**Теория Фильтрации и Прогнозирование данных**

Липатов Данила МСМТ 243

Лабораторная работа №2

**Пункт 1**

Полином второго порядка имеет вид:

График полинома второго порядка для X.

Нахождение полинома производилось через встроенные функции ЯП Python: numpy

np.polyval(coef, time)

np.polyfit(time, signal, deg)

Коэффициенты полинома

-6.76844480e-04 2.50515932e+00 -2.30078157e+03

Где

-6.76844480e-04 - маленькое значение говорит о слабом квадратичном тренде, так как он отрицательный, ветви параболы направлены вниз

2.50515932 - отвечает за линейный тренд, показывая увеличение на каждом временном шаге, чем больше значение, тем больше наклон графика.

-2300.7 – Пересечение при х = 0

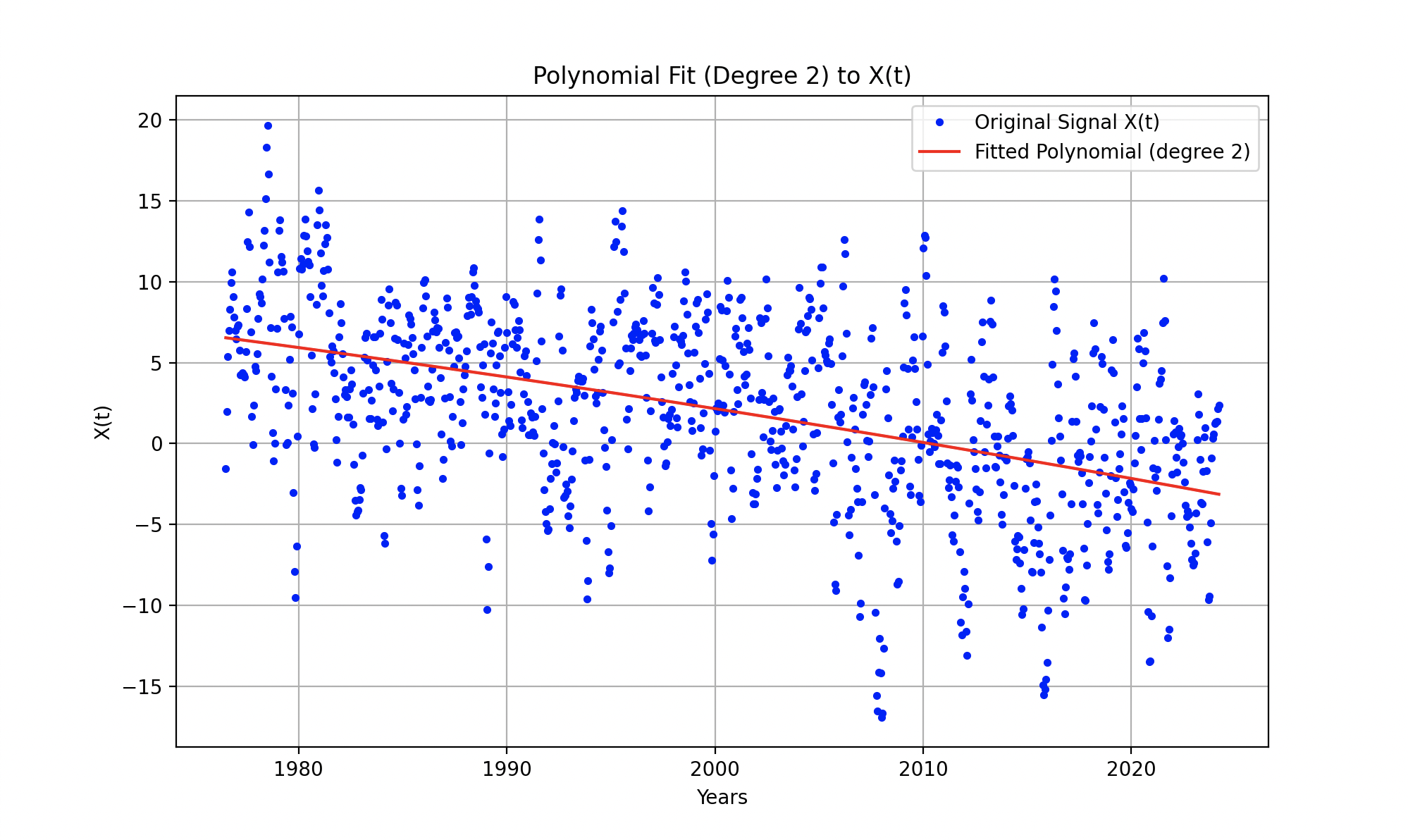


Рис. 1 График полинома 2 рода для Х.

**Пункт 2**

Для нахождения АКФ сигнала Х воспользуемся встроенной функцией numpy.correlate(), в качестве аргументов передается сигнал.

def autocovariance(signal\_):  
 N = len(signal\_)  
 acf = np.correlate(signal\_, signal\_, mode='full')/ N  
 lags = signal.correlation\_lags(len(signal\_), len(signal\_))  
 return acf, lags

Для каждого сигнала (а, b) найдем его АКФ

X\_mean = X - np.mean(X)  
X\_detrended = X - polyn

acf\_mean, lags\_mean = autocovariance(X\_mean)  
acf\_detrended, lags\_detrended = autocovariance(X\_detrended)

Получим следующий графики АКФ для каждого из варинатов:

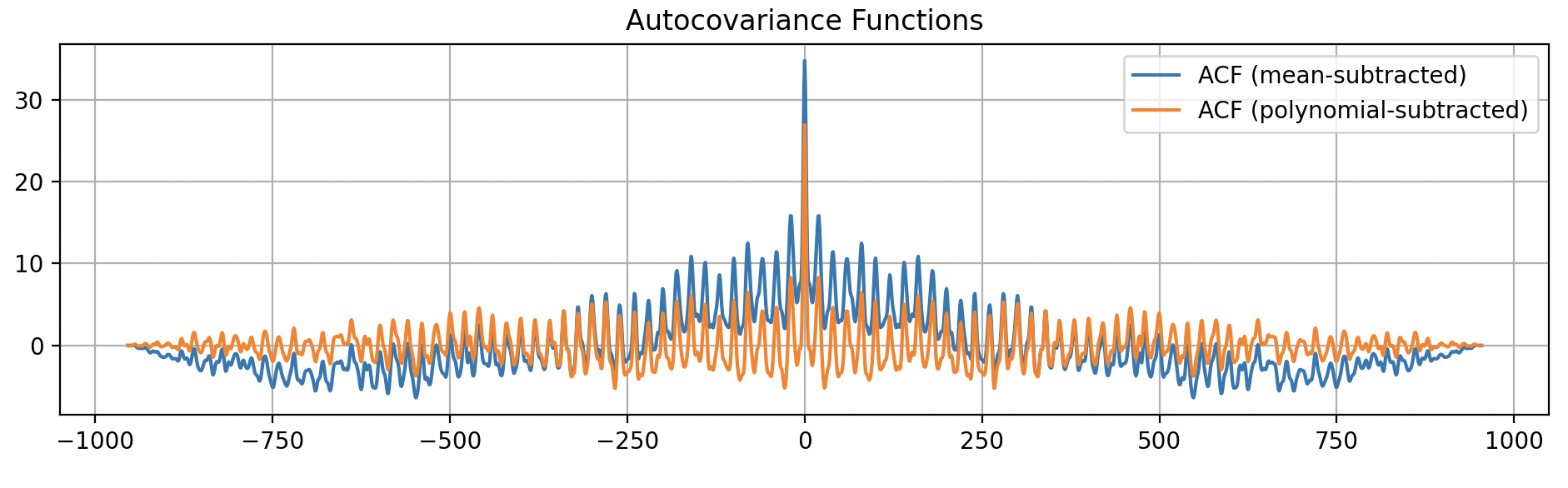


Рис. 2 АКФ для X\_mean / X\_detrended

Аналогично алгоритмам, реализованных в лаб.работе №1 БФП найдем спектры АКФ для обеих пунктов и отдельно для сигнала Х, получим следующее:

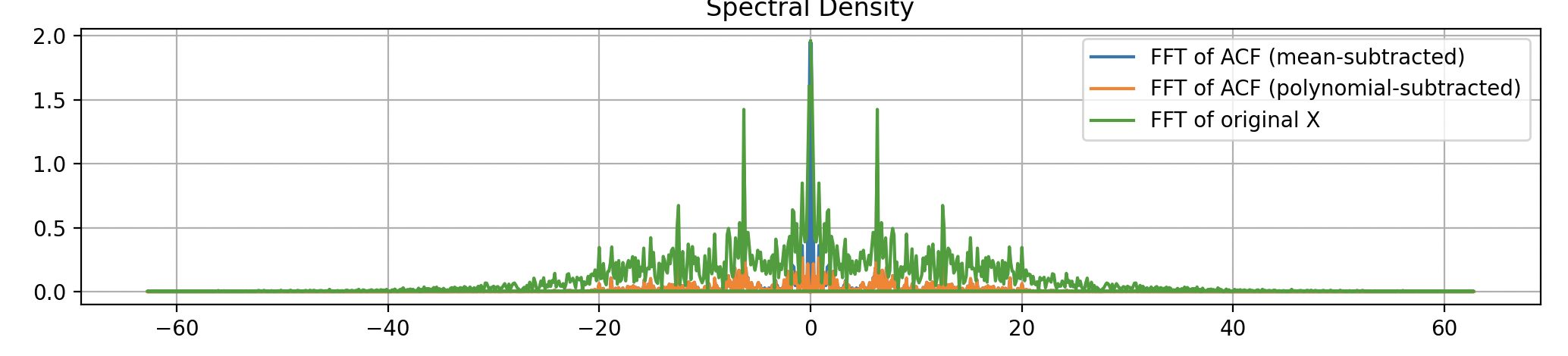


Рис. 3 Спектрограммы для АКФ и Х

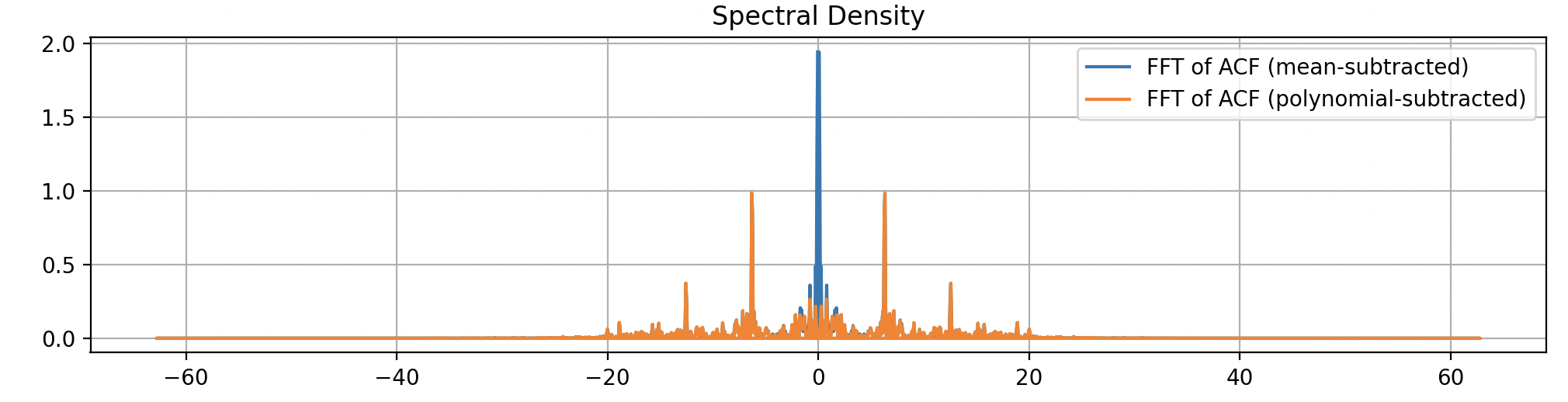


Рис. 3.2 Спектрограммы для АКФ и Х

При увеличении спектрограммы можно заметить, что АКФ с вычетом полинома в 0 имеет 0-ую амплитуду. А если мы посмотрим наложение одного спектра на другой, то в целом можно заметить схожее поведение со спектром оригинального сигнала.

**Пункт 3**

Для вычисления кросс-корреляционную функцию достаточно применить функцию из пункта 2, где вторым аргументом будет сигнал Y.

Получим следующий график:

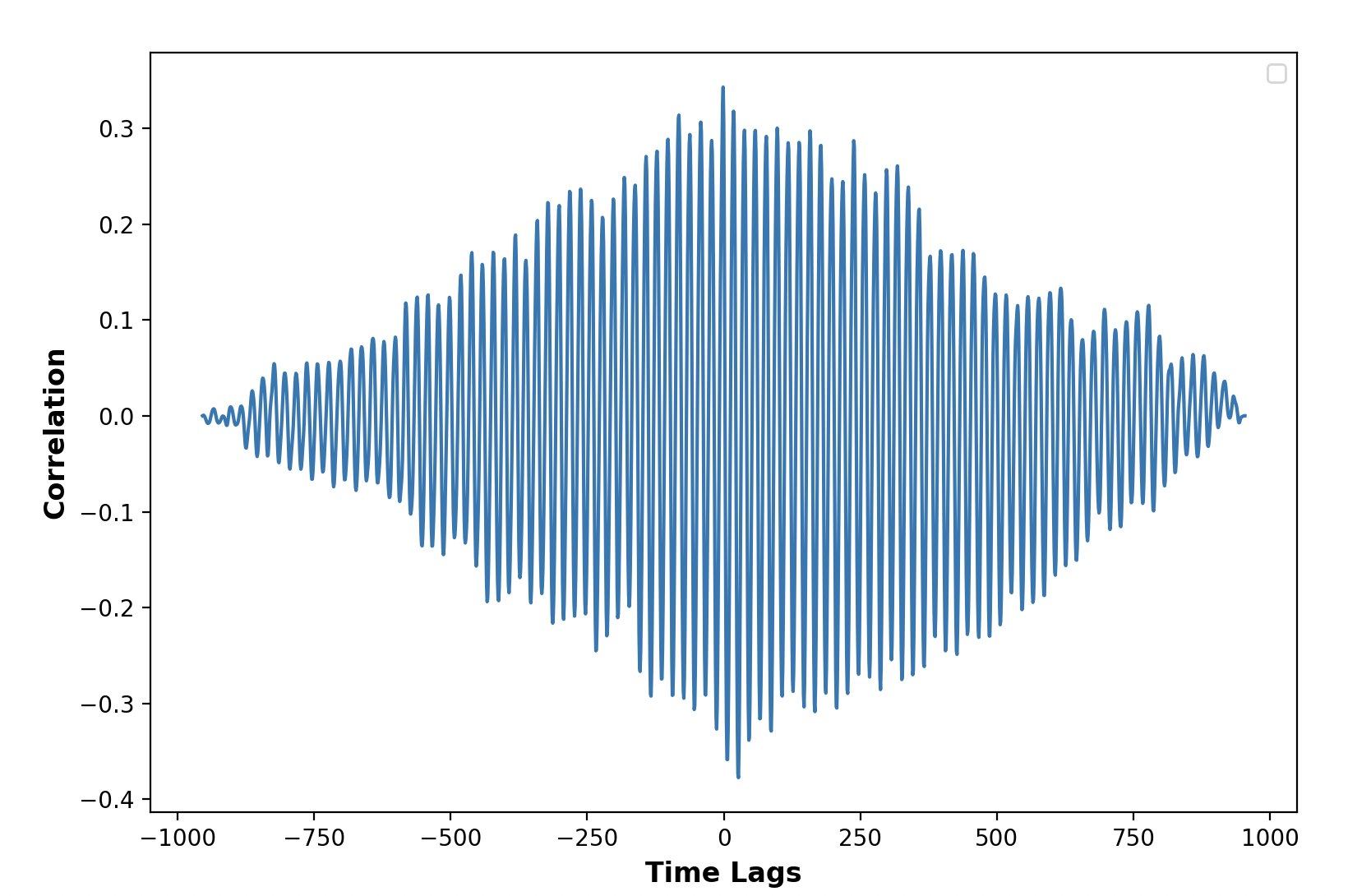


Рис. 4 Кросс-корреляционая функция между Х и Y

В целом из графика видно, что в окрестности 0 находится максимальное значение. Однако найдем точное значение лага, которому соответствует максимальное значение

Это означает, что второй временной рядопережаетпервый на один временной шаг.

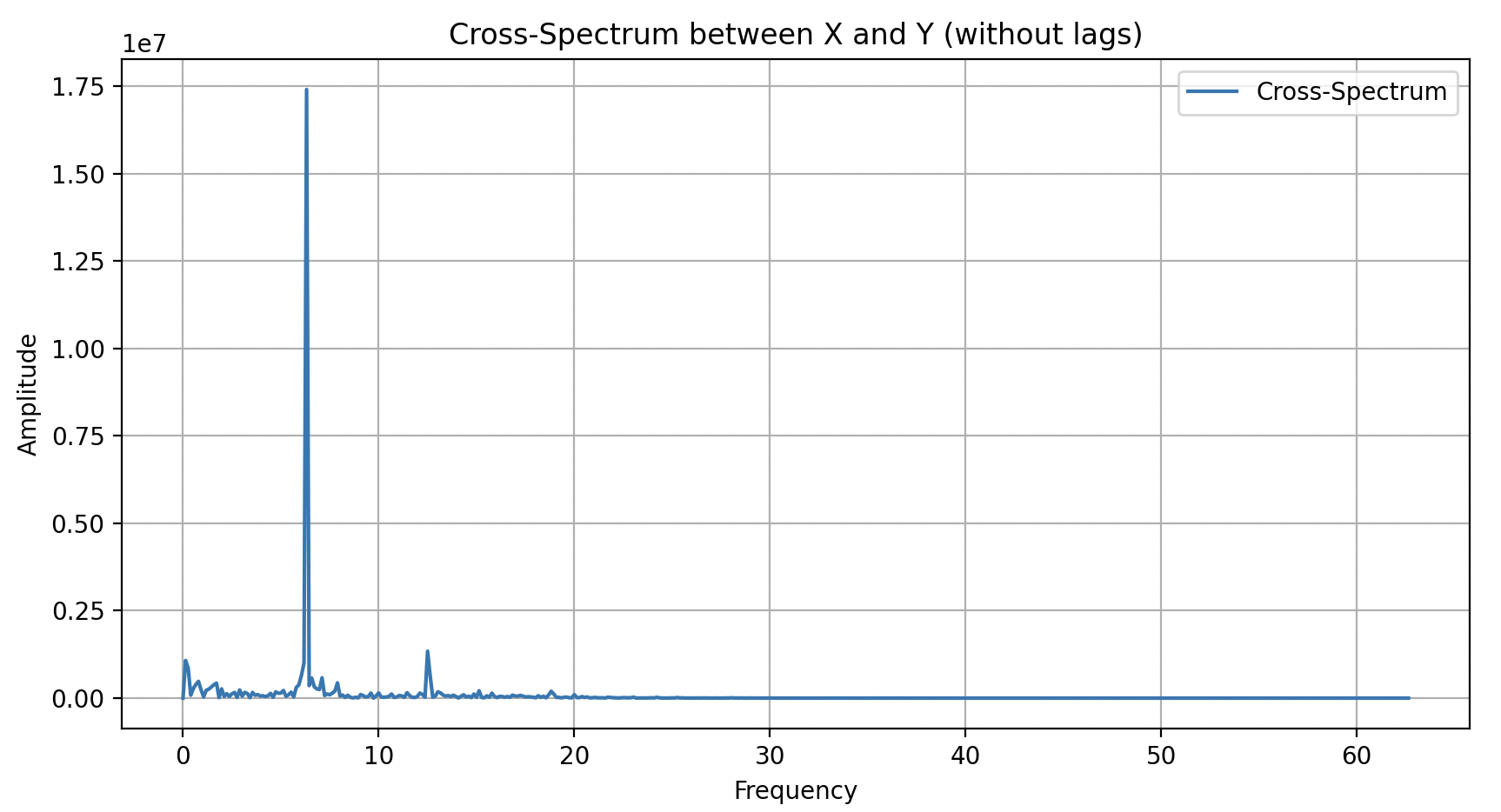
Кросс-спектр между Х и Y 

Рис. 5 Кросс-спектр между Х и Y

**Пункт 4**

Для генерации случайный сигнал АРСС (ARMA) используем следующие коэффициенты и функции в Python

ar\_1 = 0.5  
ar\_2 = 0.25  
ar\_3 = 0.1  
ma\_1 = -0.3  
ma\_2 = -0.2  
ma\_3 = 0.1  
ar\_root = np.array([-ar\_1, -ar\_2, -ar\_3, 1]) # Коэффициенты AR (авторегрессия)  
ma\_root = np.array([-ma\_1, -ma\_2, -ma\_3, 1]) # Коэффициенты MA (скользящее среднее)

arma\_process = ArmaProcess(ar, ma)  
n\_samples = 1000

ARMA = arma\_process.generate\_sample(nsample=n\_samples)

Тогда корни полинома будут следующие (отдельно для AR / MA)

[-0.7837762+1.12211372i -0.7837762-1.12211372i 1.0675524+0.i]

[-1.83649528+0.i 0.58491431+1.2136419i 0.58491431-1.2136419i]

Взяв модуль каждого корня увидим, что все они лежат вне

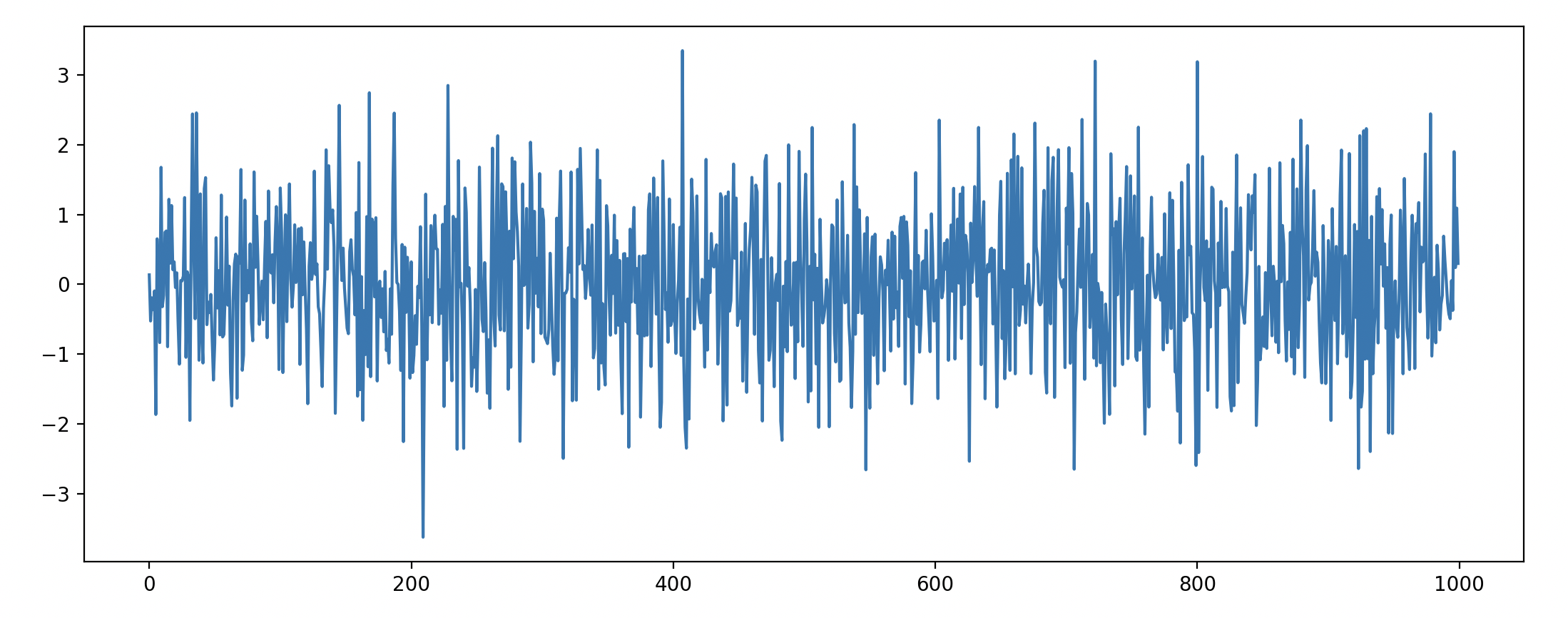


Рис. 6 Смоделированный сигнал ARMA

Так же необходимо было построить смещенную и несмещенную АКФ ARMA.

Для этого воспользуемся так же функцией np.correlate(), предварительно рассчитывая необходимую АКФ

if biased:  
 result /= n # Смещённая АКФ  
else:  
 result = result[n - 1:]  
 result /= (n - np.arange(n)) # Несмещённая АКФ

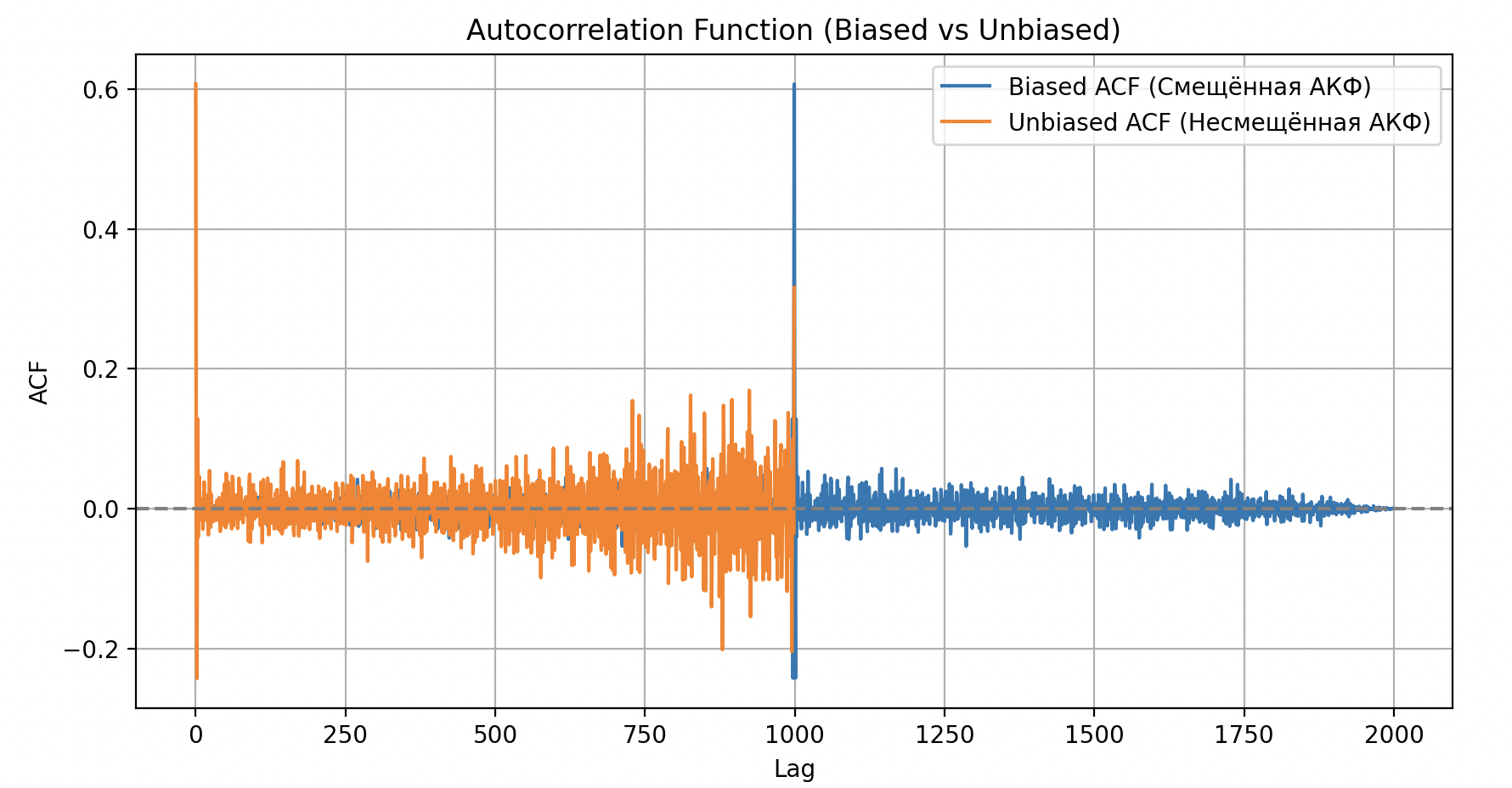


Рис. 7 Графики смещенной и несмещенной АКФ (ограниченные слева и размещенные, чтобы проще было понять поведение)

АКФ смещенная:

АКФ несмещенная:

Изменяя нормировочный делитель с

N на N−k, мы переходим с смещенной на несмещенную оценку

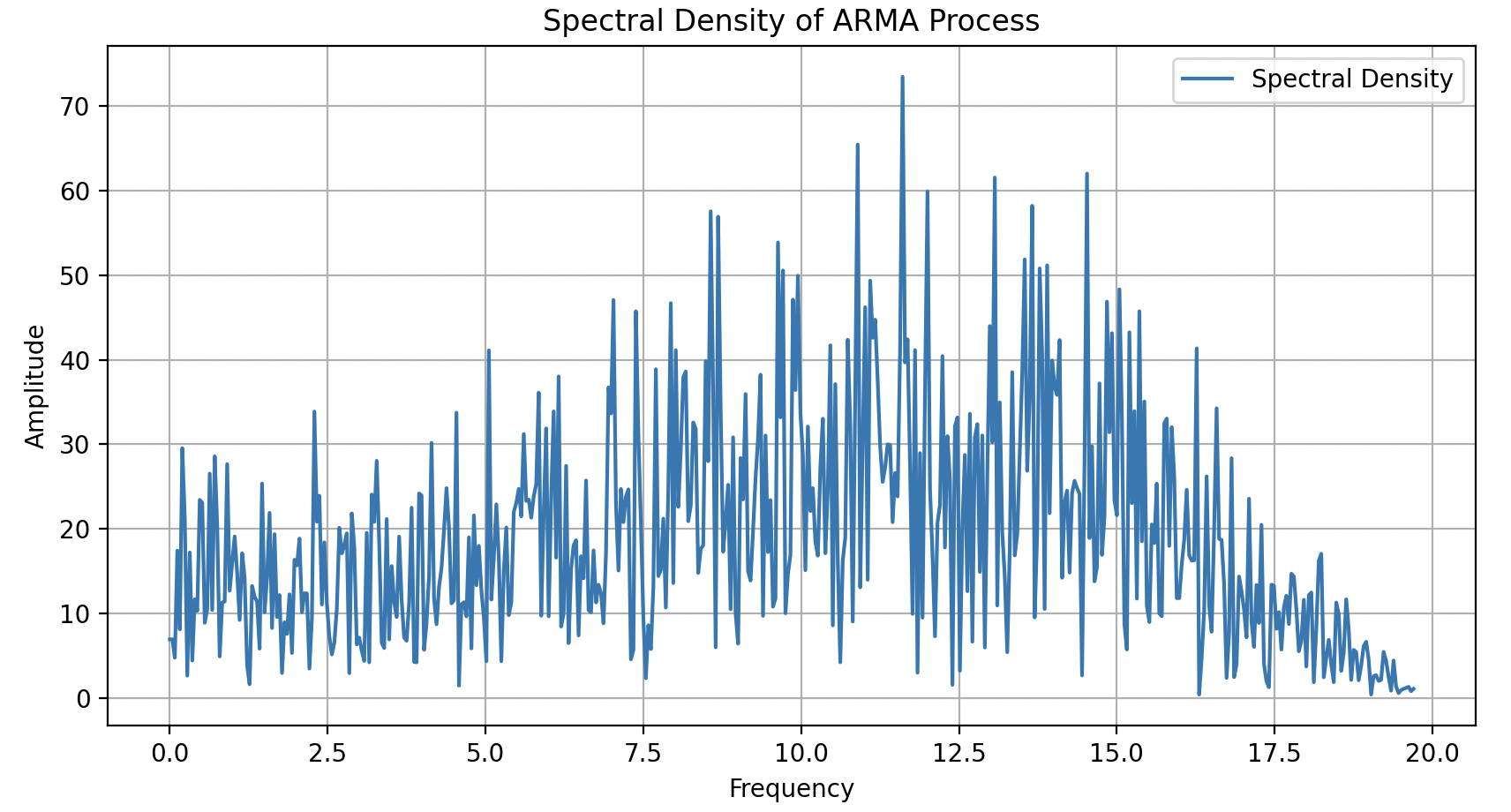


Рис. 8 Спектр произвольной ARMA модели

Примечание:

Несмещённая АКФ компенсирует уменьшение количества точек при больших лагах, что приводит к большим колебаниям её значений на концах ряда

Для оценки устойчивости корней в Python у функции ArmaProcess(ar, ma) есть метод проверки на устойчивость для ar и ma отдельно

