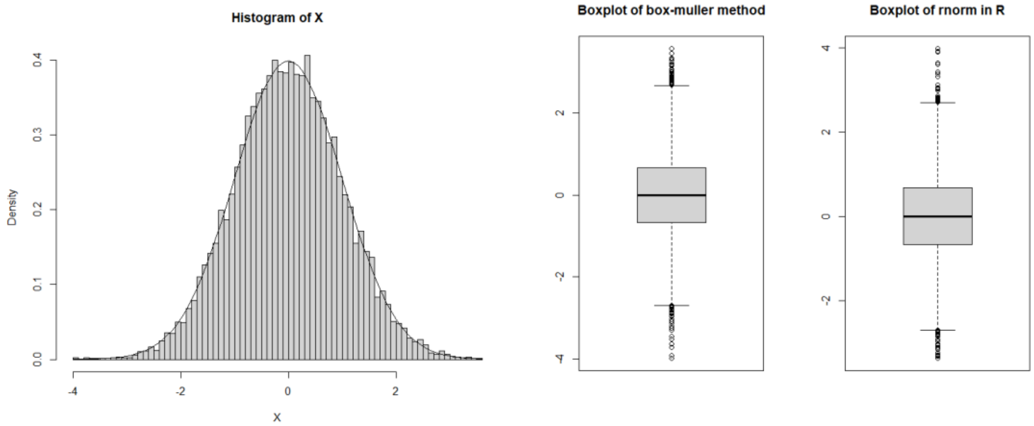
Problem 1 (a)



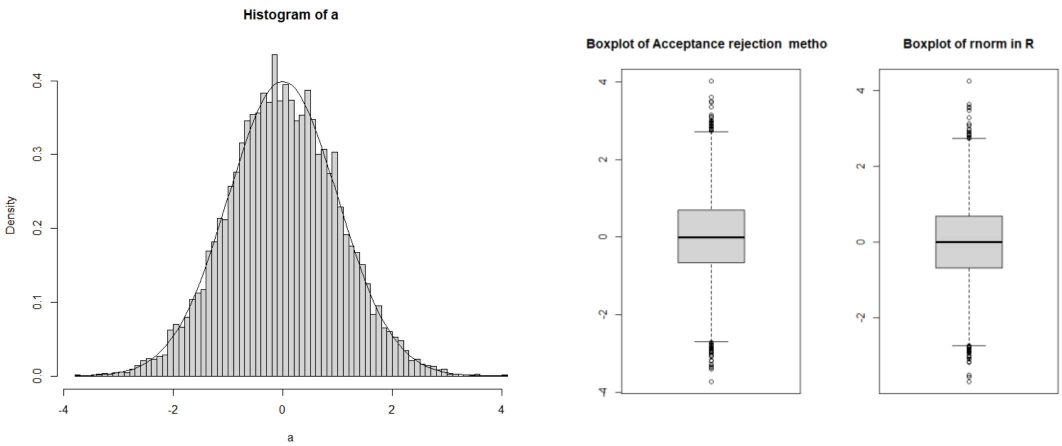
Box-Muller 變換是通過服從均勻分佈的隨機變數，來構建服從標準常態分佈的隨機變數的一種方法。具體的描述為：選取兩個服從[0,1]上均勻分佈的隨機變數，服從$\sqrt{-2\ln U1}\cos 2\pi U2$ 和 $\sqrt{-2\ln U1}\sin 2\pi U2$，則 X 與 Y 服從均值為 0，方差為 1 的標準常態分佈。

從圖上的結果看來，box-muller 方法可以很好的模擬標準常態分佈，然而 box-muller 的運行速度卻慢於 rnorm 的生成速度。

```
> time_box_muller
Time difference of 0.01099396 secs
> time_rnorm
Time difference of 0.003021002 secs
```

從程式結果可以明顯看出時間差異超過 30 倍，在生成效率上差很多。

Problem 1 (b)



由已知分佈的概率密度函數 f(x) ，產生服從此分佈的樣本 X。

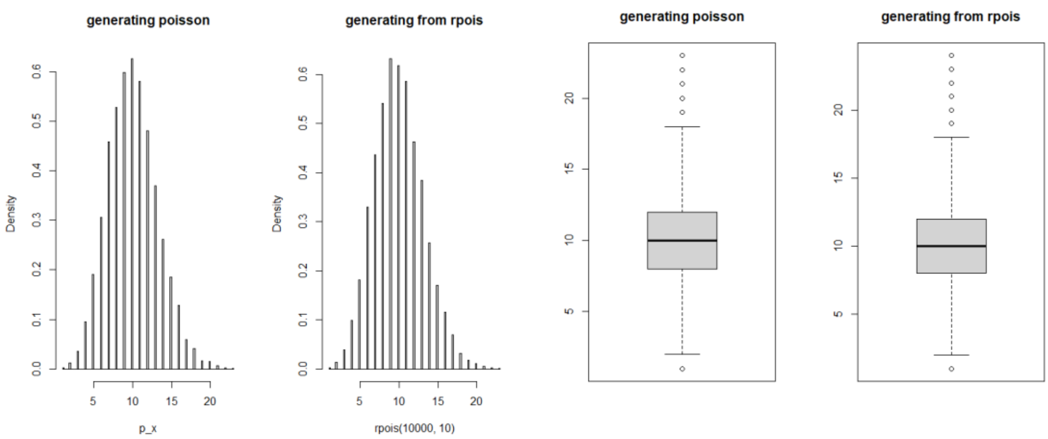
需要一個輔助的“建議分佈” G （概率密度函數 g(y) 已知）來產生候選樣

本。
還需要另一個輔助的均勻分佈 U(0,1) 。
計算一個常數值 c 。——滿足不等式 c*g(x) ≥ f(x)的最小值 c （當然，我們非常希望 c 接近於 1)
樣本生成：
從建議分佈 G 抽樣，得到樣本 Y 。
從分佈 U(0,1) 抽樣，得到樣本 U 。
如果 U ≤ $\frac{f(Y)}{c*g(Y)}$ ，則令 X=Y （接受 Y ），否則繼續執行步驟 1（拒絕）

從圖的結果來看，Acceptance-Rejection approach 方法可以很好的模擬標準常態分佈，然而 Acceptance-Rejection approach 的運行速度卻遠慢於 rnorm 的生成速度(幾乎差了 100 倍)，甚至比 box-muller 方法也慢很多(約慢了 30 倍)，因為在演算法的過程中，Acceptance-Rejection approach 方法一直在尋找合適的值，因此在拒絕以及接受的過程中會花更多時間。

```
> time_box_muller
Time difference of 0.01099396 secs
> time_rnorm
Time difference of 0.003021002 secs
> time_acc_rej
Time difference of 0.306824 secs
```

Problem 2 (a)



我採用泊松過程的定義。假設 μ 代表[0,t]中的事件（到達）數。如果到達間的時間是指數分佈（用參數λ）且獨立，則在[0,t]中發生的到達數，具有 Poisson 的分佈，參數為λt。因此，為了解決這個問題，我們可以重複以下步驟生成指

數(λ)隨機變數，而它們的總和不大於 1(選擇)t = 1)。 由於題目給定的 μ 是 10, 這代表 λT=10, 在這裡 T=1, λ=10。

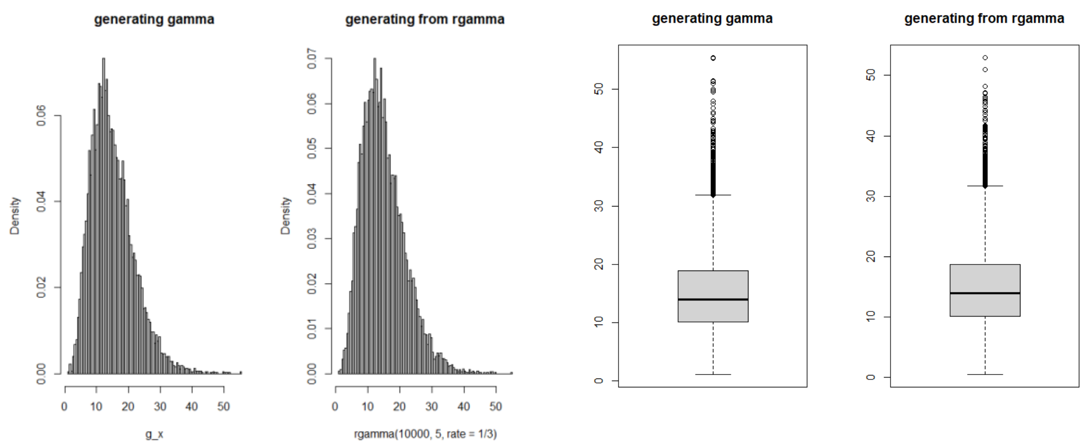
演算法:

1. Set t = 0, X=-1
2. Generate U ~ U(0,1)
3. t = t- 1/λ* log(U)
4. X=X+1
5. 回到步驟二

從圖上結果看來，利用演算法所模擬的 poisson 分布和用 rpois 所生成的 poisson 分布有很高的相似度，在所需的時間上，利用演算法所需的時間比用 rpois 所需的時間長很多(約 58.75 倍)，在效率上差非常大。

```
> poi_generate_time
Time difference of 0.47036  secs
> rpois_time
Time difference of 0.007993937 secs
```

Problem 2 (b)



Gamma 分配可以視為廣義的指數分配。題目中，$\alpha = 5,\ \beta = 3$， 則在生成 gamma 分配所需要的 N $= \alpha =$ 每次所需的 Y~Exp(λ), 而因 $\beta = 3$, 則 $\lambda = \frac{1}{\beta} = \frac{1}{3}$

演算法:

1. 生成 U(0,1)
2. 令 X = $\frac{1}{\lambda}\log U,\ \lambda = \frac{1}{\beta} = \frac{1}{3}$ (這是 Exp(λ))
3. 生成 Y~ Exp(λ), i= 1,.... α
4. 回傳 X=$\sum_{i=1}^{n} Y_i$

從圖上結果看來，利用演算法所模擬的 gamma 分布和用 rgamma 所生成的 gamma 分布有很高的相似度，在所需的時間上，利用演算法所需的時間比用 rgamma 所需的時間長許多(約 20 倍)，因此利用 rgamma 演算法的效率會高許多。

```
> generate_gamma_time
Time difference of 0.06096816 secs
> rgamma_time
Time difference of 0.003000975 secs
```

Problem 3 (a)

Derive the marginal distribution of X

Suppose $X|\mu \sim Poi(\mu), \mu \sim Gamma(\alpha, \beta)$

$$f(x;\mu) = \frac{\mu^x}{x!}e^{-\mu}\frac{1}{\Gamma(\alpha)\beta^\alpha}\mu^{\alpha-1}e^{\frac{-\mu}{\beta}}$$

$$f(x) = \frac{1}{\Gamma(\alpha)\beta^\alpha x!}\int_0^\infty \mu^{x+\alpha-1}e^{-(1+1/\beta)\mu}\,d\mu$$

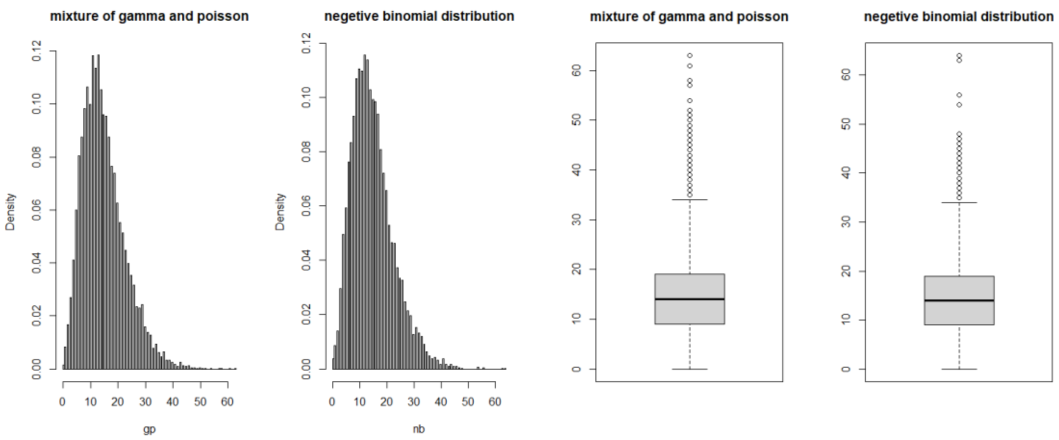
$$= \frac{1}{\Gamma(x+1)\Gamma(\alpha)\beta^\alpha}\Gamma(\alpha+x)(\frac{\beta}{1+\beta})$$

$$= \binom{\alpha-1+x}{x}(\frac{1}{1+\beta})^\alpha(1-\frac{1}{1+\beta})^x$$

$$\text{which } p = \frac{1}{1+\beta}$$

$$\text{Thus, } X \sim NB(\alpha, \frac{1}{1+\beta})$$

Problem 3 (b)



負二項分佈是一個具有 Gamma 混合權重的泊松分佈家族的混合物。因此，它可以看作是一個泊松分佈，其中泊松參數本身就是一個隨機變數，按照 Gamma 分佈進行分佈。負二項分佈的方差大於均值。在泊松分佈中，均值等於方差。因此，N 的無條件分佈比其條件分佈更分散，這是混合物分佈的特點。參數變

數 Lambda 的不確定性具有增加 N 的混合物分佈的無條件方差的作用，混合物分佈的方差有兩個分量，即條件方差的加權平均和條件均值的方差。第二部分代表參數 Lambda 的不確定性所引入的額外方差。

演算法:

利用題二的 poisson 和 gamma 演算法

1. 利用 gamma 演算法生成 λ
2. 利用 poisson 演算法，並將上一步驟生成的 λ 代入
3. 最終會生成 poisson 和 gamma 的混和函數，根據第一題的結果，這個混合函數會服從負二項分布。

從圖上結果看來，利用演算法所生成的 poisson 和 gamma 的混和函數分布和用 rnbinom 所生成的負二項分布有很高的相似度，在所需的時間上，利用演算法所需的時間比用 rnbinom 所需的時間長許多(約 44 倍)，因此利用 rnbinom 演算法的效率會高許多。

```
> time_mix
Time difference of 0.5287311 secs
> nbtime
Time difference of 0.011796 secs
```

Problem 4(a)

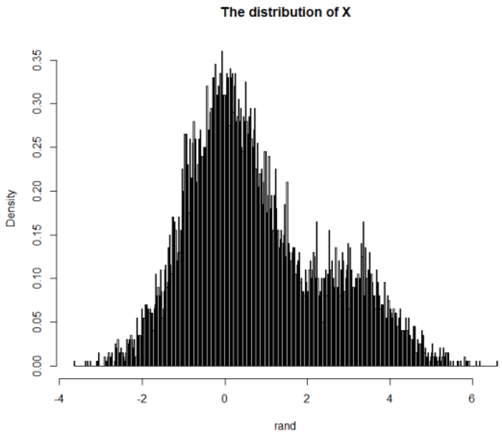
$n = 2$, $(p_1, p_2) = (p_1, 1-p_1)$, $\mu = (0, 3)$, and $\sigma = (1, 1)$

The finite mixture distribution is

$$p_1 N(0,1) + (1-p_1) N(3,1)$$

$$f(x) = \frac{p_1}{\sqrt{2\pi}} exp\{-\frac{1}{2}(x)^2\} + \frac{1-p_1}{\sqrt{2\pi}} exp\{-\frac{1}{2}(x-3)^2\}$$

Problem 4(b)



X 的機率分佈似乎是雙峰分佈，因為我們將兩個正態分佈結合在一起，將兩個機率密度函數乘以其混合概率並相加。因此，圖中在 X=0 和 X=3 處會有峰值，這是兩個正態分佈的平均值。此外，我們可以發現 X=3 處的峰值比 X=0 處的峰值要低，這是由於混合概率的原因。由於 p1 為 0.75，p2 為 0.25，我們可以預計 X=0 時的峰值將比 X=3 時的峰值高約 3 倍。