Модуль 1 · Python и анализ данных

Введение в среду Python и первичная обработка климатических данных

Ph.D., старший преподаватель Мадина Турсумбаева

Кафедра метеорологии и гидрологии

Курс: Современные цифровые технологии в метеорологии

Цель лекции

Познакомить со средой Python (Miniconda, Jupyter Notebook) и на примере ряда глобальной температуры показать полный цикл первичной обработки климатических данных: прочитать, осмотреть, проверить, обработать, представить.

Лекция — первая из десяти. Цикл «прочитать → осмотреть → проверить → обработать → представить» повторяется в каждой следующей лекции.

Результаты обучения

После лекции студент будет:

Знать

Четыре базовых типа в Python: float, int, str, bool

Списки, циклы for и функции def

Что такое DataFrame и чем он отличается от списка

Уметь

Прочитать CSV-файл через pandas

Осмотреть таблицу и проверить пропуски и повторы

Посчитать аномалию относительно нормы и построить график

Понимать

Зачем сначала проверять качество данных

Почему исходный файл не меняют

Почему работа в коде воспроизводима

Курс: три модуля

Модуль 1 · Лекции 1–7

Python и анализ данных

Таблицы, статистика, станции, радиозонды

Модуль 2 · Лекции 8–11

Данные на сетке

NetCDF, реанализ ERA5, сравнение с наблюдениями

Модуль 3 · Лекции 12–15

Пространственный анализ

Карты, ГИС, Google Earth Engine

Цель курса: самостоятельно получить данные, обработать их и представить результат.

Каждая лекция построена вокруг реального набора данных; на семинаре вы повторяете и расширяете анализ своими руками.

Данные современной метеорологии

Станции

многолетние ряды наблюдений у поверхности

Радиозонды

вертикальный профиль давления, температуры, влажности и ветра

Спутники и радары

поля облачности и осадков

Модели

прогнозы и климатические сценарии

Реанализ ERA5

Каждый час с 1940 года до настоящего времени · сетка 0,25° · 137 уровней по вертикали

- Такие данные не помещаются в электронную таблицу.
- Нужен инструмент: разные форматы, большие объёмы и повторяемый расчёт.

Почему Python?

1

Бесплатный и открытый

Лицензия не нужна; работает в любой организации.

2

Простой синтаксис

Код читается почти как текст.

3

Библиотеки для наук о Земле

pandas, xarray, Cartopy, MetPy и другие.

4

Воспроизводимость

Программа — точная запись всех шагов. Её можно запустить заново, проверить и передать коллеге.

Установка: Miniconda и окружения

- Miniconda или Anaconda — Python и менеджер пакетов conda.
- Окружение — отдельная папка со своей версией Python и набором библиотек: проекты не конфликтуют.
- Для курса используйте одно окружение и не устанавливайте пакеты в базовое.

Команды вводятся в терминале (Windows — Anaconda Prompt)

```
# 1. создать окружение с Python 3.12
conda create -n meteo python=3.12
# 2. активировать окружение
conda activate meteo
# 3. установить библиотеки курса
conda install -c conda-forge numpy pandas matplotlib seaborn jupyterlab
conda install -c conda-forge xarray netcdf4 cartopy metpy scikit-learn
# 4. запустить Jupyter Lab
jupyter lab
```

Jupyter Notebook: код, текст и результат вместе

1

Текстовая ячейка

Пояснения и выводы. Пишите их для себя: через месяц они нужнее кода.

2

Ячейка кода

Команды Python. Результат выводится прямо под ячейкой.

3

Shift + Enter

Выполнить текущую ячейку и перейти к следующей.

4

Перед сдачей

Перезапустите ядро и выполните все ячейки сверху вниз.

Главная ошибка новичка

Ячейка выполнена раньше, а потом её код изменили. Результат на экране уже не соответствует коду. Поэтому перед сдачей всегда делайте полный перезапуск.

Основы Python: переменные и типы данных

```
# переменная = значение
t_c = 25.4              # float
n_obs = 8               # int
station = "Алматы"      # str
is_rain = False         # bool
t_k = t_c + 273.15
print(station, t_k)
```

Результат (округлён)

```
Алматы 298.55
type(t_c)  →  float
```

Типичная ошибка

Запятая в числе: 25,4 — это не число, а два значения через запятую. Пишите точку.

- Знак = означает «присвоить значение» переменной.
- Текст в кавычках имеет тип str; без кавычек Python считает это именем переменной.
- Всё после # — комментарий, который Python пропускает.

Списки, циклы и функции

```
temps = [12.1, 14.3, 15.0, 13.2, 9.8]
temps[0]          # первый элемент: 12.1
temps[-1]         # последний элемент: 9.8

def c_to_k(t):
    return t + 273.15

for t in temps:
    print(c_to_k(t))
# печатает пять значений в кельвинах
```

Список

Набор значений в квадратных скобках. Нумерация начинается с нуля.

Функция def

Одно действие, которое можно повторять с разными входами.

Цикл for

Повторяет действие для каждого элемента. Отступ — часть синтаксиса.

Отступ в четыре пробела обязателен: без него Python выдаст ошибку.

NumPy: вычисления без циклов

```
import numpy as np
# среднемесячные температуры, °C (условные)
t = np.array([-8.2, -5.1, 2.4, 11.6, 16.9, 21.8,
              24.3, 23.1, 17.5, 9.6, 1.2, -5.4])
t_k = t + 273.15          # все 12 значений сразу
t.mean()                  # среднее за год
t.max() - t.min()         # годовая амплитуда
(t > 20).sum()            # месяцев теплее 20 °C
```

9,14 °C

среднее за год

32,5 °C

годовая амплитуда (max − min)

3

месяца теплее 20 °C

- Массив — основной объект NumPy: операция применяется ко всем элементам сразу (векторизация).
- Код короче и заметно быстрее явного цикла по элементам.

Климатические данные в формате CSV

- Разделитель: запятая, точка с запятой или табуляция.
- Заголовок: есть ли первая строка с именами столбцов.
- Кодировка: UTF-8 или cp1251 — частая проблема файлов из России и Казахстана.
- Десятичный знак: точка или запятая.
- Пропуски: пустая ячейка, NaN или условный код (например, −999).

Фрагмент файла annual.csv

```
Source,Year,Mean
GCAG,1850,-0.4265
GCAG,1851,-0.2635
GCAG,1852,-0.2241
# ...
GISTEMP,2025,1.1850
```

Сначала откройте файл как текст и посмотрите на первые строки. Только потом читайте его в pandas.

Источники

Набор global-temp (datahub.io/core/global-temp), NASA GISTEMP и NOAA; для второй части курса — Our World in Data.

Чтение файла: pandas.read_csv

```
import pandas as pd
df = pd.read_csv("annual.csv")
df.head()              # первые 5 строк
df.shape               # (строки, столбцы)
```

Результат

```
  Source  Year    Mean
0   GCAG  1850 -0.4265
1   GCAG  1851 -0.2635
2   GCAG  1852 -0.2241
(323, 3)
```

Что означают столбцы

Source — источник данных (два разных ряда в одном файле)

Year — год

Mean — среднегодовая аномалия, °C

323 строки, 3 столбца

323 строки — это два ряда (GCAG и GISTEMP), а не один ряд длиной 323 года.

Первый осмотр: info, describe, value_counts

```
df.info()                          # типы и пропуски
df["Source"].value_counts()        # строк у каждого источника
df.describe().round(2)             # статистика числовых столбцов
```

Результат: value_counts

```
GCAG       177
GISTEMP    146
```

	Year	Mean
count	323	323
mean	1944,55	0,01
std	47,88	0,41
min	1850	−0,60
max	2026	1,28

Вывод

В одном файле два разных ряда: GCAG (NOAA) и GISTEMP (NASA). Смешивать их нельзя — сначала выберите один ряд, и только потом считайте.

Выбор строк по условию

```
# только ряд NASA GISTEMP
g = df[df["Source"] == "GISTEMP"]
g = g.sort_values("Year").reset_index(drop=True)
len(g)                              # 146 лет
g.nlargest(5, "Mean")               # пять самых тёплых лет
```

Год	Аномалия, °C
2024	1,2817
2025	1,1850
2023	1,1675
2016	1,0100
2020	1,0050

Логическая индексация

Условие даёт True или False для каждой строки.

В результат попадают строки, где стоит True.

Три самых тёплых года (2023–2025) — последние три года ряда.

Один знак «=» присваивает значение, а «==» — сравнивает. Путаница — частая ошибка.

Контроль качества: пропуски, повторы, разрывы

1

Пропуски

isna().sum() — есть ли NaN и в каких столбцах

2

Повторы

duplicated() — нет ли двух строк на один год

3

Разрывы

diff().max() — не выпали ли годы из ряда

4

Коды пропусков

na_values=[−999] — заменить условный код на NaN

```
g.isna().sum()                    # пропуски
g["Year"].duplicated().sum()      # повторы
g["Year"].diff().max()            # разрывы, в годах
pd.read_csv("station.csv", na_values=[-999, -9999])
```

Результат

```
Пропуски: 0 во всех столбцах
Повторы: 0
Наибольший разрыв: 1,0 года
```

Один код −999 в данных станции сильно искажает среднее. Сначала контроль качества, потом анализ.

Первая обработка: норма и аномалия

```
# средняя за 1991–2020 гг. (норма ВМО)
sel = (g["Year"] >= 1991) & (g["Year"] <= 2020)
base = g[sel]["Mean"].mean()
g["Anom_9120"] = (g["Mean"] - base).round(3)
g.to_csv("gistemp_clean.csv", index=False)
```

Результат

```
base = 0,611
2024: аномалия 0,670
2025: аномалия 0,574
```

- & означает «и»; каждое условие заключается в скобки.
- Новый столбец создаётся записью в квадратных скобках.
- Аномалия = значение − норма. Знак показывает, теплее или холоднее нормы.
- Исходный файл не перезаписываем: результат сохраняем в новый.

Первый график: Matplotlib

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots(figsize=(9, 4))
ax.plot(g["Year"], g["Mean"], color="tab:red")
ax.axhline(0, color="gray")
ax.set_xlabel("Год")
ax.set_ylabel("Аномалия, °C")
ax.set_title("Аномалия глобальной температуры")
fig.savefig("gistemp.png", dpi=300)
```

fig — рисунок, ax — оси. Подписи осей и единицы измерения обязательны. Источник данных — в подписи под рисунком.

Данные: NASA GISTEMP v4; аномалии относительно 1951–1980 гг.

```
# упражнение: сравните два периода
early = g[g.Year.between(1880, 1909)]
ref = early["Mean"].mean()
recent = g[g.Year >= 2016]["Mean"].mean()
print(round(recent - ref, 2))
```

Задание

Почему разность двух средних аномалий одной базы даёт реальную разницу температур? Объясните в отчёте.

Типичные ошибки и их исправление

Ошибка	Что происходит	Как исправить
Запятая вместо точки в числах	Число читается как текст, среднее не считается	Указать decimal="," в read_csv или заменить символ
Не указан код пропуска	−999 попадает в среднее	na_values=[−999] при чтении
Смешаны два источника	Ряды накладываются, тренд неверен	Сначала фильтровать по Source
Исходный файл перезаписан	Расчёт нельзя повторить	Результат сохранять в новый файл
Блокнот выполнен не по порядку	Результат не соответствует коду	Restart & Run All перед сдачей

Заключение

1

Прочитать

read_csv: подобрать разделитель, кодировку и строку заголовка

2

Осмотреть

head, info, describe, value_counts

3

Проверить

isna, duplicated, diff, коды пропусков

4

Обработать

условия, новые столбцы, норма и аномалия

5

Представить

plot с подписями, to_csv в новый файл

Главный вывод

Любая работа с климатическими данными проходит пять шагов: прочитать, осмотреть, проверить, обработать, представить. Проверка качества идёт до расчётов, исходный файл не меняется, а код делает анализ воспроизводимым.

Вопросы для самопроверки

1. Что произойдёт с выражением 25.4 + "1" и почему?
2. Зачем перед расчётом смотреть на столбец Source?
3. Почему значения в столбце Mean — аномалии, а не температура?
4. Почему пропуски не заменяют нулём?

Семинар 1 и следующая лекция

Семинар 1 · Установка Python и анализ выбросов CO_2

Установить Python и Jupyter Notebook.

Загрузить набор данных Our World in Data по глобальным выбросам CO_2.

Выполнить первичную обработку, построить график и определить страны-лидеры.

Лекция 2

Обработка и визуализация табличных климатических данных.

Выбор столбцов и строк, пропуски, группировка и графики по странам.

Перед семинаром: установите окружение по слайду 6 и проверьте, что jupyter lab запускается.

Литература

- Harris, C. R., et al. (2020). Array programming with NumPy. Nature, 585, 357–362. https://doi.org/10.1038/s41586-020-2649-2
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90–95.
- McKinney, W. (2022). Python for Data Analysis (3rd ed.). O’Reilly Media. https://wesmckinney.com/book/
- VanderPlas, J. (2016). Python Data Science Handbook. O’Reilly Media. https://jakevdp.github.io/PythonDataScienceHandbook/
- pandas development team. pandas documentation. https://pandas.pydata.org/docs/
- GISTEMP Team (2025). GISS Surface Temperature Analysis, version 4. NASA GISS. https://data.giss.nasa.gov/gistemp/
- Hersbach, H., et al. (2020). The ERA5 global reanalysis. QJRMS, 146, 1999–2049. https://doi.org/10.1002/qj.3803
- Набор данных Global Temperature Time Series: https://datahub.io/core/global-temp
- Ritchie, H., Rosado, P., & Roser, M. CO_2 and Greenhouse Gas Emissions. Our World in Data. https://ourworldindata.org/co2-and-greenhouse-gas-emissions