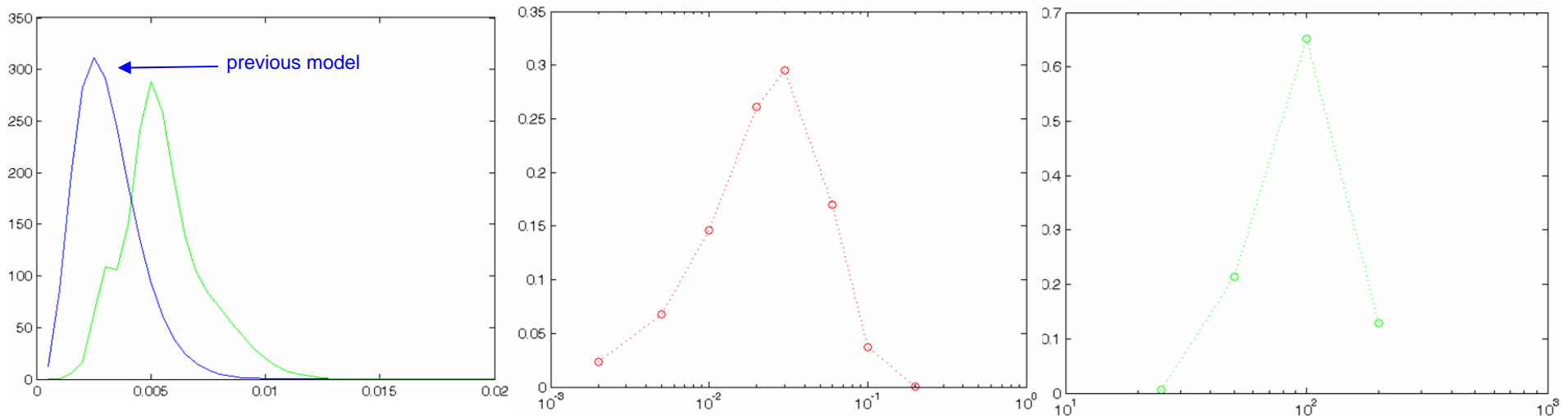
We get individual parameter distributions by marginalization

```

f2r = sum(sum(f2vals, 3), 2);
f2c = sum(sum(f2vals, 3), 1);
f2lca = sum(squeeze(sum(f2vals, 1)), 1);
plot(rvals, f2r./del, '-g');
semilogx(cvals, f2c, ':or');
semilogx(lvals, f2lca, ':og');

```

Hint: use size() to debug this kind of stuff!



Calculate mixture probabilities

```
function p = nonpatprob(k,n,loci,rgrid,cgrid,lcagrid,f2vals)
    pat = squeeze(sum(sum(sum( arrayfun(@ppat,rgrid,cgrid,lcagrid) .* f2vals ,3),2),1));
    nonpat = squeeze(sum(sum(sum( arrayfun(@pnon,rgrid,cgrid,lcagrid) .* f2vals ,3),2),1));
    p = nonpat/(pat+nonpat);
    function p = ppat(r,c,lca)
        p = (1-c).^n * poisspdf(k,n*loci*r);
    end
    function p = pnon(r,c,lca)
        p = (1-(1-c).^n) * poisspdf(k,(n+lca)*loci*r);
    end
end
end
```

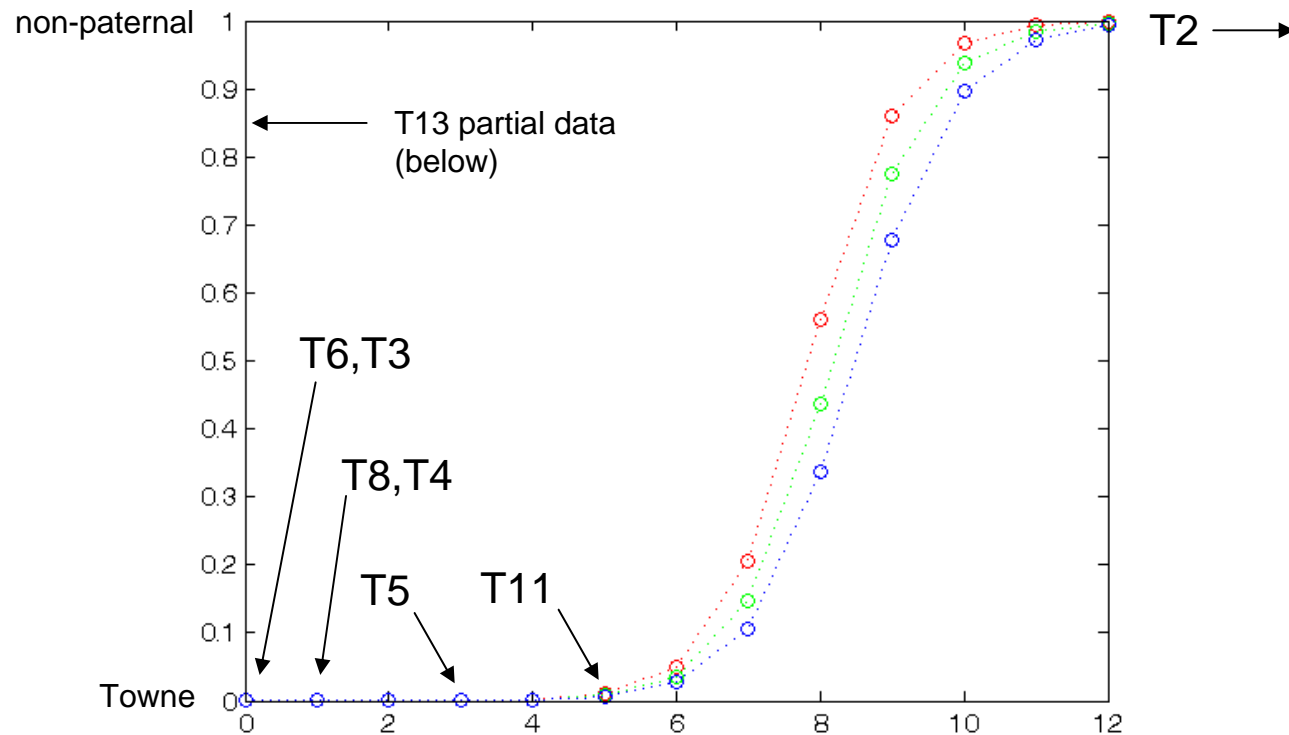
```
for k=0:12, gen9(k+1) = nonpatprob(k,9,37,rgrid,cgrid,lcagrid,f2vals); end
for k=0:12, gen10(k+1) = nonpatprob(k,10,37,rgrid,cgrid,lcagrid,f2vals); end
for k=0:12, gen11(k+1) = nonpatprob(k,11,37,rgrid,cgrid,lcagrid,f2vals); end
plot([0:12],gen9,':or')
plot([0:12],gen10,':og')
plot([0:12],gen11,':ob')
```

OK, I apologize for the loops.



father was a sailor!

And the answers are...



`p13 = nonpatprob(4, 10, 12, rgri d, cgri d, l cagri d, f2val s)`

`p13 =`
`0.8593`

So, by Bayesian statistical modeling, T11 fought his way back to legitimacy. I guess this a happy ending.

(Actually, I'd guess that our LCA model is too crude: no single L is consistent with both T2 and T11 – so our model “promoted” T11 to legitimacy. We could certainly do better. But for pedagogy, I think this is enough of Towne!)