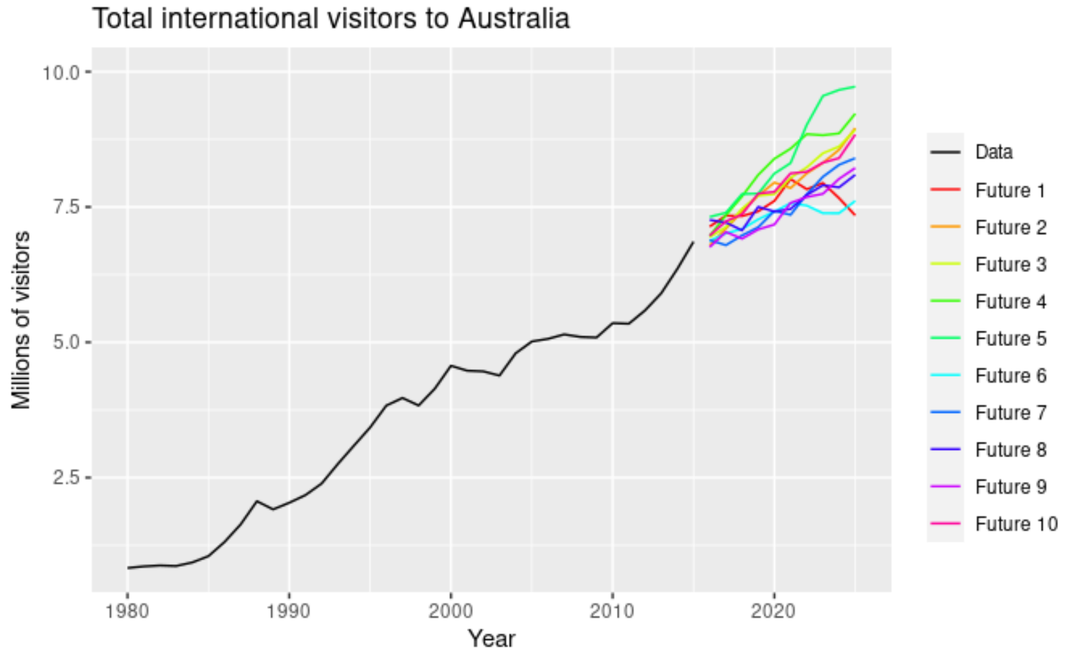
Statistical Techniques for Data Science & Robotics

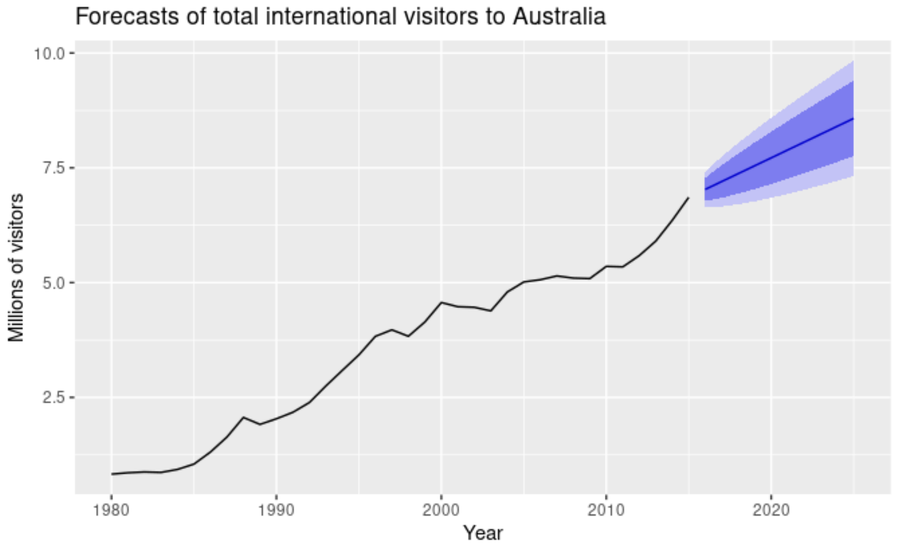
Week 8

Bootstrap in Time series

Time series. Forecasting and residuals

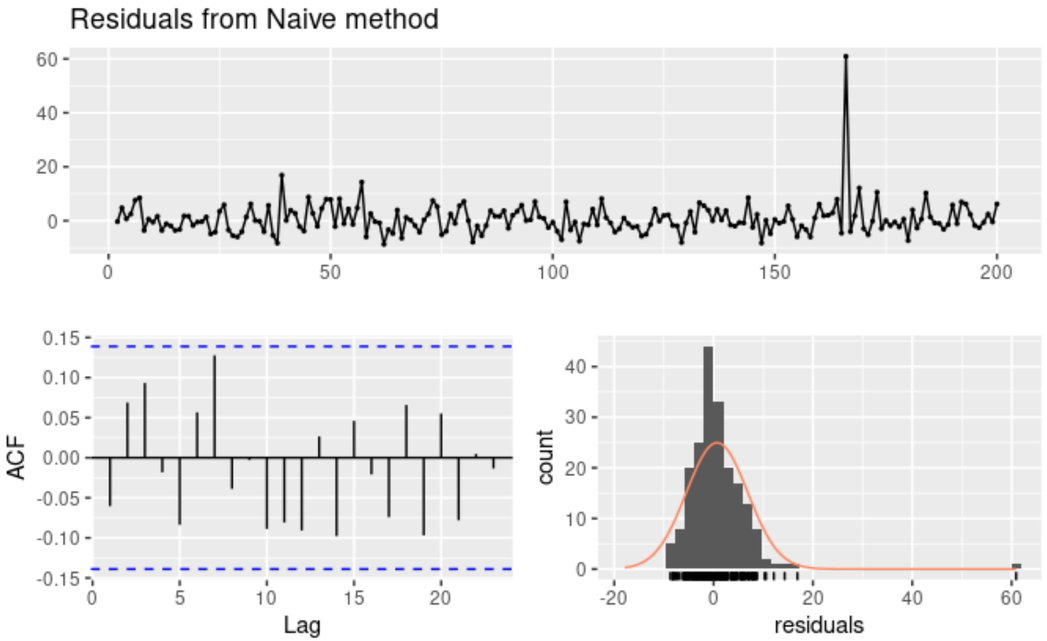
- Series $y_1, y_2, \ldots, y_t, \ldots, y_n$
- When we talk about the “forecast” we usually mean the average value of the forecast distribution
- We put a “hat”
  - $\hat{y}$ is a forecasted value
- If we take into account all previous observations, we write $\hat{y}_{t|t-1}$

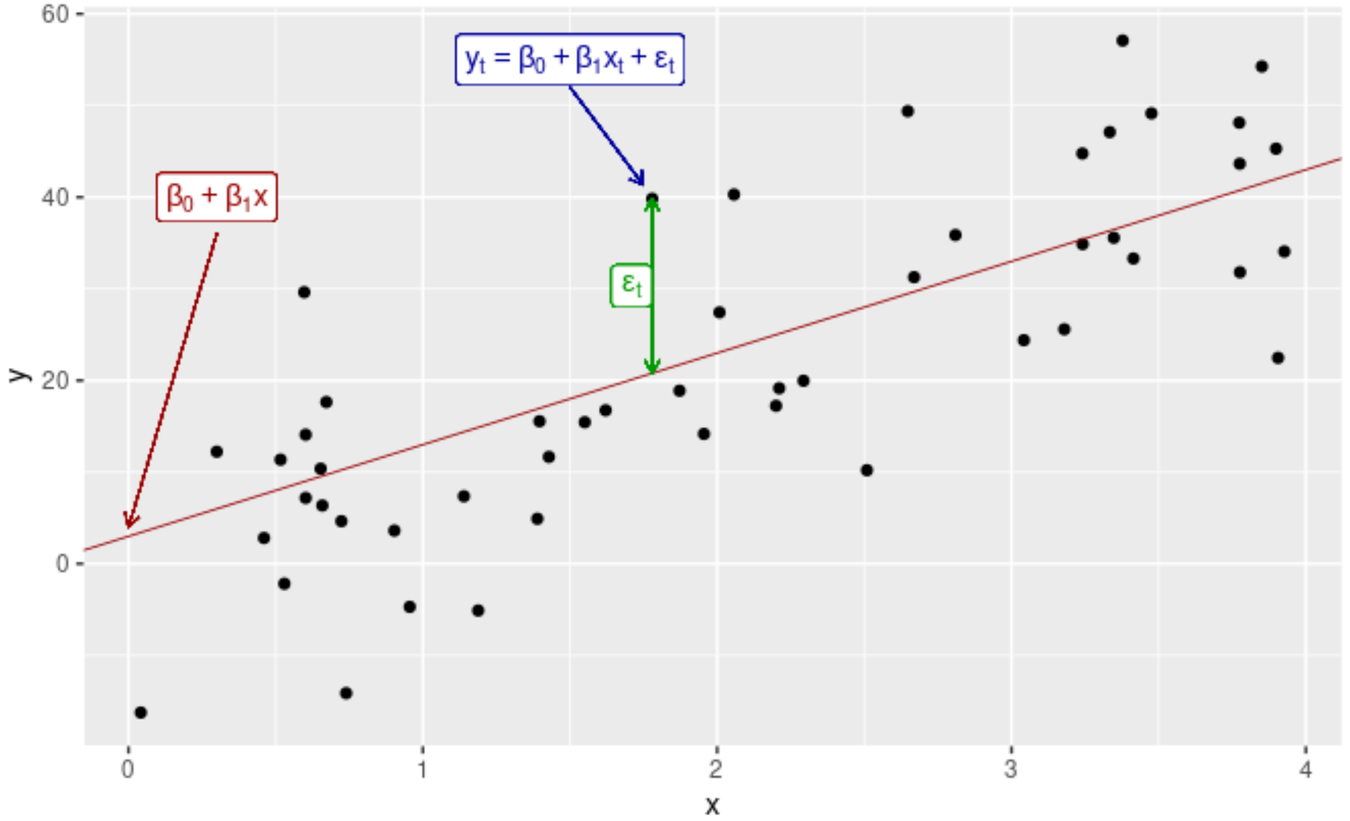




Time series. Residuals

- The “residuals” in a time series model are what is left over after fitting a model.



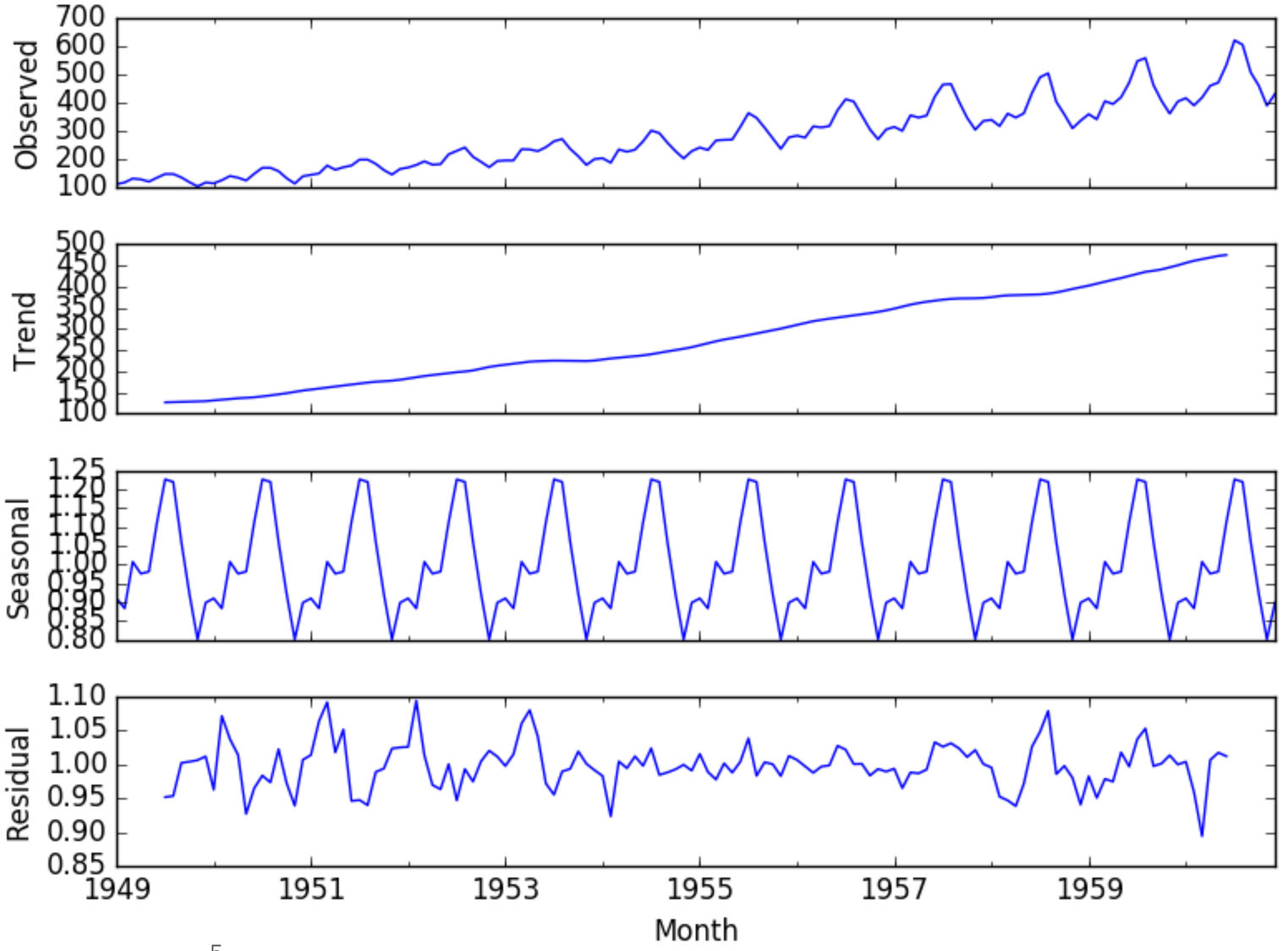


a simple linear model

Time Series. Decomposition

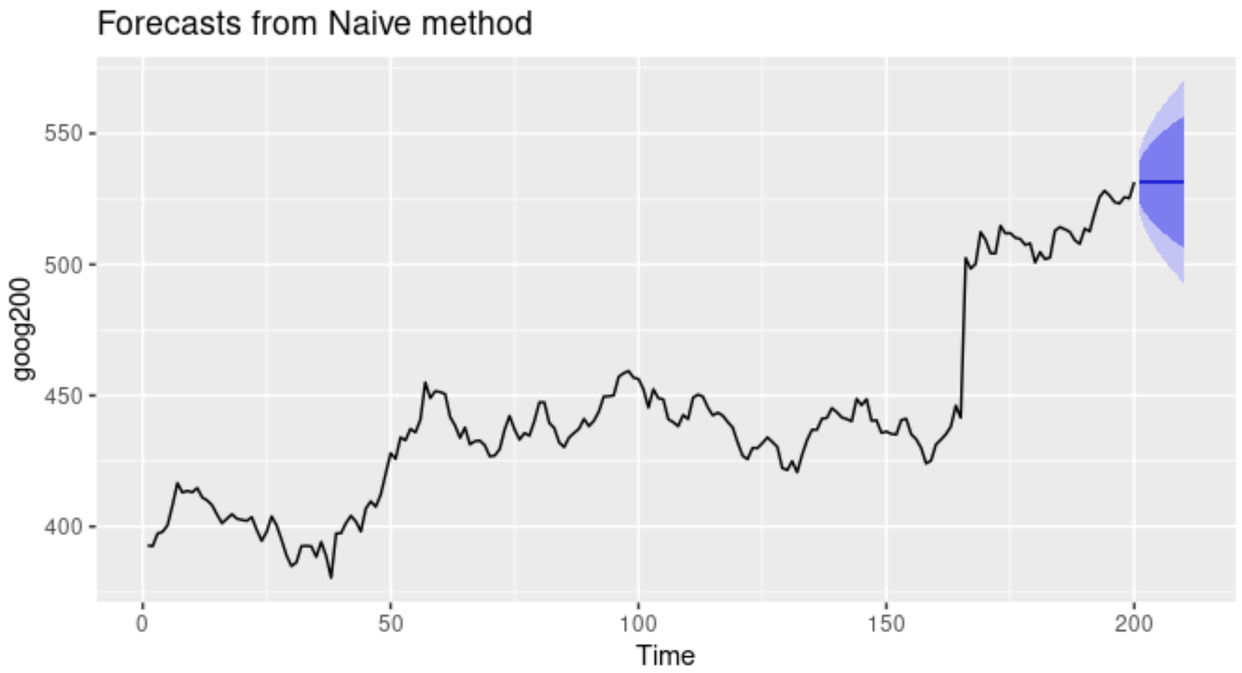
- Common approach is to decompose a time series into 3 components:
  - **Trend,**
  - **Seasonal,**
  - **Residual**

- So, the time series is a summation of them



Forecasting and Prediction intervals

- A prediction interval gives an interval within which we expect $y_{T+h|T}$ to lie with a specified probability.
- For example, assuming that the forecast errors are normally distributed, a 95% prediction interval
  - $y_{T+h|T} \pm 1.96\sigma_h$
  - where σ_h is the standard deviation of the h-step forecast distribution.

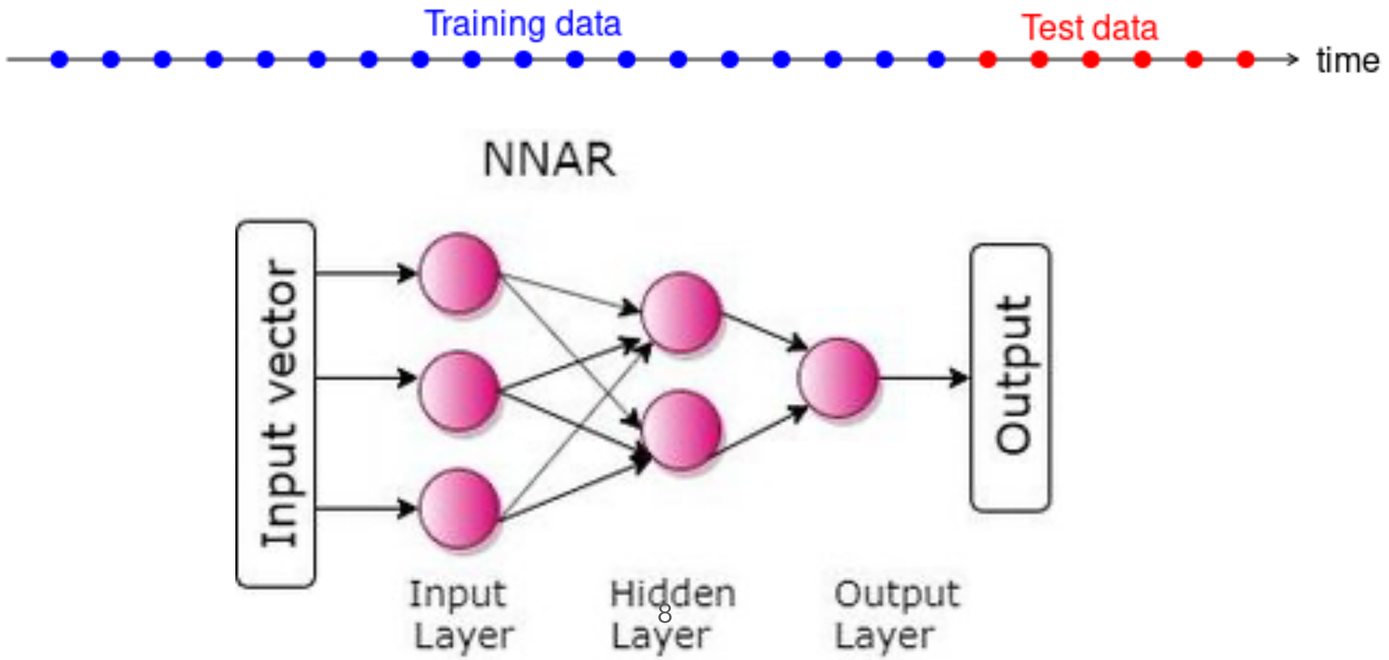


Models for forecasting

- There are many models with nice statistical properties:
  - AR
  - ARMA
  - ARIMA
  - SARIMA
  - SARIMAX
  - VARIMA
  - …
- But of course, we are NOT going to use them,
  - because we like **Neural Networks**!!!

Neural network autoregression, NNAR

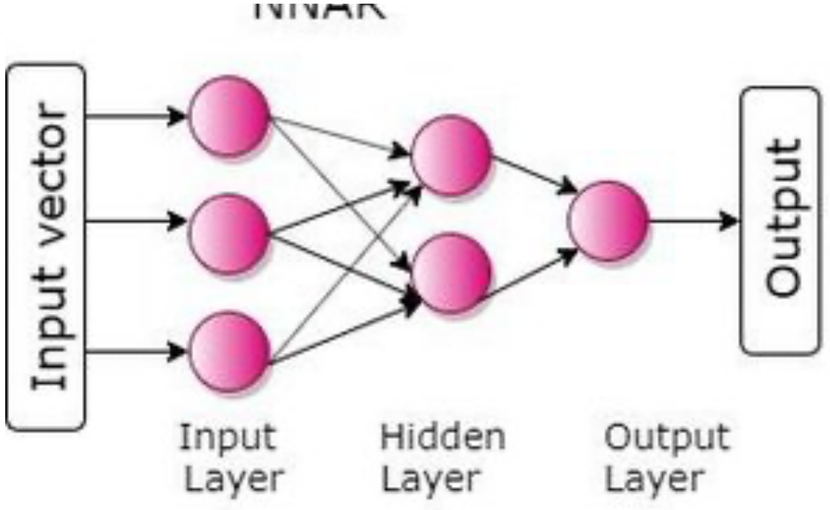
- With time series data, lagged values of the time series can be used as inputs to a neural network
- When it comes to the forecasting, the network is applied iteratively.
- For forecasting one step ahead, we simply use the available historical inputs



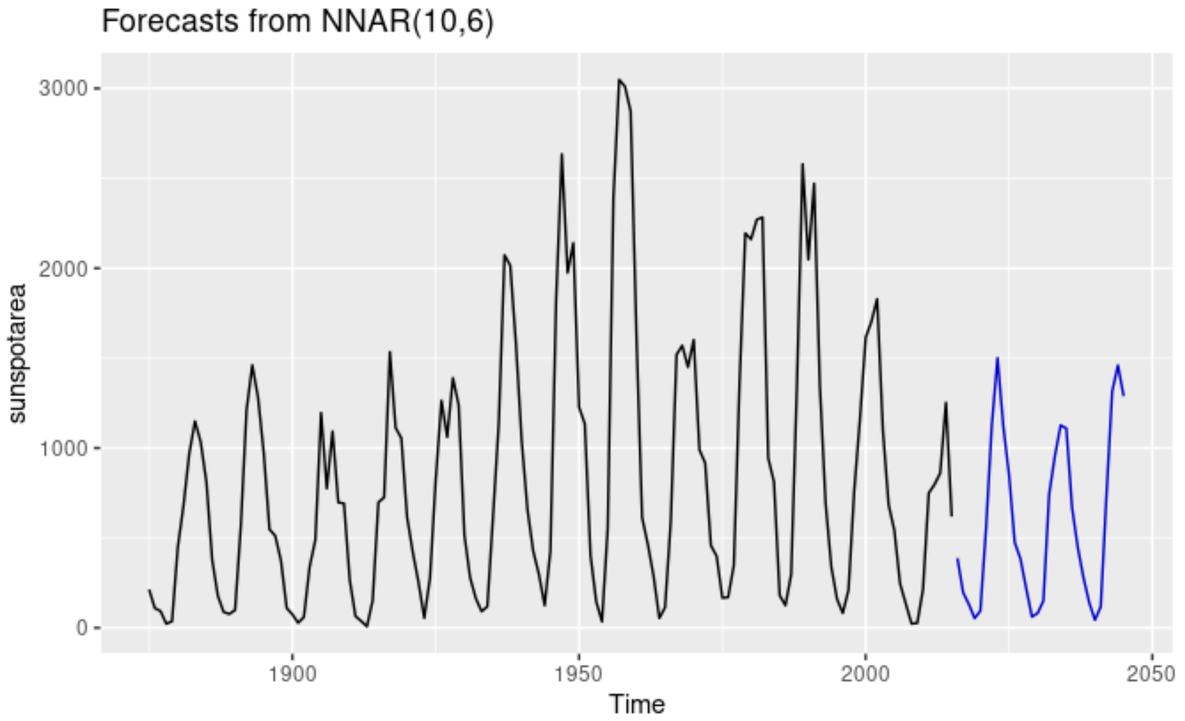
Neural network autoregression, NNAR

- The neural network fitted to the data can be written as

- $y_t = f(\boldsymbol{y}_{t-1}) + \varepsilon_t$

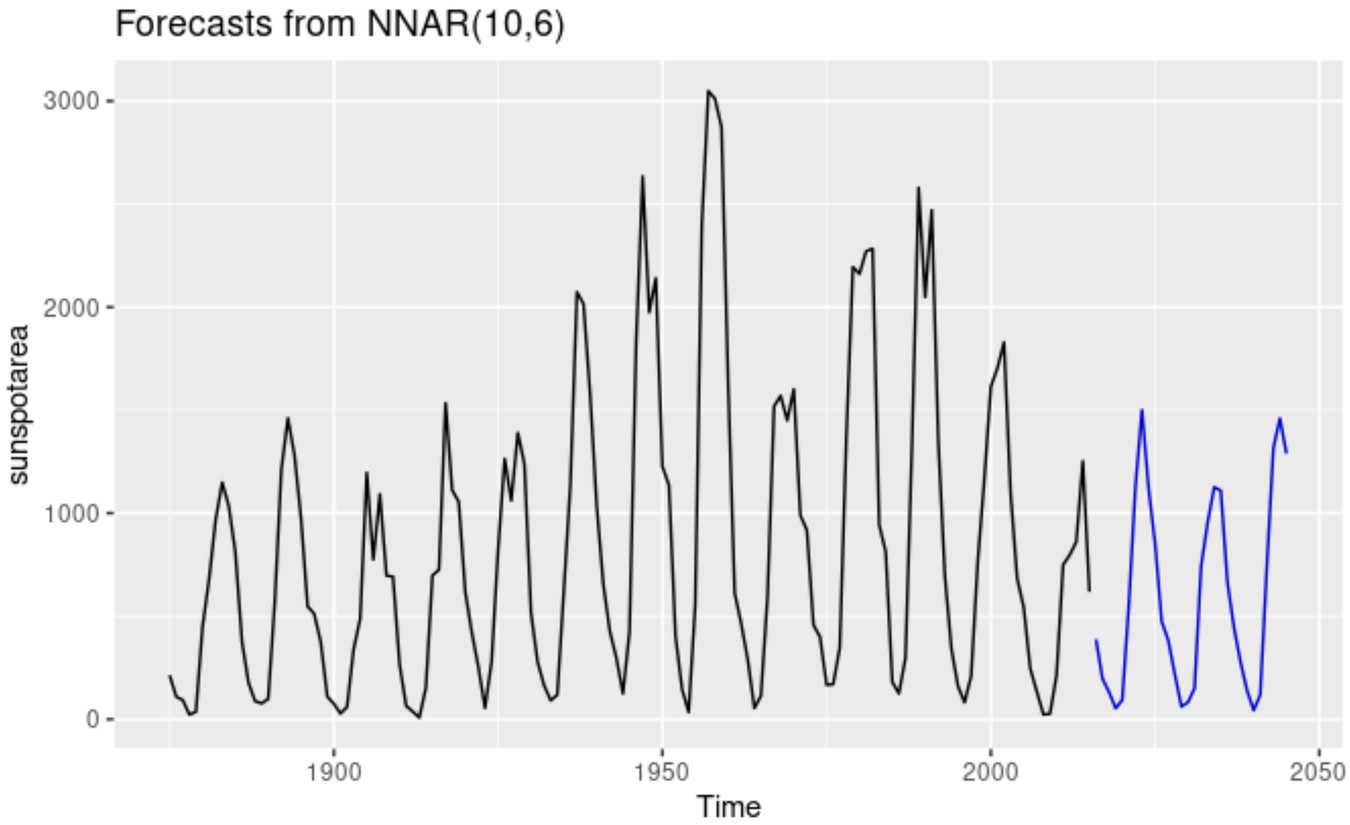


Forecasting



Discussion: How to assess the error of the prediction?

- We would like to have something like this
  - $y_{T+h|T} \pm 1.96\hat{\sigma}_h$
  - but we cannot assume normal distribution of the residuals,



Prediction Intervals

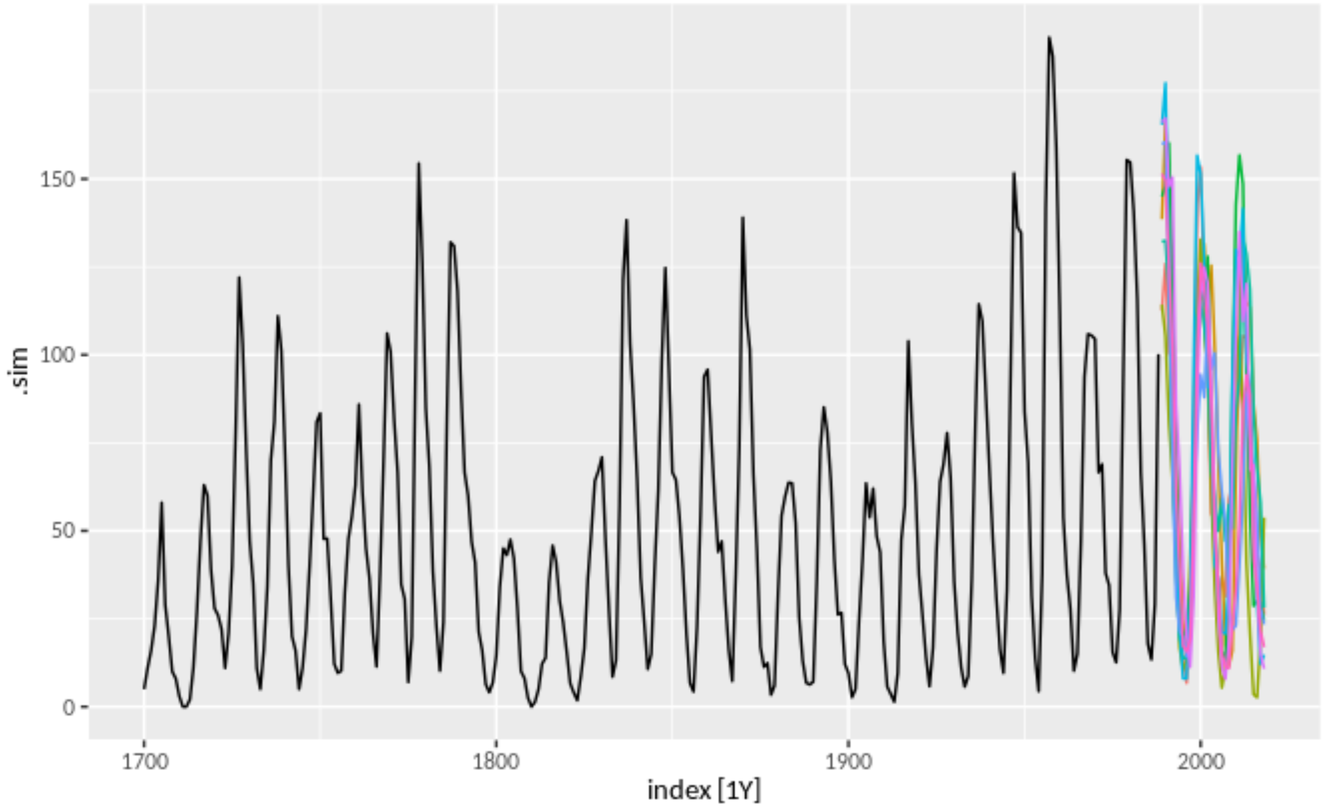
- Neural networks are not based on a well-defined stochastic model, and so it is not straightforward to derive prediction intervals for the resultant forecasts.
- However, we can still compute prediction intervals using simulation where future sample paths are generated using bootstrapped residuals

  - $y_t = f(\mathbf{y_{t-1}}) + \varepsilon_t$
  - $\mathbf{y_{t-1}}$ is a vector containing lagged values of the series

- How can we generate ε_t from historical data using bootstrap?

Example

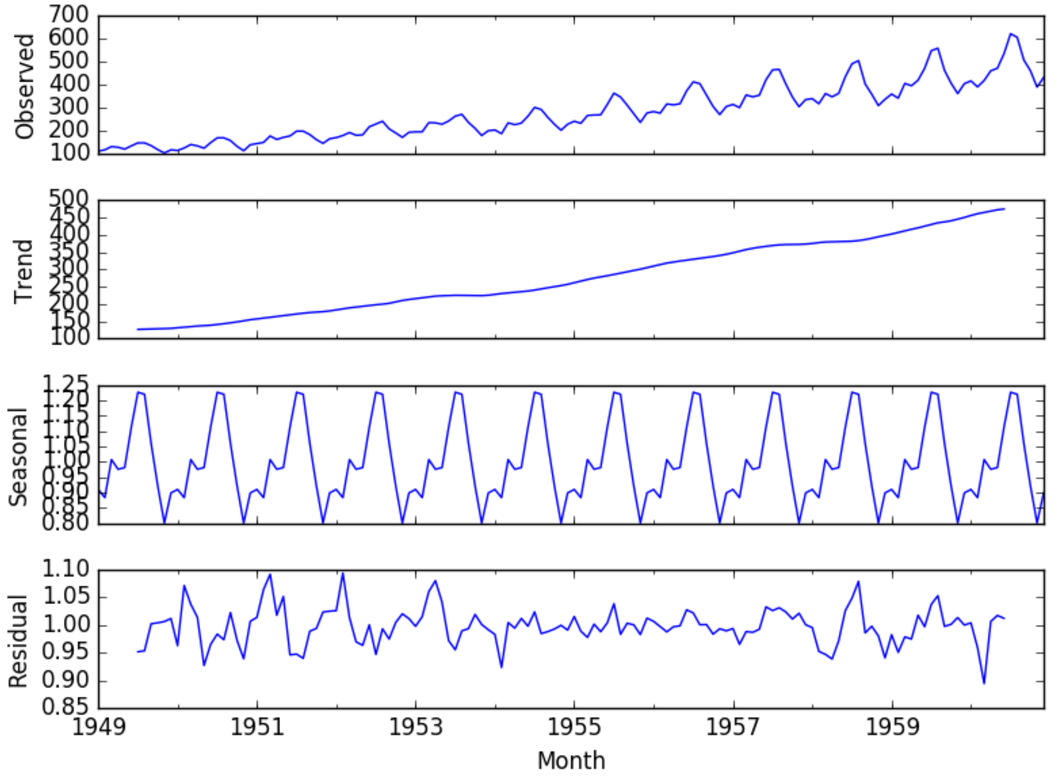
- Here is a simulation of 9 possible future sample paths
- Each sample path covers the next 30 years after the observed data.



Bootstrapped residuals

- First, the time series is transformed, and then decomposed into 3 components
  - trend,
  - seasonal and
  - remainder

- Then we obtain shuffled versions of the remainder component to get bootstrapped remainder series.

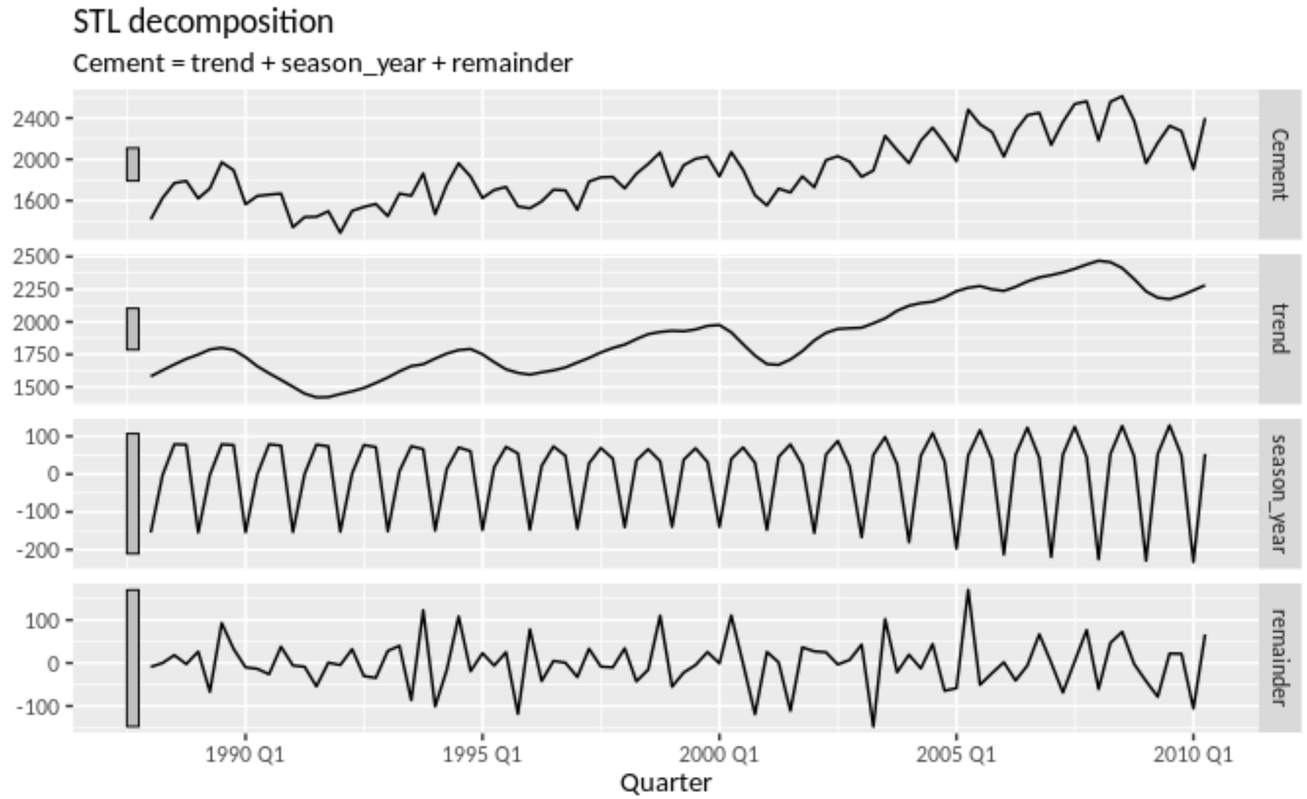


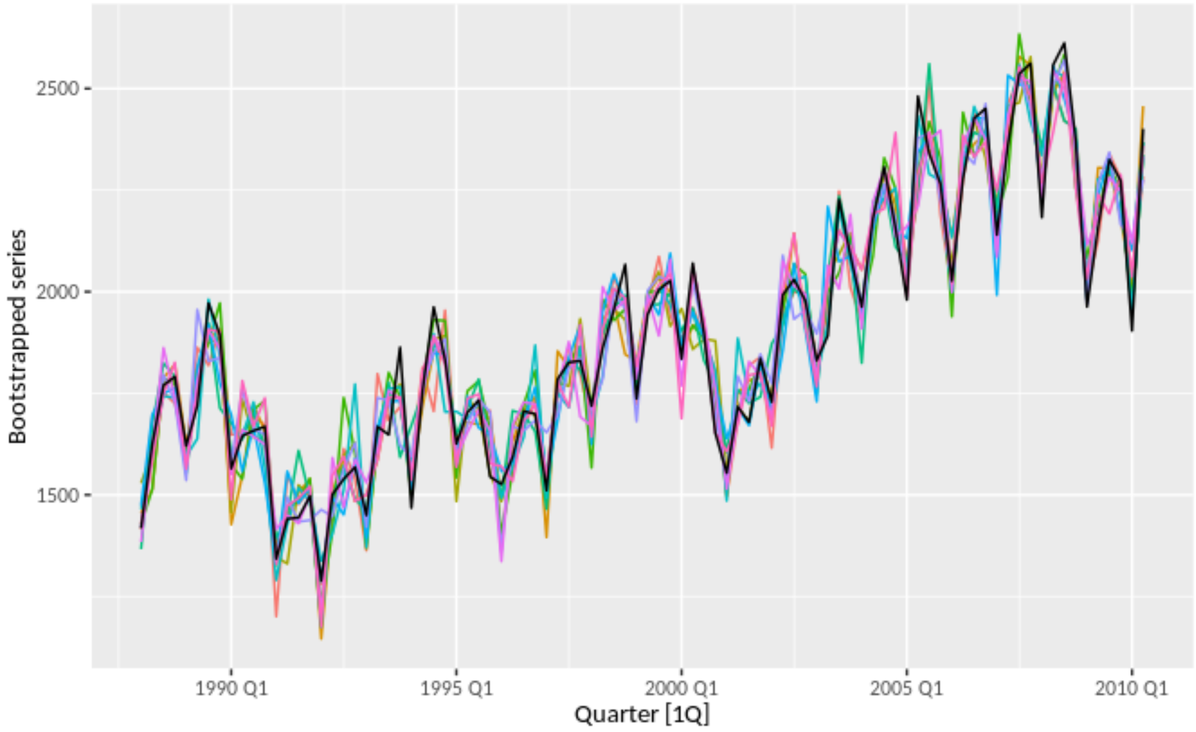
Bootstrapped residuals

- We use a “blocked bootstrap,” where contiguous sections of the remainder time series are selected at random and joined together.

- These bootstrapped remainder series are added to the trend and seasonal components, and the transformation is reversed to give variations on the original time series.

Example

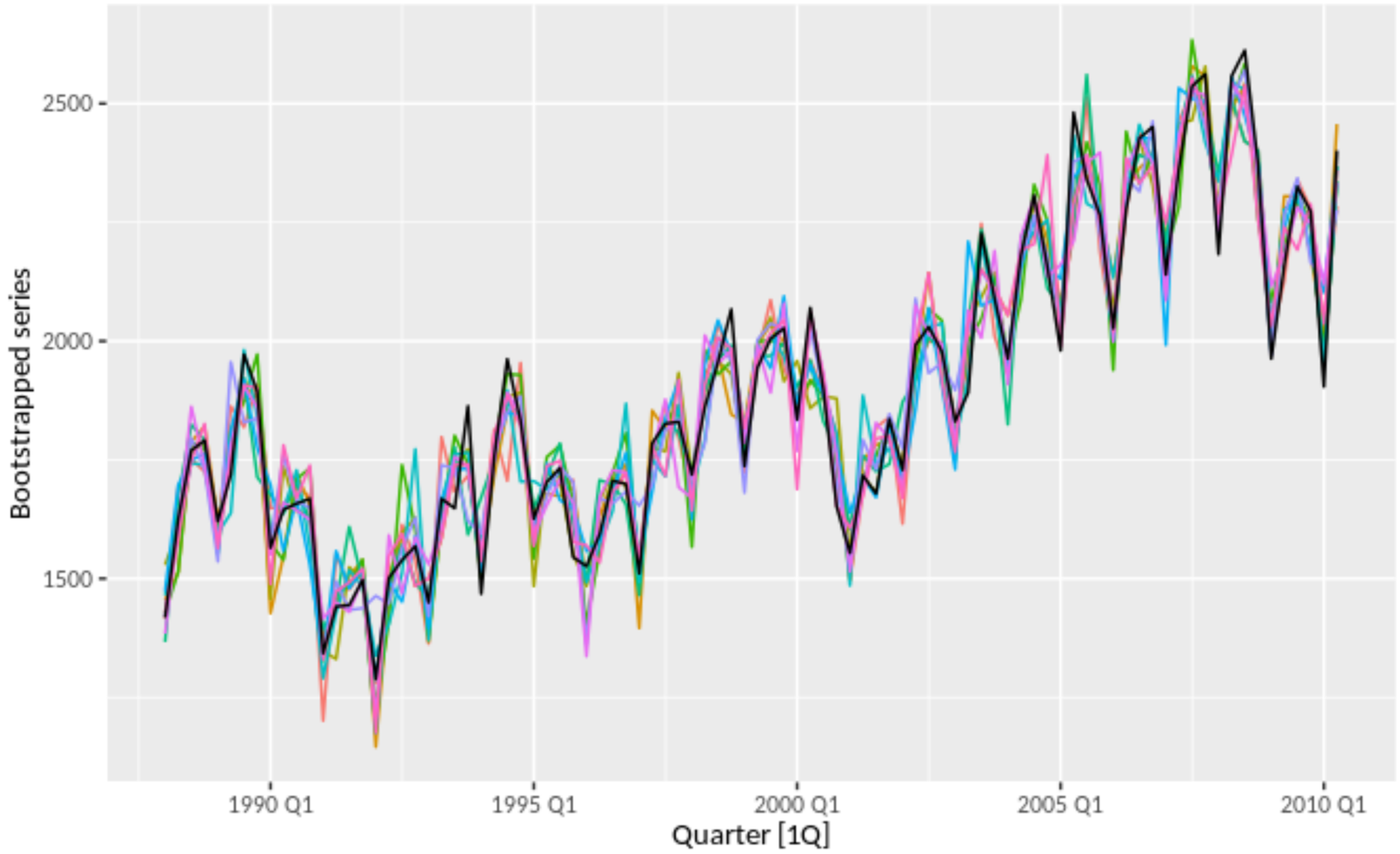




https://otexts.com/fpp3/bootstrap.html

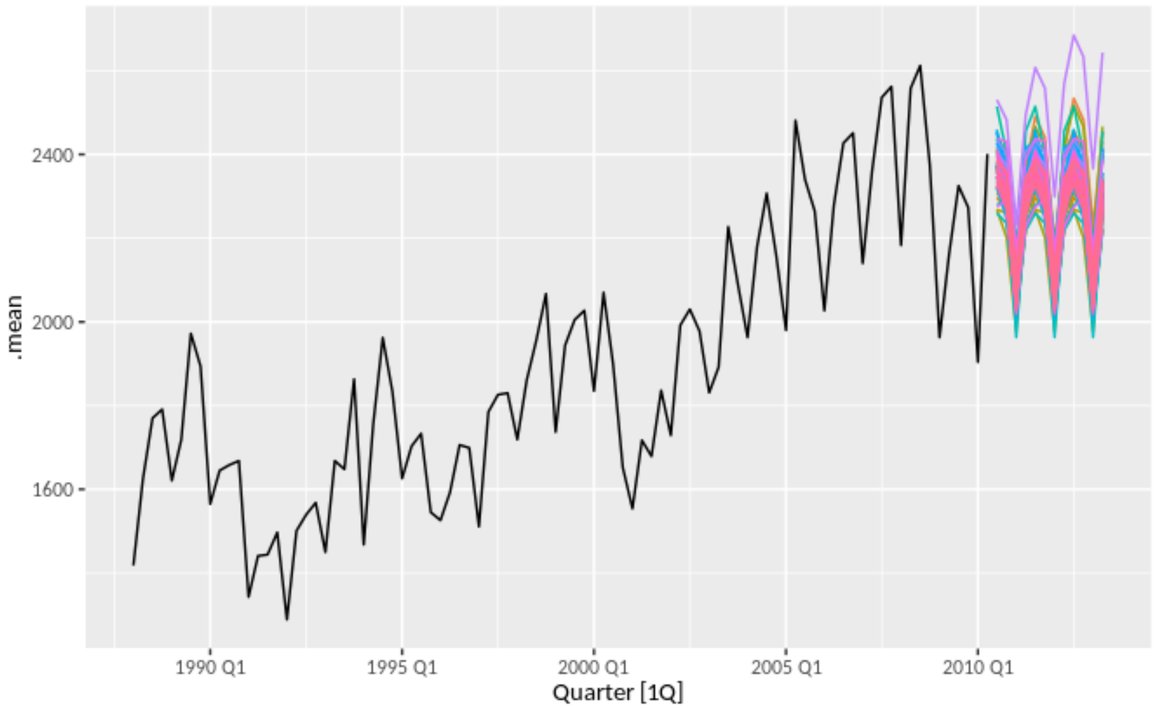
How to use the series / bootstrapped trajectories?

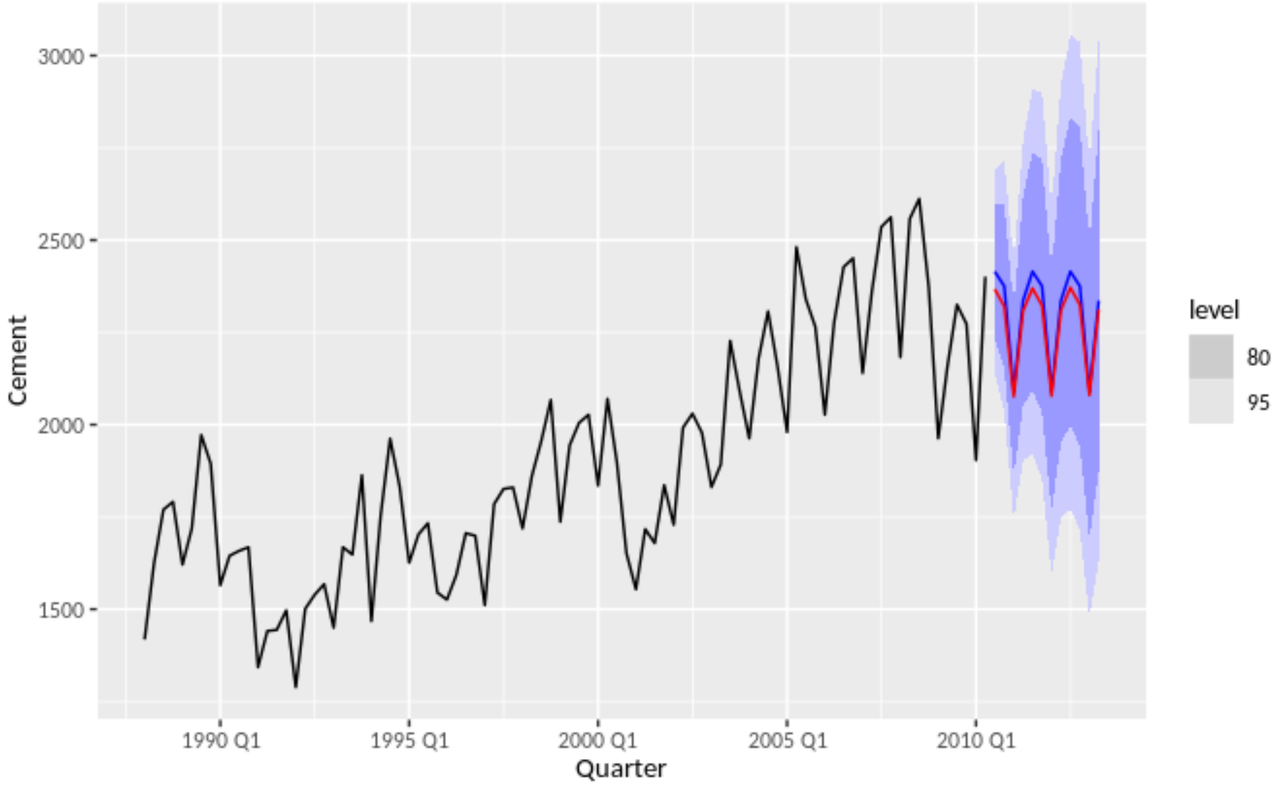
- So, we have several trajectories derived through bootstrap!



Bagging (“bootstrap aggregating”) !

- Predict for each trajectory and aggregate!
- It is shown that bagging gives better forecasts than classic methods such as ETS





Bergmeir, C., Hyndman, R. J., & Benítez, J. M. (2016). Bagging exponential smoothing methods using STL decomposition and Box-Cox transformation. International Journal of Forecasting, 32(2), 303–312. https://doi.org/10.1016/j.ijforecast.2015.07.002

Summary

Objectives for today

- to understand
  - Histograms
  - Smoothing
  - Kernel Density Estimation

Smoothing. Kernel Density Estimation

Outline

- Part 1: Histogram
- Part 2: Kernel density estimation

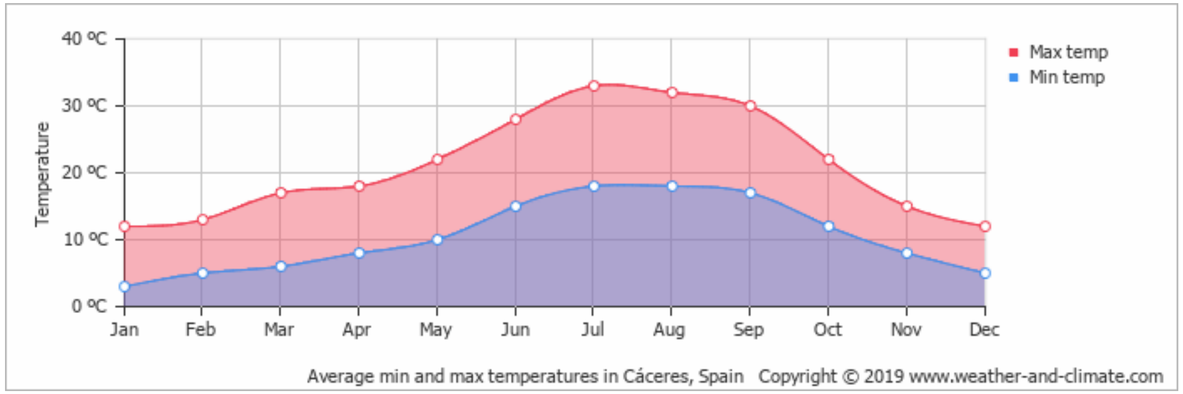
Problem Statement

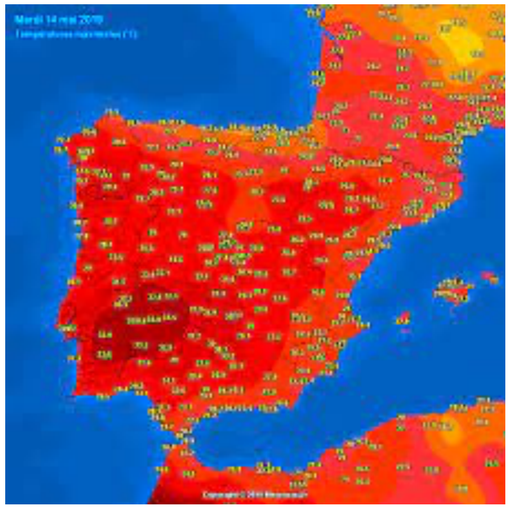
- Goal: Better EDA
  - **Visual summaries** convey concisely the deep structure of the data which would otherwise be difficult to discern from purely numerical summaries

- Dataset
  - We suppose that the observed dataset consists of n realisations of a continuous multi-dimensional random variable $\mathbf{X}$
  - d is the number of dimensions, n is a sample size

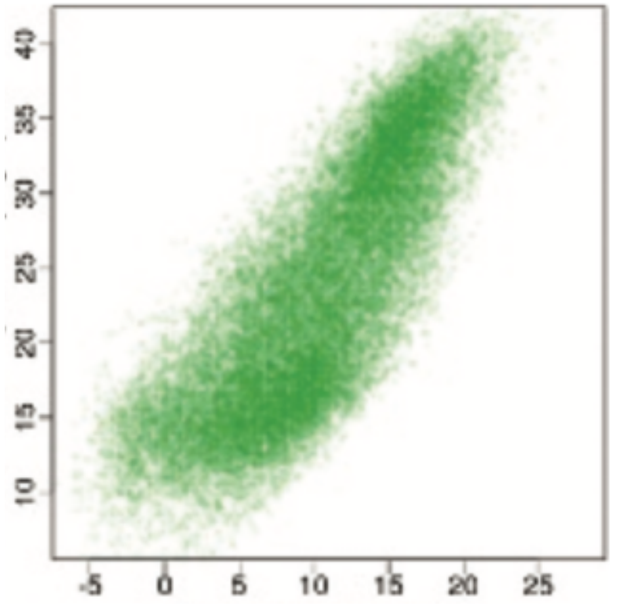
Example

- Data about min/max temperatures in a certain place:
  - n = 21908 days between 1 Jan 1955 to 31 Dec 2015,





Scatter-plot + alpha blending

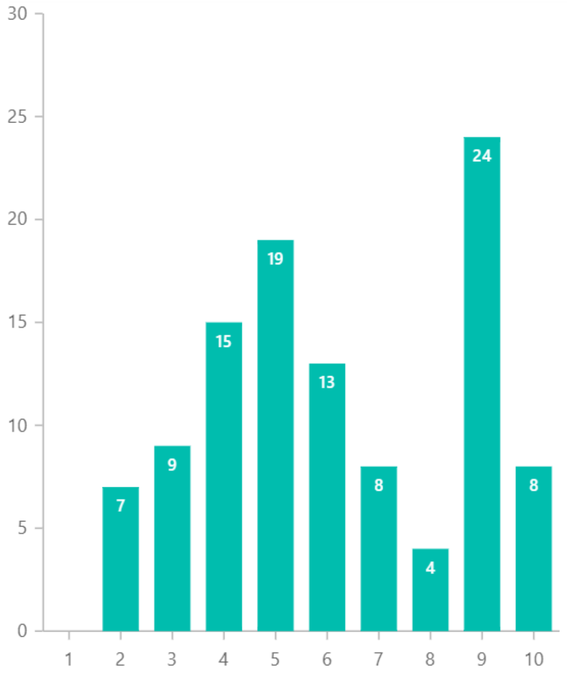


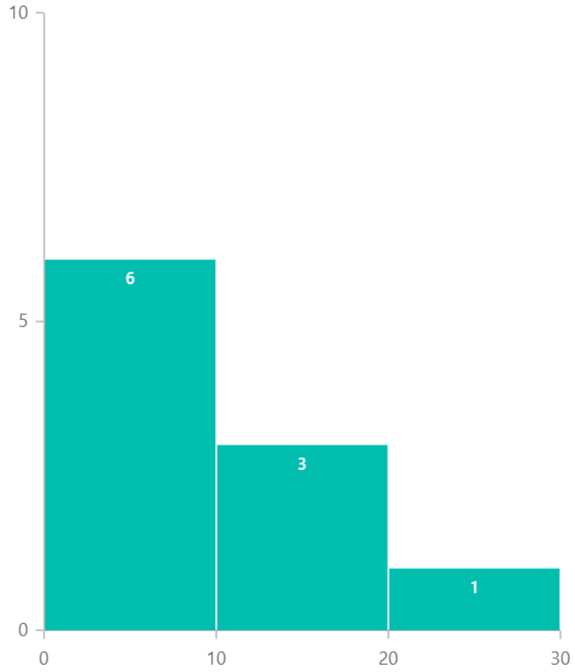
Definitions

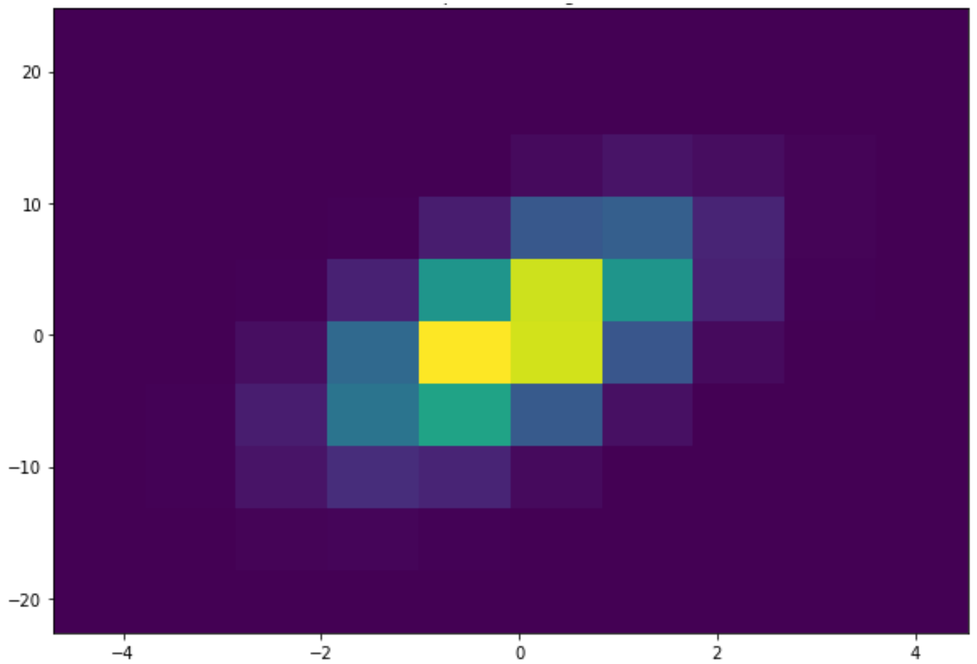
- Sample with n observations:
  - $\mathbf{X_1}, \mathbf{X_2}, \ldots, \mathbf{X_n}$ where
  - each $\mathbf{X}_i$ is a d-dimensional random vector with density function f
  - a realisation of $\mathbf{X}_i$ in $\mathbb{R}^d : \boldsymbol{x} = (x_1, \ldots, x_d)$

Part 1. Histogram density estimation

Which of them are histograms?

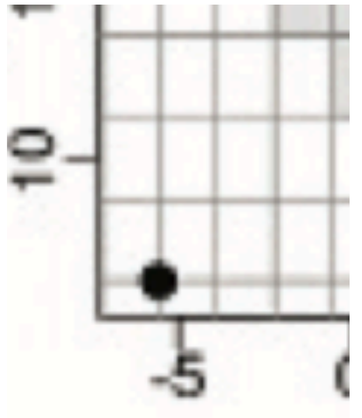




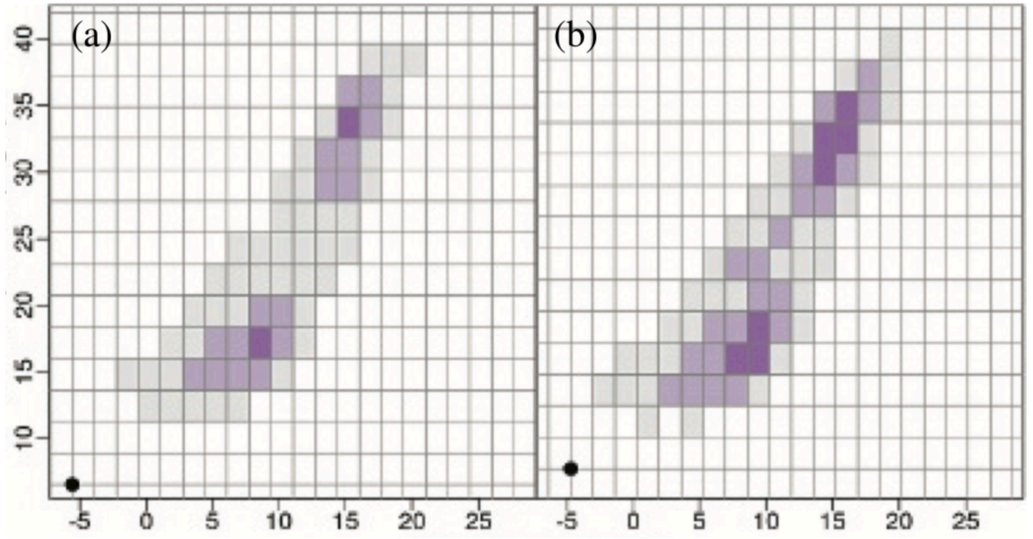


Histogram: Step by step

d=2



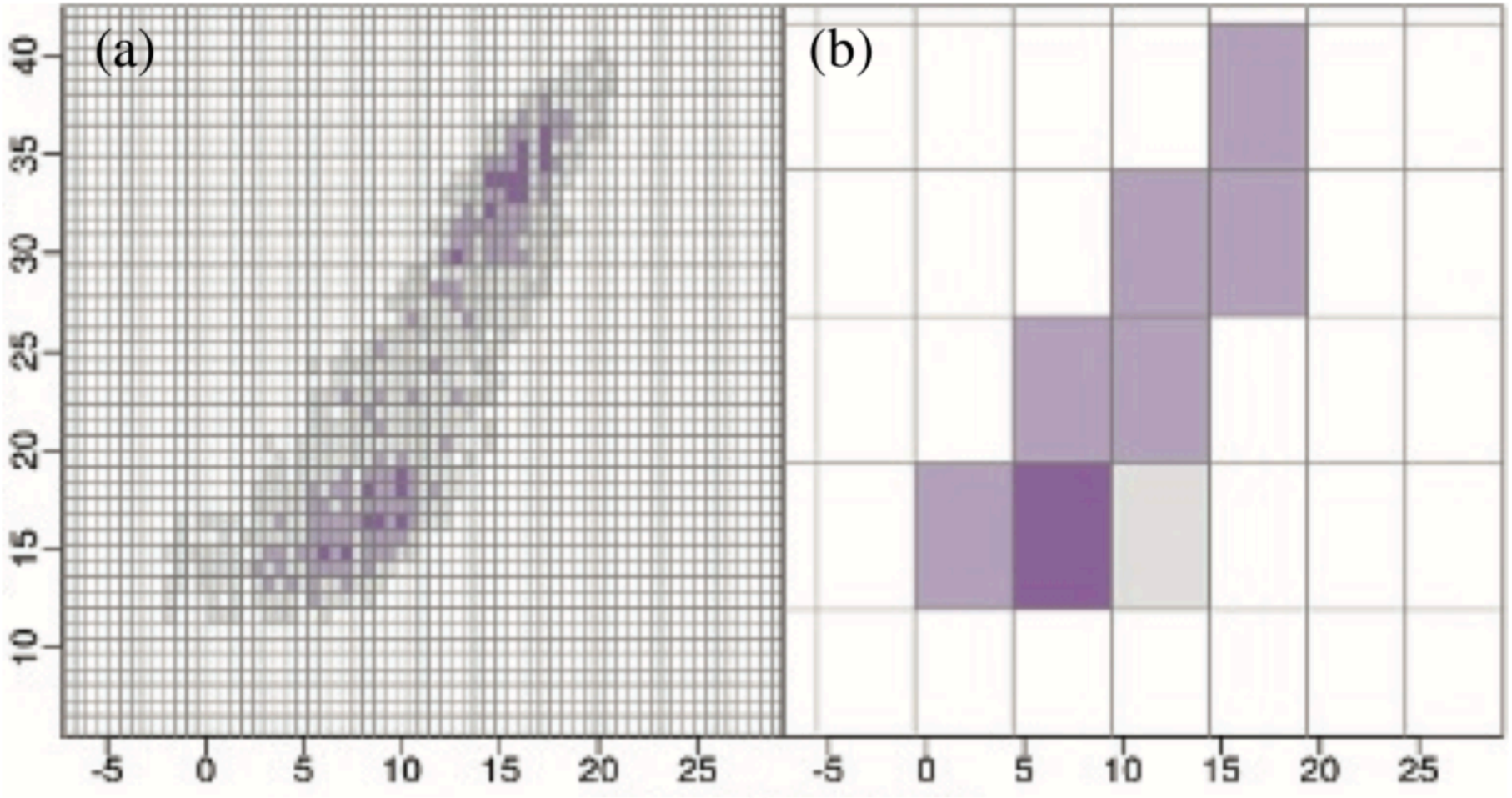
- discretise the sample space in subregions known as bins
- discretisation **usually** consists of an <u>equidistant</u> rectangular grid
- the grid is defined by:
  - the **anchor** point (black dot) and
  - M **bins** $B_1, \ldots, B_M$, each with width b_i $(i = 1,...,d)$



Histogram

- A histogram density estimator is a step function with a constant value within each of the bins, where the constant is given by the proportion of data points $\mathbf{X}_i$ which fall in the bin divided by the bin volume.
  - $\hat{f}_{hist}(\boldsymbol{x}, B_j) = \dfrac{N_j}{nb_1 \ldots b_d}, \quad \forall \boldsymbol{x} \in B_j$
  - where $N_j = \sum_{i=1}^{n} 1\{\mathbf{X}_i \in B_j\}$, the random variable that counts the number of data points fall in the j-th bin

Undersmoothing and oversmoothing



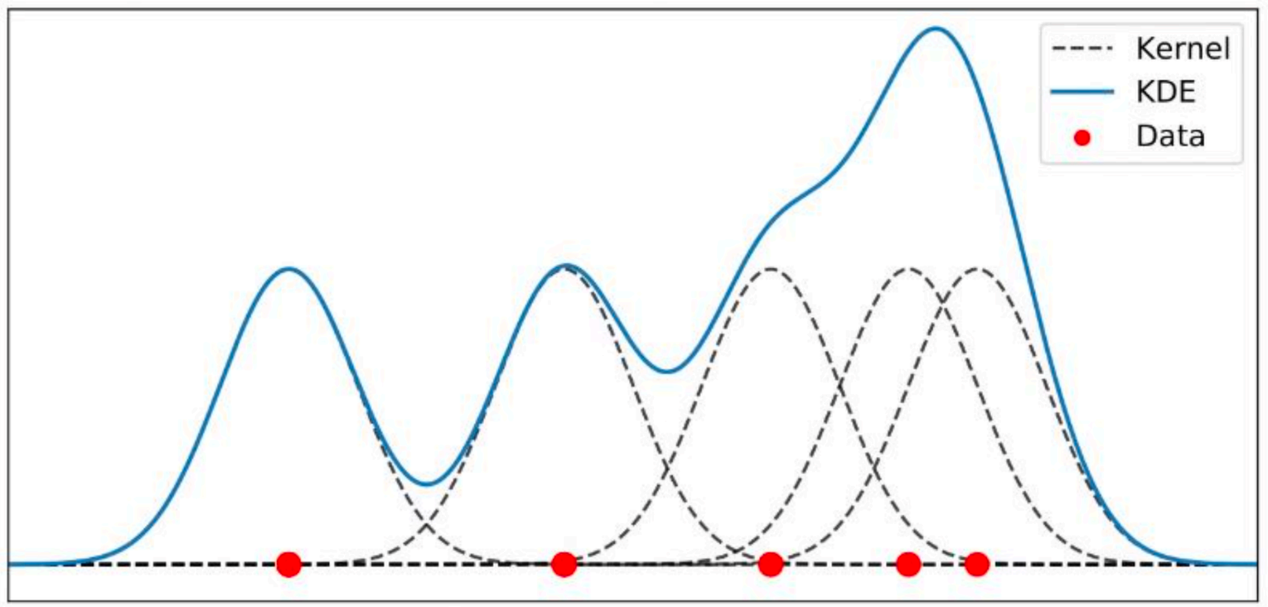
Is there an optimal binwidth?

- If the target density f is normal, then the following $\hat{b}_{NS,i}$ is optimal
  - $\hat{b}_{NS,i} = 2 \cdot 3^{\frac{1}{d+2}} \pi^{\frac{d}{2d+4}} s_i n^{-\frac{1}{d+2}}$ where
  - s_i is the i-th marginal standard deviation for i-th dimension
  - the estimation error $MSE\{\hat{f}(\boldsymbol{x}\} = \mathbb{E}\{[\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})]^2\}$ grows as f becomes “less” normal

Kernel Density Estimation and Smoothing

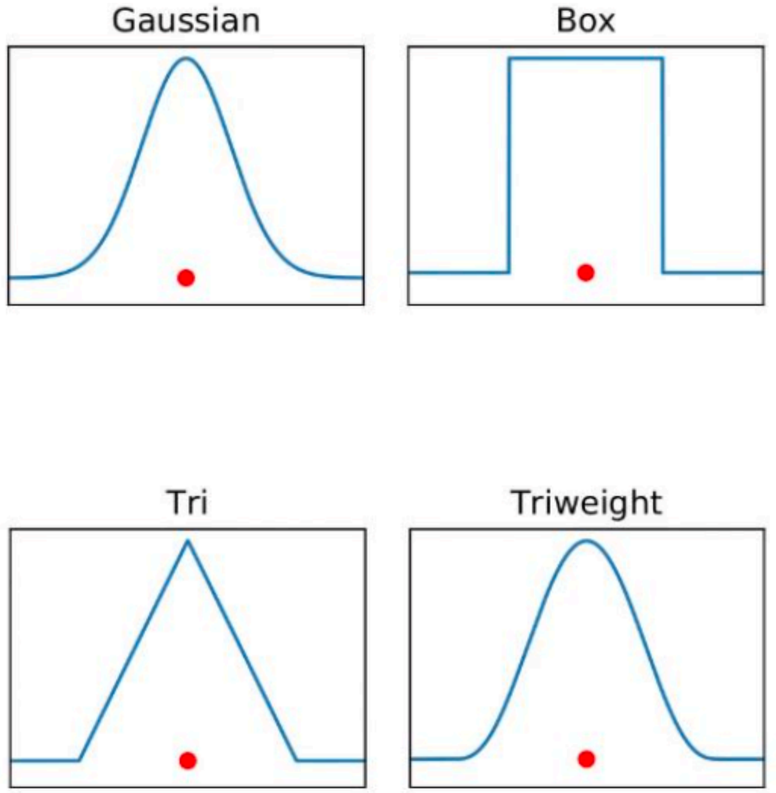
Kernel density estimation

- Kernel density estimation (KDE) is a non-parametric way to estimate the PDF of a random variable
- KDE is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample



Kernel density estimation

- Let $X_1, \ldots, X_n$ be a univariate i.i.d. sample drawn from some distribution with an unknown density f
- We are interested in estimating the shape of this function f. Its kernel density estimator is
  - $$\hat{f} = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$
- where $K(\cdot)$ is a kernel, a a non-negative function — and $h > 0$ is a smoothing parameter called the bandwidth.



Kernels

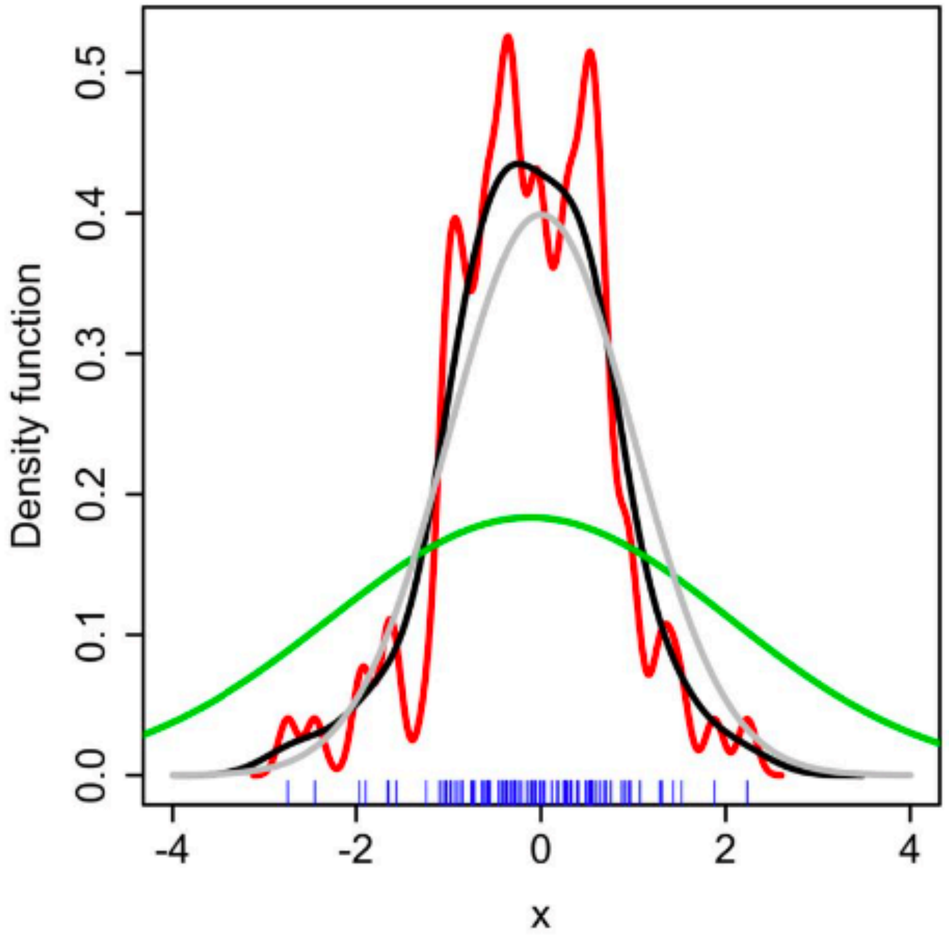
- The choice of kernel is not so important for big samples
- However, the choice of **bandwidth (h)** is very important.

- Properties of any non-negative function f to be a kernel :
  - 1) f integrates to one
  - 2) Expectation of f is zero
  - 3) Variance of f is finite

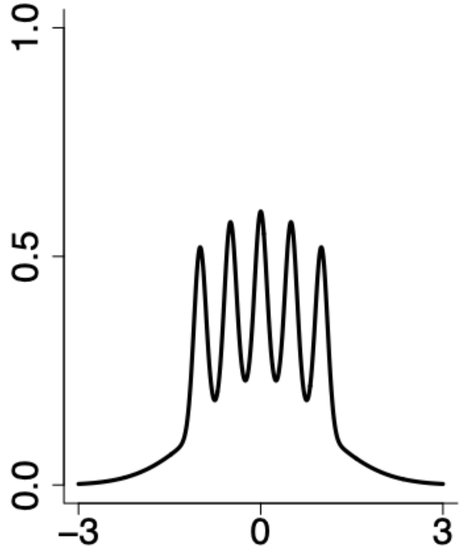
Have you met the term kernel before?

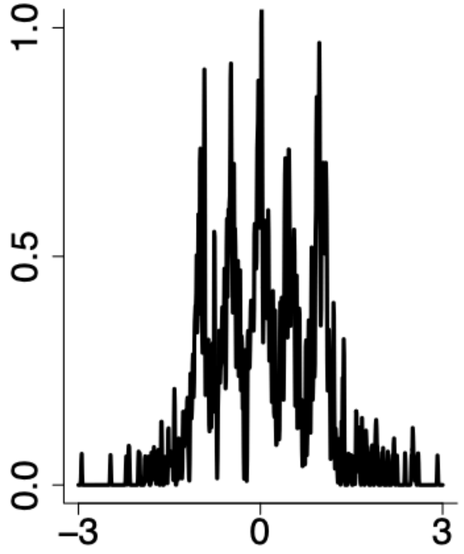
KDE. Bandwidth

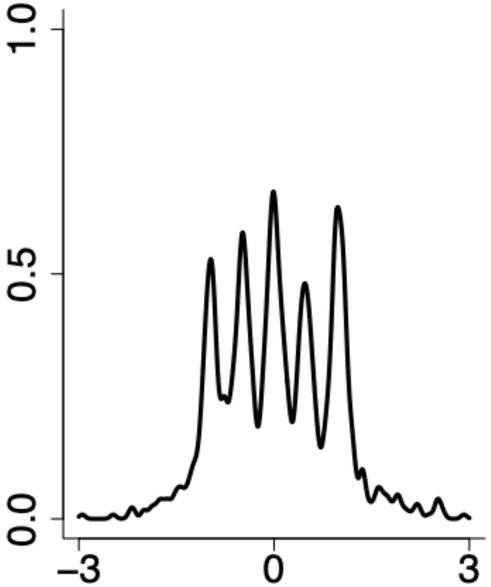
- Intuitively, one wants to choose h as small as the data will allow.
- However, there is always a trade-off between the bias of the estimator and its variance.
- We take a simulated random sample from the N(0, 1).
  - the grey curve is the true density
  - the red curve is undersmoothed (h = 0.05)
  - the green curve is oversmoothed (h = 2)
  - the black curve is considered to be optimally smoothed (h = 0.337)

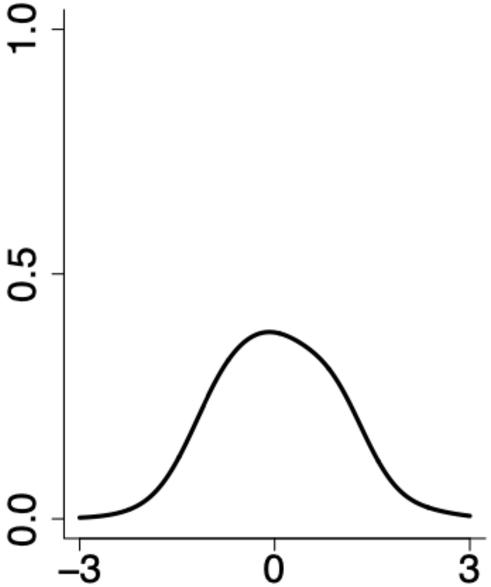


Which is where?



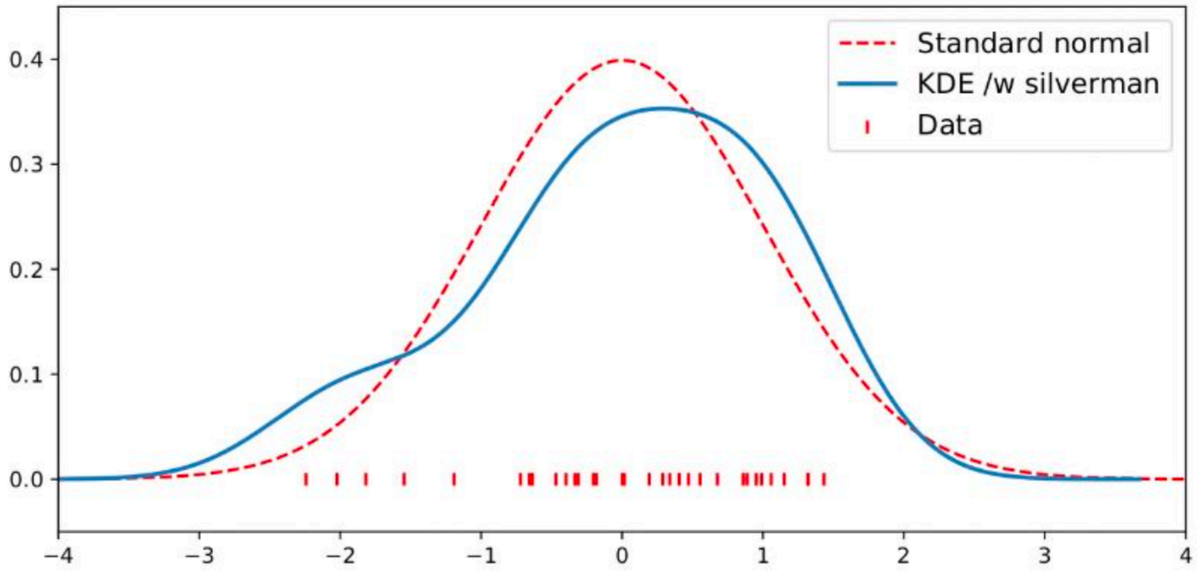






Silverman rule-of-thumb

- If Gaussian basis functions are used to approximate the data, and the underlying density being estimated is Gaussian, the optimal choice for h is
  - $h = \left(\frac{4\hat{\sigma}^5}{3n} \right)^{\frac{1}{5}} \approx 1.06\hat{\sigma}n^{-1/5}$
    - where $\hat{\sigma}$ is an estimate of standard deviation of samples
- it should be used with caution as it can yield widely inaccurate estimates when the density is not close to normal.



Why do we need KDE?

Break

Kernel Density Estimation in more than 1 dimension

- $\hat{f}(\boldsymbol{x}, \mathbf{H}) = \frac{1}{n} \sum_{i=1}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i)$

- K_H is a scaled kernel build using a kernel function K,

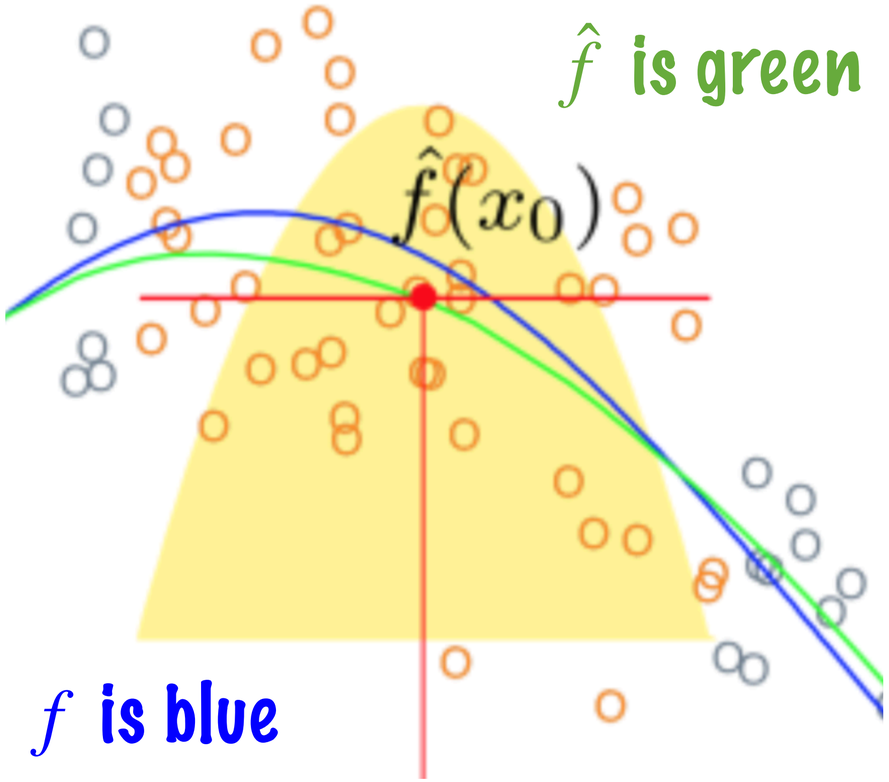
- K it is an integrable function with unit integral

KDE. Interpretation

$$\hat{f}(\boldsymbol{x}, \mathbf{H}) = \frac{1}{n} \sum_{i=1}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i)$$

- (i) from an estimation point $\boldsymbol{x}$, it is a **local** weighted averaging estimator where the weight of $\mathbf{X}_i$ decreases as its distance to $\boldsymbol{x}$ increases,

- (ii) from a data point $\mathbf{X}_i$, its probability mass is smoothed in the **local** neighbourhood according to the scaled kernel to represent the unobserved data points



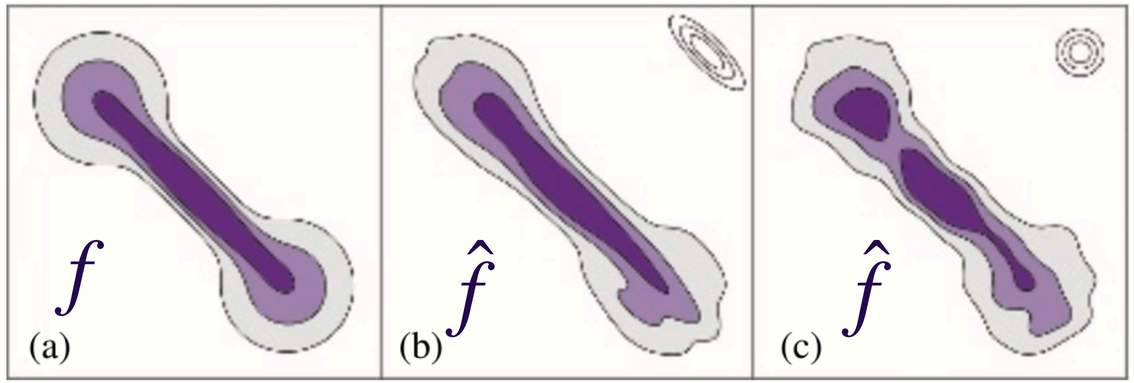
Bandwidth, $\mathbf{H}$

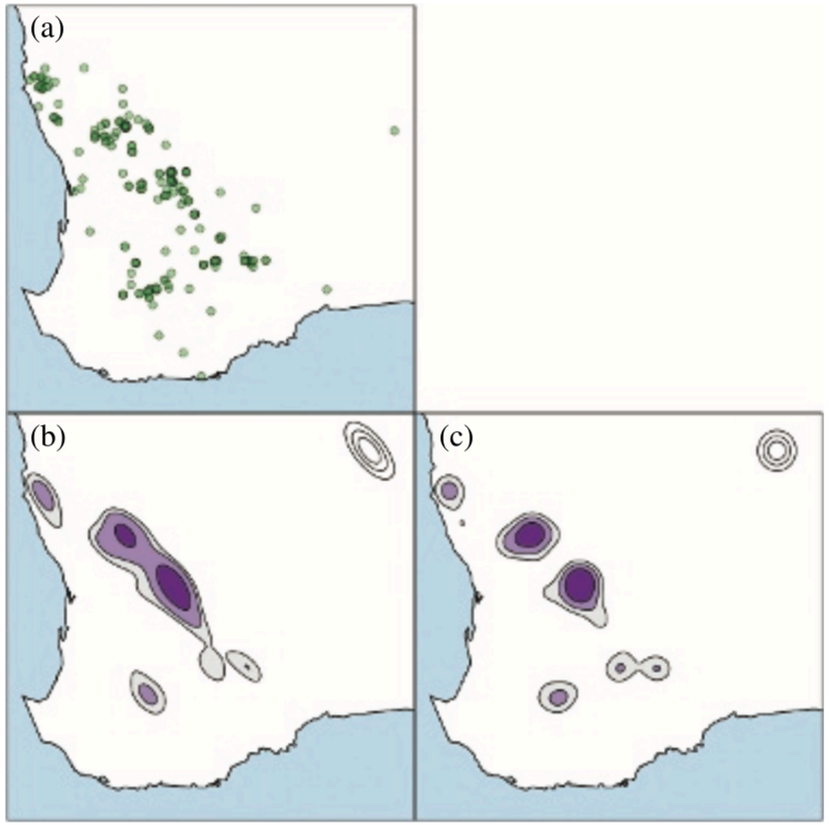
- The crucial tuning parameter is the bandwidth, $\mathbf{H}$
- $\mathbf{H}$ is a symmetric, positive definite, $d \times d$ matrix of smoothing parameters
- The bandwidth controls the orientation and the extent of the smoothing applied via the scaled kernel

$$K_{\mathbf{H}}(\boldsymbol{x}) = |\mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}}\boldsymbol{x})$$

Example bandwidths

- Options for the bandwidth:
  - $\mathbf{H} = h^2 I$
  - $\mathbf{H} = Diag(h_1^2, \ldots, h_d^2); h_i > 0$





Scaling, elimination of anchor

- $K_{\mathbf{H}}(\boldsymbol{x}) = |\mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}}\boldsymbol{x})$

- $K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i)$ is centred at each data point $\mathbf{X}_i$
- then we averaging over all data points…

$$\hat{f}(\boldsymbol{x}, \mathbf{H}) = \frac{1}{n} \sum_{i=1}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i)$$

Normal (gaussian) kernel

- $K(\boldsymbol{x}) = (2\pi)^{-d/2}\exp(-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x})$ - standard normal: N(0,I)

- Scaled kernel: $K_{\mathbf{H}}(\boldsymbol{x}) = |\mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}}\boldsymbol{x})$

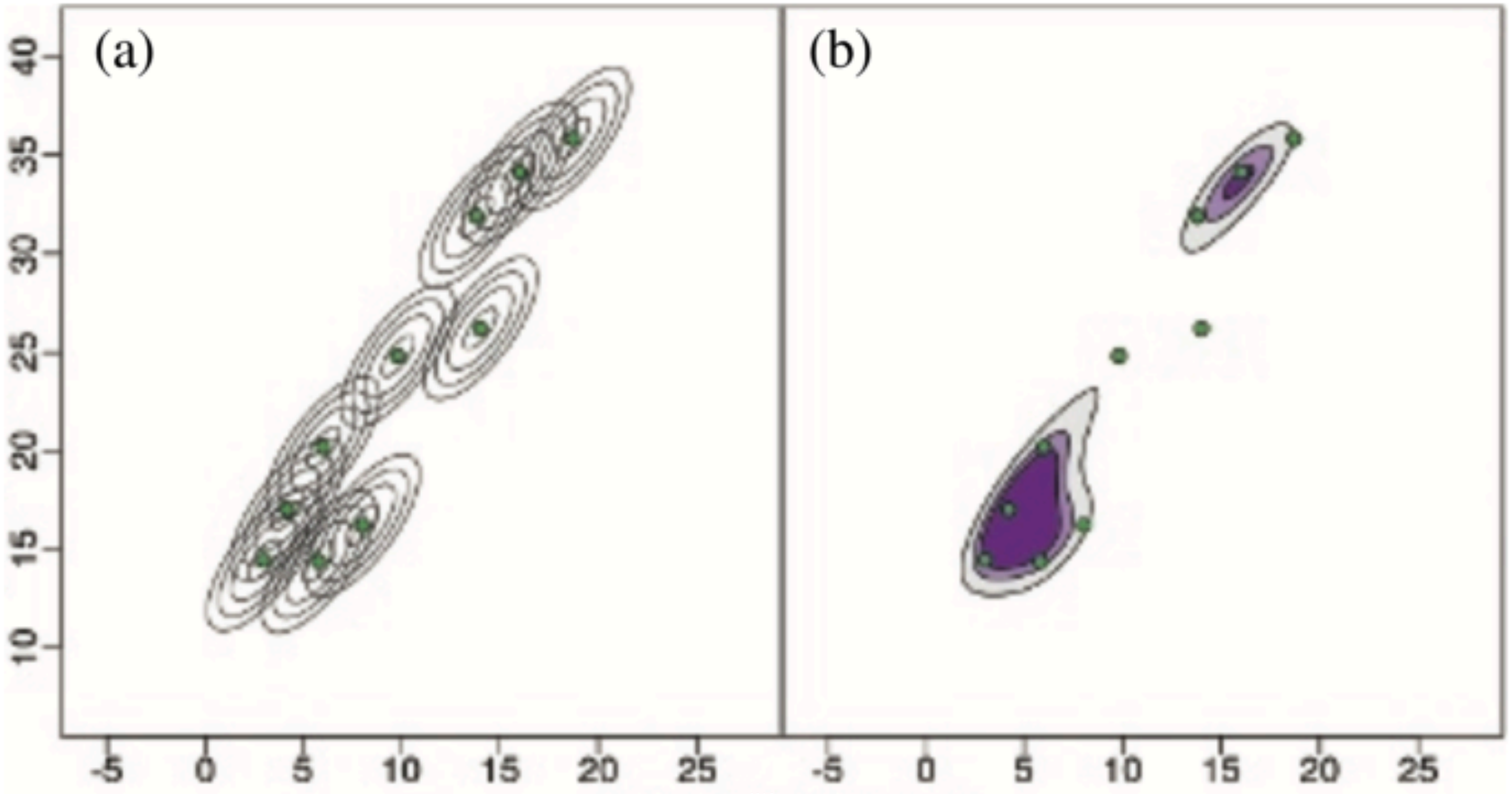
- Let us derive the explicit form for $K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i)$ for MVN (multi variate normal)

$$K_{\mathbf{H}}(\mathbf{x} - \mathbf{X}_i) = (2\pi)^{-d/2} |\mathbf{H}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{X}_i)^T \mathbf{H}^{-1} (\mathbf{x} - \mathbf{X}_i)\right)$$

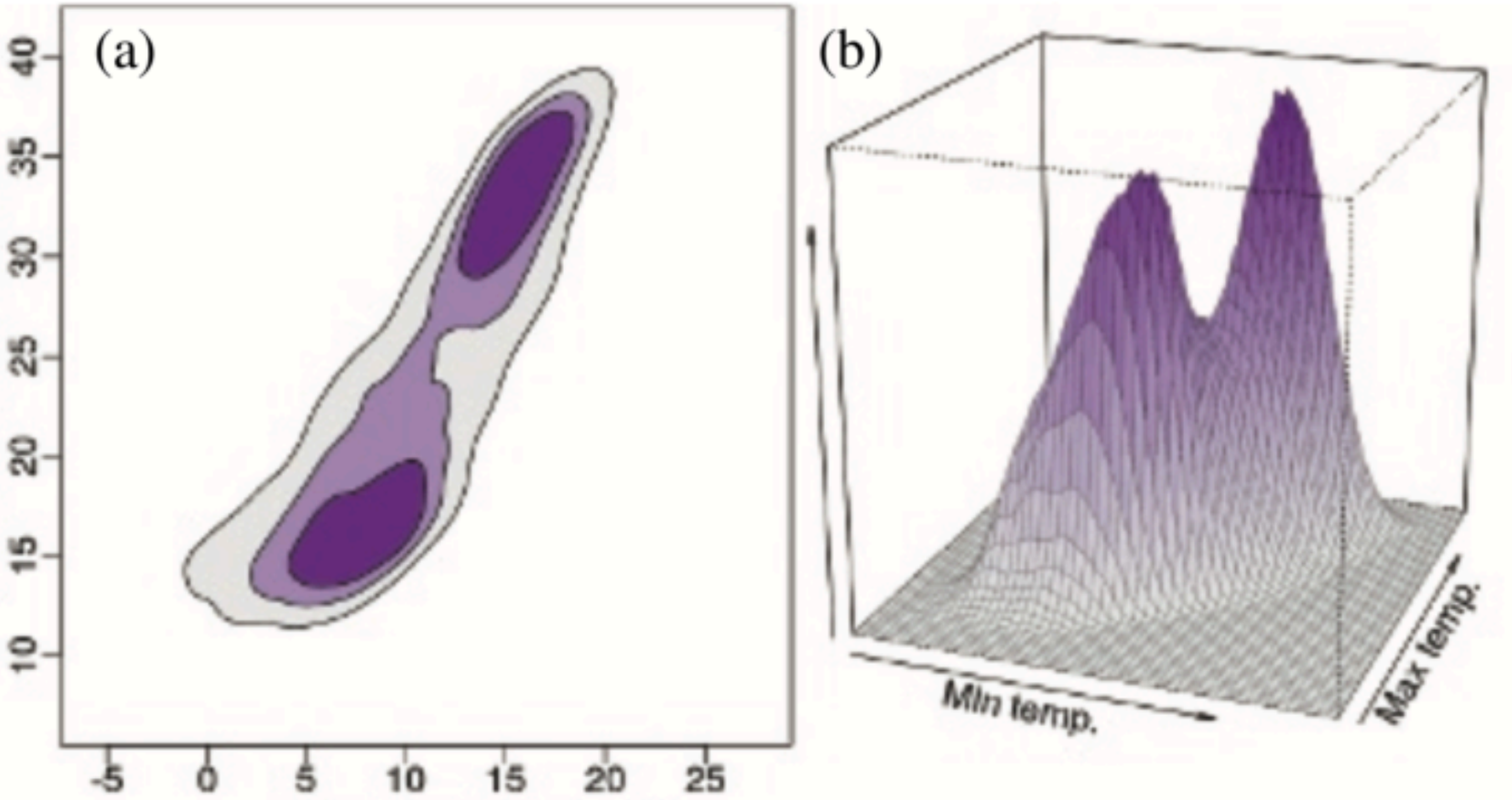
MVN density and normal kernel

- Gaussian (normal) kernel:
  - $K_{\mathbf{H}}(\boldsymbol{x} - \mathbf{X}_i) = (2\pi)^{-d/2} |\mathbf{H}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mathbf{X}_i)^T \mathbf{H}^{-1} (\boldsymbol{x} - \mathbf{X}_i)\right)$
- MVN:
  - $f(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{d}{2}} \det(\boldsymbol{\Sigma})^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$
- normal kernel is usually used in a multivariate case

Examples: 2 dim.

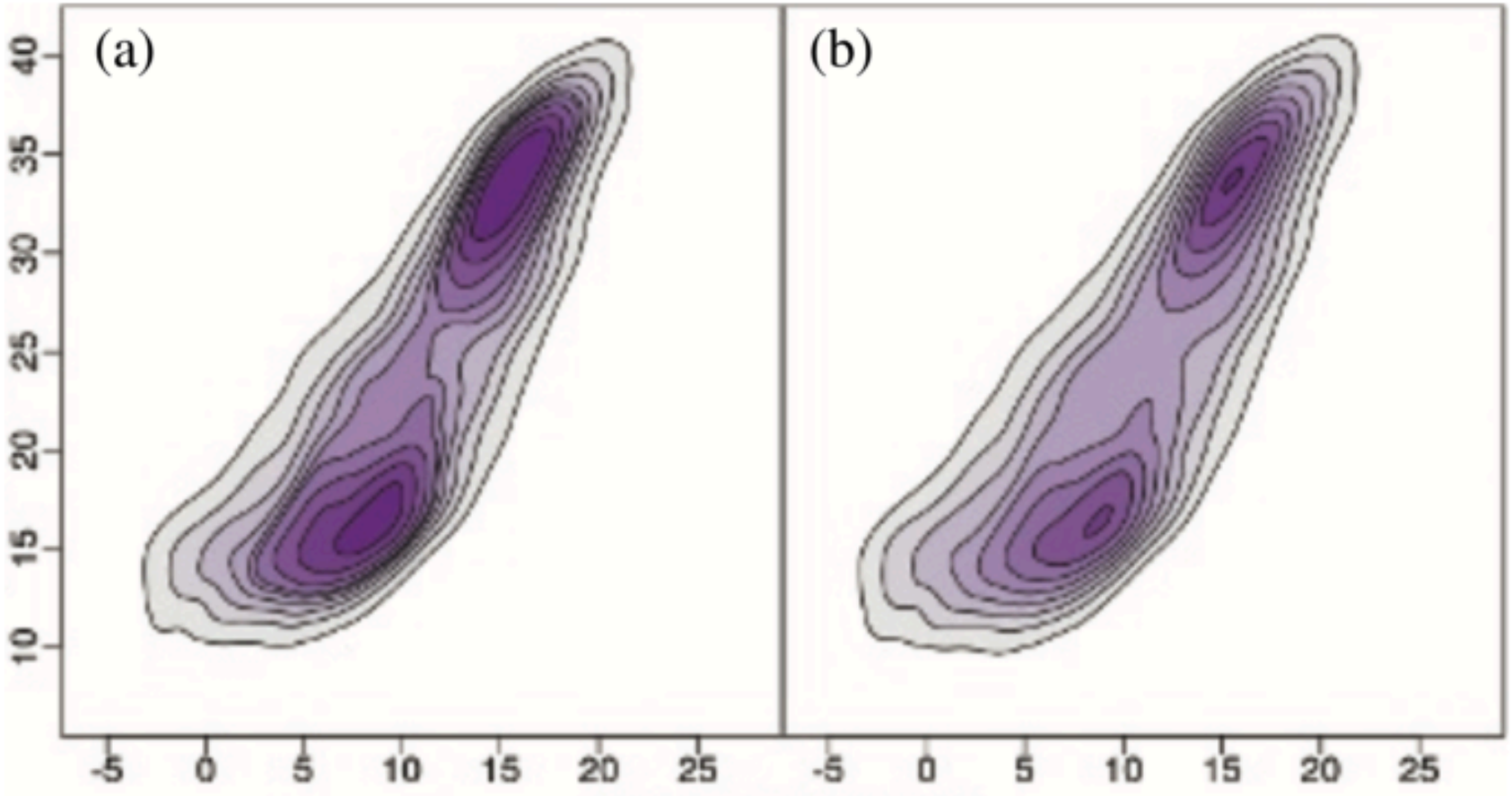


Contour plot with quartile



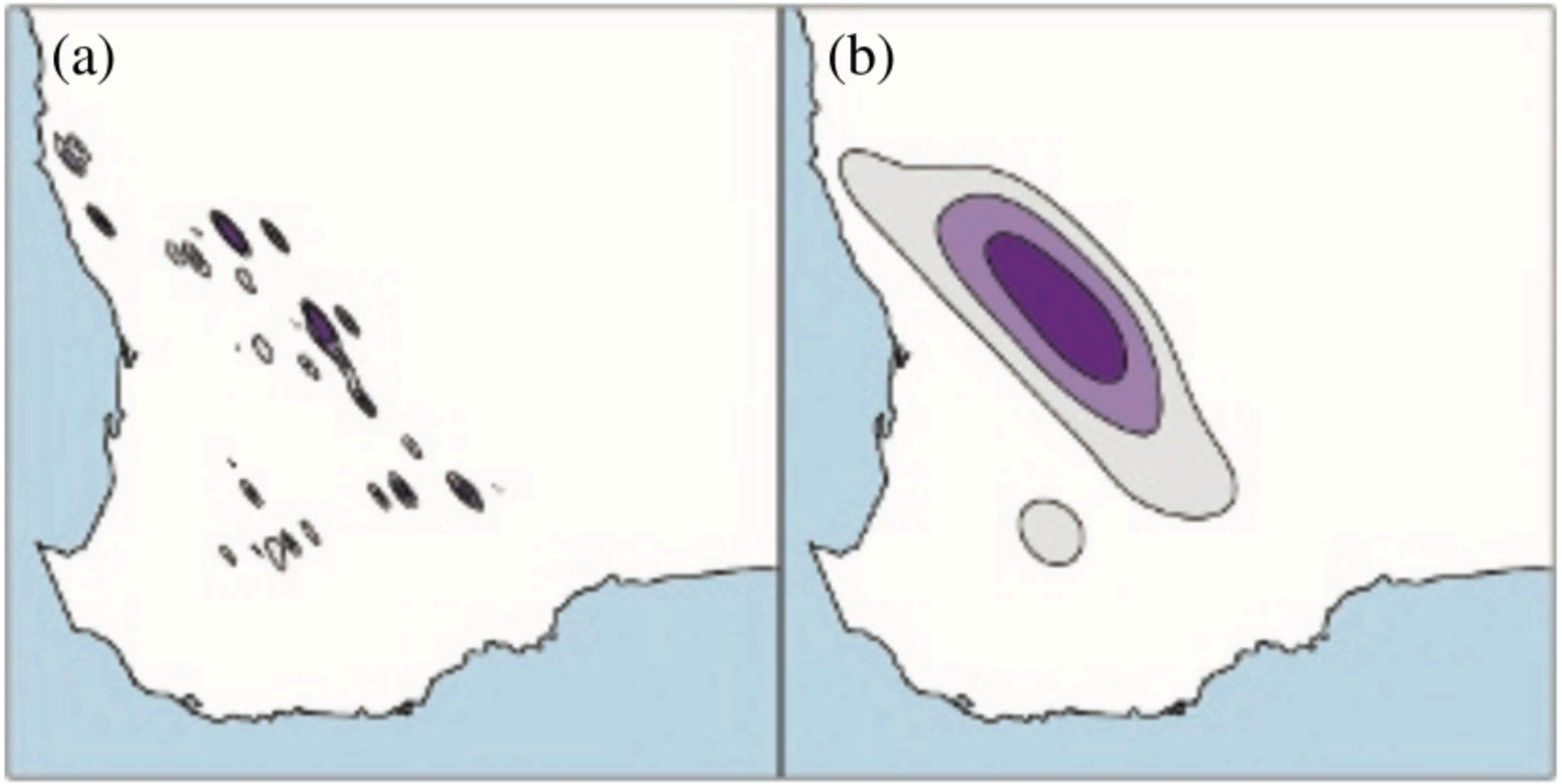
Probability contour levels vs. Linear contour levels

- (a) Probability (decile) contour levels. (b) Linear contour levels.



Selecting an optimal amount of smoothing

- What is an optimal bandwidth?



Squared error analysis

- A notion of an optimal bandwidth requires a discrepancy measure
  - (MSE, local point-wise).
- At any fixed point $\boldsymbol{x}$ (if you knew target density $f(x)$):
  - $$MSE\{\hat{f}(\boldsymbol{x};\mathbf{H})\} = \mathbb{E}\{[\hat{f}(\boldsymbol{x};\mathbf{H}) - f(\boldsymbol{x})]^2\} = \\ = Var\{\hat{f}(\boldsymbol{x};\mathbf{H})\} + Bias^2\{\hat{f}(\boldsymbol{x};\mathbf{H})\}$$

Global performance. What H is the best on average?

- Mean Integrated Squared Error (MISE):
  - $MISE\{\hat{f}(\,\cdot\,;\mathbf{H})\} = \mathbb{E}\int_{\mathbb{R}^d}(\hat{f}(\boldsymbol{x};\mathbf{H}) - f(\boldsymbol{x}))^2 d\boldsymbol{x}$

- MISE is the expected distance between $\hat{f}$ and f
- it describes the performance of the kernel density estimator with respect to a typical sample from the true density
- Goal: minimize MISE **over set of all possible matrices** H

Asymptotic behaviour

- Asymptotic Mean Integrated Squared Error (AMISE), it is equivalent to MISE, as $n \to \infty$
  - minimal rate of MISE: $O(n^{-\frac{4}{d+4}})$
- => `curse of dimensionality'