INFORMATION TECHNOLOGY
ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

UDC 004

https://doi.org/10.23947/2587-8999-2025-9-2-52-64

Original Empirical Research

Automated Processing of Primary Field Data on the Behavior of Natural-Technological Systems under Climate Change and Anthropogenic Impacts in the Far North

Sergey V. Kushukov[1], **Konstantin N. Ivanov**[1], **Sergey P. Levashkin**[1] ✉, **Mikhail V. Yakobovskiy**[2]
[1] Povolzhskiy State University of Telecommunications and Informatics, Samara, Russian Federation
[2] Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences (IPM RAS), Moscow, Russian Federation
✉ ai_lab@psuti.ru

Abstract
Introduction. This work addresses the scientific problem of studying natural-technological systems (NTS) of the Far North under conditions of climate change and anthropogenic impacts. The relevance of ensuring their stability is emphasized, which requires a comprehensive analysis of field data. Problems in automated processing methods of such specific data have been identified. The aim of the study is to develop automated methods for processing field data to reveal patterns. Python libraries for data analysis, processing, and visualization are used as tools.
Materials and Methods. The research object is described — the Main Building of the Yakutsk Thermal Power Plant (TPP) in permafrost conditions. The study materials include field data obtained from engineering-geological boreholes at the Yakutsk TPP, monitoring stations Chabyda and Tuymaada, as well as a section of the Amur-Yakutsk railway (AYR). The data include measurements of soil temperature and moisture, seasonal thaw layer dynamics, snow cover characteristics, and others. A detailed sequence of automated processing of primary data from XLS files using the pandas library is presented, including reading, cleaning, format conversion, filling or replacing missing values, removing duplicates, and saving processed data in CSV, JSON, and XLSX formats.
Results. Specific results of automated processing and systematization of primary field data are presented. Heterogeneous measurements were successfully unified into a single format, ensuring their proper use. A unique data array was formed based on empirical observations under the specific conditions of the Far North. The practical application of Python libraries for executing key stages of preprocessing and data preparation is demonstrated.
Discussion and Conclusion. It is shown that the application of a systematic approach and automated data processing significantly improves the quality and reliability of natural-technological system data analysis. Handling missing data and normalization enhance accuracy, and the final data formats are convenient for further modeling. The universality of Python is highlighted. Prospects for further research include applying machine learning, clustering, and modeling methods aimed at uncovering patterns and forecasting the behavior of natural-technological systems in the Far North under climate and anthropogenic influences.

Keywords: data analysis, Python libraries, data preparation, data preprocessing, information technologies.

Funding. This work was supported financially by the Russian Science Foundation (grant No. 23–61–10032).

For Citation. Kushukov S.V., Ivanov K.N., Levashkin S.P., Yakobovsky M.V. Automated Processing of Primary Field Data on the Behavior of Natural-Technological Systems under Climate Change and Anthropogenic Impacts in the Far North. *Computational Mathematics and Information Technologies*. 2025;9(2):52–64. https://doi.org/10.23947/2587-8999-2025-9-2-52-64

Оригинальное эмпирическое исследование

Автоматическая обработка первичных данных натурных исследований поведения природно-технических систем при изменении климата и антропогенных воздействиях в условиях Крайнего Севера

С.В. Кушуков[1], **К.Н. Иванов**[1], **С.П. Левашкин**[1]✉, **М.В. Якобовский**[2]

[1] Поволжский государственный университет телекоммуникаций и информатики, г. Самара, Российская Федерация

[2] Институт прикладной математики имени М.В. Келдыша РАН, г. Москва, Российская Федерация

✉ ai_lab@psuti.ru

Аннотация

Введение. Рассматривается научная проблема изучения природно-технических систем (ПТС) Крайнего Севера в условиях изменения климата и антропогенных воздействий. Отмечается актуальность задачи обеспечения их устойчивости, что требует комплексного анализа натурных данных. Выявлены проблемы в методах автоматизированной обработки таких специфических данных. Целью работы является разработка автоматизированных методов обработки натурных данных для выявления закономерностей. В качестве инструментов используются библиотеки Python для анализа, обработки и визуализации данных.

Материалы и методы. Описан объект исследования — Главный корпус Якутской ТЭЦ в условиях вечной мерзлоты. В качестве материалов исследования использованы натурные данные, полученные из инженерно-геологических скважин Якутской ТЭЦ, стационаров Чабыда и Туймаада, а также железнодорожного участка Амуро-Якутской магистрали (АЯМ). Данные включают измерения температуры и влажности грунтов, динамики сезонноталого слоя, характеристик снежного покрова и др. Представлена детальная последовательность автоматизированной обработки первичных данных из XLS-файлов с использованием библиотеки pandas, включая чтение, очистку, преобразование форматов, заполнение или замену значений, удаление дубликатов, а также сохранение обработанных данных в форматах CSV, JSON и XLSX.

Результаты исследования. Представлены конкретные результаты автоматизированной обработки и систематизации первичных натурных данных. Успешно выполнено приведение разнородных измерений к единому формату, обеспечивающему их корректное использование. Сформирован уникальный массив данных на основе эмпирических наблюдений в специфических условиях Крайнего Севера. Продемонстрировано практическое применение библиотек Python для выполнения основных этапов предобработки и подготовки данных.

Обсуждение и заключение. Доказано, что применение системного подхода и автоматизированной обработки данных существенно повышает качество и надежность анализа натурных данных ПТС. Устранение пропусков и нормализация данных улучшают точность, а итоговые форматы данных удобны для дальнейшего использования в моделировании. Подчеркивается универсальность применения Python. Обозначены перспективы исследования — применение методов машинного обучения, кластеризации и моделирования, предназначенных для выявления закономерностей и прогнозирования поведения природно-технических систем в условиях Крайнего Севера под воздействием климатических и антропогенных факторов.

Ключевые слова: анализ данных, библиотеки Python, подготовка данных, предобработка данных, информационные технологии

Финансирование. Работа выполнена при финансовой поддержке РНФ (грант № 23–61–10032).

Для цитирования. Кушуков С.В., Иванов К.Н., Левашкин С.П., Якобовский М.В. Автоматическая обработка первичных данных натурных исследований поведения природно-технических систем при изменении климата и антропогенных воздействиях в условиях Крайнего Севера. *Computational Mathematics and Information Technologies*. 2025;9(2):52–64. https://doi.org/10.23947/2587-8999-2025-9-2-52-64

Introduction. The study of natural-technological systems (NTS) under climate change and anthropogenic impacts, especially in the challenging conditions of the Far North, represents an important scientific problem. These systems are formed as a result of interactions between natural processes and technical objects, and their stability largely depends on the dynamics of external factors. To identify patterns of NTS functioning and predict possible changes, a comprehensive data analysis is required.

Information processing obtained during field studies involves several stages: collection of primary data, their preliminary processing, analysis, and interpretation of results. The goal of this work is to develop methods for automated processing of field data, which will allow identification of key trends and patterns in the interaction between natural and technogenic factors [1–3].

The relevance of the research is driven by the need to make informed decisions to ensure sustainable functioning and management of natural-technological systems under increasing climatic and anthropogenic impacts [2–8].

First, we present algorithmic libraries [4] that provide diverse tools for data processing, analysis, and modelling [5]:

• scikit-learn: a universal machine learning toolkit covering the entire process from data preparation to model evaluation. This library includes all necessary functions to solve machine learning and statistical analysis tasks [9, 10];

• statsmodels: a library for estimating statistical models, offering tools for statistical tests, regression analysis, and time series analysis.

Next, we list visualization libraries that enable effective data presentation:

• matplotlib: helps create static, interactive, and animated graphs in Python. It provides a broad arsenal of tools for visualizing data in various formats, revealing a wide range of possibilities for illustrative information presentation [11–14];

• seaborn: a data visualization library built on top of matplotlib, offering a high-level interface for creating informative statistical graphics.

Additionally, we mention scientific computing libraries used for mathematical calculations and data processing:

• pandas: a framework for data manipulation and analysis. It provides powerful tools including data structures like DataFrame, which simplify working with tabular data. The library is widely used in loading, cleaning, analyzing, and preparing data before integration into machine learning algorithms;

• numpy: a library designed for manipulating multidimensional arrays and matrices, as well as performing mathematical operations on them. It offers a variety of functions optimized for efficient numerical data processing;

• scipy: a highly efficient library for scientific and technical computations, extending functionalities for optimization, integration, interpolation, and many other scientific tasks.

Materials and Methods

Review of Existing Research. In recent years, there has been a growing interest in studying the influence of climatic and anthropogenic factors on natural-technological systems (NTS). Various studies emphasize the necessity of a comprehensive approach to managing these systems to ensure their stability and functionality.

R.S. Rozhkov and co-authors, in their work “Management of Natural-Technological Systems through the Concept of Sustainable Development and Acceptable Risk” [15], highlight the importance of considering environmental, economic, and social factors in the management of NTS. The authors propose using systems analysis and risk assessment to develop optimal strategies aimed at reducing anthropogenic load and improving the ecological state of the environment.

N. Dregulo investigates the impact of climatic factors on the operation of NTS related to wastewater treatment. In [16], it is emphasized that increased atmospheric precipitation can adversely affect the operational and environmental performance of such systems, necessitating the revision of regulatory criteria and adaptation to changing climatic conditions.

The article “Impact of Anthropogenic Factors on Urban Heat Pollution” [17] presents a comprehensive assessment of human activity’s effects on atmospheric heat pollution. It is noted that a significant share of heat pollution is contributed by heat consumption and transportation, underscoring the need to develop measures to reduce anthropogenic thermal impact in urbanized areas.

A study published on the website of LLC “Biochem-TL” analyzes the influence of anthropogenic factors on agricultural development [18]. Particular attention is given to soil changes caused by prolonged use of fertilizers and livestock waste, which affect the ecological condition of agricultural landscapes.

Taken together, these studies highlight the necessity of a comprehensive and adaptive approach to managing natural-technological systems amid changing climate and increasing anthropogenic pressure. Accounting for the specifics of each NTS component and continuous monitoring of their condition are key elements to ensure their sustainable functioning.

Research Subject Area. The object of the study is the Main Building of the Yakutsk Thermal Power Plant (Yakutsk TPP) — a unique engineering structure constructed on permafrost (cryolithozone) conditions. This facility is particularly interesting because it was the first industrial building in the USSR erected on permafrost soils following the pioneering principle of construction. This means that during construction, measures were taken to preserve the frozen soils as a reliable foundation for the power plant. Construction began in 1933, with active participation from scientists specializing in permafrost studies. In 1937, the TPP was commissioned and has since been supplying electricity to Yakutsk, and since 1961 — also heat.

To understand how the TPP operates under permafrost conditions and how it is affected by various external factors, a computational model was developed. This model covers an area of 180 by 150 meters and extends to a depth of up to 30 meters, enabling analysis of the permafrost soils (multi-year frozen ground, MYFG) on which the plant stands. The model is based on data from engineering-geological boreholes. These data allow determining the behavior of soils under varying temperature and external conditions, which is crucial for ensuring the stable operation of the TPP.

Special attention in the study is given to the thermal regime of the surrounding environment. Average monthly air temperature data for the Yakutsk region over the past 10 years were collected and analyzed. These data are significant for understanding how climatic conditions change and how these changes may affect the state of frozen soils and the operation of the TPP. It is important to note that these data differ from those provided by the Hydrometeorological Service based on norms established since 1966. Normative data are statistical indicators obtained by long-term averaging of climatic observations (usually over a 30-year period). These indicators serve as reference values for assessing the climatic conditions of the region. However, under current climate change conditions, if these norms are not regularly updated, they may not reflect modern trends and often underestimate temperature values, limiting their applicability in analyzing the current climate state.

For studying thermodynamic processes in permafrost soils, field data obtained from engineering-geological boreholes and stationary observation sites were used. These data represent detailed time series covering multi-year measurements of key parameters of permafrost soils in various regions of Yakutia.

Data were collected from three main sources:

• Yakutsk Thermal Power Plant (TPP) — 8 engineering-geological boreholes where systematic measurements of soil temperature and moisture are conducted, as well as studies of the lithological composition, physical, and thermophysical properties of frozen soils;

• Chabyda and Tuymaada observation stations — 16 sites where soil temperature and moisture, seasonal dynamics of thaw depth, interannual variability of the seasonally thawed layer (STS), snow cover height, snow density, and physical-mechanical soil properties are recorded;

• Railway section of the Amur-Yakutsk railway line — 41 engineering-geological boreholes collecting data on soil temperature and moisture, variability of the STS, snow cover height, and other geocryological parameters.

Unlike generalized climatic norms presented in official sources, these data reflect the actual processes occurring in permafrost soils, with high resolution in depth and time. This makes them a unique source of information for studying the stability of natural-technical systems of the Far North under conditions of climate change and anthropogenic impacts.

Data Processing and Preliminary Analysis. Consider the preprocessing of data using the example of a monitoring dataset from sixteen sites in Yakutia, provided by the Permafrost Institute of the Siberian Branch of the Russian Academy of Sciences (SB RAS). Each site contains its own observational data. For example, "Site 9_Chabyda" includes information on soil temperature, moisture, seasonal dynamics of thaw depth, interannual variability of the seasonally thawed layer, monthly snow height, interannual dynamics of snow density, seasonal dynamics of snowpack density and height, lithological profile, and physical and thermophysical properties of soil and surface covers.

The objective of the study is to prepare data analysis using machine learning methods and to find relationships between their attributes. It should be emphasized that the processes automated by the authors are usually performed manually or using interactive methods. As a result of this interaction, structured and analysis-ready datasets are formed, transformed from the raw data initially provided.

The structural scheme of the automatic data processing workflow is shown in Fig. 1.

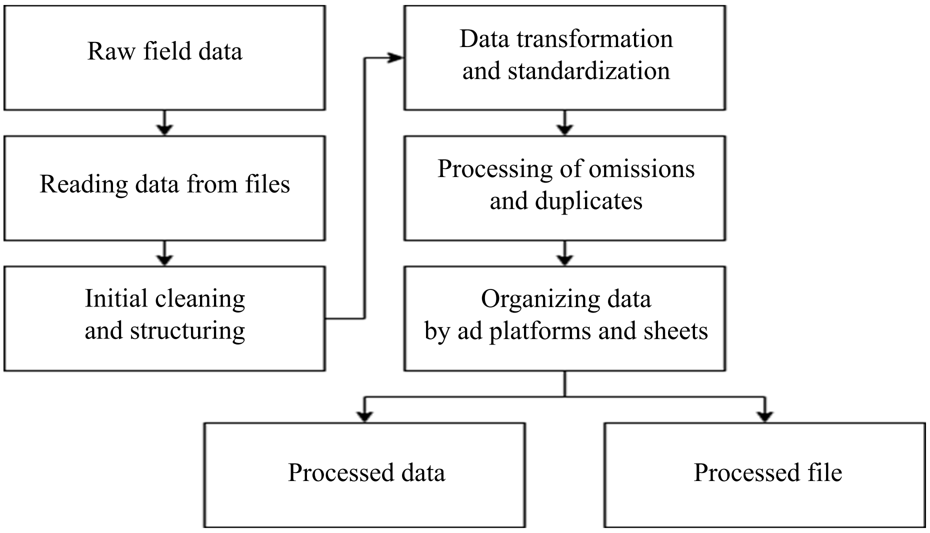

Raw field data
Reading data from files
Initial cleaning and structuring
Data transformation and standardization
Processing of omissions and duplicates
Organizing data by ad platforms and sheets
Processed data
Processed file

Fig. 1. Structural scheme of automated primary data processing

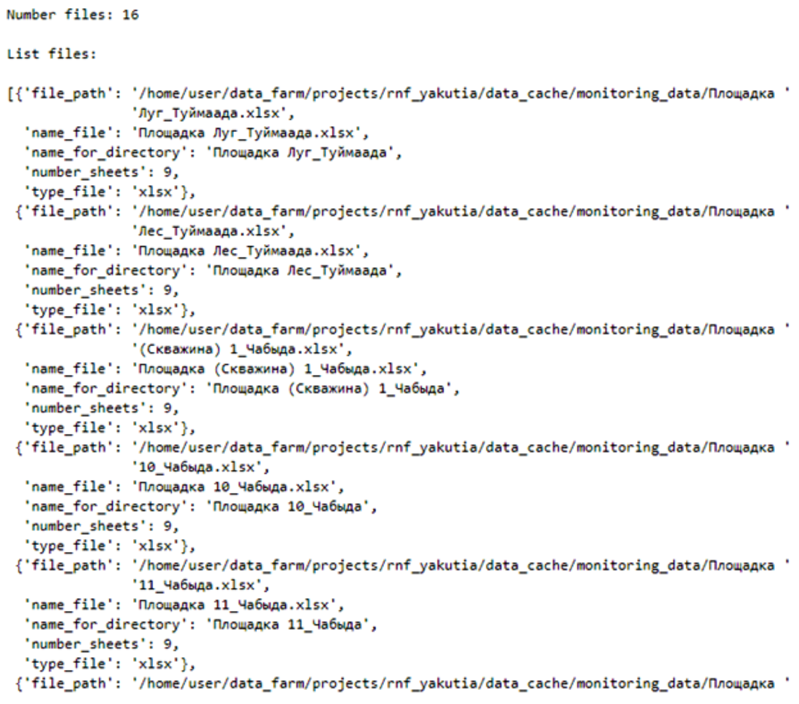
For data preprocessing, it is necessary to read data from XLS files (the original format of the monitoring data) and load them into a DataFrame object using the pandas library. This stage includes detecting and handling missing values, choosing a strategy for filling gaps in columns, as well as removing irrelevant features.

To analyze the dataset, all relevant data should be prepared and the described Python modules applied in practice [7–8]. The first stage is extracting data from the XLS file. Efficient storage and processing of tabular data are critical aspects when analyzing data.

We obtain information about the number of files per site:

```
def get_excel_files_info(folder_path: str):
""" """
excel_files_info = []
for filename in tqdm(os.listdir(folder_path)):
if filename.endswith('.xls') or filename.endswith('.xlsx'):
file_path = os.path.join(folder_path, filename)
if filename.endswith('.xls'):
xls_workbook = xlrd.open_workbook(file_path)
number_sheets = xls_workbook.nsheets
else:
workbook = load_workbook(file_path)
number_sheets = len(workbook.sheetnames)
name_for_directory, extension = os.path.splitext(filename)
file_info = {
'file_path': file_path,
'name_file': filename,
'type_file': 'xls' if filename.endswith('.xls') else 'xlsx',
'number_sheets': number_sheets,
'name_for_directory': name_for_directory,
}
excel_files_info.append(file_info)
return excel_files_info[::-1]
raw_data = get_excel_files_info(data_cache)
print(f'Number files: {len(raw_data)}\n\nList files:\n')
pprint.pprint(raw_data).
```

After executing the code, the variable raw_data will contain information about the files located in the directory, which allows for easy viewing of the contents of these files (Fig. 2).



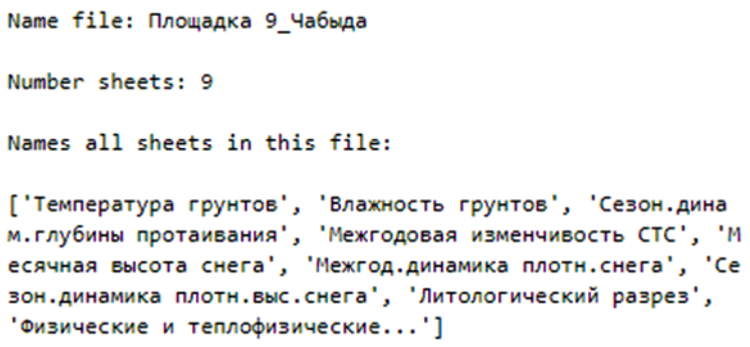
```
Number files: 16

List files:

[{'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
               'Луг_Туймаада.xlsx',
  'name_file': 'Площадка Луг_Туймаада.xlsx',
  'name_for_directory': 'Площадка Луг_Туймаада',
  'number_sheets': 9,
  'type_file': 'xlsx'},
 {'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
               'Лес_Туймаада.xlsx',
  'name_file': 'Площадка Лес_Туймаада.xlsx',
  'name_for_directory': 'Площадка Лес_Туймаада',
  'number_sheets': 9,
  'type_file': 'xlsx'},
 {'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
               '(Скважина) 1_Чабыда.xlsx',
  'name_file': 'Площадка (Скважина) 1_Чабыда.xlsx',
  'name_for_directory': 'Площадка (Скважина) 1_Чабыда',
  'number_sheets': 9,
  'type_file': 'xlsx'},
 {'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
               '10_Чабыда.xlsx',
  'name_file': 'Площадка 10_Чабыда.xlsx',
  'name_for_directory': 'Площадка 10_Чабыда',
  'number_sheets': 9,
  'type_file': 'xlsx'},
 {'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
               '11_Чабыда.xlsx',
  'name_file': 'Площадка 11_Чабыда.xlsx',
  'name_for_directory': 'Площадка 11_Чабыда',
  'number_sheets': 9,
  'type_file': 'xlsx'},
 {'file_path': '/home/user/data_farm/projects/rnf_yakutia/data_cache/monitoring_data/Площадка '
```

Fig. 2. File information

Next, we analyze the number of sheets in each dataset file:

```
def read_non_empty_excel(file_path, sheet_name):
""" """
df = pd.read_excel(file_path, sheet_name=sheet_name)
return df if not df.empty else None
raw_file = raw_data[15]
state_name = raw_file.get('name_for_directory')
df = pd.ExcelFile(raw_file.get('file_path'))
dfs = {name_sheet: read_non_empty_excel(df, name_sheet) for name_sheet in df.sheet_names
if read_non_empty_excel(df, name_sheet) is not None}
key_list = list(dfs.keys())
print(f'Name file: {state_name}\n\nNumber sheets: {len(key_list)}\n\nNames all sheets in
this file:\n\n{key_list}').
```

After running the code, we will have information about the number of sheets in the dataset, as well as the name of each sheet, which allows us to directly access a specific sheet by its name (Fig. 3).



```
Name file: Площадка 9_Чабыда

Number sheets: 9

Names all sheets in this file:

['Температура грунтов', 'Влажность грунтов', 'Сезон.дина
м.глубины протаивания', 'Межгодовая изменчивость СТС', 'М
есячная высота снега', 'Межгод.динамика плотн.снега', 'Се
зон.динамика плотн.выс.снега', 'Литологический разрез',
'Физические и теплофизические...']
```

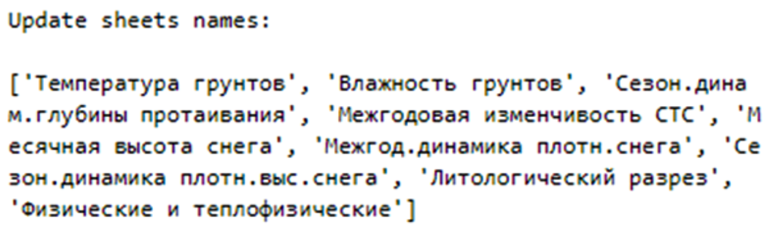
Fig. 3. Output of sheet information

However, it should be noted that some datasets contain errors in the sheet names that need to be corrected to avoid problems in further analysis:

```
# Naming errors 1
if 'Interannual snow density dynamics' in dfs:
   dfs['Interannual snow density dynamics'] = dfs.pop('Interannual snow dnamics density')
if 'Interannual snow density dynamics ' in dfs:
   dfs['Interannual snow density dynamics'] = dfs.pop('Interannual snow density dynamics ')
if 'Interannual snow density dynamics' in dfs:
   dfs['Interannual snow density dynamics'] = dfs.pop('Interannual snow density dynamics')
# Naming errors 2
if 'Seasonal dynamics of thaw depth' in dfs:
    dfs['Seasonal thaw depth dynamics'] = dfs.pop('Seasonal dynamics of thaw depth')
if 'Seasonal thaw depth dynamics' in dfs:
    dfs['Seasonal thaw depth dynamics'] = dfs.pop('Seasonal thaw depth dynamics')
if 'Seasonal dynamics of snow density and height' in dfs:
     dfs['Seasonal snow density and height dynamics'] = dfs.pop('Seasonal dynamics of
snow density and height')
if 'Seasonal snow density and height dynamics' in dfs:
     dfs['Seasonal snow density and height dynamics'] = dfs.pop('Seasonal snow density
and height dynamics')
# Naming errors 3
if 'Physical and thermophysical...' in dfs:
    dfs['Physical and thermophysical'] = dfs.pop('Physical and thermophysical...')
if 'Physical and thermophysical (typo)...' in dfs:
   dfs['Physical and thermophysical'] = dfs.pop('Physical and thermophysical (typo)...')
key_list = list(dfs.keys())
```

```
print(f'Updated sheet names:\n\n{key_list}').
```

After executing the code, all the sheet names in the document were corrected (Fig. 4).

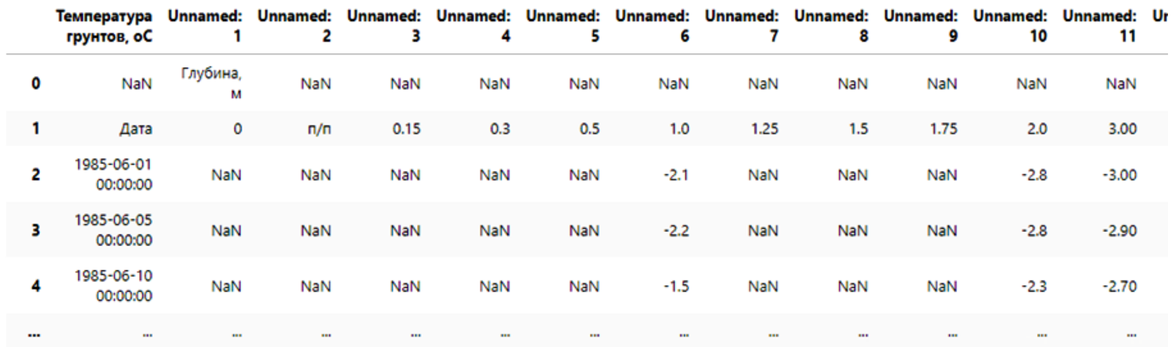


```
Update sheets names:

['Температура грунтов', 'Влажность грунтов', 'Сезон.дина
м.глубины протаивания', 'Межгодовая изменчивость СТС', 'М
есячная высота снега', 'Межгод.динамика плотн.снега', 'Се
зон.динамика плотн.выс.снега', 'Литологический разрез',
'Физические и теплофизические']
```

Fig. 4. Updated sheet names

Now we can proceed to processing the Excel sheets themselves.

The first sheet is titled "Soil Temperature", which contains information about the measurement date and depth, as well as the measurement values themselves (Fig. 5).



	Температура грунтов, оС	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10	Unnamed: 11	Ur
0	NaN	Глубина, м	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	Дата	0	п/п	0.15	0.3	0.5	1.0	1.25	1.5	1.75	2.0	3.00	
2	1985-06-01 00:00:00	NaN	NaN	NaN	NaN	NaN	-2.1	NaN	NaN	NaN	-2.8	-3.00	
3	1985-06-05 00:00:00	NaN	NaN	NaN	NaN	NaN	-2.2	NaN	NaN	NaN	-2.8	-2.90	
4	1985-06-10 00:00:00	NaN	NaN	NaN	NaN	NaN	-1.5	NaN	NaN	NaN	-2.3	-2.70	
...	...	...	...	...	...	...	...	...	...	...	...	...	

Fig. 5. Sheet "Soil Temperature"

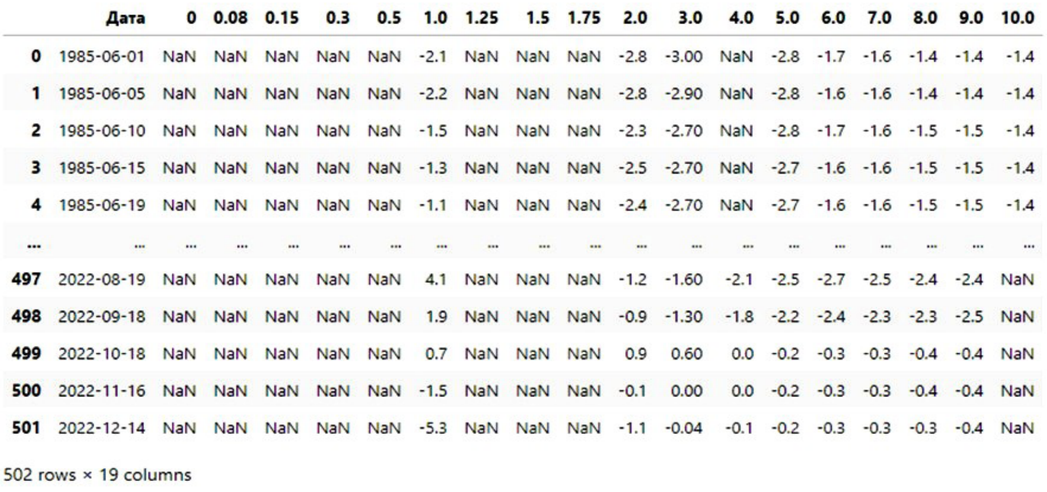
For successful data processing, it is necessary to take into account the specifics of how the data are presented. In particular, in some entries of the column containing the depth of the temperature sensors, the designation "п/п" appears. This is not a missing value (NaN – Not a Number), but a special marker indicating that the sensor is located directly under the snow cover. During data preparation, the "п/п" values should be replaced with the corresponding numerical depth values according to the characteristics of the sites.

It should also be noted that the table contains missing values denoted as NaN. In this context, NaN means that a specific parameter was not measured at a given time and depth. Additionally, the column containing dates needs to be converted to a unified format, and the data should be checked for duplicate records, which should be removed if found. A universal code was written to perform all of the above tasks, applicable to all the "Soil Temperature" sheets in the dataset as a whole.

```
   df1 = dfs['Soil Temperature']
   # Dictionary for replacing «п/п» values for specific sites
   dict_pp_values = {'Site 9_Chabyda': 0.08,
                     'Site 10_Chabyda': 0.03,
                     'Site 11_Chabyda': 0.02}
   if state_name in dict_pp_values:
       df1.replace({'п/п': dict_pp_values[state_name]}, inplace=True)
   dict_pkr_values = {'Site 8_Chabyda': 0.06}
   if state_name in dict_pkr_values:
       df1.replace({'под пкр': dict_pkr_values[state_name]}, inplace=True)
   date_indices = df1[df1.apply(lambda row: row.astype(str).str.contains('Date').any(),
axis=1)].index[1:]
   df1 = df1.drop(date_indices)
   df1 = df1.iloc[1:]
   df1 = df1.reset_index(drop=True)
   df1.columns = df1.iloc[0]
   df1 = df1.iloc[1:]
   df1 = df1.reset_index(drop=True)
   df1 = df1.rename(columns={'Date ': 'Date'})
   df1['Date'] = pd.to_datetime(df1['Date'])
```

```
for column in df1.columns:
    if column != 'Date':
        df1[column] = df1[column].astype(float)
def drop_duplicate(df, sort_columns):
    """ Remove duplicate rows and sort the dataframe. """
    duplicate_df = df[df.duplicated()]
    if not duplicate_df.empty:
        print(f'Duplicate rows:\n{duplicate_df}\n')
        df = df.drop_duplicates(keep='last')
    else:
        print('No duplicate rows found')
    df = df.sort_values(by=sort_columns)
    df = df.reset_index(drop=True)
    return df
df1 = drop_duplicate(df1, 'Date')
df1
```

After executing the code, the variable df1 contains the processed sheet «Soil Temperature,» which is easy to read and understand (Fig. 6).

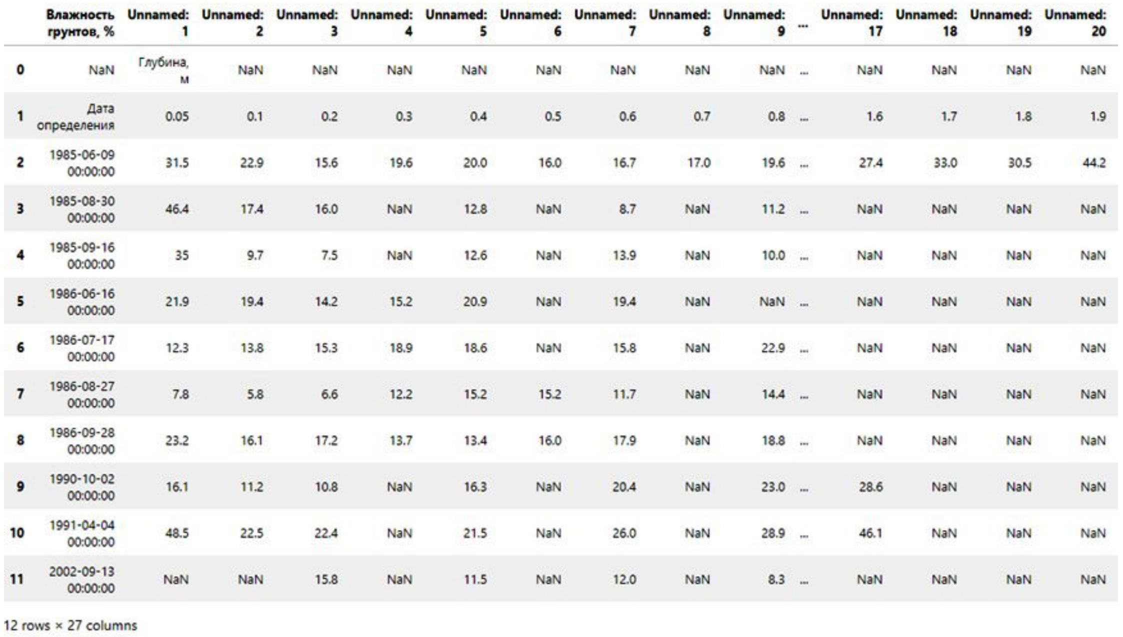


	Дата	0	0.08	0.15	0.3	0.5	1.0	1.25	1.5	1.75	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0
0	1985-06-01	NaN	NaN	NaN	NaN	NaN	-2.1	NaN	NaN	NaN	-2.8	-3.00	NaN	-2.8	-1.7	-1.6	-1.4	-1.4	-1.4
1	1985-06-05	NaN	NaN	NaN	NaN	NaN	-2.2	NaN	NaN	NaN	-2.8	-2.90	NaN	-2.8	-1.6	-1.6	-1.4	-1.4	-1.4
2	1985-06-10	NaN	NaN	NaN	NaN	NaN	-1.5	NaN	NaN	NaN	-2.3	-2.70	NaN	-2.8	-1.7	-1.6	-1.5	-1.5	-1.4
3	1985-06-15	NaN	NaN	NaN	NaN	NaN	-1.3	NaN	NaN	NaN	-2.5	-2.70	NaN	-2.7	-1.6	-1.6	-1.5	-1.5	-1.4
4	1985-06-19	NaN	NaN	NaN	NaN	NaN	-1.1	NaN	NaN	NaN	-2.4	-2.70	NaN	-2.7	-1.6	-1.6	-1.5	-1.5	-1.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
497	2022-08-19	NaN	NaN	NaN	NaN	NaN	4.1	NaN	NaN	NaN	-1.2	-1.60	-2.1	-2.5	-2.7	-2.5	-2.4	-2.4	NaN
498	2022-09-18	NaN	NaN	NaN	NaN	NaN	1.9	NaN	NaN	NaN	-0.9	-1.30	-1.8	-2.2	-2.4	-2.3	-2.3	-2.5	NaN
499	2022-10-18	NaN	NaN	NaN	NaN	NaN	0.7	NaN	NaN	NaN	0.9	0.60	0.0	-0.2	-0.3	-0.3	-0.4	-0.4	NaN
500	2022-11-16	NaN	NaN	NaN	NaN	NaN	-1.5	NaN	NaN	NaN	-0.1	0.00	0.0	-0.2	-0.3	-0.3	-0.4	-0.4	NaN
501	2022-12-14	NaN	NaN	NaN	NaN	NaN	-5.3	NaN	NaN	NaN	-1.1	-0.04	-0.1	-0.2	-0.3	-0.3	-0.3	-0.4	NaN

502 rows × 19 columns

Fig. 6. Processed data of the "Soil Temperature" sheet

The second sheet, "Soil Moisture", contains similar information to the "Soil Temperature" sheet. However, more transformations are required, such as removing empty rows, renaming the column "Date of determination" to "Date" and converting it to date format, replacing letter values like "Ice" with numeric values, searching for duplicates, and removing them (Fig. 7).



	Влажность грунтов, %	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 17	Unnamed: 18	Unnamed: 19	Unnamed: 20
0	NaN	Глубина, м	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
1	Дата определения	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	...	1.6	1.7	1.8	1.9
2	1985-06-09 00:00:00	31.5	22.9	15.6	19.6	20.0	16.0	16.7	17.0	19.6	...	27.4	33.0	30.5	44.2
3	1985-08-30 00:00:00	46.4	17.4	16.0	NaN	12.8	NaN	8.7	NaN	11.2	...	NaN	NaN	NaN	NaN
4	1985-09-16 00:00:00	35	9.7	7.5	NaN	12.6	NaN	13.9	NaN	10.0	...	NaN	NaN	NaN	NaN
5	1986-06-16 00:00:00	21.9	19.4	14.2	15.2	20.9	NaN	19.4	NaN	NaN	...	NaN	NaN	NaN	NaN
6	1986-07-17 00:00:00	12.3	13.8	15.3	18.9	18.6	NaN	15.8	NaN	22.9	...	NaN	NaN	NaN	NaN
7	1986-08-27 00:00:00	7.8	5.8	6.6	12.2	15.2	15.2	11.7	NaN	14.4	...	NaN	NaN	NaN	NaN
8	1986-09-28 00:00:00	23.2	16.1	17.2	13.7	13.4	16.0	17.9	NaN	18.8	...	NaN	NaN	NaN	NaN
9	1990-10-02 00:00:00	16.1	11.2	10.8	NaN	16.3	NaN	20.4	NaN	23.0	...	28.6	NaN	NaN	NaN
10	1991-04-04 00:00:00	48.5	22.5	22.4	NaN	21.5	NaN	26.0	NaN	28.9	...	46.1	NaN	NaN	NaN
11	2002-09-13 00:00:00	NaN	NaN	15.8	NaN	11.5	NaN	12.0	NaN	8.3	...	NaN	NaN	NaN	NaN

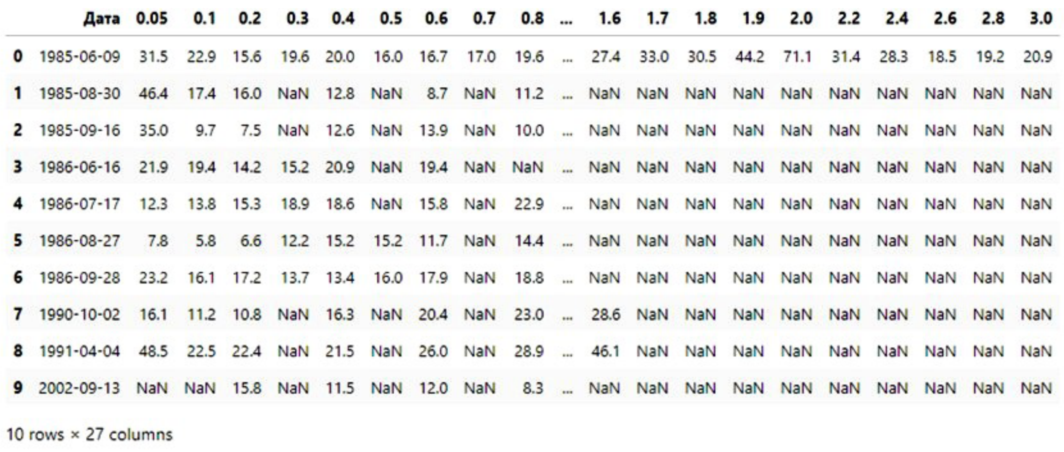
12 rows × 27 columns

Fig. 7. The "Soil Moisture" sheet

To accomplish this, code was written that performs all these transformations for the “Soil Moisture” sheets:

```
df2 = dfs['Влажность грунтов']
df2 = df2.iloc[1:]
df2 = df2.reset_index(drop=True)
df2.columns = df2.iloc[0]
df2 = df2.iloc[1:]
df2 = df2.reset_index(drop=True)
df2 = df2.rename(columns={'Дата определения': 'Дата'})
df2 = df2.dropna(how='all')
df2 = df2.dropna(thresh=2)
if state_name == 'Площадка Лес_Туймаада':
df2['Дата'] = df2['Дата'].astype(str)
df2['Скважена'] = df2['Дата'].apply(lambda x: 1 if '*' in x else 2)
df2['Дата'] = df2['Дата'].astype(str).str.replace(' *', '')
df2['Дата'] = df2['Дата'].astype(str).str.replace('*', '')
df2['Дата'] = df2['Дата'].apply(lambda x: x + ' 00:00:00' if len(x) == 10 else x)
for index, row in df2.iterrows():
try:
df2.at[index, 'Дата'] = pd.to_datetime(row['Дата'], format='%d.%m.%Y %H:%M:%S').
strftime('%Y-%m-%d %H:%M:%S')
except ValueError:
pass
if state_name == 'Площадка Луг_Туймаада':
df2['Дата'] = df2['Дата'].astype(str)
df2 = df2[~df2['Дата'].str.contains('Дата')]
def fill_skvazhena(row):
if '*' in row['Дата']:
return '2'
elif row['Дата'].count(' ') == 0 and not row['Дата'].isdigit():
return '3'
else:
return '1'
df2['Скважена'] = df2.apply(fill_skvazhena, axis=1)
df2['Дата'] = df2['Дата'].astype(str).str.replace(' *', '')
df2['Дата'] = df2['Дата'].astype(str).str.replace('*', '')
df2['Дата'] = df2['Дата'].apply(lambda x: x + ' 00:00:00' if len(x) == 10 else x)
for index, row in df2.iterrows():
try:
df2.at[index, 'Дата'] = pd.to_datetime(row['Дата'], format='%d.%m.%Y %H:%M:%S').
strftime('%Y-%m-%d %H:%M:%S')
except ValueError:
pass
months_dict = {'Январь': 'January',
               'Февраль': 'February',
               'Март': 'March',
               'Апрель': 'April',
               'Май': 'May',
               'Июнь': 'June',
               'Июль': 'July',
               'Август': 'August',
               'Сентябрь': 'September',
               'Октябрь': 'October',
               'Ноябрь': 'November',
               'Декабрь': 'December'}
months_dict_number = {'January': '01',
                      'February': '02',
                      'March': '03',
                      'April': '04',
                      'May': '05',
                      'June': '06',
                      'July': '07',
                      'August': '08',
                      'September': '09',
                      'October': '10',
                      'November': '11',
                      'December': '12'}
df2['Год'] = np.where(df2['Скважена'] == '3', '1979', np.nan)
df2['Дата'].replace(months_dict, inplace=True, regex=True)
df2['Дата'].replace(months_dict_number, inplace=True, regex=True)
df2.loc[df2['Скважена'] == '3', 'Дата'] = df2['Год'] + '-' + df2['Дата'] + '-01 00:00:00'
df2 = df2.drop(['Год'], axis=1)
df2 = df2.sort_values(by='Дата')
df2 = df2.reset_index(drop=True)
df2['Дата'] = pd.to_datetime(df2['Дата'])
# Замена значения «Лед» на «-1.0»
df2.replace('Лед', -1.0, inplace=True)
for column in df2.columns:
if column != 'Дата':
df2[column] = df2[column].astype(float)
if 'Скважена' in df2.columns:
df2['Скважена'] = df2['Скважена'].astype(int)
df2 = drop_duplicate(df2, 'Дата').
```

After running the code, the variable df2 contains the processed sheet «Soil Moisture,» presented in a human-readable format (Fig. 8).



	Дата	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	...	1.6	1.7	1.8	1.9	2.0	2.2	2.4	2.6	2.8	3.0
0	1985-06-09	31.5	22.9	15.6	19.6	20.0	16.0	16.7	17.0	19.6	...	27.4	33.0	30.5	44.2	71.1	31.4	28.3	18.5	19.2	20.9
1	1985-08-30	46.4	17.4	16.0	NaN	12.8	NaN	8.7	NaN	11.2	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1985-09-16	35.0	9.7	7.5	NaN	12.6	NaN	13.9	NaN	10.0	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1986-06-16	21.9	19.4	14.2	15.2	20.9	NaN	19.4	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1986-07-17	12.3	13.8	15.3	18.9	18.6	NaN	15.8	NaN	22.9	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	1986-08-27	7.8	5.8	6.6	12.2	15.2	15.2	11.7	NaN	14.4	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	1986-09-28	23.2	16.1	17.2	13.7	13.4	16.0	17.9	NaN	18.8	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7	1990-10-02	16.1	11.2	10.8	NaN	16.3	NaN	20.4	NaN	23.0	...	28.6	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
8	1991-04-04	48.5	22.5	22.4	NaN	21.5	NaN	26.0	NaN	28.9	...	46.1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9	2002-09-13	NaN	NaN	15.8	NaN	11.5	NaN	12.0	NaN	8.3	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

10 rows × 27 columns

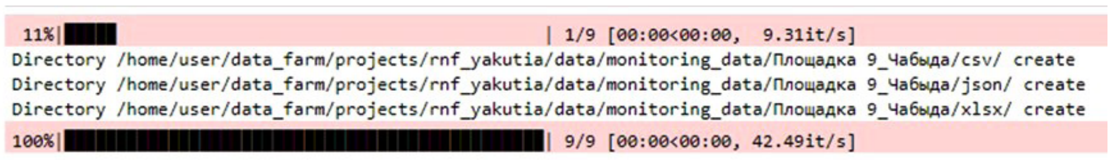
Fig. 8. Processed data of the "Soil Moisture" sheet

In a similar way, the remaining seven sheets for "Site 9_Chabyda" and the other fifteen sites, each containing nine sheets of data, were processed. To save the results of this work, a function was written that creates a directory with subfolders to store the processed data by site and corresponding sheets (Fig. 9) in the following formats:

• CSV (Comma-Separated Values) — one of the most common formats for representing tabular data, a simple and universal format used for storing data as text files based on comma-separated values;

• JSON (JavaScript Object Notation) — a standard text format for storing and transmitting structured data;

• XLSX — the Microsoft Excel file format used for storing spreadsheets.



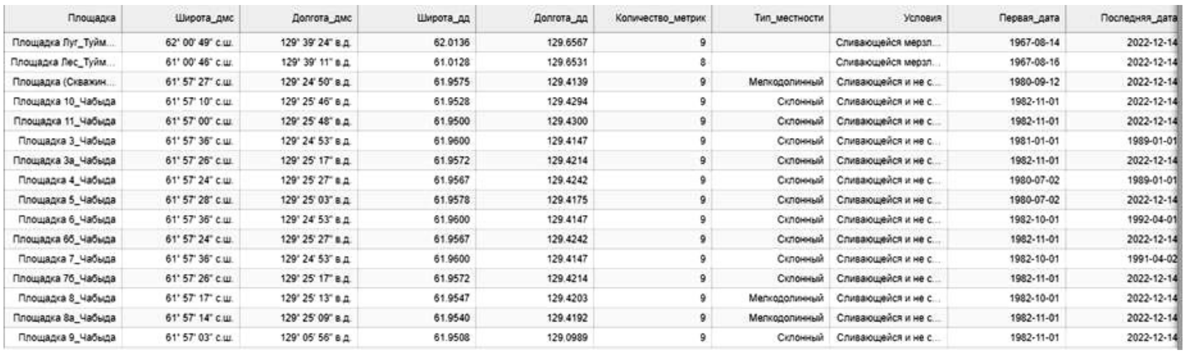
```
 11%|████                                    | 1/9 [00:00<00:00,  9.31it/s]
Directory /home/user/data_farm/projects/rnf_yakutia/data/monitoring_data/Площадка 9_Чабыда/csv/ create
Directory /home/user/data_farm/projects/rnf_yakutia/data/monitoring_data/Площадка 9_Чабыда/json/ create
Directory /home/user/data_farm/projects/rnf_yakutia/data/monitoring_data/Площадка 9_Чабыда/xlsx/ create
100%|████████████████████████████████████████| 9/9 [00:00<00:00, 42.49it/s]
```

Fig. 9. Saving processed data in CSV, JSON, and XLSX formats

A script was also written to create an XLSX file containing summary information about the sites:

```
xlsx_files = [f for f in os.listdir(f'{folder_path}/xlsx/') if f.endswith(«.xlsx»)]
xlsx_dfs = {}
# Reading data from each file and adding it to a dictionary
for file in xlsx_files:
file_path = os.path.join(f'{folder_path}/xlsx/', file)
xl = pd.ExcelFile(file_path)
for sheet_name in xl.sheet_names:
df = xl.parse(sheet_name)
if sheet_name not in xlsx_dfs:
xlsx_dfs[sheet_name] = []
xlsx_dfs[sheet_name].append(df)
with  pd.ExcelWriter(f'{folder_path}/xlsx/{state_name}.xlsx',  engine='xlsxwriter')  as
writer:
for sheet_name, dataframes in xlsx_dfs.items():
for i, df in enumerate(dataframes):
df.to_excel(writer, sheet_name=sheet_name, index=False).
```

After running it, a file with summary information for each of the sites is generated (Fig. 10).



Площадка	Широта_дмс	Долгота_дмс	Широта_дд	Долгота_дд	Количество_метрик	Тип_местности	Условия	Первая_дата	Последняя_дата
Площадка Луг_Туйм...	62° 00' 49" с.ш.	129° 39' 24" в.д.	62.0136	129.6567	9		Сливающейся мерзл...	1967-08-14	2022-12-14
Площадка Лес_Туйм...	61° 00' 46" с.ш.	129° 39' 11" в.д.	61.0128	129.6531	8		Сливающейся мерзл...	1967-08-16	2022-12-14
Площадка (Скважин...	61° 57' 27" с.ш.	129° 24' 50" в.д.	61.9575	129.4139	9	Мелкодолинный	Сливающейся и не с...	1980-09-12	2022-12-14
Площадка 10_Чабыда	61° 57' 10" с.ш.	129° 25' 46" в.д.	61.9528	129.4294	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 11_Чабыда	61° 57' 00" с.ш.	129° 25' 48" в.д.	61.9500	129.4300	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 3_Чабыда	61° 57' 36" с.ш.	129° 24' 53" в.д.	61.9600	129.4147	9	Склонный	Сливающейся и не с...	1981-01-01	1989-01-01
Площадка 3а_Чабыда	61° 57' 26" с.ш.	129° 25' 17" в.д.	61.9572	129.4214	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 4_Чабыда	61° 57' 24" с.ш.	129° 25' 27" в.д.	61.9567	129.4242	9	Склонный	Сливающейся и не с...	1980-07-02	1989-01-01
Площадка 5_Чабыда	61° 57' 28" с.ш.	129° 25' 03" в.д.	61.9578	129.4175	9	Склонный	Сливающейся и не с...	1980-07-02	2022-12-14
Площадка 6_Чабыда	61° 57' 36" с.ш.	129° 24' 53" в.д.	61.9600	129.4147	9	Склонный	Сливающейся и не с...	1982-10-01	1992-04-01
Площадка 6б_Чабыда	61° 57' 24" с.ш.	129° 25' 27" в.д.	61.9567	129.4242	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 7_Чабыда	61° 57' 36" с.ш.	129° 24' 53" в.д.	61.9600	129.4147	9	Склонный	Сливающейся и не с...	1982-10-01	1991-04-02
Площадка 7б_Чабыда	61° 57' 26" с.ш.	129° 25' 17" в.д.	61.9572	129.4214	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 8_Чабыда	61° 57' 17" с.ш.	129° 25' 13" в.д.	61.9547	129.4203	9	Мелкодолинный	Сливающейся и не с...	1982-10-01	2022-12-14
Площадка 8а_Чабыда	61° 57' 14" с.ш.	129° 25' 09" в.д.	61.9540	129.4192	9	Мелкодолинный	Сливающейся и не с...	1982-11-01	2022-12-14
Площадка 9_Чабыда	61° 57' 03" с.ш.	129° 05' 56" в.д.	61.9508	129.0989	9	Склонный	Сливающейся и не с...	1982-11-01	2022-12-14

Fig. 10. File with summary information on the sites

Results. The processing and systematization of data made it possible to unify disparate measurements into a consistent format, ensuring their correct use in subsequent calculations. As a result, a unique dataset was obtained that has no analogs, since it is based on direct empirical observations and covers the specific conditions of the studied area.

Data analysis using Python is a highly demanded skill in the modern world. This language is characterized by relative ease of learning, making it accessible for beginners. The work considered the main stages of data processing using Python libraries: preprocessing and data preparation. Using the example of the dataset "Site 9_Chabyda," key preprocessing techniques were demonstrated, such as detection and handling of missing values, data transformation, optimization, as well as removal of insignificant attributes.

Discussion and Conclusion. The results demonstrate that applying a systematic approach to data processing significantly improves the quality and reliability of the analysis. Addressing missing data and normalizing the dataset improves the reliability of the dataset. The final processed data is provided in convenient formats for further use in modelling, including CSV, JSON, and XLSX formats. This highlights the versatility of Python in solving data processing tasks.

Future research plans include expanding the study by incorporating data analysis methods such as clustering, machine learning model development, and result visualization [12, 14, 19]. These methods will not only help uncover hidden patterns within the data but also enable the prediction of the behavior of natural-technical systems under changing external conditions. Special attention will be given to modelling the impact of climatic factors and anthropogenic loads on natural-technical systems, providing a more comprehensive understanding of processes occurring in the conditions of the Far North.

References

1. Poruchikov M. A. *Data analysis*. Samara: Samara University Press; 2016. 88 p. (In Russ.)
2. *Introduction to Data Analysis*. URL: https://mipt-stats.gitlab.io/courses/ad_fivt/titanik.html (accessed: 08.08.2024)
3. Leskovec Yu. *Analysis of Large Datasets.* Moscow: DMK; 2016. 498 p. (In Russ.)

4. McKinney W. *Python and Data Analysis*. Moscow: DMK; 2015. 482 p. (In Russ.)
5. Makshanov A.V. *Data Mining Technologies*. Saint Petersburg: Lan; 2018. 212 p. (In Russ.)
6. Mirkin B.G. *Introduction to Data Analysis*. Lyubertsy: Yurayt; 2016. 174 p. (In Russ.)
7. Rafalovich V. *Data Mining, or Data Analysis for the Busy: Practical Course*. Moscow: SmartBook; 2018. 352 p. (In Russ.)
8. Chashkin Yu.R. *Mathematical Statistics. Data Analysis and Processing*. Rostov-on-Don: Feniks; 2017. 236 p. (In Russ.)
9. Gron A. *Machine Learning and Data Analysis with Python*. Saint Petersburg: Piter; 2019. 460 p. (In Russ.)
10. Vorontsov K.V. *Machine Learning and Data Mining*. Moscow: MIPT Publishing; 2018. 328 p. (In Russ.)
11. Algozina E.R. *Data Processing and Analysis with Python and Pandas: Beginner's Guide*. Moscow: DMK; 2020. 312 p. (In Russ.)
12. Troyan I.V. *Data Analysis and Machine Learning in Python*. Saint Petersburg: BHV-Peterburg; 2019. 320 p. (In Russ.)
13. McKinney W. *Python Programming and Data Analysi*s. Saint Petersburg: Piter; 2020. 504 p. (In Russ.)
14. Mueller A.C., Guido S. *Introduction to Machine Learning with Python*. Saint Petersburg: Piter; 2017. 420 p. (In Russ.)
15. Rozhkov R.S., Molchanova E.S., Fadeev M.K., Pimenov N.A. Managing Natural-Technical Systems through the Concept of Sustainable Development and Acceptable Risk. *Bulletin of Eurasian Science*. 2024;16(5):1–10. (In Russ.)
16. Dregulo N. Influence of Climatic Factors on the Operation of Natural-Technical Systems for Wastewater Treatment. URL: https://pureportal.spbu.ru/ru/publications/---------%28a9c91d8e-c5e0-43d9-98b5-8aac39cc4a61%29.html (accessed: 13.01.2025) (In Russ.)
17. Impact of Anthropogenic Factors on Thermal Pollution of the Urban Environment. URL: https://www.abok.ru/for_spec/articles.php?nid=7635 (accessed: 13.01.2025) (In Russ.)
18. Impact of Anthropogenic Factors on Agricultural Development. URL: https://biohim.com.ru/articles/page2 (accessed: 13.01.2025) (In Russ.)
19. Levashkin S.P., Ivanov K.N., Kushukov S.V. Data Farm: An Information System for Collecting, Storing, and Processing Unstructured Data from Heterogeneous Sources. *Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS)*. 2023;35(2):57–72. https://doi.org/10.15514/ISPRAS-2023-35(2)-5 (In Russ.)

About the Authors:

Sergey V. Kushukov, Engineer, Povolzhskiy State University of Telecommunications and Informatics, Research Laboratory "Artificial Intelligence" (77, Moskovskoe Shosse, Samara, 443090, Russian Federation), ORCID, SPIN-code, s.kushukov@psuti.ru

Konstantin N. Ivanov, Engineer, Povolzhskiy State University of Telecommunications and Informatics, Research Laboratory "Artificial Intelligence" (77, Moskovskoe Shosse, Samara, 443090, Russian Federation), ORCID, SPIN-code, k.ivanov@psuti.ru

Sergey P. Levashkin, Candidate of Physical and Mathematical Sciences, Associated Professor, Povolzhskiy State University of Telecommunications and Informatics, Head of Research Laboratory "Artificial Intelligence" (77, Moskovskoe Shosse, Samara, 443090, Russian Federation), ORCID, SPIN-code, ai_lab@psuti.ru

Mikhail V. Yakobovskiy, Corresponding Member of the Russian Academy of Sciences, Doctor of Physical and Mathematical Sciences, Director, Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences (IPM RAS), (4, Miusskaya pl., Moscow, 125047, Russian Federation), ORCID, SPIN-code, lira@imamod.ru

Contributions of the authors:
S.V. Kushukov: development, testing of existing code components; writing a draft of a manuscript.
K.N. Ivanov: formal analysis.
S.P. Levashkin: problem statement; formulation of research ideas; goals and objectives.
M.V. Yakobovskiy: review and editing of the manuscript.

***Conflict of Interest Statement:* the authors declare no conflict of interest.**

All authors have read and approved the final manuscript.

Об авторах:

Сергей Владимирович Кушуков, инженер Поволжского государственного университета телекоммуникаций и информатики, Научно-исследовательской лаборатории «Искусственный интеллект» (443090, Российская Федерация, г. Самара, Московское ш., 77), ORCID, SPIN-код, s.kushukov@psuti.ru

Константин Николаевич Иванов, инженер Поволжского государственного университета телекоммуникаций и информатики, Научно-исследовательской лаборатории «Искусственный интеллект» (443090, Российская Федерация, г. Самара, Московское ш., 77), ORCID, SPIN-код, k.ivanov@psuti.ru

Сергей Павлович Левашкин, кандидат физико-математических наук, доцент Поволжского государственного университета телекоммуникаций и информатики, заведующий Научно-исследовательской лаборатории «Искусственный интеллект» (443090, Российская Федерация, г. Самара, Московское ш., 77), ORCID, SPIN-код, ai_lab@psuti.ru

Михаил Владимирович Якобовский, член-корреспондент РАН, доктор физико-математических наук, директор Института прикладной математики имени М.В. Келдыша РАН (ИПМ РАН), (125047, Российская Федерация, г. Москва, Миусская пл., 4), ORCID, SPIN-код, lira@imamod.ru

Заявленный вклад авторов:
С.В. Кушуков: разработка, тестирование существующих компонентов кода; написание черновика рукописи.
К.Н. Иванов: формальный анализ.
С.П. Левашкин: постановка задачи; формулировка идей исследования, целей и задач.
М.В. Якобовский: рецензирование и редактирование рукописи.

Конфликт интересов: ***авторы заявляют об отсутствии конфликта интересов.***

Все авторы прочитали и одобрили окончательный вариант рукописи.

Received / Поступила в редакцию 15.05.2025
Revised / Поступила после рецензирования 02.06.2025
Accepted / Принята к публикации 18.06.2025