Chapter 10: Time Series Analysis

Prachya Boonkwan (Arm)

School of ICT, SIIT,
Thammasat University

prachya@siit.tu.ac.th, kaamanita@gmail.com

https://tinyurl.com/5b6ch5px

Who? Me?

- Nickname: **Arm** (P'/N' Arm, etc.)
- Born: Aug 1981
- Work
  - Researcher at NECTEC 2005-2024
  - Lecturer at SIIT, Thammasat University 2025-now
- Education
  - B.Eng & M.Eng, CPE Kasetsart University
  - Obtained Ministry of Science Scholarship in early 2008
  - Did a PhD in Informatics (AI & Computational Linguistics) at University of Edinburgh, UK from 2008 to 2013 (4.5 years)

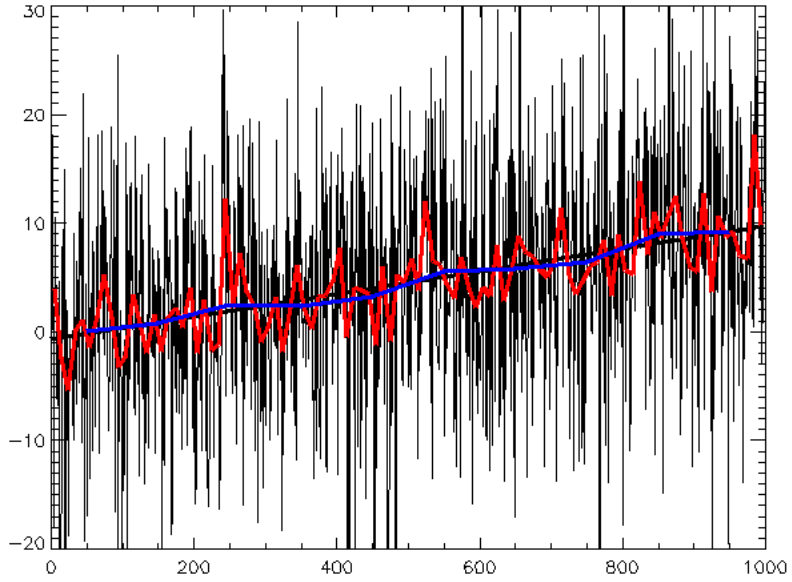
Outline

- Introduction
- Time-domain methods
  - Autoregressive integrated moving average (ARIMA)
  - Convolutional analysis and CNNs
- Frequency-domain methods
  - Spectral density analysis
  - Wavelet analysis
- Transformer for time series
- Conclusion

1. Introduction

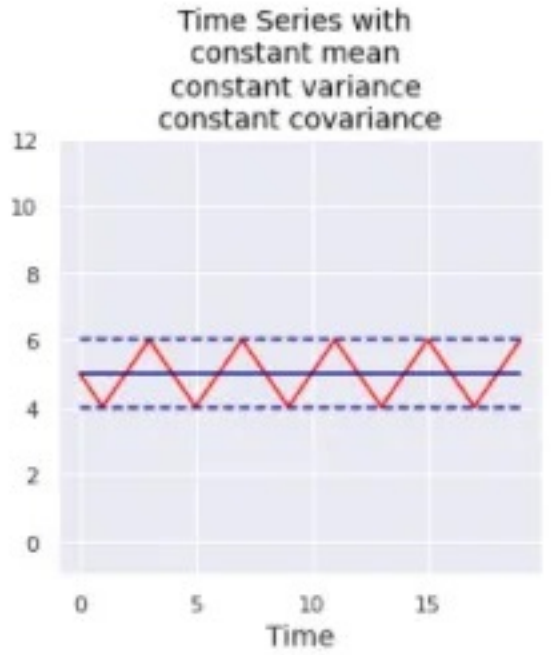
Time Series

- A sequence of data points read at equally spaced points of time
  - E.g. daily stock price, monthly rice price, heights of ocean tides, audio signals, counts of celestial meteriorites, and activity of tectonic plates
- Time series forecasting
  - Predicting future values based on previously observed values
  - **Stochastic process:** observations close together in time are more closely related than those further apart



Stationary vs. Non-Stationary

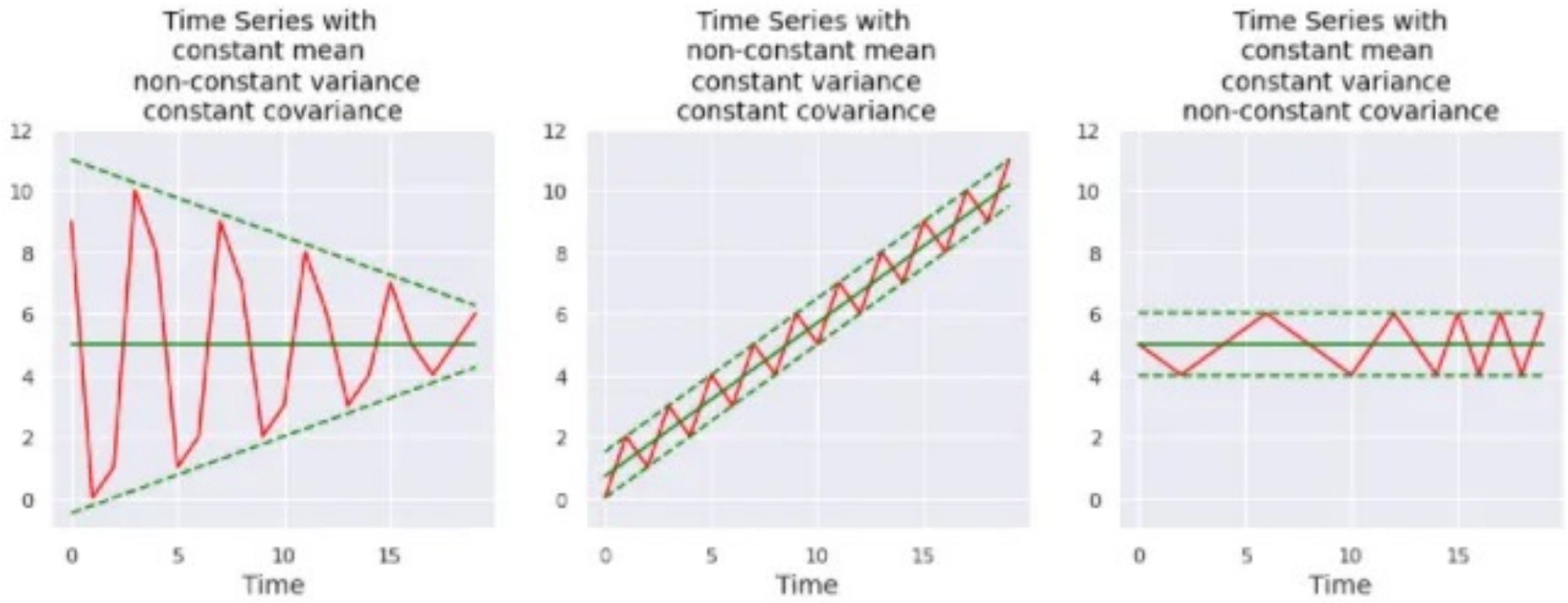
- **Stationary time series** is one whose statistical properties do not change over time
  - **Constant mean:** average value remains the same throughout the time series
  - **Constant variance:** spread of data points do not change
  - **Constant autocovariance:** relationship between past and present values depends on the lag, not the time
- Non-stationary time series is one whose any of these properties change over time



Credit: https://medium.com/codex/what-is-stationarity-in-time-series-how-it-can-be-detected-7e5dfa7b5f6b

Stationary vs. Non-Stationary

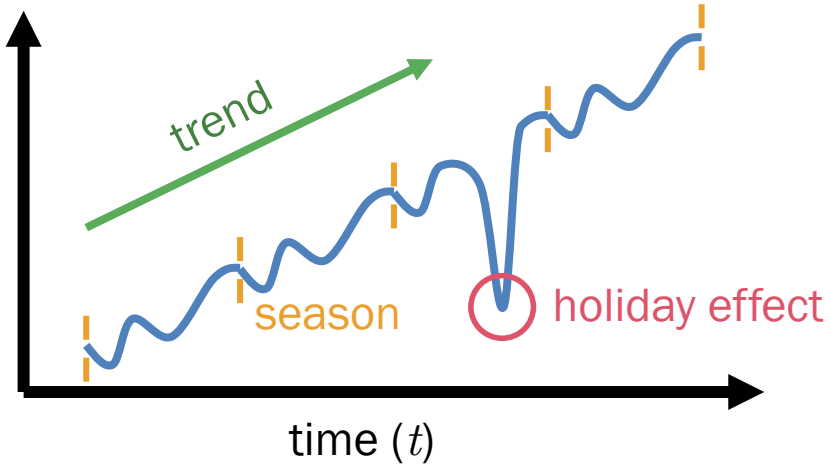
- **Non-stationary time series** is one whose mean, variance (spread), or autocovariance (lag) change over time



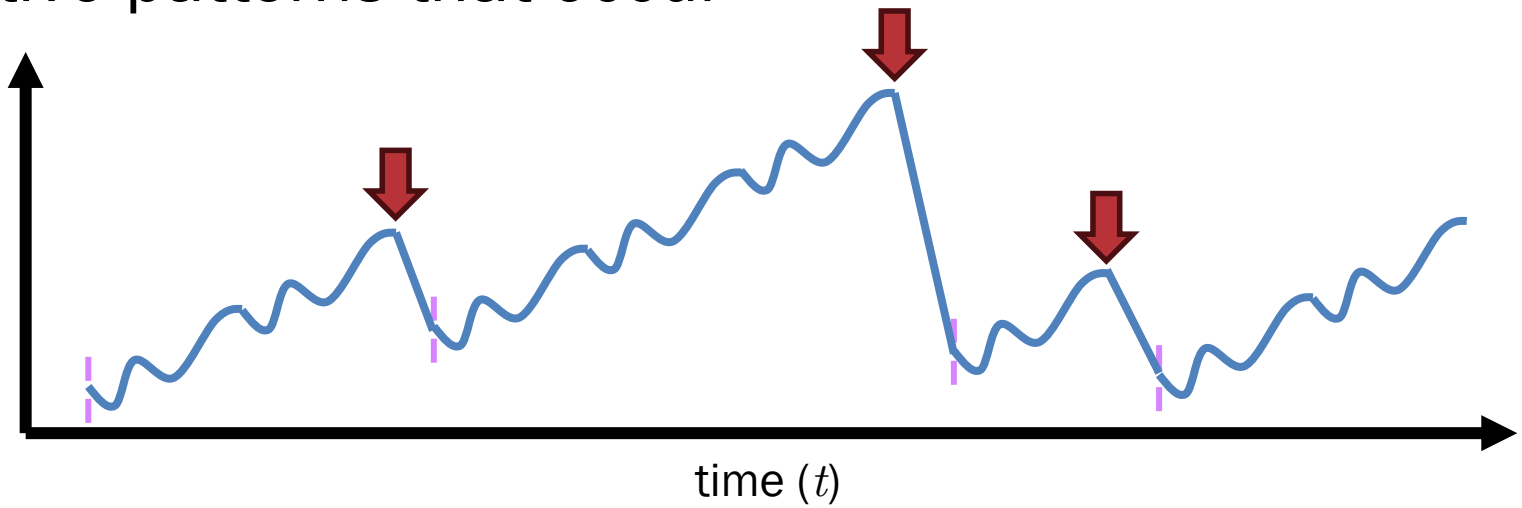
Credit: https://medium.com/codex/what-is-stationarity-in-time-series-how-it-can-be-detected-7e5dfa7b5f6b

Patterns in Time Series

- **Trend:** general direction over time
- **Seasonality:** repetitive patterns that occur at regular predictable intervals
- **Holiday effects:** irregular patterns caused by special calendar events
- **Cycle:** long-term repetitive patterns that occur at irregular intervals



Cycle
1. Slow increase
2. Catastrophe
3. Rapid decline



Assumptions about Time Series

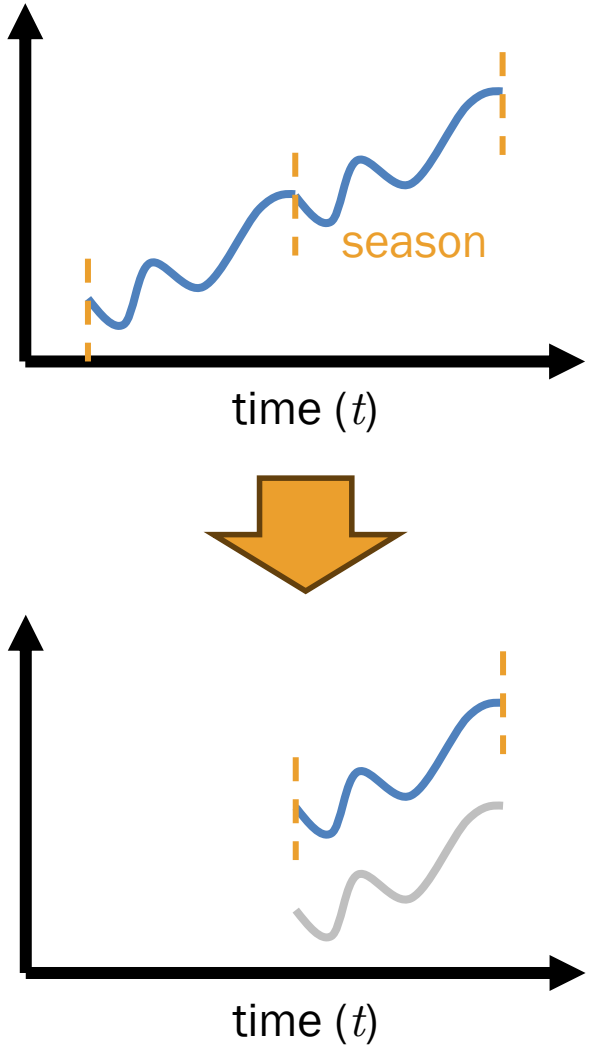
- Time domain
  - **A1:** Stochastic process (observations closer in time are more closely related)
  - **A2:** Combination of temporal structures
- Frequency domain
  - **A3:** Combination of continuous waves
  - **A4:** Combination of wavelets (i.e. wavelike pieces)
- Sequence-to-sequence prediction
  - **A5:** Transformer-based models

2. Time-Domain Methods

2.1 ARIMA Model

ARIMA Model

- Auto-Regressive Integrated Moving Average
  - **Assumption:** The dataset is seasonal and the difference between seasons can be predicted by linear regression
  - **How:** Predict a future value across the season by linear regression of previous cross-seasonal differences and adjust the error by linear regression of previous errors
- Three parameters of $\mathrm{ARIMA}(p,d,q)$
  - Season duration: d timesteps
  - No. cross-seasonal differences: p timesteps
  - No. previous errors: q timesteps



$\mathrm{ARIMA}(p,d,q)$

- **Step 1:** Integrate the season of length d

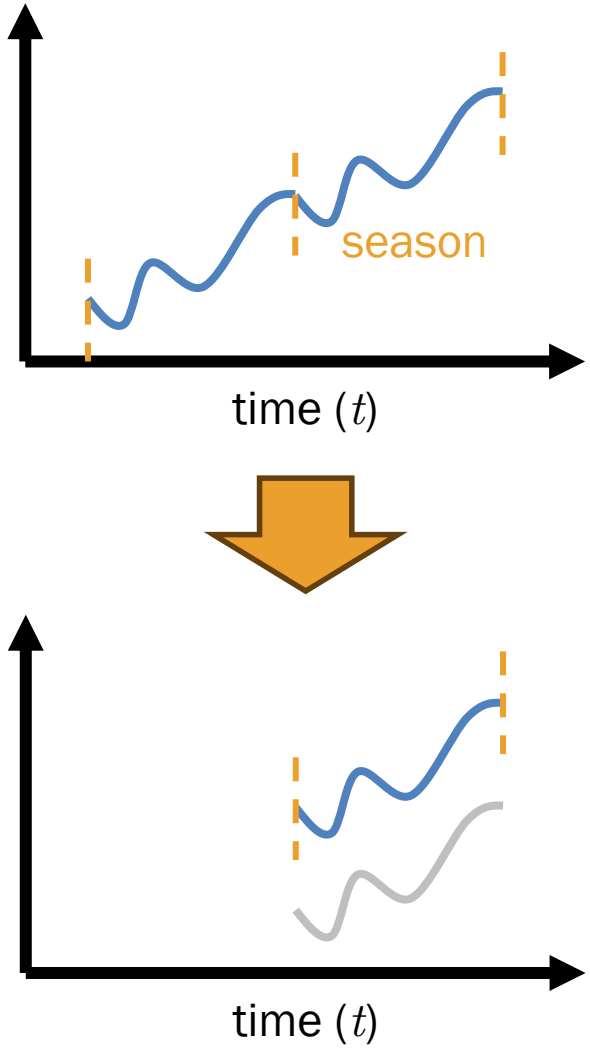
$$x'_t = x_t - x_{t-d}$$

for every point x_t in the time series

- **Example:**

Season length $d = 3$

t	x_t
1	10
2	15
3	20
4	25
5	28
6	38



$\mathrm{ARIMA}(p,d,q)$

- **Step 1:** Integrate the season of length d

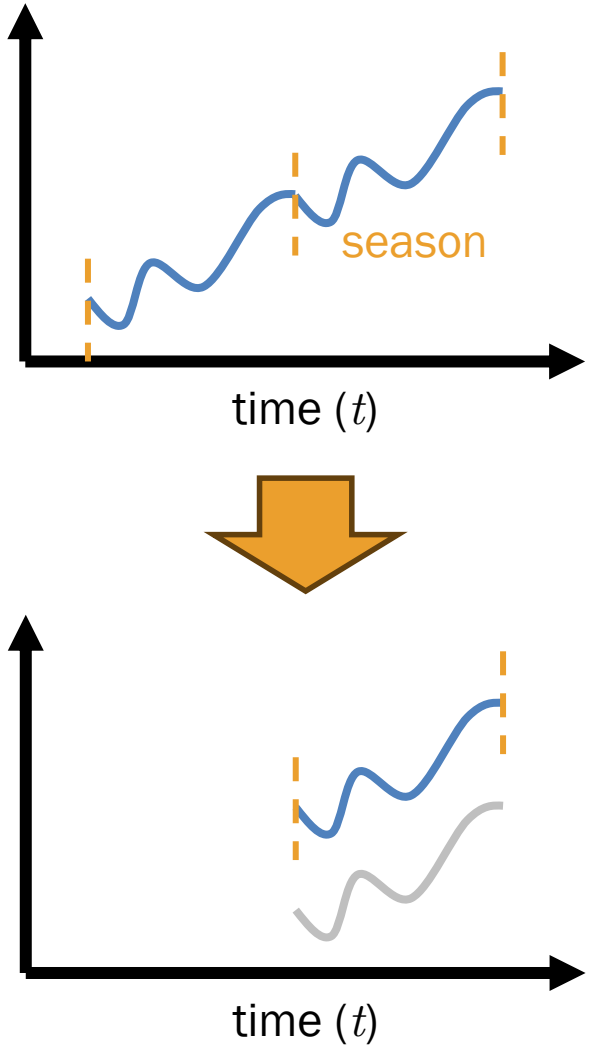
$$x'_t = x_t - x_{t-d}$$

for every point x_t in the time series

- **Example:**

Season length $d = 3$

t	x_t	x_{t-d}	x'_t
1	10	–	–
2	15	–	–
3	20	–	–
4	25	10	15
5	28	15	13
6	38	20	18



$\mathrm{ARIMA}(p,d,q)$

- **Step 1:** Integrate the season of length d

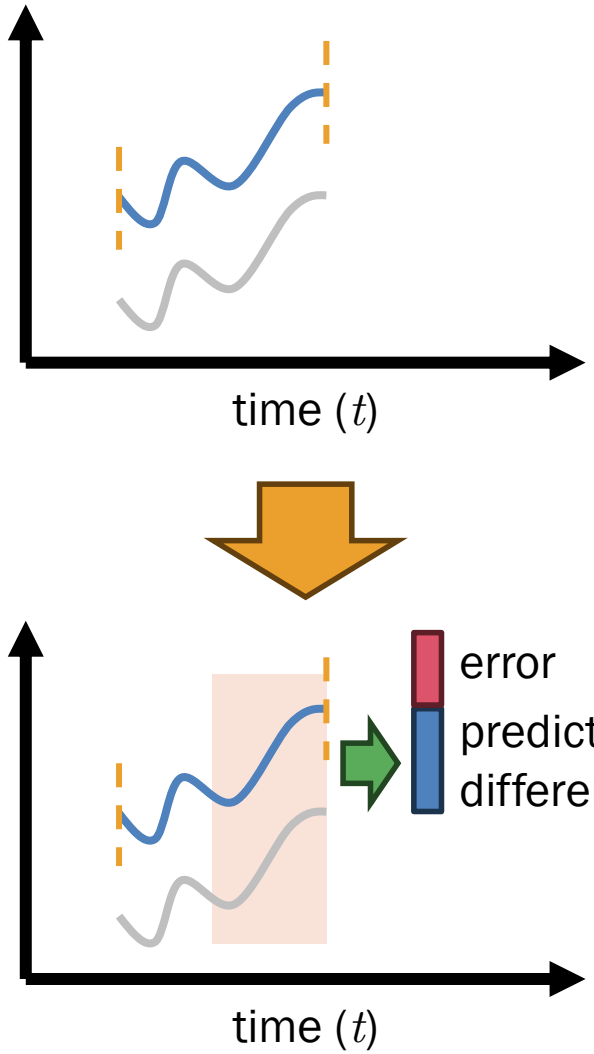
$$x'_t = x_t - x_{t-d}$$

- **Step 2:** Predict the current difference with a linear regression of p previous differences

$$x'_t = \underbrace{\left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right]}_{\text{predicted difference}} + \underbrace{e_t}_{\text{error}}$$

$$= x_t - (x_{t-d} + \text{diff})$$

The term ‘autoregressive’ means taking the recent outputs as inputs for the next computation



$\mathrm{ARIMA}(p,d,q)$

- **Step 1:** Integrate the season of length d

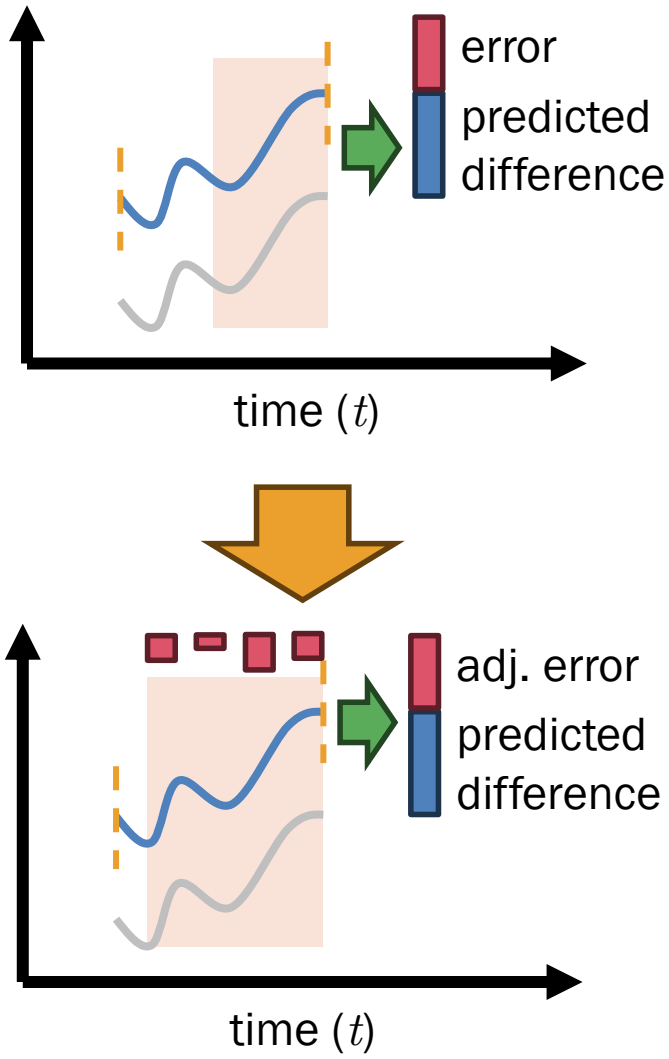
$$x'_t = x_t - x_{t-d}$$

- **Step 2:** Predict the current difference with a linear regression of p previous differences

$$x'_t = \left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right] + e_t$$

- **Step 3:** Adjust the prediction error with a linear regression of q previous errors

$$x'_t = \left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right] + e_t + \left[\sum_{j=1}^{q} \theta_j e_{t-j}\right]$$



Training Algorithm of $\mathrm{ARIMA}(p,d,q)$

- Suppose each data point x_t is in the training set
  - Compute the differences x'_t for each data point

$$x'_t = x_t - x_{t-d}$$

  - Estimate parameters φ_k with MSE of all e_t

$$e_t = x'_t - \left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right]$$

$$\mathrm{MSE} = \frac{1}{N}\sum_{t=1}^{N} e_t^2$$

  - Estimate parameters θ_k with MSE of all e_t with fixed φ_k

$$e_t = x'_t - \left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right] - \left[\sum_{j=1}^{q} \theta_j e_{t-j}\right]$$

Prediction of $\mathrm{ARIMA}(p,d,q)$

- Suppose we want to predict future values x_{t+1} to $x_{t+\mathrm{N}}$
  - We compute the differences and errors of timesteps $t{+}1$ to $t{+}N$

$$x'_t = \sum_{k=1}^{p} \phi_k x'_{t-k} + \sum_{j=1}^{q} \theta_j e_{t-j}$$

$$e_t = x'_t - \left[\sum_{k=1}^{p} \phi_k x'_{t-k}\right] - \left[\sum_{j=1}^{q} \theta_j e_{t-j}\right]$$

  - We compute the future values from the differences

$$x_t = x'_t + x_{t-d}$$

Incomplete Time Series

- Interpolation techniques
  - **Newton's interpolation:** each a_i is computed by divided differences

$$N(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \ldots + a_n \prod_{k=0}^{n-1}(x - x_k)$$

  - **Cubic spline interpolation:** cubic curve

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

  - **Chebyshev's interpolation:** sinusoidal seasonality

$$x_i = \frac{a+b}{2} + \frac{b-a}{2}\cos\left(\frac{2i+1}{2n}\pi\right)$$

  - **Radial basis function interpolation:** spread around the means x_i

$$R(x) = \sum_{i=1}^{n} \lambda_i \cdot \phi\left(|x - x_i|\right)$$

Evaluation of Prediction Models

- **Mean absolute error (MAE):** average absolute difference between prediction and gold standard

$$\text{MAE} = \frac{1}{N}\sum_{k=1}^{N}|y_k - \hat{y}_k|$$

- **Root mean squared error (RMSE):** square root of mean squared difference between prediction and gold standard

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2}$$

- **Mean absolute percentage error (MAPE):** average percentage difference between prediction and gold standard (w.r.t. gold standard)

$$\text{MAPE} = \frac{1}{N}\sum_{k=1}^{N}\left|\frac{y_k - \hat{y}_k}{y_k}\right|$$

- **Forecast bias:** average bias (prediction – gold standard)

$$\text{bias} = \frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)$$

2.2 Convolutional Analysis and CNNs

Convolution $*$

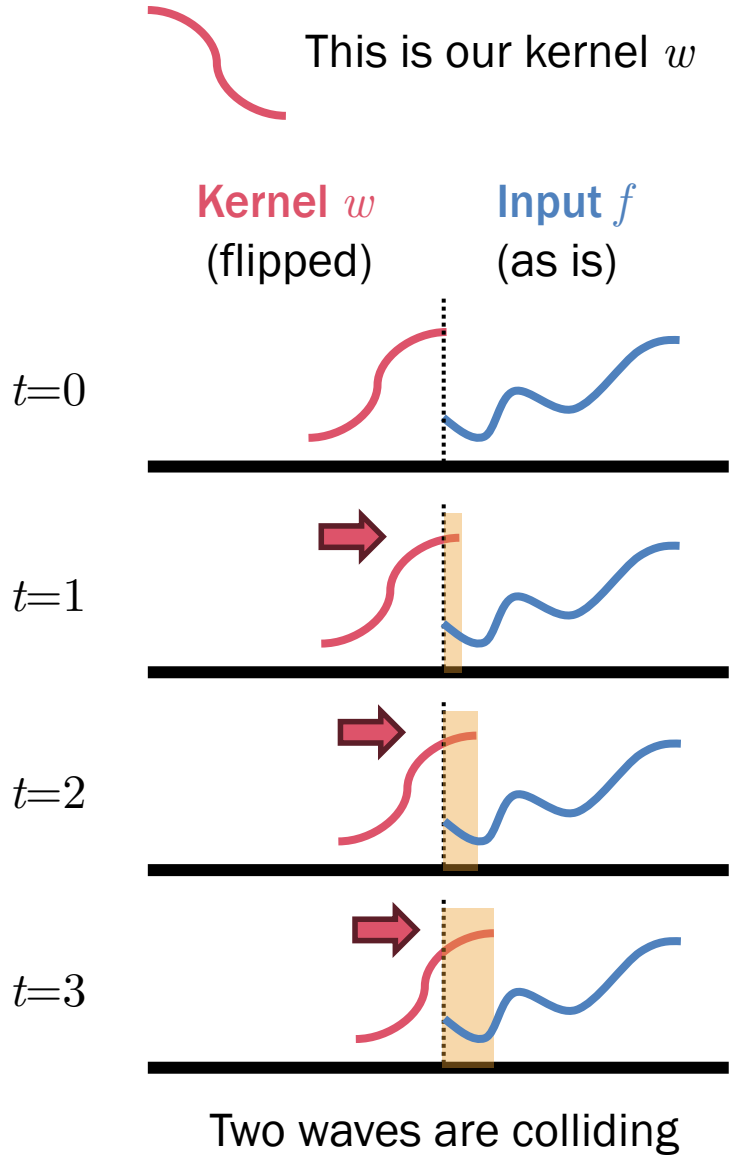
- Combining two functions to produce a third one that represents how one amplifies the other in each step of their overlapping
  - **Continuous convolution**

$$[f * w](t) = \int_{-\infty}^{\infty} f(\tau) \cdot w(t - \tau)\ d\tau$$

  - **Discrete convolution**

$$[f * w](t) = \sum_{k=-\infty}^{\infty} f[k] \cdot w[t - k]$$

  - where f is an input signal and w is a kernel (i.e. pattern filter)



Cross-Correlation $\odot$

This is our kernel w

- The intuitive counterpart of convolution, where the kernel function is not flipped
  - **Continuous cross-correlation**

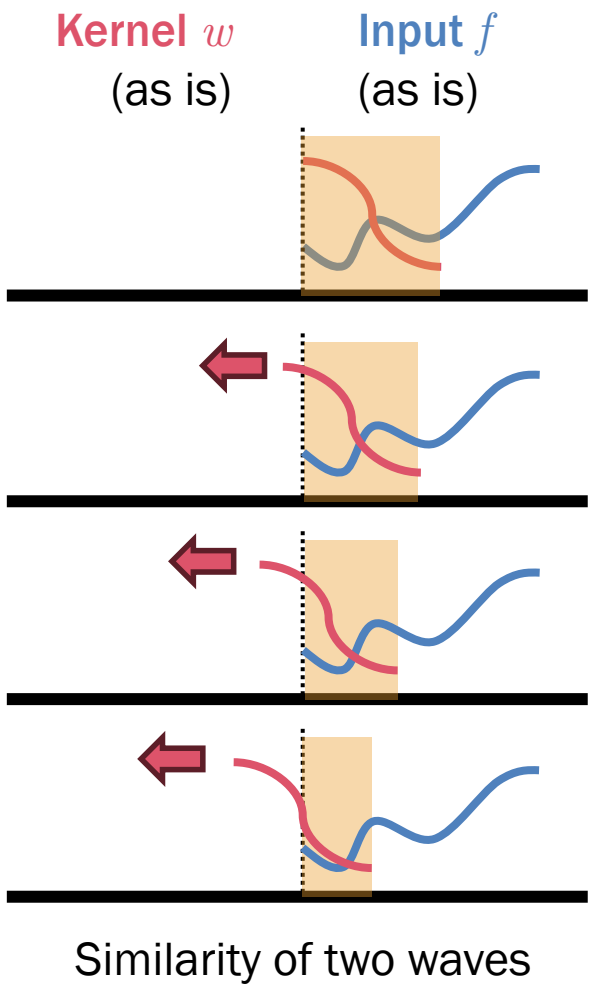
$* $ is the complex conjugate
e.g. $(3 + 4\hat{\imath})^* = 3 - 4\hat{\imath}$

$$[f \odot w](t) = \int_{-\infty}^{\infty} f(\tau) \cdot w(t+\tau)^* \, d\tau$$

  - **Discrete cross-correlation**

$$[f \odot w](t) = \sum_{k=-\infty}^{\infty} f[k] \cdot w[t+k]^*$$

- We will say '**convolution**' to refer to cross-correlation for ease of understanding



Similarity of two waves

Convolutional Filtering

Find 

(kernel)

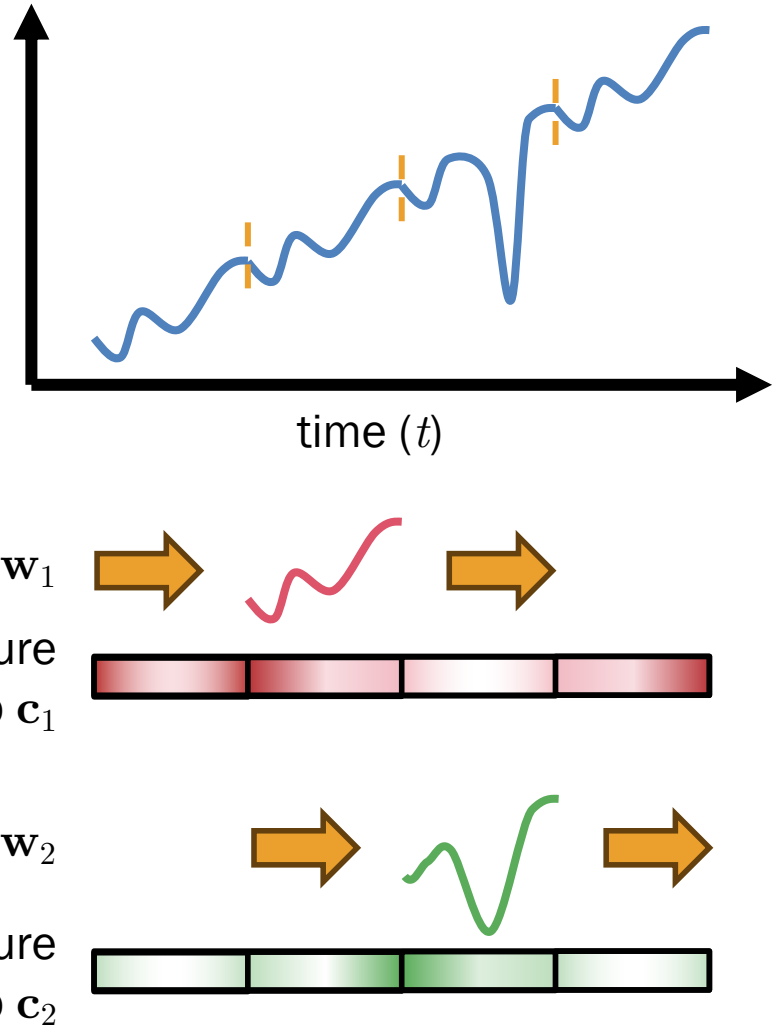
Convolutional Filtering

Find  (kernel)

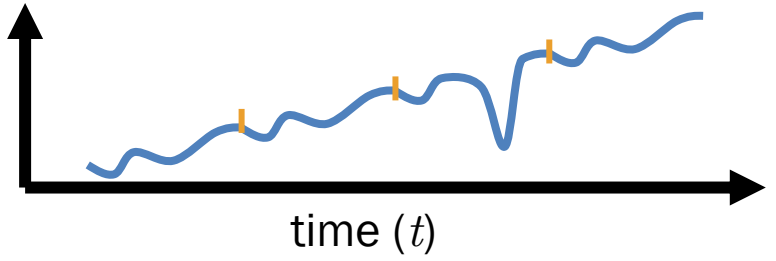
- Matched areas are amplified
- Objects are detected via convolution
- We have extracted the **local features**

Convolution in Time Series

- Pattern matching with kernels
  - **Assumption:** Local features are detected by a series of peaks
  - Period of seasonality can be identified by the peaks of cross-correlations
  - Holiday effects are also identified by the irregularity in cross-correlation peaks
  - One time series may align (correlate) with a mixture of kernels



Convolutional Neural Networks



- Learn several kernels from the dataset and identify which kernels to be used at which time
  - Convolution layer with 3 kernels (width=10)

$$\mathbf{c}_i = \mathbf{x} \odot \mathbf{w}_i$$

N.B. These kernels can learn any shape of trends

where each $\mathbf{w}_i$ is a vector of 10 parameters

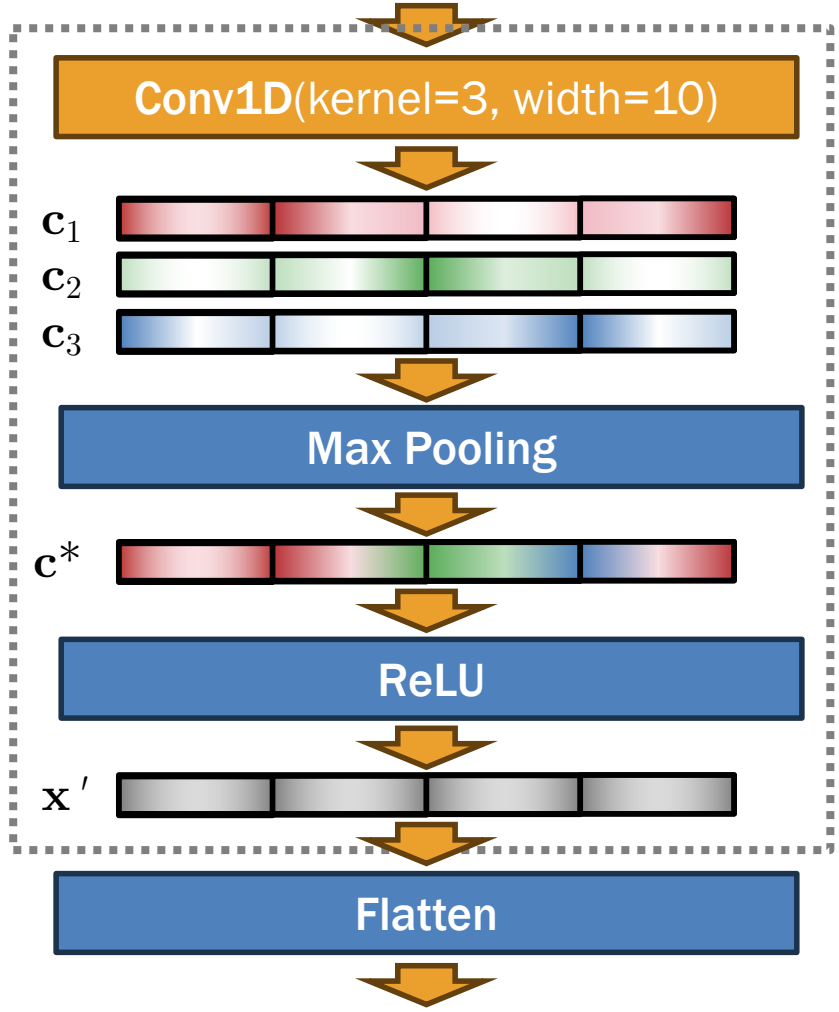
  - Max-pooling for most prominent local features

$$\mathbf{c}^* = \max\left[\mathbf{c}_1 | \mathbf{c}_2 | \mathbf{c}_3\right]$$

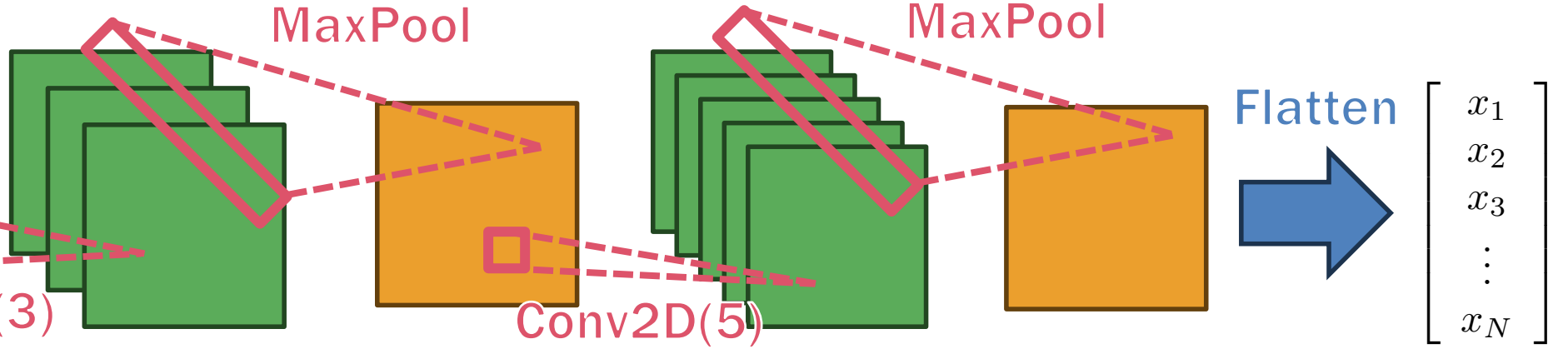
  - Nonlinear function

$$\mathbf{x}' = \mathrm{ReLU}(\mathbf{c}^*)$$

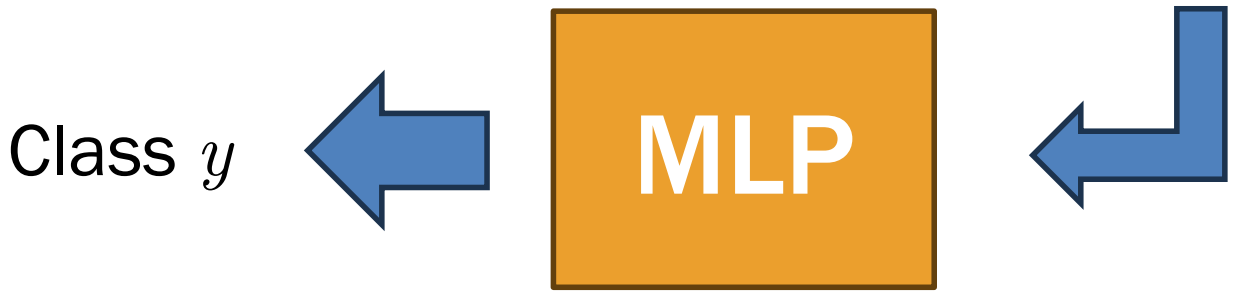
  - The matrix is then flattened to become a vector



CNNs for Computer Vision



Feature maps | Pooled feature map | Feature maps | Pooled feature map | Flattened vector

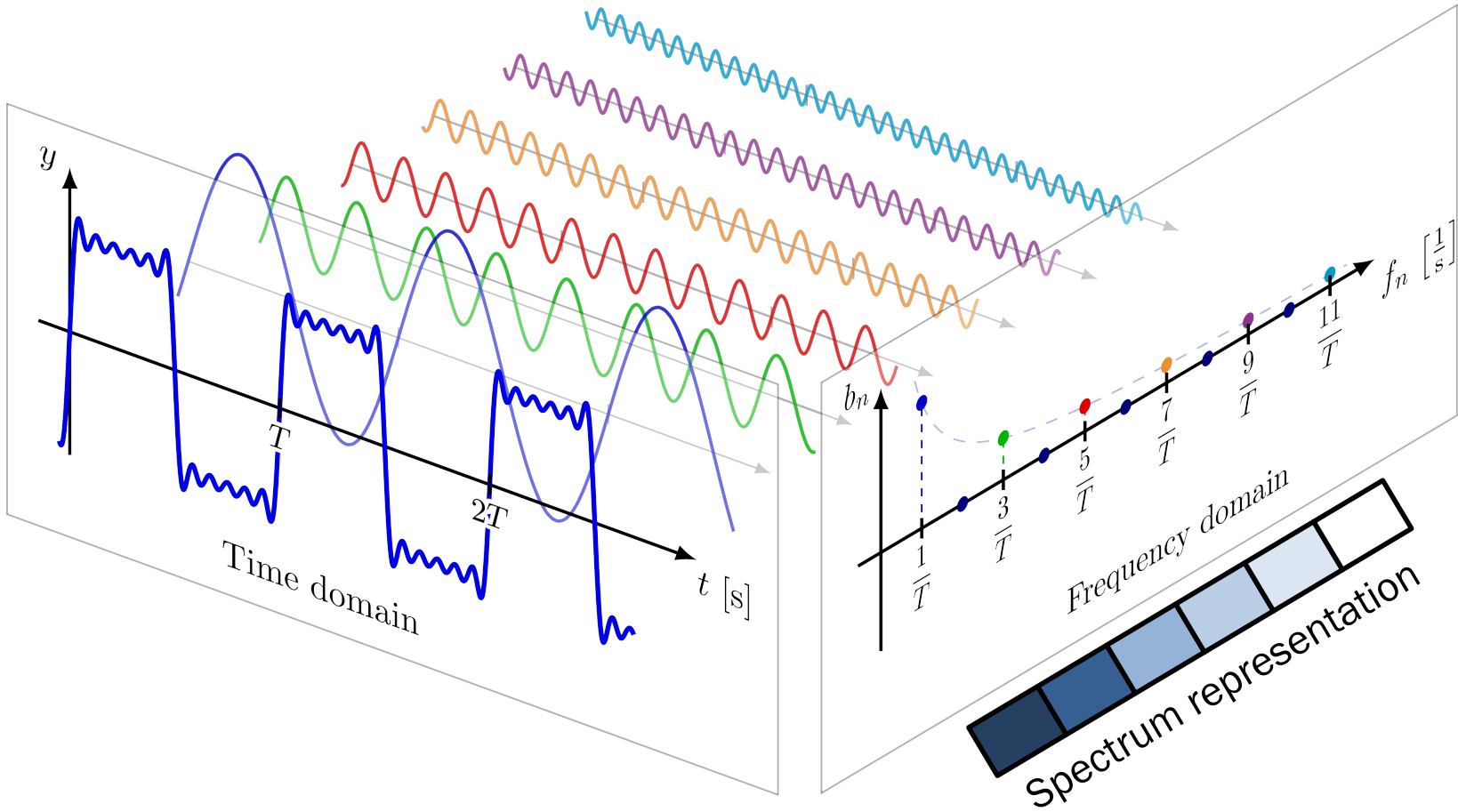


3. Frequency-Domain Methods

3.1 Spectral Density Analysis

Spectral Density

- Describing a time series in terms of power distribution according to frequency components
- **Assumption:** Time series is a mixture of periodic sinusoid waves (seasonal patterns)
- Transforming a time series into **spectrum** (squared amplitude) in the frequency domain

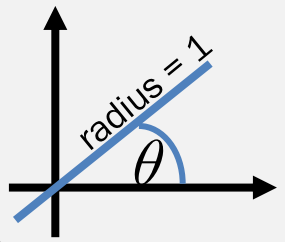


Source: https://dibsmethodsmeetings.github.io/fourier-transforms/

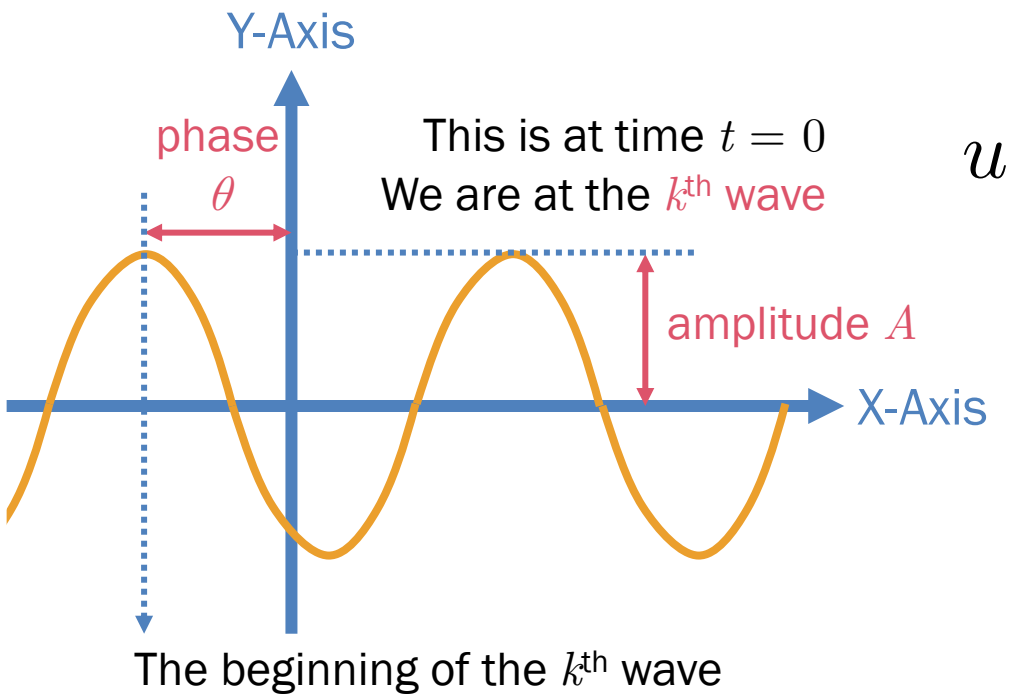
Wave as Complex Number

- A complex number $a+\hat{\imath}b$ consists of the real part a and the imaginary part b, where $\hat{\imath} = \sqrt{-1}$
- Wave can be represented as a complex number

Euler's formula



$$\exp[\hat{\imath}\theta] = \cos\theta + \hat{\imath}\sin\theta$$

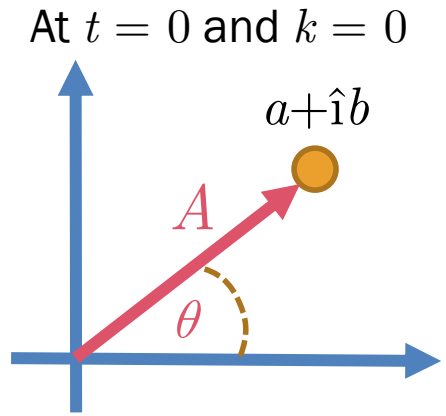


frequency

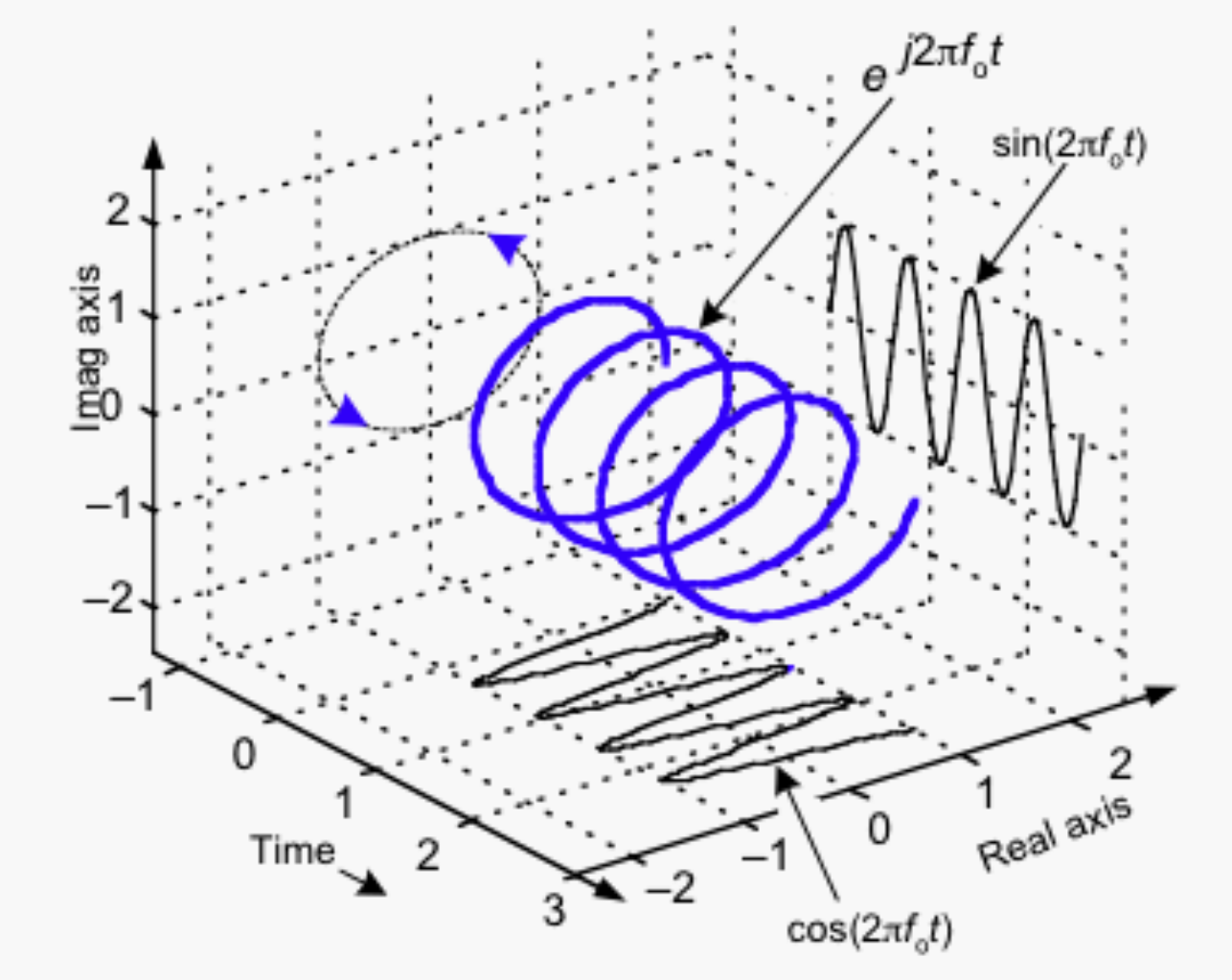
$$u(x,t) = A\cos(kx - \omega t + \theta)$$

$$u(x,t) = u_0 e^{-\hat{\imath}(\omega t - kx)}$$

where $u_0 = Ae^{\hat{\imath}\theta}$



Wave as 3D Spiral



- Wave can be seen as a counterclockwise spiral in 3D space, whose base plane are real and imaginary parts

$$\exp[\hat{\imath}\omega t] = \underbrace{\cos\omega t}_{\text{real part}} + \underbrace{\hat{\imath}\sin\omega t}_{\text{imaginary part}}$$

- Conjugate of a wave is therefore a clockwise spiral

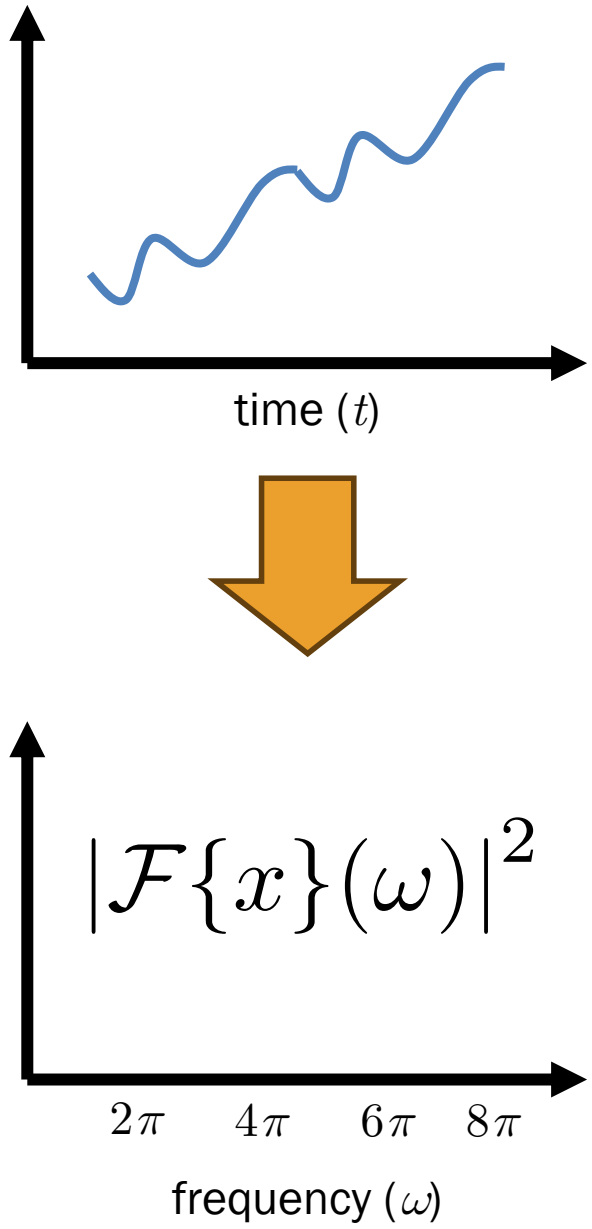
$$\exp[\hat{\imath}\omega t]^* = \exp[-\hat{\imath}\omega t]$$
$$= \cos\omega t - \hat{\imath}\sin\omega t$$

Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

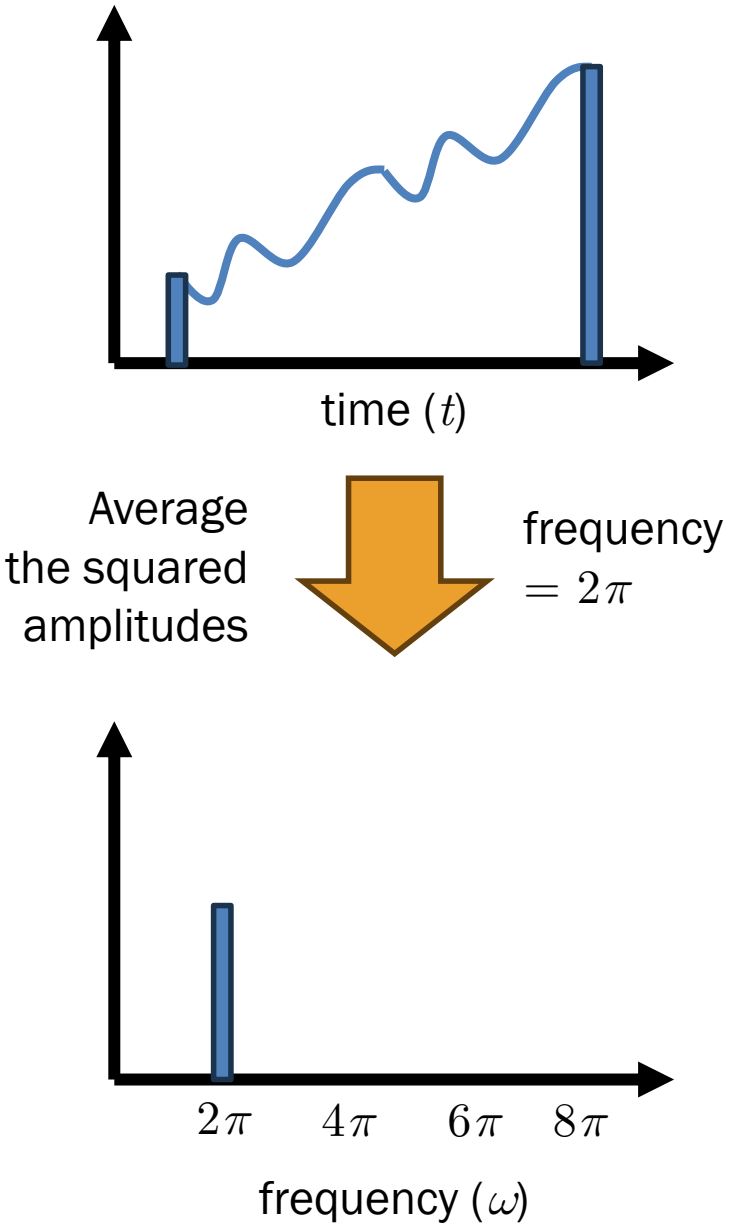


Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

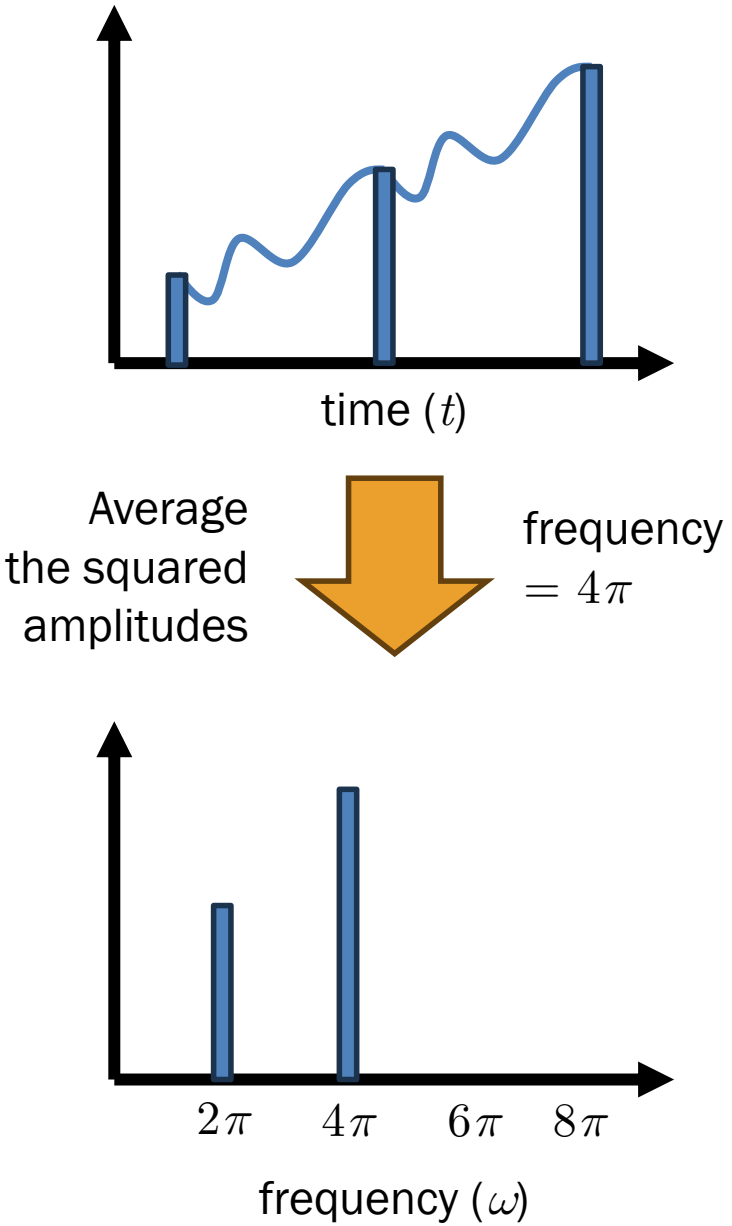


Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

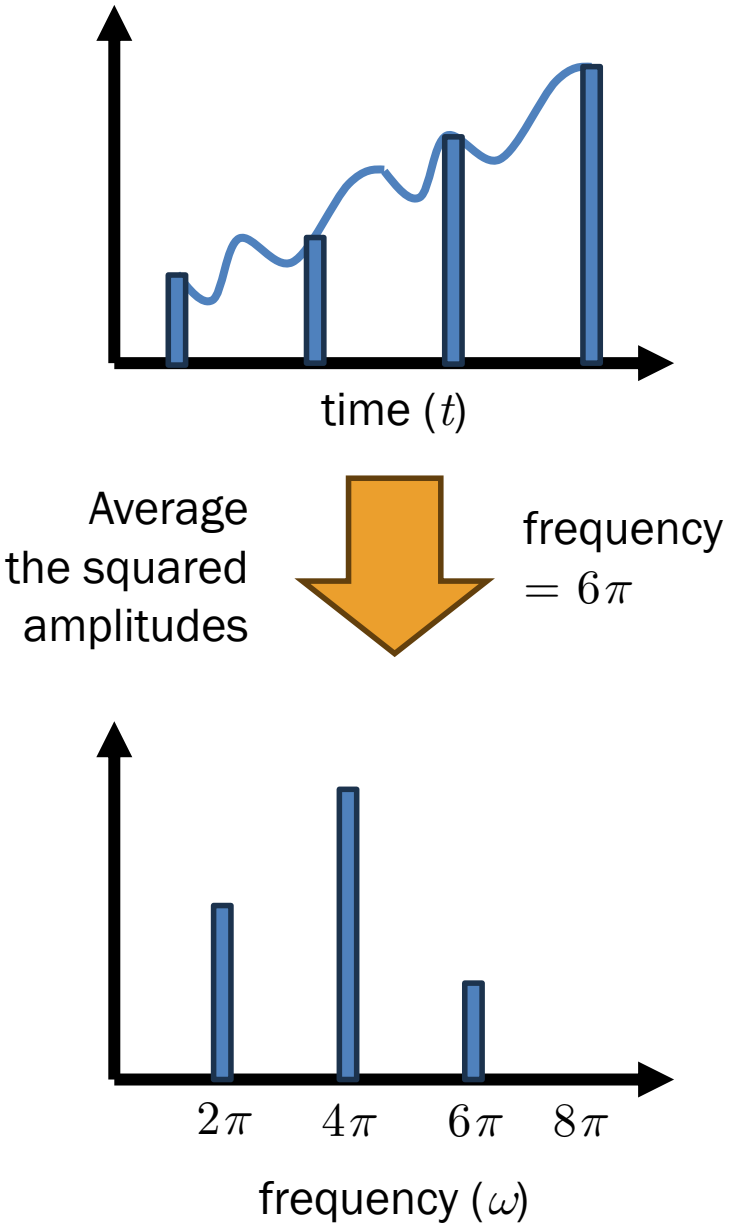


Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

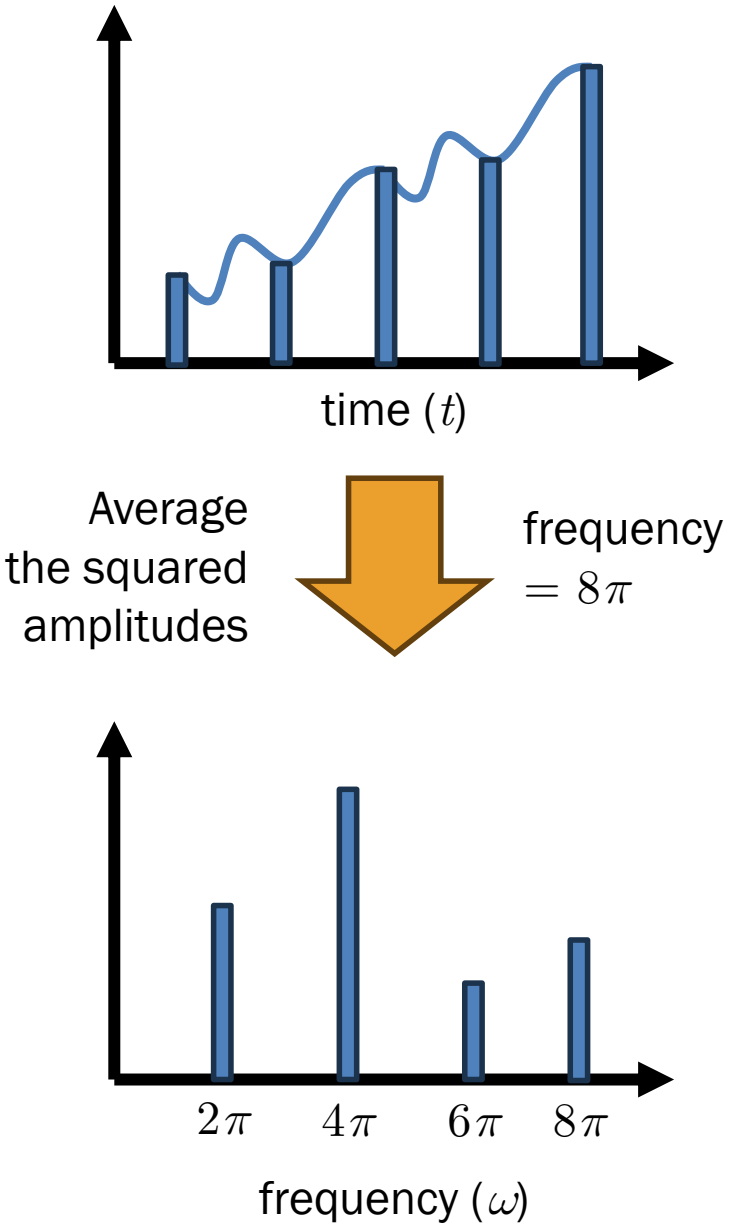


Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

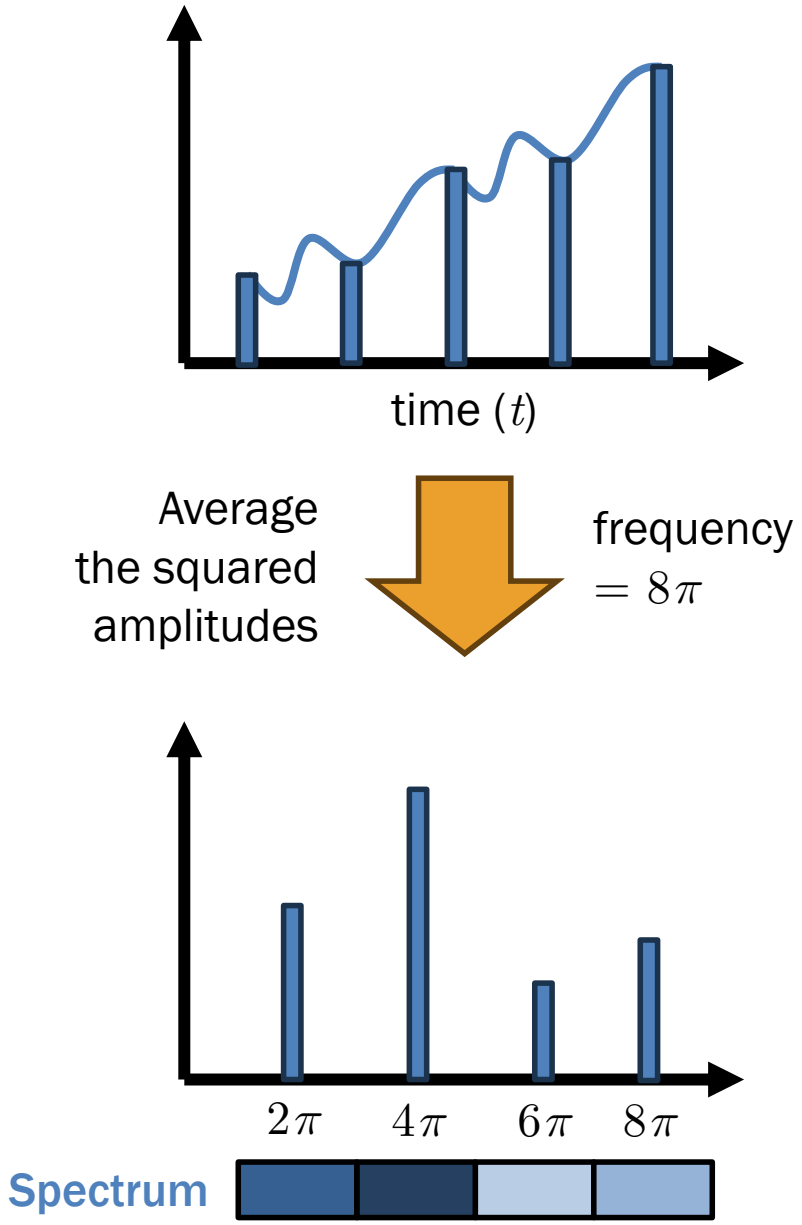


Fourier Transform

- Fourier transform of a time series $x(t)$ is a function of frequency ω that reflects how well $x(t)$ aligns with the wave of frequency ω

$$\begin{aligned}\mathcal{F}\{x\}(\omega) &= x(t) \odot \text{wave}(\omega, t) \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[\hat{\imath}\omega t]^* \; dt \\ &= \int_{-\infty}^{\infty} x(t) \cdot \exp[-\hat{\imath}\omega t] \; dt\end{aligned}$$

- The frequency is usually a multiple of 2π (i.e. one round of a circle in radian)

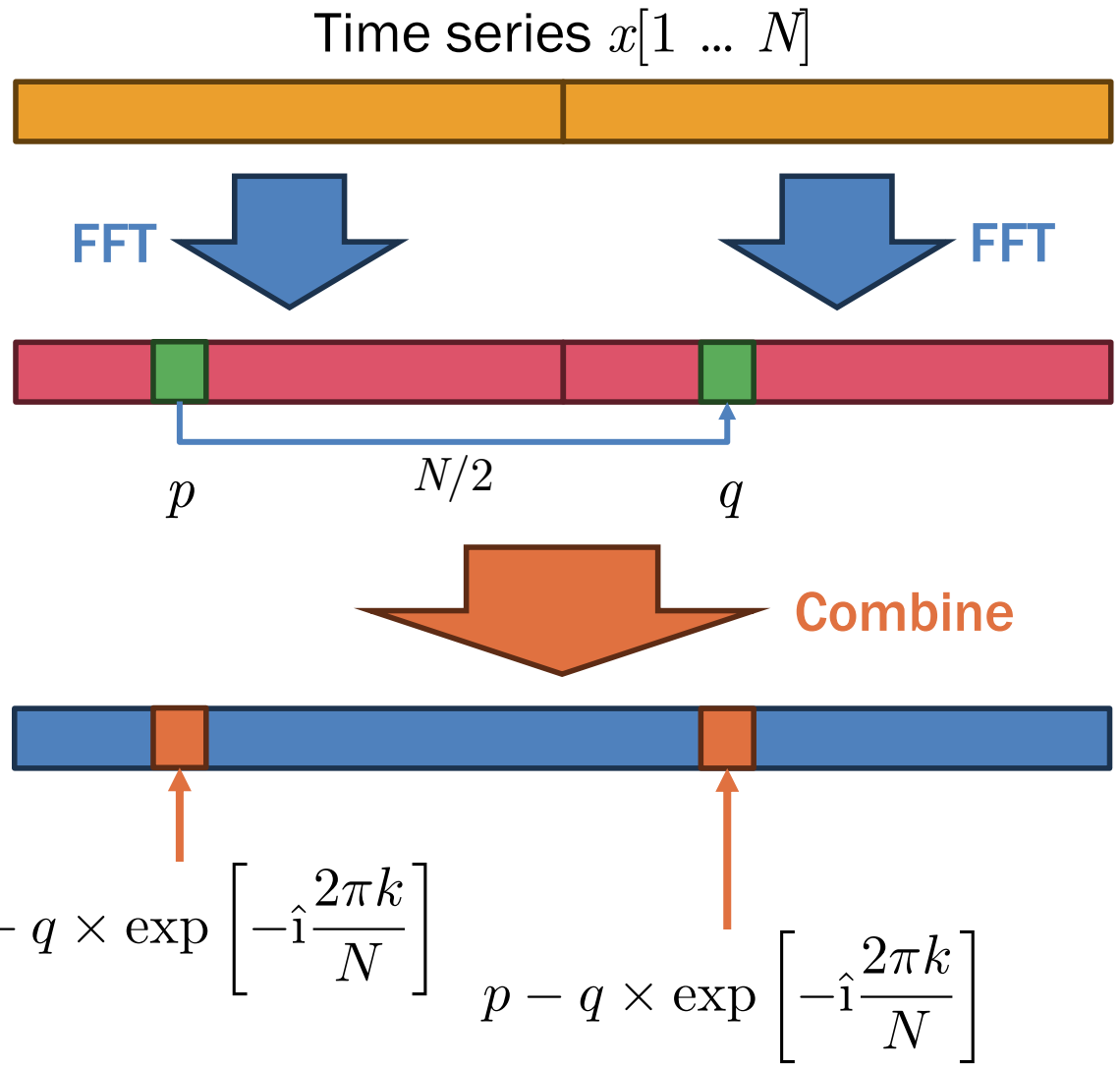


Fast Fourier Transform (Cooley & Tukey, 1965)

```
def fft(x: array, N: length, s: stride):
    result := create_array(size=N)

    if N == 1:
        result[0] = x[0]
    else:
        result[0 : N/2] := fft(x, N/2, 2*s)        # x[0], x[2s], x[4s], ...
        result[N/2 : N] := fft(x[s:], N/2, 2*s)    # x[s], x[3s], x[5s], ...
        for k in range(N/2):
            p := result[k]
            q := result[k + N/2] * exp(-2 * π * î * k/N)
            result[k] := p + q
            result[k + N/2] := p - q

    return result

HOW TO RUN ==> fft(x, N, s=1) ==> Then compute the squared magnitude of each element
```

$O(n \log n)$ time complexity

Fast Fourier Transform (Cooley & Tukey, 1965)



- Base case

$$\mathcal{F}(x[k]) = x[k]$$

- Recursive case

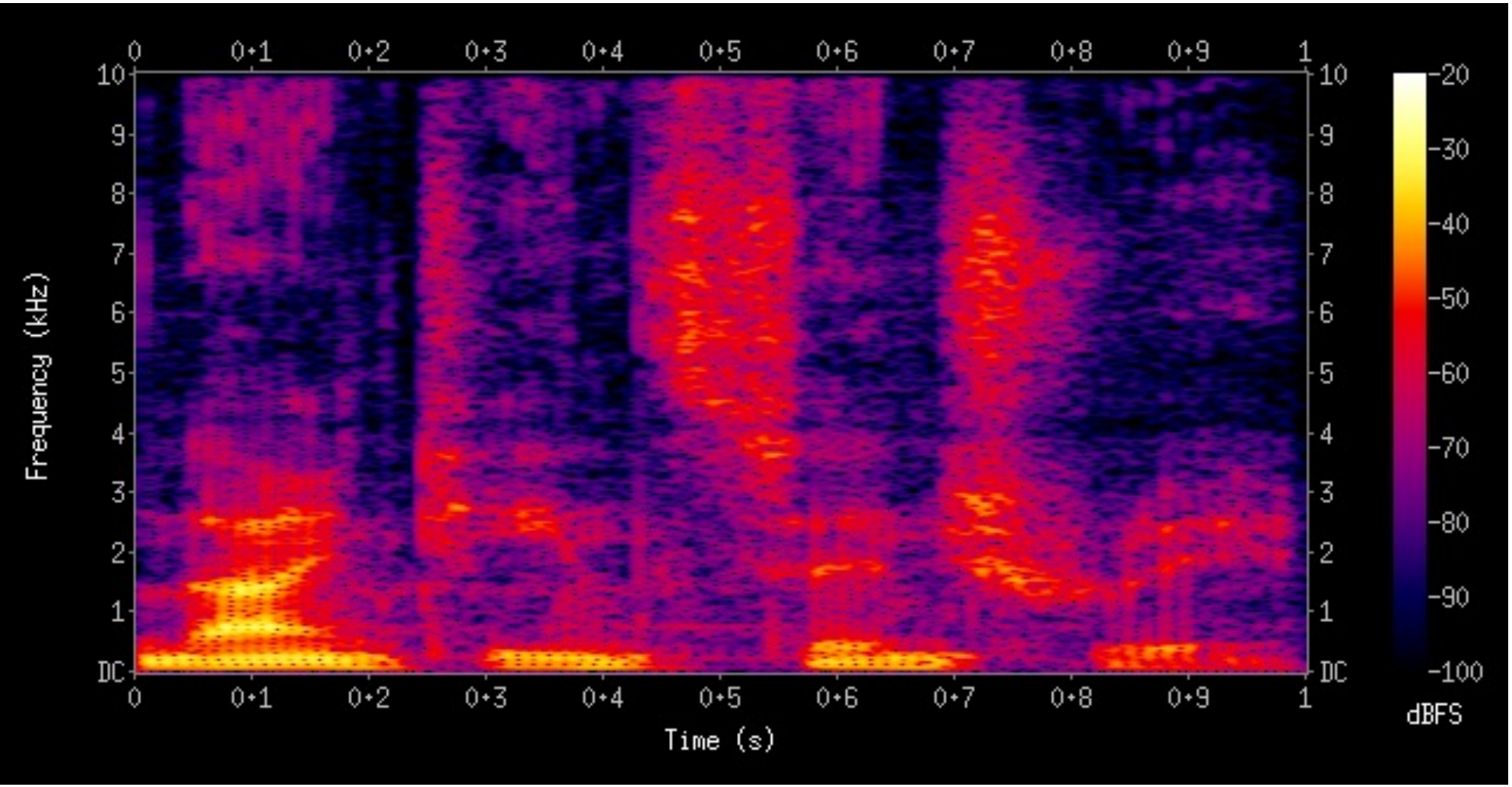
$$p = \mathcal{F}[x[k]]$$

$$q = \mathcal{F}[x[k + \frac{N}{2}]]$$

$$\mathcal{F}[x[k]] = p + q \times \exp\left[-\hat{\imath}\frac{2\pi k}{N}\right]$$

$$\mathcal{F}[x[k + \frac{N}{2}]] = p - q \times \exp\left[-\hat{\imath}\frac{2\pi k}{N}\right]$$

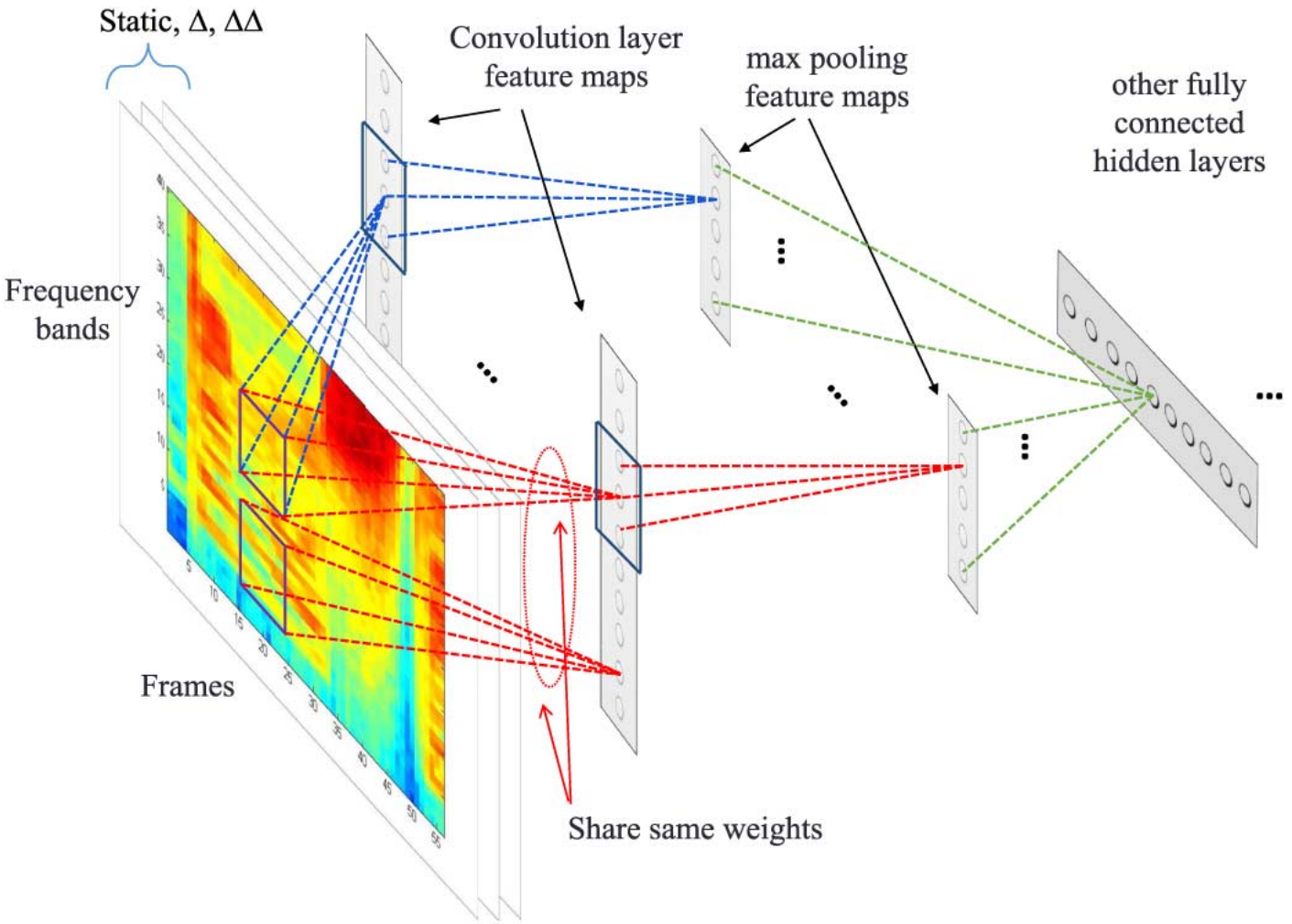
Spectogram



Source: https://en.wikipedia.org/wiki/Spectrogram

- Heatmap is used for visual representation of the spectrum of frequency
- Time series is tokenized into equal chunks (w.r.t. window size) and analyzed with FFT
- Seasonality and holiday effects are present
- Good for stationary signals e.g. long-term climate data

Asynchronous Speech Recognition



- Spectrogram is used as an input picture for CNN-based models
- Local features are changes in specific frequency ranges
- Conv2D is usually employed

Source: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/CNN_ASLPTrans2-14.pdf

3.2 Wavelet Analysis

Wavelet Transform

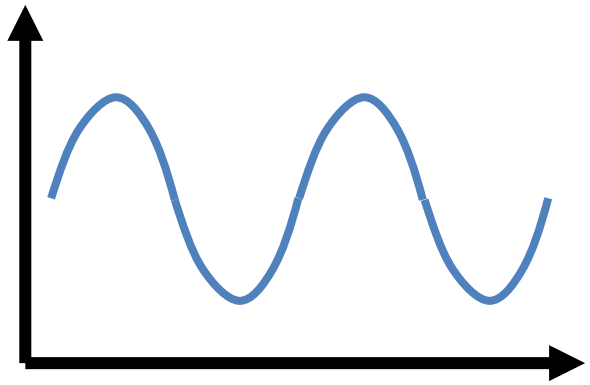
- Wavelet transform of $x(t)$ is a function of frequency a and position b that reflects how well $x(t)$ aligns with the wavelet $\Psi(a,b,t)$

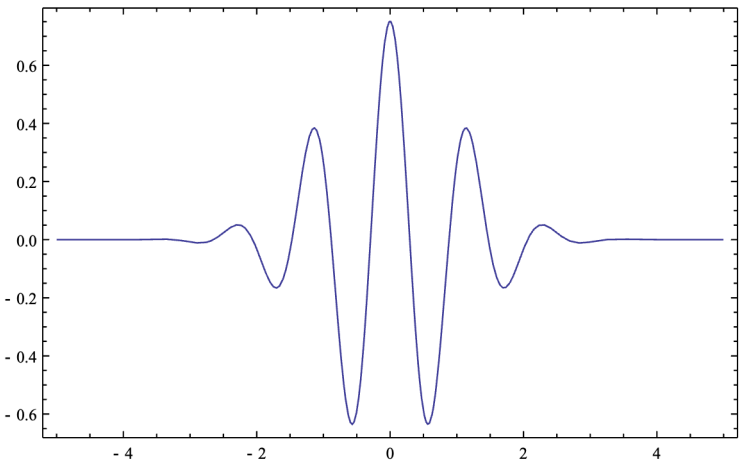
$$\mathcal{W}\{x\}(a,b) = x(t) \odot \Psi(a,b,t)$$
$$= \int_{-\infty}^{\infty} x(t) \cdot \cos a(t-b) \exp\left[(t-b)^2\right] \; dt$$

  where Morlet wavelet (1984) is used here:

$$\Psi(a,b,t) = \cos a(t-b) \exp\left[(t-b)^2\right]$$

- In practice, we need a low-pass filter $\Phi(t)$ to eliminate the inherent noise

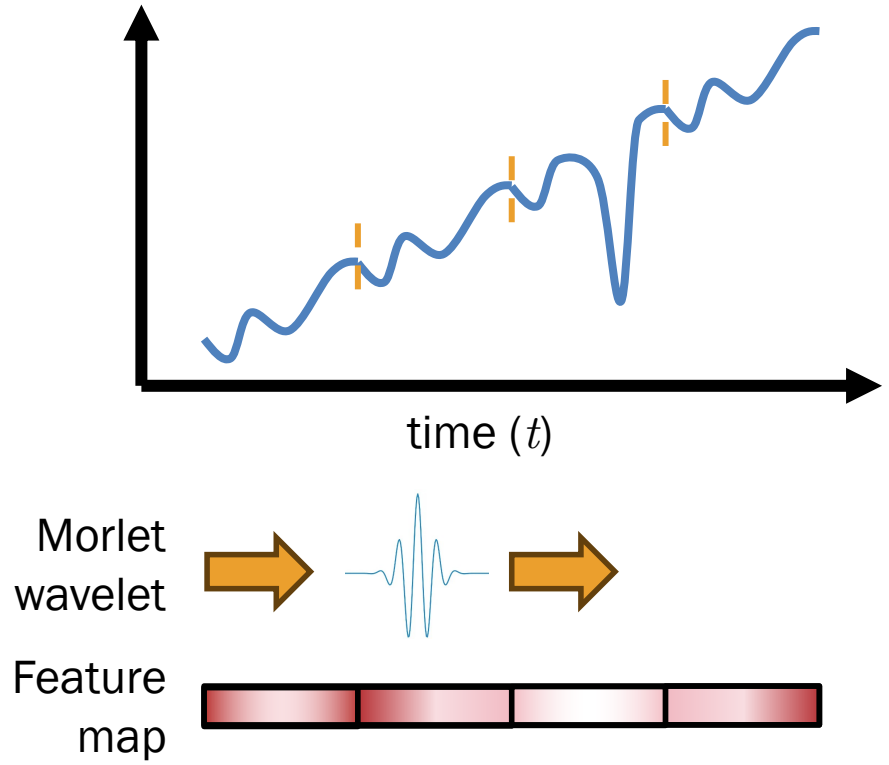




Morlet wavelet $\Psi(a,b,t)$

Wavelet Transform in Time Series

- Pattern matching with wavelet
  - **Assumption:** Local features are detected by a series of peaks
  - Period of seasonality can be identified by the peaks of cross-correlations
  - Holiday effects are also identified by the irregularity in cross-correlation peaks
  - One time series may align (correlate) with a mixture of wavelets

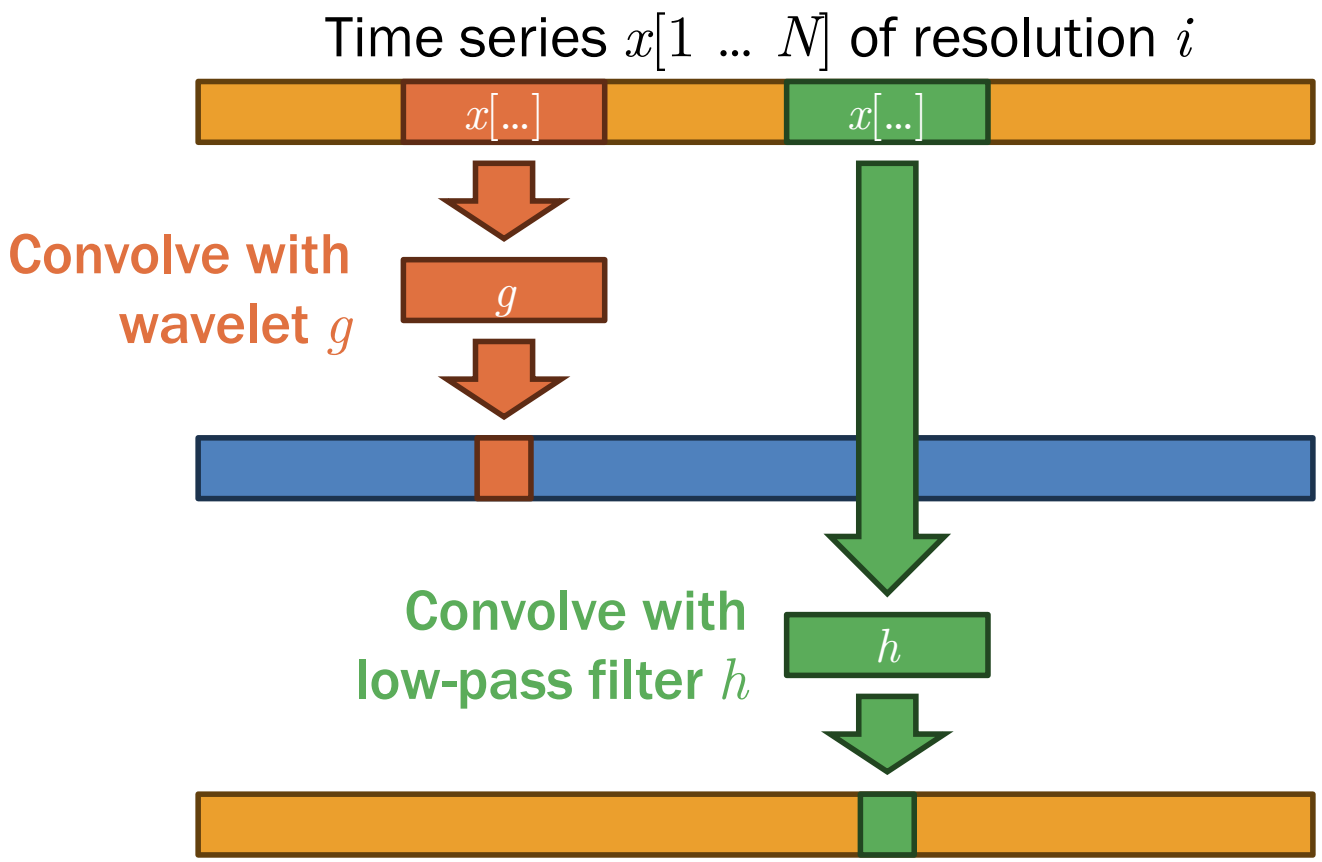


Fast Wavelet Transform (Mallat, 1989)

```
def fwt(x: array, N: length, h: low-pass filter, g: wavelet, L: resolution level)
    result := []
    arr := x

    for i = 1 to L:    # Resolution i
        row = []
        for k = 0 to N/2 - 1:
            coeff := sum(g[j] * arr[2*k - j] for j = 0 to len(g))
            row.append(coeff)
        result.insert(0, row)

        arr_next := []
        for k = 0 to N/2 - 1:
            coeff := sum(h[j] * arr[2*k - j] for j = 0 to len(h))
            arr_next.append(coeff)
        arr := arr_next

     result.insert(0, arr)
     return result
```

$O(n \log n)$ time complexity

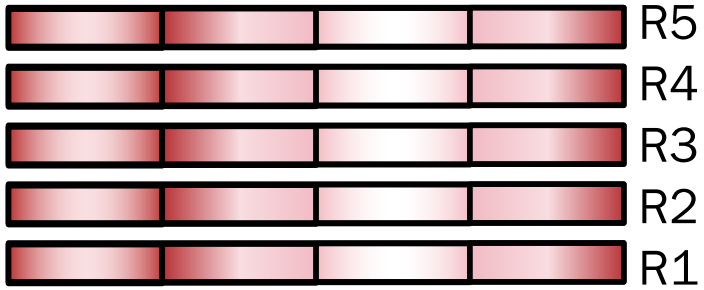
Fast Wavelet Transform (Mallat, 1989)



Resolution $i+1$

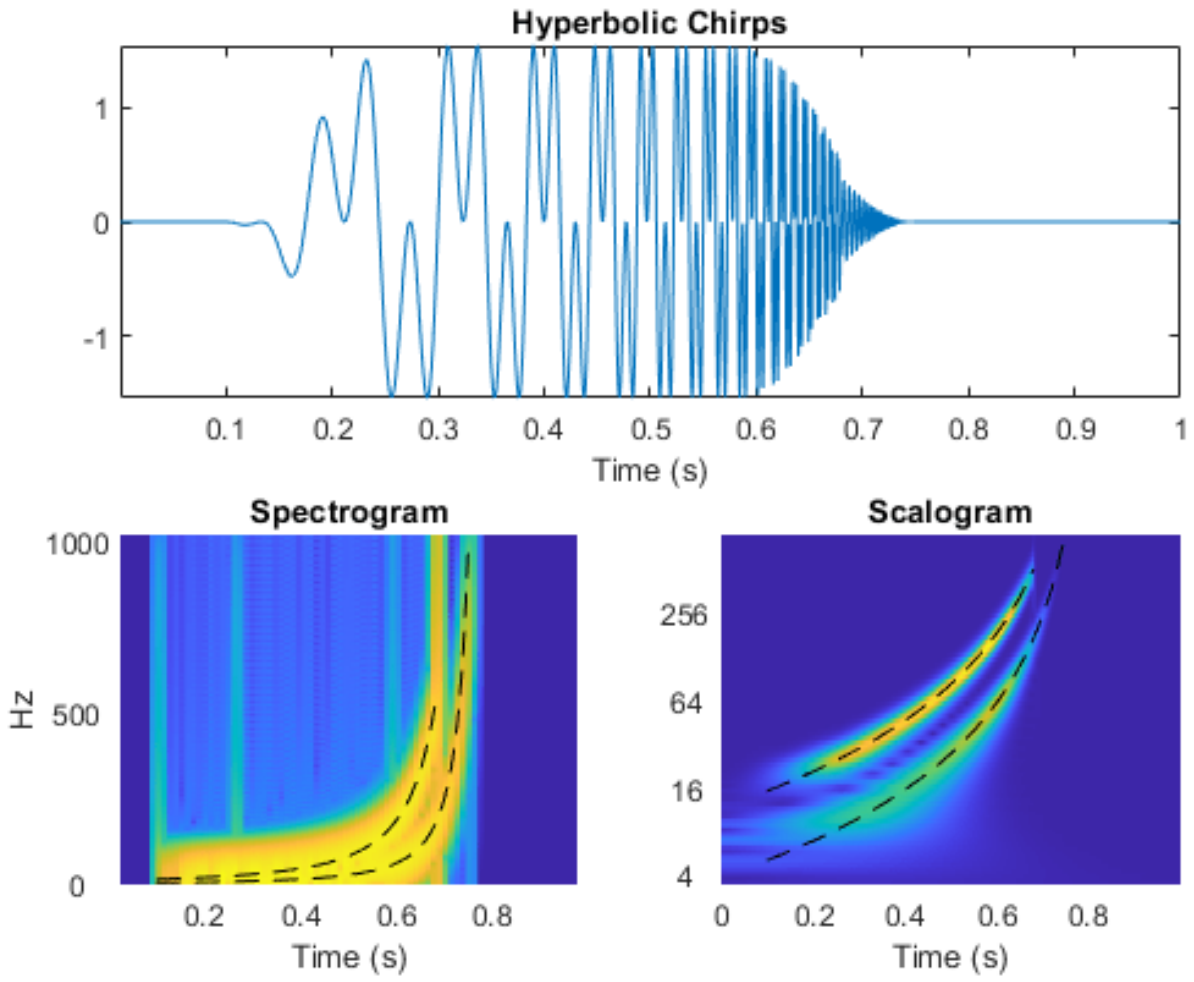
Noise-reduced data

- Iterative procedure



SCALOGRAM

Scalogram

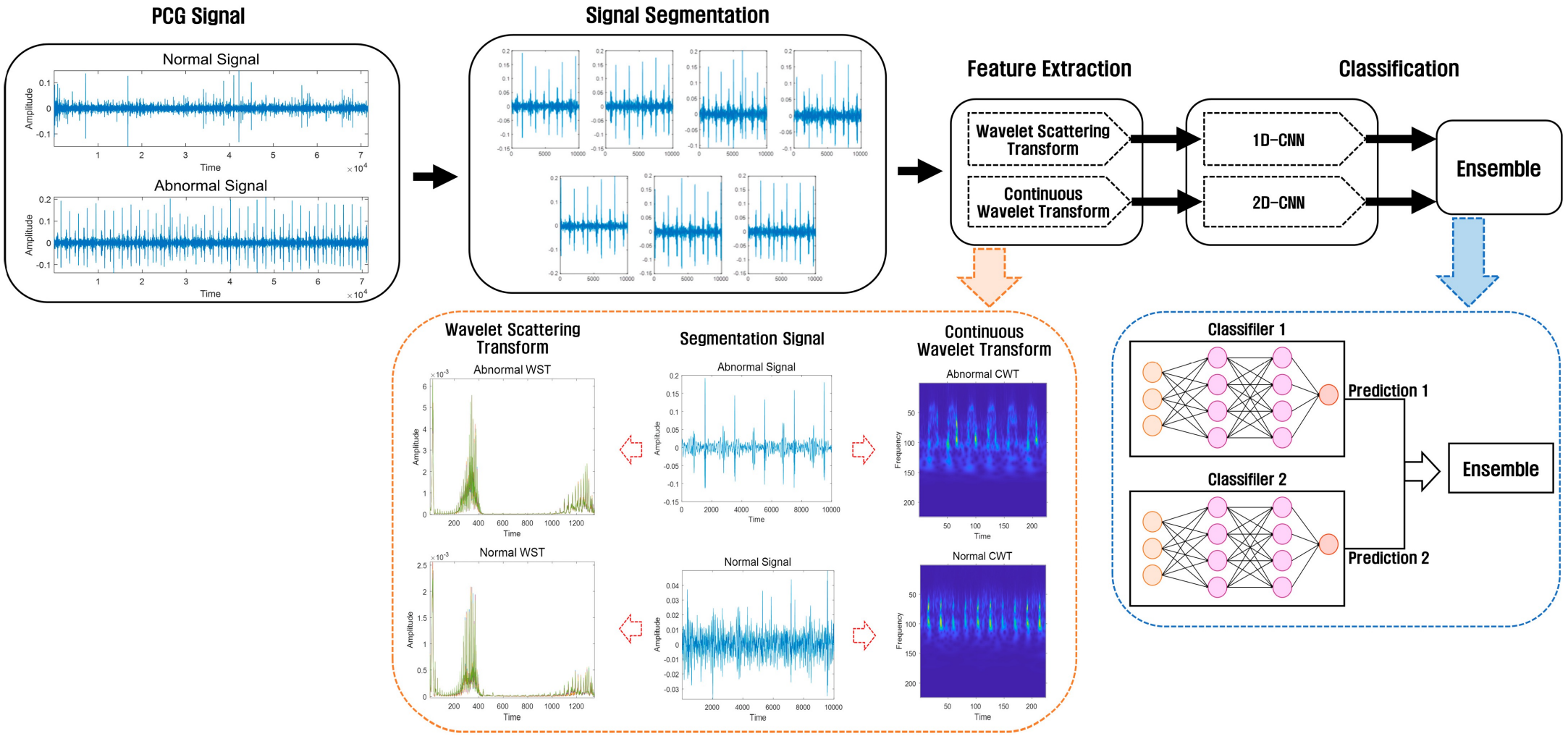


- Peak reflects existence of a wavelet
- Horizontal ridge means time-consistent frequency
- Vertical ridge means sequence of constant frequency
- Separate ridges show a mixture of several tunes

Source: https://www.mathworks.com/help/wavelet/gs/choose-a-wavelet.html

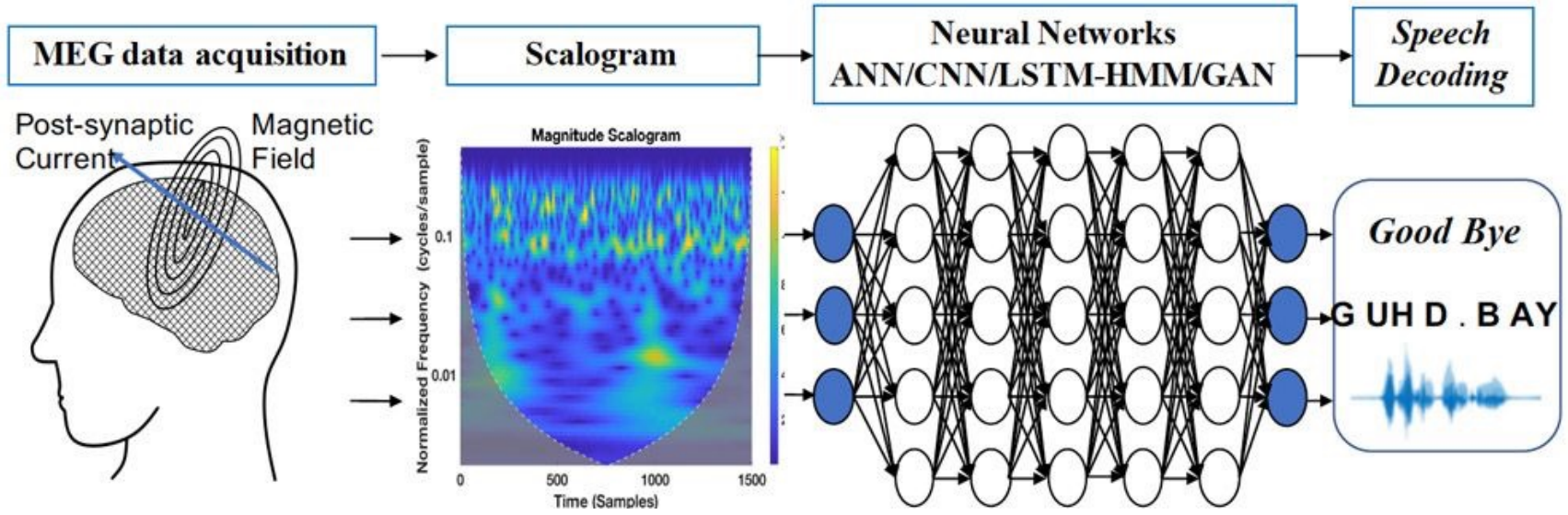
- Heatmap is used for visual representation of the spectrum of frequency
- The entire time series is analyzed with FWT to extract peaks
- Both seasonality (frequency) and holiday effects (positions) are present
- Good for non-stationary and noisy signals e.g. speech, EEG, and brain waves

Heart Sound Classification via Wavelets



Lee, J.-A., & Kwak, K.-C. (2023). Heart Sound Classification Using Wavelet Analysis Approaches and Ensemble of Deep Learning Models. *Applied Sciences*, *13*(21), 11942.

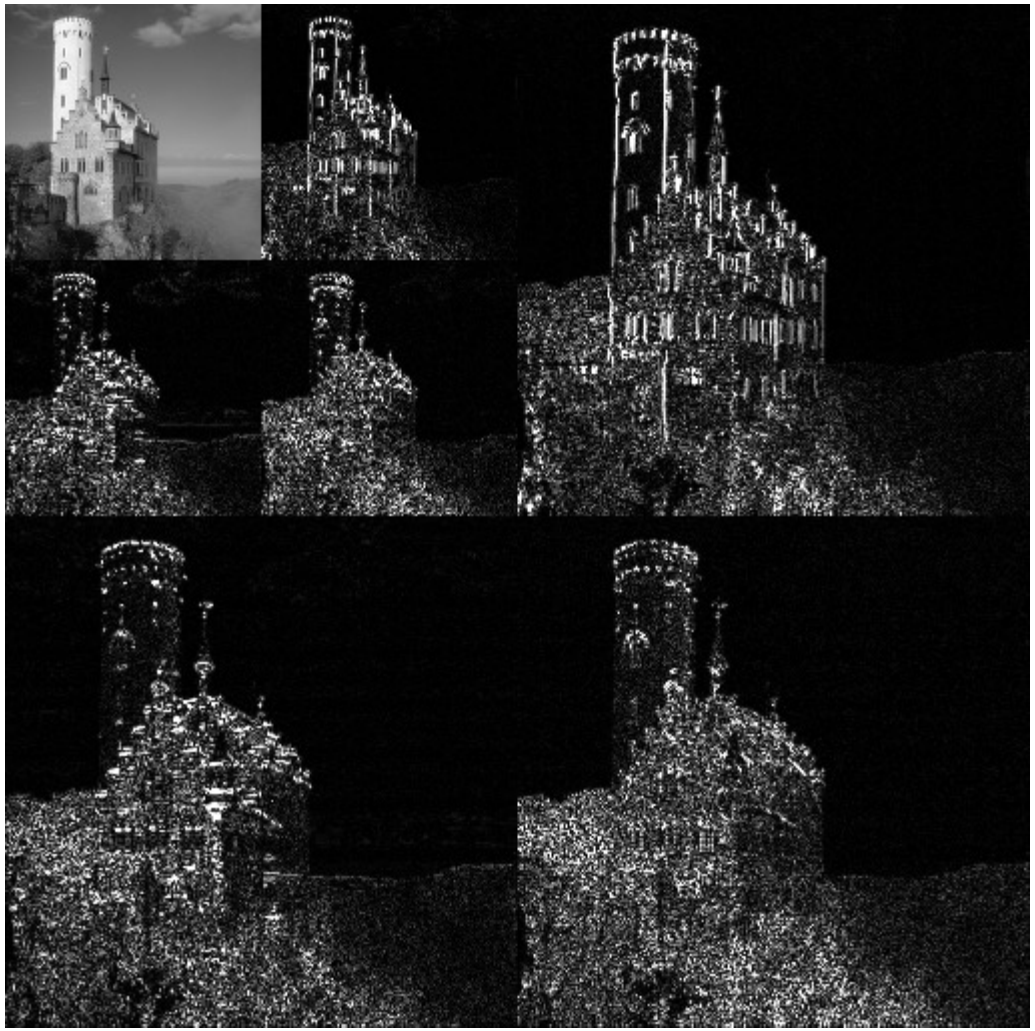
Brain Wave-to-Word Classification



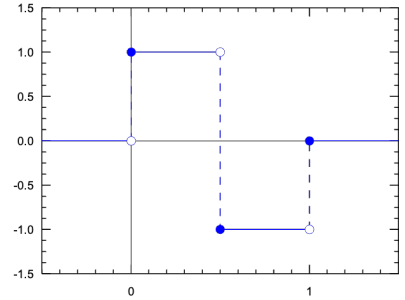
- Scalogram is used as an input picture for CNN-based models
- Stimulations and background noises are preserved
- Conv2D is usually employed

Source: https://www.mathworks.com/company/user_stories/ut-austin-researchers-convert-brain-signals-to-words-and-phrases-using-wavelets-and-deep-learning.html

From Wavelet to JPEG



- Each wavelet of different resolutions extracts pixel changes in the image via Haar wavelet

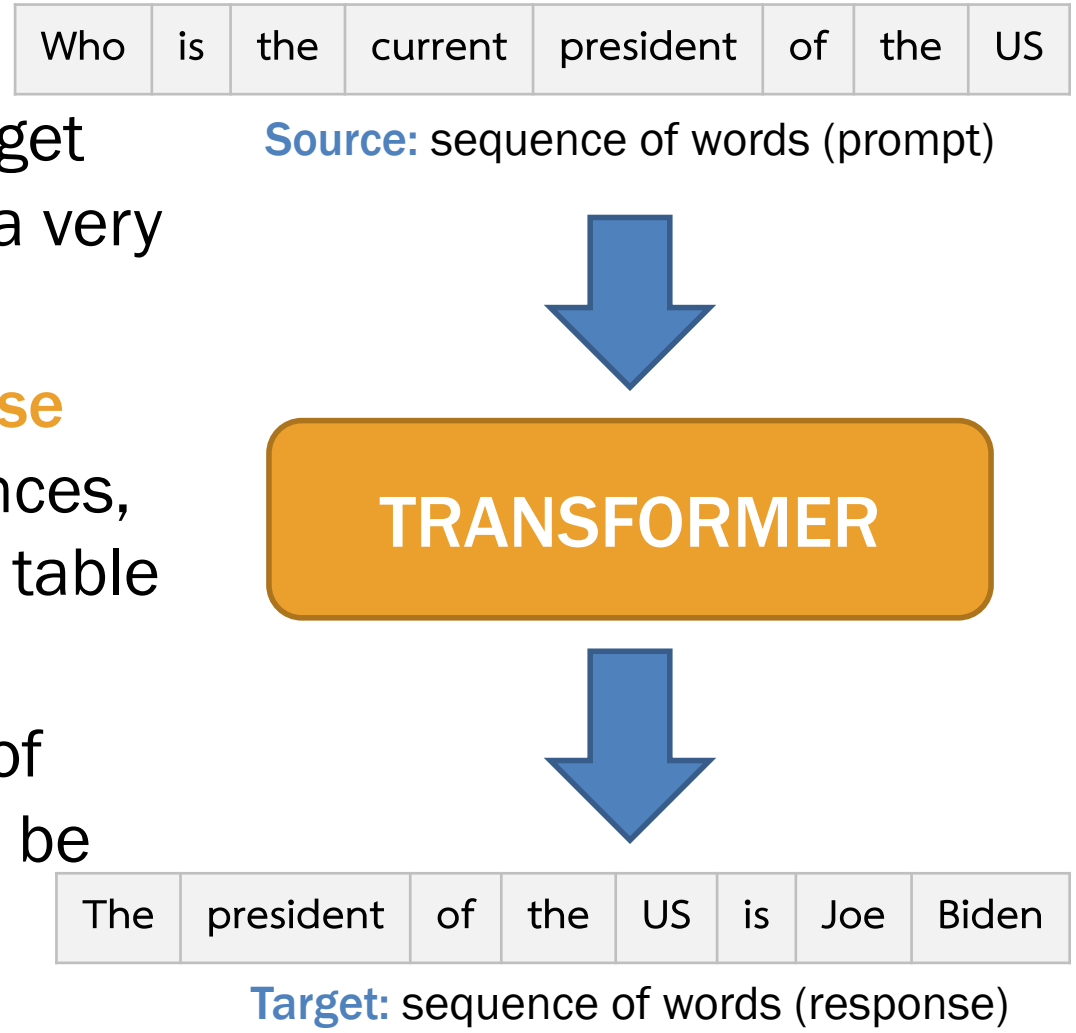


- Compression becomes multiresolution extraction via fast wavelet transform
- Image can be reconstructed by combining these pixel changes

4. Transformer for Time Series

Transformer Model (Vaswani et al., 2016)

- Sequence-to-sequence generation
  - **Translation:** It learns how to produce a target sequence from a source sequence, given a very large dataset of sequence pairs
  - **Pros:** It learns **word collocations** and **phrase structures** on the input and output sequences, and associates them cross-lingually in the table of **translation alignments**
  - **Cons:** It consists of an expansive amount of neuron cells, and the training process can be quite time-consuming



Scaled Dot-Product Attention

- Semantic similarity ⇒ search engine
  - Query is compared against each key with dot product
  - The more similar the key is to the query, the more weight its value will get

Simple Form

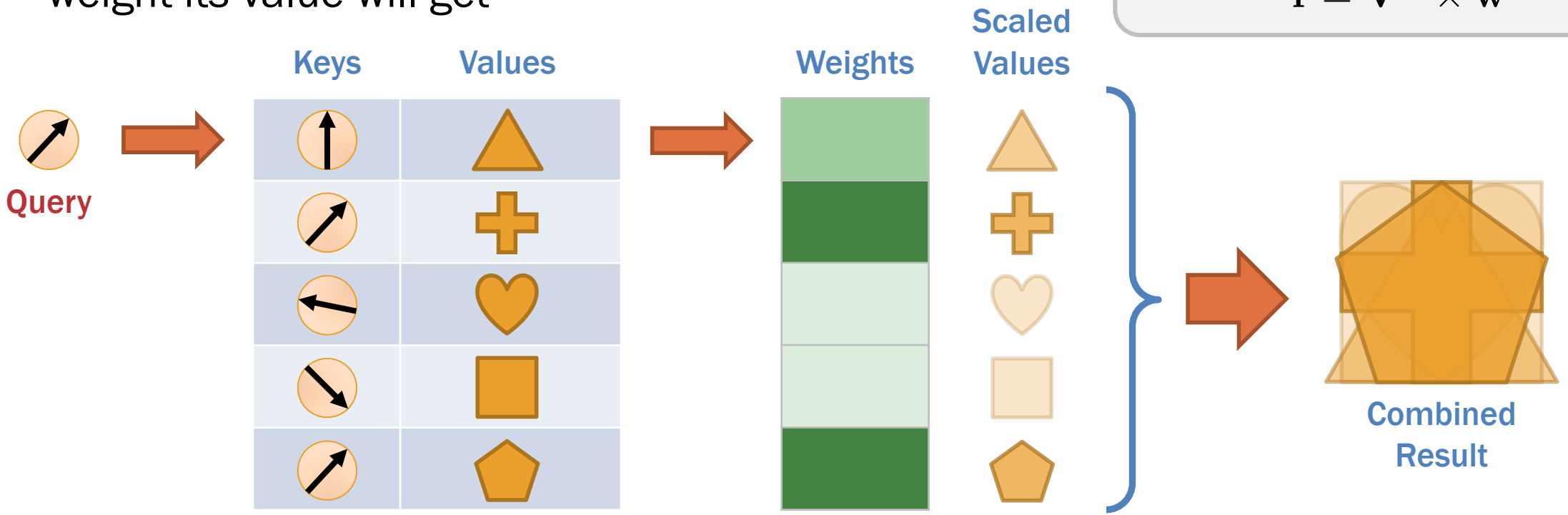
$$w_i \propto \mathbf{k}_i \cdot \mathbf{q}$$

$$\mathbf{r} = \sum_{i=1}^{N} w_i \mathbf{v}_i$$

Matrix Form

$$\mathbf{w} = \text{Softmax}(\mathbf{K} \times \mathbf{q})$$

$$\mathbf{r} = \mathbf{V}^\top \times \mathbf{w}$$



Scaled Dot-Product Attention

- Semantic similarity ⇒ search engine
  - Query is compared against each key with dot product
  - The more similar the key is to the query, the more weight its value will get

Simple Form

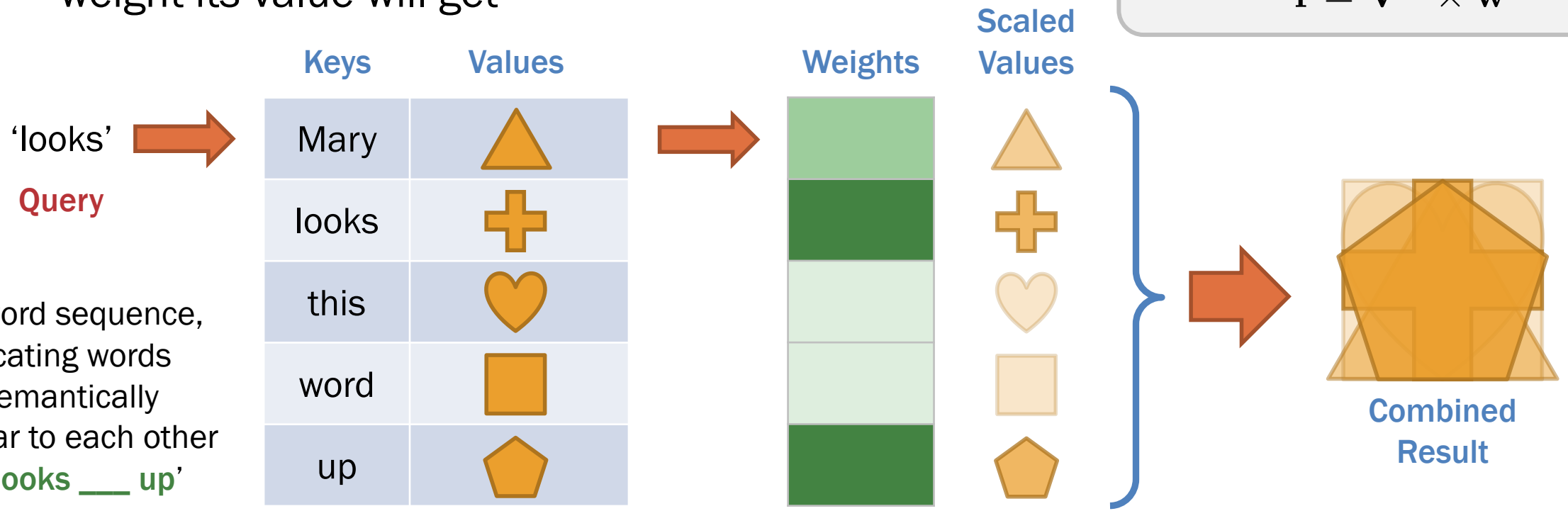
$$w_i \propto \mathbf{k}_i \cdot \mathbf{q}$$

$$\mathbf{r} = \sum_{i=1}^{N} w_i \mathbf{v}_i$$

Matrix Form

$$\mathbf{w} = \mathrm{Softmax}(\mathbf{K} \times \mathbf{q})$$

$$\mathbf{r} = \mathbf{V}^\top \times \mathbf{w}$$



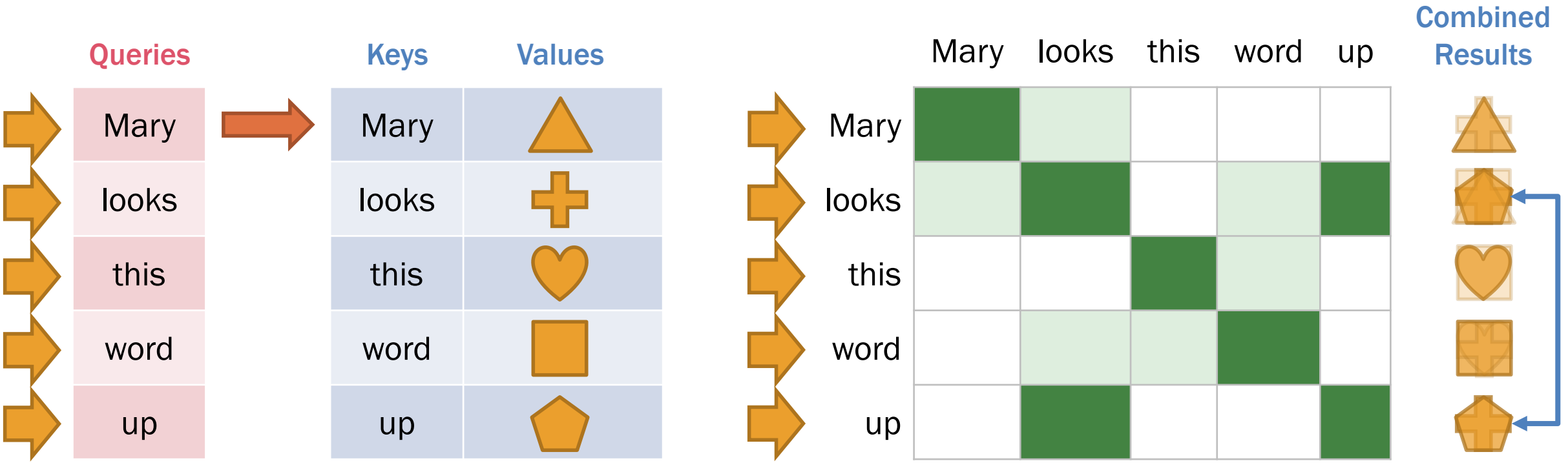
For word sequence, collocating words are semantically similar to each other e.g. '**looks ___ up**'

Self-Attention

- Scaled dot-product attention whose queries and keys are the same
- Collocations will have almost similar results

Matrix Form

$$\mathbf{W} = \mathrm{Softmax}(\mathbf{K} \times \mathbf{K}^\top)$$
$$\mathbf{R} = \mathbf{W} \times \mathbf{V}$$

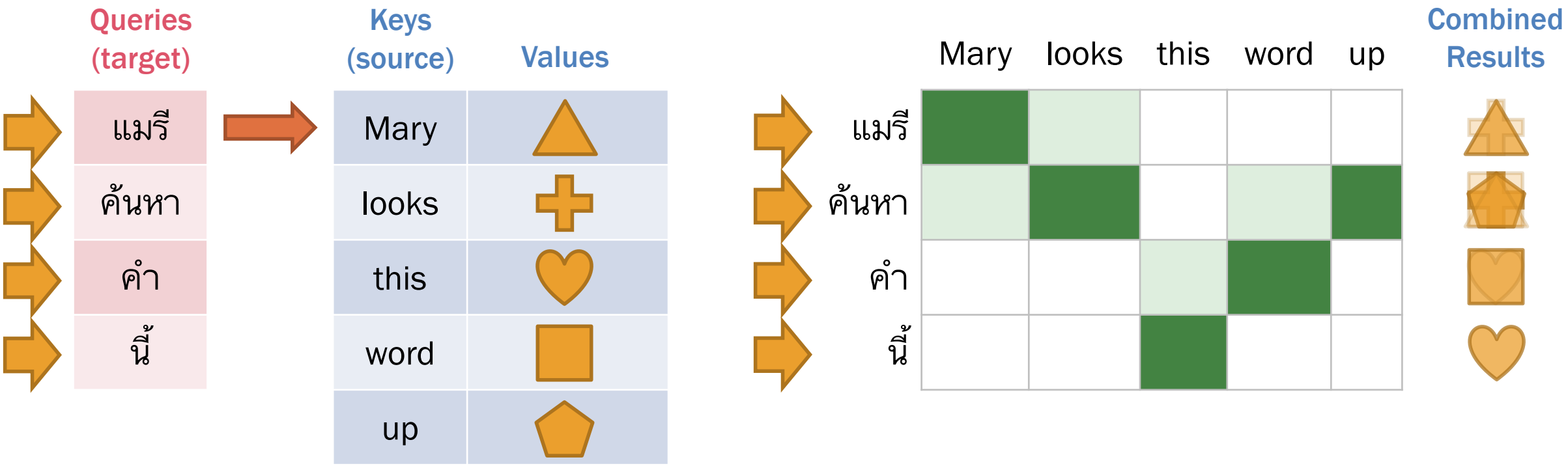


Cross-Attention

- Scaled dot-product attention whose queries are the target and whose keys are the source
- Collocation alignment via semantic similarity

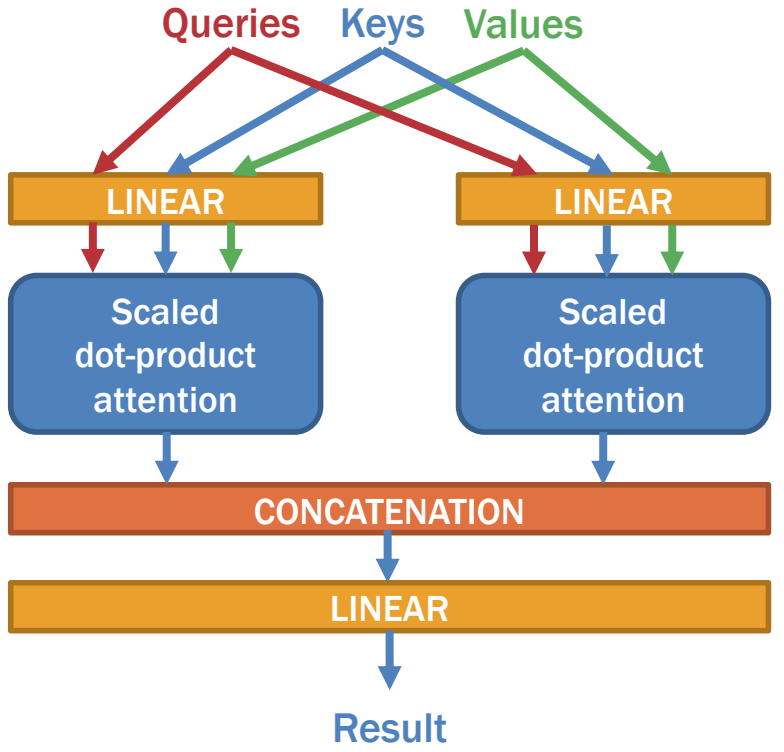
Matrix Form

$$\mathbf{W} = \mathrm{Softmax}(\mathbf{Q} \times \mathbf{K}^\top)$$
$$\mathbf{R} = \mathbf{W} \times \mathbf{V}$$

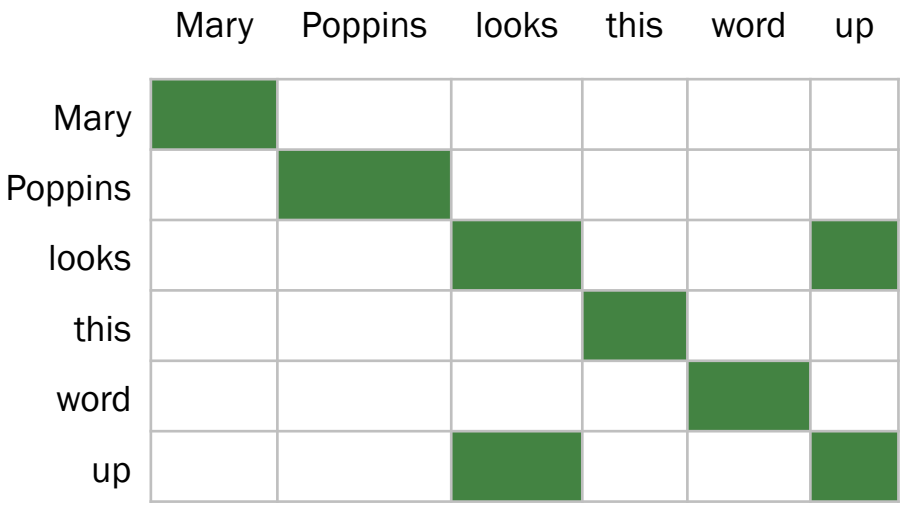


Multihead Attention

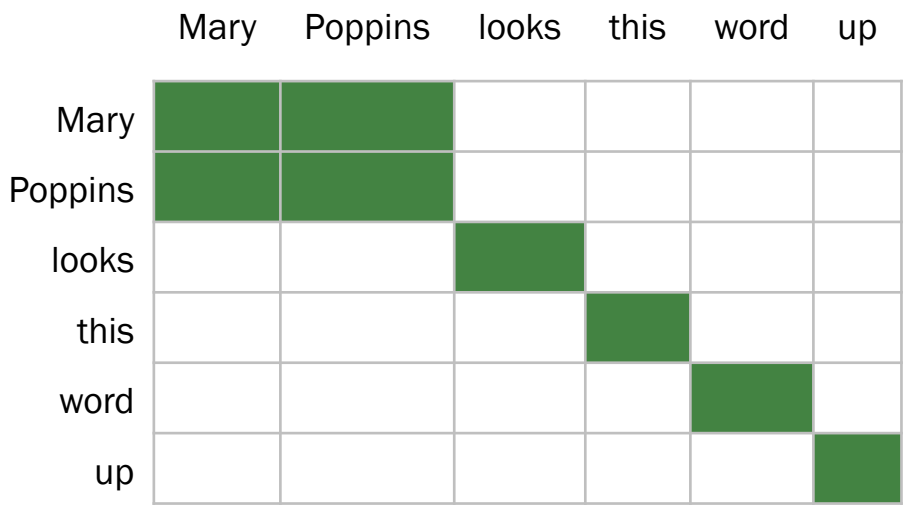
- Scaled dot-product attention has a drawback
  - It recognizes **only one** type of word collocation
  - If we assume more than one type of word collocation per sequence, then we have to combine multiple attention heads [default = 8 heads]

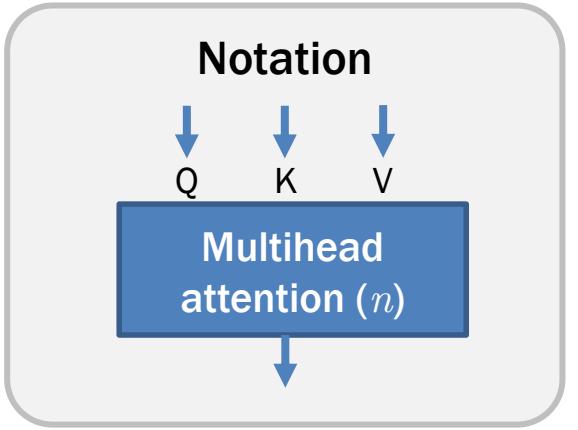


HEAD 1 (looks ___ up)



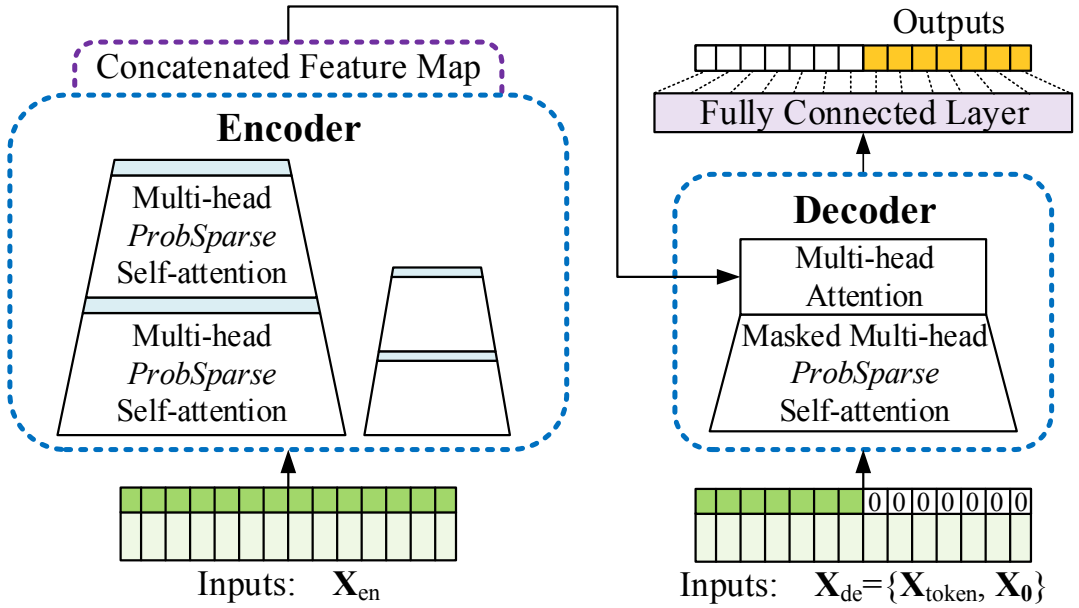
HEAD 2 (Mary Poppins)





Informer

(Zhou et al., AAAI-2021)

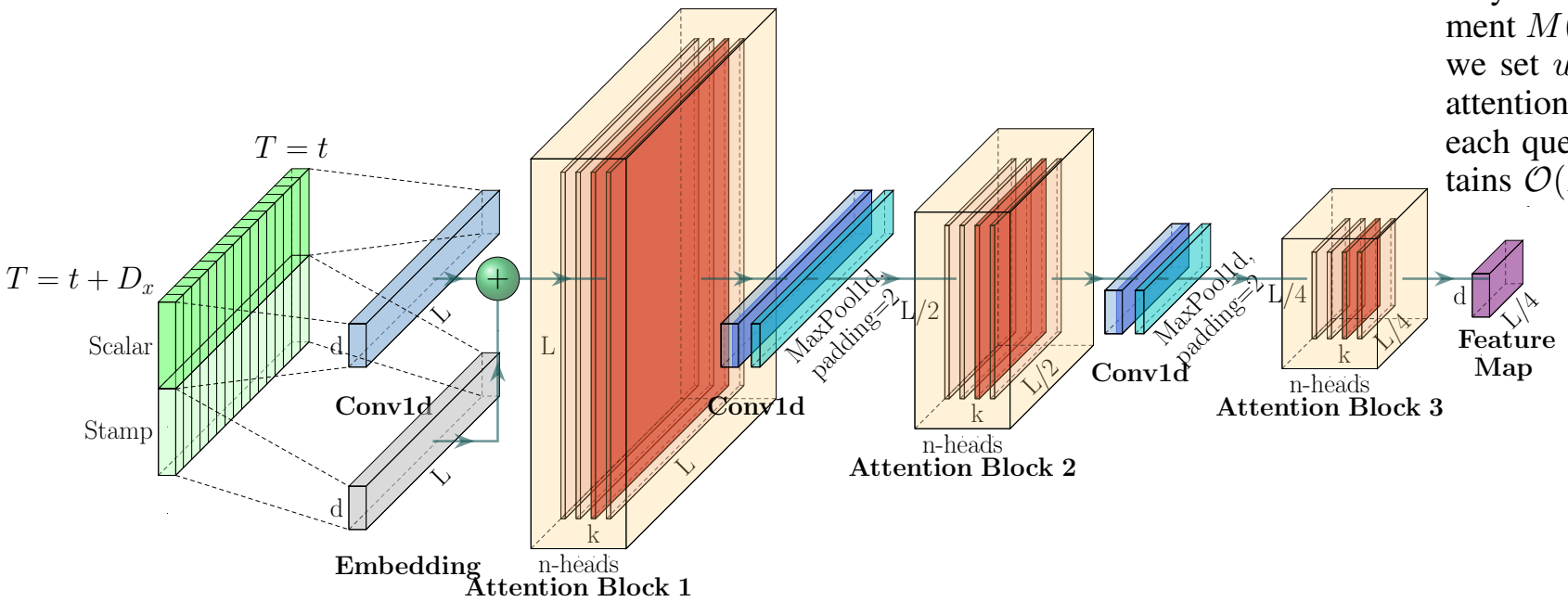


Overview

***ProbSparse* Self-attention** Based on the proposed measurement, we have the *ProbSparse* self-attention by allowing each key to only attend to the u dominant queries:

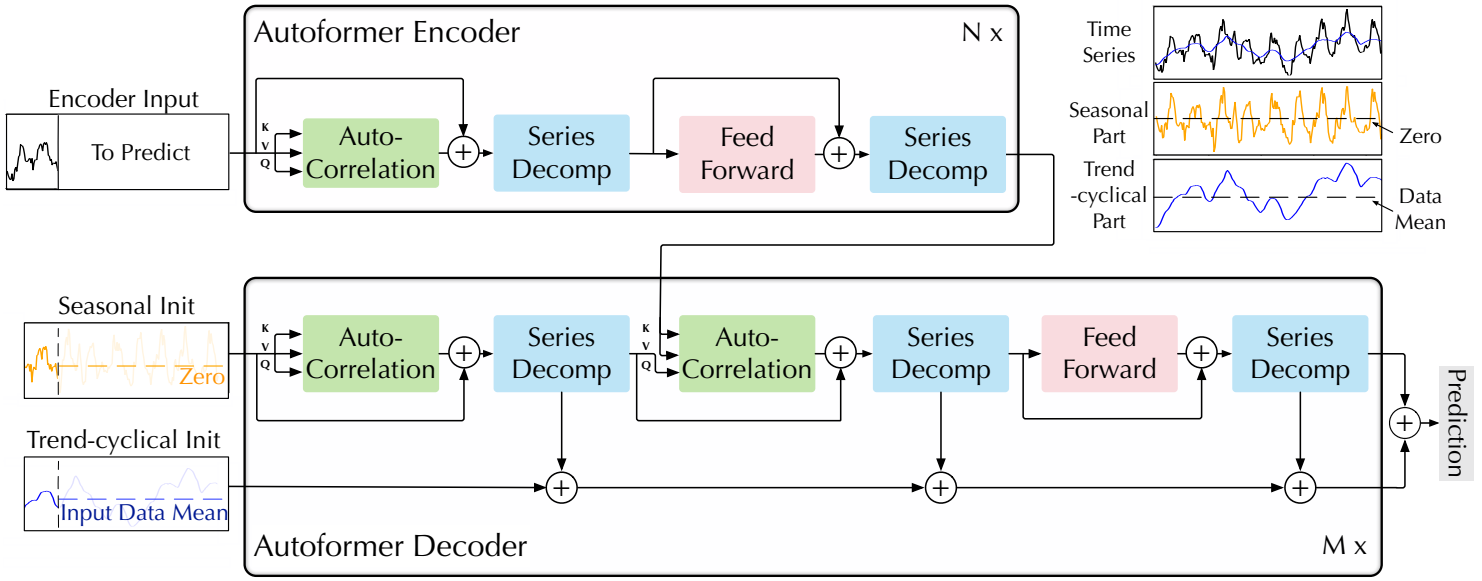
$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\overline{\mathbf{Q}}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V} \quad , \qquad (3)$$

where $\overline{\mathbf{Q}}$ is a sparse matrix of the same size of $\mathbf{q}$ and it only contains the Top-u queries under the sparsity measurement $M(\mathbf{q},\mathbf{K})$. Controlled by a constant sampling factor c, we set $u = c \cdot \ln L_Q$, which makes the *ProbSparse* self-attention only need to calculate $\mathcal{O}(\ln L_Q)$ dot-product for each query-key lookup and the layer memory usage maintains $\mathcal{O}(L_K \ln L_Q)$.



Encoder Block

Autoformer (Wu et al., NIPS-2021)

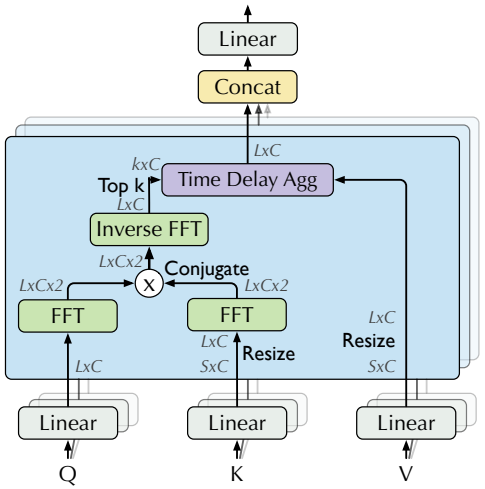


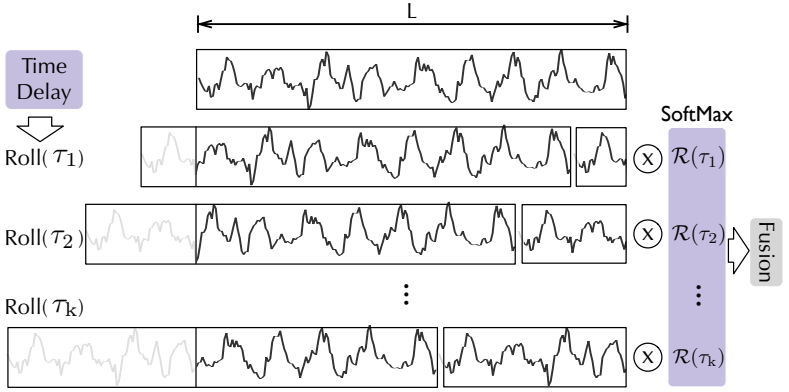
SeriesDecomp

$$\mathcal{X}_{\mathrm{t}} = \mathrm{AvgPool}(\mathrm{Padding}(\mathcal{X}))$$
$$\mathcal{X}_{\mathrm{s}} = \mathcal{X} - \mathcal{X}_{\mathrm{t}},$$

Output = $(X_{\mathrm{s}},\ X_{\mathrm{t}})$

Autocorrelation Block

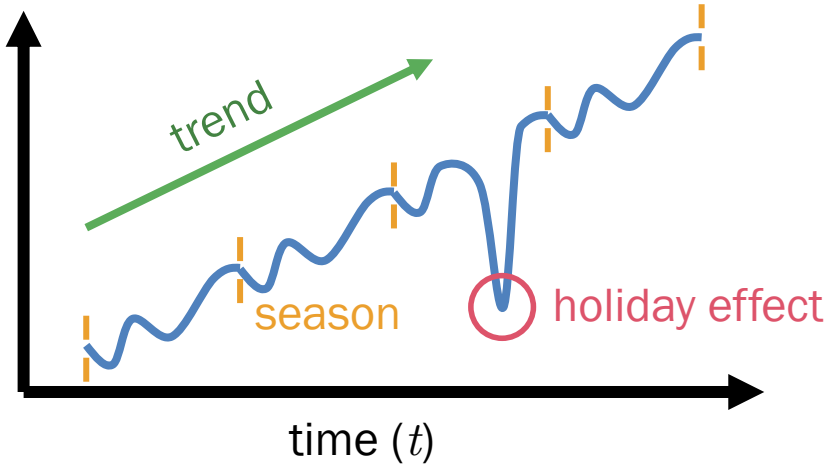




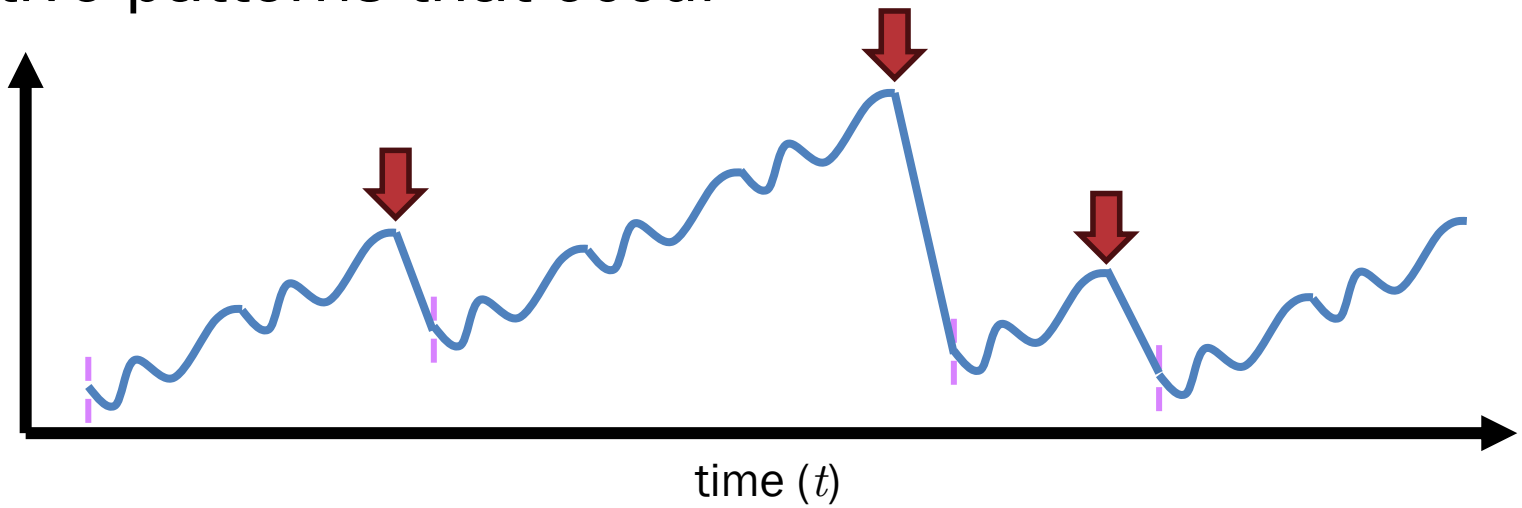
5. Conclusion

Patterns in Time Series

- **Trend:** general direction over time
- **Seasonality:** repetitive patterns that occur at regular predictable intervals
- **Holiday effects:** irregular patterns caused by special calendar events
- **Cycle:** long-term repetitive patterns that occur at irregular intervals



Cycle
1. Slow increase
2. Catastrophe
3. Rapid decline



Time-Series Models

Models	Trend	Seasonality	Holiday Effects	Cycle	Suitable for Signal Types
ARIMA	**Yes** (learned by linear regression)	**Yes** (limited season length)	**No**	**No**	—
Convolution	**Yes** (learned by CNNs)	**Yes** (limited window size)	**Yes** (learned by CNNs)	**Possibly no** (due to limited window size)	—
Spectral density	**Yes** (learned by CNNs)	**Yes** (limited window size)	**Yes** (learned by CNNs)	**Yes** (present in spectogram)	Stationary
Wavelet analysis	**Yes** (learned by CNNs)	**Yes** (unlimited window size)	**Yes** (learned by CNNs)	**Yes** (present in scalogram)	Stationary and non-stationary
Transformer models	**Yes** (learned by attention)	**Yes** (limited time delay)	**Yes** (learned by attention)	**Possibly no** (due to limited time delay)	Stationary and non-stationary

Thank You

prachya@siit.tu.ac.th

kaamanita@gmail.com