Homework 3

November 23, 2015

Let's look at the haplotypes:

	AB	Ab	bb
AB	(AB,AB)	(AB,Ab)	(Ab,Ab)
Ab	(AB,aB)	(AB,bb);(Ab,aB)	(AB,aB)
bb	(aB,aB)	(aB,bb)	(bb,bb)

Ambiguity in haplotypes occur whenever any of loci '1,2' is heterozygous or both are heterozygous.
$n_{aB,aB}$ in this case gives rise to twp haplotype pairs: $(bb, AB); (aB, Ab)$
We cannot directly determine the exact count from the genotype information.
In otherwords the haplotype counts $n_{(bb/AB)}$ and $n_{(aB/Ab)}$ are the missing data.
Thus, missing data: $n_{AB/bb}$ and $n_{Ab/Ab}$.
We assume there N individuals and hence there are $2N$ haplotypes.
Observed data: $Y = (n_{ABAB}, n_{bbAB}, n_{bbbb}, n_{ABAb}, n_{bbAb}, n_{Abbb}, n_{AbAb})$
Missing Data: $n_{AB/bb}$ and $n_{Ab/aB}$
We construct complete data as the haplotype counts:
Complete Data: $n_{AB}, n_{Ab}, n_{aB}, n_{bb}$
Parameters: $\theta = (p_{AB}, p_{Ab}, p_{aB}, p_{bb})$
and hence the Complete data likelihood is given by:

$$g(n_{AB}, n_{Ab}, n_{aB}, n_{bb}|\theta) = \frac{2N}{n_{AB}!n_{Ab}!n_{aB}!n_{bb}!} p_{AB}^{n_{AB}} p_{Ab}^{n_{Ab}} p_{aB}^{n_{aB}} p_{bb}^{n_{bb}}$$

In the E step. we perform (m^{th} step):

$$\begin{aligned} \hat{n_{AB}} &= E[n_{AB}|Y, \theta_m] \\ \hat{n_{Ab}} &= E[n_{Ab}|Y, \theta_m] \\ \hat{n_{aB}} &= E[n_{aB}|Y, \theta_m] \\ \hat{n_{bb}} &= E[n_{bb}|Y, \theta_m] \end{aligned}$$

where $\theta_m = (p_{AB}^{(m)}, p_{Ab}^{(m)}, p_{aB}^{(m)}, p_{bb}^{(m)})$
Just consider n_{AB} for now.

$$\begin{aligned} n_{AB} &= E[n_{AB}|Y, \theta_m] \\ &= 2n_{ABAB} + n_{ABaB} + n_{AbAB} + E[n_{AB/bb}|Y, \theta_m] \end{aligned}$$

where the last term comes because the AB haplotype can also come from the ambiguos we highlighted in the table above $(bb, AB); (aB, Ab)$
Now, we need to consider:

$$\begin{aligned} E[n_{AB/bb}|Y, \theta_m] &= n_{AbAb} P(AB/bb|Ab/aB, AB/bb) \\ &= n_{AbAb} \times (\frac{2p_{AB}p_{bb}}{2p_{AB}p_{bb} + 2p_{Ab}p_{aB}}) \end{aligned}$$

Where the latter term comes out from the conditional probability of observing Ab/aB haploltype given it is coming from
a heterozygous subpopulation at both A,B

Thus, the E step gives us:

$$n\hat{_{AB}} = 2n_{ABAB} + n_{ABaB} + n_{AbAB} + n_{AbAb}\frac{p_{AB}p_{bb}}{p_{AB}p_{bb} + p_{Ab}p_{aB}}$$
$$n\hat{_{Ab}} = 2n_{ABbb} + n_{ABAb} + n_{AbAB} + n_{AbAb}\frac{p_{Ab}p_{aB}}{p_{AB}p_{bb} + p_{Ab}p_{aB}}$$
$$n\hat{_{aB}} = 2n_{bbAB} + n_{1AB0} + n_{bbAb} + n_{AbAb}\frac{p_{aB}p_{Ab}}{p_{AB}p_{bb} + p_{Ab}p_{aB}}$$
$$n\hat{_{bb}} = 2n_{bbbb} + n_{1Ab0} + n_{bbaB} + n_{AbAb}\frac{p_{bb}p_{AB}}{p_{AB}p_{bb} + p_{Ab}p_{aB}}$$

In [1]:
```
from __future__ import division
from ipy_table import *

locus1_alleles = ['a', 'A']
locus2_alleles = ['b', 'B']

locus1_genotypes = set([('').join(sorted(x+y)) for x in locus1_alleles for y in  locus1_alleles]
locus2_genotypes = set([('').join(sorted(x+y)) for x in locus2_alleles for y in  locus2_alleles]
locus1_genotypes = sorted(locus1_genotypes)
locus2_genotypes = sorted(locus2_genotypes)
```

In [2]:
```
phenotypes = sorted(locus2_genotypes)
phenotypes = [['Locus'] + phenotypes]
for i, l1 in enumerate(locus1_genotypes):
    phenotypes.append([locus1_genotypes[i]])
    for j, l2 in enumerate(locus2_genotypes):
        phenotype = list(set([('').join(sorted(x+y)) for x in l1 for y in l2]))
        if len(phenotype)==1:
            element = '{}/{}'.format(phenotype[0], phenotype[0])
        elif len(phenotype)==2:
            element = '{}/{}'.format(phenotype[0], phenotype[1])
        else:
            element = '{}/{}, {}/{}'.format(phenotype[0], phenotype[3], phenotype[1], phenotype
        phenotypes[i+1].append(element)
```

In [3]:
```
make_table(phenotypes)
apply_theme('basic_both')
set_cell_style(2, 2, color='red')
```

Out[3]: <ipy_table.IpyTable at 0x7f198a798090>

In [4]:
```
data = [[10,15,5], [10,50,13], [3,13,10]]
total = sum([sum(i) for i in zip(*data)])
```

In [5]:
```
print 'Genotype table'
make_table(data)
```

```
Genotype table
```

Out[5]: <ipy_table.IpyTable at 0x7f198a798f90>

In [6]:
```
total
```

```
Out[6]: 129

In [7]: # Initialise x
        x = 10
        # Initialise x
        newx = 20

        counts = {}

        # Initialise genotypes
        for i, l1 in enumerate(locus1_genotypes):
            for j, l2 in enumerate(locus2_genotypes):
                counts[l1+l2] = data[i][j]

        # Initialise Probability
        p = {'AB':1/6, 'Ab': 1/3, 'aB': 1/3, 'ab': 1/6}

In [8]: print 'Initial Probabilities:'
        for key in p.keys():
            print key, p[key]

Initial Probabilities:
AB 0.166666666667
ab 0.166666666667
aB 0.333333333333
Ab 0.333333333333

In [9]: while abs(newx-x)>= 1e-6:
            x = newx
            newx = counts['AaBb'] * (p['AB']*p['ab']/(p['AB']*p['ab']+p['Ab']*p['aB']))
            p['AB'] = (2*counts['AABB']+ counts['AABb'] + x)/(2*total)
            p['Ab'] = (counts['AABb']+2*counts['AAbb']+counts['AaBb']-x)/(2*total)
            p['aB'] = (counts['AaBB']+counts['AaBb']-x+2*counts['aaBB']+counts['Aabb'])/(2*total)
            p['ab'] = (x+counts['aaBb']+counts['Aabb']+2*counts['aabb'])/(2*total)

In [10]: x

Out[10]: 42.762664440703766

In [11]: print 'Final probabilities'
         for key in p.keys():
             print key, p[key]

Final probabilities
AB 0.301405676127
ab 0.344041335041
aB 0.14045478899
Ab 0.124950913021
```

Thus the final value of x is: 42 where x is the count of AB/ab phenotypes.