Программный модуль IntvalPy для работы с интервальными данными на языке Python

ВЫСТУПАЮЩИЙ: АНДРОСОВ АРТЁМ СТАНИСЛАВОВИЧ

АНДРОСОВ АРТЁМ СТАНИСЛАВОВИЧ
ФИЦ ИВТ

ШАРЫЙ СЕРГЕЙ ПЕТРОВИЧ
ФИЦ ИВТ, Д. Ф.-М. Н

Структура доклада

1. Мотивация работы и постановка проблемы;
2. Архитектура библиотеки;
3. Функциональность библиотеки:
   - Низкоуровневая функциональность;
   - Верхнеуровневая функциональность;
4. Количественный анализ.

Задача восстановления функциональной зависимости

$$y = f(x_1, x_2, \dots, x_m, \beta_1, \dots, \beta_l) = f(x, \beta)$$

x – экзогенные переменные

y – эндогенные переменные

f – модельная функция

β – модельные параметры

модельный риск = риск спецификации + риск оценки

- *риск спецификации* – выбранная модель не оптимально отражает реальную функциональную зависимость,
- *риск оценки* — при использовании оптимальной модели были получены неточные оценки параметров.

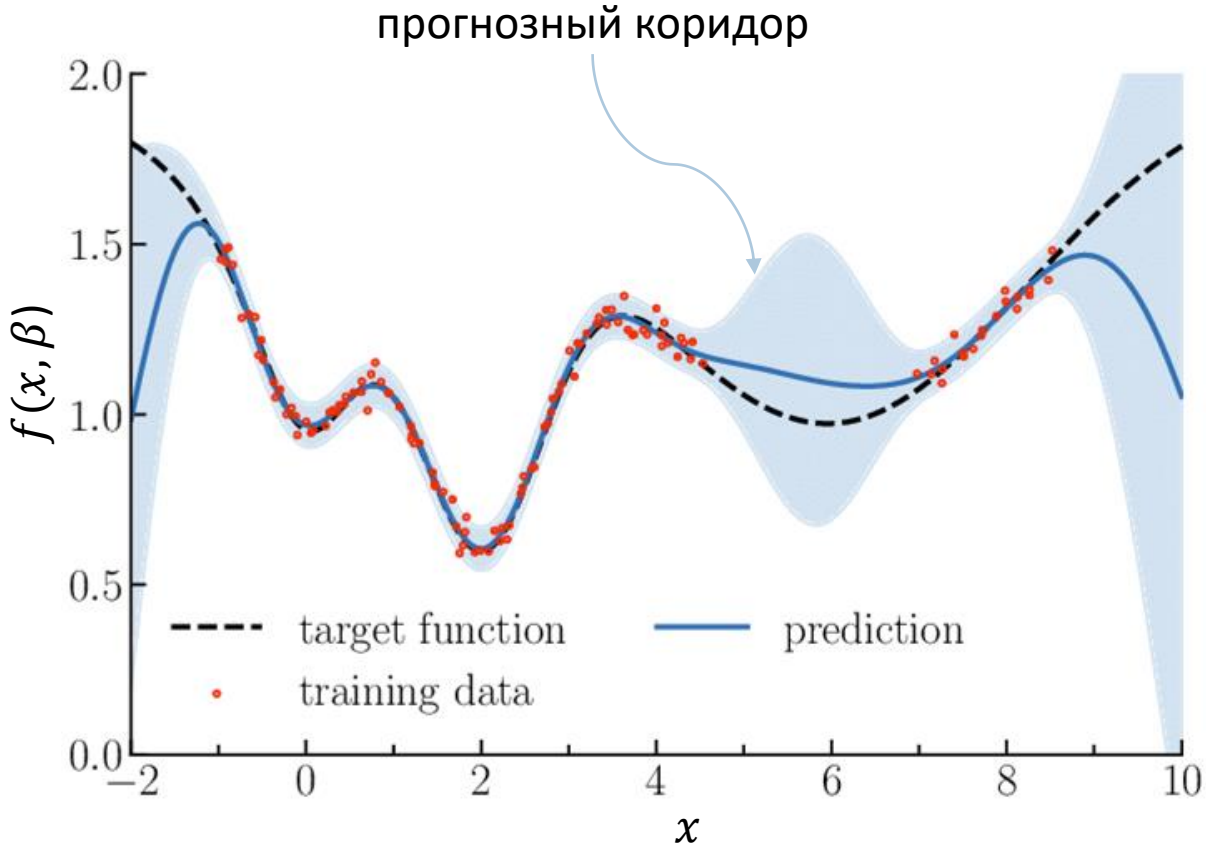


Рис. 1 – Сравнение прогнозного коридора и фактических данных

Области применения

Динамический баланс средств

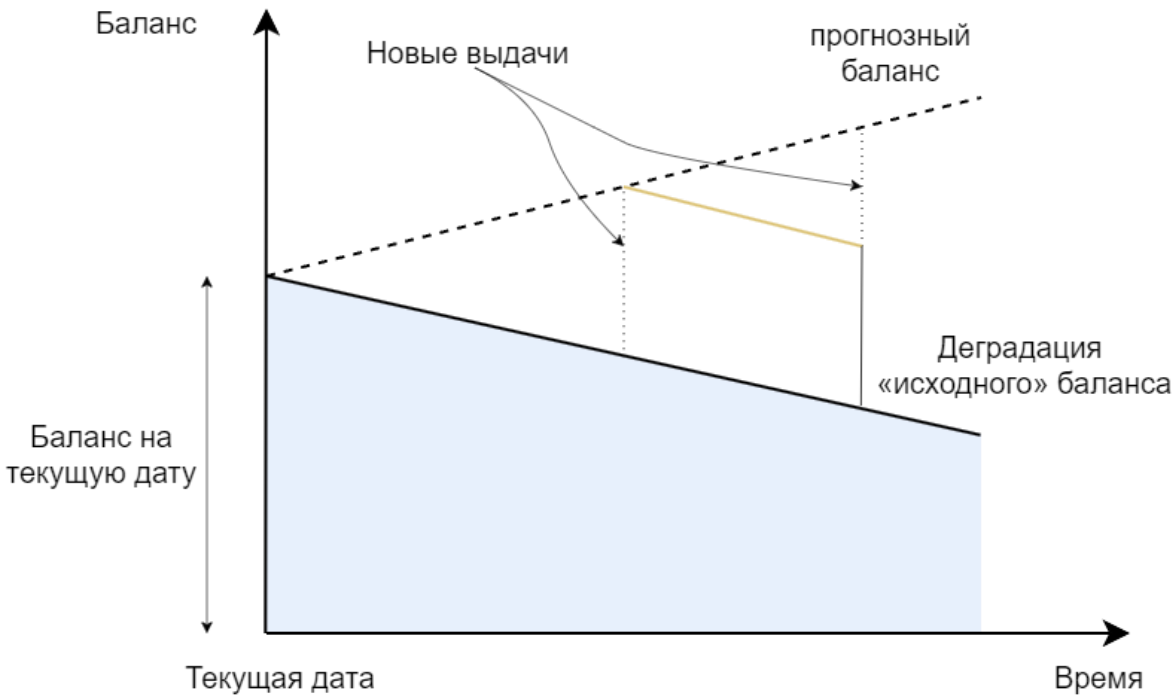


Рис. 2 - Схематическое представление динамики средств физических и юридических лиц

Задачи:

1. Спрогнозировать приток новых средств;
2. Построить кривую оттока уже имеющихся средств.

Предположение:

Приток и отток средств физ. и юр. лиц зависит от макроэкономических параметров. Например:

а) ключевая ставка;

б) средняя заработная плата.

Трудности при обучении и прогнозах:

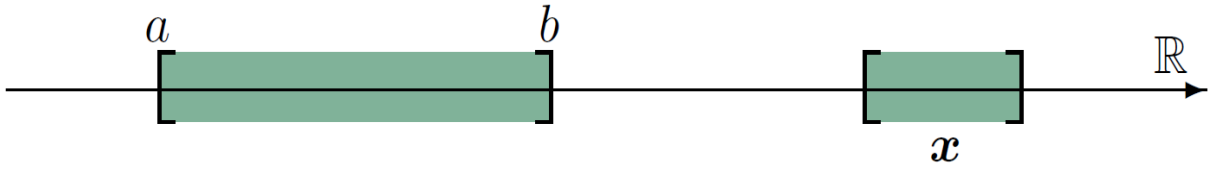
1. Агрегация данных;
2. Различные источники данных;
3. Недостаточный объём данных;
4. Используются признаки неизвестные в будущем.

Интервальный анализ

Интервалом $[a, b]$ вещественной оси R называется множество всех чисел между значениями a и b, включая их самих, т.е.

$$[a, b] := \{x \in \mathbb{R} \mid a \leq x \leq b\}.$$

При этом, a и b называются границами интервала $[a, b]$, левой (или нижней) и правой (или верхней), соответственно.



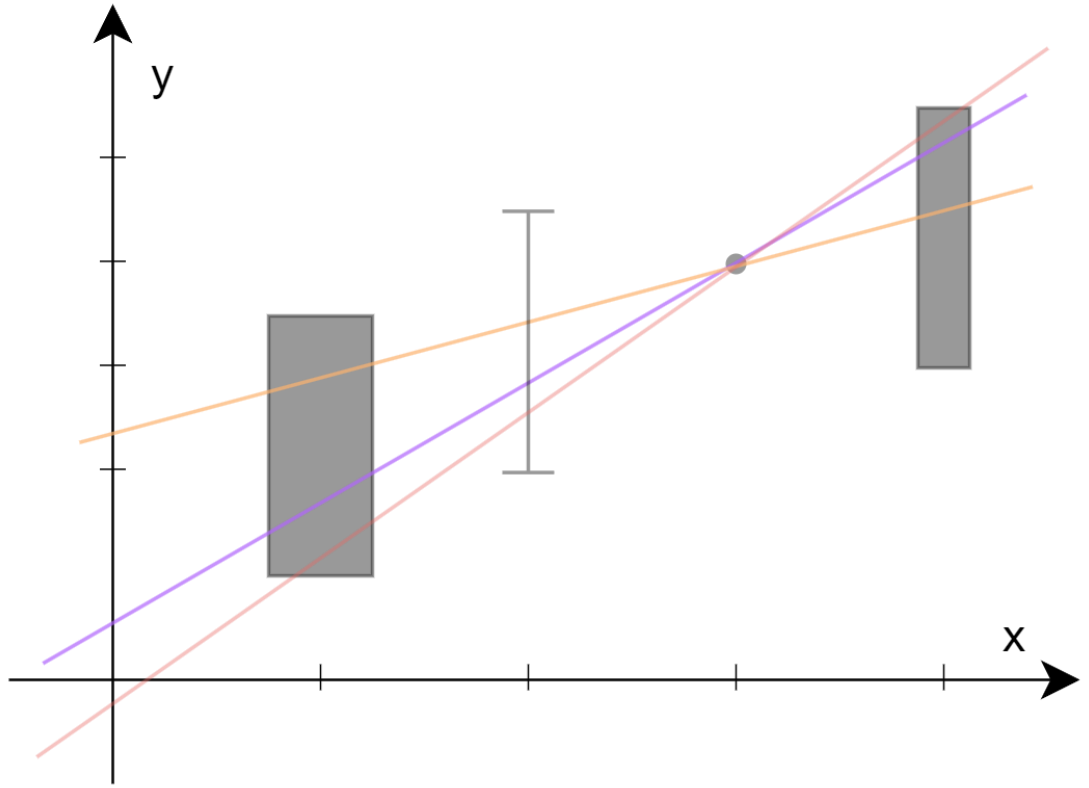


Рис. 3 – Восстановление линейной зависимости в данных с интервальной неопределённостью

Существующие программные модули

Цель: Найти или разработать программный модуль для решения различных математических задач с интервальной неопределённостью.

Основные требования:

1. Различные интервальные арифметики;
2. Матричные операции;
3. Распознавание разрешимости интервальных систем;
4. Визуализация данных и множеств решений;
5. Оценка решений для интервальных линейных и нелинейных систем;
6. Интервальная глобальная оптимизация.

Библиотека	Различные арифметики	Матричные операции	Визуализация	Линейные системы	Нелинейные системы	Оптимизация
IntvalPy	+	+	+	+	+	+
PyInterval	+	–	–	–	+	–
PyIbex	–	+	+	–	+	+
Octave	–	+	–	+	+	–
Wolfram Mathematica	+	+	–	+	–	–
Jinterval	+	+	+	+	+	+
JuliaIntervals	–	+	+	–	+	+
IntLab	+	+	–	+	+	+

Таблица 1 – Качественное сравнение существующих библиотек

Структура доклада

1. Мотивация работы и постановка проблемы;
2. **Архитектура библиотеки**;
3. Функциональность библиотеки:
   - Низкоуровневая функциональность;
   - Верхнеуровневая функциональность;
4. Количественный анализ.

Архитектура IntvalPy

Кодекс разработчика:

- Соблюдение лицензий сторонних библиотек;
- Удобство использования, документация;
- Соответствие общепринятому стандарту IEEE 1788-2015 для интервальных вычислений;
- Высокая производительность;
- Кроссплатформенность;
- Работа с различными видами интервальных арифметик;
- Гибкость к изменениям и расширению функциональности.

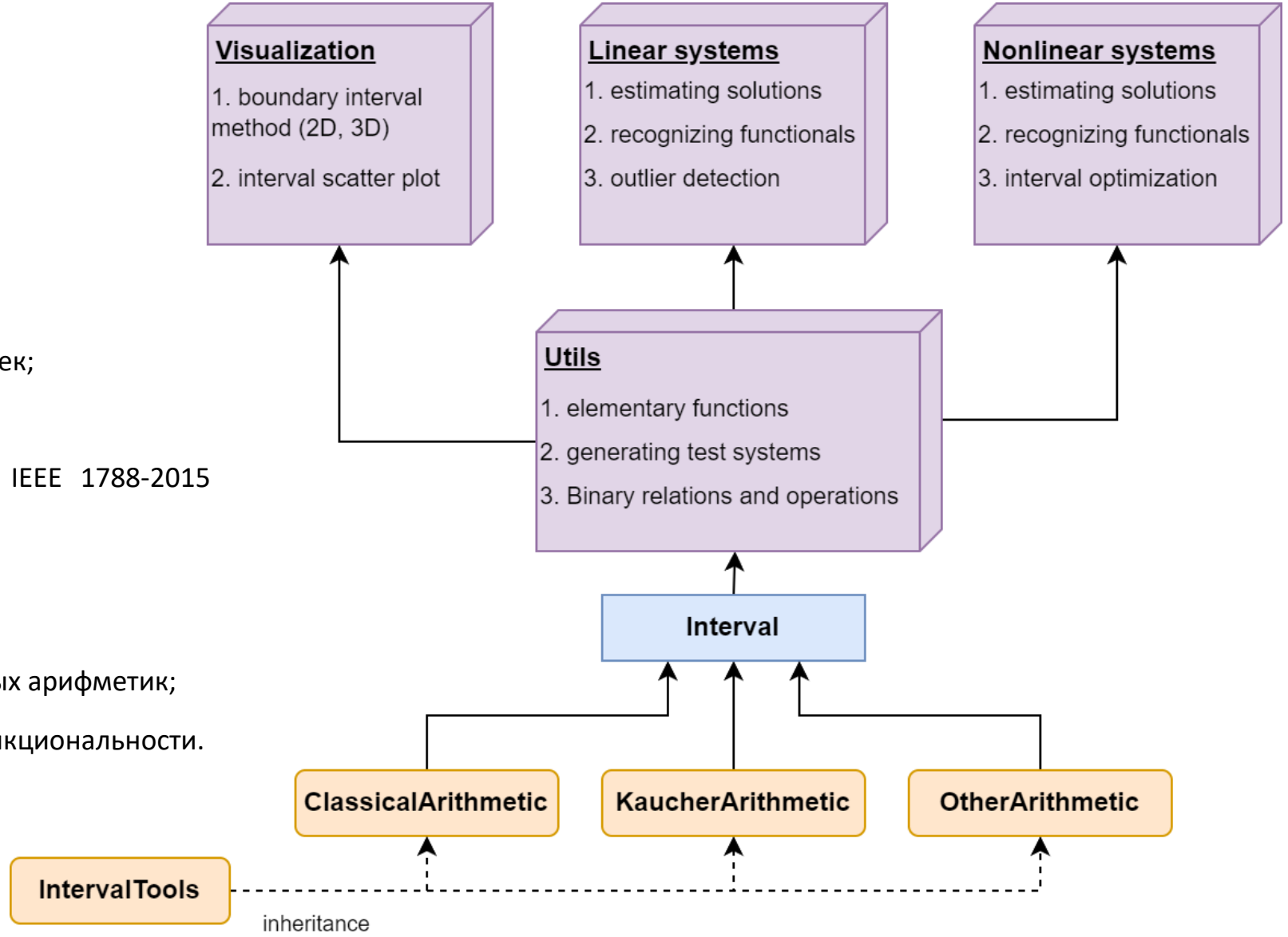


Рис. 4 – Архитектура библиотеки IntvalPy

Структура доклада

1. Мотивация работы и постановка проблемы;
2. Архитектура библиотеки;
3. **Функциональность библиотеки**:
   - Низкоуровневая функциональность;
   - Верхнеуровневая функциональность;
4. Количественный анализ.

Функциональность библиотеки

Низкоуровневая

Операция	Описание
$+ \quad - \quad \cdot \quad /$	Арифметические операции
pow, abs, exp, log, sin, cos	Элементарные функции
matmul, dot, transp	Матричные операции
$< \quad \leq \quad = \quad \geq \quad > \quad \subset \quad \subseteq$	Бинарные отношения
$\cap$	Теоретико-множественные операции
rad, wid, mid, mig, mag	Интервальные характеристики
dual, pro, opp, inv	Интервальные трансформации

Верхнеуровневая

Функция	Описание
Uni, Uss, Tol	Распознающие функционалы
PSS, PPS, Subdiff	
Gauss, Gauss-Seidel, HBR, Rohn	Решатели линейных систем
Hansen-Sengupta, Krawczyk	Решатели нелинейных систем
ralgb5, globopt	Алгоритмы оптимизации
Interval scatter plot, boundary interval method	Инструменты рисования

Нахождение «наилучших» параметров

Нахождение границ возможных значений параметров

Структура доклада

1. Мотивация работы и постановка проблемы;
2. Архитектура библиотеки;
3. Функциональность библиотеки:
   - **Низкоуровневая функциональность**;
   - Верхнеуровневая функциональность;
4. Количественный анализ.

Арифметические операции. Пример Румпа

Утверждение: Если вычисления дают схожие приближения с разной точностью, то это приближение не обязательно должно быть близко к истинному решению.

Функция, предложенная Румпом:

$$f(a,b) = 333.75 \cdot b^6 + a^2(11 \cdot a^2 b^2 - b^6 - 121 \cdot b^4 - 2) + 5.5 \cdot b^8 + \frac{a}{2b}$$

Результат вычисляется в точке $a = 77617, b = 33096$:

single precision	$f \approx 1.172603\ldots$
double precision	$f \approx 1.1726039400531\ldots$
extended precision	$f \approx 1.172603940053178\ldots$

истинное значение $f = -0.827396\ldots = \frac{a}{2b} - 2$

Пути решения проблемы

Направленные округления

Плюсы:

+ Доверительные вычисления
+ Скорость вычислений

Минусы:

– Получение бессмысленного результата

```
import pyibex as pb

f = lambda a, b: 333.75 * b**6 + a**2 * (11 * a**2 * b**2 - \
                b**6 - 121 * b**4 - 2) + 5.5 * b**8 + a/(2*b)

a = pb.Interval(77617, 77617)
b = pb.Interval(33096, 33096)
f(a, b)
```

```
[-4.72237e+21, 3.54177e+21]
```

Повышенная точность

Плюсы:

+ Осмысленный результат при достаточной точности

Минусы:

– Недоверительные вычисления
– Длительные вычисления

```
import intvalpy as ip

ip.precision.extendedPrecisionQ = True
ip.precision.dps(40)

f = lambda a, b: 333.75 * b**6 + a**2 * (11 * a**2 * b**2 - \
                b**6 - 121 * b**4 - 2) + 5.5 * b**8 + a/(2*b)

a = ip.Interval(77617, 77617)
b = ip.Interval(33096, 33096)
f(a, b)
```

```
'[-0.827396, -0.827396]'
```

Пути решения проблемы

Направленные округления

Плюсы:

+ Доверительные вычисления
+ Скорость вычислений

Минусы:

– Получение бессмысленного результата

При условии дальнейшей интеграции с C++, где будет реализован «гибридный» подход

Повышенная точность

Плюсы:

+ Осмысленный результат при достаточной точности

Минусы:

– Недоверительные вычисления
– Длительные вычисления

```
import pyibex as pb

f = lambda a, b: 333.75 * b**6 + a**2 * (11 * a**2 * b**2 - \
                b**6 - 121 * b**4 - 2) + 5.5 * b**8 + a/(2*b)

a = pb.Interval(77617, 77617)
b = pb.Interval(33096, 33096)
f(a, b)
```

```
[-4.72237e+21, 3.54177e+21]
```

```
import intvalpy as ip

ip.precision.extendedPrecisionQ = True
ip.precision.dps(40)

f = lambda a, b: 333.75 * b**6 + a**2 * (11 * a**2 * b**2 - \
                b**6 - 121 * b**4 - 2) + 5.5 * b**8 + a/(2*b)

a = ip.Interval(77617, 77617)
b = ip.Interval(33096, 33096)
f(a, b)
```

```
'[-0.827396, -0.827396]'
```

Структура доклада

1. Мотивация работы и постановка проблемы;
2. Архитектура библиотеки;
3. Функциональность библиотеки:
   - Низкоуровневая функциональность;
   - **Верхнеуровневая функциональность**;
4. Количественный анализ.

Множества решений

Функциональная зависимость: $y = \beta_0 x + \beta_1$

x	1	2	3
y	1	1.5	2

x	[0.6, 1.4]	[1.6, 2.4]	[2.7, 3.5]
y	[0.7, 1.5]	[1, 2]	[1.5, 2.5]

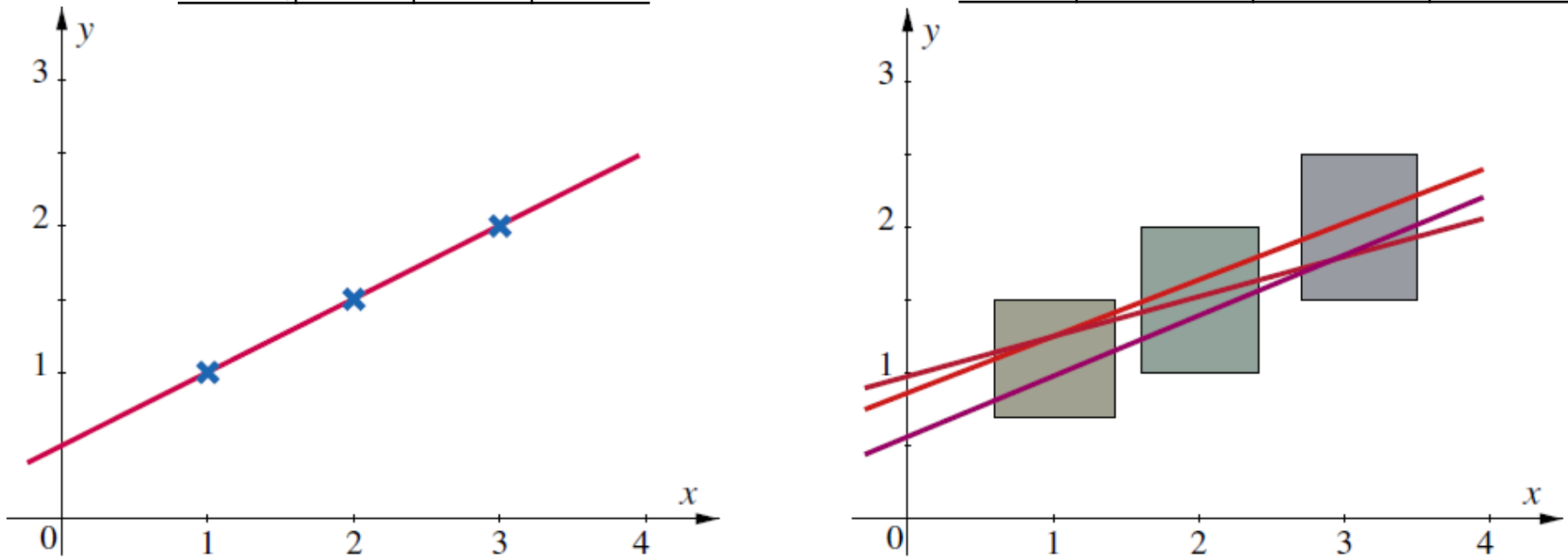


Рис. 4 – Восстановление линейной функциональной зависимости

Объединённое множество решений: $\Xi_{uni}(\mathbf{X}, \mathbf{y}) = \{ \beta \in \mathbb{R}^m \mid (\exists X \in \boldsymbol{X})(\exists y \in \boldsymbol{y})(\mathrm{X}\beta = y) \}$

Допусковое множество решений: $\Xi_{tol}(\mathbf{X}, \mathbf{y}) = \{ \beta \in \mathbb{R}^m \mid (\forall X \in \boldsymbol{X})(\exists y \in \boldsymbol{y})(\mathrm{X}\beta = y) \}$

Распознающие функционалы

Зачем нужны: распознавание непустоты множества решений, а также нахождения наиболее «репрезентативной» точки из этого множества.

```
data = {
    'const': [1, 1, 1],
    'x': ip.Interval([[0.6, 1.4], [1.6, 2.4], [2.7, 3.5]]),
    'y': ip.Interval([[0.7, 1.5], [1, 2], [1.5, 2.5]])
}
df = pd.DataFrame(data)
df
```

	const	x	y
0	1	'[0.6, 1.4]'	'[0.7, 1.5]'
1	1	'[1.6, 2.4]'	'[1, 2]'
2	1	'[2.7, 3.5]'	'[1.5, 2.5]'

```
features = ['const', 'x']
target = 'y'

model_tol = ip.WILS()
model_tol.fit(df[features], df[target], rec_func='Tol')
model_tol.estimator
```

```
(array([0.71904762, 0.38095238]), 0.2476190476188148, 74, 109, 3)
```

```
x = np.linspace(0, 4, 1000)
const = [1]*len(x)
x_test = pd.DataFrame(
    data = np.array([const, x]).T,
    columns = features
)

y_pred_tol = model_tol.predict(x_test[features])
```

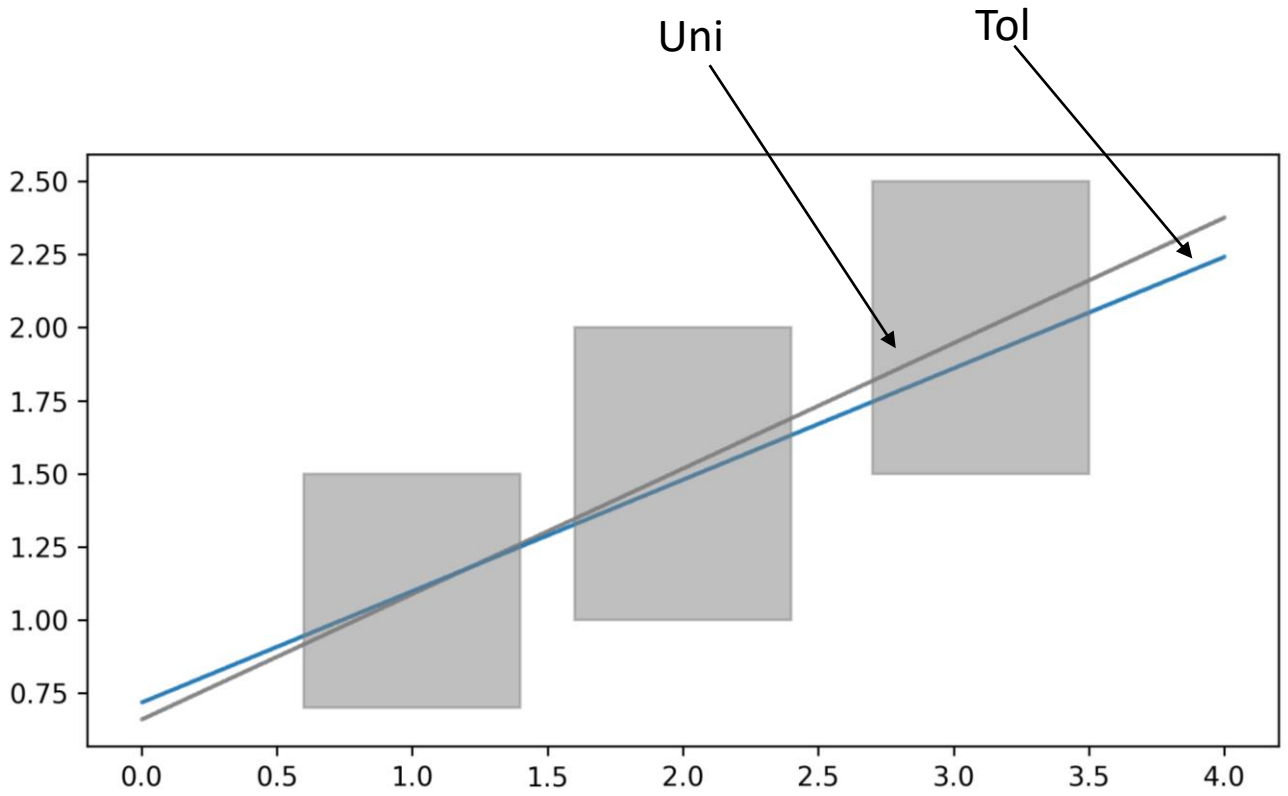


Рис. 5 – Восстановленная линейная зависимость.

Методы для получения оптимальных оценок

Объединённое множество решений: PPS, PSS

Вычисления для алгоритма lippsr (Matlab) производились на
PC Intel(R) Core 2 Duo CPU @ 1.6GHz, 1 GB RAM

Вычисления для алгоритмов PPS и PSS производились на
PC Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz, 16 GB RAM

Допусковое множество решений: Rohn

Формальные решения: Subdiff

Параметры и размерность интервальной системы		**Время работы алгоритма, с**		
		linppsr (Matlab)	**PPS**	**PSS**
Интервальная система Ноймайера				
n=5	θ=10	8.012	0.236	2.465
	θ=20	1.642	0.179	0.475
	θ=30	1.641	0.183	0.154
n=10	θ=25	617.328	293.920	206.983
	θ=30	197.024	5.598	32.120
	θ=45	51.422	5.661	31.625
	θ=60	31.065	5.805	33.649
n=12	θ=30	2943.612	2508.253	-
	θ=50	246.804	11.058	41.546
	θ=70	99.308	11.198	42.348
	θ=90	57.049	11.682	43.486
Интервальная система Шарого				
α=0.4, β=0.6	n=10, N=15	45.528	39.247	0.307
	n=20, N=25	882.344	2109.701	2.025
	n=30, N=35	5483.106	-	9.401
α=0.6, β=0.8	n=10, N=15	17.477	5.562	0.322
	n=20, N=25	209.250	43.169	1.981
	n=30, N=35	969.280	1072.956	164.335

Метод граничных интервалов

Наименование функций:

lineqs – линейные неравенства, 2D

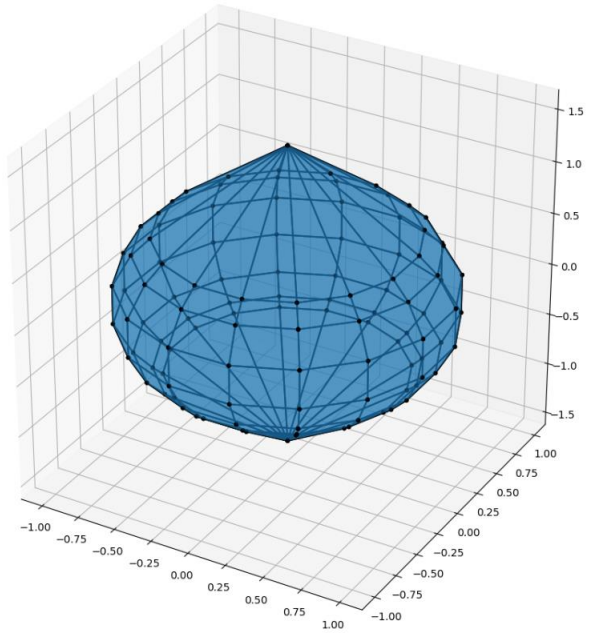
IntLinIncR2 – интервальные линейные равенства, 2D

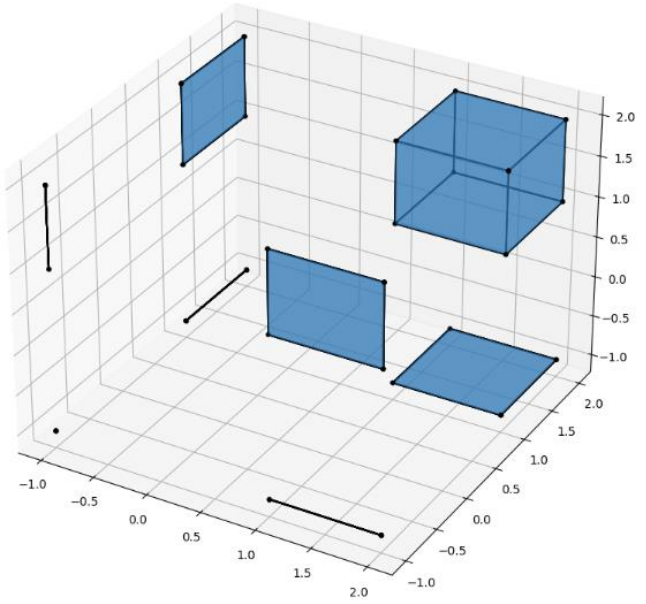
lineqs3D – линейные неравенства, 3D

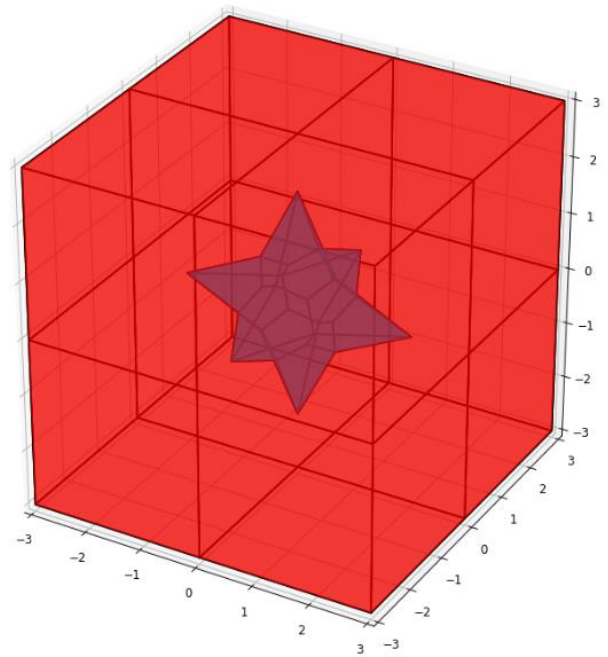
IntLinIncR3 – интервальные линейные равенства, 3D

Преимущества:

1. работа с неограниченными множествами;
2. работа с «тощими» множествами.







Структура доклада

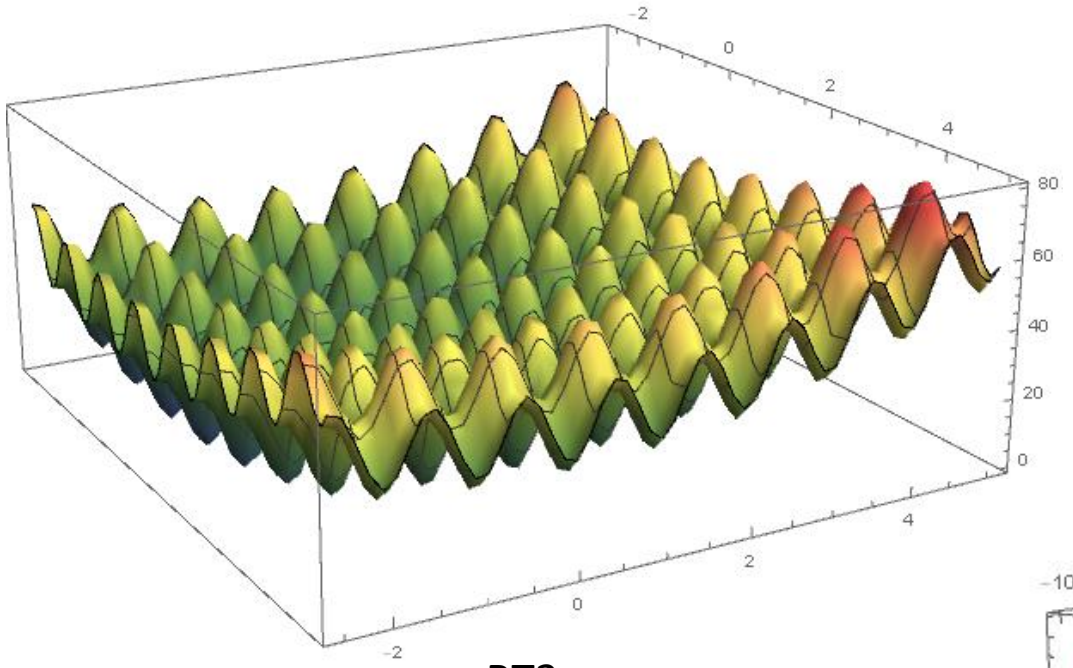
1. Мотивация работы и постановка проблемы;
2. Архитектура библиотеки;
3. Функциональность библиотеки:
   - Низкоуровневая функциональность;
   - Верхнеуровневая функциональность;
4. **Количественный анализ**.

Количественный анализ

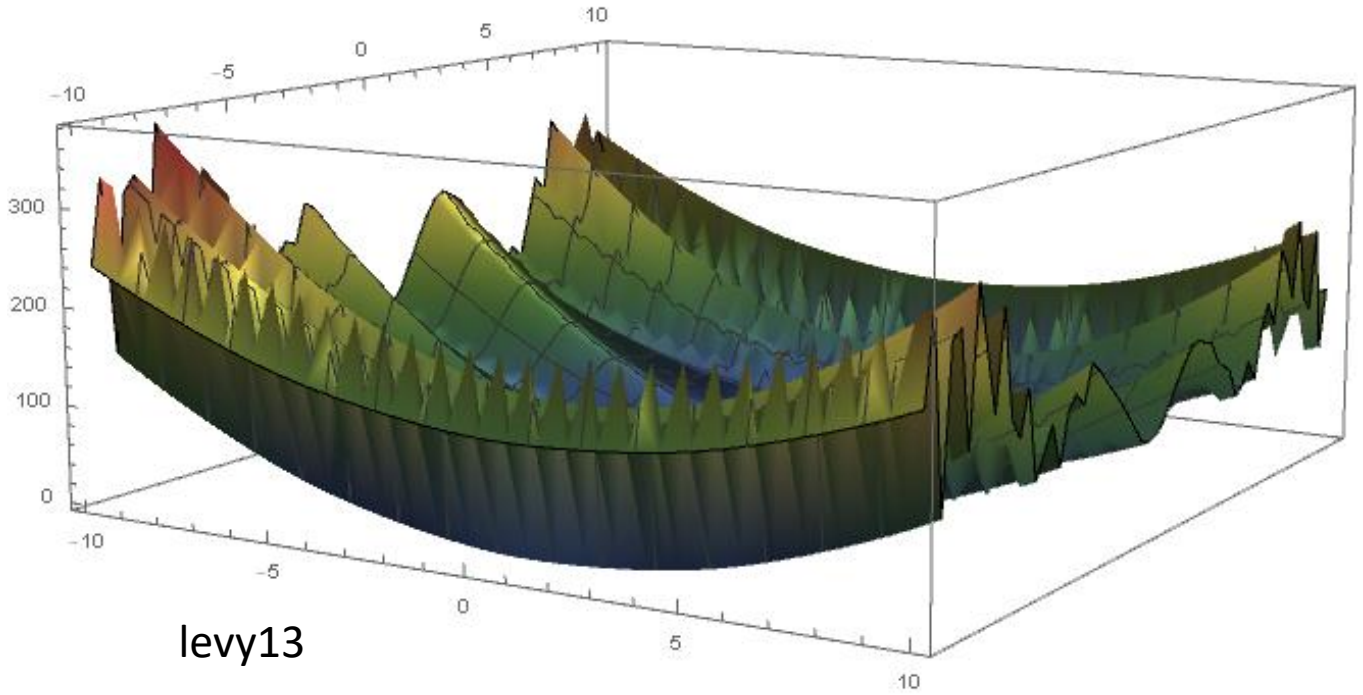
		PyInterval			PyIbex			IntvalPy			
F	dim	N_{it}	ϵ	t_{cpu}	N_{it}	ϵ	t_{cpu}	N_{it}	ϵ	t_{cpu}	
										double	ext.
Branin (RC)	2	160	7.1^{-15}	0.301	160	8.88^{-15}	0.047	142	8.90^{-15}	0.107	0.517
Bohachevsky (B_m)	2	66	6.66^{-15}	0.149	66	6.66^{-15}	0.016	66	6.44^{-15}	0.061	0.278
Beale (BL)	2	342	5.66^{-15}	0.833	342	5.66^{-15}	0.087	342	5.66^{-15}	0.373	1.637
Booth (BO)	2	146	7.90^{-15}	0.214	146	7.90^{-15}	0.035	146	7.90^{-15}	0.079	0.316
De Joung (DJ)	3	80	8.73^{-15}	0.070	80	8.73^{-15}	0.011	80	8.73^{-15}	0.029	0.157
Easom (ES)	2	63	8.10^{-15}	0.116	63	8.55^{-15}	0.016	70	4.44^{-15}	0.061	0.217
Eggholder	2	10000	1.06^{-5}	246.7	10000	1.06^{-5}	5.450	10000	1.06^{-5}	15.94	57.32
Rastrigin (RT_n)	4	10000	4.26^{-14}	42.73	10000	4.26^{-14}	78.03	129	7.11^{-15}	0.311	1.137
Schaffer № 4	2	10000	2.83^{-7}	27.30	10000	2.83^{-7}	3.897	10000	2.83^{-7}	11.57	68.18
Levy № 13	2	10000	1.70^{-14}	30.57	10000	3.40^{-14}	4.085	68	8.57^{-15}	0.097	0.468

F – целевая функция, dim – размерность задачи, N_{it} - ограниченное количество итераций для остановки оптимизации, ϵ – минимальная допустимая погрешность, t_{cpu} – усреднённое 50-ю запусками время выполнения задачи, double – двойная точность, extended – повышенная точность.

Визуализация функций



RT2



levy13

Документация и немного статистики

Python

Навигация

Installation
Using intervals
General purpose functions
Recognizing functionals
Interval linear systems

Быстрый поиск

Иска

IntvalPy

Welcome to the IntvalPy documentation! This Python module implements an algebraically closed system for working with intervals, and provides the ability to work with both classical interval arithmetic and full Kaucher interval arithmetic. The top-level functionality of the IntvalPy library implements the latest methods for recognizing and evaluating sets of solutions to interval linear systems of equations, calculating their formal solutions, and visualizing sets of solutions to interval equations and systems of equations.

Contents

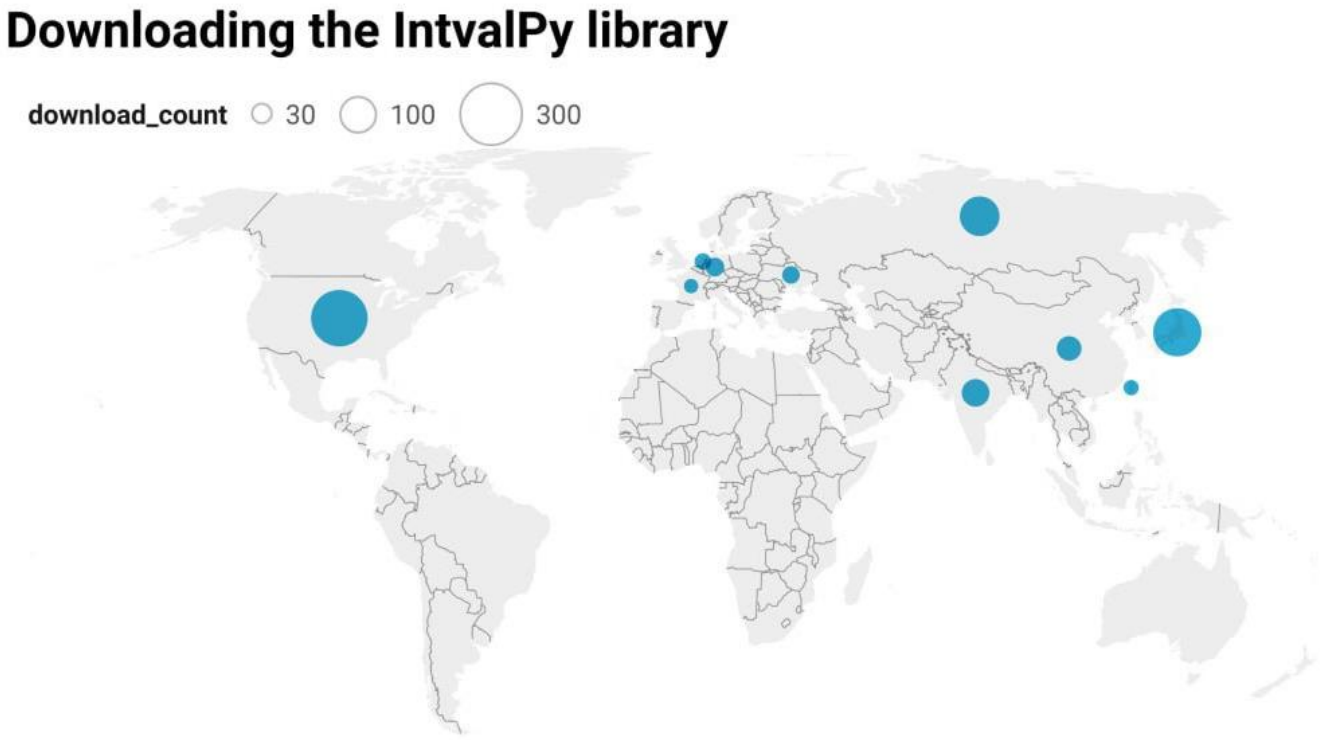


Рис. 6 – Количество скачиваний библиотеки IntvalPy за 2023 год

Спасибо за внимание!