УНИВЕРСИТЕТ ИТМО

Predictive models for the random processes

Workshop

Nikolay Nikitin, PhD, assistant professor

2022

Plan of the workshop

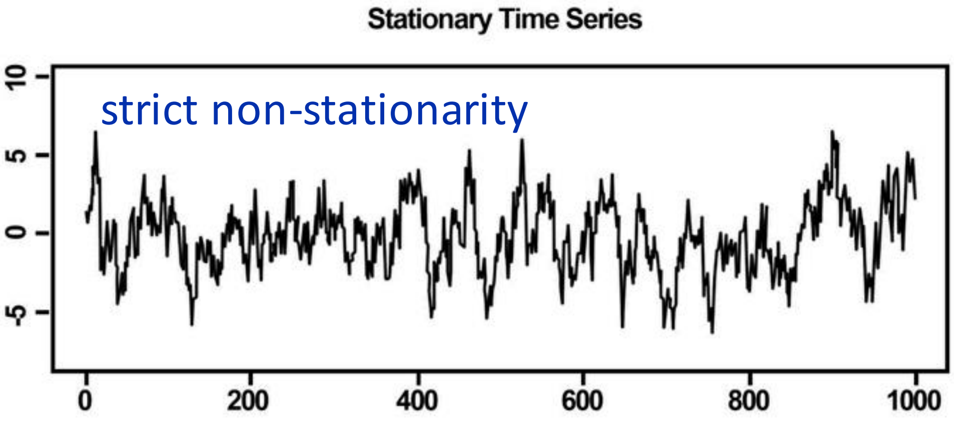
УНИВЕРСИТЕТ ИТМО

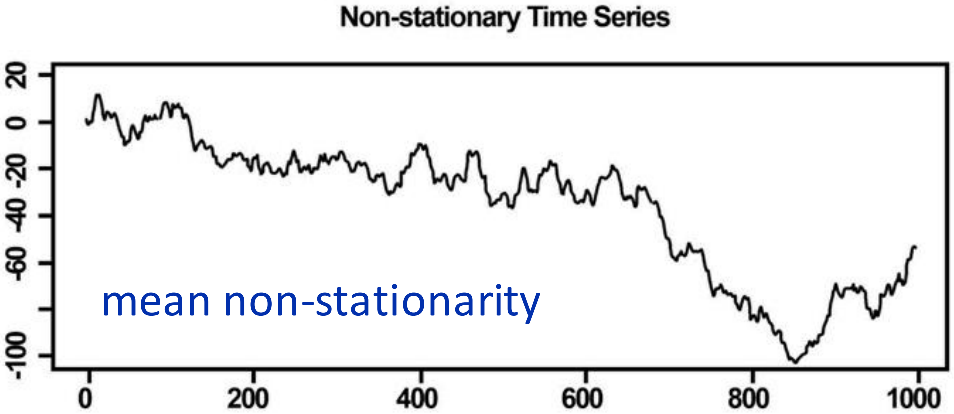
What we want to learn: how to build statistical forecasts

- To analyze **stationarity** of a process (for mathematical expectation and variance);
- To analyze **covariance** function. To define covariance (or correlation window);
- To estimate **spectral density** function with using different functions for spectral window;
- To filter high frequencies (**noise**) with using various **filters** (e.g. moving average, Gaussian filter);
- To repeat estimation of spectral density and compare with result for non-filtered data;
- To build **auto-regression model for** filtered and non-filtered data. To analyze residual error and to define appropriate order of model; Compare different approach for hyperparameter tuning;
- To find **additional factors that** influence on chosen varaiable;
- To analyze **mutual correlation** functions among factors;
- To build model in a form of **linear dynamical system** using additional factors. To analyze residual error and to define appropriate order of model.

Analyze stationarity of a process

УНИВЕРСИТЕТ ИТМО





Stationarity – the constant of mean and variance over time.

<u>**Non-stationary (by mean) series generation**</u>

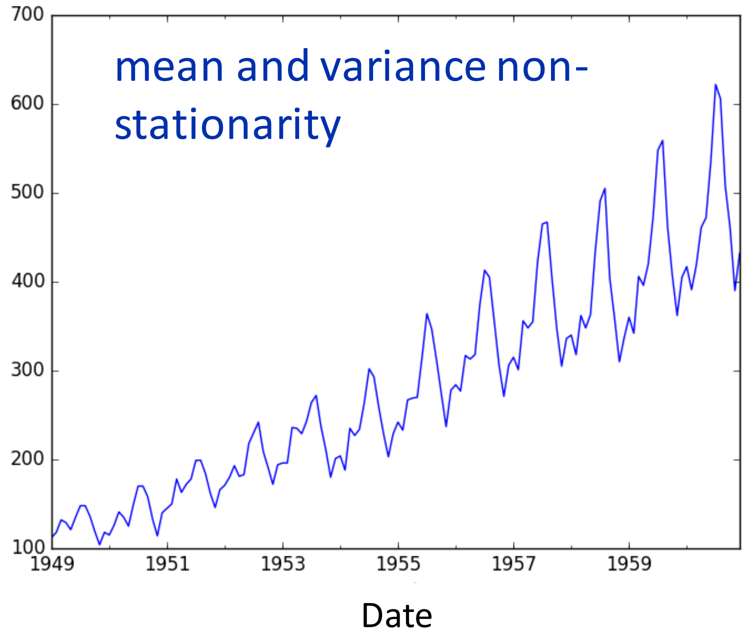
y_=np.random.uniform(-1,1,[n])

mu=0
sigma=0.01
e= np.random.normal(mu, sigma, n)
t = [v/1000 for v in range(0,n)]
y=y_+e+t

plt.plot(x,y)
plt.show()

<u>**(Stationarity.ipynb)**</u>

Mean and variance non-stationarity

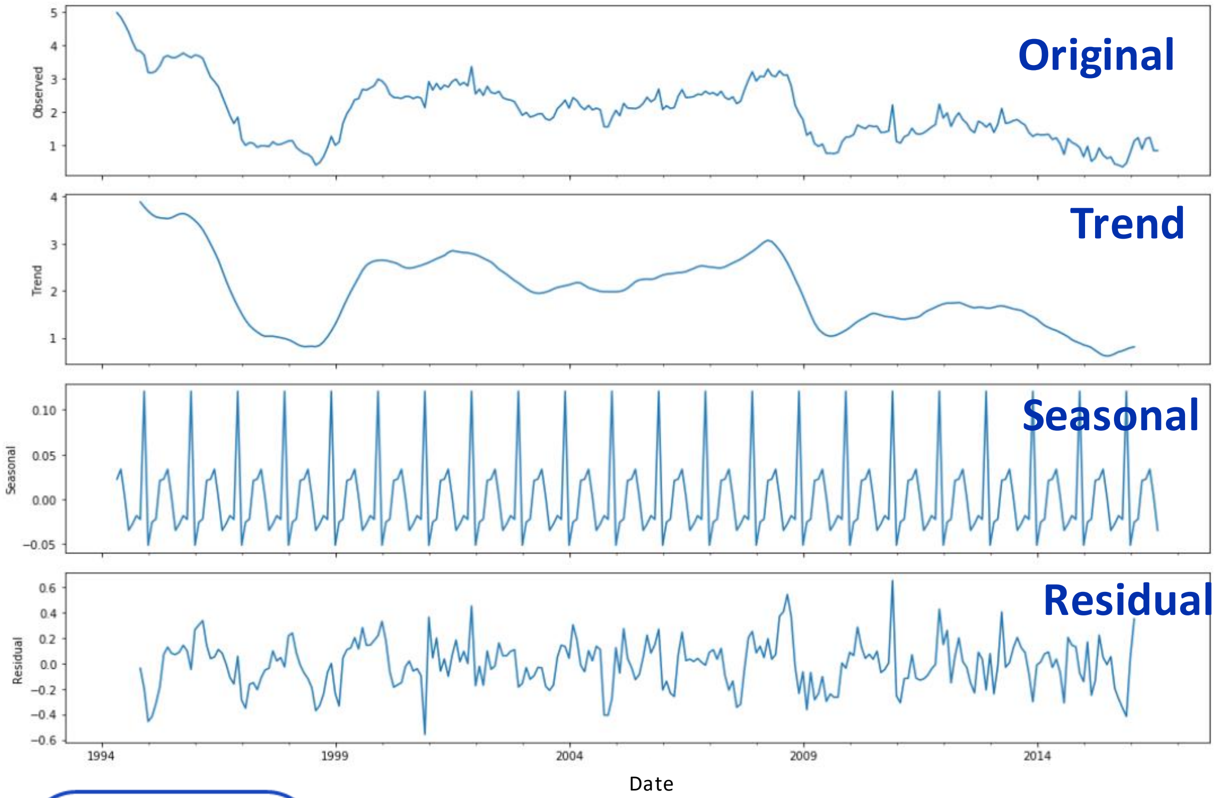


Non-stationary (by mean and variance) series generation

```
y_=np.random.uniform(-1,1,[n])

mu=0
sigma=0.01
e = np.zeros(n)
for i in range(n):
    e[i]= np.random.normal(mu, sigma+i/500, 1)
t = [v/200 for v in range(0,n)]
y=y_+e+t

plt.plot(x,y)
plt.show()
```

(Stationarity.ipynb)

Non-stationary process decomposition



Components:

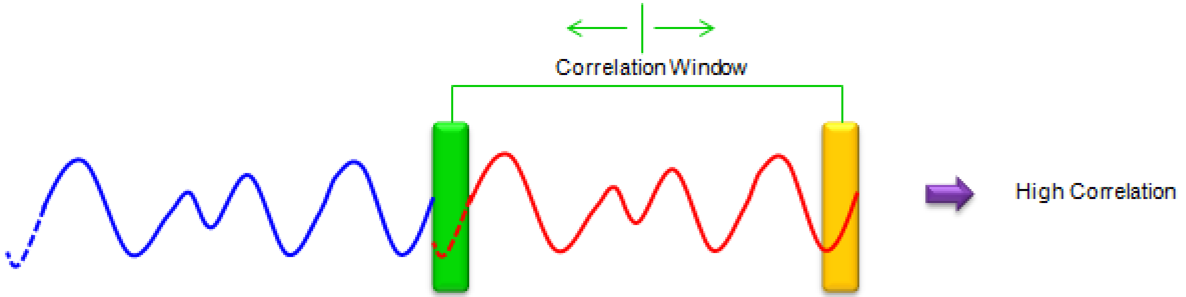
- **Trend** - long-term time series change;
- **Seasonality** – time series changes with constant period;
- **Cyclic** - time series changes with variable period;
- **Residuals —** a component that is left after other components have been calculated and removed from time series data.

Stationarity analysis and decomposition (example)

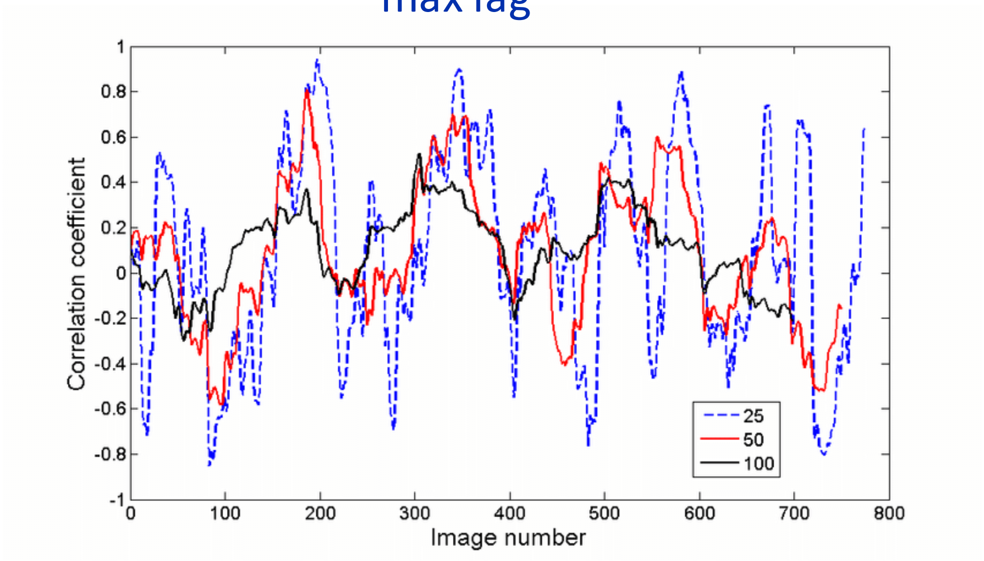
[Stationarity.ipynb](https://colab.research.google.com/drive/10fQ4421jhxjuRNMGBhF4FF_4y01jrK3s)

https://colab.research.google.com/drive/10fQ4421jhxjuRNMGBhF4FF_4y01jrK3s

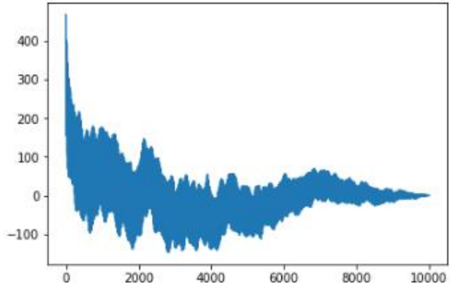
Analyze covariance function with window



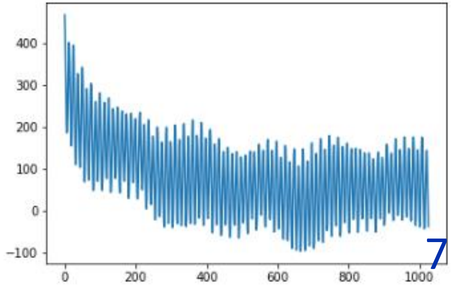
max lag



The finite sliding correlation window allow to analyse the correlation change through time.



Covariance function with different window



Estimate spectral density function

Reminder from the lecture

$$\lambda_1(\tau) = \begin{cases} 1 \text{ при } \tau \le \tau_{max}, \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Bartlett

$$\lambda_2(\tau) = \begin{cases} 1 - \dfrac{|\tau|}{\tau_{max}} \text{ при } 0 \le \tau \le \tau_{max}, \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Bartlett (modified)

$$\lambda_3(\tau) = \begin{cases} 0{,}5\left(1 - \cos\dfrac{\pi\tau}{\tau_{max}}\right) \text{ при } 0 \le \tau \le \tau_{max} \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Hann

$$\lambda_4(\tau) = \begin{cases} 0{,}54 + 0{,}46\cos\dfrac{\pi\tau}{\tau_{max}} \text{ при } 0 \le \tau \le \tau_{max}, \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Hamming

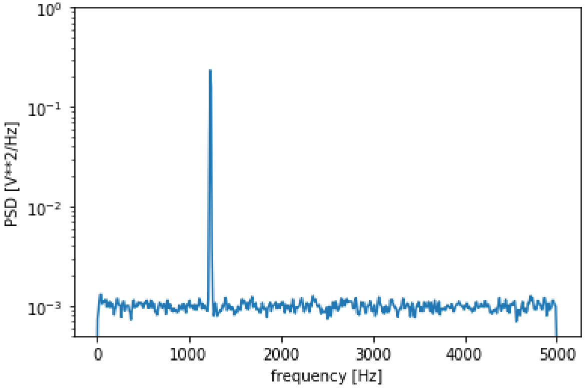
$$\lambda_5(\tau) = \begin{cases} 1 - \left(\dfrac{|\tau|}{\tau_{max}}\right)^g \text{ при } \tau \le \tau_{max},\ g \ge 1 \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Parsen

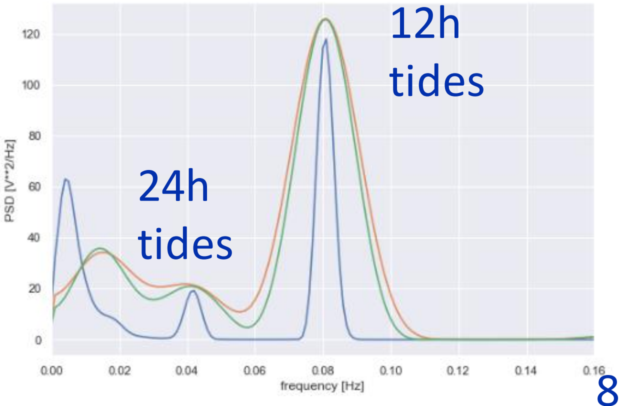
$$\lambda_6(\tau) = \begin{cases} 1 - 6\left(\dfrac{\tau}{\tau_{max}}\right)^2 + 6\left(\dfrac{\tau}{\tau_{max}}\right)^3 \text{ при } 0 \le \dfrac{\tau_{max}}{} \\ 2\left(1 - \dfrac{\tau}{\tau_{max}}\right)^3 \text{ при } \dfrac{1}{2}\tau_{max} \le \tau \le \tau_{m} \\ 0 \text{ при } \tau > \tau_{max} \end{cases}$$

Parsen (2)

Spectral density estimation using Barlett function



Spectral density estimation for the real data (sea surface height)



ITMOre than a UNIVERSITY

Spectral density function estimation (example)

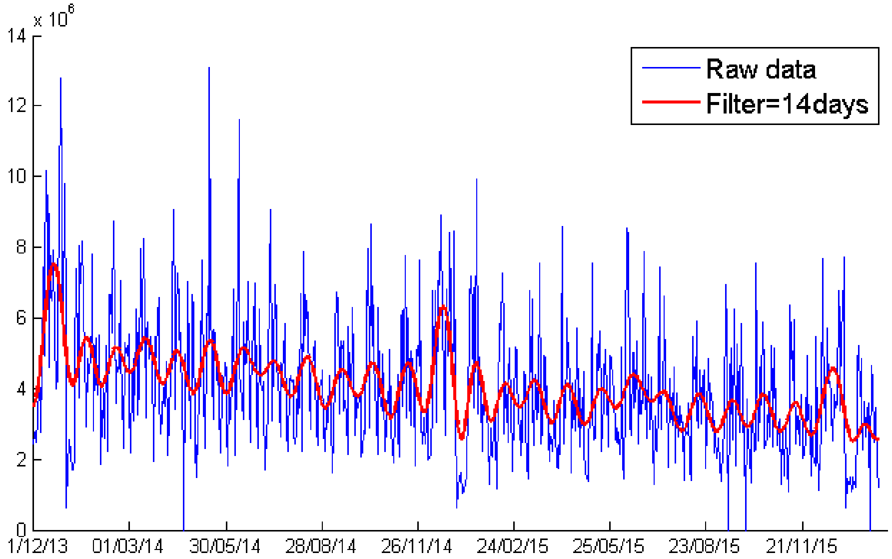
Open the autocov.ipynb

https://colab.research.google.com/drive/1PcBwAvA8KHIWGjNjYqK37rG6lN0547xu

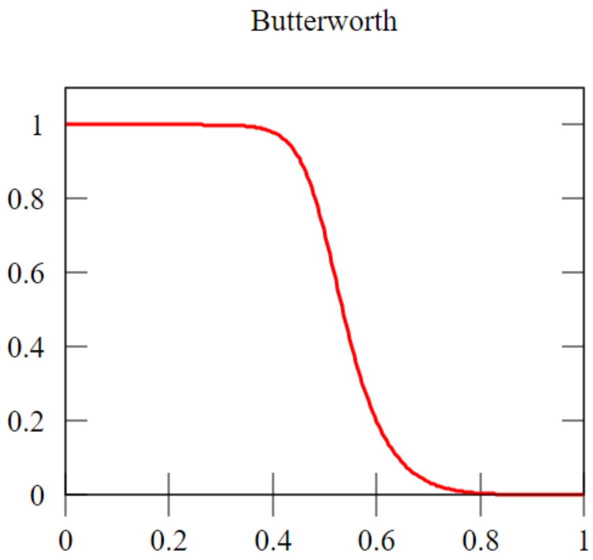
Filter high frequencies (noise)

УНИВЕРСИТЕТ ИТМО

Rolling window filtering



Reminder for lecture



Butterworth filter

ITsMOre than a UNIVERSITY

Filtering example

УНИВЕРСИТЕТ ИТМО

Open the filtering.ipynb
https://github.com/Dreamlone/ITMO_materials/blob/master/fedot-workshop/Time_series_filters.ipynb

In this example, the features of
FEDOT framework
are used.

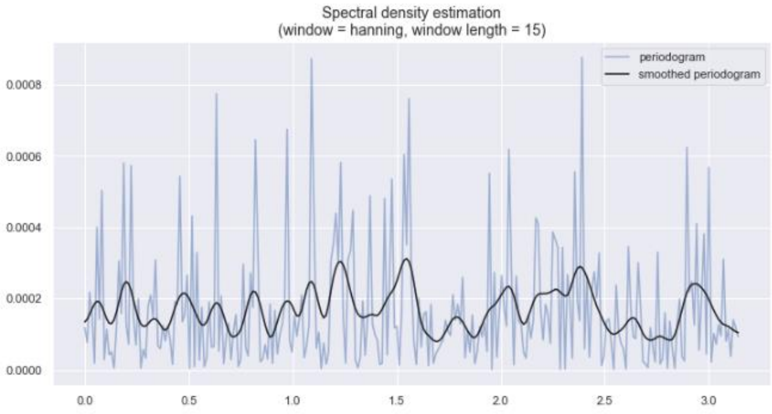
FEDOT

https://github.com/aimclub/FEDOT

ITsMOre than a UNIVERSITY

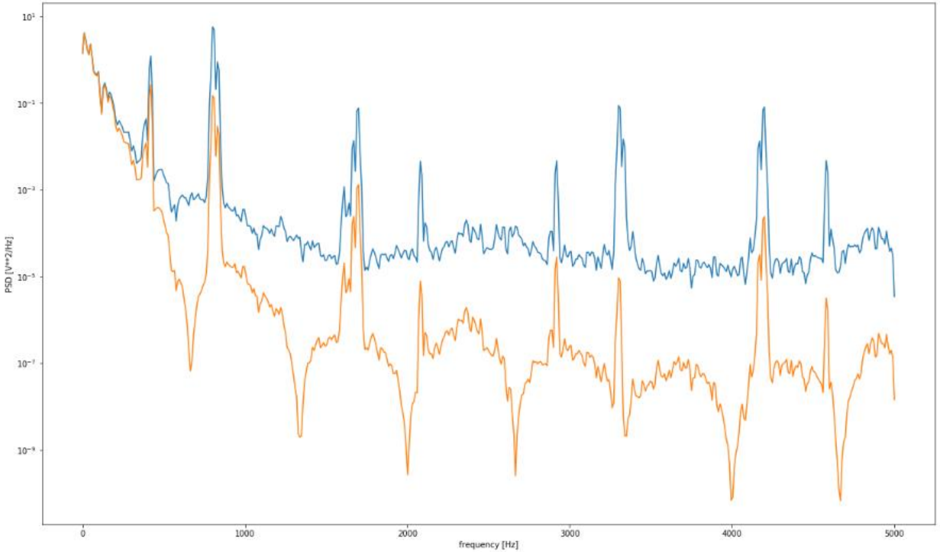
Spectral density for filtered data

Spectral density



frequency

Periodogramm without filtering (blue) and with moving average filter (black).



Spectral density without filtering (blue) and with moving average filter (yellow).

ITsMOre than a UNIVERSITY

Auto-regression model

Shows the best results with time series with clear seasonality and low noise levels;

Requires customization of parameters for each individual case;

AR(p): $$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t.$$

MA(q): $$y_t = \alpha + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

ARMA(p,q): $$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}.$$

ARIMA (p, d, q) - ARMA for n-times-differentiated time series;

Seasonality: $$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

$$+\phi_S y_{t-S} + \phi_{2S} y_{t-2S} + \cdots + \phi_{PS} y_{t-PS}$$ + P components with period S

$$+\theta_S \varepsilon_{t-S} + \theta_{2S} \varepsilon_{t-2S} + \cdots + \theta_{PS} \varepsilon_{t-QS}.$$ + Q components with period S

SARMA(p,q)x(P,Q)

Prediction quality metrics

R^2 – explained variance

```
from sklearn.metrics import r2_score

print("Linear Regression R^2:",  round(r2_score(y, y_pred_lr), 3))
print("SMA R^2:",  round(r2_score(y , y_sma), 3))
```

```
Linear Regression R^2: 0.942
SMA R^2: 0.822
```

MSE/RMSE

```
from sklearn.metrics import mean_squared_error

print("Linear Regression MSE:", round(mean_squared_error(y, y_pred_lr), 3))
print("SMA MSE:",  round(mean_squared_error(y , y_sma), 3))
```

```
Linear Regression MSE: 1882343.713
SMA MSE: 5774211.042
```

MAPE

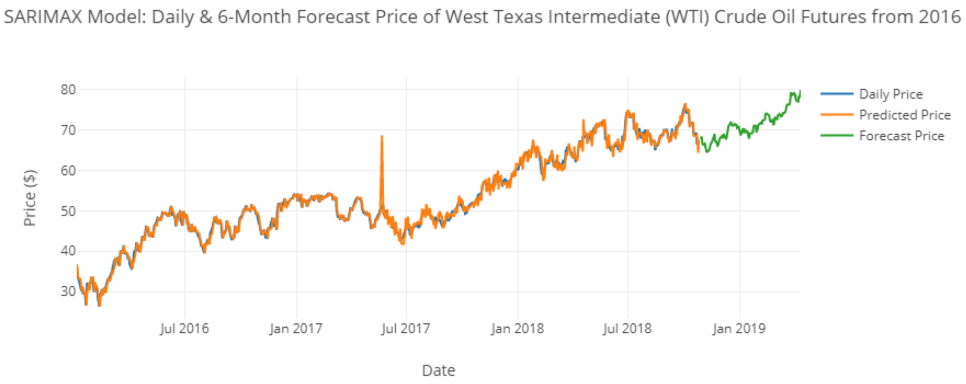
```
def mean_absolute_percentage_error(y_true, y_pred):
    return round(np.mean(np.abs((y_true - y_pred) / y_true)) * 100, 3)

print("Linear Regression MAPE:",  mean_absolute_percentage_error(y, y_pred_lr))
print("SMA MAPE:", mean_absolute_percentage_error(y , y_sma))

```

```
Linear Regression MAPE: 4.0
SMA MAPE: 22.493
```

Additional factors

УНИВЕРСИТЕТ ИТМО

Example:




```python
def predict_arima(trained_model, predict_data: InputData) -> OutputData:
    start, end = trained_model.nobs, \
                 trained_model.nobs + len(predict_data.target) - 1
    exog = predict_data.features

    if trained_model.data.endog is predict_data.target:
        # if train sample used
        start, end = 0, len(predict_data.target)

    prediction = trained_model.predict(start=start, end=end,
                                       exog=exog)

    return prediction[0:len(predict_data.target)]
```

Variables:

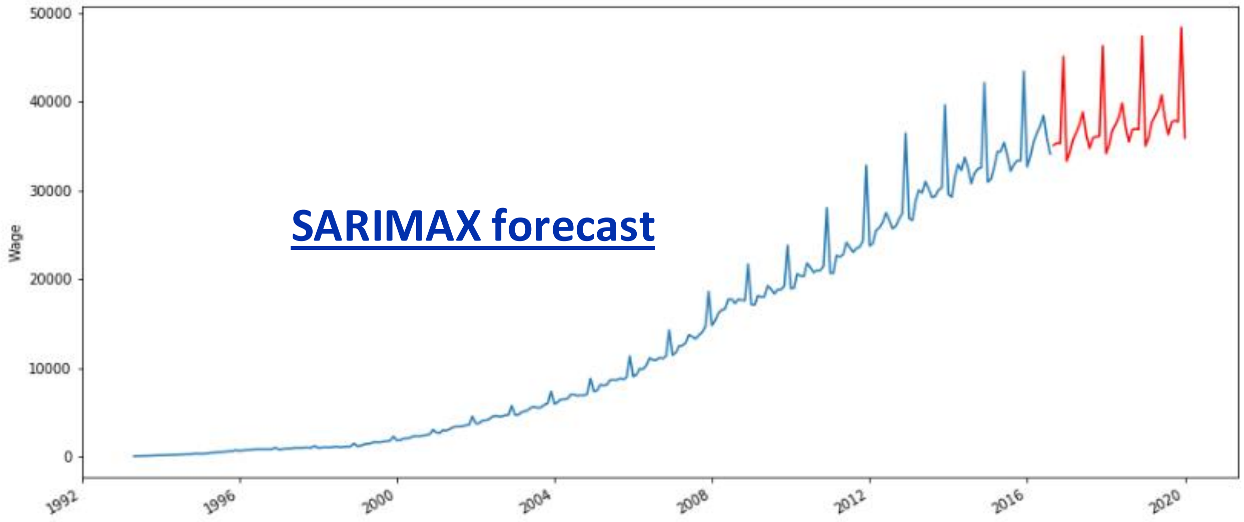
1. Date - Daily based on Business Days
2. **Price - Daily Closing Price – predictor**
3. Open - Daily Opening Price
4. High - Intraday Maximum Price
5. Low - Intraday Minimum Price
6. Volume - # of futures traded
7. % Change - Percent change from previous day's closing price

```python
def fit_arima(train_data: InputData, params):
    return ARIMA(train_data.target, **params,
                 exog=train_data.features).fit(disp=0)
```

SARIMAX - Seasonal AutoRegressive Integrated Moving Average with **eXogenous regressors**

ITsMOre than a UNIVERSITY

SARIMAX example



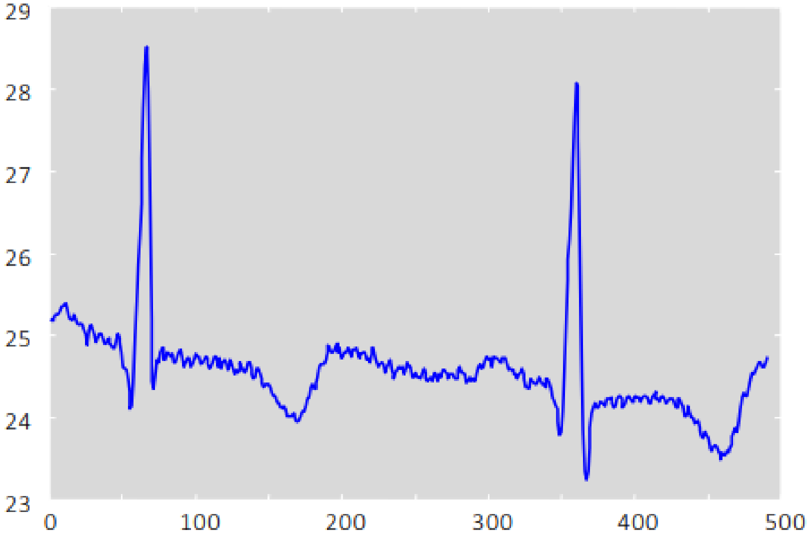
Open the ARIMA_Forecast.ipynb

https://github.com/Dreamlone/ITMO_materials/blob/master/fedot-workshop/ARIMA.ipynb

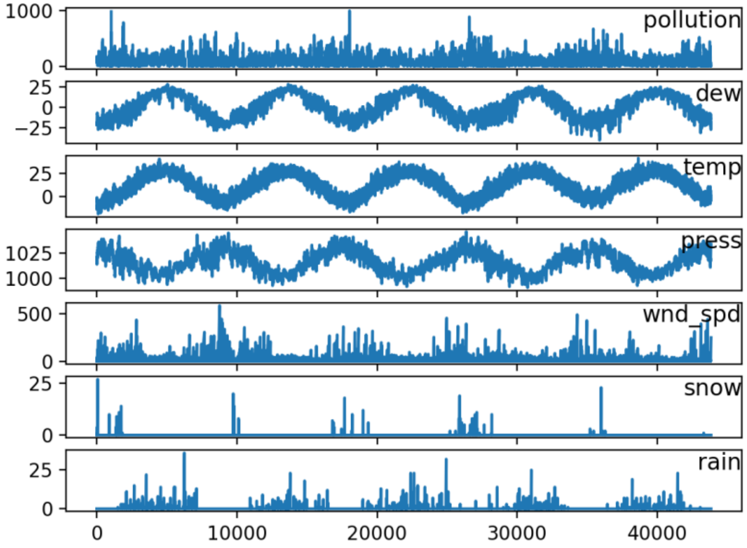
Univariate and multivariate time series

УНИВЕРСИТЕТ ИТМО

Univariate time series



Multivariate time series

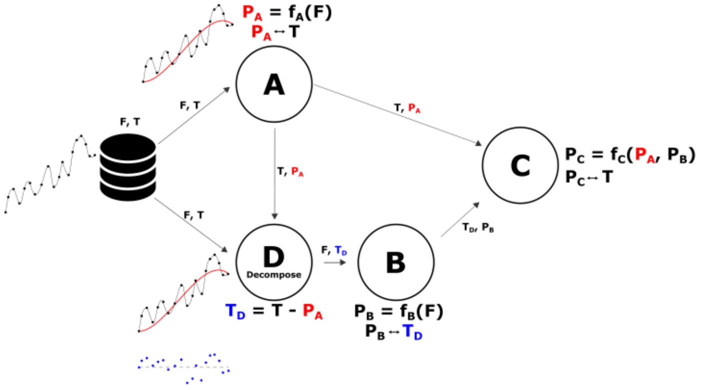


ITsMOre than a UNIVERSITY

Advanced forecasting and validation

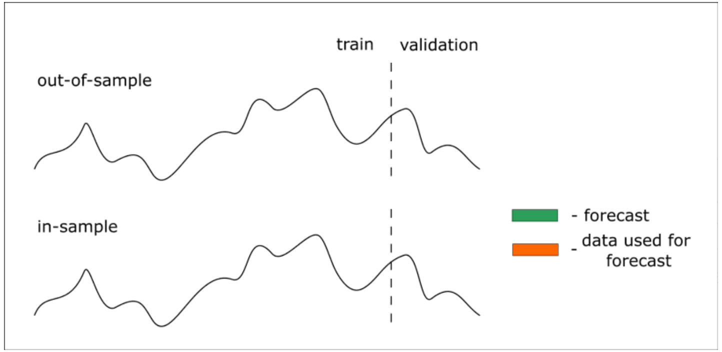
Multiscale forecasting

https://github.com/Dreamlone/ITMO_materials/blob/master/fedot-workshop/Multiscale_forecasting.ipynb



Validation for time series forecasts

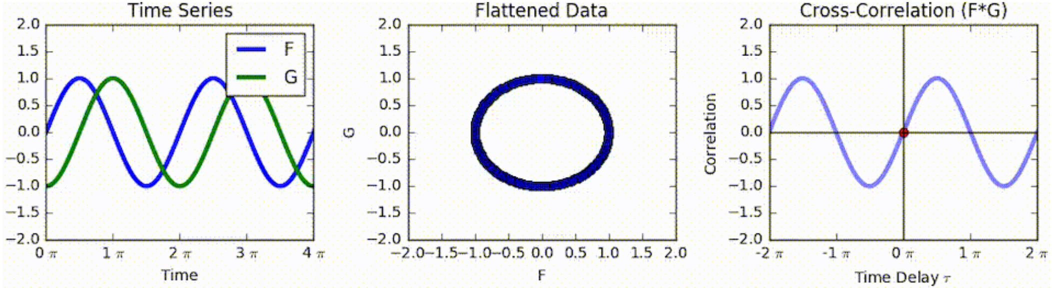
https://github.com/Dreamlone/ITMO_materials/blob/master/fedot-workshop/Advanced_validation.ipynb

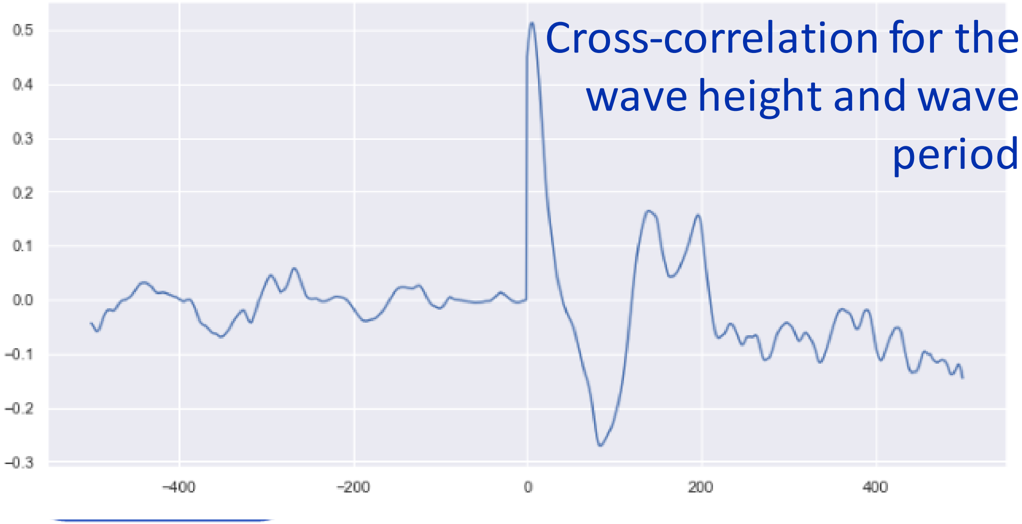


ITMOre than a UNIVERSITY

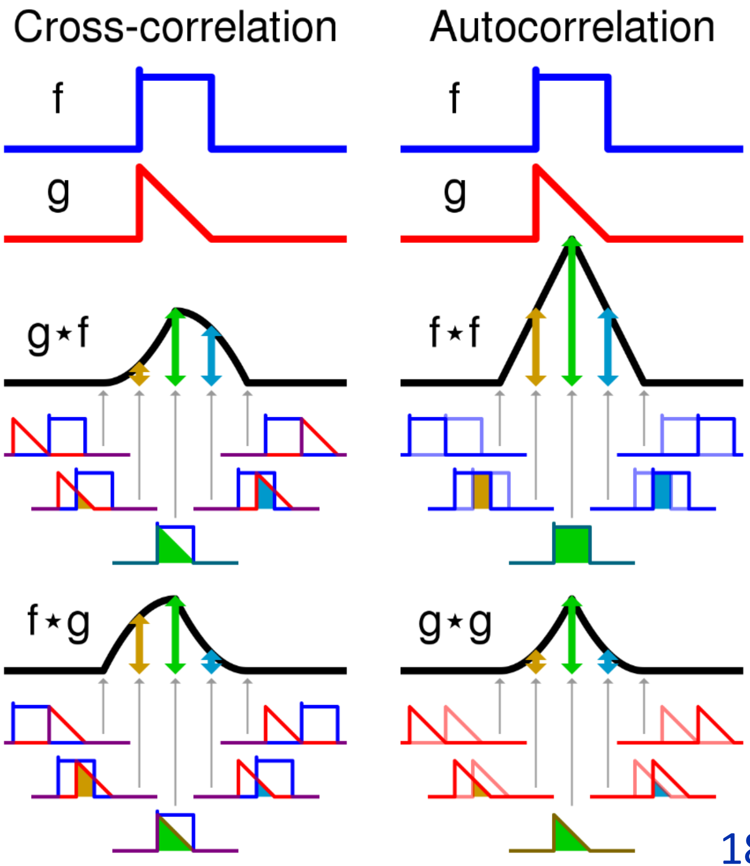
Cross-correlation (mutual correlation)

Cross-correlation for the synthetic data





Cross-correlation for the wave height and wave period

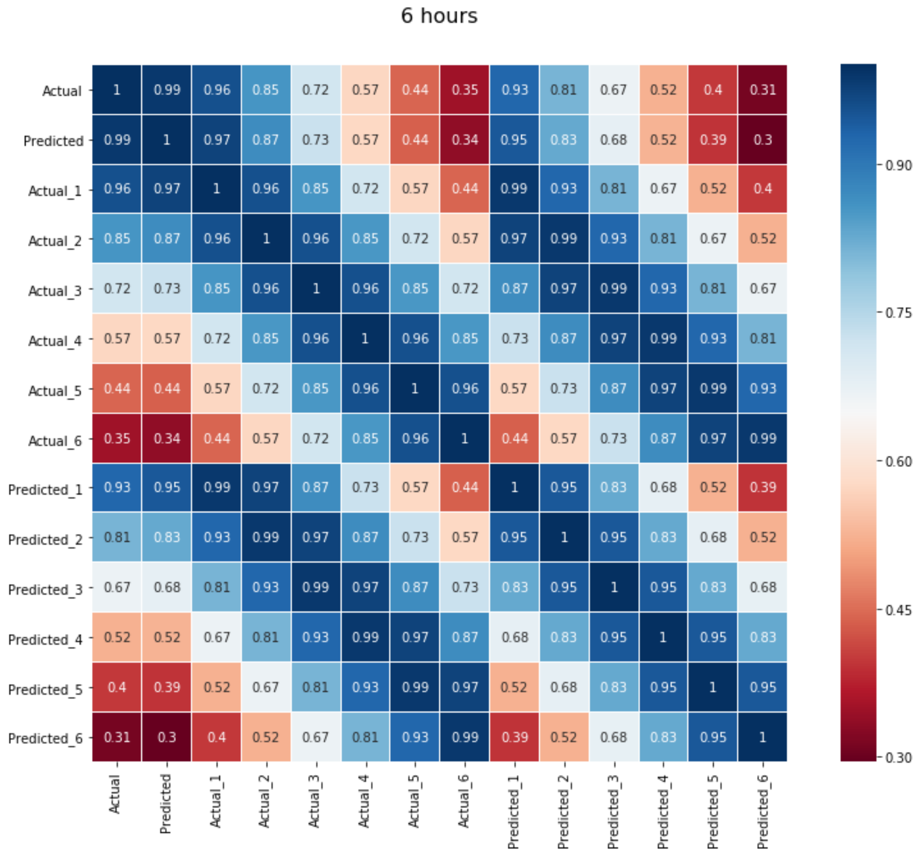


Cross-correlation matrix for random process

УНИВЕРСИТЕТ ИТМО

Cross-correlation matrix

$$\mathrm{R}_{\mathbf{XY}} = \begin{bmatrix} \mathrm{E}[X_1 Y_1] & \mathrm{E}[X_1 Y_2] & \cdots & \mathrm{E}[X_1 Y_n] \\ \mathrm{E}[X_2 Y_1] & \mathrm{E}[X_2 Y_2] & \cdots & \mathrm{E}[X_2 Y_n] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[X_m Y_1] & \mathrm{E}[X_m Y_2] & \cdots & \mathrm{E}[X_m Y_n] \end{bmatrix}$$

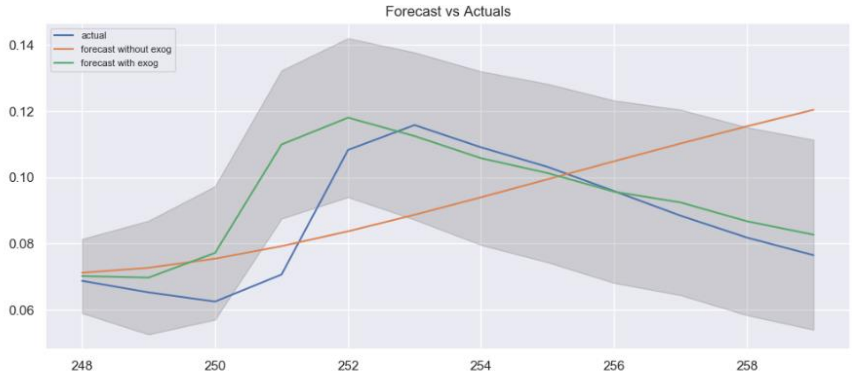


Cross-correlation matrix visualisation

ITsMOre than a UNIVERSITY

Model of linear dynamical system

```
from statsmodels.tsa.vector_ar.var_model import VAR

train, test = df[['Hsig', 'RTpeak']][:-test_size], df[['Hsig', 'RTpeak']][-test_size:]

history = train
predictions = list()

for t in range(test.shape[0]):
    model = VAR(endog=history)
    model_fit = model.fit(maxlags=16)
    output = model_fit.forecast(model_fit.y, steps=1)
    yhat = output[0]
    predictions.append(yhat)
    obs = test.iloc[t]
    history = history.append(obs)
```



Vector Auto Regression model implementation

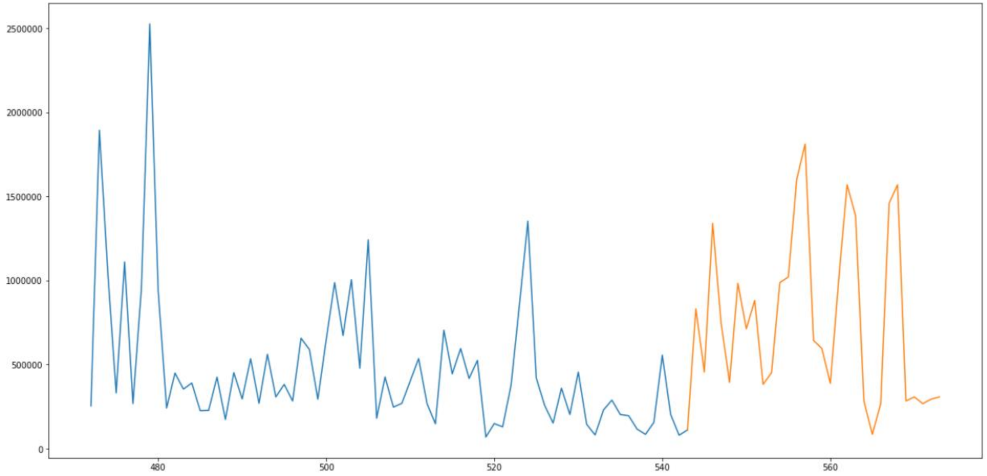
Wave height forecasting with additional variables

Open
https://github.com/Dreamlone/ITMO_materials/blob/master/fedot-workshop/Multivariate.ipynb

Automation of the predictive modelling


```
[ ] from fedot.api.main import Fedot

    # init model for the time series forecasting
    model = Fedot(problem='ts_forecasting')

    #run AutoML model design in the same way
    chain = model.fit(features=train_data_path, target='target')

    /usr/local/lib/python3.7/dist-packages/fedot/core/data/data.py:196: UserWarning: A
      warnings.warn(f'Automatic factorization for the column {column_name} with type '
    light preset is used. Parameters tuning: False. Set of candidate models: ['linear'
    Model composition started
    Model composition finished

[ ] ts_forecast = model.forecast(pre_history=train_data_path, forecast_length = 30)

[ ] #plot forecasting result
    model.plot_prediction()
```

Example for AutoML framework FEDOT (https://github.com/nccr-itmo/FEDOT)



Forecasted time series for financial dataset (**prehistory** is used to fit the model)

ITMOre than a UNIVERSITY

https://colab.research.google.com/drive/1cRFhC3GwkmfDmzgqof0q7M7i3ocJESSF?usp=sharing

Materials for workshop

Notebooks with examples:
https://github.com/Dreamlone/ITMO_materials/tree/master/fedot-workshop

FEDOT:
https://github.com/aimclub/FEDOT

Additional info:
https://towardsdatascience.com/automl-for-time-series-definitely-a-good-idea-c51d39b2b3f
https://towardsdatascience.com/automl-for-time-series-advanced-approaches-with-fedot-framework-4f9d8ea3382c
https://towardsdatascience.com/what-to-do-if-a-time-series-is-growing-but-not-in-length-421fc84c6893

Thank you!

ITsMOre than a UNIVERSITY