Chapter 7-1: **Sequential Data**

Jilles Vreeken

Revision 1, November 26th
Definition of smoothing clarified

IRDM '15/16

24 Nov 2015

UNIVERSITÄT DES SAARLANDES
M2CI CLUSTER OF EXCELLENCE
max planck institut informatik

IRDM Chapter 7, overview

- Time Series
  1. Basic Ideas
  2. Prediction
  3. Motif Discovery
- Discrete Sequences
  4. Basic Ideas
  5. Pattern Discovery
  6. Hidden Markov Models

You'll find this covered in
Aggarwal Ch. 3.4, 14, 15

IRDM Chapter 7, today

- Time Series
  1. Basic Ideas
  2. Prediction
  3. Motif Discovery
- Discrete Sequences
  4. Basic Ideas
  5. Pattern Discovery
  6. Hidden Markov Models

You'll find this covered in
Aggarwal Ch. 3.4, 14, 15

Chapter 7.1:
Basic Ideas

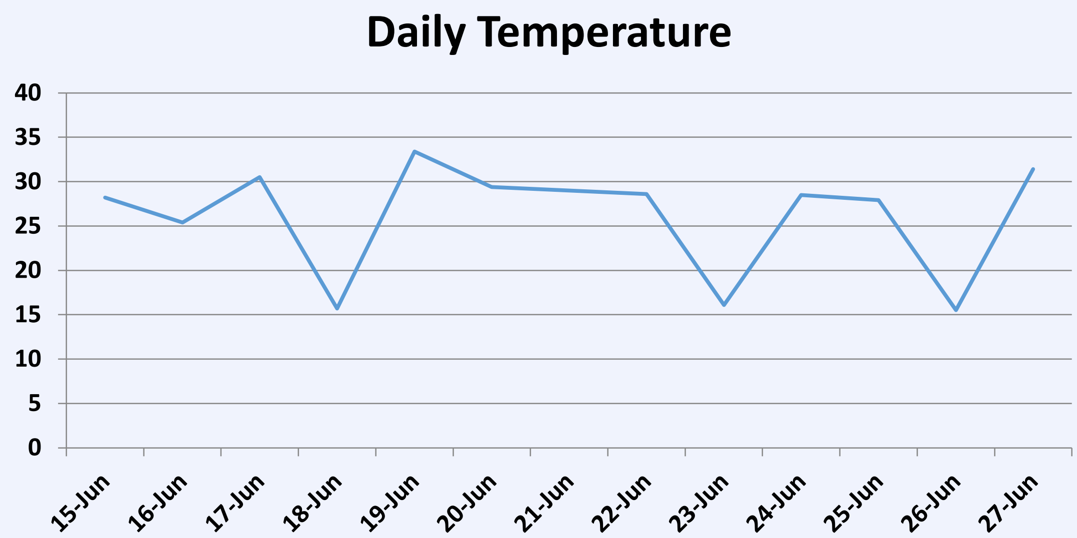
Aggarwal Ch. 14.1-14.2

Temperature Data

Temp (°C)
28.2
25.4
30.5
15.7
33.4
29.4
28.6
16.1
28.5
27.9
15.5
31.4

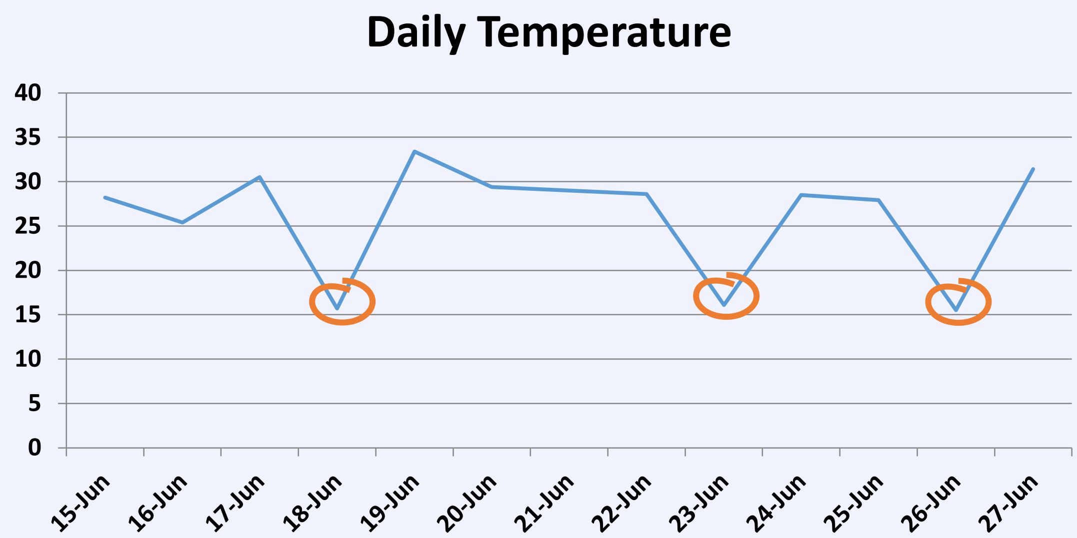
Temperature Data

Time	Temp (°C)
June-15	28.2
June-16	25.4
June-17	30.5
June-18	**15.7**
June-19	33.4
June-20	29.4
June-22	28.6
June-23	**16.1**
June-24	28.5
June-25	27.9
June-26	**15.5**
June-27	31.4



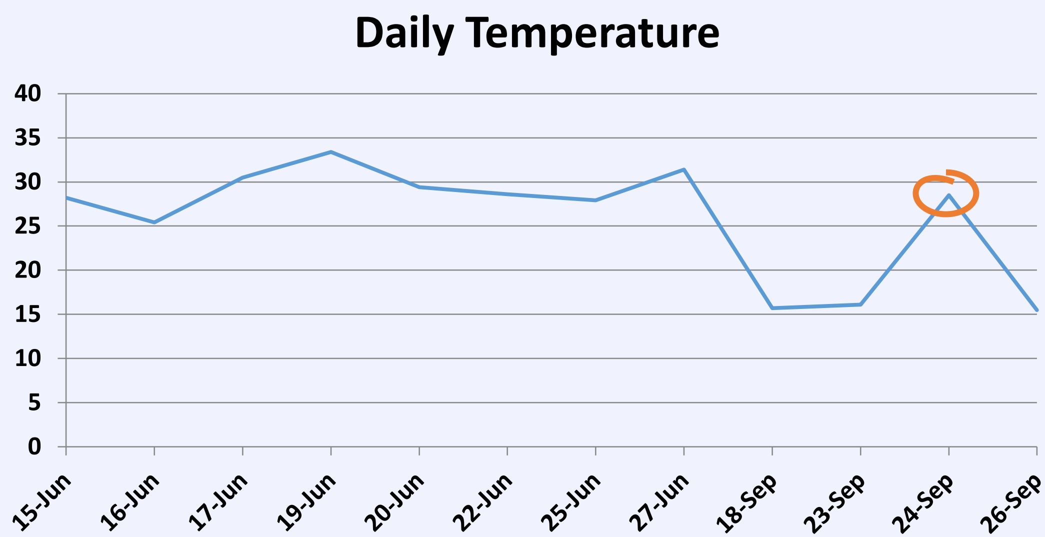
Temperature Data

Time	Temp (°C)
June-15	28.2
June-16	25.4
June-17	30.5
June-18	**15.7**
June-19	33.4
June-20	29.4
June-22	28.6
June-23	**16.1**
June-24	28.5
June-25	27.9
June-26	**15.5**
June-27	31.4

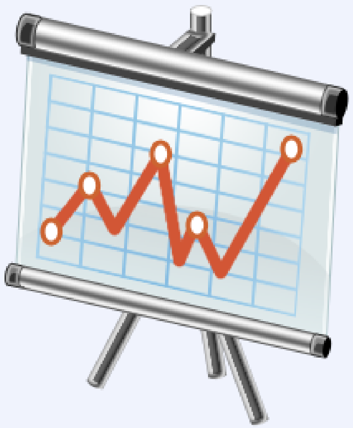


Temperature Data

Time	Temp (°C)
June-15	28.2
June-16	25.4
June-17	30.5
Sept-18	**15.7**
June-19	33.4
June-20	29.4
June-22	28.6
Sept-23	**16.1**
Sept-24	**28.5**
June-25	27.9
Sept-26	**15.5**
June-27	31.4

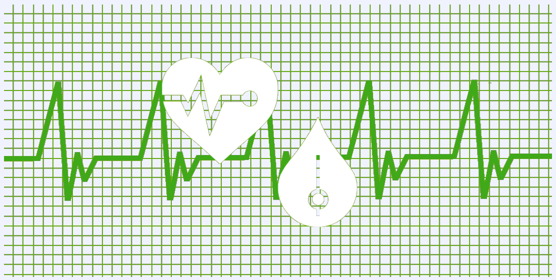


Applications

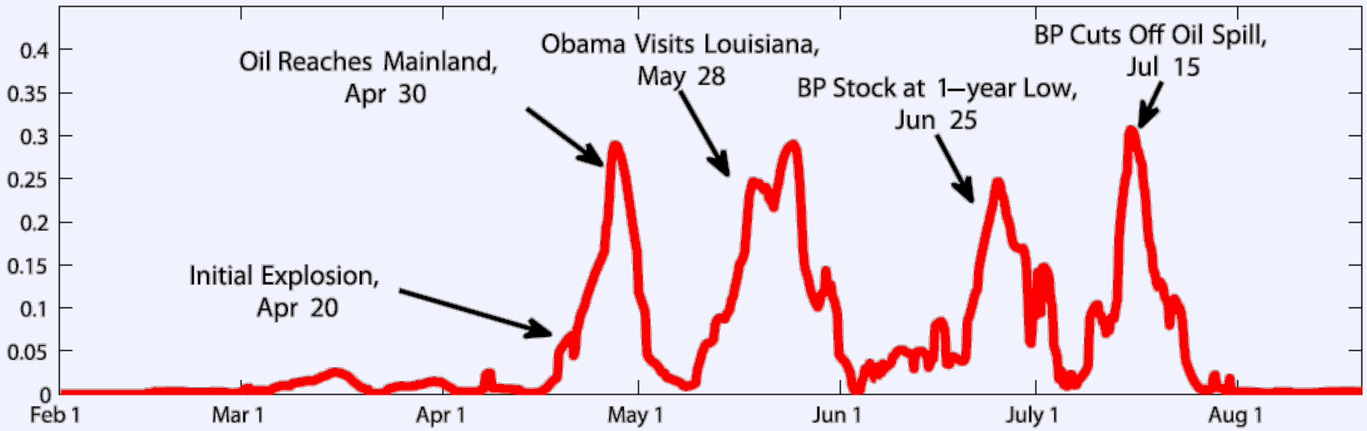


Stock analysis

Weather Forecasting



Health Monitoring



Social Network Analysis

Definition

A **time series** of **length** n consists of n tuples $(t_1, X_1), (t_2, X_2), \ldots (t_n, X_n)$ where for a tuple (t_i, X_i), t_i is the **time stamp**, and X_i is the **data** at time t_i, and we have a total order on the time stamps $t_1 < t_2 < \cdots < t_n$

Length
- may either be finite or infinite

Time stamps
- may be contiguous, in practice integers are easier

Data
- when talking about time series, usually numeric, **continuous real-valued**
- may be univariate (one attribute) or multivariate (multiple attributes)

Probabilistic Model of Time Series

Consider data X_i at time t_i as a random variable

- the actual data we observe at t_i is a **realization** of X_i

Some probabilistic properties can be **stable** over time

- e.g. the mean μ_i of X_i does not change (much)
- the covariance between pairs (X_i, X_{i+h}) is (almost) the same as (X_1, X_{1+h}), i.e., the **autocovariance** of X_i does not change (much)

A time series is **stationary** if the process behind it **does not change**

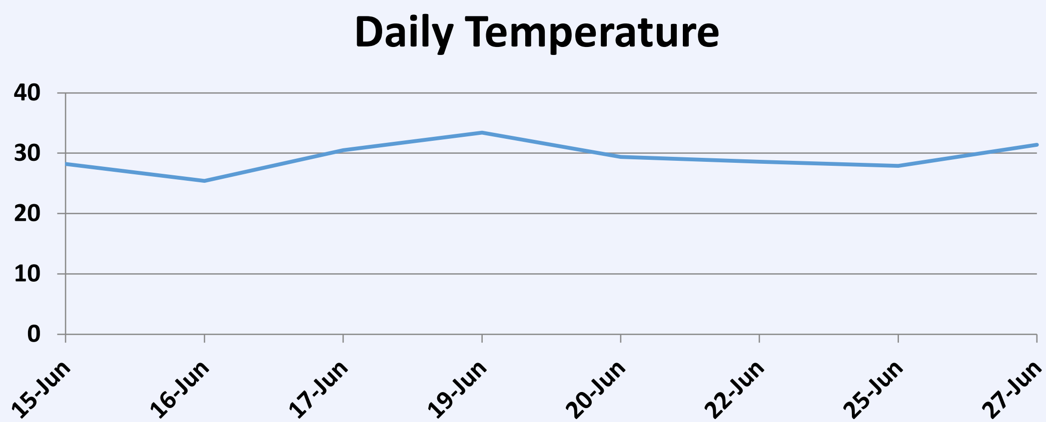
- $\mu_t = \mu_s = \mu$ for all t, s, and
- $C_{XX}(t,s) = C_{XX}(s-t) = C_{XX}(\tau)$ where $\tau = |s-t|$ is the amount of time by which the signal is shifted

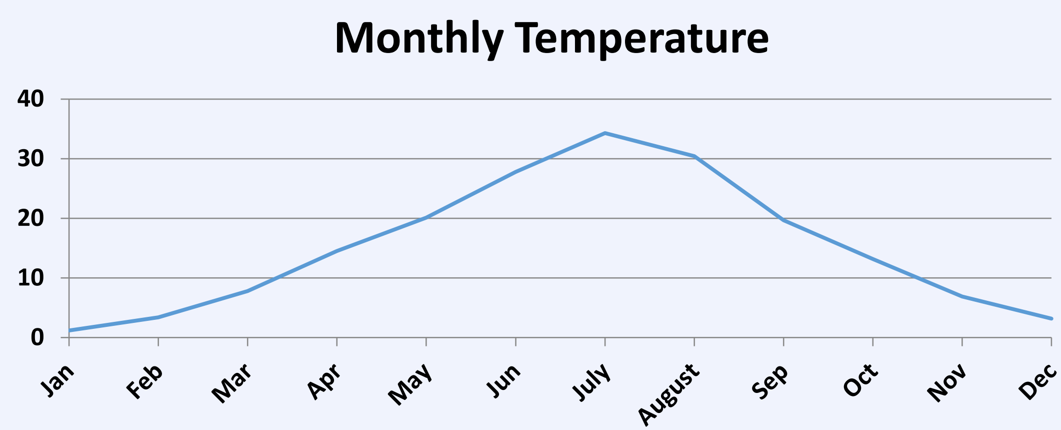
Stationary time series are easy to model and predict

- most real-world time series, however, are anything but stationary

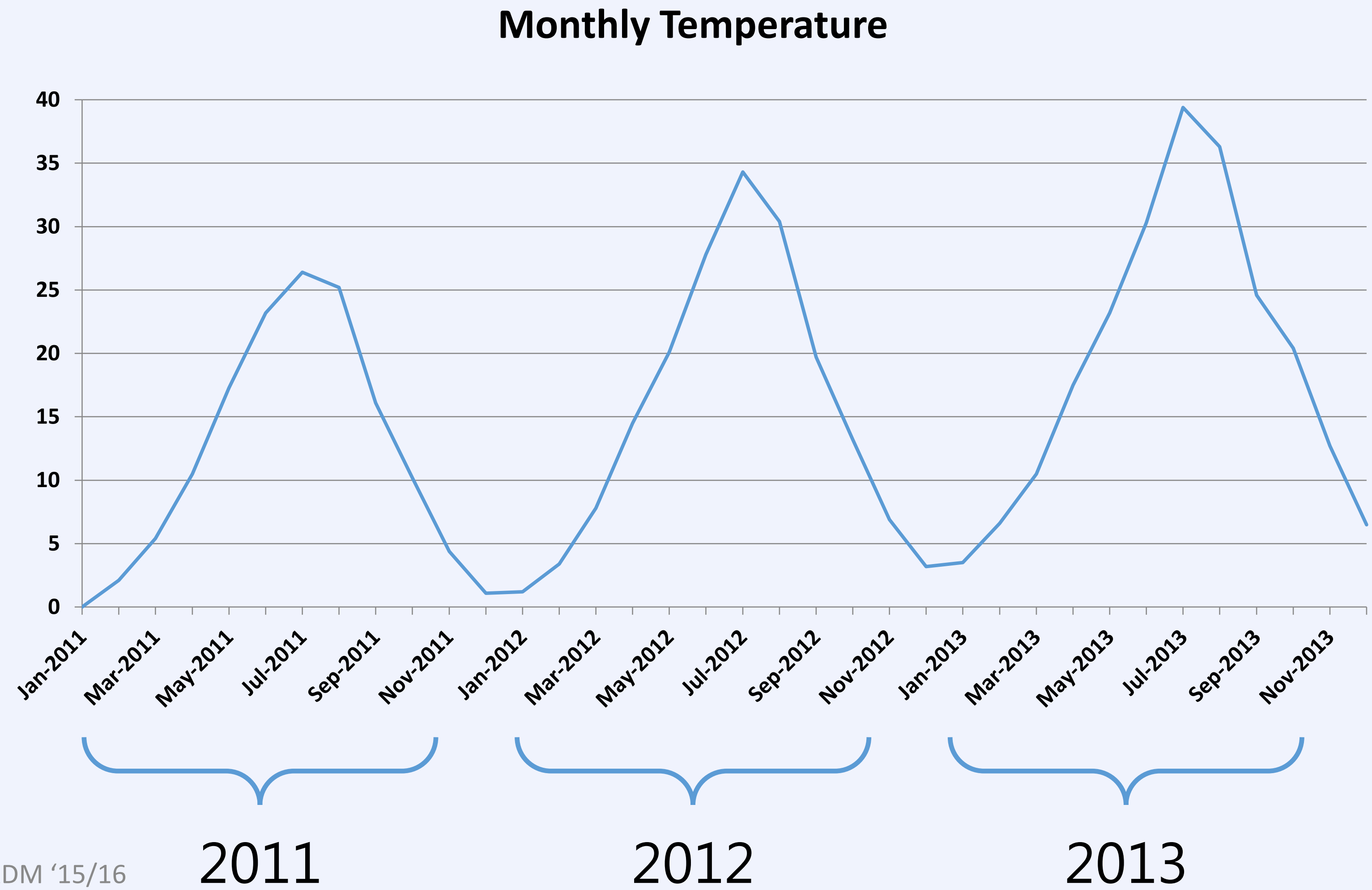
(recall, if X_i has mean $\mu_i = E[X_i]$, $C_{XX}(t,s) = cov(X_t, X_s) = E[X_t X_s] - \mu_t \mu_s$)

Stationarity of Time Series





Seasonality & trend



Formulation

Classically, we assume a time series X is composed of

$$X_i = seasonality_i + \ trend_i + noise_i$$

where $noise_i$ is stationary.

To make X stationary, we simply have to remove seasonality and trend.

Seasonality

Seasonality is essentially **periodicity**

- seasonality is a **periodic function** of time with period d

$$seasonality_i = seasonality_{i-d}$$

How to find the **seasonality function**?

1. by fitting a **sine** or **cosine** function

   difficult – the signal may also be sine'ish

2. by **differencing**

$$X_i = seasonality_i + trend_i + noise_i$$
$$X_{i-d} = seasonality_{i-d} + trend_{i-d} + noise_{i-d}$$

Seasonality

Seasonality is essentially periodicity

- seasonality is a periodic function of time with period d

$$seasonality_i = seasonality_{i-d}$$

How to find the seasonality function?

1. by fitting a sine or cosine function

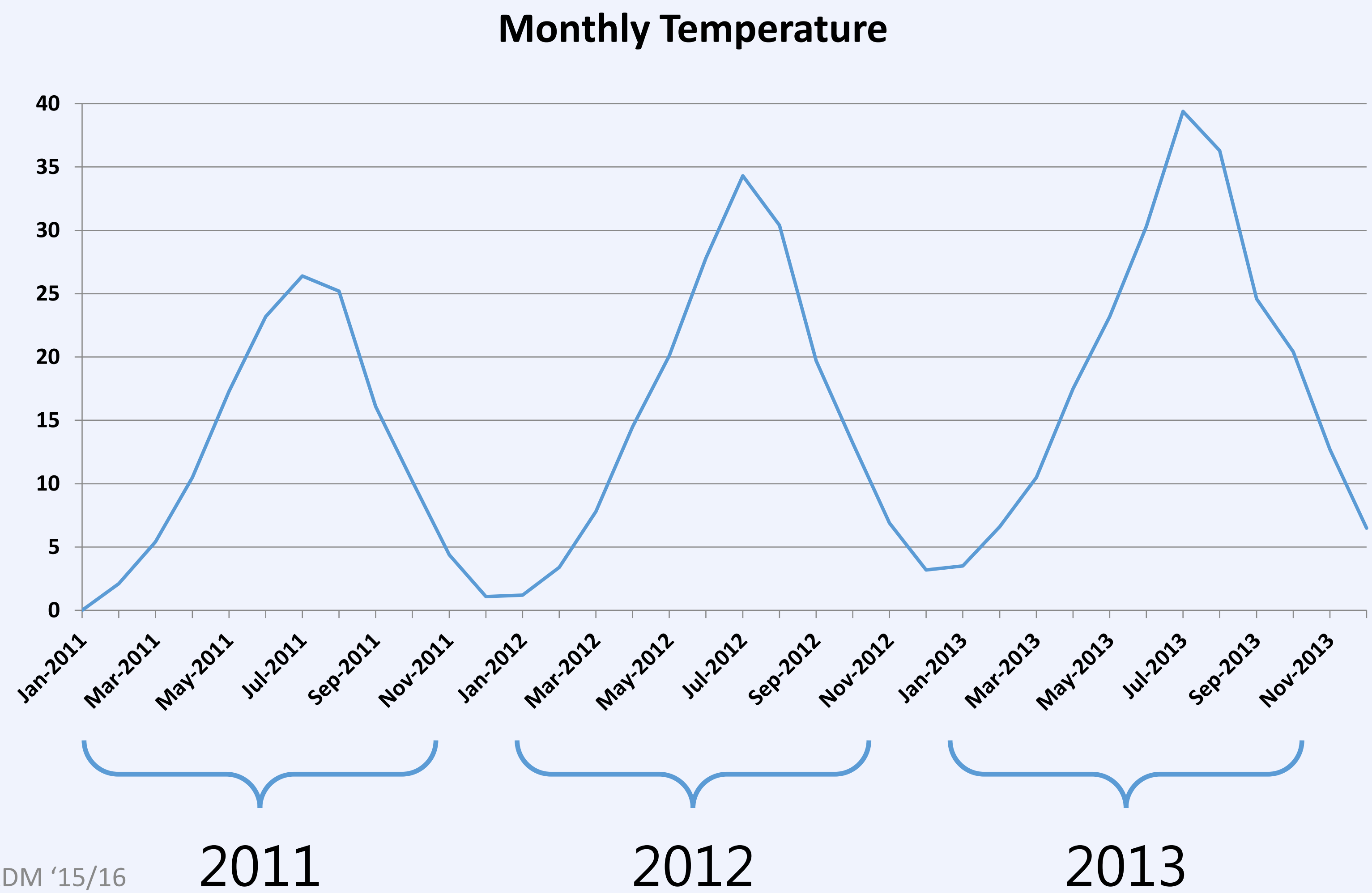
   difficult – the signal may also be sine'ish

2. by differencing

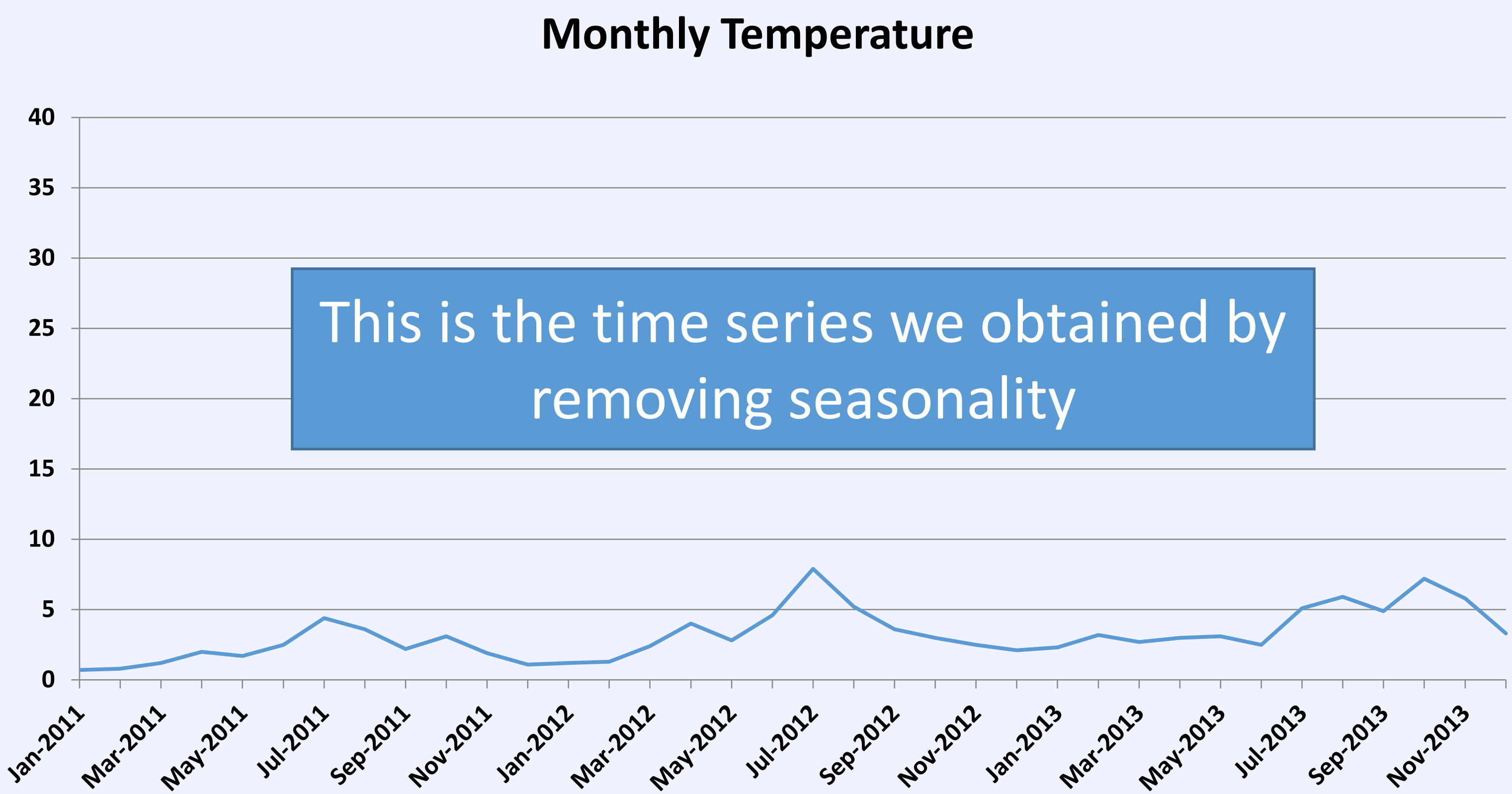
$$X_i = \cancel{seasonality_i} + trend_i + noise_i$$
$$X_{i-d} = \cancel{seasonality_{i-d}} + trend_{i-d} + noise_{i-d}$$

$$X_i' = X_i - X_{i-d}$$

$$X_i' = X_i - X_{i-d} \text{ where } d = 12$$



Example: Removing Seasonality



Trend

Trend is a **polynomial function** of time (assumption)

How to find the trend function?

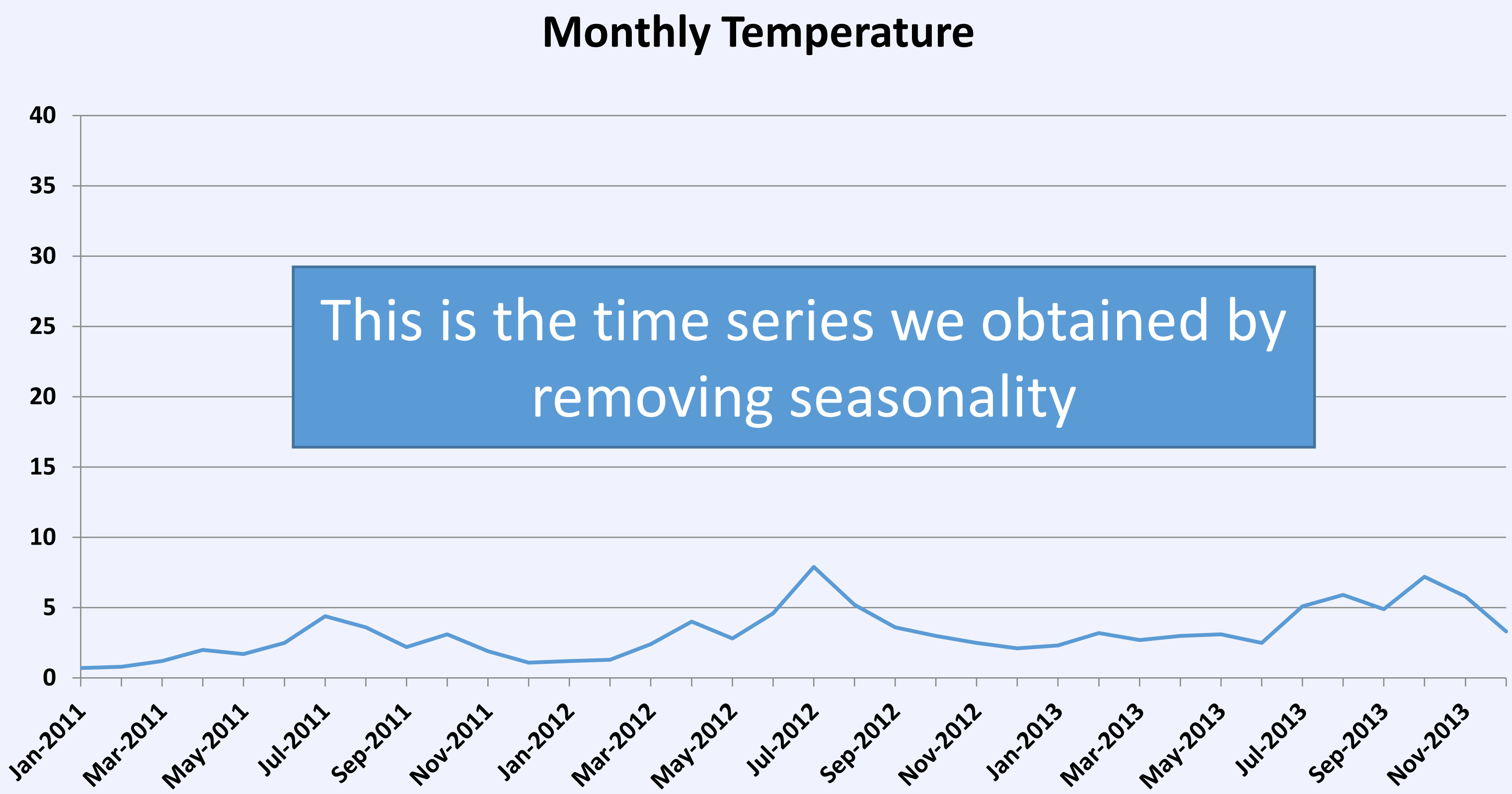
1. by **fitting functions**
   - difficult to do, up to what order, when to stop?

2. by **differencing**

$$X_i' = X_i - X_{i-1}$$
$$X_i'' = X_i' - X_{i-1}'$$

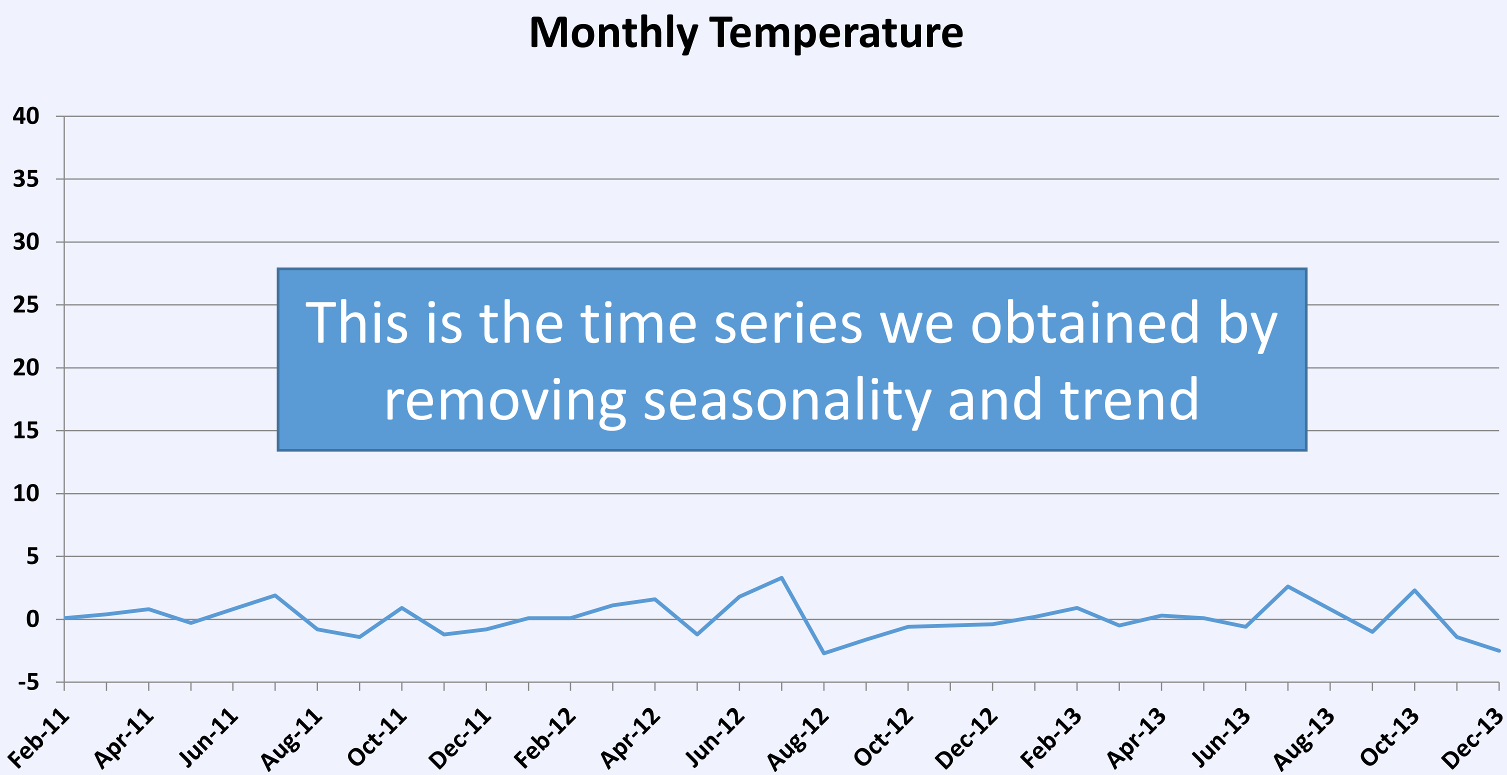
   - usually 2 times is enough

Example: Removing Trend



Example: Removing Trend

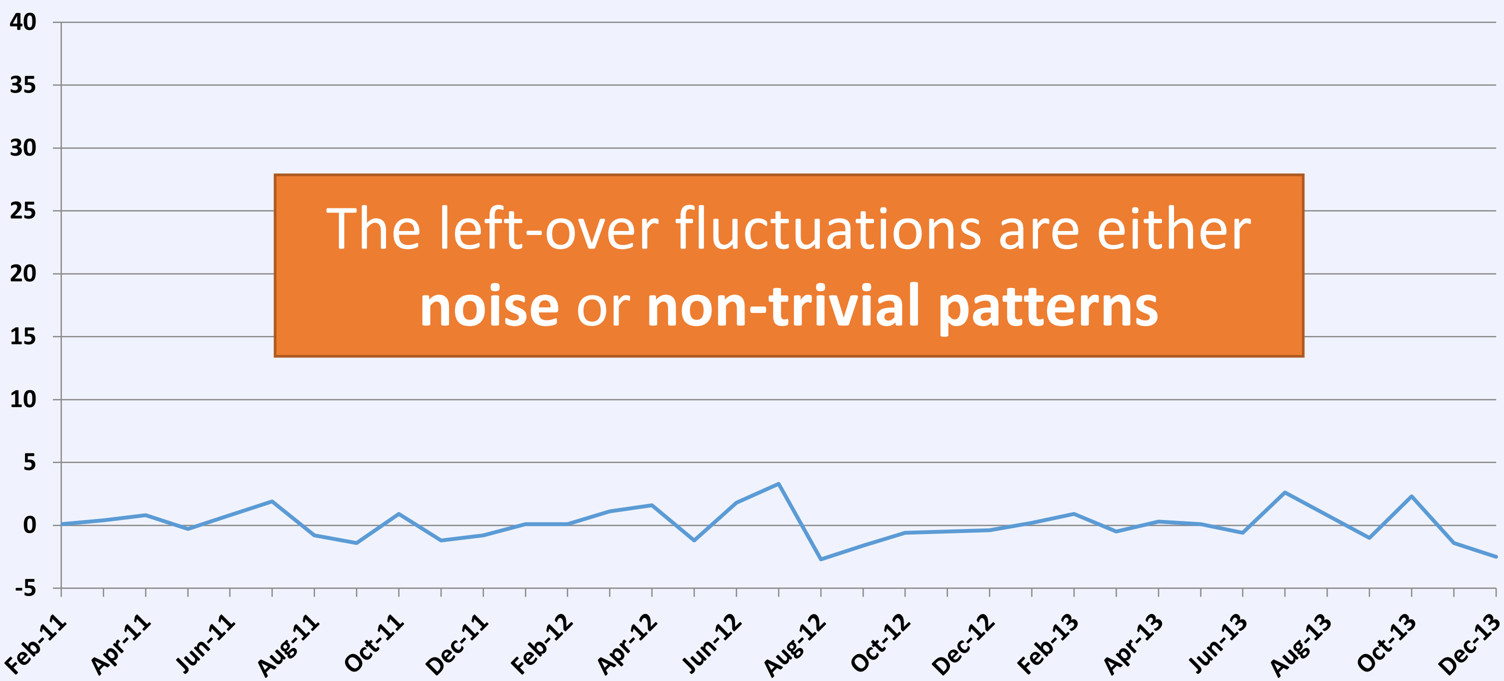
$$X_i' = X_i - X_{i-1}$$



Example: Removing Trend

$$X'_i = X_i - X_{i-1}$$

Monthly Temperature



The left-over fluctuations are either **noise** or **non-trivial patterns**

Pre-processing

We can infer missing values by interpolation

$$X_k = X_i + \left(\frac{t_k - t_i}{t_j - t_i}\right) \times (X_j - X_i)$$

where $t_i < t_k < t_j$

Pre-processing

We can infer missing values by interpolation

$$X_k = X_i + \left(\frac{t_k - t_i}{t_j - t_i}\right) \times (X_j - X_i)$$

where $t_i < t_k < t_j$

	Time	Temp (°C)
1	June-19	33.4
2	June-20	29.4
4	June-22	
5	June-23	16.1

Temperature on June-22:

$$X_4 = X_2 + \left(\frac{t_4 - t_2}{t_5 - t_2}\right) \times (X_5 - X_2)$$
$$= 29.4 + \left(\frac{4-2}{5-2}\right) \times (16.1 - 29.4)$$
$$= 20.5$$

Smoothing

We can remove noise by **smoothing**

Standard options include **averaging**

$$X_i' = avg(X_{i-w}, \dots, X_i)$$

where **window length** w is a user-specified parameter

We can more weight to recent values by **exponential smoothing**

$$X_i' = (1-\alpha)^i \cdot X_0' + \alpha \sum_{j=1}^{i} X_j \cdot (1-\alpha)^{i-j}$$

where the user chooses decay factor α

(updated on Nov 26th : we now average explicitly over past values)

Chapter 7.2: Forecasting

Aggarwal Ch. 14.3

Principle of Forecasting

If we wish to make predictions, then clearly we must **assume** that something is **stable** over time.

Autoregressive (AR) model

Future values depend on **past values** + random noise

- assumption: the time series depends on **autocorrelation**

Which past values?

- the w **immediately** previous values

What relation between past and future?

- linear combination

What kind of noise?

- Gaussian

AR, formally

Future value is
a linear combination of **past values** + white noise

$$X_t = \underbrace{\sum_{i=1}^{w} a_i \cdot X_{t-i}}_{\text{Linear combination of past values}} + \underbrace{c + \epsilon_t}_{\text{noise with shifted mean}}$$

where $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$

Least-square regression

$$\epsilon_t = \underbrace{X_t}_{\text{actual value}} - \underbrace{(a_1 \cdot X_{t-1} + a_2 \cdot X_{t-2} + \cdots + a_w \cdot X_{t-w} + c)}_{\text{predicted value}}$$

the prediction error is simply the Gaussian noise in the AR model, the smaller we can get this value, the better!

Given data $\boldsymbol{D}$ of N training instances, we want to find $a_1, \dots, a_w$ and c that minimise the **mean squared error**

$$\frac{1}{N-w} \sum_{t=w+1}^{N} \epsilon_t^2$$

Solving AR

Find $a_1, \dots, a_w$ and c that **minimize** $\frac{1}{N-w} \sum_{t=w+1}^{N} \epsilon_t^2$

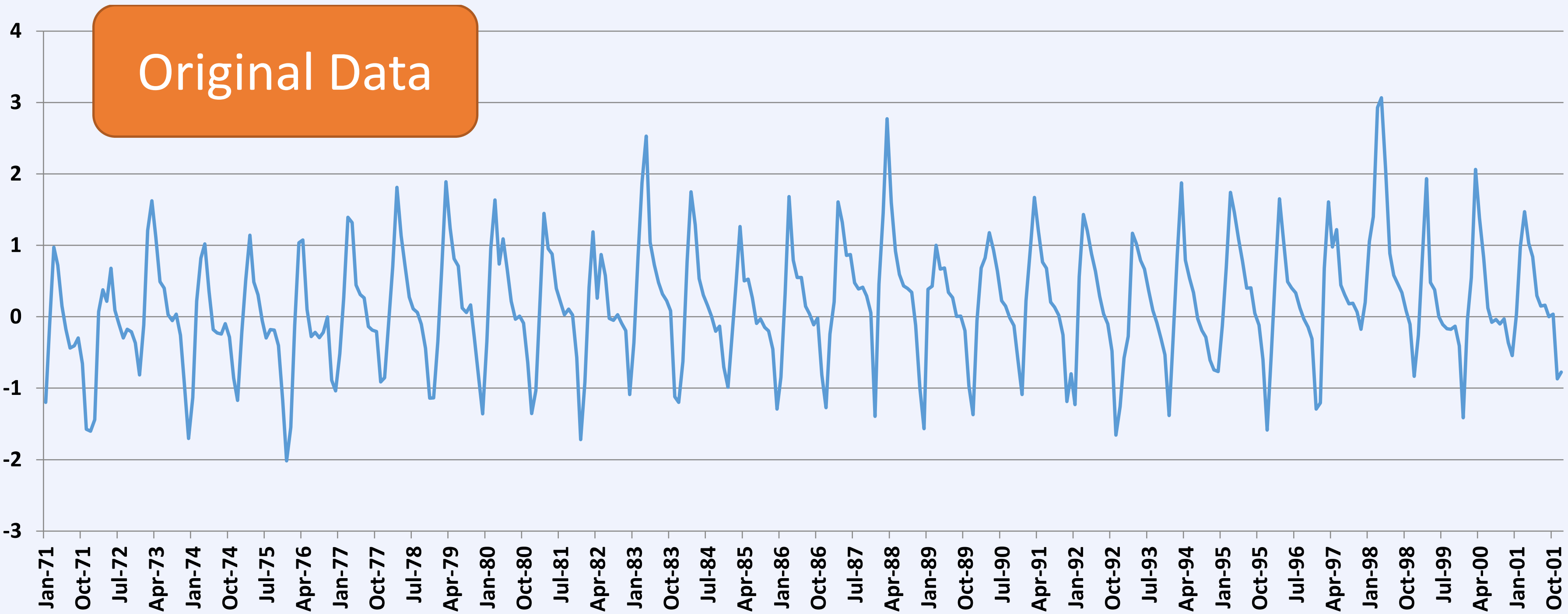
There are different solving strategies available

- ordinary least squares, assumes ϵ_t and X_t are uncorrelated
- generalized least squares, assumes correlation exists but is known
- iteratively reweighted least squares, assumes correlation is unknown

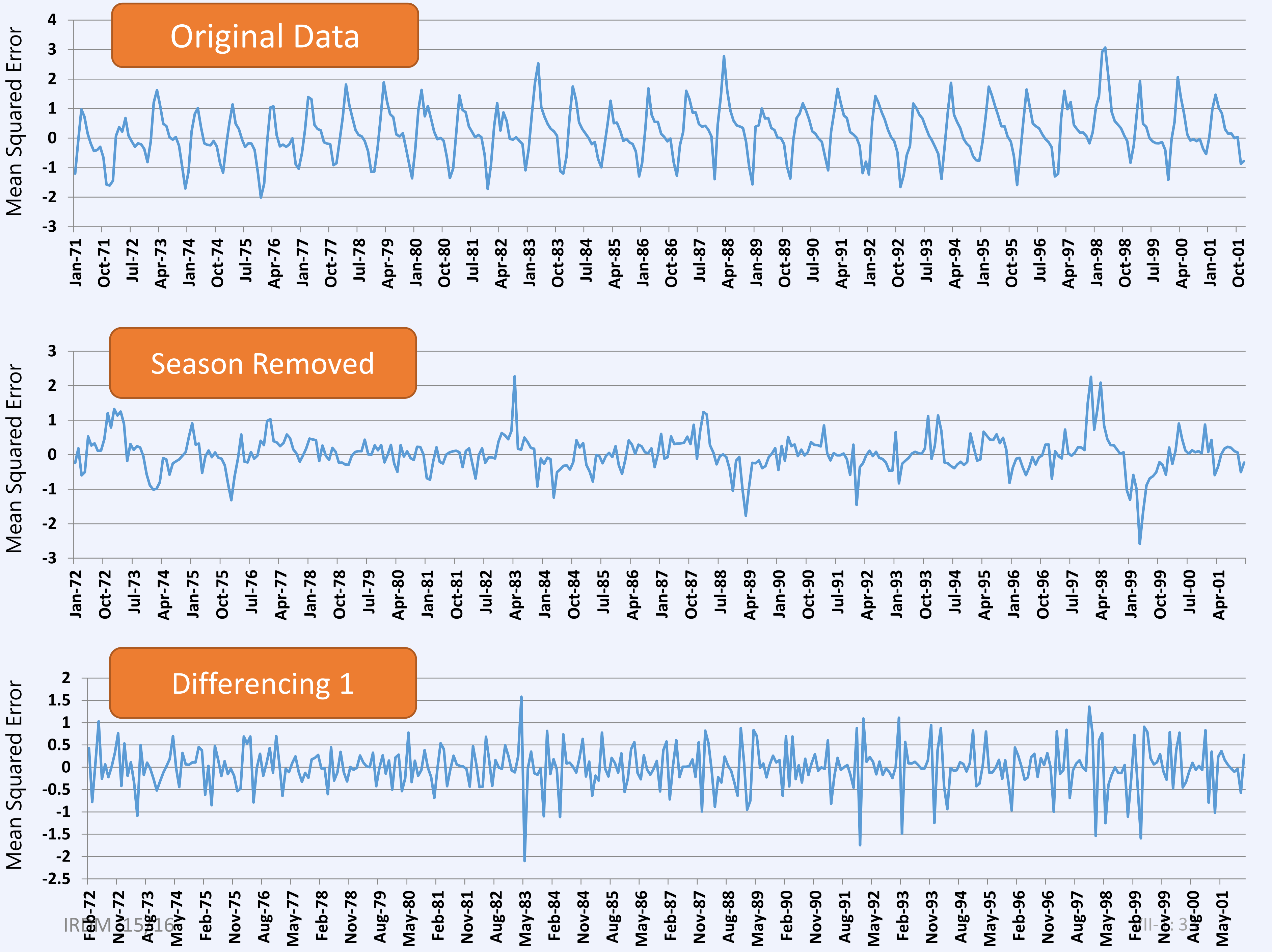
Many standard tools available to do AR

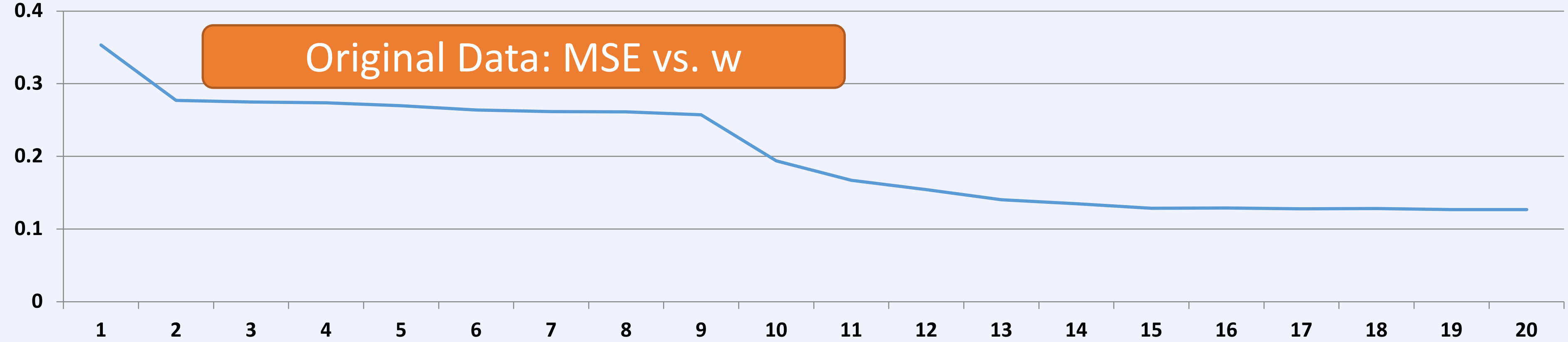
- MATLAB: **ar** function
- R: **arima** function

Example: AR

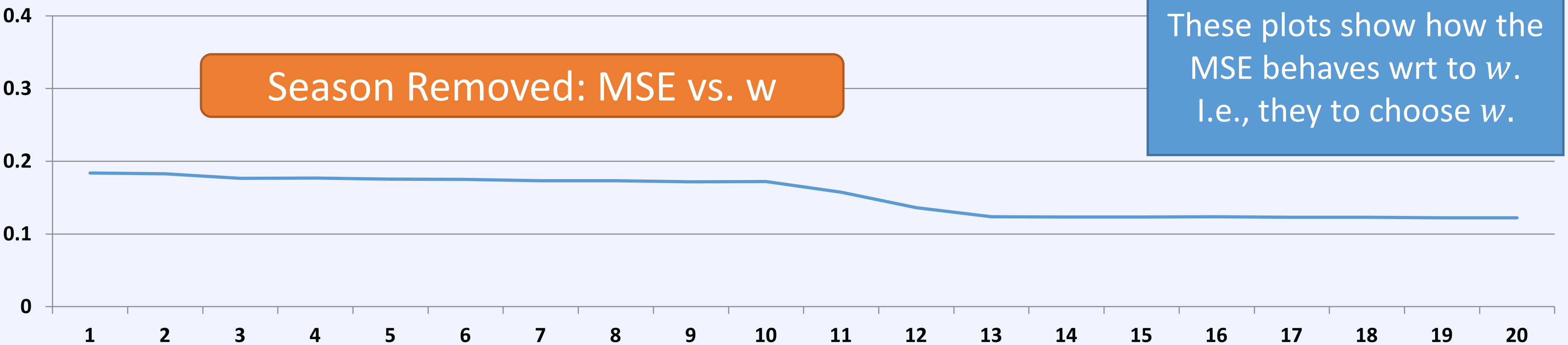


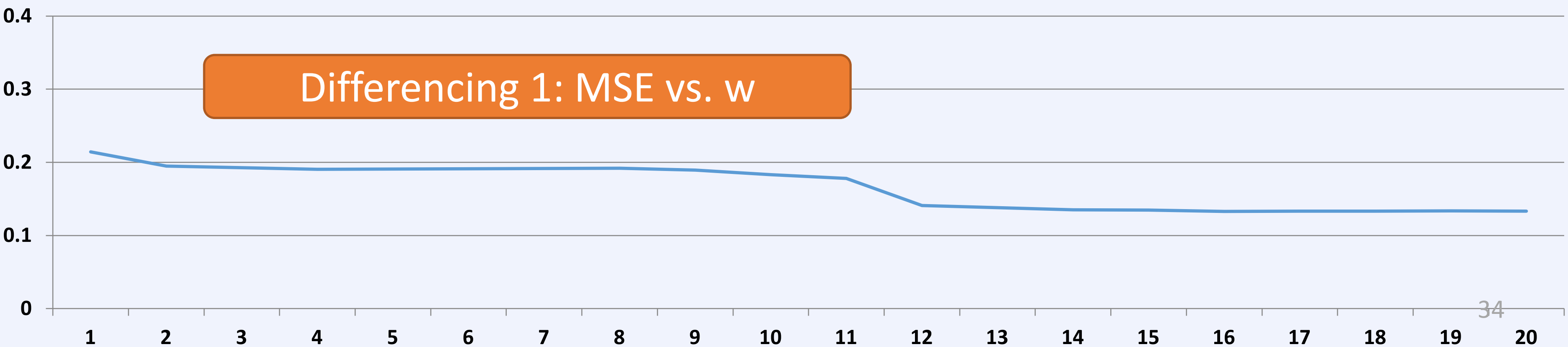
Monthly temperature measured above the ground in a province of Vietnam from 1971 to 2001


Original Data
Mean Squared Error
4
3
2
1
0
-1
-2
-3
Jan-71 Oct-71 Jul-72 Apr-73 Jan-74 Oct-74 Jul-75 Apr-76 Jan-77 Oct-77 Jul-78 Apr-79 Jan-80 Oct-80 Jul-81 Apr-82 Jan-83 Oct-83 Jul-84 Apr-85 Jan-86 Oct-86 Jul-87 Apr-88 Jan-89 Oct-89 Jul-90 Apr-91 Jan-92 Oct-92 Jul-93 Apr-94 Jan-95 Oct-95 Jul-96 Apr-97 Jan-98 Oct-98 Jul-99 Apr-00 Jan-01 Oct-01
Season Removed
Mean Squared Error
3
2
1
0
-1
-2
-3
Jan-72 Oct-72 Jul-73 Apr-74 Jan-75 Oct-75 Jul-76 Apr-77 Jan-78 Oct-78 Jul-79 Apr-80 Jan-81 Oct-81 Jul-82 Apr-83 Jan-84 Oct-84 Jul-85 Apr-86 Jan-87 Oct-87 Jul-88 Apr-89 Jan-90 Oct-90 Jul-91 Apr-92 Jan-93 Oct-93 Jul-94 Apr-95 Jan-96 Oct-96 Jul-97 Apr-98 Jan-99 Oct-99 Jul-00 Apr-01
Differencing 1
Mean Squared Error
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
Feb-72 Nov-72 Aug-73 May-74 Feb-75 Nov-75 Aug-76 May-77 Feb-78 Nov-78 Aug-79 May-80 Feb-81 Nov-81 Aug-82 May-83 Feb-84 Nov-84 Aug-85 May-86 Feb-87 Nov-87 Aug-88 May-89 Feb-90 Nov-90 Aug-91 May-92 Feb-93 Nov-93 Aug-94 May-95 Feb-96 Nov-96 Aug-97 May-98 Feb-99 Nov-99 Aug-00 May-01



These plots show how the MSE behaves wrt to w. I.e., they to choose w.





Moving Average (MA) model

Future values depend on deterministic factor + **noise**

- assumption: the time series depends on **history of shocks**

What deterministic factor?

- the **mean** of the time series

Noise over what past values?

- the current value and q **immediately** previous values

What kind of noise?

- Gaussian

MA, formally

The $MA(q)$ is defined as

$$X_t = \mu + \epsilon_t + \underbrace{\sum_{i=1}^{q} b_i \cdot \epsilon_{t-i}}_{\text{past noise}}$$

mean

current noise

where $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$

Recall, for the $AR(w)$ model we had

$$X_t = c + \epsilon_t + \sum_{i=1}^{w} a_i \cdot X_{t-i}$$

Solving MA

Find those $b_1, \dots, b_q$ that **minimize** the error

Unlike for AR, this problem is not linear

- to identify noise terms, we need to know $b_1, \dots, b_q$
- to identify $b_1, \dots, b_q$, we need to know the noise terms
- typically we use an iterative non-linear fitting approach, instead of linear least-squares

The ARMA model

ARMA combines the **AR** model with the **MA** model

Future values depend on **past values** + **history of noise**

- the time series depends on
  both **autocorrelation** and **history of shocks**

The ARMA model has two parameters, w and q

- window length w for autocorrelation
- history length q for noise

What kind of noise?

- Gaussian

ARMA, formally

ARMA combines the **AR** model with the **MA** model

Autoregressive model, $AR(w)$:

$$X_t = \boxed{c + \epsilon_t + \sum_{i=1}^{w} a_i \cdot X_{t-i}}$$

Moving Average model, $MA(q)$

$$X_t = \mu + \epsilon_t + \boxed{\sum_{i=1}^{q} b_i \cdot \epsilon_{t-i}}$$

Autoregressive Moving Average model, $ARMA(w, q)$

$$X_t = \boxed{c + \epsilon_t + \sum_{i=1}^{w} a_i \cdot X_{t-i}} + \boxed{\sum_{i=1}^{q} b_i \cdot \epsilon_{t-i}}$$

Solving ARMA

Find those a_i and b_i and c that **minimize** the error

We need **non-linear least-square regression**

- many standard tools to do this
- MATLAB and R implement ARMA as ‘arma’ resp. ‘arima’

How to set w and q?

- as small as possible, so that the model still fits the data well
- aka, good luck

Chapter 7.3:
Motif Discovery

Aggarwal Ch. 14.4, 3.4

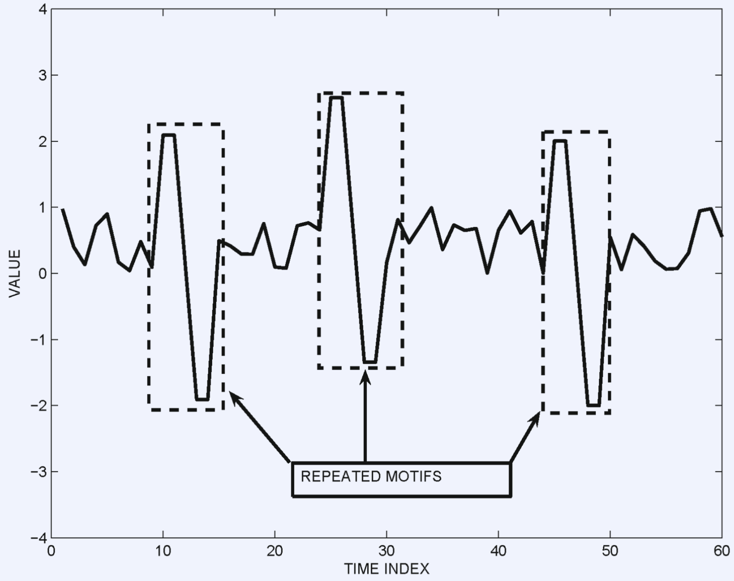
Motifs

A **motif** is a shape that frequently repeats in a time series

- shape can also be called 'pattern'

Many variations of **motif discovery** exist

- contiguous versus non-contingous shapes
- low versus high granularities
- single time series versus databases of time series



What is a motif?

When does a motif belong to a time series?

- there are two main methods for deciding

1. **distance-based support**
   A **segment** $X[i,j]$ of a sequence X is said to **support** a motif Y when the distance $d(X[i,j],Y)$ between the segment and the motif is below some threshold ϵ.

2. **discrete-matching based support**
   first we discretise time series X into a discrete sequence s. A motif is now a (frequent) subsequence of s.

Distance-based motifs, formally

A motif, a sequence $S = S_1, \ldots, S_w$ of real values, is said to **approximately match** a contiguous subsequence of length w in time series X, if the distance between $(S_1, \ldots, S_w)$ and $(X_i, \ldots, X_{i+w-1})$ is at most ϵ.

- commonly, Euclidean distance or Dynamic Time Warping

The frequency of a motif is its number of occurrences

- the number of matches of a motif $S = S_1, \ldots, S_w$ to the time series $X_1, \ldots, X_n$ at threshold ϵ is equal to the number of windows of length w in X for which the distance is at most ϵ

Top-k motifs

Nobody wants all motifs

- lots of many ϵ-similar matches for even a single true occurrence
- instead, we aim for the top-k best motifs

As with frequent itemset mining, redundancy is an issue

- we need to keep the top-k diverse
- distances between any pair of motifs must be at least $2 \cdot \epsilon$

FINDBESTMOTIF(X, w, ϵ)

begin
  for $i = 1$ to $n - w + 1$ **do begin**
    $Candidate = (X_i, \ldots, X_{i+w-1})$
    for $j = 1$ to $n - w + 1$ **do begin**
      $CompareTo = (X_j, \ldots, X_{j+w-1})$
      $d = distance(Candidate, CompareTo)$
      if $d < \epsilon$ **and** (non-trivial-match)
        then increment support count of $Candidate$
    endfor
    if $Candidate$ has the highest count found so far
      then update $BestCandidate$
  endfor
  return $BestCandidate$
end

(trivially expanded to top-k)

Computational Complexity

Finding the best motif takes $O(n^2)$ distance computations

Practical complexity largely depends on distance function

- Euclidean distance is fast
- Dynamic Time Warping is often better, but much slower

Lower bounds are our friend

- if the lower bound on the distance between a motif and a windows is greater than ϵ, the window will never support the motif
- piecewise-aggregate approximations (PAA) allow fast computation of lower bounds by considering simplified (compressed) time series

Conclusions

Prediction over time is one of the most important and most used data analysis problems – **predictive analytics**

There exist two main types of sequential data

- continuous real-valued **time series** and discrete **event sequences**
- for both specialised algorithms exist

In practice, despite many assumptions **ARMA** is powerful

- often used in industry, learn how to use it, learn when to use it

Patterns in time series are called **motifs**

- by choosing a distance function can be mined directly from time series

Thank you!

Prediction over time is one of the most important and most used data analysis problems – **predictive analytics**

There exist two main types of sequential data
- continuous real-valued **time series** and discrete **event sequences**
- for both specialised algorithms exist

In practice, despite many assumptions **ARMA** is powerful
- often used in industry, learn how to use it, learn when to use it

Patterns in time series are called **motifs**
- by choosing a distance function can be mined directly from time series