Homework 2

Michael Li

1. (a) possible combinations of X_1, X_2, X_3 for $Y=6$

$\{1,1,4\}, \{1,2,3\}, \{1,3,2\}, \{2,2,2\}$
$\{1,4,1\}, \{2,1,3\}, \{2,3,1\}$
$\{4,1,1\}, \{3,1,2\}, \{3,2,1\}$

$$P(Y=6) = \sum P(\{X_1, X_2, X_3\} = \{\text{one of the above}\})$$
$$= \left(\tfrac{1}{2}\cdot\tfrac{1}{2}\cdot\tfrac{1}{16}\right)\cdot 3 + \left(\tfrac{1}{2}\cdot\tfrac{1}{4}\cdot\tfrac{1}{8}\right)\cdot 6 + \left(\tfrac{1}{4}\cdot\tfrac{1}{4}\cdot\tfrac{1}{4}\right)$$
$$= \frac{5}{32}$$

(b) $h_{Y_n}(y) = \text{prob}\left(\sum_{i=1}^{n} X_i = y\right)$

$h_{Y_{n-1}}(y') = \text{prob}\left(\sum_{i=1}^{n-1} X_i = y'\right)$ This is basically one step before $h_{Y_n}(y)$

so $h_{Y_n}(y) = \text{prob}\left(\sum_{i=1}^{n-1} X_i = y'\right) \cdot \text{prob}(X_n = y - y')$

$= h_{Y_{n-1}}(y') \cdot f_X(y-y')$

six cases: $(y-y') \in \{1,2,3,4,5,6\}$ sum all cases

so $h_{Y_n}(y) = \sum_{i=1}^{6} h_{Y_{n-1}}(y-i) f_X(i)$

$= \frac{1}{2} h_{Y_{n-1}}(y-1) + \frac{1}{4} h_{Y_{n-1}}(y-2) + \frac{1}{8} h_{Y_{n-1}}(y-3) + \frac{1}{16} h_{Y_{n-1}}(y-4) + \frac{1}{32} h_{Y_{n-1}}(y-5) + \frac{1}{32} h_{Y_{n-1}}(y-6)$

(c) Initializations: $h_{Y_1}(1) = \pi_1, h_{Y_1}(2) = \pi_2, h_{Y_1}(3) = \pi_3, h_{Y_1}(4) = \pi_4, h_{Y_1}(5) = \pi_5, h_{Y_1}(6) = \pi_6$

$h_{Y_1}(\text{other numbers}) = 0$

loop through 2 to n and calculate probabilities for the possible values of y at each level, which will be $y \in \{n, n+1, \dots, 6n\}$

The worst case running time will be $\underset{\text{outer loop}}{(n-1)} \cdot \underset{\text{inner loop}}{5n} = 5n^2 - 5n$

This will just be $O(n^2)$

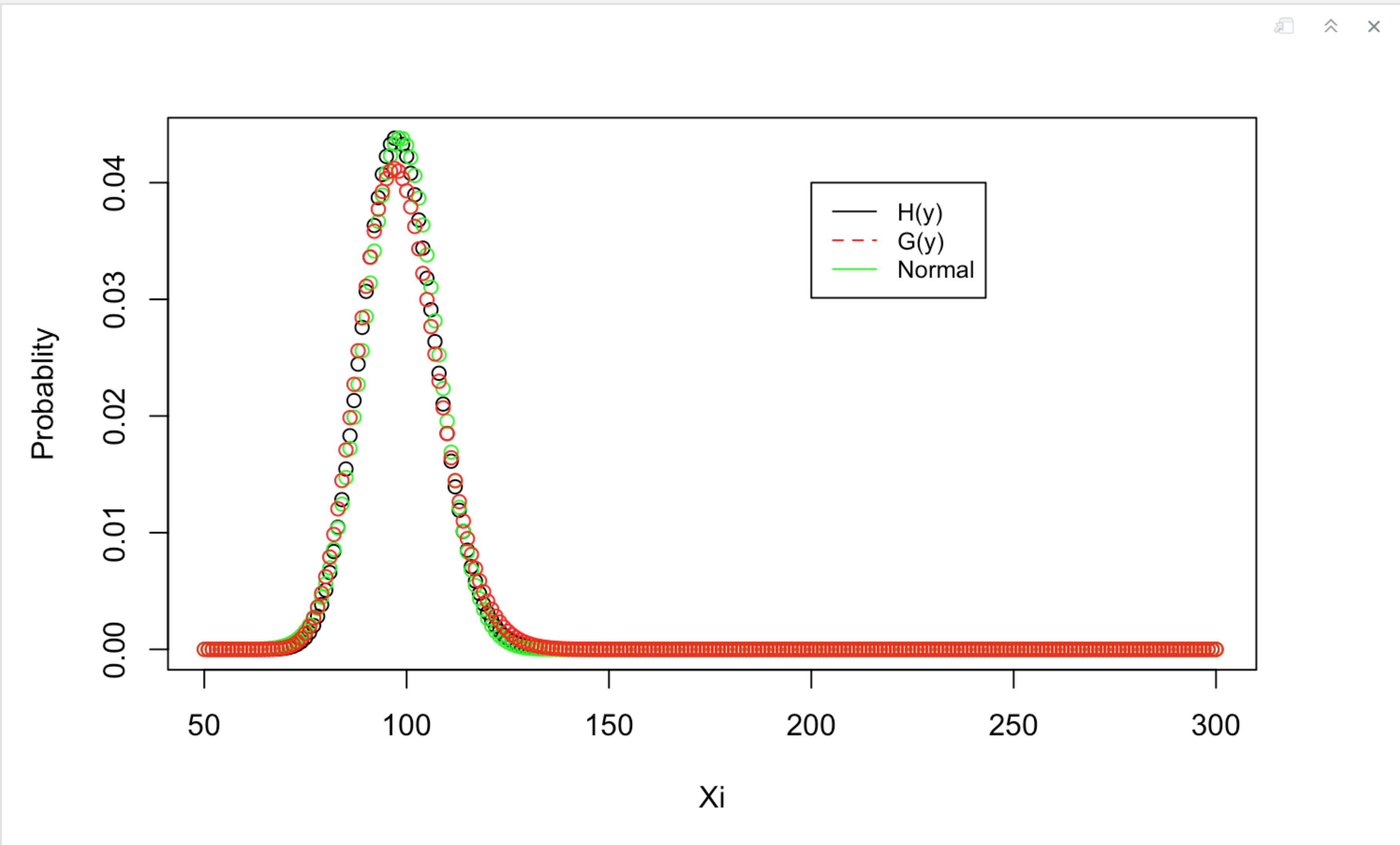
Source code is submitted separately

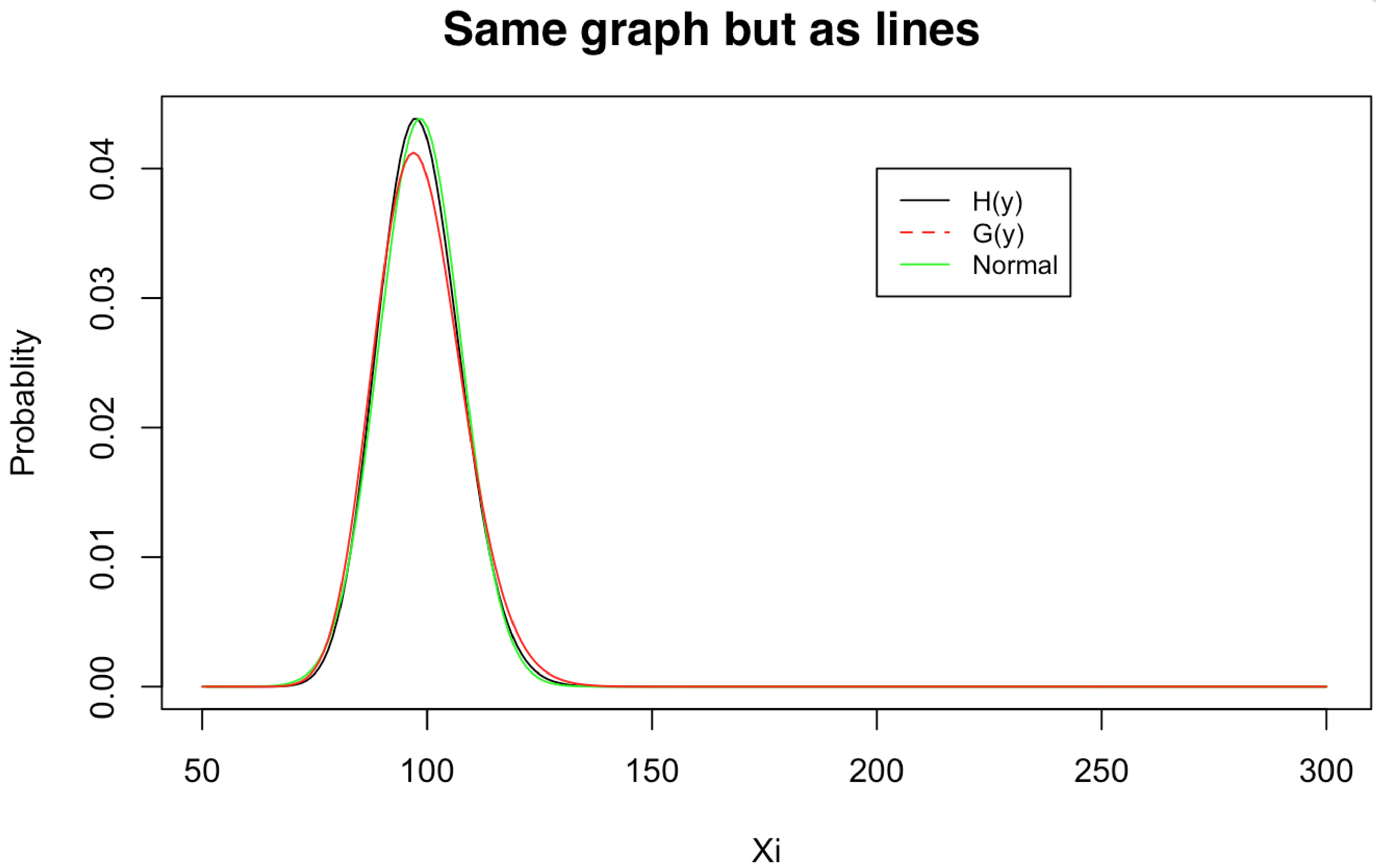
it works with the example input

(d)

Problem 1d graph

~~~
```{r}
library(readxl)
Data <- read_excel("~/Documents/Junior/CS4775/HW2Q1d.xlsx")
plot(Data, xlab="Xi", ylab="Probablity")
range <-seq(50,300,1)
Norm <- dnorm(range,mean=98.4375,sd=sqrt(82.7637))
points(range, Norm, col="green")
Gy <- dnbinom(range-50, 50, 0.5079)
points(range, Gy, col="red")
legend(200,.04, legend=c("H(y)", "G(y)","Normal"),
       col=c("black", "red", "green"), lty=1:2, cex=0.8)
```
~~~




Same graph but as lines
H(y)
G(y)
Normal
Probablity
0.00
0.01
0.02
0.03
0.04
50
100
150
200
250
300
Xi

(d) $E(f_x(x)) = \frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 2 + \frac{1}{8}\cdot 3 + \frac{1}{16}\cdot 4 + \frac{1}{32}\cdot 5 + \frac{1}{32}\cdot 6 = \frac{63}{32}$

$E(f_x^2(x)) = \frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 4 + \frac{1}{8}\cdot 9 + \frac{1}{16}\cdot 16 + \frac{1}{32}\cdot 25 + \frac{1}{32}\cdot 36 = \frac{177}{32} = 5.53125$

so for $g_{Y_n}(y)$, $\frac{1}{\phi} = \frac{63}{32}$, $\phi = \frac{32}{63} = 0.5079$

When applying CLT, $\mu_n = n E(X_i) = 50 \cdot \frac{63}{32} = 98.4375$

$Var(f_x(x)) = E(f_x^2(x)) - E^2(f_x(x)) = \frac{177}{32} - \left(\frac{63}{32}\right)^2 = 1.65527$

$\sigma_n^2 = n\,Var(X_i) = 50 \cdot 1.65527 = \boxed{82.7637}$

$Var(g_{Y_n}(y)) = \frac{(1-\phi)n}{\phi^2} = 95.36$, code variance: $\boxed{82.76}$

The variance from the normal distribution and the one generated by my code were very similar, while the variance of the negative binomial distribution was larger

(e) $y_n = 300$ where $n = 50$

this only happens when $x = 6$ for all preceeding x_i's

$f_i(x) = 6 \ \forall\, i \in \{1, \dots, 50\}$

So $Pr(y_n = 300 \mid n = 50) = \left(\frac{1}{32}\right)^{50}$

$= 5.527e-76$

my program gives the same value

when calculating the normal distribution's p-value in R

we get $4.5736e-109$

The p-value for the negative binomial is $3.4712e-43$

In this case the normal distribution provided a value that was smaller than the actual probability, while the negative binomial provided a value that was larger. Both were off by about the same amount.

The observed inaccuracies might have been due to the difference in variances. The negative binomial had the largest variance so the estimate was larger.

2. (a)

Sequence X

	ε	G	A	C	T	T
	0	-2	-4	-6	-8	-10
G	-2	1	-1	-3	-5	-7
G	-4	-1	0	-2	-4	-6
C	-6	-3	-2	1	-1	-3
A	-8	-5	-2	-1	-2	-4
A	-10	-7	-4	-3	-4	-5
T	-12	-9	-6	-5	-2	-3
C	-14	-11	-8	-5	-4	-3

Sequence Y

best alignment

X: G A C _ _ T T
Y: G G C A A T C

(b) Followed lecture slides and implemented the Needleman-Wunsch algorithm with linear gap penalties

optimal beta-globin sequence alignment using d=100:
attached as screenshot

source code turned in separately

(c) Alignments plus score attached as screenshots.
The alignments are different. This one has a smaller score, but that doesn't mean anything because of the different gap penalties.
I do like this alignment better because it is cleaner and has many fewer random single gaps.

2b alignment

```
[Michaels-MacBook-Pro:CS4775 michaelli$ python HW2Q2b.py -f sequences.fasta -s score_matrix.json]
 -d 100
Alignment:
GGGTGGGAAA-ATAGACCAATAGG-CAGAGAGAGTCAGTGCCTATCAGAAACCCAAGAGTCTTCTCTGTCTCCACA-TGC
AAA-GGGAAACATAGA-CAG-GGGACACTCAAAGTTAGTGCCTGCTGGAAA-GC-AGA--C--CTCTGTCTCCA-AGCAC

CCAGTTTCTATTGGTCTCCT-TAAACCTGTCTTGTAACCTTGATA
CCAACTTCTA-----CT--TGTGAG-CTGCCTTGTAACCTGGATA

Score: 4225.0
```

```
[Michaels-MacBook-Pro:CS4775 michaelli$ python HW2Q2c.py -f sequences.fasta -s score_matrix.json]
 -d 430 -e 30
0
Alignment:
GGGTGGGAAAATAGACCAATAGGCAGAGAGAGTCAGTGCCTATCAGAAACCCAAGAGTCTTCTCTGTCTCCACATGCCCA
AAAGGGAAACATAGA-CAGGGGACACTCAAAGTTAGTGCCTGCTGGAAAGCAGA------CCTCTGTCTCCAAGCACCCA

GTTTCTATTGGTCTCCTTAAACCTGTCTTGTAACCTTGATA
ACTTCTACTTGT-------GAGCTGCCTTGTAACCTGGATA

Score: 3077.0
```

2c alignment

(e) The closer e and d are, the more smaller gaps there are. As e and d grow further apart, the gaps become less frequent but longer. This makes sense because as the opening gap is penalized higher compared to gap extensions the algorithm will want to have fewer but longer gaps to maximize the score.

Also, if you make d and e small compared to the alignment scores, the gaps will become more frequent since they are penalized less.

I think that the right score matrix and gap penalties would probably be some moderate combination.

It would be very interesting to run a machine learning algorithm to determine the best estimated score matrix and penalties.