2 正規分布

(1)

一様乱数をたくさん足し合わせた数の分布は正規分布に近づいていくか，確認しよう．

スクリプト

```
# 初期化
N <- 0; data <- 0
f <- function(data, N) {
  # 描画デバイスの作成
  pdf(sprintf("02\\hist%02d.pdf", N), width=10, height=10)
  hist_dev = dev.cur()
  pdf(sprintf("02\\qq%02d.pdf", N), width=10, height=10)
  qq_dev = dev.cur()

  # データの加算とグラフ表示
  data <- data + runif(10000)
  dev.set(hist_dev)
  hist(data)
  dev.set(qq_dev)
  qqplot(qnorm(ppoints(10000)), data)

  dev.off(hist_dev)
  dev.off(qq_dev)
  return(data)
}

# p-が大きければ正規分布であると判断できるvalue
N <- N + 1; data <- f(data, N); ks.test(data, "pnorm", mean(data), sd(data))

# 後処理
rm(N, f, data)
```

結果

一様乱数を足し合わせたデータのヒストグラムを図 1，分布を正規分布と比較した際の Q-Q プロットを図 2 に示す．また，Kolmogorov-Smirnov 検定を行った際の p 値の値を 1 に示す．

考察

図 1 のヒストグラムから，加算した回数が増えるごとに分布が正規分布に近づいていることが分かる．また，図 2 の Q-Q プロットから，加算した回数が増えるごとにプロットが直線に近づいていることが分かる．Q-Q プロットは，比較した 2 つの分布が似ているほどプロットが直線に近づく．したがって，正規分布に近づいていることが確認できる．さらに，表 1 から，加算した回数が増えるごとに p 値が増加していることが分かる．Kolmogorov-Smirnov 検定では，p 値が高ければ，そのデータの分布は正規分布に基づいているといえる．したがって，表 1 からも正規分布に近づいていくことが分かる．

4 最尤推定

(1)

平均，分散の異なる 2 種の正規乱数を発生させ，その和の分布が式 1 で示す結果に一致しているか，パラメータを最尤推定し確認しよう．

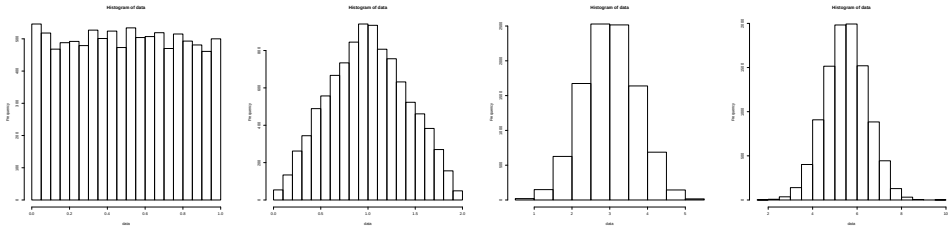
和の密度関数

確率変数 x, y がそれぞれ密度関数 $p_1(x) = \mathcal{N}(x; \mu_1, \sigma_1^2), p_2(y) = \mathcal{N}(y; \mu_2, \sigma_2^2)$ に従うとき，2 つの和 $z = x + y$ は畳み込み積分に従う．

$$p(z) = p_1(x) * p_2(y) = \int_{-\infty}^{\infty} p_1(z-t)p_2(t)dt = \mathcal{N}(z; \mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2) \tag{1}$$

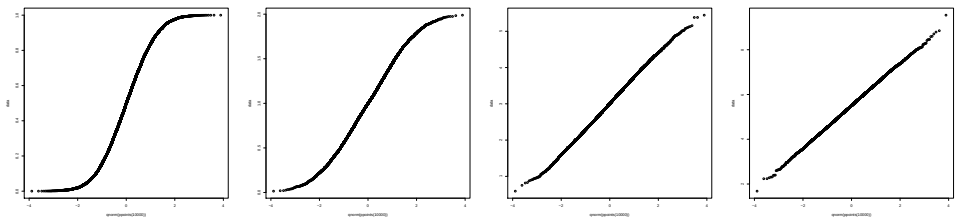
スクリプト

```
data1 <- rnorm(1000000, mean=0.4, sd=1)
data2 <- rnorm(1000000, mean=0.6, sd=2)
data <- data1 + data2

# 最尤推定法 (mle) を使用
library(stats4)
```



(a) 初期化 (一様乱数)　(b) 1 回目　(c) 5 回目　(d) 10 回目

図 1: ヒストグラム



(a) 初期化 (一様乱数)　(b) 1 回目　(c) 5 回目　(d) 10 回目

図 2: Q-Q プロット

表 1: Kolmogorov-Smirnov 検定の p 値

初期化 (一様乱数)	1 回目	5 回目	10 回目
2.20×10^{-16}	0.000691	0.6155	0.9611

```
nLL <- function(mean) -sum(dnorm(data, mean, 2, log=T))
(fit <- mle(nLL, start=list(mean=0)))

# 最尤推定法 (optim) を使用
library(stats)
nLL <- function(arg) -sum(dnorm(data, mean=arg[1], sd=arg[2], log=T))
(fit <- optim(c(0,1), nLL))$par

# 後処理
rm(data1, data2, data, nLL)
```

結果

スクリプトを実行した結果を以下に示す.

```
> nLL <- function(mean) -sum(dnorm(data, mean, 2, log=T))
> (fit <- mle(nLL, start=list(mean=0)))
Call:
mle(minuslogl = nLL, start = list(mean = 0))

Coefficients:
     mean
0.9990161
> nLL <- function(arg) -sum(dnorm(data, mean=arg[1], sd=arg[2], log=T))
> (fit <- optim(c(0,1), nLL))$par
[1] 0.9997631 2.2365586
```

考察

$p_1(x) = \mathcal{N}(x; 0.4, 1^2), p_2(y) = \mathcal{N}(y; 0.6, 2^2)$ に従う確率変数 x, y の和 z の密度関数は $p(z) = \mathcal{N}(z; 1, 5)$ となるはずである．最尤推定を行ったところ，平均はどちらの手法でも 1 に近い値となった．また，標準偏差も $\sqrt{5} \fallingdotseq 2.236$ と近い値となった．したがって，式 1 とほぼ一致すると考えられる．

6 混合分布

(1)

MASS パッケージに含まれる多変量正規乱数生成関数 `mvrnorm()` を使って，3 つのガウス成分からなる混合分布から発生させた，2 次元の乱数ベクトル系列を作成しよう．

スクリプト

以下のような 2 変数 3 成分ガウス混合分布に従う確率変数 x を考える．

$$p(x) = 0.2\mathcal{N}(x; \mu_a, \Sigma_a) + 0.5\mathcal{N}(x; \mu_b, \Sigma_b) + 0.3\mathcal{N}(x; \mu_c, \Sigma_c) \tag{2}$$

$$\mu_a = \begin{bmatrix} -5 \\ -10 \end{bmatrix}, \Sigma_a = \begin{bmatrix} 4 & 3 \\ 3 & 9 \end{bmatrix}, \mu_b = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma_b = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}, \mu_c = \begin{bmatrix} 7 \\ 5 \end{bmatrix}, \Sigma_c = \begin{bmatrix} 4 & 0 \\ 0 & 9 \end{bmatrix} \tag{3}$$

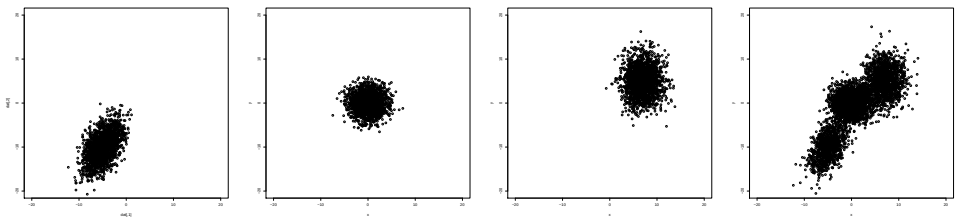
```
library(MASS)

# 乱数ベクトル系列作成関数
rgmix <- function(n) {
  y <- c();
  for(i in 1:n) {
    u <- runif(1);
    if(u < 0.2) {
      v <- mvrnorm(1, c(-5, -10), matrix(c(4, 3, 3, 9), 2, 2))
    } else if(u < 0.7) {
      v <- mvrnorm(1, c(0, 0), matrix(c(4, 0, 0, 4), 2, 2))
    } else {
      v <- mvrnorm(1, c(7, 5), matrix(c(4, 0, 0, 9), 2, 2))
    }
    y <- c(y, v)
  }
  y <- matrix(y, ncol=2, byrow=T)
  return(y)
}

# 描画デバイスの作成
pdf("02\\mvrnorm-plot.pdf", width=10, height=10)
dev = dev.cur()

# データ生成
dat <- rgmix(5000)
dat <- data.frame(x=dat[,1], y=dat[,2])

# プロット
dev.set(dev)
plot(dat, xlim=c(-20, 20), ylim=c(-20, 20))
dev.off(dev)
```

結果

それぞれの正規分布のみで生成した乱数のプロットを図 3 (a)–(c)，ガウス混合分布で生成した乱数のプロットを図 3(d) に示す．

考察

図 3 から，それぞれの分布が係数に合わせて組み合わされていることが分かる．密度は，係数が 0.5 の第 2 項が最も濃くなっており，係数が 0.2 の第 1 項が最も薄くなっていることも確認できる．



(a) 第 1 項の正規分布　(b) 第 2 項の正規分布　(c) 第 3 項の正規分布　(d) ガウス混合分布

図 3: 生成乱数のプロット

7 EM アルゴリズム

(1)

6 で作成した乱数ベクトル系列に対して EM アルゴリズムを適用し，元の 2 変数 3 成分ガウス混合分布が推定できるか試してみよう．

スクリプト

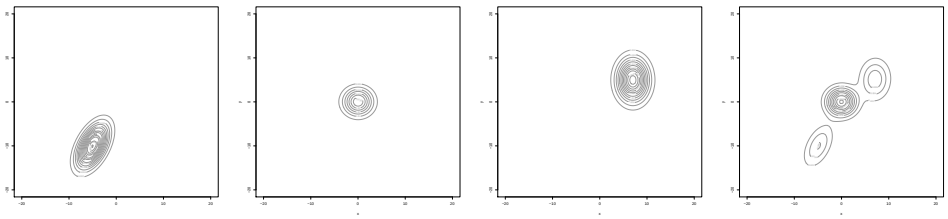
```
#install.packages("mclust")
library(mclust)

# 描画デバイスの作成
pdf("02\\mvrnorm-density.pdf", width=10, height=10)
dev = dev.cur()

# アルゴリズムの適用EM
res <- densityMclust(dat)

# プロット
dev.set(dev)
density(res, what="density", xlim=c(-20, 20), ylim=c(-20, 20))
dev.off(dev)

# 推測されたパラメータの確認
res$parameter$pro
res$parameter$mean
res$parameter$variance$sigma
```

結果

それぞれの正規分布のみで生成した乱数に対し EM アルゴリズムを適用した結果を図 4 (a)–(c)，ガウス混合分布で生成した乱数に対し EM アルゴリズムを適用した結果を図 4(d) に示す．

また，スクリプトの実行結果を以下に示す．

```
> res$parameter$pro
[1] 0.5087467 0.2898862 0.2013671
> res$parameter$mean
         [,1]      [,2]       [,3]
```



(a) 第 1 項の正規分布　(b) 第 2 項の正規分布　(c) 第 3 項の正規分布　(d) ガウス混合分布

図 4: 生成乱数のプロット

```
x -0.01517243 7.061727  -4.918003
y  0.01686892 5.102905 -10.013976
> res$parameter$variance$sigma
, , 1
           x          y
x 3.93317810 0.02366543
y 0.02366543 3.95526324

, , 2
          x         y
x 4.1815102 0.2287938
y 0.2287938 8.6670173

, , 3
         x        y
x 3.847105 2.898212
y 2.898212 9.065151
```

考察

図 3 から，それぞれの分布が係数に合わせて組み合わされていることが分かる．密度関数の値は，係数が 0.5 の第 2 項が最も高くなっており，係数が 0.2 の第 1 項が最も低くなっていることも確認できる．また，第 2 項の等高線の間隔は狭く，第 1 項，第 3 項の等高線の間隔が広くなっていることが確認できる．このことから，元の 2 変数 3 成分ガウス混合分布が推定できたと考えられる．

推測されたパラメータは，係数の高い順に，第 2 項，第 3 項，第 1 項の順で並んでいた．推測された係数は，順に $0.509, 0.290, 0.201$ となり，設定した $0.5, 0.3, 0.2$ と一致している．また，平均 μ，分散共分散行列 Σ についても式 3 で設定した値と概ね一致していた．

10 区間推定

(1)

`rnorm(N, mean=0.86, sd=1.3)` から生成される乱数集合を用いてその平均値の 95% 信頼区間が，N の変化によってどう変わるか，グラフに示してみよう．また，この乱数集合の分散の信頼区間についても同様に示してみよう．

スクリプト

```
f <- function(max) {
  sdkn <- c() # 既知の標準偏差を用いた信頼区間正規分布 ()
  sdun <- c() # 推定標準偏差を用いた信頼区間正規分布 ()
  sdkt <- c() # 既知の標準偏差を用いた信頼区間 (分布t)
  sdut <- c() # 推定標準偏差を用いた信頼区間 (分布t)
  vux <- c() # 不偏分散を用いた区間推定カイ二乗分布 ()

  for(N in 2:max) {
```

```
    # 初期化
    dat <- rnorm(N, mean=0.86, sd=1.3)
    mu <- mean(dat); sigma <- sd(dat)

    # 平均値の区間推定
    sdkn <- rbind(sdkn, (qnorm(c(0.025, 0.975)) * 1.3 / sqrt(N) + mu))
    sdun <- rbind(sdun, (qnorm(c(0.025, 0.975)) * sigma /  sqrt(N) + mu))
    sdkt <- rbind(sdkt, (qt(c(0.025, 0.975), df=(N-1)) * 1.3 / sqrt(N) + mu))
    sdut <- rbind(sdut, (qt(c(0.025, 0.975), df=(N-1)) * sigma /  sqrt(N) + mu))
    # 分散の区間推定
    vux <- rbind(vux, ((N-1) * sigma**2 / qchisq(c(0.975, 0.025), df=(N-1))))
  }

  res <- list(sdkn, sdun, sdkt, sdut, vux)
  return(res)
}

# グラフの保存
save <- function(res){
  for(i in 1:length(names)) {
    pdf(sprintf("02\\%s-%d.pdf", names[i], N), width=10, height=10)
    devnum = dev.cur()

    a = res[[i]]

    dev.set(devnum)
    plot(0, 0, type = "n", xlab = "N", ylab = names[i],
      xlim=c(0, N), ylim=c(tv[i]-2, tv[i]+2))
    lines(2:N-1, a[,1], col="red"); lines(2:N-1, a[,2], col="red")
    lines(c(0, N-1), c(tv[i], tv[i]), col="green4")
    dev.off(devnum)
  }
}

names <- c("sdkn", "sdun", "sdkt", "sdut", "vux")
tv <- c(0.86, 0.86, 0.86, 0.86, 1.3**2)
N <- 500; res <- f(N); save(res)
```

結果

データ数 N を変化させながら，平均 μ を区間推定した結果を図 5 に，分散 σ^2 を区間推定した結果を図 6 に示す．グラフの横軸は N，縦軸は値である．上の赤線が推定区間の上の値を，下の赤線が推定区間の下の値を，緑の直線が母集団の平均，および分散を示す．

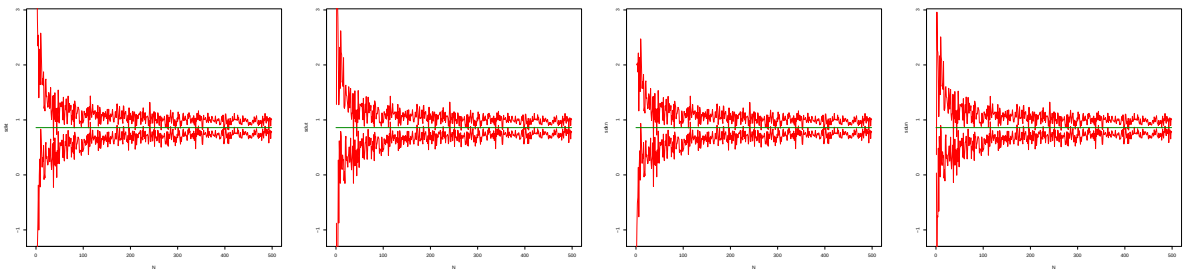
考察

図 5, 6 より，平均，分散共に 95% 信頼区間はデータ数 N を増やすほど狭くなっていることが分かる．したがって，N を増やすほど推定の精度が上がることが分かる．

平均の区間推定においては，$N = 200$ 程度から大きく差が出ないことが確認できる．正規分布を用いて区間推定を行った場合 (図 5(c)) と，t 分布を用いて区間推定を行った場合 (図 5(a)) を比較すると，N が小さいときは正規分布では推定区間が非常に大きく，精度が低いが，t 分布では推定区間が狭く，比較的精度が高いことが分かる．既知の標準偏差を用いた場合 (図 5(a), (c)) と推定標準偏差を用いた場合 (図 5(b), (d)) の間には大きな差はないと考える．

また，図 5 に比べ，図 6 は N が小さいときの推定区間が広く，その後もなかなか推定区間が狭くならな

い．このことから，分散の区間推定においては，推定の精度を高くするためには，平均の区間推定よりも多くのデータ数が必要であると考えられる．



(a) 既知の標準偏差使用 (t 分布)

(b) 推定標準偏差使用 (t 分布)

(c) 既知の標準偏差使用 (正規分布)

(d) 推定標準偏差使用 (正規分布)

図 5: 平均の区間推定 (95% 信頼区間)

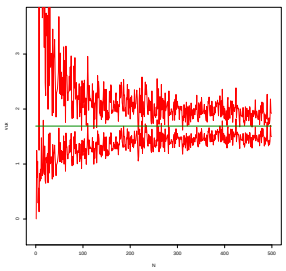


図 6: 分散の区間推定 (95% 信頼区間)