Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий фізико-технічний інститут

Реалізація ЕМ-алгоритму

предмет «Марковські моделі та їхнє застосування»

Роботу виконав:
Студент групи ФІ-91,
Цибульник Антон Владиславович

Роботу перевірила:
Ніщенко Ірина Іванівна

2022

Теоретична довідка

Нехай задана послідовність m незалежних випадкових величин $Y_i \sim N(\mu_i, \sigma_i^2)$, кожна з яких має нормальний розподіл. На додачу введемо дискретну випадкову величину X зі значеннями в множині $\{1, 2, \dots, m\}$, для якої виконується закон розподілу $\sum\limits_{i=1}^{m} p_i = 1$ при значеннях $p_i = P(X = i)$.

Сумішшю нормальних розподілів $N(\mu_1, \sigma_1^2), \dots, N(\mu_m, \sigma_m^2)$ зі змішувальними коефіцієнтами $(p_i)_{i=\overline{1,m}}$ називають випадкову величину

$$Z = \sum_{i=0}^{m} Y_i \cdot \mathbb{1}(X = i) = \begin{cases} Y_1, & \text{при } X = 1, \\ Y_2, & \text{при } X = 2, \\ \dots \\ Y_m, & \text{при } X = m \end{cases}$$

із такою щільністю розподілу:

$$f_Z(t) = \sum_{i=1}^{m} p_i \, \tfrac{1}{\sqrt{2\pi}\,\sigma_i} \, e^{-\frac{(t-\mu_i)^2}{2\sigma_i^2}}$$

Розглянемо ЕМ-алгоримт для задачі оцінки невідомих параметрів деякої суміші нормальних розподілів: $\theta = (p_1, \dots, p_m;\ \mu_1, \dots, \mu_m;\ \sigma_1^2, \dots, \sigma_m^2)$. Кожна ітерація алгоритму складається із таких кроків:

(1) Ініціалізація: нехай $\theta^{(0)} = (p_1^{(0)}, \dots, p_m^{(0)};\ \mu_1^{(0)}, \dots, \mu_m^{(0)};\ \sigma_1^{2^{(0)}}, \dots, \sigma_m^{2^{(0)}})$ – деяке початкове наближення шуканих параметрів;

(2) E-крок (expectation): обчислюємо математичне сподівання логарифма функції правдоподібності, використовуючи наближені параметри з попереднього кроку;

(3) M-крок (maximization): вираховуємо значення, яке максимізує щойно знайдене математичне сподівання;

(4) Крок ітерації: покладаємо покращену оцінку параметрів як точку відліку для наступного витка алгоритму. Припиняємо алгоритм при досягненні певного числа ітерацій або за виконання умов збіжності (які можна досягти у випадку нормального розподілу).

Наведемо явний вигляд формул переоцінки шуканих m параметрів суміші нормального розподілу для заданої кількості n спостережень $\{Y_k\}_{k \geqslant 1}$:

$$p_j^{(t+1)} = \frac{1}{n} \sum_{k=1}^{n} \omega_j(y_k \mid \theta^{(t)}),$$

$$\mu_j^{(t+1)} = \frac{\sum\limits_{k=1}^{n} y_k \omega_j(y_k \mid \theta^{(t)})}{\sum\limits_{k=1}^{n} \omega_j(y_k \mid \theta^{(t)})},$$

$$\sigma_j^{2^{(t+1)}} = \frac{\sum\limits_{k=1}^{n} (y_k - \mu_j^{(t+1)})^2 \omega_j(y_k \mid \theta^{(t)})}{\sum\limits_{k=1}^{n} \omega_j(y_k \mid \theta^{(t)})},$$

де величина $\omega_j(y_k \mid \theta^{(t)})$ матиме такий вигляд:

$$\omega_j(y_k \mid \theta^{(t)}) = \frac{p_j^{(t)} \frac{1}{\sqrt{2\pi}\,\sigma_j^{(t)}}\, e^{-\frac{(y_k-\mu_j^{(t)})^2}{2\sigma_j^{2^{(t)}}}}}{\underbrace{\sum\limits_{i=1}^{m} p_i^{(t)} \frac{1}{\sqrt{2\pi}\,\sigma_j^{(t)}}\, e^{-\frac{(y_k-\mu_i^{(t)})^2}{2\sigma_i^{2^{(t)}}}}}_{f(y_k|\theta^{(t)})}}$$

Оцінка параметрів суміші двох розподілів

Нехай згенеровано величини $\{Y_k\}_{k\geqslant 1}$ із суміші такого виду:

$$\tfrac{2}{3}N(10,1) + \tfrac{1}{3}N(5,1)$$

Зобразимо гістограму та криву набору з $n = 500$ елементів:

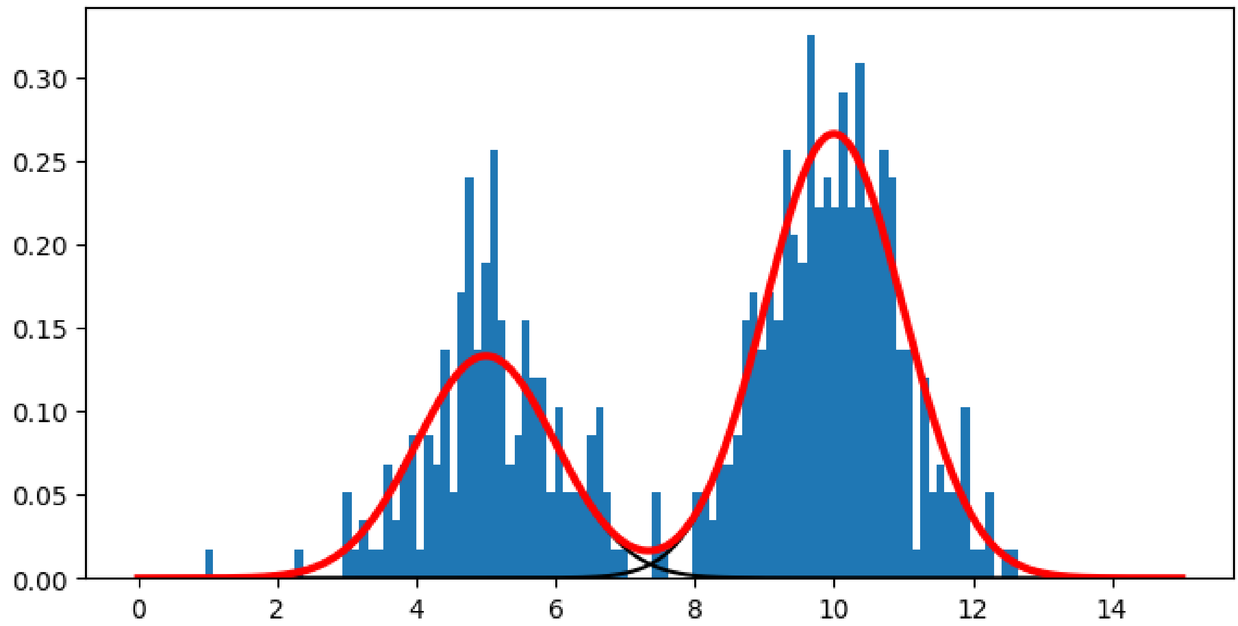


Рис. 1: Гістограма суміші двох розподілів

Програматично таке генерування виконано засобами мови `Python`: у рядках 2-4 Лістингу 1 на черговій ітерації до загального масиву величин додається елемент, який згідно заданих імовірностей матиме розподіл або $\frac{2}{3}N(10,1)$, або $\frac{1}{3}N(5,1)$.

Лістинг 1: Генерування даних

```
1 ksi = []
2 for i in range(n):
3     index = np.argmax(np.random.multinomial(1, p))
4     ksi_i = np.random.normal(mu[index], sigma[index])
5     ksi.append(ksi_i)
```

Наступним етапом застосуємо ЕМ-алгоритм, вважаючи, що параметри суміші p та $\mu = (\mu_1, \mu_2)$ є невідомими, а у нас на руках лише набір з n згенерованих спостережень $\{y_k\}$ та відомі значення дисперсій: $\sigma^2 = (\sigma_1^2, \sigma_2^2) = (1, 1)$. Програмна ралізація переоцінки параметів p та μ зображеня на лістингу нижче:

Лістинг 2: Функція ЕМ-алгоритму

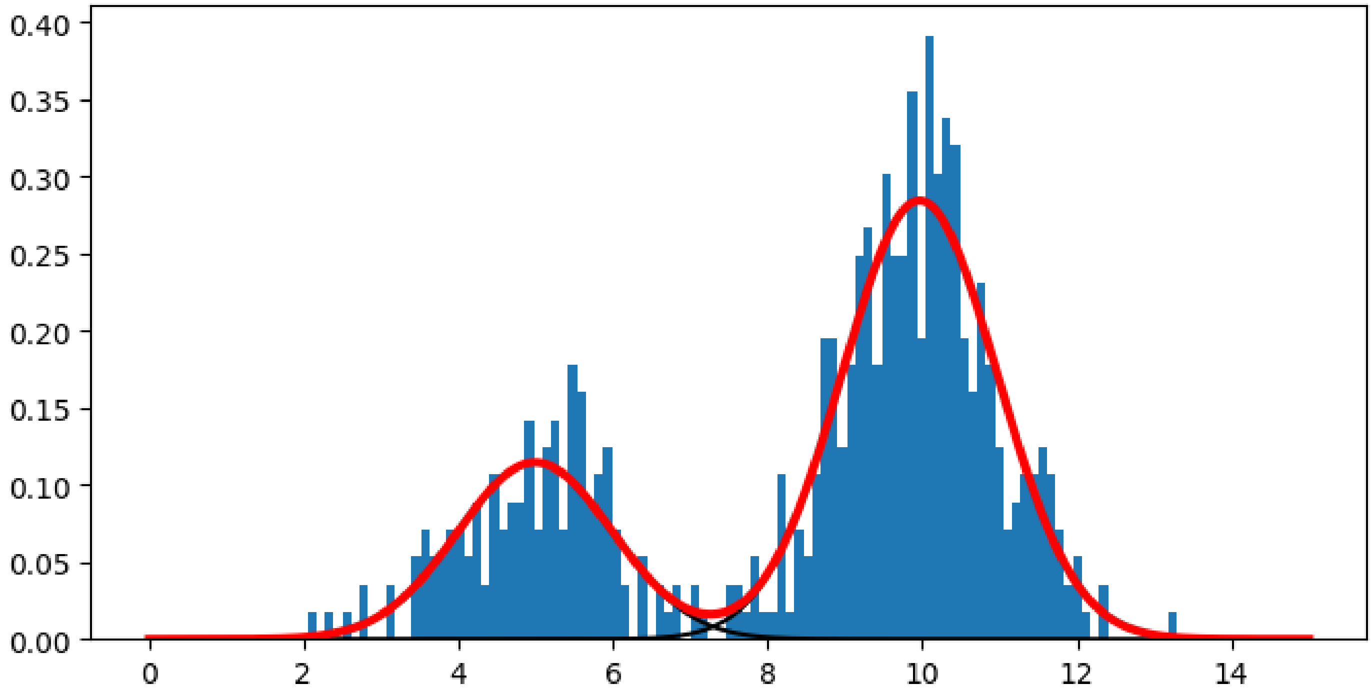
```
def EM(Q_previous, y):
Q = Q_previous

# overestimation of probabilities
sum = 0
for k in range(n):
    f = Q[0]*np.exp(-0.5*pow((y[k]-Q[2]),2)) +
        (1-Q[0])*np.exp(-0.5*pow((y[k]-Q[1]),2))
    sum += Q[0]*np.exp(-0.5*pow((y[k]-Q[2]),2))/f

Q[0] = sum/n

# overestimation of mathematical expectation
w1, sum_up, sum_down = 0, 0, 0
for k in range(n):
    f = Q[0]*np.exp(-0.5*pow((y[k]-Q[2]),2)) +
        (1-Q[0])*np.exp(-0.5*pow((y[k]-Q[1]),2))
    w1 = (1-Q[0])*np.exp(-0.5*pow((y[k]-Q[1]),2))/f
    sum_up += y[k]*w1
    sum_down += w1

Q[1] = sum_up/sum_down

w2, sum_up, sum_down = 0, 0, 0
for k in range(n):
    f = Q[0]*np.exp(-0.5*pow((y[k]-Q[2]),2)) +
        (1-Q[0])*np.exp(-0.5*pow((y[k]-Q[1]),2))
    w2 = Q[0]*np.exp(-0.5*pow((y[k]-Q[2]),2))/f
    sum_up += y[k]*w2
    sum_down += w2

Q[2] = sum_up/sum_down

return Q
```

Результати роботи алгоритму

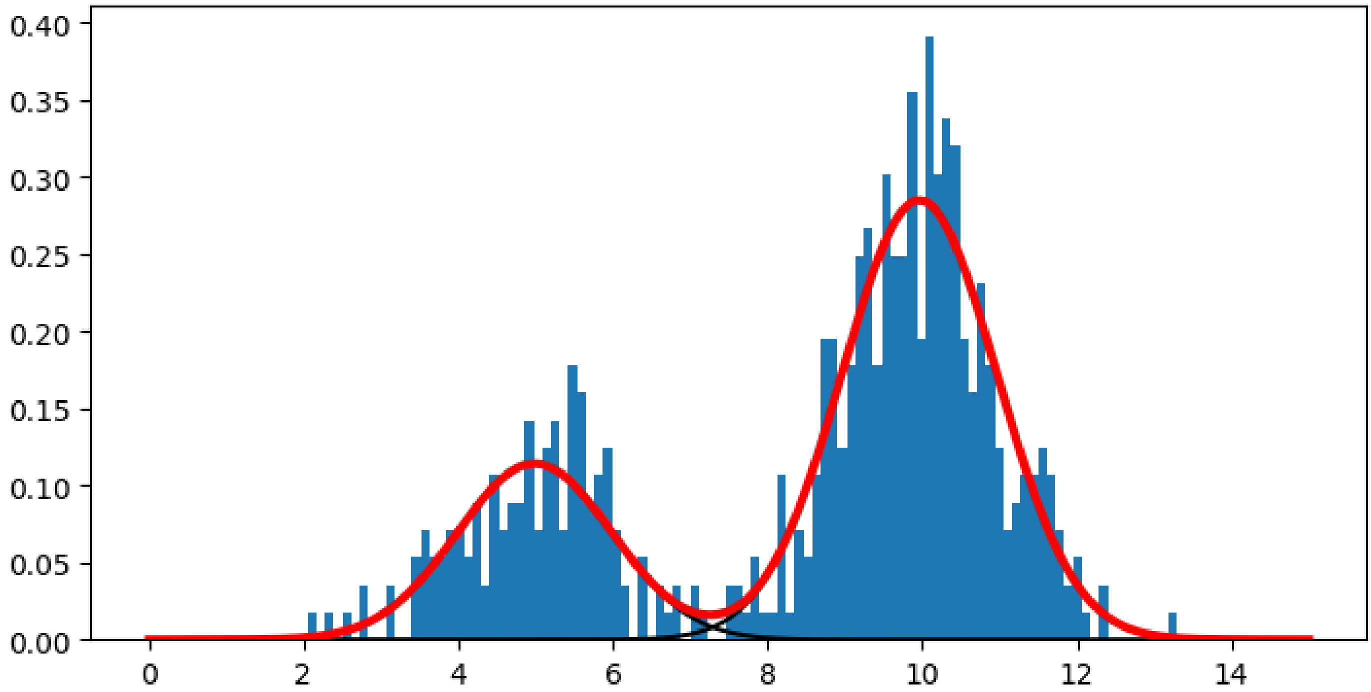
Від ітерації до ітерації

Зобразимо результати роботи алгоритму, взявши за початкове наближення, наприклад, такі параметри:

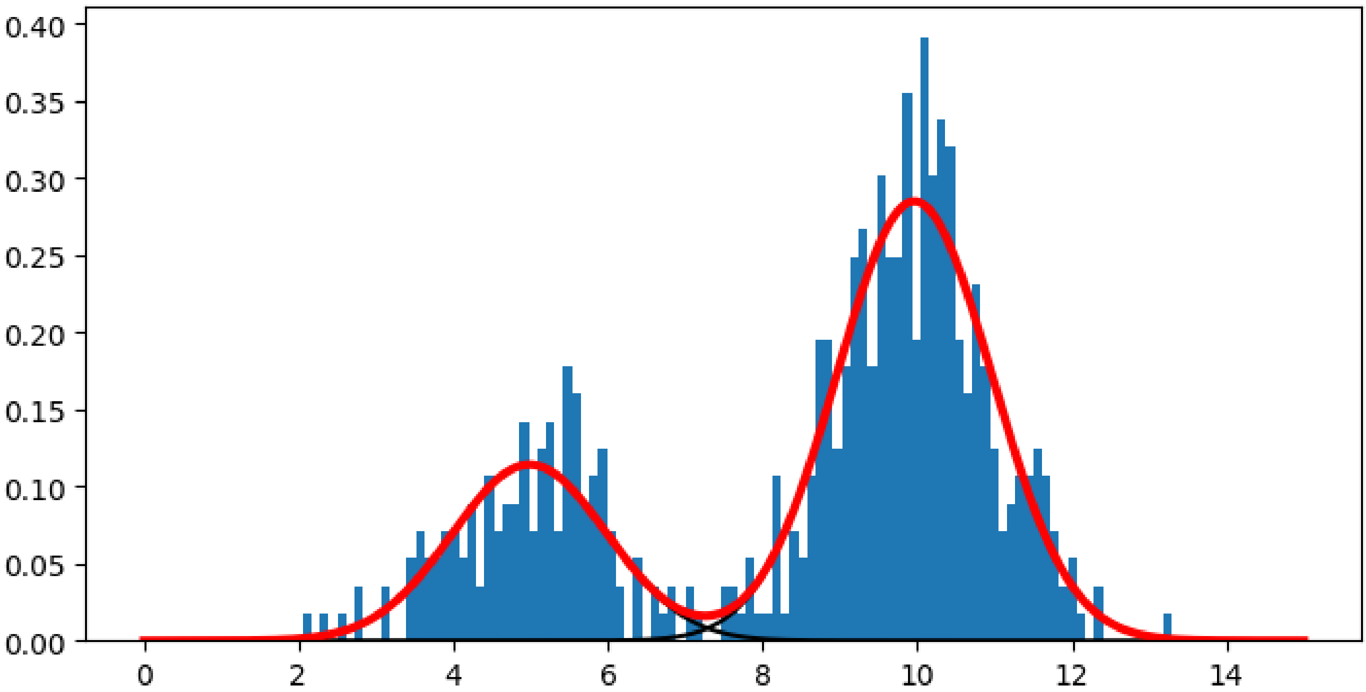
$$\theta^{(0)} = (p^{(0)}, \mu_1^{(0)}, \mu_2^{(0)}) = (0.8,\ 8,\ 7)$$



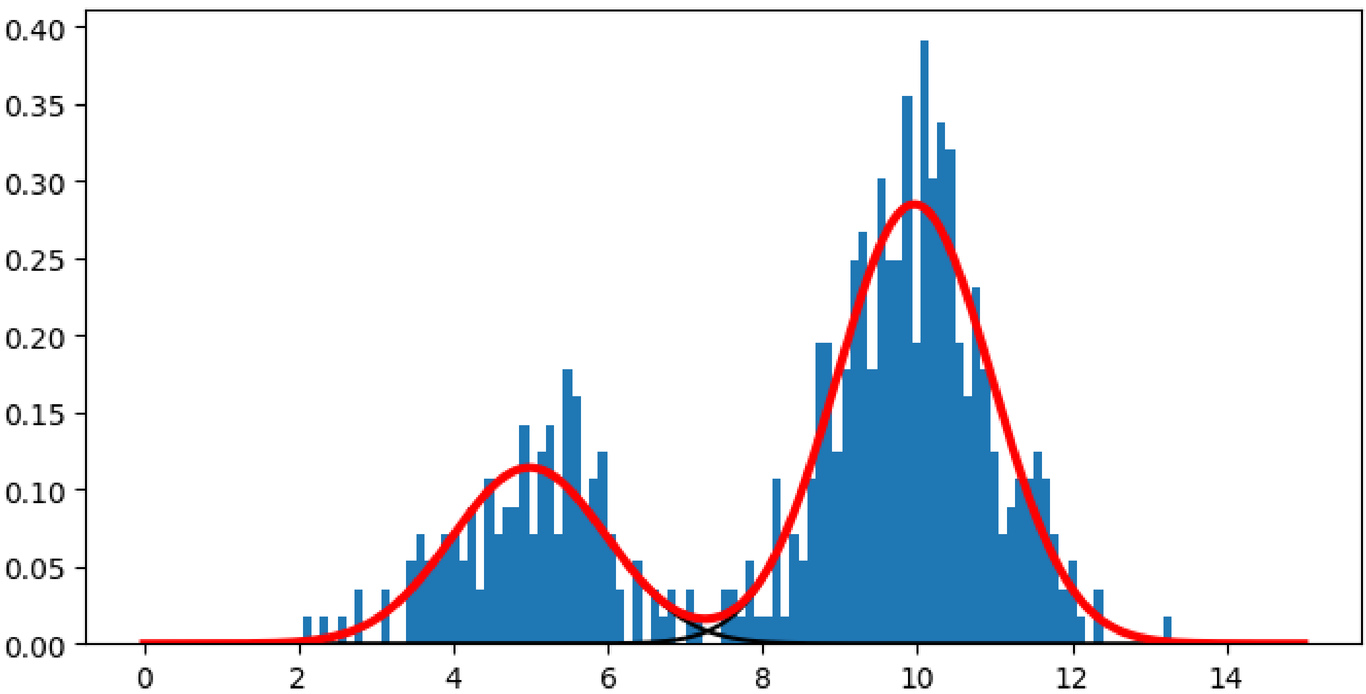
Ітерація №2: [0.28719535408729163, 9.964985178812421, 4.993402055506498]



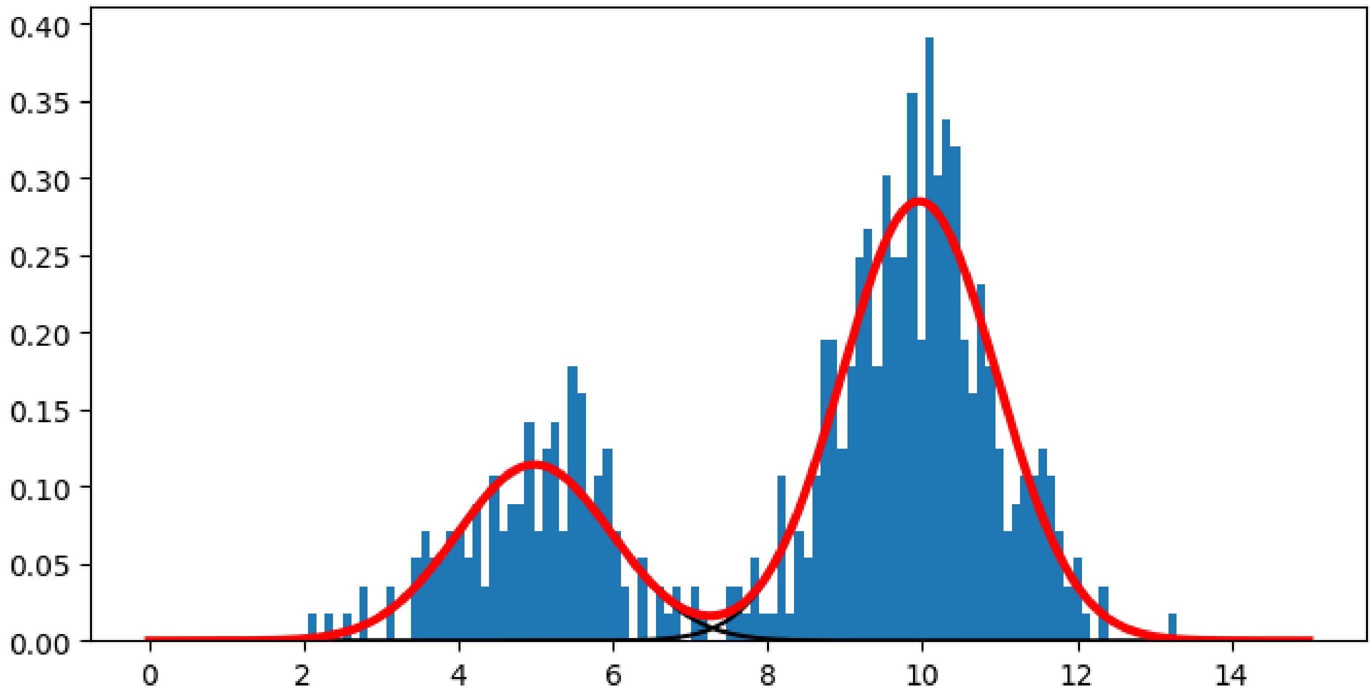
Ітерація №4: [0.28609300530597354, 9.961857617679273, 4.986958379907583]



Ітерація №8: [0.2860821898239672, 9.961825817267654, 4.986907989367006]



Ітерація №16: [0.28608218911775435, 9.961825815224344, 4.9869079862629935]



Ітерація №32: [0.2860821891177544, 9.961825815224342, 4.9869079862629935]

В залежності від початкового значення

Зауважимо, що у прикладі вище при $\theta^{(0)} = (p^{(0)}, \mu_1^{(0)}, \mu_2^{(0)}) = (0.8,\ 8,\ 7)$ алгоритм демонструє збіжність математичних сподівань із точністю $\varepsilon = 0.04$ вже за $k = 3$ ітерацій. Прослідкуємо за збіжністю до аналогічної точності в залежності від вибору початкових параметрів суміші:

Початкові параметри	Ітерації	Точність
$(p^{(0)} = 0.8,\ \mu_1^{(0)} = 12,\ \mu_2^{(0)} = 3)$	2	0.04
$(p^{(0)} = 0.8,\ \mu_1^{(0)} = 15,\ \mu_2^{(0)} = 10)$	6	
$(p^{(0)} = 0.8,\ \mu_1^{(0)} = 18,\ \mu_2^{(0)} = -2)$	3	
$(p^{(0)} = 0.5,\ \mu_1^{(0)} = 12,\ \mu_2^{(0)} = 3)$	2	
$(p^{(0)} = 0.5,\ \mu_1^{(0)} = 15,\ \mu_2^{(0)} = 10)$	6	
$(p^{(0)} = 0.5,\ \mu_1^{(0)} = 18,\ \mu_2^{(0)} = -2)$	3	
$(p^{(0)} = 0.2,\ \mu_1^{(0)} = 12,\ \mu_2^{(0)} = 3)$	2	
$(p^{(0)} = 0.2,\ \mu_1^{(0)} = 15,\ \mu_2^{(0)} = 10)$	6	
$(p^{(0)} = 0.2,\ \mu_1^{(0)} = 18,\ \mu_2^{(0)} = -2)$	3	

Табл. 1: Порівняння результатів в залежності від початкової точки

Оцінка параметрів суміші чотирьох розподілів

Нехай тепер згенеровано величини $\{Y_k\}_{k\geqslant 1}$ із суміші таких чотирьох нормальних розподілів:

$$\tfrac{1}{4}N(-20, 2^2) + \tfrac{1}{4}N(0, 5^2) + \tfrac{1}{4}N(3, 4^2) + \tfrac{1}{4}N(10, 3^2)$$

Зобразимо гістограму та криву набору з $n = 1000$ спостережень (чорним кольором позначено криві окремих складових суміші):

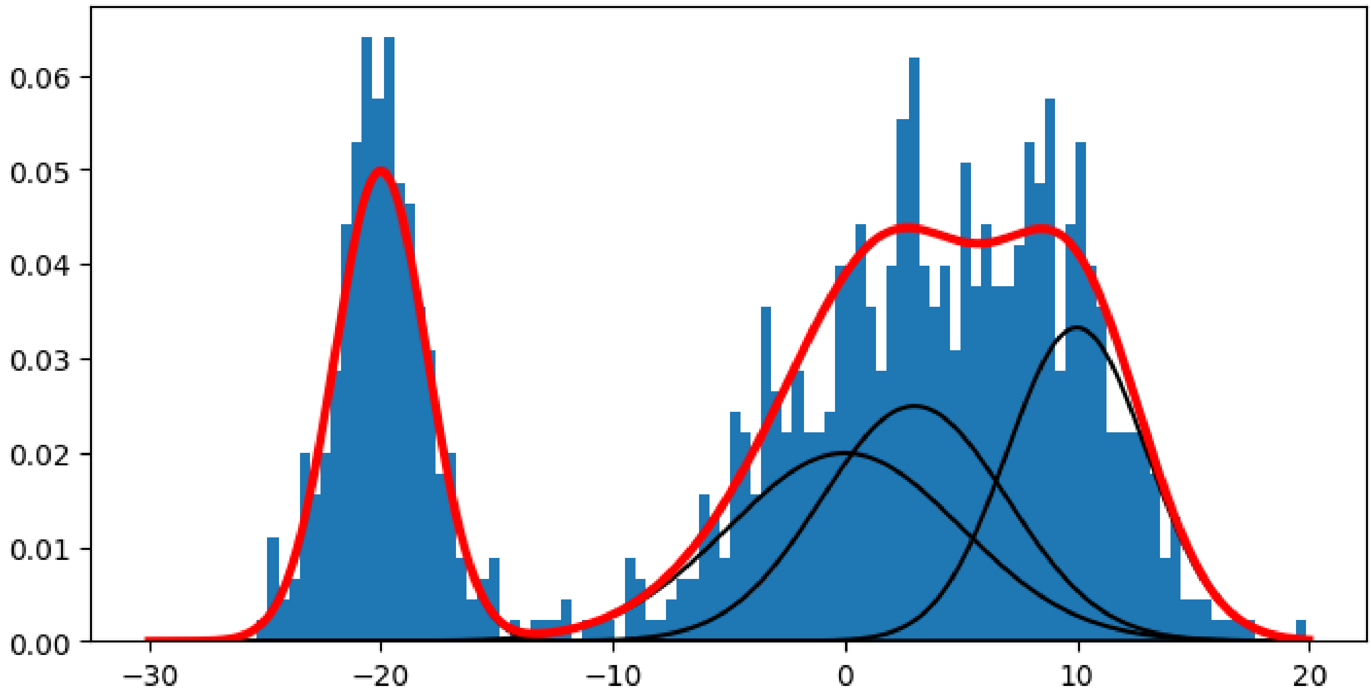


Рис. 2: Гістограма суміші чотирьох розподілів

Генерування значень відбувається за аналогічною схемою, як це показано на Лістингу 1. А от сам алгоритм цього разу реалізовано за загальними формулами, наведеними у пешому розділі теоретичної довідки (тобто для довільної кількості невідомих параметрів):

Лістинг 3: Узагальнена функція ЕМ-алгоритму

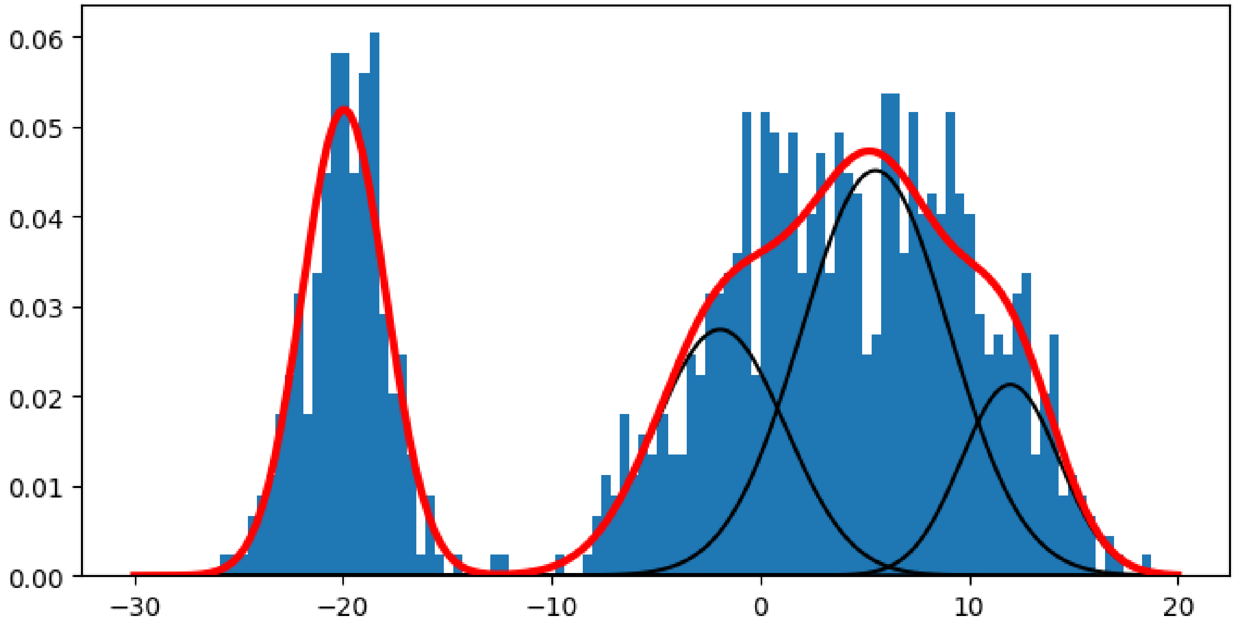
```
def EM(Q_previous, y):
    Q = Q_previous

    n = len(y)
    m = len(Q[0])

    # overestimation of probabilities
    for j in range(m):
        f = np.array([0.0 for i in range(n)])
        for k in range(n):
            for i in range(m):
                f[k] += Q[0][i]*np.exp(-pow(y[k]-Q[1][i],2)/(2*Q[2][i]**2))/Q[2][i]
            if f[k] == 0.0: f[k] = f.mean()
```

```
        sum = 0
        for k in range(n):
            sum += Q[0][j]*np.exp(-pow(y[k]-Q[1][j],2)/(2*Q[2][j]**2))/Q[2][j]/f[k]

        Q[0][j] = sum/n

    # overestimation of mathematical expectations
    for j in range(m):
        f = np.array([0.0 for i in range(n)])
        for k in range(n):
            for i in range(m):
                f[k] += Q[0][i]*np.exp(-pow(y[k]-Q[1][i],2)/(2*Q[2][i]**2))/Q[2][i]
            if f[k] == 0.0: f[k] = f.mean()

        w = np.array([0.0 for i in range(n)])
        sum_up, sum_down = 0, 0
        for k in range(n):
            w[k] = Q[0][j]*np.exp(-pow(y[k]-Q[1][j],2)/(2*Q[2][j]**2))/Q[2][j]/f[k]
            sum_up += y[k]*w[k]
            sum_down += w[k]

        Q[1][j] = sum_up/sum_down

    # overestimation of variances
    for j in range(m):
        f = np.array([0.0 for i in range(n)])
        for k in range(n):
            for i in range(m):
                f[k] += Q[0][i]*np.exp(-pow(y[k]-Q[1][i],2)/(2*Q[2][i]**2))/Q[2][i]
            if f[k] == 0.0: f[k] = f.mean()

        w = np.array([0.0 for i in range(n)])
        sum_up, sum_down = 0, 0
        for k in range(n):
            w[k] = Q[0][j]*np.exp(-pow(y[k]-Q[1][j],2)/(2*Q[2][j]**2))/Q[2][j]/f[k]
            sum_up += pow(y[k]-Q[1][j],2)*w[k]
            sum_down += w[k]

        Q[2][j] = np.sqrt(sum_up/sum_down)

    return Q

Q = np.array([[0.2, 0.3, 0.2, 0.3], [-15, -5, 5, 15], [1, 2, 2, 2]])

for i in range(64):
    Q = EM(Q, ksi)
    if (i in [2,4,8,16,32,64]):
        draw(ksi, Q[0], Q[1], Q[2], -30, 20)
```

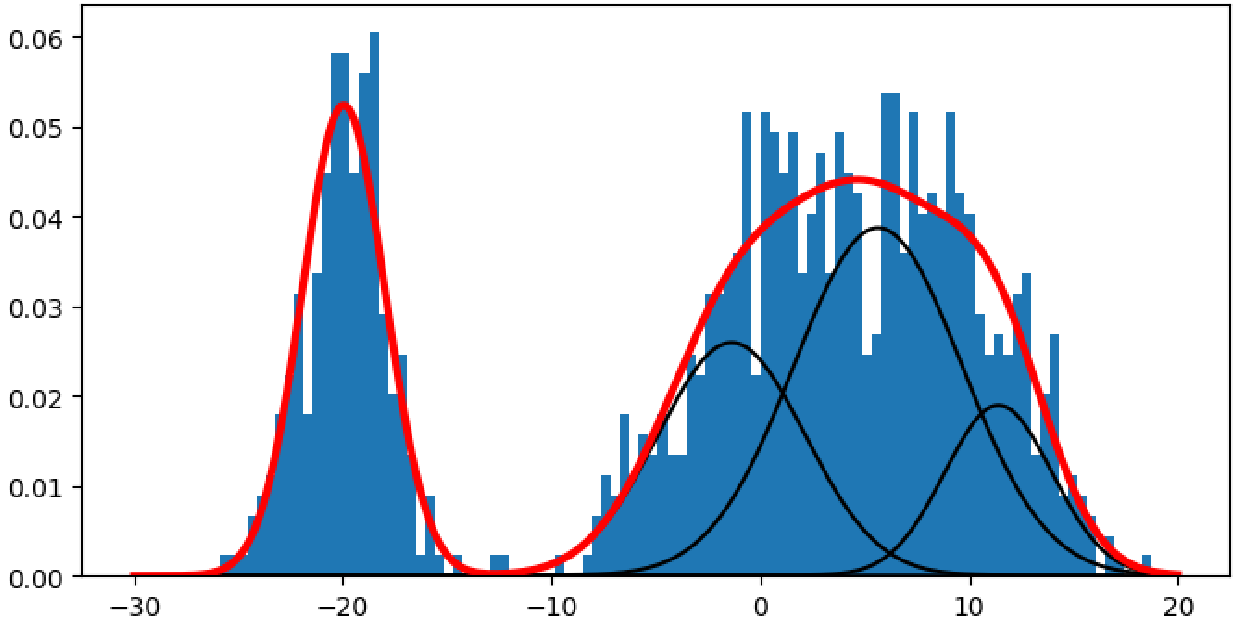
Результати роботи алгоритму

Зобразимо результати роботи алгоритму на $k = 4, 16, 64$ ітераціях для різних початкових точок. Наприклад, спершу візьмемо такі параметри:

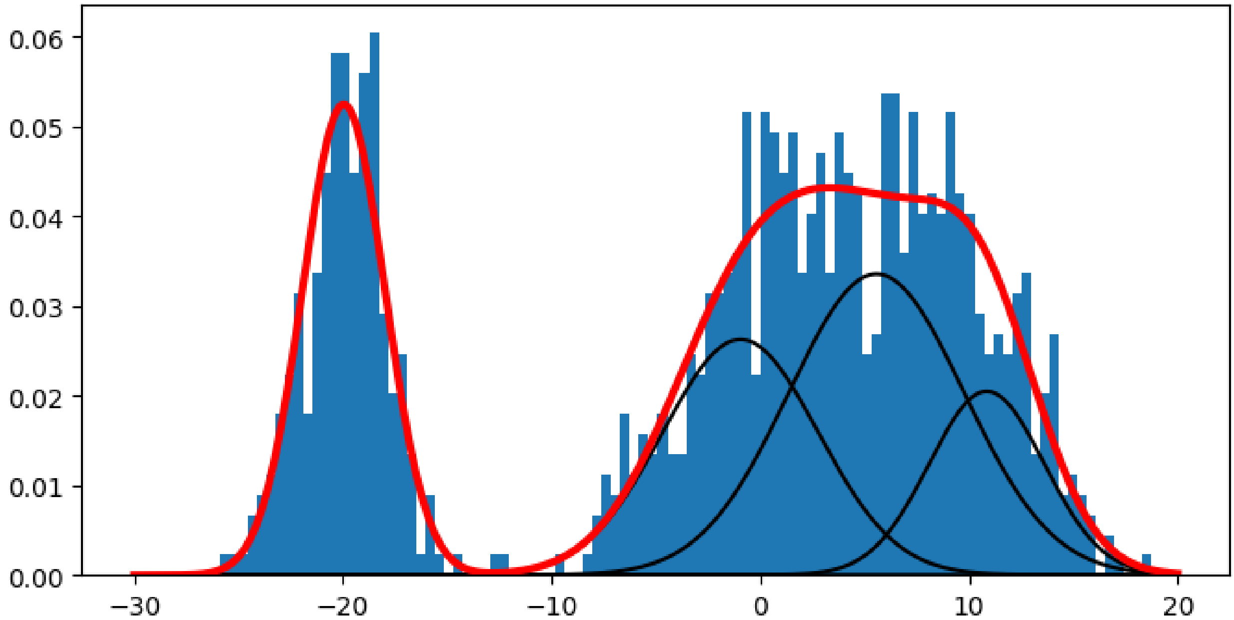
$$\theta^{(0)} = \Big((p_1^{(0)}, p_2^{(0)}, p_3^{(0)}, p_4^{(0)}), (\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}, \mu_4^{(0)}), (\sigma_1^{(0)}, \sigma_2^{(0)}, \sigma_3^{(0)}, \sigma_4^{(0)})\Big)$$
$$= ((0.2,\ 0.3,\ 0.2,\ 0.3),\ (-15, -5,\ 5,\ 15),\ (1,\ 2,\ 2,\ 2))$$



Ітерація №4:
$p = [0.2607, 0.2145, 0.3992, 0.1249]$, $\mu = [-19.9428, -1.9204, 5.5187, 11.9785]$,
$\sigma = [2.0044, 3.1232, 3.5316, 2.3427]$



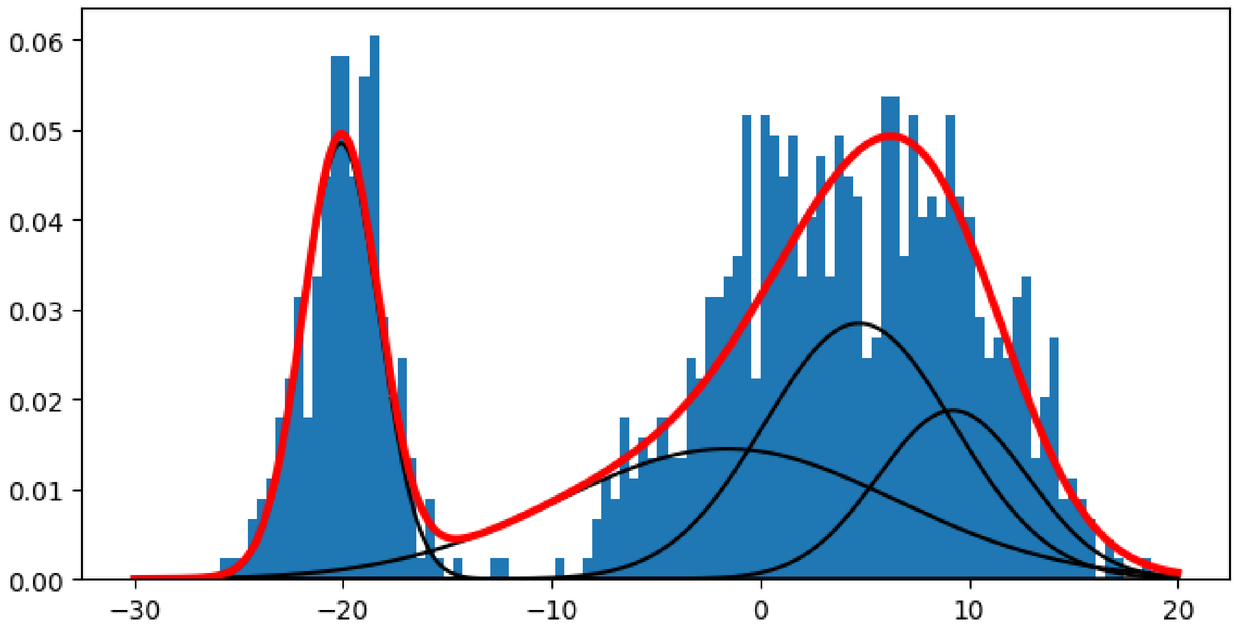
Ітерація №16:
$p = [0.2603, 0.2281, 0.3894, 0.1221]$, $\mu = [-19.954, -1.3782, 5.6342, 11.4019]$,
$\sigma = [1.9826, 3.5079, 4.0118, 2.5686]$



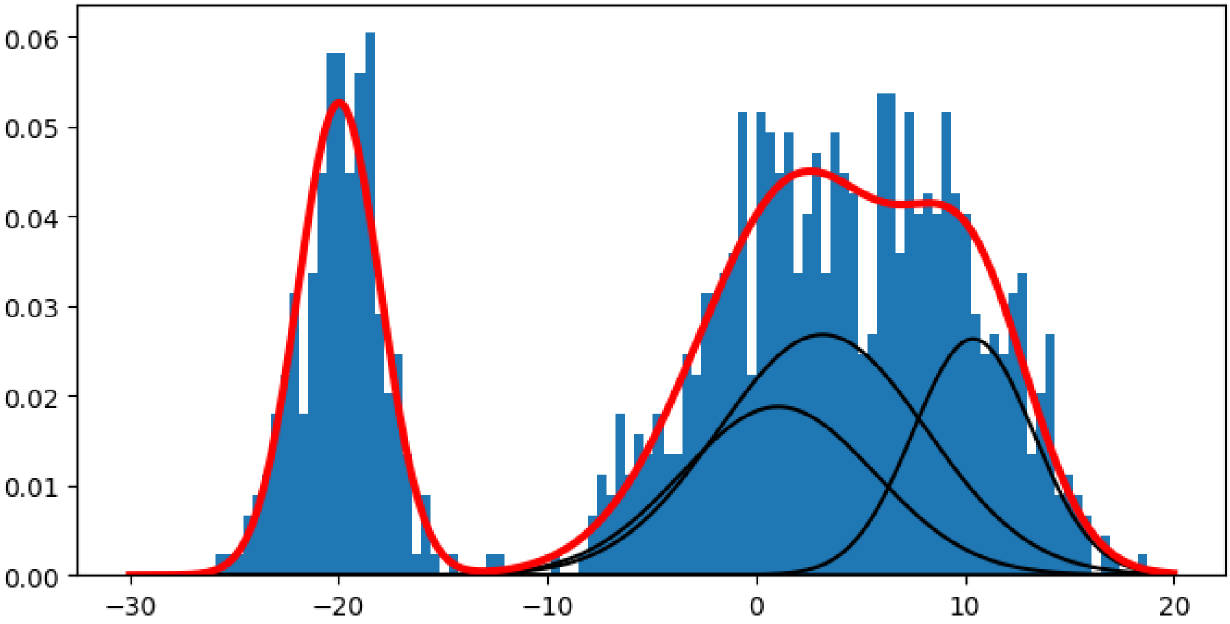
Ітерація №64:
$p = [0.2602, 0.245, 0.3533, 0.1415]$, $\mu = [-19.9562, -0.9495, 5.5667, 10.8491]$,
$\sigma = [1.9787, 3.7211, 4.1995, 2.7564]$

Наступними візьмемо, наприклад, такі початкові величини:

$$\theta^{(0)} = \Big((p_1^{(0)}, p_2^{(0)}, p_3^{(0)}, p_4^{(0)}), (\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}, \mu_4^{(0)}), (\sigma_1^{(0)}, \sigma_2^{(0)}, \sigma_3^{(0)}, \sigma_4^{(0)})\Big)$$
$$= ((0.4,\ 0.2,\ 0.2,\ 0.2),\ (-30, -5,\ 10,\ 10),\ (1,\ 3,\ 3,\ 3))$$

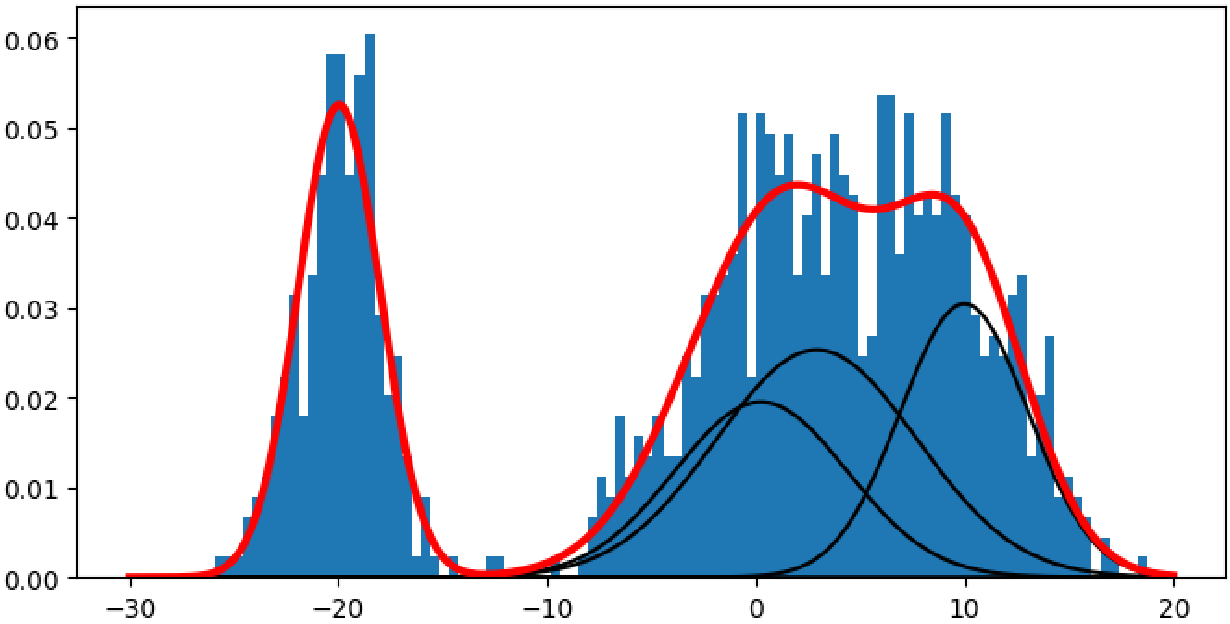


Ітерація №4:
$p = [0.2203, 0.2934, 0.3235, 0.169]$, $\mu = [-20.0842, -1.5914, 4.7291, 9.2192]$,
$\sigma = [1.8132, 8.0791, 4.5351, 3.5958]$



Ітерація №16:

$p = [0.26, 0.2174, 0.3333, 0.1898], \quad \mu = [-19.962, 1.0557, 3.1866, 10.3907],$
$\sigma = [1.9698, 4.6248, 4.9625, 2.8784]$

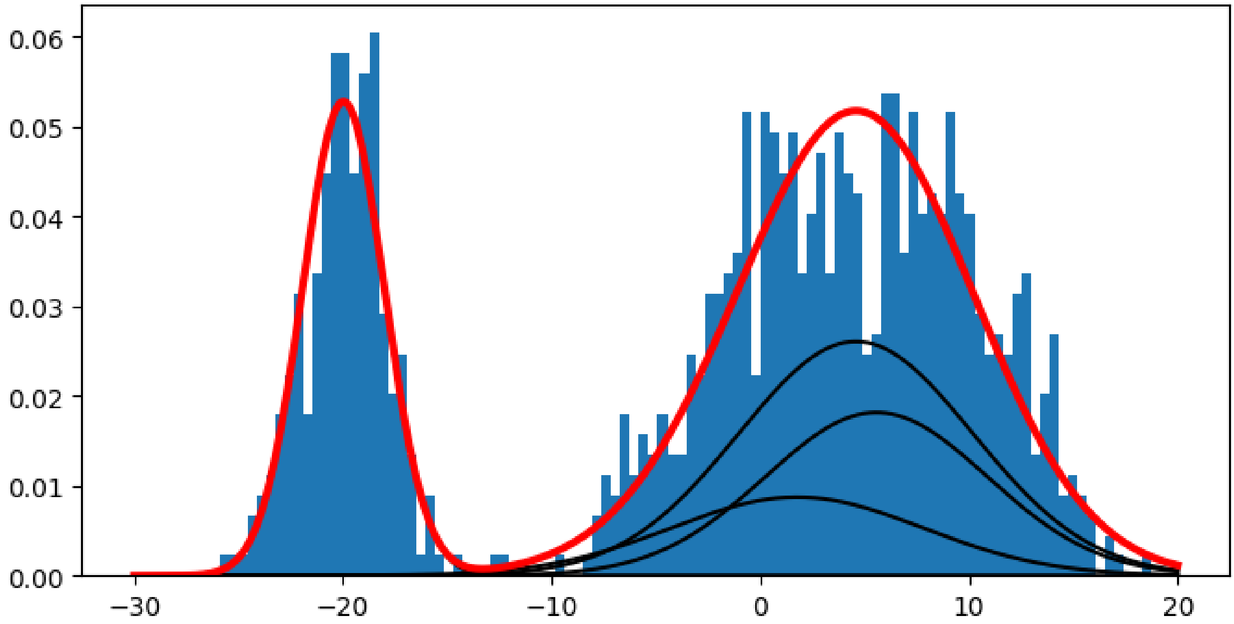


Ітерація №64:

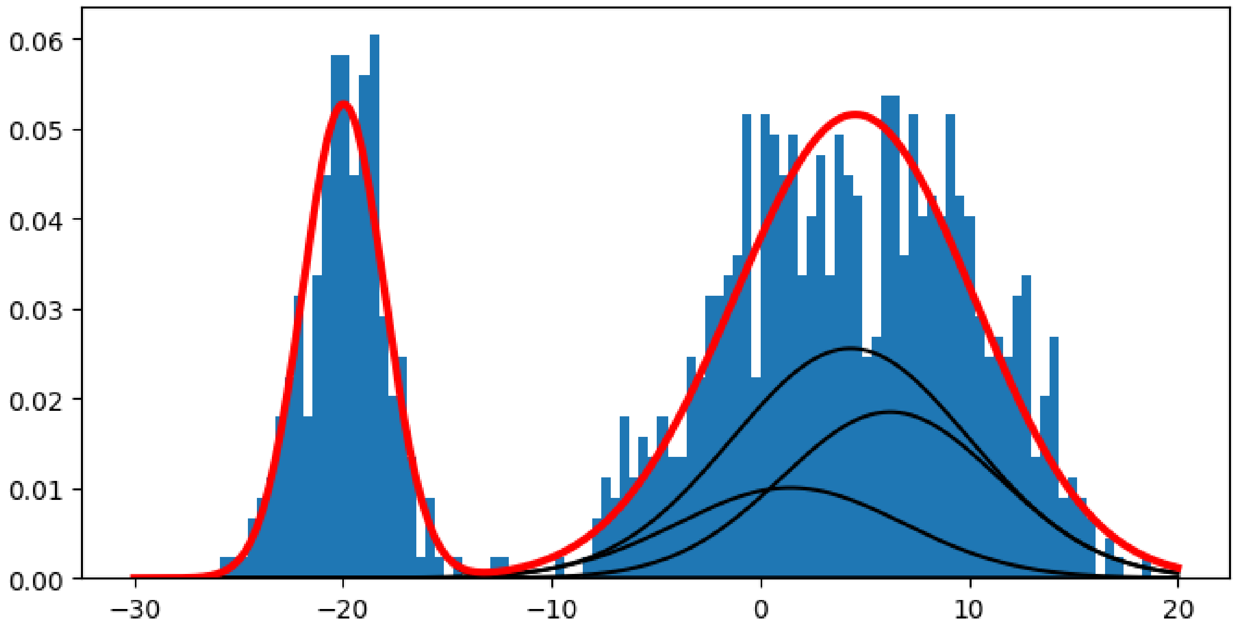
$p = [0.2601, 0.2018, 0.3049, 0.2333], \quad \mu = [-19.9593, 0.2397, 2.9149, 9.9838],$
$\sigma = [1.9736, 4.1279, 4.8099, 3.06]$

І наостанок покладемо $\theta^{(0)}$ таким чином:

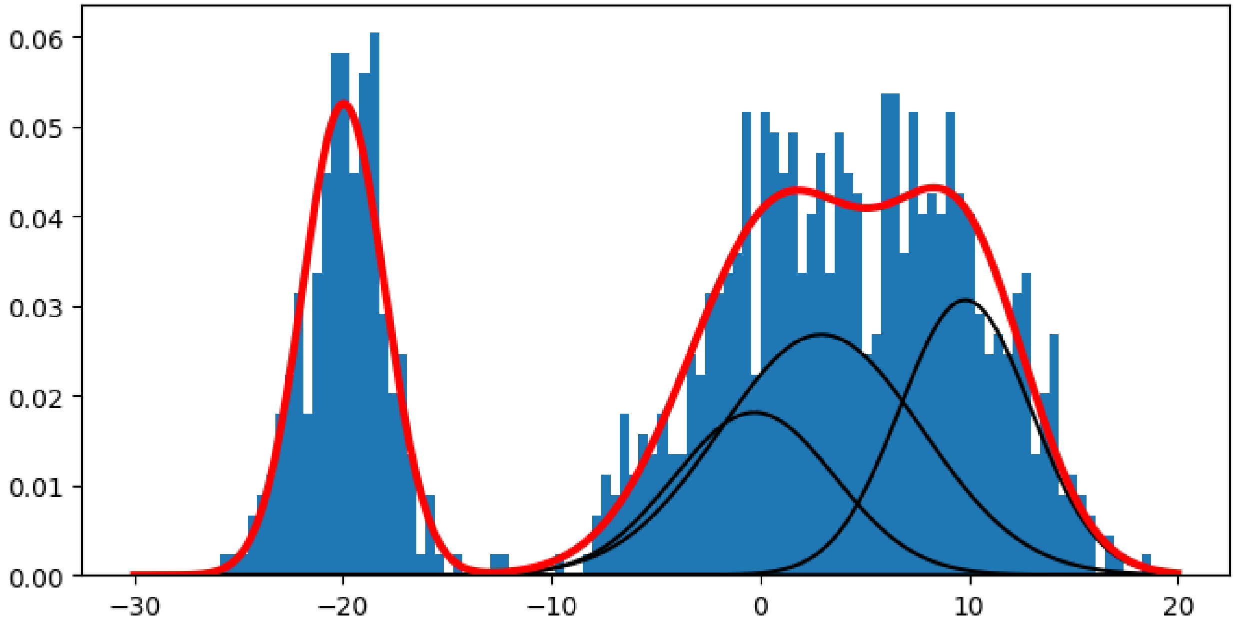
$$\theta^{(0)} = \Big((p_1^{(0)}, p_2^{(0)}, p_3^{(0)}, p_4^{(0)}), (\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}, \mu_4^{(0)}), (\sigma_1^{(0)}, \sigma_2^{(0)}, \sigma_3^{(0)}, \sigma_4^{(0)})\Big)$$
$$= ((0.1,\ 0.5,\ 0.1,\ 0.3),\ (-20,\ 5,\ 5,\ 5),\ (1,\ 4,\ 8,\ 4))$$



Ітерація №4:
$p = [0.2593, 0.3669, 0.1321, 0.2419]$, $\mu = [-19.9715, 4.5798, 1.7643, 5.5634]$, $\sigma = [1.9602, 5.6141, 6.0358, 5.3111]$



Ітерація №16:
$p = [0.2597, 0.3642, 0.136, 0.2401]$, $\mu = [-19.9667, 4.3286, 1.4494, 6.2221]$, $\sigma = [1.9643, 5.6876, 5.4001, 5.1847]$



Ітерація №64:
$$p = [0.2601, 0.3242, 0.1745, 0.2411], \quad \mu = [-19.9584, 2.9267, -0.2773, 9.8184],$$
$$\sigma = [1.9752, 4.8275, 3.8428, 3.1372]$$