Density Estimation

Goal: We are interested in estimation of a density function f using observations of random variables $Y_1, \ldots, Y_n$ sampled independently from f.

Parametric Solution:

We will focus on **nonparametric** approaches to density estimation.

1 Histograms

One familiar density estimator is a histogram. Histograms are produced automatically by most software packages and are used so routinely to visualize densities that we rarely talk about their underlying complexity.

1.1 Motivation

Recall the definition of a density function

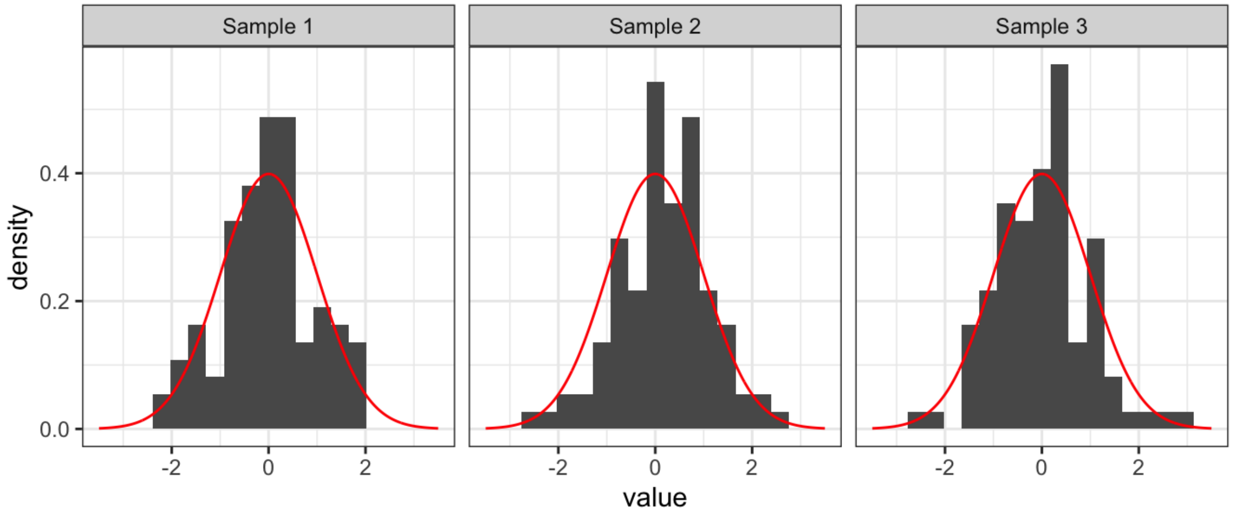
$$f(y) \equiv \frac{d}{dy}F(y) \equiv \lim_{h\to 0}\frac{F(y+h)-F(y-h)}{2h} = \lim_{h\to 0}\frac{F(y+h)-F(y)}{h},$$

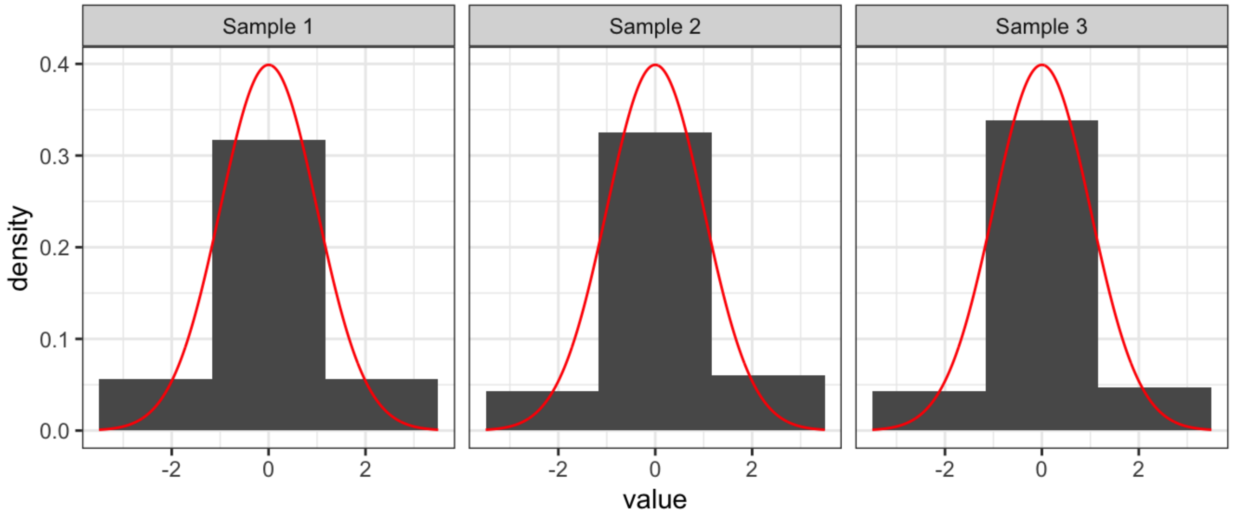
where $F(x)$ is the cdf of the random variable Y.

Now, let $Y_1, \dots, Y_n$ be a random sample of size n from the density f.

A natural finite-sample analog of $f(y)$ is to divide the support of Y into a set of K equi-sized bins with small width h and replace $F(x)$ with the empirical cdf.

1.2 Bin Width


Sample 1
Sample 2
Sample 3
density
0.0
0.2
0.4
-2
0
2
value


Sample 1
Sample 2
Sample 3
density
0.0
0.1
0.2
0.3
0.4
-2
0
2
value

1.3 Measures of Performance

Squared Error

Mean Squared Error

Integrated Squared Error

Mean Integrated Squared Error

1.4 Optimal Binwidth

We will investigate bias and variance of $\hat{f}$ pointwise, because
$\mathrm{MSE}(y) = (\mathrm{bias}(\hat{f}(y))^2 + \mathrm{Var}\hat{f}(y)$.

The roughness of the underlying density, as measured by $R(f')$ determines the optimal level of smoothing and the accuracy of the histogram estimate.

We cannot find the optimal binwidth without known the density f itself.

Simple (plug-in) approach: Assume f is a $N(\mu, \sigma^2)$, then

Data driven approach:

2 Frequency Polygon

The histogram is simple, useful and piecewise constant.

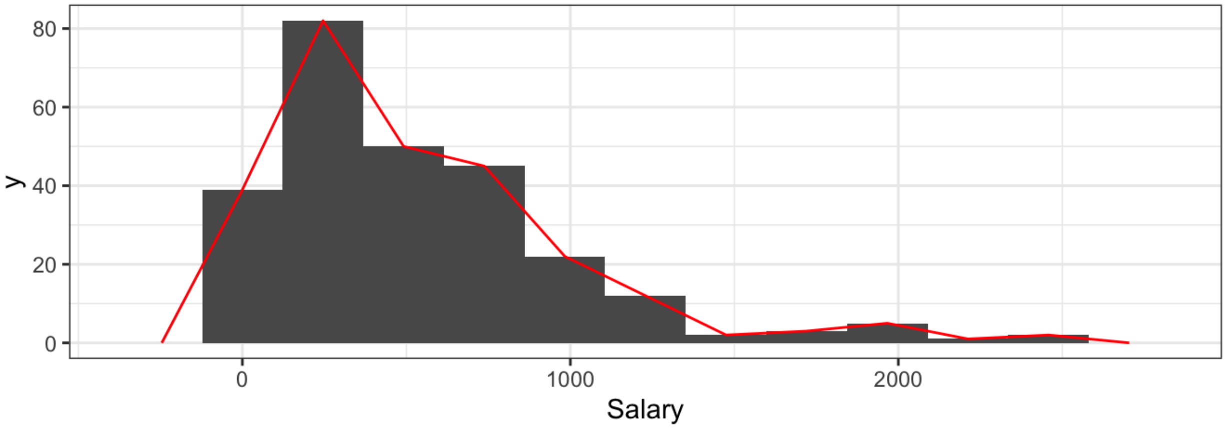
```
library(ISLR)

# optimal h based on normal method
h_0 <- 3.491 * sd(Hitters$Salary, na.rm = TRUE) *
          sum(!is.na(Hitters$Salary))^(-1/3)

## original histogram with optimal h
ggplot(Hitters) +
  geom_histogram(aes(Salary), binwidth = h_0) -> p

## get values to build freq polygon
vals <- ggplot_build(p)$data[[1]]
poly_dat <- data.frame(x = c(vals$x[1] - h_0,
                             vals$x, vals$x[nrow(vals)] + h_0),
                       y = c(0, vals$y, 0))

## plot freq polygon
p + geom_line(aes(x, y), data = poly_dat, colour = "red")
```



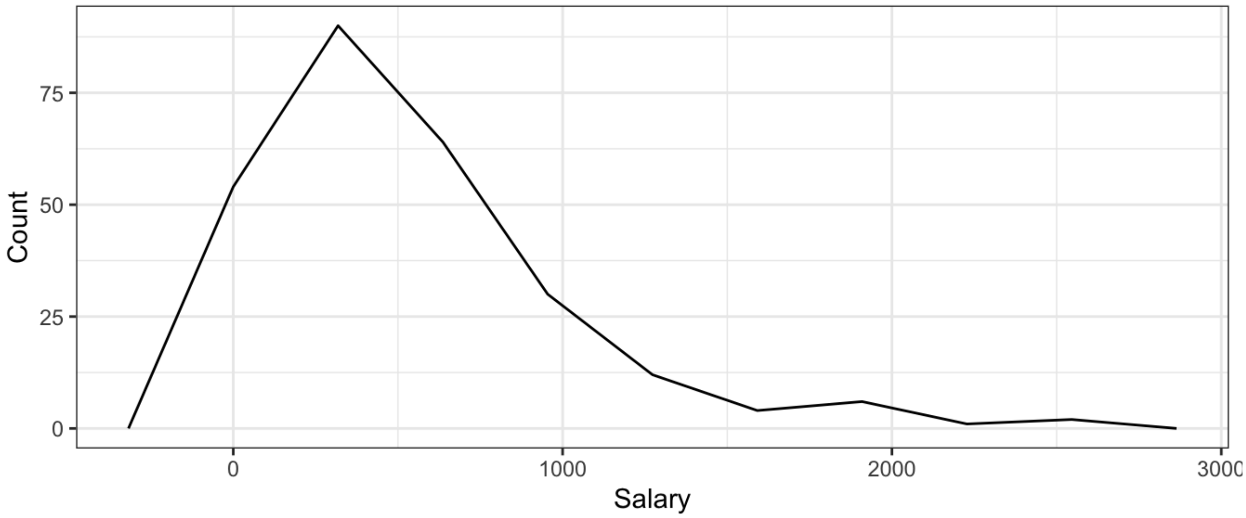
Let $b_1, \ldots, b_{K+1}$ represent bin edges of bins with width h and $n_1, \ldots, n_K$ be the number of observations falling into the bins. Let $c_0, \ldots, c_{k+1}$ be the midpoints of the bin interval.

The frequency polygon is defined as

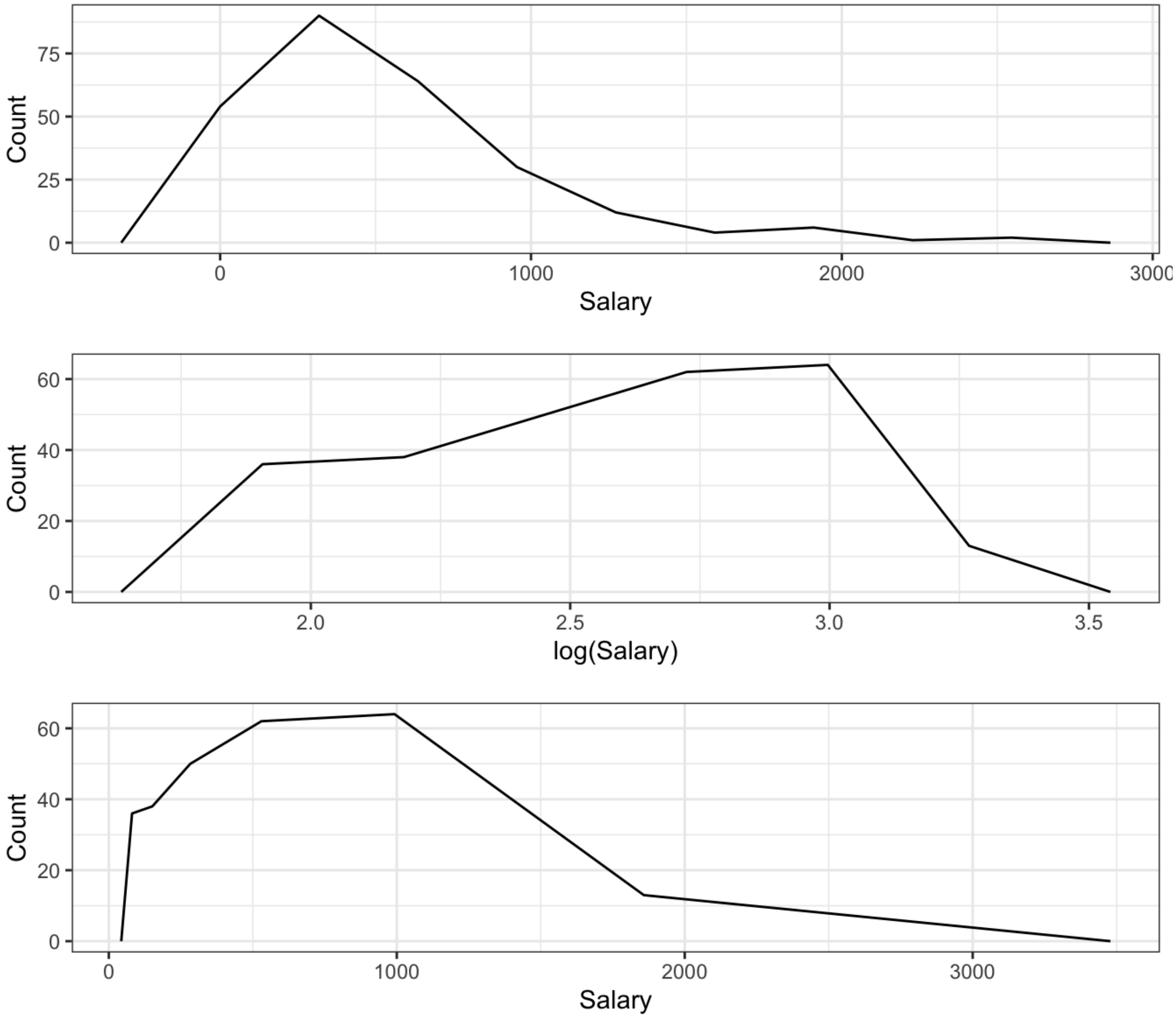
MISE

AMISE

Gaussian rule for binwidth


Count
75
50
25
0
0
1000
2000
3000
Salary

In practice, a simple way to construct locally varying binwidth histograms is by transforming the data to a different scale and then smoothing the transformed data. The final estimate is formed by simply transforming the constructed bin edges $\{b_j\}$ back to the original scale.


Count
Salary
Count
log(Salary)
Count
Salary