ПРИЛОЖЕНИЕ Б

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ (PYTHON)

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Библиотека открытого доступа DataReconcile для согласования неточных индустриальных данных за счет учета взаимосвязей между ними для нужд промышленных предприятий Индустрии 4.0 и целей Национальной технологической инициативы на языке Python

2025 год

Б.1 НАЗНАЧЕНИЕ БИБЛИОТЕКИ

Библиотека предназначена для решения задач **согласования неточных данных** (Data Reconcile) в условиях, когда входные данные содержат ошибки, неточности или противоречия. Она позволяет корректировать данные таким образом, чтобы они удовлетворяли заданным ограничениям, представленным в виде систем линейных и нелинейных уравнений и неравенств.

Библиотека помогает устранить противоречия в данных, минимизируя отклонения от исходных значений при соблюдении всех заданных ограничений. Это особенно полезно в задачах, где данные поступают из разных источников и могут содержать ошибки измерений или несоответствия.

Применение в различных областях:

Промышленность: согласование данных в системах управления технологическими процессами.

Экономика и финансы: обработка данных в моделях прогнозирования и анализа.

Наука и инженерия: решение задач оптимизации и калибровки моделей.

Энергетика: балансировка данных в энергосистемах.

Гибкость и масштабируемость:

Библиотека поддерживает работу с большими объемами данных и может быть интегрирована в различные программные системы.

Она предоставляет инструменты для настройки и адаптации под конкретные задачи пользователя.

Примеры задач, где библиотека может быть полезна:

Балансировка данных в химических процессах (например, согласование материальных и энергетических балансов).

Корректировка данных в системах мониторинга (например, устранение противоречий в показаниях датчиков).

Оптимизация данных в моделях прогнозирования (например, согласование исторических данных с прогнозными значениями).

Библиотека является мощным инструментом для анализа, корректировки и оптимизации данных в условиях наличия ограничений, что делает её незаменимой в задачах, требующих высокой точности и согласованности данных.

Б.2 ОСНОВНЫЕ ВОЗМОЖНОСТИ

Поддержка стационарных и динамических задач:

Стационарные задачи: работа с данными, которые не изменяются во времени (например, балансовые уравнения в статических системах).

Динамические задачи: обработка данных, которые изменяются во времени (например, временные ряды или процессы, описываемые дифференциальными уравнениями).

Работа с ограничениями:

Библиотека поддерживает ограничения в виде:

- **Линейных уравнений и неравенств** (например, $\mathbf{Ax} = \mathbf{b}$ или $\mathbf{Ax} \leq \mathbf{b}$).
- **Нелинейных уравнений и неравенств** (например, $\mathbf{F(x)} = \mathbf{0}$ или $\mathbf{g(x)} \leq \mathbf{0}$).

Библиотека разрешает использование дифференциальных и интегро-дифференциальных уравнений в качестве ограничений, но в форме разностных уравнений по выбранной пользователем вычислительной схеме.

Это позволяет учитывать физические, экономические или другие законы, которым должны соответствовать данные.

Минимизация отклонений:

Библиотека минимизирует отклонения скорректированных данных от исходных, используя различные метрики.

Б.3 СИСТЕМНЫЕ ТРЕБОВАНИЯ

Минимальные системные требования

Минимальные требования позволяют запускать библиотеку и выполнять базовые задачи на компьютерах с ограниченными ресурсами. Однако производительность может быть низкой при работе с большими объемами данных или сложными задачами.

- **Операционная система**:
  - Windows 7/8/10/11 (64-битная) или новее.
  - macOS 10.13 (High Sierra) или новее.
  - Linux: Ubuntu 18.04 LTS или совместимые дистрибутивы.
- **Процессор (CPU)**:
  - Минимум: 2-ядерный процессор с тактовой частотой 1.8 ГГц (например, Intel Core i3 или аналогичный AMD).
- **Оперативная память (RAM)**:
  - Минимум: 4 ГБ (для небольших задач).

- Рекомендуется: 8 ГБ для более комфортной работы.

- **Место на диске**:
  - Минимум: 500 МБ свободного места для установки Python и библиотек.
  - Дополнительное место для хранения данных и результатов.
- **Программное обеспечение**:
  - Python 3.7 или новее.
  - Установленные библиотеки:
    - numpy (для работы с массивами и матрицами).
    - scipy (для оптимизации и научных вычислений).
    - Дополнительные зависимости, указанные в документации библиотеки.

Рекомендуемые системные требования

Рекомендуемые требования обеспечивают высокую производительность при работе с большими объемами данных, сложными моделями и динамическими задачами.

- **Операционная система**:
  - Windows 10/11 (64-битная) или новее.
  - macOS 12 (Monterey) или новее.
  - Linux: Ubuntu 20.04 LTS или совместимые дистрибутивы.
- **Процессор (CPU)**:
  - Рекомендуется: 4-ядерный процессор с тактовой частотой 2.5 ГГц или выше (например, Intel Core i5/i7, AMD Ryzen 5/7).
- **Оперативная память (RAM)**:
  - Рекомендуется: 16 ГБ или больше (для работы с большими наборами данных и сложными задачами).
- **Место на диске**:
  - Рекомендуется: SSD с 1 ГБ свободного места для установки Python и библиотек.
  - Дополнительное место для хранения данных и результатов (в зависимости от объема данных).
- **Программное обеспечение**:
  - Python 3.9 или новее.
  - Установленные библиотеки:
    - numpy (последняя версия).

        - scipy (последняя версия).
        - Дополнительные библиотеки, такие как pandas (для работы с таблицами), matplotlib (для визуализации), joblib (для параллельных вычислений).
- **Графика**:
    - Дискретная видеокарта с 2 ГБ видеопамяти и поддержкой OpenGL 4.0 (например, NVIDIA GeForce GTX 1050 или выше).

Особые рекомендации

- **Для больших задач**:
    - Если вы работаете с очень большими наборами данных или сложными моделями, рекомендуется использовать серверные решения с многоядерными процессорами (например, 16+ ядер) и большим объемом оперативной памяти (32 ГБ и более).
    - Используйте библиотеки для параллельных вычислений, такие как joblib или dask.
- **Для динамических задач**:
    - Рекомендуется использовать SSD для быстрого чтения/записи данных, особенно при работе с временными рядами или большими объемами данных.
- **Для разработчиков**:
    - Убедитесь, что у вас установлены все необходимые пакеты и библиотеки, включая numpy, scipy и другие зависимые инструменты.
    - Используйте виртуальные окружения (venv или conda) для изоляции зависимостей.

Проверка совместимости

Перед установкой библиотеки убедитесь, что ваша система соответствует требованиям:

Проверьте версию Python и установленных библиотек (bash)

```
python --version
pip show numpy scipy
```

- Убедитесь, что ваш процессор и оперативная память соответствуют минимальным или рекомендуемым требованиям.
- Проверьте наличие свободного места на диске.

Установка библиотек

1. Установите Python с официального сайта: python.org.

Установите необходимые библиотеки с помощью pip (bash):

```
pip install numpy scipy
```

Подготовка файлов библиотеки

Здесь и далее в целях лаконичности используемая функция согласования данных обозначается условным именем *LibFunctionName*.

Создайте структуру папок:

Организуйте файлы библиотеки в отдельной папке. Например:

```
DataReconcile/
├── __init__.py
├── LibFunctionName1.py
├── LibFunctionName2.py
└── LibFunctionName3.py
└── …
```

- Файл __init__.py необходим, чтобы Python распознал папку как пакет. Он может быть пустым или содержать код инициализации.

Подключение библиотеки к проекту

Скопируйте папку библиотеки в проект:

Поместите папку `DataReconcile` в папку вашего проекта. Например:

```
MyProject/
├── main.py
└── DataReconcile/
   ├── __init__.py
   ├── LibFunctionName1.py
   ├── LibFunctionName2.py
   └── LibFunctionName3.py
   └── …
```

Импортируйте библиотеку в вашем проекте:

В файле main.py (или другом файле проекта) используйте импорт:

```
from DataReconcile import LibFunctionName
```

Б.4 ПОШАГОВАЯ ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ БИБЛИОТЕКИ DATARECONCILE (PYTHON)

1. Если ранее этого не было сделано, Скачайте и установите интерпретатор Python версии 3.0 или новее.

2. Скачайте (А) файлы библиотеки DataReconcile, либо клонируйте (Б) на вычислительную машину, где планируется сборка проекта, использующего функционал библиотеки.

2.А Как скачать содержимое репозитория DataReconcile.

– Перейдите на главную страницу репозитория по ссылке https://github.com/scientific-soft/DataReconcile.

– Кликните на кнопку «<>», расположенную над списком содержащихся в корневой директории файлов и папок библиотеки (рисунок Б.1).

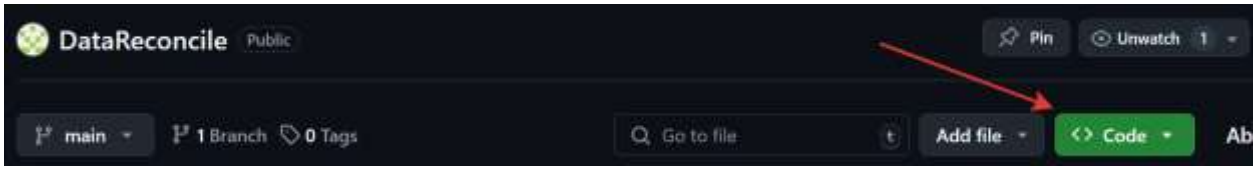

Рисунок Б.1 – К развертыванию библиотеки DataReconcile

– Во всплывающем меню выберите вкладку «Local» (1), после чего кликнете на строку «Download ZIP» (2) (рисунок Б.2).

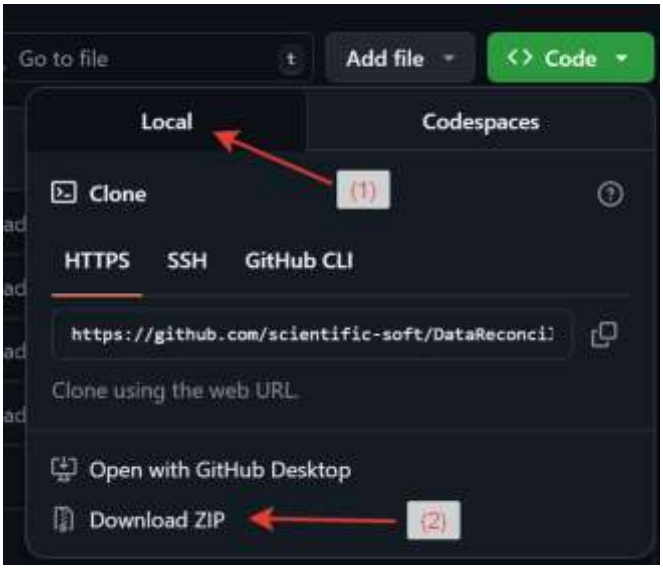

Рисунок Б.2 – К развертыванию библиотеки DataReconcile

– После выполнения данных операций содержимое библиотеки начнет скачиваться в директорию загрузки на локальном компьютере в виде файла с расширением «.zip».

– После того, как скачивание завершено, скаченный архивный файл «DataReconcile-main.zip» необходимо разархивировать программой архиватором, поддерживающей формат «.zip».

– Разархивация с использованием свободно распространяемого архиватора 7-Zip (рисунок Б.3).

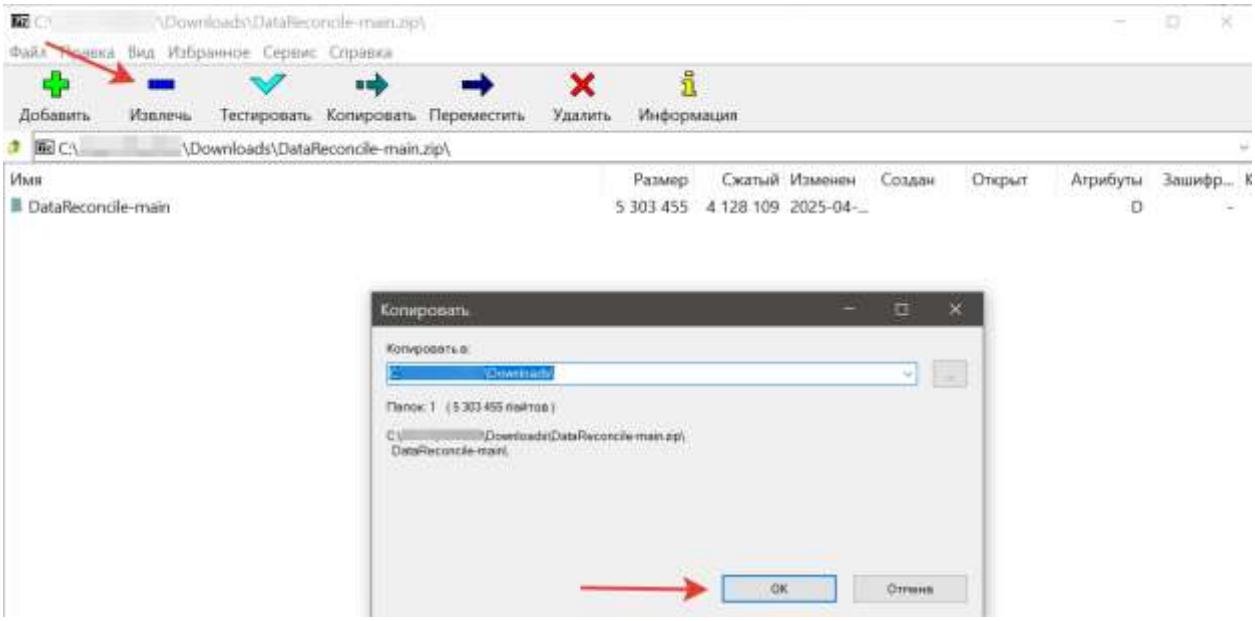

Рисунок Б.3 – К развертыванию библиотеки DataReconcile

– Результат скачивания файлов библиотеки – папка с именем DataReconcile-main, содержимое которой приведено на скриншоте ниже (рисунок Б.4).

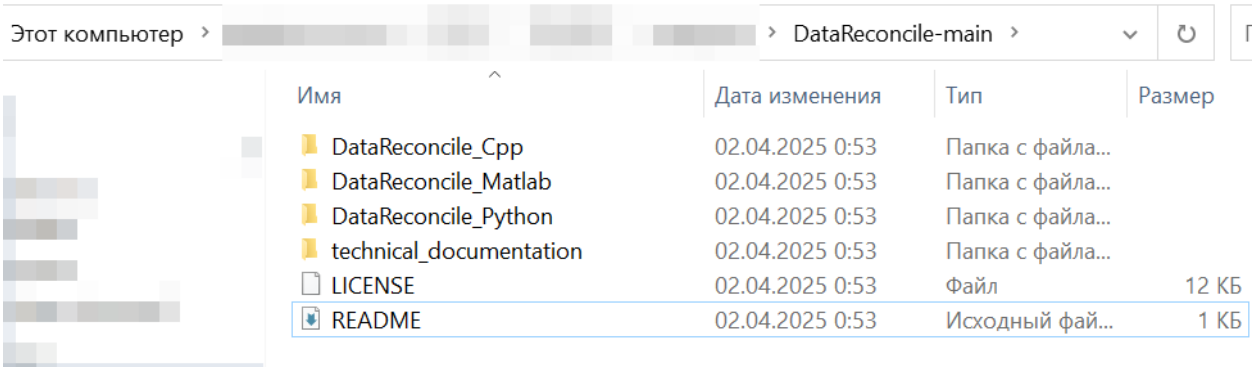

Рисунок Б.4 – К развертыванию библиотеки DataReconcile

2.Б Как клонировать содержимое репозитория DataReconcile.

– Перейдите на главную страницу репозитория по ссылке https://github.com/scientific-soft/DataReconcile

– Кликните на кнопку «<>», расположенную над списком содержащихся в корневой директории фалов и папок библиотеки (рисунок Б.5).

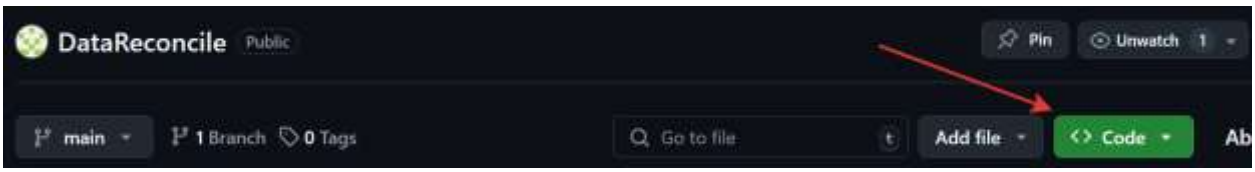

Рисунок Б.5 – К развертыванию библиотеки DataReconcile

– Скопируйте URL репозитория в буфер обмена.

1) Чтобы клонировать репозиторий с помощью HTTPS, нажмите кнопку копирования в разделе «HTTPS».

2) Чтобы клонировать репозиторий с помощью SSH ключа, включая сертификат, выданный центром сертификации SSH вашей организации, нажмите кнопку копирования в разделе SSH.

3) Чтобы клонировать репозиторий с помощью GitHub CLI, нажмите кнопку копирования в разделе GitHub CLI (рисунок Б.6).

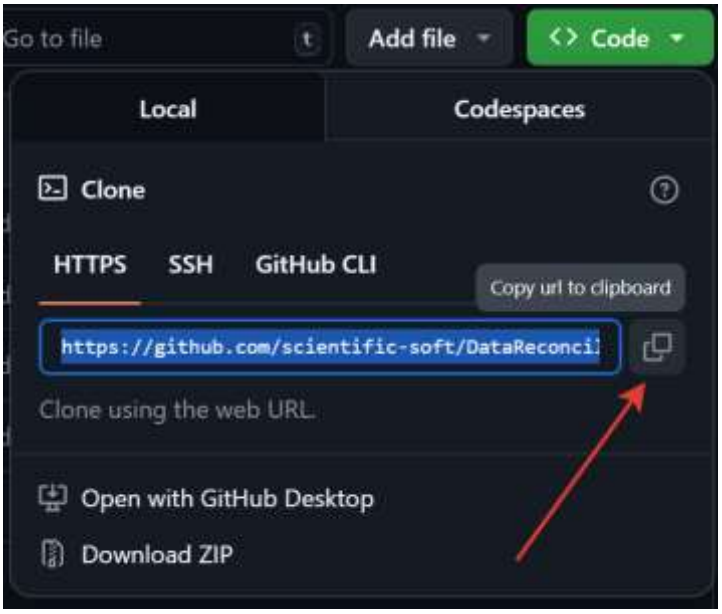

Рисунок Б.6 – К развертыванию библиотеки DataReconcile

– Откройте консоль Git Bash.

– Измените текущий рабочий каталог на место, куда вы хотите клонировать содержимое DataReconcile.

– Введите git clone, а затем вставьте URL-адрес, скопированный ранее в формате

git clone https://github.com/scientific-soft/DataReconcile.git

– Нажмите Enter, чтобы создать локальный клон библиотеки.

3. Установите следующие библиотеки языка Python: numpy, scipy. Для этой цели можно воспользоваться модулем pip.

4. Далее выполните одну из двух операций:

4.1 Разместите необходимые вам модули библиотеки DataReconcile из папки DataReconcile_Python в директории вашего исполняемого файла .py.

Пример добавления функции из библиотеки, именуемой условным именем my_function и хранящейся в файле с тем же именем my_function приведен ниже:

```
from my_function import *
```

4.2 В начало исполняемого скрипта .py, в которым вы желаете воспользоваться функцией/ями из библиотеки допишите путь к директории, где сохранены/куда клонированы модули из папки DataReconcile_Python.

Пример указания пути и добавления функции из библиотеки, именуемой условным именем my_function и хранящейся в файле с тем же именем my_function приведен ниже:

```
import sys
sys.path.append('/path/to/directory/containing/my_function')
from my_function import *
```

5. Все доступные функции по названию совпадают с названием файла расширением «.py». Если возникает ошибка вызова функции из библиотеки, проверьте, действительно в директории по указанному пути присутствует файл с названием функции.

Б.5 ОСНОВНЫЕ ФУНКЦИИ И API

Б.5.1 Описание ключевых модулей

Решаемая в рамках программной библиотеки задача относится к задачам повышения точности результатов совместных измерений, выполняемых на сложном объекте измерений, для которого предоставлена или получена в ходе эмпирических исследований математическая модель (возможно, не вполне точная). Снижение неопределенности получаемых результатов достигается за счет учета функциональных зависимостей между измеряемыми величинами, формализованных в рамках упомянутой математической модели. В качестве последней могут выступать как различные уравнения (алгебраические – линейные и нелинейные, дифференциальные и интегро-дифференциальные), а также наборы ограничений, вытекающие из условий решаемой измерительной задачи и условий функционирования и эксплуатации наблюдаемого объекта измерений (системы неравенств – односторонних и двусторонних). Решение задачи производится также при разном объеме известной информации, ключевое место среди которой занимают сведения о распределении погрешностей выполняемых измерений – как известно из математической статистики, при разных законах распределения случайных погрешностей разные числовые оценки для одного и того же параметра распределений будут являться эффективными (то есть будут обеспечивать наименьший статистический разброс от выборки к выборке). Данное обстоятельство указывает на обязательную необходимость привлечения сведений о

распределении погрешностей (в том объеме, в котором она доступна). Решение задачи повышения точности (за счет согласования индустриальных данных) под перечисленными ограничениями представляет собой комплекс задач, решения которых имеет как общие черты, так и разнится в зависимости от типа ограничений. Эффективные вычислительные решения для разных значимых для измерительной практики ситуаций представляет собой основу настоящей библиотеки. На первом этапе основным типом рассматриваемых ограничений выступают математические модели, составленные из алгебраических уравнений (как линейных, так и нелинейных) при условии, что закон распределения случайных погрешностей отличается от нормального (возможно, несильно). Данный тип задачи должен включать в себя традиционные решения (в предположении гауссовости распределения погрешностей), так и новые, до сего момента не представленные в научной литературе по вопросу методы и подходы. Другим важным моментом является необходимость учета режима выполняемых измерений – статического (когда изменениями во времени значений измеряемых величин можно пренебречь без значимого снижения точности) или динамического (когда изменения сигналов измерительной информации во времени приводят к возникновению значимых составляющих погрешностей, отсутствие учета которых приводит к существенному снижению достоверности конечных результатов). Данная постановка задачи крайне важна для измерительной практики и предложена впервые в контексте задачи согласования индустриальных данных.

Рассматривая задачу согласования индустриальных данных целостно, следует отметить, что задача так или иначе сводится к условной оптимизационной задаче некоторой размерности и с тем или иным набором условий. Данное обстоятельство позволяет построить эффективную модель решения задачи согласования данных, выбрав для того или иного набора ограничений наиболее подходящий алгоритм оптимизации. Программная библиотека, реализующей разные методы согласования неточных данных друг с другом, также содержит в себе ряд автоматизированных инструментов.

На рисунке Б.7 ниже представлена визуализация структуры библиотеки, а также входные и выходные данные.

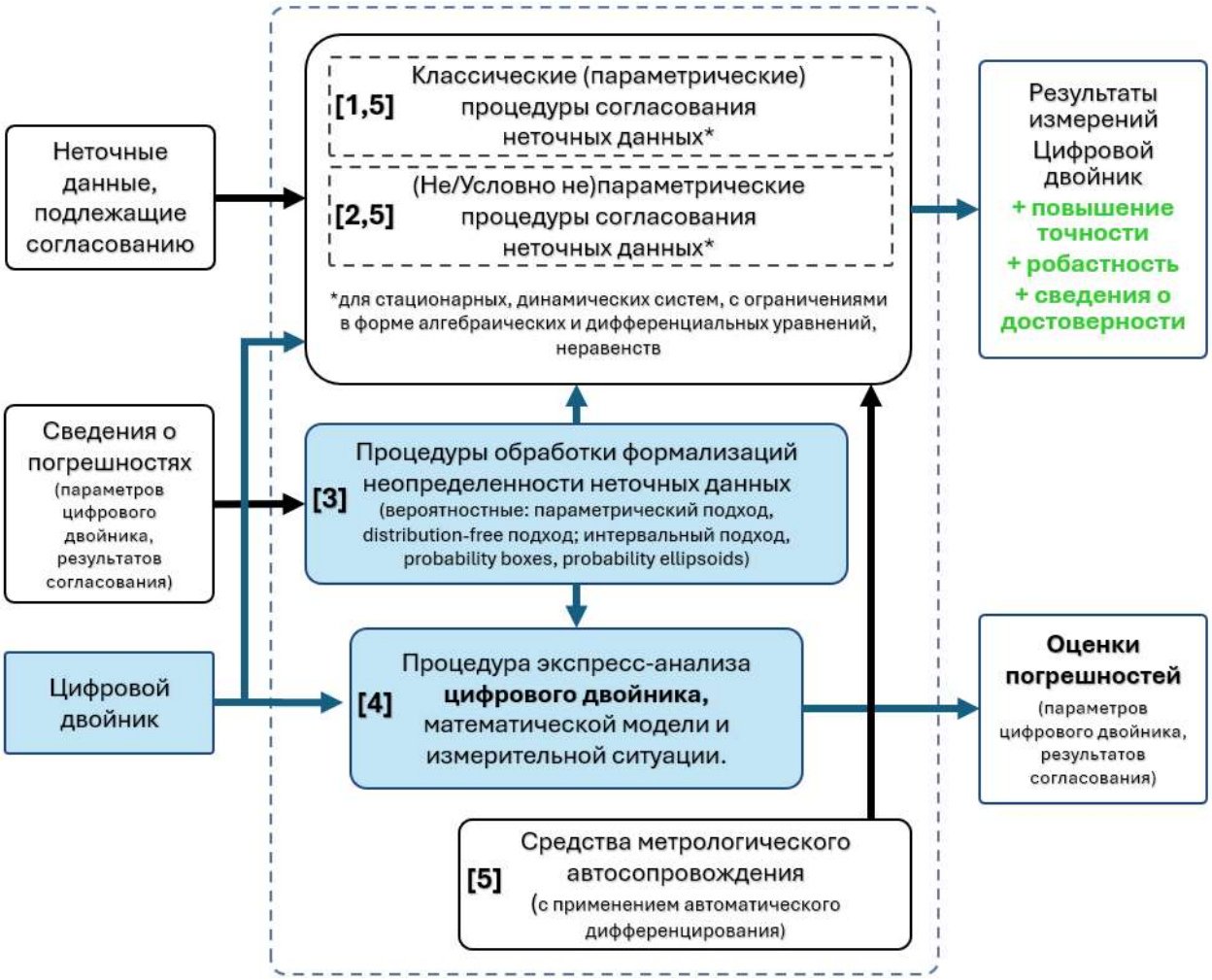

Рисунок Б.7 – Структура библиотеки DataReconcile

В соответствии с приведенной на рисунке нумерацией, библиотека содержит:

1) программные средства, реализующие традиционные методы согласования индустриальных данных, которые содержатся и применяются в зарубежных коммерческих продуктах,

2) программные средства, реализующие непараметрические и условно непараметрические методы согласования индустриальных данных, позволяющие учитывать в рамках согласования существенные или малые отклонения действительного закона распределения погрешностей от нормального, применимые как в статическом режиме измерения, так и в динамическом,

3) программные представления и преобразования современных формализмов описания погрешности результатов измерений, таковых, чтобы результат преобразования/представления мог быть эффективно использован в качестве мера

неопределенности неточных индустриальных данных при их согласовании по известной модели.

4) программные предварительной экспресс-оценки потенциального уточнения, достигаемого при математической обработке результатов выполняемых измерений за счет выполнения их согласования на основе известной априорной информации,

5) программные средства для обеспечения метрологического автосопровождения производимых вычислений [2-4], поскольку без него нет возможности оценить точность конечных результатов согласования, а также – при необходимости – произвести согласование моментов остановки выполняемых итерационных процедур с точностью исходных данных (то есть результатов выполненных измерений).

Б.5.2 Доступные функции, их параметры и возвращаемые значения

Презентуемая библиотека оформлена в виде независимых ру-файлов, для работы с которыми требуется установленный интерпретатор Python версии 3.0 или новее, с установленными библиотеками numpy и Eigen. Ниже приведено описание процедур для не/полу/параметрического согласования данных и выполнения метрологического анализа потенциальных результатов согласования.

Для каждого параметра `msrd_data` функции каждой функции справедливо следующее: в случае согласования результатов однократного совестного измерения нескольких взаимосвязанных величин, согласуемые результаты нужно передавать в формате вектор-столбца формата numpy array; если согласованию подлежит массив многократных совместных измерений, согласуемые значения нужно передавать в виде матрицы `msrd_data`, где каждый столбец – набор однократного совместного измерения всех согласуемых величин (таким образом в *j*-ой строке предаваемой матрицы `msrd_data` располагаются все результаты измерений *j*-ой измеряемой физической величины).

Презентуемая библиотека оформлена в виде независимых ру-файлов, для работы с которым требуется интерпретатор языка программирования Python версии 3.0 или новее, включающий библиотеки numpy и scipy. Ниже приведено описание ключевых процедур для не/полу/параметрического согласования данных и выполнения метрологического анализа потенциальных результатов согласования.

Для каждого параметра `msrd_data` функции каждой функции справедливо следующее: в случае согласования результатов однократного совестного измерения

нескольких взаимосвязанных величин, согласуемые результаты нужно передавать в формате вектор-столбца; если согласованию подлежит массив многократных совместных измерений, согласуемые значения нужно передавать в виде матрицы `msrd_data`, где каждый столбец – набор однократного совместного измерения всех согласуемых величин (таким образом в *j*-ой строке предаваемой матрицы `msrd_data` располагаются все результаты измерений *j*-ой измеряемой физической величины).

Процедура оценки степени потенциального уточнения совместных измерений за счет согласования

`estAccuracyIncreaseByDR`

Вызов	`[accuracy_increase_ratio, variances_of_DR_result] = estAccuracyIncreaseByDR(dep_func, ChosenMode, vals_for_reconc, no_error_params, errors)`
Описание	Функция выполняет приближенную оценку потенциального уточнения совместных измерений, достигаемого за счет учета известных функциональных взаимосвязей между измеряемыми величинами, на основе локальной линеаризации модели и метода декомпозиции алгоритма условной оптимизации.
Аргументы	`func` – указатель на функцию, возвращающую значения уравнений, формализующих взаимосвязь между измеряемыми величинами. `ChosenMode` – строка (`string`), позволяющей выбрать в каком режиме работает функция: по методу наименьших квадратов `'LS'` или по обобщенному методу наименьших квадратов `'WLS'`, `vals_for_reconc` – одномерный массив (`ndarray`), содержащий результаты совместных измерений, подлежащих уточнению; `no_error_params` – одномерный массив (`ndarray`) значений параметров уравнений, описывающих зависимости между измеряемыми величинами. `errors` – массив (`ndarray`), содержащий меры погрешностей согласуемых результатов измерений и параметров модели, используемой для согласования.
Возвращаемое значение	`accuracy_increase_ratio` – одномерный массив (`ndarray`) оценок степени повышения точности, которое может быть достигнуто за счет процедуры согласования; `variances_of_DR_result` – одномерный массив (`ndarray`) оценок дисперсий результатов согласования.

Функции согласования, учитывающего зависимости между взаимосвязанными величинами, выраженные в виде систем уравнений

Классическая процедура согласования совместных измерений в предположении, что случайные погрешности распределены нормально

`DRparamEq`

Вызов	`[reconciled] = DRparamEq(depend_func, msrd_data, error_params, params)`
Описание	Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей.
Аргументы	`depend_func` – указатель на функцию (`function`), возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами, `msrd_data` – массив (`ndarray`), содержащий согласуемые результаты измерений, `error_params` – одномерный массив (`ndarray`), содержащий меры погрешности независимо полученных результатов измерений, подлежащих согласованию, `params` – одномерный массив (`ndarray`) параметров модели взаимосвязей.
Возвращаемое значение	`reconciled` – одномерный массив (`ndarray`) результатов параметрического согласования совместных измерений, выполненного при следующих допущениях: случайные погрешности результатов совместных измерений распределены нормально; согласуемые результаты измерений получены независимо, и, следовательно, корреляция между случайными погрешностями отсутствует

Процедура семи-непараметрического согласования с представлением неизвестного закона распределения погрешностей рядом Грама-Шарлье

`DRsemiparamEq`

Вызов	`[reconciled] = DRsemiparamEq(depend_func, msrd_data, error_params, alpha_params, params)`
Описание	Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье.
Аргументы	`depend_func` – указатель на функцию (`function`), возвращающую значения левой части уравнений взаимосвязи между измеряемыми величинами; `msrd_data` – массив (`ndarray`), значения результатов совместного измерения величин, подлежащих согласованию; `error_params` – одномерный массив (`ndarray`), содержащий априорно заданные или оцененных меры погрешностей результатов измерений; `alpha_params` – двухмерный массив (`ndarray`), содержащий оценки среднеквадратического отклонения результатов измерений,

коэффициентов асимметрии, коэффициентов и эксцесса случайных отклонений результатов измерений,
`params` – одномерный массив `(ndarray)`, содержащий априорно заданные параметры модели, описывающей зависимости между согласуемыми величинами.

Возвращаемое значение

`reconciled` – одномерный массив `(ndarray)` результатов семи-непараметрического согласования совместных измерений.

Процедура непараметрического согласования с ядерной аппроксимацией распределения случайных погрешностей

`DRnonparamEq`

Вызов

```
reconciled = DRnonparamEq(func, msrd_data, model_params, prior_vars, bandwidths)
```

Описание

Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса.

Аргументы

`func` – указатель на функцию `(function)`, возвращающую значения левой части уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающих зависимости между измеряемыми величинами **x** и параметрами **a** модели;
`msrd_data` – одномерный массив `(ndarray)`, значения результатов однократного совместного измерения величин, подлежащих согласованию;
`model_params` – одномерный массив `(ndarray)` параметров модели, описывающих зависимости между согласуемыми величинами, заданных точно;
`prior_params` – одномерный массив `(ndarray)`, содержащий априорно заданные или оцененных дисперсии результатов измерений;
`bandwidth` – одномерный массив `(ndarray)`, содержащий регуляризирующее условие на результат ядерной аппроксимации неизвестного распределения случайных погрешностей согласуемых измерений. Данные значения являются ширинами окон ядерной аппроксимации. В качестве аппроксимирующего ядра использовано ядро Гаусса.

Возвращаемое значение

`reconciled` – одномерный массив `(ndarray)` результатов непараметрического согласования результатов измерений взаимосвязанных величин. Предполагается, что измерения выполнялись независимо, и корреляция между случайными погрешностями отсутствует. В качестве процедуры идентификации неизвестного распределения случайных погрешностей согласуемых измерений применен метод ядерной аппроксимации ядром Гаусса.

Классическая процедура робастного согласования совместных измерений в предположении, что случайные погрешности распределены нормально

DRparamEqRobust

Вызов
```
[reconciled] =
DRparamEqRobust(depend_func, msrd_data,
                error_params, params_)
```

Описание — Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей.

Аргументы — `depend_func` – указатель на функцию (`function`), возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами,
`msrd_data` – массив (`ndarray`), содержащий согласуемые результаты измерений,
`error_params` – одномерный массив (`ndarray`), содержащий меры погрешности независимо полученных результатов измерений, подлежащих согласованию,
`params` – одномерный массив (`ndarray`) параметров модели взаимосвязей.

Возвращаемое значение — `reconciled` – одномерный массив (`ndarray`) результатов параметрического согласования совместных измерений, выполненного при следующих допущениях: случайные погрешности результатов совместных измерений распределены нормально; согласуемые результаты измерений получены независимо, и, следовательно, корреляция между случайными погрешностями отсутствует

Процедура семи-непараметрического робастного согласования с представлением неизвестного закона распределения погрешностей рядом Грама-Шарлье

DRsemiparamEqRobust

Вызов
```
[reconciled] =
DRsemiparamEqReconcile(depend_func, msrd_data,
        error_params, alpha_params, params)
```

Описание — Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье.

Аргументы

`depend_func` – указатель на функцию (`function`), возвращающую значения левой части уравнений взаимосвязи между измеряемыми величинами;
`msrd_data` – массив (`ndarray`), значения результатов совместного измерения величин, подлежащих согласованию;
`error_params` – одномерный массив (`ndarray`), содержащий априорно заданные или оцененных меры погрешностей результатов измерений;
`alpha_params` – двухмерный массив (`ndarray`), содержащий оценки среднеквадратического отклонения результатов измерений, коэффициентов асимметрии, коэффициентов и эксцесса случайных отклонений результатов измерений,
`params` – одномерный массив (`ndarray`), содержащий априорно заданные параметры модели, описывающей зависимости между согласуемыми величинами.

Возвращаемое значение

`reconciled` – одномерный массив (`ndarray`) результатов семи-непараметрического согласования совместных измерений.

Процедура непараметрического робастного согласования с ядерной аппроксимацией распределения случайных погрешностей

`DRnonparamEqRobust`

Вызов

```
[reconciled] = DRnonparamEqRobust(func, msrd_data, model_params,
                                  prior_vars, bandwidths)
```

Описание

Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса.

Аргументы

`func` – указатель на функцию (`function`), возвращающую значения левой части уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающих зависимости между измеряемыми величинами **x** и параметрами **a** модели;
`msrd_data` – одномерный массив (`ndarray`), значения результатов однократного совместного измерения величин, подлежащих согласованию;
`model_params` – одномерный массив (`ndarray`) параметров модели, описывающих зависимости между согласуемыми величинами, заданных точно;
`prior_params` – одномерный массив (`ndarray`), содержащий априорно заданные или оцененных дисперсии результатов измерений;
`bandwidth` – одномерный массив (`ndarray`), содержащий регуляризирующее условие на результат ядерной аппроксимации неизвестного распределения случайных погрешностей согласуемых измерений. Данные значения являются ширинами окон ядерной

аппроксимации. В качестве аппроксимирующего ядра использовано ядро Гаусса.

Возвращаемое значение

`reconciled` – одномерный массив (`ndarray`) результатов непараметрического согласования результатов измерений взаимосвязанных величин. Предполагается, что измерения выполнялись независимо, и корреляция между случайными погрешностями отсутствует. В качестве процедуры идентификации неизвестного распределения случайных погрешностей согласуемых измерений применен метод ядерной аппроксимации ядром Гаусса.

Функции согласования, учитывающего зависимости между взаимосвязанными величинами, выраженные в виде систем уравнений и неравенств

Классическая процедура согласования совместных измерений в предположении, что случайные погрешности распределены нормально

`DRparamEqIneq`

Вызов

```
[reconciled] =
DRparamEq(eqies_model, ineqies_model,
          msrd_data, error_params, params)
```

Описание

Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей. Подлежат учету зависимости в виде равенств и неравенств.

Аргументы

`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$),

`ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами,

`msrd_data` – массив (`ndarray`), содержащий согласуемые результаты измерений,

`error_params` – одномерный массив (`ndarray`), содержащий меры погрешности независимо полученных результатов измерений, подлежащих согласованию,

`params` – одномерный массив (`ndarray`) параметров модели взаимосвязей.

Возвращаемое значение

`reconciled` – одномерный массив (`ndarray`) результатов параметрического согласования совместных измерений, выполненного при следующих допущениях: случайные погрешности результатов совместных измерений распределены нормально; согласуемые результаты измерений получены независимо, и, следовательно, корреляция между случайными погрешностями отсутствует

Процедура семи-непараметрического согласования с представлением неизвестного закона распределения погрешностей рядом Грама-Шарлье

DRsemiparamEqIneq

Вызов

```
[reconciled] =
DRsemiparamEq(eqies_model, ineqies_model,
        msrd_data, error_params, alpha_params, params)
```

Описание

Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье. Подлежат учету зависимости в виде равенств и неравенств.

Аргументы

`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$),

`ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами,

`msrd_data` – массив `(ndarray)`, значения результатов совместного измерения величин, подлежащих согласованию;

`error_params` – одномерный массив `(ndarray)`, содержащий априорно заданные или оцененных меры погрешностей результатов измерений;

`alpha_params` – двухмерный массив `(ndarray)`, содержащий оценки среднеквадратического отклонения результатов измерений, коэффициентов асимметрии, коэффициентов и эксцесса случайных отклонений результатов измерений,

`params` – одномерный массив `(ndarray)`, содержащий априорно заданные параметры модели, описывающей зависимости между согласуемыми величинами.

Возвращаемое значение

`reconciled` – одномерный массив `(ndarray)` результатов семи-непараметрического согласования совместных измерений.

Процедура непараметрического согласования с ядерной аппроксимацией распределения случайных погрешностей

DRnonparamEqIneq

Вызов

```
[reconciled] = DRnonparamEqIneq (
    eqies_model, ineqies_model,
    msrd_data, model_params, prior_vars, bandwidths)
```

Описание	Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса. Подлежат учету зависимости в виде равенств и неравенств.
Аргументы	`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$), `ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами; `msrd_data` – одномерный массив `(ndarray)`, значения результатов однократного совместного измерения величин, подлежащих согласованию; `model_params` – одномерный массив `(ndarray)` параметров модели, описывающих зависимости между согласуемыми величинами, заданных точно; `prior_params` – одномерный массив `(ndarray)`, содержащий априорно заданные или оцененных дисперсии результатов измерений; `bandwidth` – одномерный массив `(ndarray)`, содержащий регуляризирующее условие на результат ядерной аппроксимации неизвестного распределения случайных погрешностей согласуемых измерений. Данные значения являются ширинами окон ядерной аппроксимации. В качестве аппроксимирующего ядра использовано ядро Гаусса.
Возвращаемое значение	`reconciled` – одномерный массив `(ndarray)` результатов непараметрического согласования результатов измерений взаимосвязанных величин. Предполагается, что измерения выполнялись независимо, и корреляция между случайными погрешностями отсутствует. В качестве процедуры идентификации неизвестного распределения случайных погрешностей согласуемых измерений применен метод ядерной аппроксимации ядром Гаусса.

Классическая процедура робастного согласования совместных измерений в предположении, что случайные погрешности распределены нормально

`DRparamEqIneqRobust`

Вызов	`[reconciled] = DRparamEqIneqRobust(eqies_model, ineqies_model, msrd_data, error_params, params_)`
Описание	Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей

которых соответствует нормальному распределению вероятностей. Подлежат учету зависимости в виде равенств и неравенств.

Аргументы

`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$),
`ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами;
`msrd_data` – массив (`ndarray`), содержащий согласуемые результаты измерений,
`error_params` – одномерный массив (`ndarray`), содержащий меры погрешности независимо полученных результатов измерений, подлежащих согласованию,
`params` – одномерный массив (`ndarray`) параметров модели взаимосвязей.

Возвращаемое значение

`reconciled` – одномерный массив (`ndarray`) результатов параметрического согласования совместных измерений, выполненного при следующих допущениях: случайные погрешности результатов совместных измерений распределены нормально; согласуемые результаты измерений получены независимо, и, следовательно, корреляция между случайными погрешностями отсутствует

Процедура семи-непараметрического робастного согласования с представлением неизвестного закона распределения погрешностей рядом Грама-Шарлье

`DRsemiparamEqIneqRobust`

Вызов

```
[reconciled] =
DRsemiparamEqIneqReconcile(eqies_model,
        ineqies_model,
        msrd_data, error_params,
        alpha_params, params)
```

Описание

Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье. Подлежат учету зависимости в виде равенств и неравенств.

Аргументы

`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$),

`ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами;

`msrd_data` – массив (`ndarray`), значения результатов совместного измерения величин, подлежащих согласованию;

`error_params` – одномерный массив (`ndarray`), содержащий априорно заданные или оцененных меры погрешностей результатов измерений;

`alpha_params` – двухмерный массив (`ndarray`), содержащий оценки среднеквадратического отклонения результатов измерений, коэффициентов асимметрии, коэффициентов и эксцесса случайных отклонений результатов измерений,

`params` – одномерный массив (`ndarray`), содержащий априорно заданные параметры модели, описывающей зависимости между согласуемыми величинами.

Возвращаемое значение

`reconciled` – одномерный массив (`ndarray`) результатов семи-непараметрического согласования совместных измерений.

Процедура непараметрического робастного согласования с ядерной аппроксимацией распределения случайных погрешностей

`DRnonparamEqIneqRobust`

Вызов

```
[reconciled] = DRnonparamEqIneqRobust(
            eqies_model, ineqies_model,
            msrd_data, model_params,
            prior_vars, bandwidths)
```

Описание

Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса. Подлежат учету зависимости в виде равенств и неравенств.

Аргументы

`eqies_model` – функция, возвращающую результат вычисления левых частей системы уравнений вида $\mathbf{f}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$, описывающую взаимосвязи между согласуемыми величинами (размерность возвращаемого вектора соответствует размерности вектор-функции $\mathbf{f}_M$),

`ineqies_model` – функция, возвращающую результат вычисления левых частей системы неравенств вида $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) \geq \mathbf{0}$, представленных в формате равенств $\mathbf{g}_M(\mathbf{x}, \mathbf{a}) = \mathbf{0}$ (в соответствии с методом множителей Лагранжа), описывающую взаимосвязи (ограничения) между согласуемыми величинами;

	`msrd_data` – одномерный массив (`ndarray`), значения результатов однократного совместного измерения величин, подлежащих согласованию; `model_params` – одномерный массив (`ndarray`) параметров модели, описывающих зависимости между согласуемыми величинами, заданных точно; `prior_params` – одномерный массив (`ndarray`), содержащий априорно заданные или оцененных дисперсии результатов измерений; `bandwidth` – одномерный массив (`ndarray`), содержащий регуляризирующее условие на результат ядерной аппроксимации неизвестного распределения случайных погрешностей согласуемых измерений. Данные значения являются ширинами окон ядерной аппроксимации. В качестве аппроксимирующего ядра использовано ядро Гаусса.
Возвращаемое значение	`reconciled` – одномерный массив (`ndarray`) результатов непараметрического согласования результатов измерений взаимосвязанных величин. Предполагается, что измерения выполнялись независимо, и корреляция между случайными погрешностями отсутствует. В качестве процедуры идентификации неизвестного распределения случайных погрешностей согласуемых измерений применен метод ядерной аппроксимации ядром Гаусса.

Конструктор объектов типа PBox, служащих для отображения множества возможных значений согласуемых величин и/или их неопределенности в форме области возможных значений функции распределения

`PBox`

Вызов	`[obj] = PBox(x, lowerCDF, upperCDF)`
Описание	Функция представляет собой конструктор класса типа PBox, служащего для отображения возможных значений и/или неопределенности согласуемых величин в виде области возможных значений функции распределения.
Аргументы	`x` – набор значений, отражающих узлы сетки дискретизации значений аргумента кумулятивной функции распределения (cdf); `lowerCDF` – набор значений нижней границы области допустимых значений функции распределения; `upperCDF` – набор значений верхней границы области допустимых значений функции распределения.
Возвращаемое значение	`obj` – созданный объект типа PBox.

Конструктор объектов типа Hist, служащих для отображения множества возможных значений согласуемых величин и/или их неопределенности в форме интервальной гистограммы по Берлинту

Hist

Вызов	`[obj] = Hist(x, lowerPDF, upperPDF)`
Описание	Функция представляет собой конструктор класса типа Hist, служащего для отображения возможных значений и/или неопределенности согласуемых величин в виде гистограммы функции плотности распределения с интервальным заданием возможных значений высоты ее полос.
Аргументы	`x` – массив значений, последовательно содержащий границы полос гистограммы (pdf); `lowerPDF` – массив значений, отображающих нижние границы множества допустимых значений высоты полос гистограммы; `upperPDF` – массив значений, отображающих нижние границы множества допустимых значений высоты полос гистограммы.
Возвращаемое значение	`obj` – созданный объект типа Hist.

Конструктор объектов типа DermpsterShafer, служащих для отображения множества возможных значений согласуемых величин и/или их неопределенности в форме структуры Демпстера-Шафера

DempsterShafer

Вызов	`[obj] = DempsterShafer(intervals, masses)`
Описание	Функция представляет собой конструктор класса типа DempsterShafer, служащего для отображения возможных значений и/или неопределенности согласуемых величин в виде структуры Демпстера-Шафера.
Аргументы	`intervals` – массив интервалов для фокальных элементов; `masses` – соответствующие им массы.
Возвращаемое значение	`obj` – созданный объект типа DempsterShafer.

Конструктор объектов типа Fuzzy, служащих для отображения множества возможных значений согласуемых величин и/или их неопределенности в форме нечеткой переменной по Заде

Fuzzy

Вызов	`[obj] = Fuzzy(universe, membership)`
Описание	Функция представляет собой конструктор класса типа Fuzzy, служащего для отображения возможных значений и/или неопределенности согласуемых величин в виде нечеткой переменной.
Аргументы	`universe` – значения носителя нечеткой переменной; `membership` – значения функции принадлежности (от 0 до 1).
Возвращаемое значение	`obj` – созданный объект типа Fuzzy.

Конструктор объектов типа FuzzyInterval, служащих для отображения множества возможных значений согласуемых величин и/или их неопределенности в форме нечеткого интервала

FuzzyInterval

Вызов	`[obj] = FuzzyInterval(alphaLevels, intervals)`
Описание	Функция представляет собой конструктор класса типа FuzzyInterval, служащего для отображения возможных значений и/или неопределенности согласуемых величин в виде нечеткого интервала (включает в себя как предельный случай классический интервал, а исчисление нечетких интервалов – классическую интервальную арифметику по Муру).
Аргументы	`alphaLevels` – так называемые значения α-cut (уровни значений функции принадлежности); `intervals` – соответствующие вложенные интервалы (nested intervals).
Возвращаемое значение	`obj` – созданный объект типа FuzzyInterval.

Конструктор объектов типа Sample, служащих для отображения выборки значений результатов многократных измерений

Sample

Вызов	`[obj] = Sample(X)`
Описание	Функция представляет собой конструктор класса типа Sample, служащего для отображения выборки значений результатов многократных измерений одной и той же отображаемой величины.

Аргументы	`x` – массив значений, образующих выборку значений результатов многократных измерений отображаемой величины.
Возвращаемое значение	`obj` – созданный объект типа Sample.

Функция преобразования переменной типа PBox в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

PBox2Hist

Вызов	`[Histogram] = PBox2Hist(self, numBins)`
Описание	Функция представляет собой процедуру преобразования переменной типа PBox в переменную типа Hist с минимизацией потерь содержащейся в PBox информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса PBox; `numBins`– количество полос в создаваемом экземпляре класса Hist
Возвращаемое значение	`obj` – созданный объект типа Hist.

Функция преобразования переменной типа PBox в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

PBox2DempsterShafer

Вызов	`[ds] = PBox2DempsterShafer(self, numFocal)`
Описание	Функция представляет собой процедуру преобразования переменной типа PBox в переменную типа DempsterShafer с минимизацией потерь содержащейся в PBox информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса PBox; `numFocal` – количество фокальных элементов в создаваемом экземпляре класса DempsterShafer
Возвращаемое значение	`ds` – созданный объект типа DempsterShafer.

Функция преобразования переменной типа PBox в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

PBox2Fuzzy

Вызов	`[fv] = PBox2Fuzzy(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа PBox в переменную типа Fuzzy с минимизацией потерь содержащейся в PBox информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса PBox; `numPoints` – количество значений универсального множества для создаваемой нечеткой переменной (экземпляре класса Fuzzy)
Возвращаемое значение	`fv` – созданный объект типа Fuzzy.

Функция преобразования переменной типа PBox в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

PBox2FuzzyInterval

Вызов	`[fi] = PBox2FuzzyInterval(self, numAlpha)`
Описание	Функция представляет собой процедуру преобразования переменной типа PBox в переменную типа FuzzyInterval с минимизацией потерь содержащейся в PBox информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса PBox; `numAlpha` – количество значений уровней значений функции принадлежности для множества вложенных интервалов, образующих создаваемый экземпляр класса FuzzyInterval.
Возвращаемое значение	`fv` – созданный объект типа FuzzyInterval.

Функция преобразования переменной типа Hist в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

Hist2PBox

Вызов	`[pbox] = Hist2PBox(self)`
Описание	Функция представляет собой процедуру преобразования переменной типа Hist в переменную типа PBox с минимизацией потерь содержащейся в Hist информации о возможных значениях отображаемой величины.

Аргументы	`self` – преобразуемый экземпляр класса Hist.
Возвращаемое значение	`pbox` – созданный объект типа PBox.

Функция преобразования переменной типа Hist в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`Hist2DempsterShafer`

Вызов	`[ds] = Hist2DempsterShafer(self)`
Описание	Функция представляет собой процедуру преобразования переменной типа Hist в переменную типа DempsterShafer с минимизацией потерь содержащейся в Hist информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Hist.
Возвращаемое значение	`ds` – созданный объект типа DempsterShafer.

Функция преобразования переменной типа Hist в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`Hist2Fuzzy`

Вызов	`[fv] = Hist2Fuzzy(self, numUniverse)`
Описание	Функция представляет собой процедуру преобразования переменной типа Hist в переменную типа Fuzzy с минимизацией потерь содержащейся в Hist информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Hist; `numUniverse` – количество значений универсального множества для создаваемой нечеткой переменной (экземпляре класса Fuzzy)
Возвращаемое значение	`fv` – созданный объект типа Fuzzy.

Функция преобразования переменной типа Hist в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`Hist2FuzzyInterval`

Вызов	`[fi] = Hist2FuzzyInterval(self, numAlpha)`
Описание	Функция представляет собой процедуру преобразования переменной типа Hist в переменную типа FuzzyInterval с минимизацией потерь содержащейся в Hist информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Hist; `numAlpha` – количество значений уровней значений функции принадлежности для множества вложенных интервалов, образующих создаваемый экземпляр класса FuzzyInterval.
Возвращаемое значение	`fv` – созданный объект типа FuzzyInterval.

Функция преобразования переменной типа DempsterShafer в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`DempsterShafer2PBox`

Вызов	`[pbox] = DempsterShafer2PBox(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа DempsterShafer в переменную типа PBox с минимизацией потерь содержащейся в DempsterShafer информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса DempsterShafer; `numPoints` – количество значений в сетке дискретизации границ области возможных значений функции распределения
Возвращаемое значение	`pbox` – созданный объект типа PBox.

Функция преобразования переменной типа DempsterShafer в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`DempsterShafer2Hist`

Вызов	`[Histogram] = DempsterShafer2Hist(self, numBins)`

Описание	Функция представляет собой процедуру преобразования переменной типа DempsterShafer в переменную типа Hist с минимизацией потерь содержащейся в DempsterShafer информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса DempsterShafer; `numBins` – количество полос в создаваемом экземпляре класса Hist (интервальнозначеной гистограмме)
Возвращаемое значение	`Histogram` – созданный объект типа Hist.

Функция преобразования переменной типа DempsterShafer в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`DempsterShafer2Fuzzy`

Вызов	`[fv] = DempsterShafer2Fuzzy(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа DempsterShafer в переменную типа Fuzzy с минимизацией потерь содержащейся в DempsterShafer информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса DempsterShafer; `numPoints` – количество значений универсального множества для создаваемой нечеткой переменной (экземпляре класса Fuzzy)
Возвращаемое значение	`fv` – созданный объект типа Fuzzy.

Функция преобразования переменной типа DempsterShafer в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`DempsterShafer2FuzzyInterval`

Вызов	`[fi] = DempsterShafer2FuzzyInterval(self)`
Описание	Функция представляет собой процедуру преобразования переменной типа DempsterShafer в переменную типа FuzzyInterval с минимизацией потерь содержащейся в DempsterShafer информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса DempsterShafer.
Возвращаемое значение	`fv` – созданный объект типа FuzzyInterval.

Функция преобразования переменной типа Fuzzy в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

Fuzzy2PBox

Вызов	`[pbox] = Fuzzy2PBox(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа Fuzzy в переменную типа PBox с минимизацией потерь содержащейся в Fuzzy информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Fuzzy; `numPoints` – количество значений в сетке дискретизации границ области возможных значений функции распределения
Возвращаемое значение	`pbox` – созданный объект типа PBox.

Функция преобразования переменной типа Fuzzy в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

Fuzzy2Hist

Вызов	`[Histogram] = Fuzzy2Hist(self, numBins)`
Описание	Функция представляет собой процедуру преобразования переменной типа Fuzzy в переменную типа Hist с минимизацией потерь содержащейся в Fuzzy информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Fuzzy; `numBins`– количество полос в создаваемом экземпляре класса Hist
Возвращаемое значение	`Histogram` – созданный объект типа Hist.

Функция преобразования переменной типа Fuzzy в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

Fuzzy2DempsterShafer

Вызов	`[ds] = Fuzzy2DempsterShafer(self, numFocal)`

Описание	Функция представляет собой процедуру преобразования переменной типа Fuzzy в переменную типа DempsterShafer с минимизацией потерь содержащейся в Fuzzy информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Fuzzy; `numFocal` – количество фокальных элементов в создаваемом экземпляре класса DempsterShafer
Возвращаемое значение	`ds` – созданный объект типа DempsterShafer.

Функция преобразования переменной типа Fuzzy в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`Fuzzy2FuzzyInterval`

Вызов	`[fi] = Fuzzy2FuzzyInterval(self, numAlpha)`
Описание	Функция представляет собой процедуру преобразования переменной типа Fuzzy в переменную типа FuzzyInterval с минимизацией потерь содержащейся в Fuzzy информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса Fuzzy; `numAlpha` – количество значений уровней значений функции принадлежности для множества вложенных интервалов, образующих создаваемый экземпляр класса FuzzyInterval.
Возвращаемое значение	`fv` – созданный объект типа FuzzyInterval.

Функция преобразования переменной типа FuzzyInterval в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

`FuzzyInterval2PBox`

Вызов	`[pbox] = FuzzyInterval2PBox(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа FuzzyInterval в переменную типа PBox с минимизацией потерь содержащейся в FuzzyInterval информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса FuzzyInterval; `numPoints` – количество значений в сетке дискретизации границ области возможных значений функции распределения
Возвращаемое значение	`pbox` – созданный объект типа PBox.

Функция преобразования переменной типа FuzzyInterval в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

FuzzyInterval2Hist

Вызов	`[Histogram] = FuzzyInterval2Hist(self, numBins)`
Описание	Функция представляет собой процедуру преобразования переменной типа FuzzyInterval в переменную типа Hist с минимизацией потерь содержащейся в FuzzyInterval информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса FuzzyInterval; `numBins`– количество полос в создаваемом экземпляре класса Hist
Возвращаемое значение	`Histogram` – созданный объект типа Hist.

Функция преобразования переменной типа FuzzyInterval в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

FuzzyInterval 2DempsterShafer

Вызов	`[ds] = FuzzyInterval2DempsterShafer(self, method)`
Описание	Функция представляет собой процедуру преобразования переменной типа FuzzyInterval в переменную типа DempsterShafer с минимизацией потерь содержащейся в FuzzyInterval информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса FuzzyInterval; `method` – метод преобразования: ‘fractional’ или ‘nested’.
Возвращаемое значение	`ds` – созданный объект типа DempsterShafer.

Функция преобразования переменной типа FuzzyInterval в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование)

FuzzyInterval2Fuzzy

Вызов	`[fv] = FuzzyInterval2Fuzzy(self, numPoints)`

Описание	Функция представляет собой процедуру преобразования переменной типа FuzzyInterval в переменную типа Fuzzy с минимизацией потерь содержащейся в FuzzyInterval информации о возможных значениях отображаемой величины.
Аргументы	`self` – преобразуемый экземпляр класса FuzzyInterval; `numPoints` – количество значений универсального множества для создаваемой нечеткой переменной (экземпляре класса Fuzzy)
Возвращаемое значение	`fv` – созданный объект типа Fuzzy.

Функция преобразования переменной типа Sample в переменную типа PBox (построение области возможных значений функции распределения по выборке результатов многократных измерений согласуемой величины)

`Sample2PBox`

Вызов	`[pbox] = Sample2PBox(self)`
Описание	Функция представляет собой процедуру преобразования переменной типа Sample в переменную типа PBox.
Аргументы	`self` – преобразуемый экземпляр класса Sample.
Возвращаемое значение	`pbox` – созданный объект типа PBox.

Функция преобразования переменной типа Sample в переменную типа Hist (построение интервальнозначной гистограммы по Берлинту по выборке результатов многократных измерений согласуемой величины)

`Sample2Hist`

Вызов	`[Histogram] = Sample2Hist(self, numBins)`
Описание	Функция представляет собой процедуру преобразования переменной типа Sample в переменную типа Hist.
Аргументы	`self` – преобразуемый экземпляр класса Sample; `numBins`– количество полос в создаваемом экземпляре класса Hist
Возвращаемое значение	`Histogram` – созданный объект типа Hist.

Функция преобразования переменной типа Sample в переменную типа DempsterShafer (построение структуры Демпстера-Шафера по выборке результатов многократных измерений согласуемой величины)

Sample2DempsterShafer

Вызов	`[ds] = Sample2DempsterShafer(self, numFocal)`
Описание	Функция представляет собой процедуру преобразования переменной типа Sample в переменную типа DempsterShafer.
Аргументы	`self` – преобразуемый экземпляр класса Sample; `numFocal` – количество фокальных элементов в создаваемом экземпляре класса DempsterShafer.
Возвращаемое значение	`ds` – созданный объект типа DempsterShafer.

Функция преобразования переменной типа Sample в переменную типа Fuzzy (построение нечеткой переменной по выборке результатов многократных измерений согласуемой величины)

Sample2Fuzzy

Вызов	`[fv] = Sample2Fuzzy(self, numPoints)`
Описание	Функция представляет собой процедуру преобразования переменной типа Sample в переменную типа Fuzzy.
Аргументы	`self` – преобразуемый экземпляр класса Sample; `numPoints` – количество значений универсального множества для создаваемой нечеткой переменной (экземпляре класса Fuzzy)
Возвращаемое значение	`fv` – созданный объект типа Fuzzy.

Функция преобразования переменной типа Sample в переменную типа FuzzyInterval (построение нечеткого интервала по выборке результатов многократных измерений согласуемой величины)

Sample2FuzzyInterval

Вызов	`[fi] = Sample2FuzzyInterval(self, numAlpha)`
Описание	Функция представляет собой процедуру преобразования переменной типа Sample в переменную типа FuzzyInterval.
Аргументы	`self` – преобразуемый экземпляр класса Sample; `numAlpha` – количество значений уровней значений функции принадлежности для множества вложенных интервалов, образующих создаваемый экземпляр класса FuzzyInterval.

Возвращаемое значение	`fv` – созданный объект типа FuzzyInterval.

Функция графического отображения экземпляров классов, служащих для отображения множества возможных значений и/или неопределенности согласуемых величин

`Plot`

Вызов	`[] = plot(self)`
Описание	Функция представляет собой процедуру графического отображения информации, содержащейся в объекте obj, который должен относиться к одному из типов представления неопределенности: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample.
Аргументы	`self` – отображаемый экземпляр одного из классов представления информации о неопределенности: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample.
Возвращаемое значение	нет.

Функция построения интервала возможных значений (доверительного интервала) среднеквадратического отклонения того множества возможных значений или неопределенности согласуемой величины, что отражается экземплярами классов PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample

`getStd`

Вызов	`[stdCint] = getStd(self, method)`
Описание	Функция оценки возможных значений среднеквадратического отклонения неточной величины, представленной в одном из форматов представления неопределенности (метод классов PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample).
Аргументы	`obj` – экземпляр одного из классов представления информации о неопределенности: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample; `method` – метод оценки множества возможных значений среднеквадратического отклонения: ‘approximate’ – приближенная, но быстрая оценка; ‘accurate’ – точная, но более продолжительная по времени оценка.
Возвращаемое значение	`stdCint` – границы интервала возможных значений среднеквадратического отклонения (доверительного интервала), большее из значений которых используется для дальнейшего выполнения согласования индустриальных данных.

Процедура выполнения согласования неточных величин в рамках аналитической модели с решением задачи Каруша-Куна-Таккера с применением теоремы Ефремова-Козлова

AnalyticalModel

Вызов	`[y] = AnalyticalModel(x, A_y, b_y, c_y, r2_y)`
Описание	Процедура выполнения согласования неточных величин в рамках аналитической модели с решением задачи Каруша-Куна-Таккера с применением теоремы Ефремова-Козлова.
Аргументы	`x` – вектор значений согласуемых величин (результаты совместных измерений на объекте, чья математическая модель описывает ограничения на возможные значения согласуемых значений); `A_y` – матрица линейных ограничений, накладываемых на согласуемые значения, вида $A\_y \cdot y = b\_y$; `b_y` – вектор правых частей в линейных ограничений, накладываемых на согласуемые значения, вида $A\_y \cdot y = b\_y$; `c_y` – вектор, соответствующий координатам центра эллипсоида в пространстве значений вектора `x`, накладывающего групповое условие типа неравенства на возможную область допустимых значений вектора `y`, вида $(y-c\_y)^T \cdot (y-c\_y) = r2\_y$; `r2_y` – величина группового ограничения на возможную область допустимых значений вектора `y`, вида $(y-c\_y)^T \cdot (y-c\_y) = r2\_y$.
Возвращаемое значение	`y` – результаты согласования значений по аналитической модели процедуры Data Reconciliation, основанной на теореме Ефремова-Козлова.

Процедура выполнения оценки неопределенности результата согласования по аналитической модели AnalyticalModel при представлении неопределенности исходных данных в одном из перечисленных форматов: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample

GetUncertainty

Вызов	`[dy] = GetUncertainty(x, dx, A_y, b_y, c_y, r2_y)`
Описание	Процедура выполнения оценки неопределенности результата согласования по аналитической модели AnalyticalModel при представлении неопределенности исходных данных в одном из перечисленных форматов: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample.
Аргументы	`x` – вектор значений согласуемых величин (результаты совместных измерений на объекте, чья математическая модель описывает ограничения на возможные значения согласуемых значений);

	dx – массив экземпляров одного из классов выражения неопределенности согласуемых величин x (на выбор пользователя – PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample); A_y – матрица линейных ограничений, накладываемых на согласуемые значения, вида A_y · y = b_y; b_y – вектор правых частей в линейных ограничений, накладываемых на согласуемые значения, вида A_y · y = b_y; c_y – вектор, соответствующий координатам центра эллипсоида в пространстве значений вектора x, накладывающего групповое условие типа неравенства на возможную область допустимых значений вектора y, вида (y−c_y)ᵀ·(y−c_y) = r2_y; r2_y – величина группового ограничения на возможную область допустимых значений вектора y, вида (y−c_y)ᵀ·(y−c_y) = r2_y.
Возвращаемое значение	dy – массив оценок неопределенности результатов согласования значений по аналитической модели процедуры Data Reconciliation, основанной на теореме Ефремова-Козлова; результаты возвращаются в виде экземпляра того же класса, что и dx.

Б.5.3 Примеры и особенности

Приведённая в листинге Б.1 функция DRparamEq() предназначена для согласования измеренных данных с использованием метода, основанного на численном решении системы уравнений. Она использует модель зависимости между измеренными величинами, заданную функцией depend_func, и априорные ошибки измерений для корректировки данных. Решается задача оптимальной оценки значений физических величин, известных с погрешностью, при этом априорно заданные взаимосвязи меду величинами формализованы в виде системы уравнений.

Листинг Б.1 – Функция DRparamEq() для согласования данных с использованием метода, основанного на численной оптимизации

```
import numpy as np
from numpy import imag
from scipy.optimize import fsolve


def gauss_system_to_solve(mu_and_l, depend_func, msrd_data, vars_, params_):
    """ args=(depend_func, msrd_data, error_params, params_)
    Solves for the gauss system to find zeros.

    Parameters:
        mu_and_l (ndarray): Array of unknown parameters (mu and lambda).
        depend_func (function): Dependency model function.
        msrd_data (ndarray): Measured data (n x xNum array).
        vars_ (ndarray): Error parameters, should not contain zeros.
```

```
        params_ (ndarray or list): Constant dependency model parameters.

    Returns:
        ndarray: A vector of values that need to be zero (mustbezeros).
    """

    if msrd_data.ndim == 1:
        xNum = msrd_data.size
        n = 1
    elif msrd_data.ndim == 2:
        xNum = msrd_data.shape[0]
        n = msrd_data.shape[1]
    else:
        raise ValueError(
                "Only 2-dimensional arrays msrd_data are supported.")

    mustbezeros = np.zeros(xNum, dtype=np.complex128)  # Preallocate zeros

    for j in range(xNum):
        if vars_[j] == 0:
            raise ValueError(
                "None of the error parameters should be 0. Note that all constant "
                "dependency model parameters need to be in the "
                "'params_' array.")
        else:
            # Imaginary perturbation for numerical derivative
            imagx = np.zeros(xNum, dtype=np.complex128)
            imagx[j] = 1j * (mu_and_l[j] * 10 ** (-100) + 10 ** (-101))

            # Numerical derivative approximation using imaginary perturbation
            df_dx = imag(depend_func(mu_and_l[:xNum] + imagx, params_)) / imag(imagx[j])

            # Update mustbezeros
            mustbezeros[j] = (mu_and_l[j] / vars_[j]
                              - np.mean(msrd_data[j,:]) / vars_[j]
                              + np.sum(mu_and_l[xNum:] * df_dx) / n)

    # Evaluate the model residuals
    model_res = depend_func(mu_and_l[:xNum], params_)
    mustbezeros = np.append(mustbezeros, model_res)
```

```
    return np.real(mustbezeros)
# ------------------------------------------------------------------------------
def GaussDataReconcil(depend_func, msrd_data, error_params, params_):
    """
    Reconciles measured data

    Parameters:
        depend_func (function): Function describing relationships between measured quantities.
                            It can return either a single equation or a system of equations.
        msrd_data (ndarray): Vector of measured data.
        error_params (ndarray): A priori error variances/limits.
        params_ (ndarray): Known  model parameters.

    Returns:
        ndarray: Reconciled measured data after applying the Alpha Gram-Charlier method.
    """
    data_init_points = np.mean(msrd_data, axis=1)
    # Step 1: Compute l (a zero vector of the same length as depend_func's output)
    l = np.zeros(len(depend_func(data_init_points, params_)))

    # Step 2: Extend the initial guess (mu_and_l_start) by appending l to msrd_data
    mu_and_l_start = np.hstack((data_init_points, l))

    # Solve the system of equations using fsolve
    mu_and_l_res = fsolve(gauss_system_to_solve, mu_and_l_start, args=(depend_func, msrd_data, error_params, params_))

    # Step 4: Extract the reconciled measured data (excluding 'l' values)
    reconciled_ = mu_and_l_res[:len(msrd_data)]

    return reconciled_
```

Пример использования функции

Предположим, у нас есть набор измеренных данных, которые мы хотим согласовать с использованием модели зависимости.

Модель зависимости может быть, например, линейной функцией:

```
def linear_model(mu, params):
    """
```

```
    Линейная модель зависимости: y = a * x + b.
    Здесь params = [a, b].
    """
    a, b = params
    return a * mu + b
```

Подготовка данных

Измеренные данные: 5 измерений, каждое измерение повторено 3 раза.

Априорные ошибки: заданы для каждого измерения.

Параметры модели: `a = 2, b = 1`.

```
import numpy as np

# Измеренные данные (5 измерений, каждое повторено 3 раза)
msrd_data = np.array([
    [1.1, 1.2, 1.3],  # Измерение 1
    [2.0, 2.1, 2.2],  # Измерение 2
    [3.0, 3.1, 3.2],  # Измерение 3
    [4.0, 4.1, 4.2],  # Измерение 4
    [5.0, 5.1, 5.2]   # Измерение 5
])

# Априорные ошибки (дисперсии) для каждого измерения
error_params = np.array([0.1, 0.1, 0.1, 0.1, 0.1])

# Параметры модели (a и b для линейной модели)
params_ = np.array([2, 1])
```

Вызов функции

```
# Согласование данных
reconciled_data = DRparamEq(linear_model, msrd_data, error_params, params_)
# Вывод результата
print("Согласованные данные:", reconciled_data)
```

Результат

Функция вернет вектор согласованных данных, которые лучше соответствуют модели зависимости и учитывают погрешность измерений.

Инициализация начальных значений:

Начальные значения для согласованных данных (`mu`) берутся как средние значения измерений.

Начальные значения для множителей Лагранжа (`l`) устанавливаются в нули.

Решение системы уравнений:

Функция `fsolve` из библиотеки `scipy.optimize` используется для решения системы уравнений, которая формируется в функции `gauss_system_to_solve`.

Система уравнений включает:

Уравнения для согласования данных с учетом априорных ошибок.

Уравнения, заданные моделью зависимости.

Функция `gauss_system_to_solve` использует метод численного дифференцирования с мнимой возмущением для вычисления производных.

Если априорные ошибки (`error_params`) содержат нули, функция вызовет ошибку, так как это приведет к делению на ноль.

Функция поддерживает одномерные и двумерные массивы для `msrd_data`, то есть позволяет согласовывать однократные и многократные совместные измерения, соответственно.

Приведённая в листинге Б.2 функция `DRparamEq()` предназначена для согласования измеренных данных с использованием метода, основанного на численном решении системы уравнений. Она использует модель зависимости между измеренными величинами, заданную функцией `depend_func`, и априорные ошибки измерений для корректировки данных. Решается задача оптимальной оценки значений физических величин, известных с погрешностью, при этом априорно заданные взаимосвязи меду величинами формализованы в виде системы уравнений.

Листинг Б.2 – Функция `DRparamEq()`, предназначенная для согласования измеренных данных с использованием метода, основанного на численном решении системы уравнений и неравенств по методу множителей Лагранжа

```
import numpy as np
from numpy import imag
from scipy.optimize import fsolve


def  gauss_system_to_solve(mu_and_l,  eqies_model,  ineqies_model,  msrd_data,  vars_,
params_, xNum, n, eqNum, ineqNum):
    """ args=(eqies_model, msrd_data, error_params, params_)
    Solves for the gauss system to find zeros.

    Parameters:
        mu_and_l (ndarray): Array of unknown parameters (mu and lambda).
        eqies_model (function): Dependency model function.
        msrd_data (ndarray): Measured data (n x xNum array).
        vars_ (ndarray): Error parameters, should not contain zeros.
```

```
        params_ (ndarray or list): Constant dependency model parameters.

    Returns:
        ndarray: A vector of values that need to be zero (mustbezeros).
    """

    mustbezeros = np.zeros(xNum, dtype=np.complex128)  # Preallocate zeros

    for j in range(xNum):
        if vars_[j] == 0:
            raise ValueError(
                "None of the error parameters should be 0. Note that all constant "
                "dependency model parameters need to be in the "
                "'params_' array.")
        else:
            # Imaginary perturbation for numerical derivative
            imagx = np.zeros(xNum, dtype=np.complex128)
            imagx[j] = 1j * (mu_and_l[j] * 10 ** (-100) + 10 ** (-101))

            # Numerical derivative approximation using imaginary perturbation
            df_dx_eqies  =  imag(eqies_model(mu_and_l[:xNum]  +  imagx,  params_))  /
imag(imagx[j])
            df_dx_ineqies  =  imag(eqies_model(mu_and_l[:xNum]  +  imagx,  params_))  /
imag(imagx[j])




            df_dx = np.hstack((df_dx_eqies, df_dx_ineqies))
            # Update mustbezeros
            mustbezeros[j] = (mu_and_l[j] / vars_[j]
                              - np.mean(msrd_data[j,:]) / vars_[j]
                              + np.sum(mu_and_l[xNum:(xNum+eqNum+ineqNum)] * df_dx) /
n)

    # Evaluate the model residuals
    eqies_model_res = eqies_model(mu_and_l[:xNum], params_)
    mustbezeros = np.append(mustbezeros, eqies_model_res)

    ineqies_model_res = ineqies_model(mu_and_l[:xNum], params_)
    mustbezeros = np.append(mustbezeros, ineqies_model_res)
```

```
    mustbezeros                          =                          np.append(mustbezeros,
ineqies_model_res*mu_and_l[(xNum+eqNum):(xNum+eqNum+ineqNum)])


    return np.real(mustbezeros)


# ------------------------------------------------------------------------


def DRparamEqIneq(eqies_model, ineqies_model, msrd_data, error_params, params_):
    """
    Reconciles measured data


    Parameters:
        depend_func (function): Function describing relationships between measured
quantities.
                        It can return either a single equation or a system of equations.
        msrd_data (ndarray): Vector of measured data.
        error_params (ndarray): A priori error variances/limits.
        params_ (ndarray): Known  model parameters.


    Returns:
        ndarray: Reconciled measured data after applying the Alpha Gram-Charlier method.
    """
    if msrd_data.ndim == 1:
        xNum = msrd_data.size
        n = 1
    elif msrd_data.ndim == 2:
        xNum = msrd_data.shape[0]
        n = msrd_data.shape[1]
    else:
        raise ValueError(
                "Only 2-dimensional arrays msrd_data are supported.")


    data_init_points = np.mean(msrd_data, axis=1)
    eqNum = len(eqies_model(data_init_points, params_))
    ineqNum = len(ineqies_model(data_init_points, params_))


    # Step 1: Compute l (a zero vector of the same length as eqies_model's output)
    l_eqies = np.zeros(len(eqies_model(data_init_points, params_)))
    l_ineqies = np.zeros(len(ineqies_model(data_init_points, params_)))
    l_ineqies_muly_by_l = np.zeros(len(ineqies_model(data_init_points, params_)))
```

```
    l = np.hstack((l_eqies, l_ineqies, l_ineqies_muly_by_l))

    # Step 2: Extend the initial guess (mu_and_l_start) by appending l to msrd_data
    mu_and_l_start = np.hstack((data_init_points, l))

    # Solve the system of equations using fsolve
    mu_and_l_res = fsolve(gauss_system_to_solve, mu_and_l_start, args=(eqies_model, ineqies_model, msrd_data, error_params, params_, xNum, n, eqNum, ineqNum))

    # Step 4: Extract the reconciled measured data (excluding 'l' values)
    reconciled_ = mu_and_l_res[:len(data_init_points)]

    return reconciled_
```

Пример использования функции `DRparamEqIneq()`

Рассмотрим пример, где у нас есть набор измеренных данных, которые мы хотим согласовать с учетом уравнений и неравенств.

Определение моделей уравнений и неравенств

Уравнение: линейная зависимость.

Неравенство: ограничение на значения данных.

```
def eqies_model(mu, params):
    """
    Уравнение: линейная зависимость y = a * x + b.
    Здесь params = [a, b].
    """
    a, b = params
    return a * mu + b


def ineqies_model(mu, params):
    """
    Неравенство: ограничение на значения данных (например, mu >= 0).
    """
    return mu - 0  # mu >= 0
```

Подготовка данных

Измеренные данные: 3 измерения, каждое измерение повторено 2 раза.

Априорные ошибки: заданы для каждого измерения.

Параметры модели: `a = 1, b = 0`.

```
import numpy as np

# Измеренные данные (3 измерения, каждое повторено 2 раза)
msrd_data = np.array([
    [1.1, 1.2],  # Измерение 1
    [2.0, 2.1],  # Измерение 2
    [3.0, 3.1]   # Измерение 3
])

# Априорные ошибки (дисперсии) для каждого измерения
error_params = np.array([0.1, 0.1, 0.1])

# Параметры модели (a и b для линейной модели)
params_ = np.array([1, 0])
```

Вызов функции `DRparamEqIneq()`

```
# Согласование данных
reconciled_data = DRparamEqIneq(eqies_model, ineqies_model, msrd_data, error_params, params_)

# Вывод результата
print("Согласованные данные:", reconciled_data)
```

Результат

Функция вернет вектор согласованных данных, которые лучше соответствуют уравнениям и неравенствам, а также учитывают априорные ошибки измерений.

Объяснение работы функции

Инициализация начальных значений:

Начальные значения для согласованных данных (mu) берутся как средние значения измерений.

Начальные значения для множителей Лагранжа (l) устанавливаются в нули.

Решение системы уравнений:

Функция `fsolve` из библиотеки `scipy.optimize` используется для решения системы уравнений, которая формируется в функции `gauss_system_to_solve`.

Система уравнений включает:

Уравнения для согласования данных с учетом априорных ошибок.

Уравнения, заданные моделью зависимости.

Неравенства, заданные моделью ограничений.

Извлечение результата:

После решения системы извлекаются согласованные данные (первые `xNum` элементов результата).

Примечания

Функция `gauss_system_to_solve` использует метод численного дифференцирования с мнимой возмущением для вычисления производных.

Если априорные ошибки (`error_params`) содержат нули, функция вызовет ошибку, так как это приведет к делению на ноль.

Функция поддерживает только двумерные массивы для `msrd_data`.

Функция `DRparamEqIneq`() полезна для задач, где необходимо согласовать измеренные данные с учетом как уравнений, так и неравенств. Она может быть адаптирована для различных моделей и типов данных.

Б.6 СПИСОК ФУНКЦИЙ И МЕТОДОВ

`estAccuracyIncreaseByDR()`	Функция выполняет приближенную оценку потенциального уточнения совместных измерений, достигаемого за счет учета известных функциональных взаимосвязей между измеряемыми величинами, на основе локальной линеаризации модели и метода декомпозиции алгоритма условной оптимизации.
`DRparamEq()`	Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей. Подлежат учету зависимости в виде равенств и неравенств.
`DRsemiparamEq()`	Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье. Подлежат учету зависимости в виде равенств и неравенств.
`DRnonparamEq()`	Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса. Подлежат учету зависимости в виде равенств и неравенств.
`DRparamEqRobust()`	Функция выполняет робастное параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей.
`DRsemiparamEqRobust()`	Файл содержит одноименную вызываемую функцию, которая выполняет робастное полу-непараметрическое согласование совместно измеренных величин, закон распределения

	которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье.
DRnonparamEqRobust()	Функция выполняет робастное непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса.
DRparamEqIneq()	Функция выполняет параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей. Подлежат учету зависимости в виде равенств и неравенств.
DRsemiparamEqIneq()	Файл содержит одноименную вызываемую функцию, которая выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье. Подлежат учету зависимости в виде равенств и неравенств.
DRnonparamEqIneq()	Функция выполняет непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса. Подлежат учету зависимости в виде равенств и неравенств.
DRparamEqIneqRobust()	Функция выполняет робастное параметрическое согласование совместно измеренных величин, закон распределения случайных погрешностей которых соответствует нормальному распределению вероятностей. Подлежат учету зависимости в виде равенств и неравенств.
DRsemiparamEqIneqRobust()	Файл содержит одноименную вызываемую функцию, которая выполняет робастное полу-непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается проекционным методом с применением в качестве модели усеченного ряда Грамма-Шарлье. Подлежат учету зависимости в виде равенств и неравенств.
DRnonparamEqIneqRobust()	Функция выполняет робастное непараметрическое согласование совместно измеренных величин, закон распределения которых оценивается методом ядерной аппроксимации с использованием ядра Гаусса. Подлежат учету зависимости в виде равенств и неравенств.
PBox()	Конструктор класса PBox для канонического представления неопределенности входных данных в форме области возможных значений функции распределения. Границы представляемой области дискретизированы и представлены кусочно-постоянными функциями.
Hist()	Конструктор класса Hist для представления неопределенности входных данных в форме интервальной гистограммы по Берлинту, высота полос которой представлена интервалом возможных значений.

DempsterShafer()	Конструктор класса DempsterShafer для представления неопределенности входных данных в форме структуры Демпстера-Шафера.
Fuzzy()	Конструктор класса Fuzzy для представления неопределенности входных данных в форме нечеткой переменной по Заде.
FuzzyInterval()	Конструктор класса FuzzyInterval для представления неопределенности входных данных в форме нечеткого интервала (допускающего в том числе представление классического интервала по арифметике Мура как частный вырожденный случай).
Sample()	Конструктор класса Sample для представления информации о входных данных в виде выборки значений результатов многократных измерений.
PBox2Hist()	Функция преобразования переменной типа PBox в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
PBox2DempsterShafer()	Функция преобразования переменной типа PBox в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
PBox2Fuzzy()	Функция преобразования переменной типа PBox в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
PBox2FuzzyInterval()	Функция преобразования переменной типа PBox в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Hist2PBox()	Функция преобразования переменной типа Hist в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Hist2DempsterShafer()	Функция преобразования переменной типа Hist в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Hist2Fuzzy()	Функция преобразования переменной типа Hist в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Hist2FuzzyInterval()	Функция преобразования переменной типа Hist в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
DempsterShafer2PBox()	Функция преобразования переменной типа DempsterShafer в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
DempsterShafer2Hist()	Функция преобразования переменной типа DempsterShafer в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
DempsterShafer2Fuzzy()	Функция преобразования переменной типа DempsterShafer в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).

DempsterShafer2 FuzzyInterval()	Функция преобразования переменной типа DempsterShafer в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Fuzzy2PBox()	Функция преобразования переменной типа Fuzzy в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Fuzzy2Hist()	Функция преобразования переменной типа Fuzzy в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Fuzzy2DempsterShafer()	Функция преобразования переменной типа Fuzzy в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Fuzzy2FuzzyInterval()	Функция преобразования переменной типа Fuzzy в переменную типа FuzzyInterval с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
FuzzyInterval2PBox()	Функция преобразования переменной типа FuzzyInterval в переменную типа PBox с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
FuzzyInterval2Hist()	Функция преобразования переменной типа FuzzyInterval в переменную типа Hist с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
FuzzyInterval2 DempsterShafer()	Функция преобразования переменной типа FuzzyInterval в переменную типа DempsterShafer с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
FuzzyInterval2Fuzzy()	Функция преобразования переменной типа FuzzyInterval в переменную типа Fuzzy с минимизацией потерь информации о неопределенности (квазиэквивалентное преобразование).
Sample2PBox()	Функция построения переменной типа PBox по выборке Sample значений одной из согласуемых величин.
Sample2Hist()	Функция построения переменной типа Hist по выборке Sample значений одной из согласуемых величин.
Sample2SempsterShafer()	Функция построения переменной типа DempsterShafer по выборке Sample значений одной из согласуемых величин.
Sample2Fuzzy()	Функция построения переменной типа Fuzzy по выборке Sample значений одной из согласуемых величин.
Sample2FuzzyInterval()	Функция построения переменной типа FuzzyInterval по выборке Sample значений одной из согласуемых величин.
plot()	Функция графического отображения экземпляров классов, выражающих неопределенность согласуемых значений (метод классов PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample).
getStd()	Функция оценки возможных значений среднеквадратического отклонения неточной величины, представленной в одном из форматов представления неопределенности (метод классов PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample).

AnalyticalModel()	Процедура выполнения согласования неточных величин в рамках аналитической модели с решением задачи Каруша-Куна-Таккера с применением теоремы Ефремова-Козлова.
GetUncertainty()	Процедура выполнения оценки неопределенности результата согласования по аналитической модели AnalyticalModel при представлении неопределенности исходных данных в одном из перечисленных форматов: PBox, Hist, DempsterShafer, Fuzzy, FuzzyInterval, Sample.