Aufgabe 8-1  *EM-Algorithm*

Given a data set with 100 points consisting of three Gaussian clusters $A$, $B$ and $C$ and the point $p$.

The cluster $A$ contains $30\%$ of all objects and is represented using the mean of all his points $\mu_A = (2, 2)$ and the covariance matrix $\Sigma_A = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix}$.

The cluster $B$ contains $20\%$ of all objects and is represented using the mean of all his points $\mu_B = (5, 3)$ and the covariance matrix $\Sigma_B = \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix}$.

The cluster $C$ contains $50\%$ of all objects and is represented using the mean of all his points $\mu_C = (1, 4)$ and the covariance matrix $\Sigma_C = \begin{pmatrix} 16 & 0 \\ 0 & 4 \end{pmatrix}$.

The point $p$ is given by the coordinates $(2.5, 3.0)$.

Compute the three probabilities of $p$ belonging to the clusters $A$, $B$ and $C$.

The following sketch is not exact, and only gives a rough idea of the cluster locations:

![Cluster Diagram](image)

Aufgabe 8-2  *Multivariate Density and Mahalanobis Distance*

The density of the multivariate normal distribution (with $\Sigma$, $\mu$) is computed using the formula

$$
\text{prob}(p, \mu, \Sigma) := \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} \cdot e^{-\frac{1}{2}((p-\mu)^T\Sigma^{-1}(p-\mu))}
$$

Find and discuss the relationship of the formula to the Mahalanobis distance (using $\Sigma$) of $p$ to $\mu$.

$$
\text{d}_{\text{Mahalanobis}}(x, y, \Sigma) := \sqrt{(x - y)^T\Sigma^{-1}(x - y)}
$$
As distance function, use Manhattan distance $L_1(a, b) := |a_1 - b_1| + |a_2 - b_2|$.

Construct an OPTICS reachability plot (see pseudo-code below) for each of the following parameter settings:

- $\varepsilon = 5$ and $\text{minPts} = 2$
- $\varepsilon = 5$ and $\text{minPts} = 4$
- $\varepsilon = 2$ and $\text{minPts} = 4$
- $\varepsilon = \infty$ and $\text{minPts} = 4$

**Pseudocode OPTICS**

```plaintext
seedlist = ∅  // implemented as a heap
for i = 0 to n-1 do
  if(seedlist = ∅ ) then seedlist = {(random_not_handled_point, ∞)}
  (x, x.reach) = get_and_remove_point_with_min_reach(seedlist)
  x.pos = i
  x.handled = TRUE
  neighbors = rangeQuery(x, ε)
  x.core = nnDist(x, neighbors, MinPts)
  if(x.core < ∞)
    for each y ∈ neighbors with not(y.handled)
      if( y ∉ seedlist ) seedlist = seedlist ∪ {(y, reach-dist(y,x))}
      else
        curr_reach = lookup(seedlist, y)
        update(y, min(curr_reach, reach-dist(y,x)))
  endfor
endfor
```