**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное

учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Направление «Информатика и вычислительная техника»

Отчет о выполнении НИРС за второй семестр обучения

В рамках темы «Исследование и визуализация данных датчиков МГР»

Выполнил:

Студент группы 8ВМ82 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А. Иванов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.К. Бокижонов

Проверил:

Доцент ОИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.А. Ботыгин

Томск 2019

Оглавление

[Введение 3](#_Toc10709829)

[Преобразование Фурье 4](#_Toc10709830)

[Вейвлет преобразование 7](#_Toc10709831)

[Листинг программы 11](#_Toc10709832)

[Результат работы 15](#_Toc10709833)

[Вывод 18](#_Toc10709834)

# Введение

Программа была разработана на языке python с GUI-интерфейсом. Среда разработки была использована PyCharm.

Данные взяты с сайта <http://mgr.imces.ru/> , программа может обрабатывать такие форматы как mgr,gr,gr2.

Во-первых, был разработан код, который позволяет динамически загружать данные в программу.

Во-вторых, был реализован выбор периода, который позволяет пользователю получить данные за определенный промежуток времени.

С полученными данными можно просмотреть описательную статистику по выбранному периоду, а также выбрать какие именно данные интересуют пользователя, далее выводим график для наглядности, производим преобразование Фурье и Вейвлет.

# Преобразование Фурье

Спектральный анализ - один из методов обработки сигналов, который позволяет охарактеризовать частотный состав измеряемого сигнала. Преобразование Фурье является математической основой, которая связывает временной или пространственный сигнал (или же некоторую модель этого сигнала) с его представлением в частотной области. Важную роль в спектральном анализе играют методы статистики, поскольку сигналы, как правило, имеют случайный характер или зашумлены при распространении или измерении. Если бы основные статистические характеристики сигнала были точно известны, или их можно было определить по конечному интервалу этого сигнала, то спектральный анализ представлял бы собой отрасль "точной науки". Однако, в действительности по отрезку сигнала можно получить только оценку его спектра. Поэтому практика спектрального анализа - некое ремесло (или искусство?) достаточно субъективного характера. Различие между спектральными оценками, получаемыми в результате обработки одного и того же отрезка сигнала разными методами, можно объяснить различием допущений, принятых относительно данных, различными способами усреднения и т.п. Если априори характеристики сигнала не известны, нельзя сказать какие из оценок лучше.

Преобразование Фурье - математическая основа спектрального анализа

Кратко обсудим разные виды преобразования Фурье

Начнем с преобразования Фурье непрерывного во времени сигнала

Начнем с преобразования Фурье непрерывного во времени сигнала

http://www.vibration.ru/images/preobraz_fur1.gif, (1)

которое идентифицирует частоты и амплитуды тех комплексных синусоид (экспонент), на которые разлагается некоторое произвольное колебание.  
   Обратное преобразование

http://www.vibration.ru/images/preobraz_fur2.gif. (2)

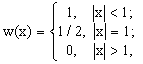
   Существование прямого и обратного преобразования Фурье (которое в дальнейшем мы будем называть непрерывно-временным преобразованием Фурье - НВПФ) определяется рядом условий. Достаточное - абсолютная интегрируемость сигнала

http://www.vibration.ru/images/preobraz_fur3.gif.            (3)

   Менее ограничительное достаточное условие - конечность энергии сигнала

http://www.vibration.ru/images/preobraz_fur4.gif.            (4)

   Приведем ряд основных свойств преобразования Фурье и функций, используемых далее, заметив, что прямоугольное окно определяется выражением

         (5)

а функция sinc - выражением

http://www.vibration.ru/images/preobraz_fur6.gif            (6)

   Функция отсчетов во временной области определяется выражением

http://www.vibration.ru/images/preobraz_fur7.gif      (7)

Эту функцию иногда называют также функцией периодического продолжения.

*Таблица 1. Основные свойства НВПФ и функции*

|  |  |  |
| --- | --- | --- |
| Свойство, функция | Функция | Преобразование |
| Линейность | ag( t ) + bh( t ) | aG( f ) + bH( f ) |
| Сдвиг по времени | h ( t - t0 ) | H( f )exp( -j2pf t0 ) |
| Сдвиг по частоте (модуляция) | h ( t )exp( j2pf0 t ) | H( f - f0 ) |
| Масштабирование | ( 1 / |a| )h( t / a ) | H( af ) |
| Теорема свертки во временной области | g( t )\*h( t ) | G( f )H( f ) |
| Теорема свертки в частотной области | g( t ) h( t ) | G( f )\*H( f ) |
| Функция окна | Aw( t / T ) | 2ATsinc( 2Tf ) |
| Функция sinc | 2AFsinc( 2Ft ) | Aw( f / F ) |
| Импульсная функция | Ad( t ) | A |
| Функция отсчетов | http://www.vibration.ru/images/psi.gifT(f) | Fhttp://www.vibration.ru/images/psi.gifF(f), F=1/T |

   Еще одно важное свойство устанавливается теоремой Парсеваля для двух функций g(t) и h(t):

http://www.vibration.ru/images/preobraz_fur8.gif.         (8)

   Если положить g(t) = h(t), то теорема Парсеваля сводится к теореме для энергии

http://www.vibration.ru/images/preobraz_fur9.gif.         (9)

   Выражение (9) - это, в сущности, просто формулировка закона сохранения энергии в двух областях (временной и частотной). В (9) слева стоит полная энергия сигнала, таким образом, функция

http://www.vibration.ru/images/preobraz_fur10.gif  (10)

описывает распределение энергии по частоте для детерминированного сигнала h(t) и поэтому называется спектральной плотностью энергии (СПЭ). С помощью выражений

http://www.vibration.ru/images/preobraz_fur11.gif(11)

можно вычислить амплитудный и фазовый спектры сигнала h( t ).

Главной математической основой спектрального анализа является преобразование Фурье, которое связывает пространственный или временной сигнал (либо некоторую модель этого сигнала) с его представлением в частотной области. Преобразование Фурье функции f является интегральным представлением и задается следующей формулой: image Но преобразование Фурье дает информацию только про частоту, которая присутствует в сигнале и не дает никакой информации про то, в какой промежуток времени эта частота присутствует в сигнале.

Таким образом, для двух абсолютно разных сигналов мы получаем почти одинаковые преобразования Фурье (не плавность графика преобразования Фурье для второго сигнала объясняется внезапной сменой частоты в этом сигнале, а разница в амплитуде разных частот объясняется тем, что эти частоты действовали на протяжении разного времени на рассматриваемом отрезке сигнала).

Таким образом преобразование Фурье по своей сути не может отличить стационарный сигнал от нестационарного, что является большой проблемой для его применимости.

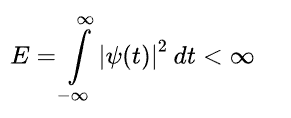
# Вейвлет преобразование

Вейвлет-преобразование— интегральное преобразование, которое представляет собой свертку вейвлет-функции с сигналом. Вейвлет-преобразование переводит сигнал из временного представления в частотно-временное.

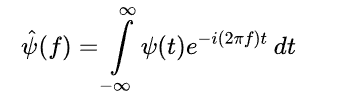
Способ преобразования функции (или сигнала) в форму, которая или делает некоторые величины исходного сигнала более поддающимися изучению, или позволяет сжать исходный набор данных. Вейвлетное преобразование сигналов является обобщением спектрального анализа. Термин в переводе с английского означает «маленькая волна». Вейвлеты — это обобщённое название математических функций определенной формы, которые локальны во времени и по частоте, и в которых все функции получаются из одной базовой, изменяя её (сдвигая, растягивая).

Для осуществления вейвлет-преобразования вейвлет-функции должны удовлетворять следующим критериям[[1]](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B9%D0%B2%D0%BB%D0%B5%D1%82-%D0%BF%D1%80%D0%B5%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5#cite_note-1):

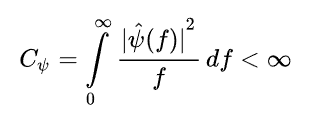
1. Вейвлет должен обладать конечной энергией:

{\displaystyle E=\int \limits \_{-\infty }^{\infty }{|\psi (t)|}^{2}\,dt<\infty } 

2. Если {\displaystyle {\hat {\psi }}(f)} [фурье-преобразование](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D1%80%D1%8C%D0%B5-%D0%BF%D1%80%D0%B5%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5" \o "Фурье-преобразование) для {\displaystyle \psi (t)}, то есть

{\displaystyle {\hat {\psi }}(f)=\int \limits \_{-\infty }^{\infty }\psi (t)e^{-i(2\pi f)t}\,dt}

тогда должно выполняться следующее условие:

{\displaystyle C\_{\psi }=\int \limits \_{0}^{\infty }{\frac {{|{\hat {\psi }}(f)|}^{2}}{f}}\,df<\infty }

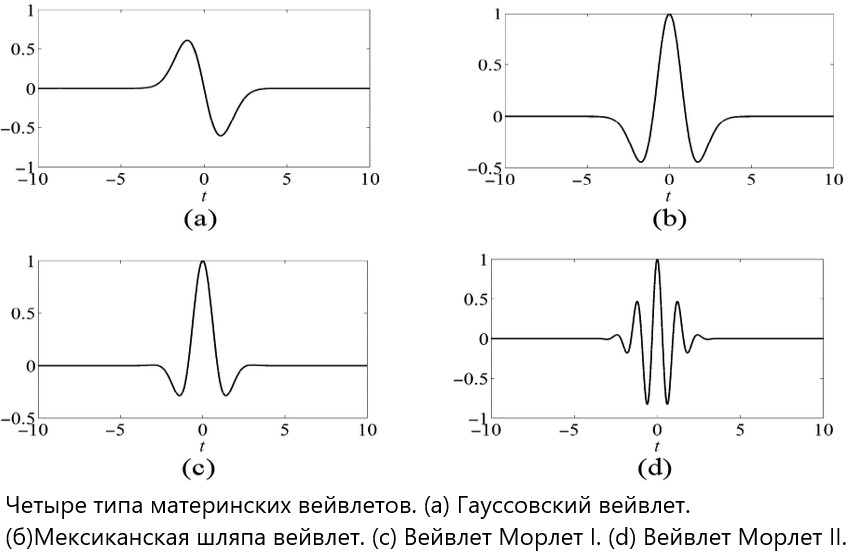
Это условие называется условием допустимости, и из него следует что вейвлет при нулевой частотной компоненте должен удовлетворять условию {\displaystyle {\hat {\psi }}(0)=0} или, в другом случае, вейвлет {\displaystyle \psi (t)} должен иметь среднее равное нулю.

3. Дополнительный критерий предъявляется для комплексных вейвлетов, а именно, что для них Фурье-преобразование должно быть одновременно вещественным и должно убывать для отрицательных частот.

4. Локализация: вейвлет должен быть непрерывным, интегрируемым, иметь компактный носитель и быть локализованным как во времени (в пространстве), так и по частоте. Если вейвлет в пространстве сужается, то его средняя частота повышается, спектр вейвлета перемещается в область более высоких частот и расширяется. Этот процесс должен быть линейным — сужение вейвлета вдвое должно повышать его среднюю частоту и ширину спектра также вдвое.

Вейвлет-преобразование широко используется для анализа сигналов. Помимо этого, оно находит большое применение в области сжатия данных. В дискретном вейвлет-преобразовании наиболее значимая информация в сигнале содержится при высоких амплитудах, а менее полезная — при низких. Сжатие данных может быть получено за счет отбрасывания низких амплитуд. Вейвлет-преобразование позволяет получить высокое соотношение сжатия в сочетании с хорошим качеством восстановленного сигнала. Вейвлет-преобразование было выбрано для стандартов сжатия изображений JPEG2000 и ICER. Однако, при малых сжатиях вейвлет-преобразование уступает по качеству в сравнении с оконным Фурье-преобразованием, которое лежит в основе стандарта JPEG.

Выбор конкретного вида и типа вейвлетов во многом зависит от анализируемых сигналов и задач анализа. Для получения оптимальных алгоритмов преобразования разработаны определенные критерии, но их еще нельзя считать окончательными, так как они являются внутренними по отношению к самим алгоритмам преобразования и, как правило, не учитывают внешних критериев, связанных с сигналами и целями их преобразований. Отсюда следует, что при практическом использовании вейвлетов необходимо уделять достаточное внимание проверке их работоспособности и эффективности для поставленных целей по сравнению с известными методами обработки и анализа.



*Рисунок 1. Виды Вейвлет преобразования*

Вейвлет преобразование было создано как инструмент, который решает проблему неопределенности Гейзенберга для построение частотно-временных характеристик сигнала.

Вейвлет преобразование сигнала f(t) имеет вид:

image

где Tau — сдвиг по времени, S — масштаб, а Psi\* — материнский вейвлет.

Понятие вейвлета означает волну, которая проходит через сигнал и является окном некоторой ширины (масштаба) для некоторого местоположения во времени, во время (простите за тавтологию) интегрирования сигнала.

Материнский вейвлет — это функция, которая является прототипом для всех окон, которые будут генерироваться во время вейвлет-преобразования.

Сдвиг по времени регулирует движение генерированных окон по временной компоненте сигнала.

Понятие масштаба является обратным к понятию ширины окна. Чем меньше ширина окна, тем больше масштаб, то есть окно захватывает меньшую часть сигнала и сигнал интегрируется более «детально».

Чем больше ширина окна, тем меньший масштаб, то есть окно захватывает большую часть сигнала и сигнал, соответственно, интегрируется менее «детально».

В результате описанного процесса мы получаем рассчитанные значения интегралов функции для каждого масштаба в каждый момент времени. Таким образом мы получаем трехмерное представление сигнала с компонентами: масштаб, время и амплитуда (значения рассчитанных интегралов).

Таким образом, вейвлет-преобразования, в отличии от оконного преобразования Фурье, которое имеет постоянный масштаб в любой момент времени для всех частот, имеет лучшее представление времени и худшее представление частоты на низких частотах сигнала и лучшее представление частоты с худшим представлением времени на высоких частотах сигнала.

На рисунках хорошо видно, что полученное вейвлет-преобразование является более детализированным по времени в области высоких значений масштаба (низких частот) и менее детализирована в области низких значений масштаба (высоких частот).

Из этого следует, что вейвлет преобразования дает возможность уменьшить влияние принципа неопределенности Гейзенберга на полученном частотно-временном представлении сигнала. С его помощью низкие частоты имеют более детальное представление относительно времени, а высокие — относительно частоты.

# Листинг программы

|  |
| --- |
| from PyQt5 import QtCore, QtGui, QtWidgets from untitled import Ui\_MainWindow from datetime import datetime from scipy import signal import sys import urllib.request import pandas as pd import numpy as np import scipy as sp import matplotlib.pyplot as plt import scipy.fftpack   #Инициализация приложения app = QtWidgets.QApplication(sys.argv) #Форма MainWindow = QtWidgets.QMainWindow() ui = Ui\_MainWindow() ui.setupUi(MainWindow) MainWindow.show() #Логика приложения  def Download():  vstat = ui.lineediturl.text()  page = urllib.request.urlopen(vstat)  content = page.read()  type\_file = vstat.split('.')  if type\_file[-1] == 'gr1':  width = 8 # shirina fayla gr1  numpy\_data\_polojitelnie = np.frombuffer(content, dtype=np.dtype('B')).reshape(int(len(content) / width), width)  length\_column = int(len(  content) / width) - 2 # вычитаем -2 так как первые 16 байтов являтся заголовком, каждая строка по 8 байтам и не читаем первые две строки  numpy\_date = np.array(numpy\_data\_polojitelnie[2:, :4], dtype=np.uint32)   if type\_file[-1] == 'gr2':  width = 12 # shirina fayla gr2  numpy\_data\_polojitelnie = np.frombuffer(content, dtype=np.dtype('B')).reshape(int(len(content) / width), width)  length\_column = int(len(content) / width) - 1 # вычитаем -1 так как первые 12 байтов являтся заголовком  numpy\_date = np.array(numpy\_data\_polojitelnie[1:, :4], dtype=np.uint32)   if type\_file[-1] == 'mgr':  width = 13 # shirina fayla mgr  numpy\_data\_polojitelnie = np.frombuffer(content, dtype=np.dtype('B')).reshape(int(len(content) / width), width)  length\_column = int(len(content) / width) - 1 # вычитаем -1 так как первые 13 байтов являтся заголовком  numpy\_date\_temp = np.array(numpy\_data\_polojitelnie[1:, :3], dtype=np.uint32)  numpy\_date = []  b = np.array(numpy\_date\_temp, dtype='U2')  for i in range(len(numpy\_date\_temp)):  numpy\_date.append(b[i, 0] + ':' + b[i, 1] + ':' + b[i, 2])  array\_date\_obichniy\_format = np.array(numpy\_date, dtype='U7')  # -----------------------------------------------------------------------------------------------------------------  if width == 8: # это означает что ширина = 8 то есть фалй gr1  numpy\_H1\_amp\_count = numpy\_data\_polojitelnie[2:, 4:6]  numpy\_H1\_amplituda = numpy\_data\_polojitelnie[2:, 6:8]   if width == 12: # это означает что ширина = 12 то есть фалй gr2  numpy\_H1\_amp\_count = numpy\_data\_polojitelnie[1:, 4:6]  numpy\_H1\_amplituda = numpy\_data\_polojitelnie[1:, 6:8]  numpy\_H2\_amp\_count = numpy\_data\_polojitelnie[1:, 8:10]  numpy\_H2\_amplituda = numpy\_data\_polojitelnie[1:, 10:12]   if width == 13: # это означает что ширина = 13 то есть фалй mgr  numpy\_H1\_amp\_count = numpy\_data\_polojitelnie[1:, 3:5]  numpy\_H2\_amp\_count = numpy\_data\_polojitelnie[1:, 5:7]  numpy\_E\_amp\_count = numpy\_data\_polojitelnie[1:, 7:9]  numpy\_H1\_amplituda = numpy\_data\_polojitelnie[1:, 9:11]  numpy\_H2\_amplituda = numpy\_data\_polojitelnie[1:, 11:13]   # -----------------------------------------------------------------------------------------------------------------   def date\_v\_sekundax(massiv):  s = np.zeros(length\_column, dtype='uint64')  for i in range(length\_column):  s[i] = massiv[i, 3] \* (pow(16, 6)) + massiv[i, 1] \* (pow(16, 4)) + massiv[i, 2] \* (pow(16, 2)) + massiv[  i, 0]  return s   if width == 8 or width == 12:  array\_date\_v\_sekundax = date\_v\_sekundax(numpy\_date)   def data\_obichniy\_format(massiv):  s = np.zeros(length\_column, dtype='<U40') # "<U36"dlina simvolov  for i in range(len(massiv)):  s[i] = datetime.fromtimestamp(massiv[i] - 2209032000).strftime("%Y.%m.%d, %I:%M:%S")  return s   if width == 8 or width == 12:  array\_date\_obichniy\_format = data\_obichniy\_format(array\_date\_v\_sekundax)   def func\_join\_2\_column(massiv):  s = np.zeros(length\_column, dtype='uint32')  for i in range(length\_column):  s[i] = massiv[i, 0] + massiv[i, 1] \* 16  return s  # -----------------------------------------------------------------------------------------------------------------   if width == 8:  array\_H1\_amp\_count = func\_join\_2\_column(numpy\_H1\_amp\_count)  array\_H1\_amplituda = func\_join\_2\_column(numpy\_H1\_amplituda)   dataset = pd.DataFrame(  {'Time': array\_date\_obichniy\_format, 'ImpulseNS': array\_H1\_amp\_count, 'AmplitudaNS': array\_H1\_amplituda})   if width == 12:  array\_H1\_amp\_count = func\_join\_2\_column(numpy\_H1\_amp\_count)  array\_H1\_amplituda = func\_join\_2\_column(numpy\_H1\_amplituda)  array\_H2\_amp\_count = func\_join\_2\_column(numpy\_H2\_amp\_count)  array\_H2\_amplituda = func\_join\_2\_column(numpy\_H2\_amplituda)   dataset = pd.DataFrame(  {'Time': array\_date\_obichniy\_format, 'ImpulseNS': array\_H1\_amp\_count, 'AmplitudaNS': array\_H1\_amplituda,  'ImpulseWE': array\_H2\_amp\_count,  'AmplitudaWE': array\_H2\_amplituda})   if width == 13:  array\_H1\_amp\_count = func\_join\_2\_column(numpy\_H1\_amp\_count)  array\_H2\_amp\_count = func\_join\_2\_column(numpy\_H2\_amp\_count)  array\_E\_amp\_count = func\_join\_2\_column(numpy\_E\_amp\_count)  array\_H1\_amplituda = func\_join\_2\_column(numpy\_H1\_amplituda)  array\_H2\_amplituda = func\_join\_2\_column(numpy\_H2\_amplituda)   dataset = pd.DataFrame(  {'Time': array\_date\_obichniy\_format, 'ImpulseNS': array\_H1\_amp\_count, 'AmplitudaNS': array\_H1\_amplituda,  'ImpulseWE': array\_H2\_amp\_count,  'AmplitudaWE': array\_H2\_amplituda})   x = array\_date\_obichniy\_format[0],array\_date\_obichniy\_format[-1]  ui.pokazatdatu.setText(str(x))   return dataset  ui.downloadpush.clicked.connect( Download )  def period():  o = ui.dataot.text()  w = ui.datado.text()  mask = (Download()['Time'] > o) & (Download()['Time'] <= w)  periodopr = Download().loc[mask]  return periodopr  ui.periodpush.clicked.connect( period )  def peredacha():  x = ui.comboBox.currentText()  if x == "Импульс (Север-Юг)":  p = period()['ImpulseNS']  if x == "Амплитуда (Север-Юг)":  p = period()['AmplitudaNS']  if x == "Импульс (Запад-Восток)":  p = period()['ImpulseWE']  if x == "Амплитуда (Запад-Восток)":  p = period()['AmplitudaWE']  return p  def colichestvo():  r1 = peredacha().count()  ui.lineEditstat.setText(str(r1)) ui.countpush.clicked.connect( colichestvo )  def sredneeznachenie():  r2 = peredacha().mean()  ui.lineEditstat.setText(str(r2))  ui.meanpush.clicked.connect( sredneeznachenie )  def mediana():  r3 = peredacha().median()  ui.lineEditstat.setText(str(r3))  ui.medianpush.clicked.connect( mediana )  def maingraph():  sst = peredacha()  n = len(sst)  time = np.arange(n)  dt = 0.25  fig = plt.figure(figsize=(20, 10))  time = np.arange(n)  xlim = [0, n]  plt.plot(time, sst)  xrange = plt.xlim(xlim)  plt.title('Исходный график', fontsize=30)  plt.show()  ui.maingraphpush.clicked.connect( maingraph )  def furietransform():  temp\_fft = sp.fftpack.fft(peredacha())  temp\_psd = np.abs(temp\_fft) \*\* 2  fftfreq = sp.fftpack.fftfreq(len(temp\_psd), 1)  i = fftfreq > 0  fig, ax = plt.subplots(1, 1, figsize=(18, 10))  ax.plot(fftfreq[i], 10 \* np.log10(temp\_psd[i]))  ax.set\_xlim(0, 0.5)  ax.set\_xlabel('Частота', fontsize=15)  ax.set\_ylabel('Амплитуда', fontsize=15)  plt.title('Преобразование Фурье', fontsize=25)  ui.furiepush.clicked.connect( furietransform )  def mhat():  widths = np.arange(1, 61)  cwtmatr = signal.cwt(peredacha(), signal.ricker, widths)  fig = plt.figure(figsize=(20, 10))  plt.title('Вейвлет преобразование Мексиканская шляпа', fontsize=30)  plt.imshow(cwtmatr, extent=[-1, 180, -1, 1], cmap='PRGn', aspect='auto', vmax=abs(cwtmatr).max(),  vmin=-abs(cwtmatr).max())  plt.xlabel('Дни', fontsize=20)  plt.ylabel('Амплитуда', fontsize=20)  plt.show()  ui.mhatpush.clicked.connect( mhat )  def morlet():  widths = np.arange(1, 41)  cwtmatr = signal.cwt(peredacha(), signal.gaussian, widths)  fig = plt.figure(figsize=(20, 10))  plt.title('Вейвлет преобразование Морлет', fontsize=30)  plt.imshow(cwtmatr, extent=[-1, 213, -1, 1], cmap='PRGn', aspect='auto', vmax=abs(cwtmatr).max(),  vmin=-abs(cwtmatr).max())  plt.xlabel('Дни', fontsize=20)  plt.ylabel('Амплитуда', fontsize=20)  plt.show()  ui.morletpush.clicked.connect( morlet )   #Выход из приложения sys.exit(app.exec\_()) |

# Результат работы

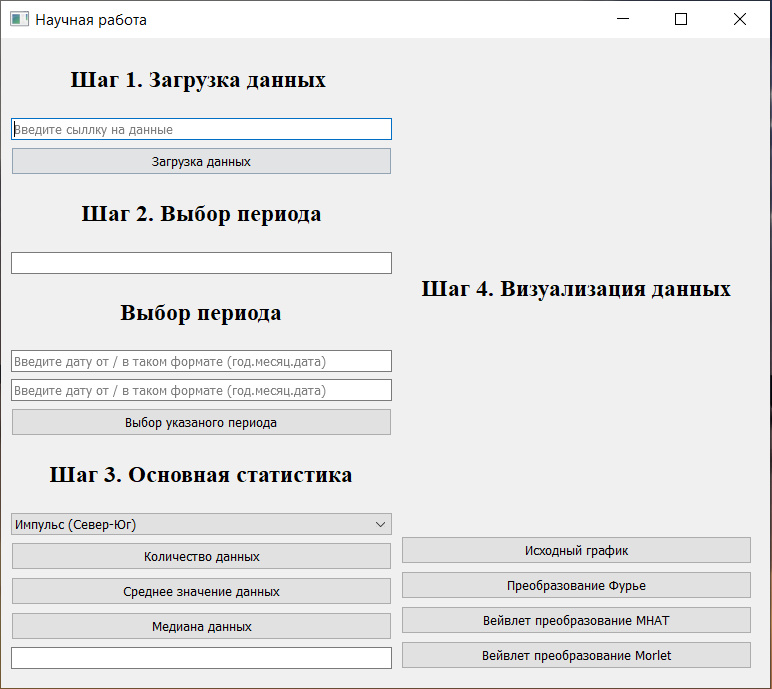


Рисунок 1 – Интерфейс программы

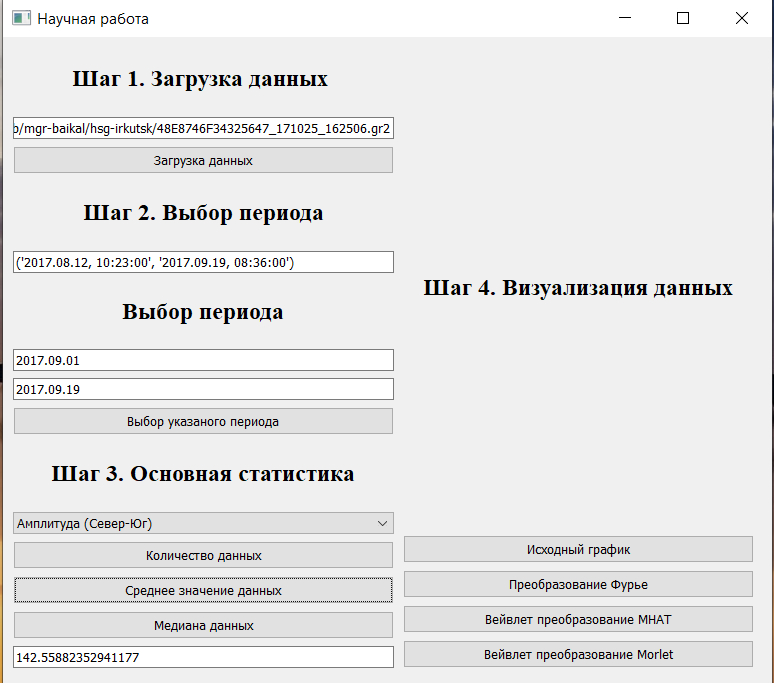


Рисунок 2 – Выбор периода, описательная статистика

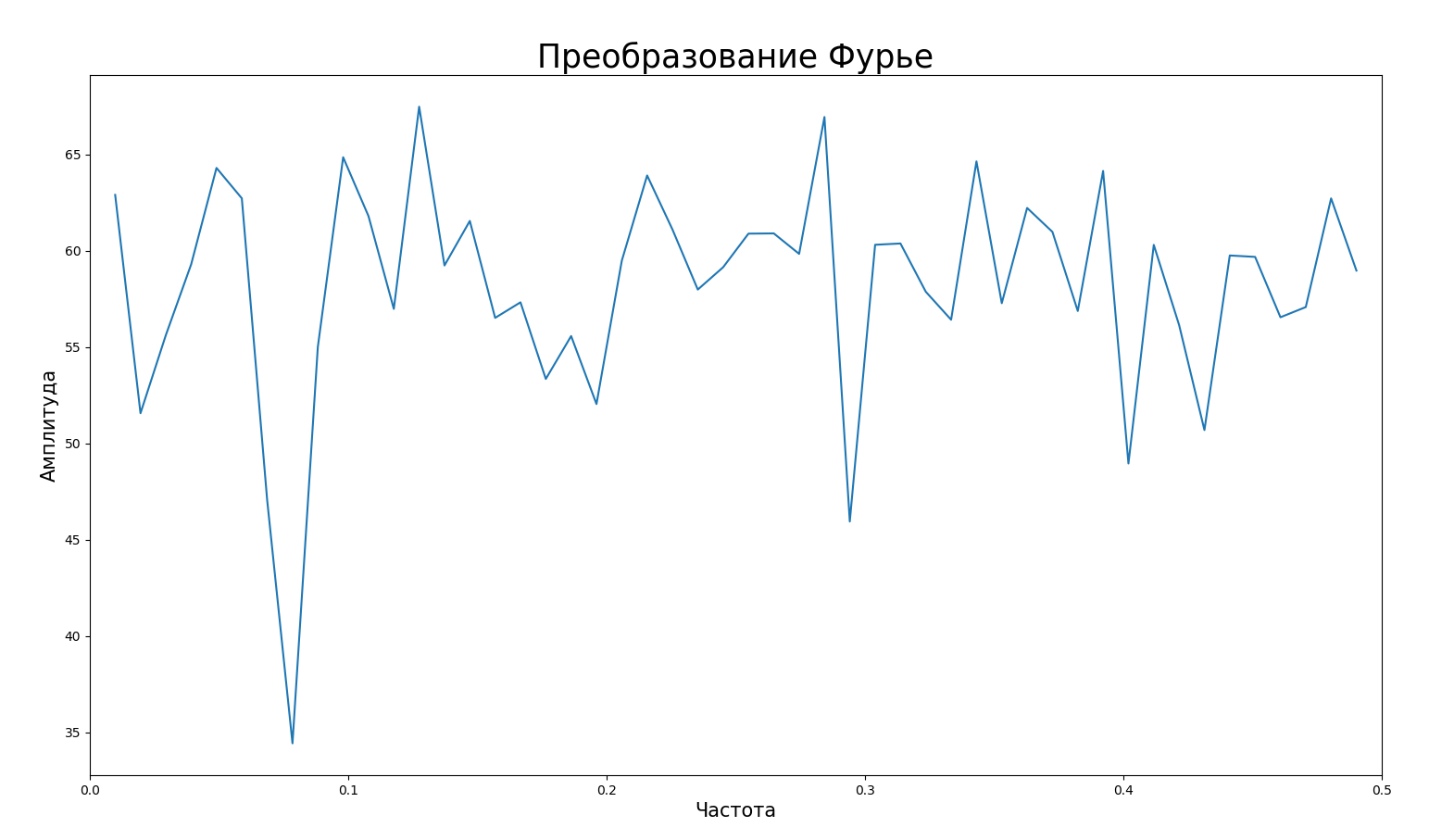


Рисунок 3 – Преобразование Фурье



Рисунок 4 – Вейвлет преобразование МHAT



Рисунок 5 – Вейвлет преобразование Морлет

# Вывод

В ходе выполнения исследовательской работы была реализована программа, которая позволяет загружать данные с сайта и получать статистику и визуализацию данных, преобразование Фурье и Вейвлет-преобразование.