Outlier Detection
Using Convex Hull

볼록 테두리를 이용한
특이값 판별

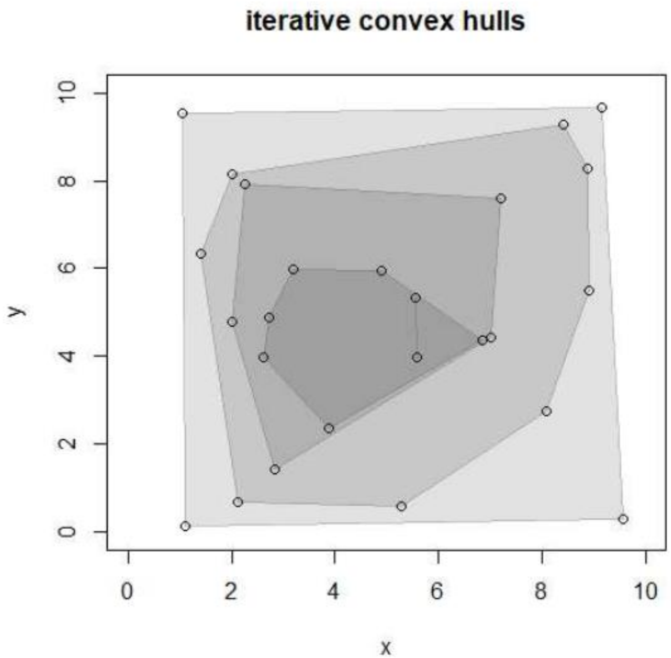
15 허은정

Outlier Detection
Using Convex Hull

과제 4-5의 볼록 테두리(Convex Hull)

5. 아래는 25개 점에 대한 볼록 테두리(convex hull)와 이어서 볼록 테두리의 내부 점들에 대하여 같은 작업이 거듭 반복된 그림이다. 이를 위한 R 스크립트를 제시하라.

  * 단, set.seed(k)에서 k=5 대신 개인 학번의 마지막 두 숫자를 쓸 것.



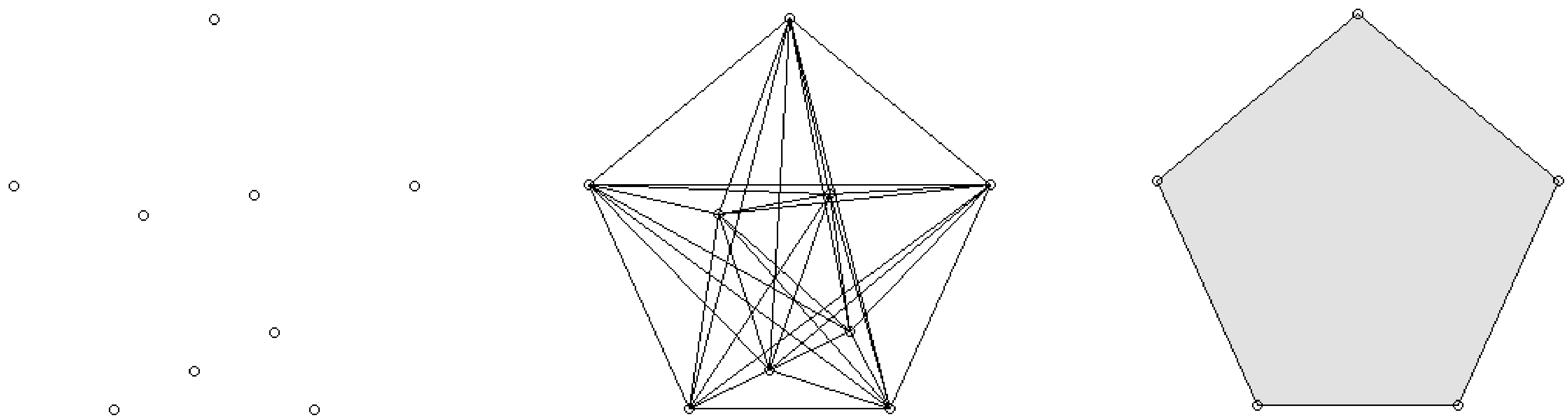

```
set.seed(5)
plot(x=c(0,10),y=c(0,10),type="n",xlab="x",ylab="y",
     main="iterative convex hulls")
n <- 25
df <- matrix(runif(2*n,0,10),n,2)
points(df)
pts <- chull(df)
polygon(df[pts,],col="#22222222",border="#55555555")
while(n > 1) {
    ________________________________
    ________________________________
    ________________________________
    ________________________________
}
```

* 계산방법_2019_과제 4, 5p

Convex Hull이란?

A set of points is defined to be convex
if it contains the line segments connecting each pair of its points.

* https://en.wikipedia.org/wiki/Convex_hull, Definitions 항목



목차

Convex Hull 구현

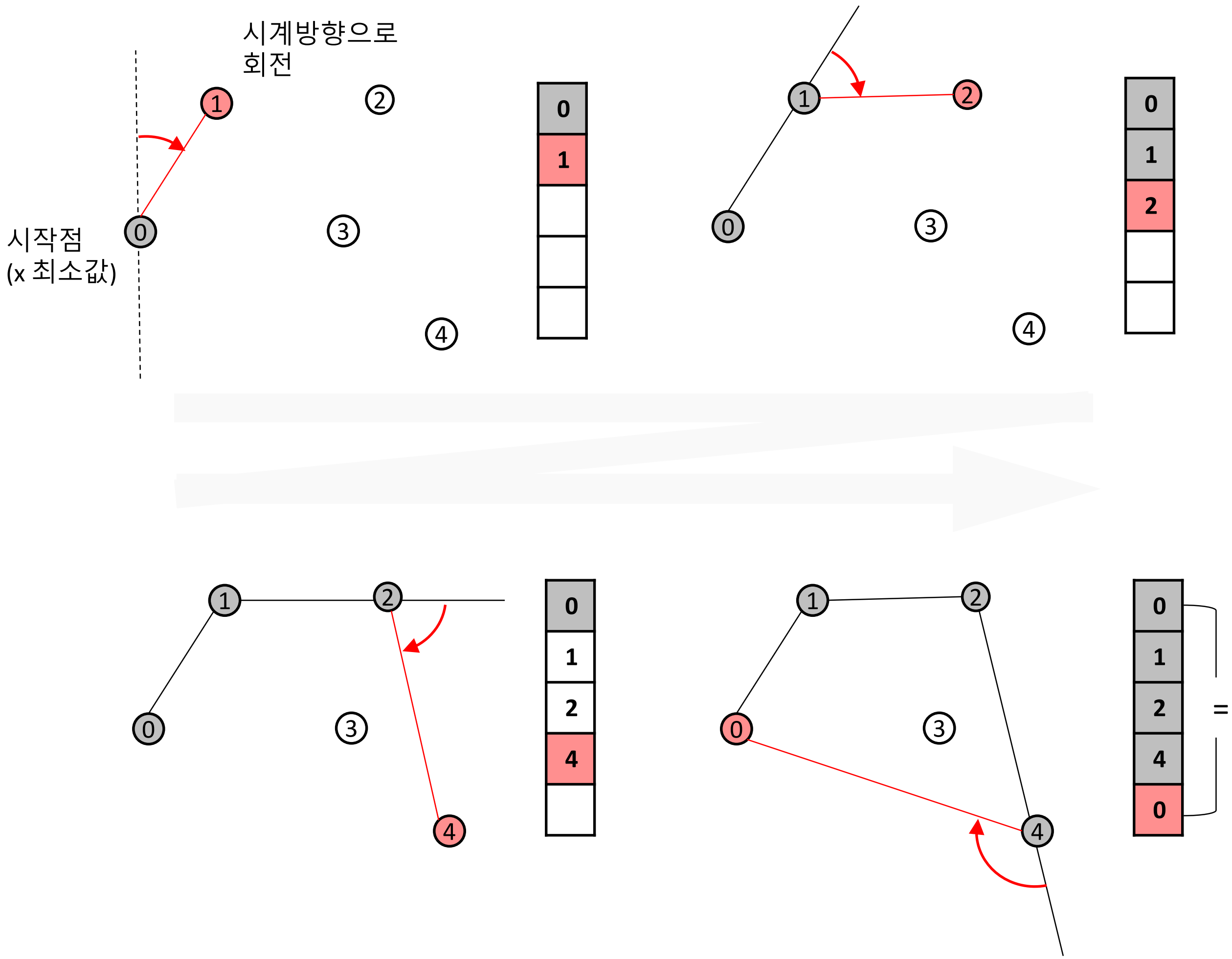
Algorithms [edit]

Known convex hull algorithms are listed below, ordered by the date of first publication. Time complexity of each algorithm is stated in terms of the number of inputs points n and the number of points on the hull h. Note that in the worst case h may be as large as n.

- **Gift wrapping**, a.k.a. **Jarvis march** — $O(nh)$
  One of the simplest (although not the most time efficient in the worst case) planar algorithms. Created independently by Chand & Kapur in 1970 and R. A. Jarvis in 1973. It has $O(nh)$ time complexity, where n is the number of points in the set, and h is the number of points in the hull. In the worst case the complexity is $\Theta(n^2)$.
- **Graham scan** — $O(n \log n)$
  A slightly more sophisticated, but much more efficient algorithm, published by Ronald Graham in 1972. If the points are already sorted by one of the coordinates or by the angle to a fixed vector, then the algorithm takes $O(n)$ time.
- **Quickhull**
  Created independently in 1977 by W. Eddy and in 1978 by A. Bykat. Just like the quicksort algorithm, it has the expected time complexity of $O(n \log n)$, but may degenerate to $O(n^2)$ in the worst case.
- **Divide and conquer** — $O(n \log n)$
  Another $O(n \log n)$ algorithm, published in 1977 by Preparata and Hong. This algorithm is also applicable to the three dimensional case.
- **Monotone chain**, a.k.a. **Andrew's algorithm**— $O(n \log n)$
  Published in 1979 by A. M. Andrew. The algorithm can be seen as a variant of Graham scan which sorts the points lexicographically by their coordinates. When the input is already sorted, the algorithm takes $O(n)$ time.
- **Incremental convex hull algorithm** — $O(n \log n)$
  Published in 1984 by Michael Kallay.
- **The ultimate planar convex hull algorithm** — $O(n \log h)$
  The first optimal output-sensitive algorithm, it uses the technique of marriage-before-conquest. Published by Kirkpatrick and Seidel in 1986.
- **Chan's algorithm** — $O(n \log h)$
  A simpler optimal output-sensitive algorithm created by Chan in 1996.

* https://en.wikipedia.org/wiki/Convex_hull_algorithms, Algorithms 항목

Convex Hull 구현 - Gift Wrapping 알고리즘



Convex Hull
점의 개수가 h이면
다음 점을 찾는 n번의
계산이 h회 반복되므로
계산복잡도는
O(nh) ~ O(n^2)

Convex Hull 구현 - R의 chull() 함수

```
> chull
function (x, y = NULL)
{
    X <- xy.coords(x, y, recycle = TRUE)
    x <- cbind(X$x, X$y)
    if (any(!is.finite(x)))
        stop("finite coordinates are needed")
    if (nrow(x) == 0)
        return(integer())
    if (nrow(x) == 1)
        return(1L)
    res <- .Call(C_chull, x) ?
    if (length(res) < 2L)
        return(res)
    xx <- sweep(x[res, ], 2L, colMeans(x[res, ]))
    angs <- atan2(xx[, 2L], -xx[, 1L])
    res[order(angs)]
}
<bytecode: 0x0000000009c6f338>
<environment: namespace:grDevices>
```

Details

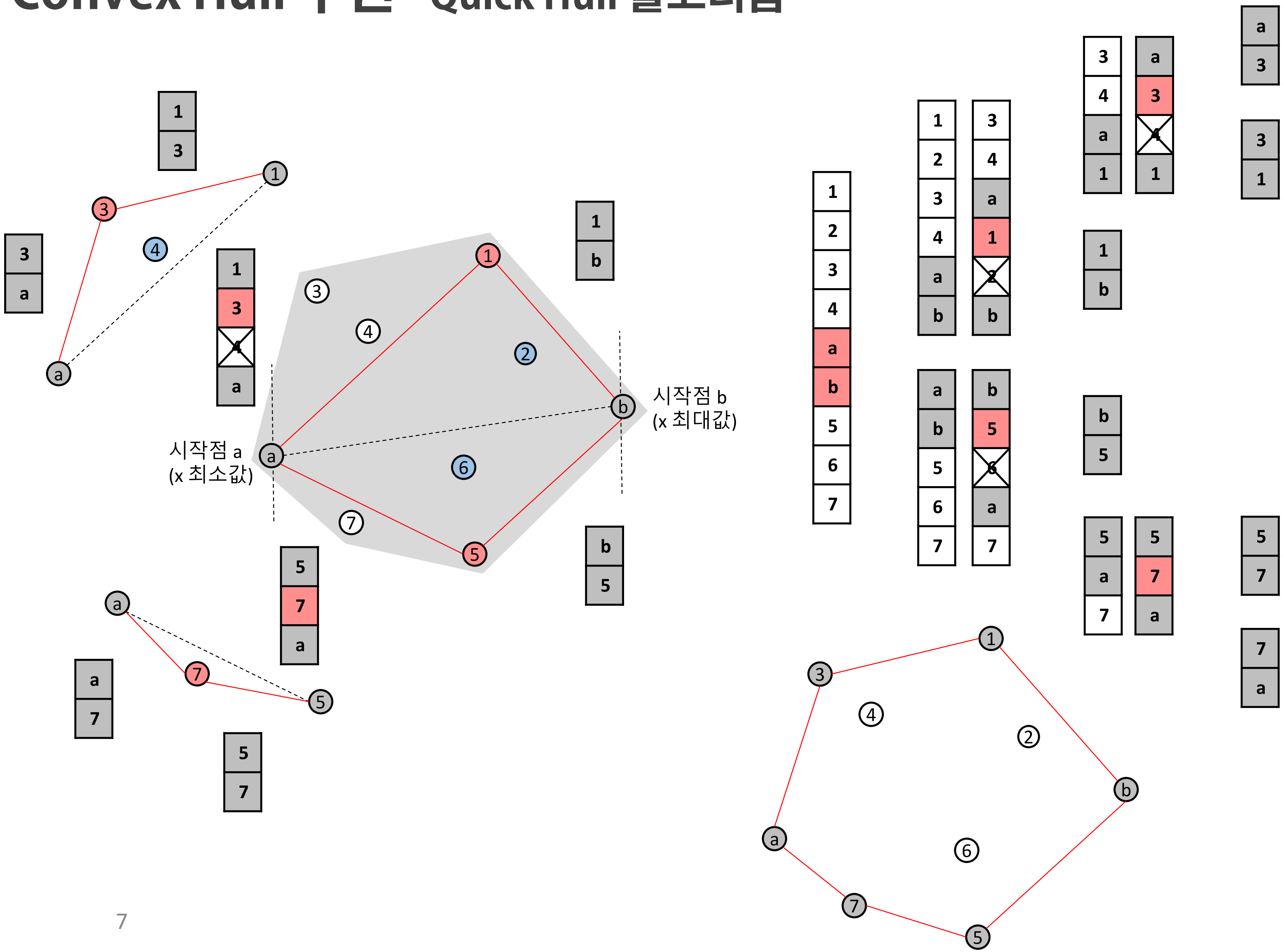
The algorithm is that given by Eddy (1977).

References

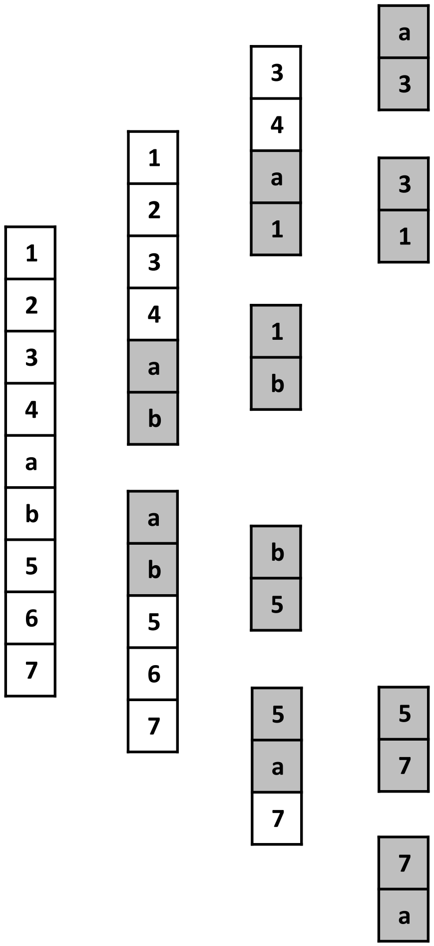
Eddy, W. F. (1977) A new convex hull algorithm for planar sets. *ACM Transactions on Mathematical Software*, **3**, 398–403.

Eddy, W. F. (1977) Algorithm 523. CONVEX, A new convex hull algorithm for planar sets[Z]. *ACM Transactions on Mathematical Software*, **3**, 411–412.

Convex Hull 구현 - Quick Hull 알고리즘



Convex Hull 구현 - Quick Hull 알고리즘



빠른 정렬: quick sort

길이 n의 수치 벡터를 크기 순서로 정렬하자.

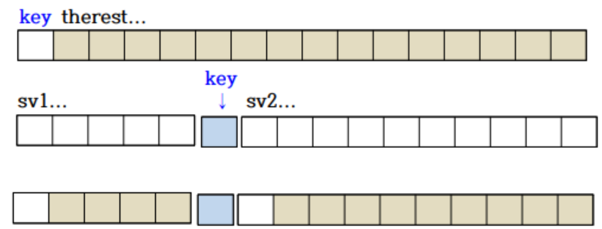
1) 한 값(key)과 비교하여 나머지들(therest)을 왼편(sv1) 또는 오른편(sv)에 붙인다.
2) sv1과 sv2 각각에 그런 과정을 반복

R 기법: recursive calls of a function

개념도:

첫 단계.

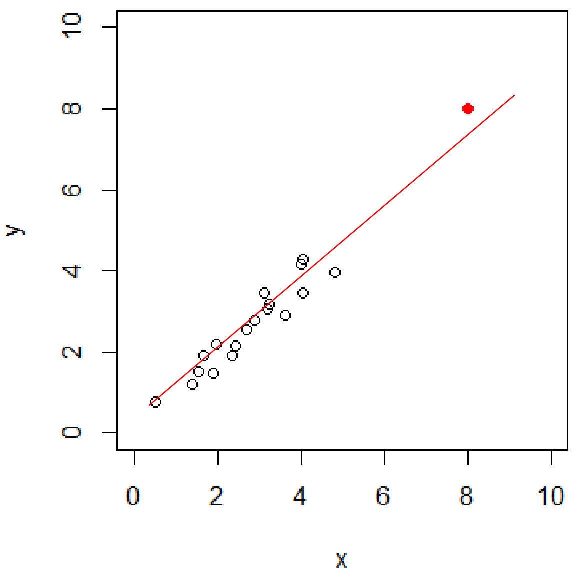
다음 단계.

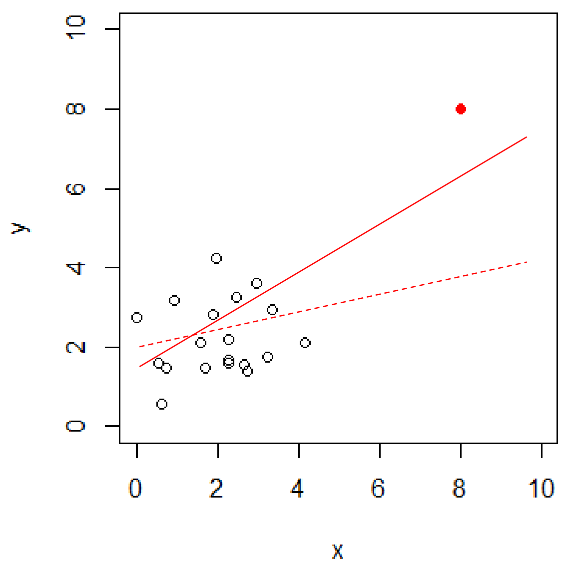


빠른 정렬 quick sort: $C_3(n) \propto n \cdot \log n$

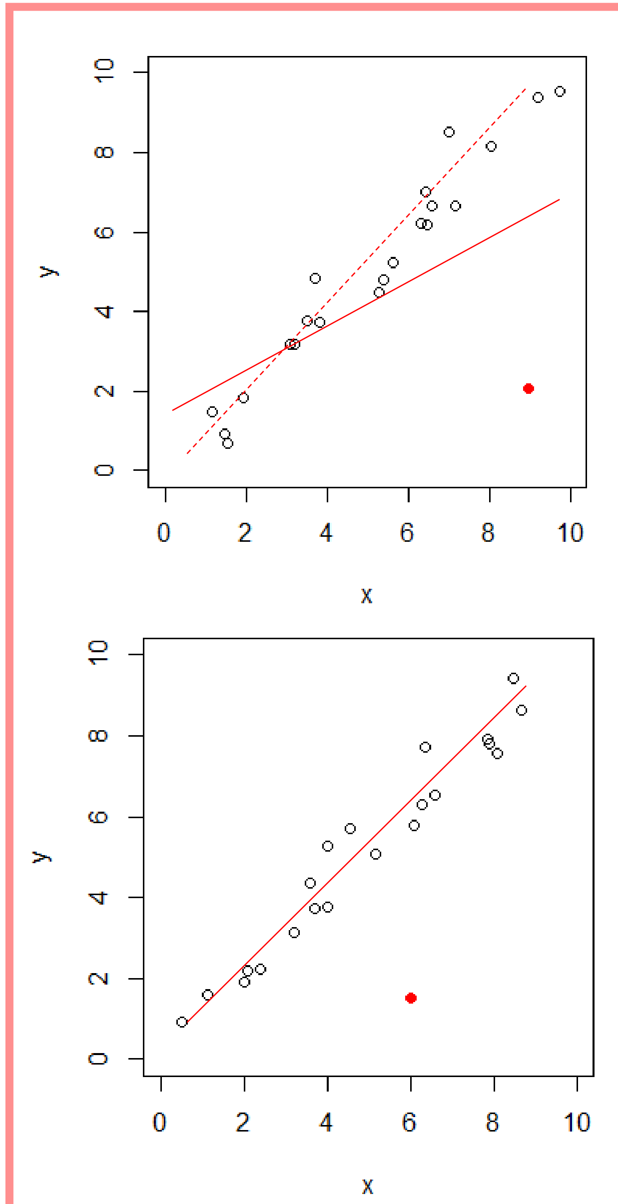
* 02_수열 PT_2019, 22-23p

이변량 분포 파악을 해치는 Outlier

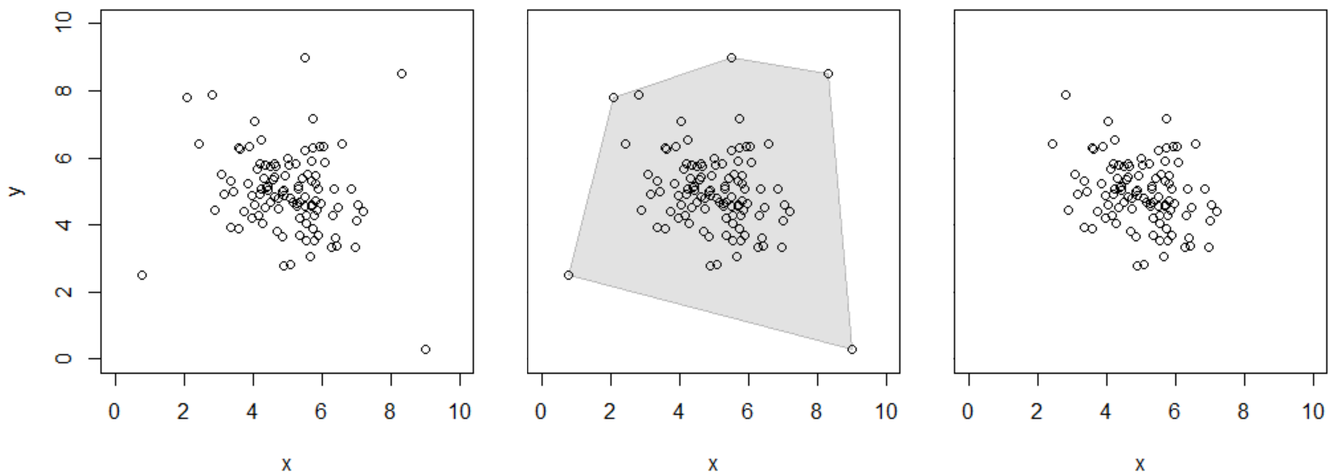




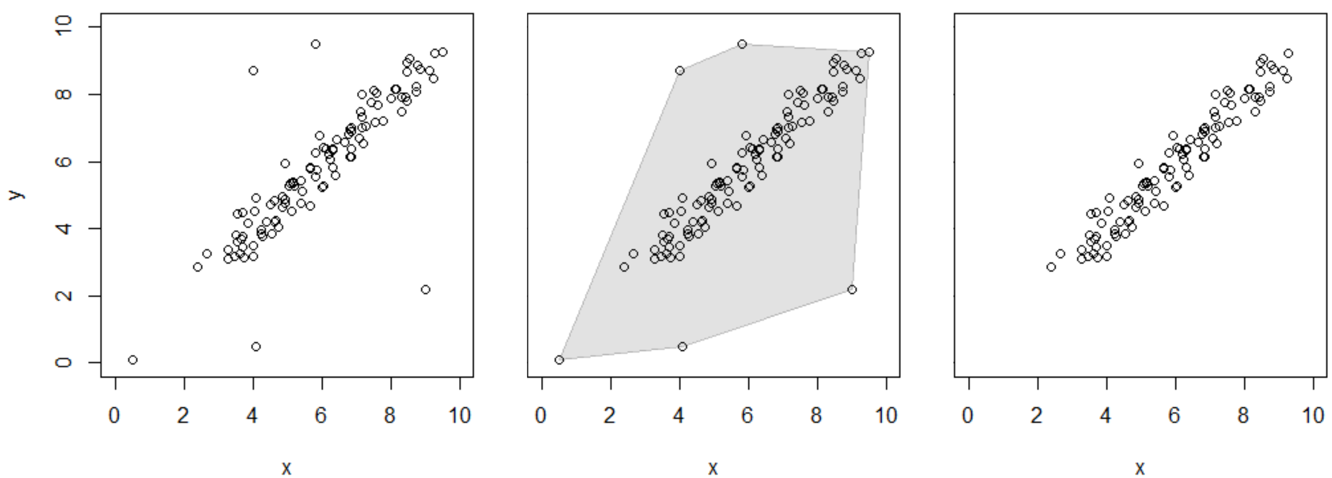
Outlier는 관측값의 주 분포를 벗어난 특이값이다.
Leverage Point는 일변량 자료를 다루듯 극단값을 제거하면 되지만
이변량 혹은 다변량 분포에서는 각 변수의 값이
극단값이 아니더라도 분포에서 벗어난 경우가 있어
시각화를 통해 직관적으로 Outlier를 찾아내거나
특수한 공식으로 Outlier를 찾아내야 한다.



이변량 분포에의 적용 - Outlier가 적절하게 제거된 경우

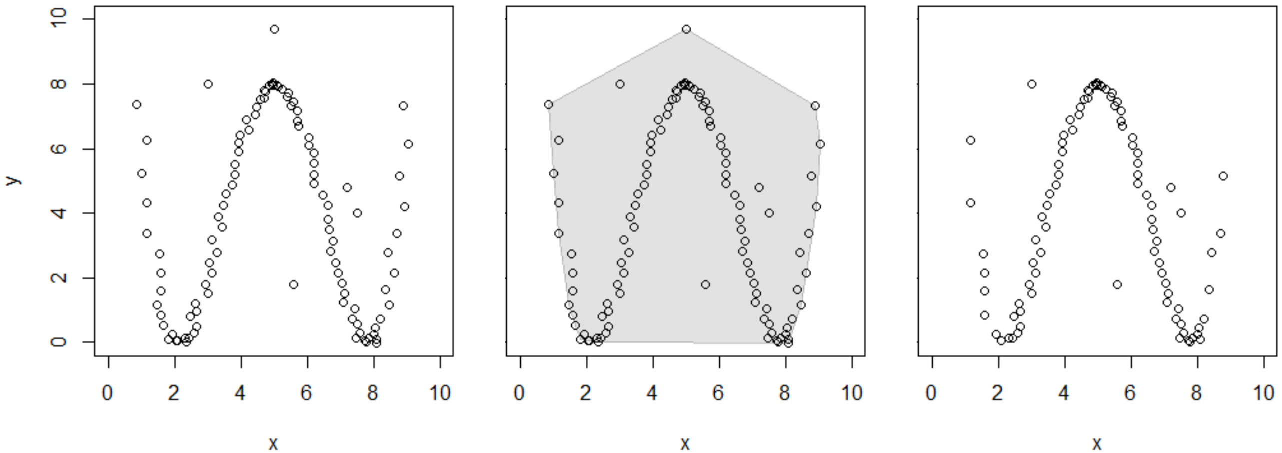


극단값이 여러 개 있는 경우

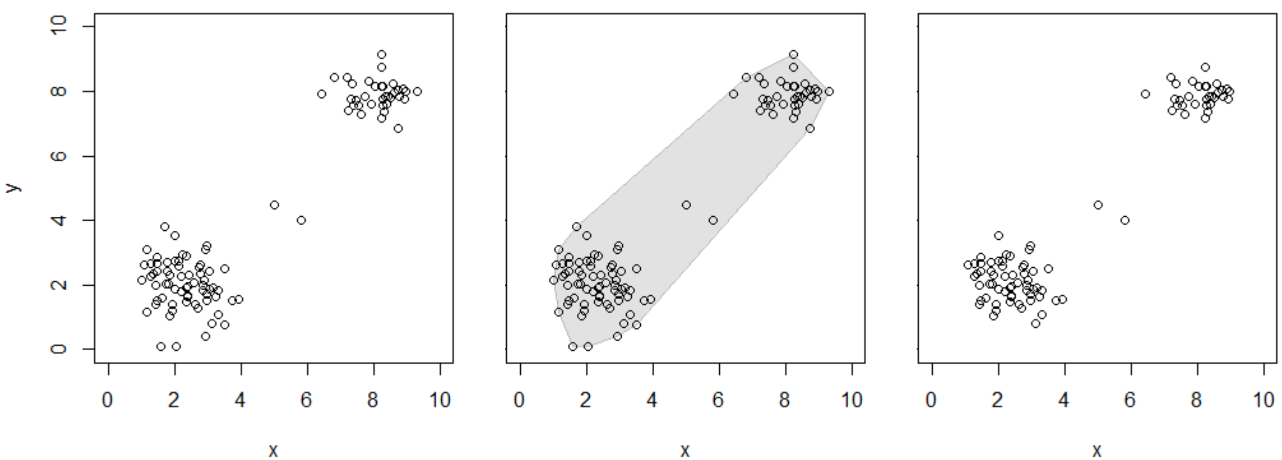


극단값은 물론 특이값이 여러 개 있는 경우

이변량 분포에의 적용 - Outlier가 적절하게 제거되지 못한 경우

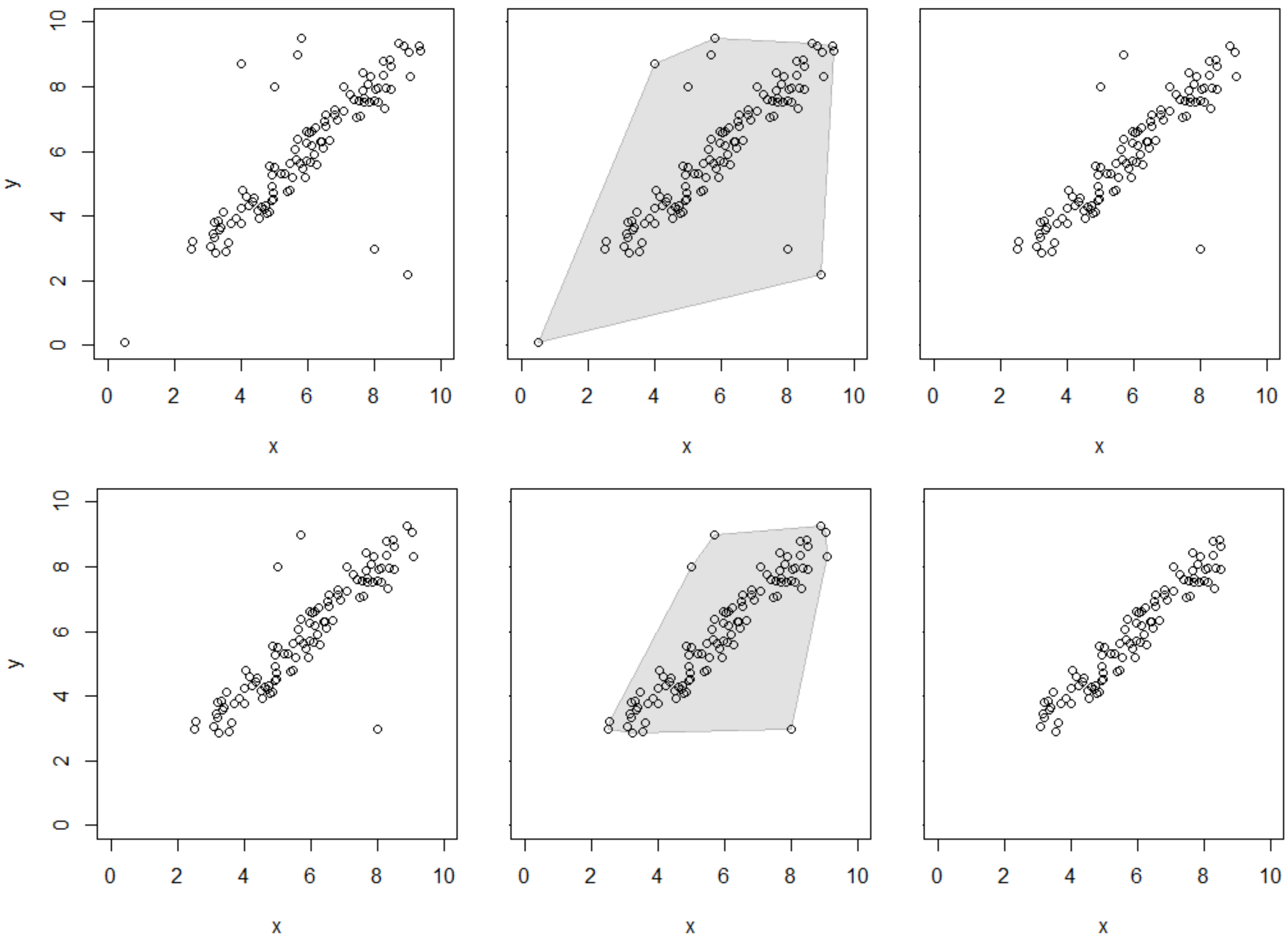


곡선 형태인 경우
Outlier는
제거되지 않고
가장자리의
유효한 값이 제거됨



군집이 있는 경우
Outlier는
제거되지 않고
가장자리의
유효한 값이 제거됨

이변량 분포에의 적용 - Outlier가 적절하게 제거되지 못한 경우



비슷한 위치의
Outlier가 여러 개면
한 번에 모두
제거되지 않음

Outlier 제거 작업의
반복으로
가장자리의 유효한
값이 많이 제거됨

이변량 분포에의 적용 - 결과

Convex Hull을 이용하여 Outlier를 제거했을 때

장점

극단값, Leverage point, Outlier를

별도의 가중치 계산 없이 직관적으로 동시에 제거할 수 있다

단점

변수들의 분포 형태에 의존하고 적절하게 Outlier가 제거되는 경우가 제한적

-> *두 변수가 선형성을 띄거나 한 곳에 뭉쳐있는 경우*

(평균이나 중위수로 분포를 나타내는 것이 적합한 경우에 이 방법 또한 적합해 보인다.)

정상 분포 내에 있는 유효한 값들도 함께 제거됨

Outlier가 뭉쳐있어 Convex Hull을 여러 번 만들어야 하는 경우 유효값의 유실이 매우 많아짐

-> *데이터 개수가 적다면 사용해서는 안 될 것*

결론: 이변량 분포에서 찾기 힘들었던 Outlier를 극단값과 동시에 제거할 수 있다.
다만 유효한 가장자리 값이 유실되기에 모델을 세운다면 가장자리에서 부정확할 수 있다.
엄밀히 말하면 Outlier와 Outlier가 아닌 값을 판별하는 것은 실패!

더 생각해볼 것

다변량 자료에서 장점이 극대화되지 않을까?

- 이변량 분포는 사람의 눈으로 확인이 가능하기에
  굳이 Convex Hull로 Outlier 제거할 필요 없음
  산포도를 보고 바로 파악 가능

Convex Hull로 생성한 Polygon의 넓이를 구해 활용할 수 있지 않을까?

- Convex Hull에 해당하는 값 제거 전과
  제거 후 새로 만든 Polygon의 넓이를 비교했을 때 차이가 크다면
  극단값, Leverage point, Outlier가 잘 제거되었다는 증거가 아닐까?

Convex Hull을 이용해 Outlier를 찾아낸 사례, 논문

Removing Outliers to Minimize Area and Perimeter, Rossen Atanassov 외 (2006)

Approximation Algorithms for Outlier Removal in Convex Hulls, Michael Biro 외 (2013)

Use of convex hull for detection of outliers in oceanographic data pertaining to Indian ocean, Murala Krishna 외 (2016)

Onion-Peeling Outlier Detection in 2-D data Sets, Archit Harsh 외 (2016)

Convex hull peeling을 이용한 eigen vector 추정, 최지수 (2018)

참고문헌

1p) 계산방법_2019_과제 4, 5p, 허명회

2p) https://en.wikipedia.org/wiki/Convex_hull, Definitions 항목

4p) https://en.wikipedia.org/wiki/Convex_hull_algorithms, Algorithms 항목

5p) Jarvis, R. A. (1973), On the identification of the convex hull of a finite set of points in the plane.

6p) https://www.rdocumentation.org/packages/grDevices/versions/3.6.0/topics/chull, chull() 도움말

7p) William F. Eddy. (1977), A New Convex Hull Algorithm for Planar Sets.

8p) 02_수열 PT_2019, 22-23p, 허명회

기타 참고 자료)

Regression Analysis by Example, Fifth Edition by Samprit Chatterjee, Ali S. Hadi

 - chapter4, Regression Diagnostics : Detection of Model Violations

https://en.wikipedia.org/wiki/Gift_wrapping_algorithm

https://leadbiz.tistory.com/683

https://en.wikipedia.org/wiki/Quickhull

https://iq.opengenus.org/quick-hull-convex-hull