**Описание алгоритма восстановления правой части СДУ**

**(v.1 20.07.2021)**

Раздел 1. Формирование исходных данных

В данном разделе описано получение исходных данных для задач тестирования алгоритма восстановления матриц полиномиальной правой части СДУ на примерах с заранее известной правой частью СДУ. Если исходные данные получены из имеющегося набора данных, то этот раздел опускается.

Для дефлектора [функция get\_training\_data\_deflector(x0, y0, t0, t1, step, reduction\_coeff)]

* 1. Формирование эталонного решения СДУ с заданными начальными условиями на заданном отрезке времени.

Дано: размерность системы n;

матрицы полиномиальной правой части

СДУ ;

начальные условия ;

конечное значение времени ;

Параметры: *step* - шаг сетки на котором рассчитывается решение или количество узлов сетки ;

Алгоритм: 1) расчёт количества узлов сетки

2) построение сетки

3) численное решение СДУ с использованием библиотечной функции

[функция scipy.integrate.odeint(right\_hand\_side, [, ], grid,parameters)]

Выход: сетка ;

сеточные функции

* 1. Формирование исходного набора данных  
     Исходный набор данных формируется из эталонного решения – набора сеточных функций на сетке , путём удаления всех значений, кроме каждого *r*-го, т.е. мы рассматриваем эталонное решение на более редкой подсетке

Дано: сетка ;

сеточные функции ;

Параметры: коэффициент урежения сетки *r=* reduction\_coeff

Алгоритм: 1) формирование уреженной сетки

2) формирование значений сеточных функций на уреженной сетке

[функция [функция get\_reduction\_curve(curve, reduction\_coeff)]

Выход: сетка ;

сеточные функции

Раздел 2. Формирование набора тренировочных данных для восстановления матриц полиномиальной правой части СДУ

* 1. Дополнение исходного набора данных линейными данными для получения замкнутой кривой

Дано: сетка с шагом ;

сеточные функции

Параметры: close\_coeff>1 – коэффициент увеличения отрезка времени

Алгоритм: 1) расчёт конечного значения времени ;

2) формирование сетки , ;

3) расчёт сеточных функций по формулам

, где коэффициенты и определяются из условий

[функция get\_linear\_addition(train\_data, close\_coeff)]

Выход: сетка ;

сеточные функции

* 1. Построение сетки для формирования коэффициентов тригонометрических полиномов и нахождение значений сеточных функций на этой сетке

Дано: сетка ;

сеточные функции

Параметры: количество точек для построения тригонометрического полинома -

Алгоритм: 1) построение из сетки уреженной сетки для построения которой мы берем каждый m-й узел, где /;

2) отбор значений сеточных функций на более редкой сетке – формирование на данной сетке сеточных функций

[функция get\_grid\_func(f, t, nz)]

Выход: сетка ;

сеточные функции

* 1. Построение сетки расчёта, на которой вычисляются значения тригонометрического полинома

Дано: t0 – левый край сетки – начальное значение;

Параметры: n\_step\_new – число шагов сетки;

step\_new – шаг сетки;

Алгоритм: 1) построение сетки с шагом ;

[функция get\_grid(t0, n\_step\_new, step\_new)]

Выход: сетка ;

* 1. Построение контрольно-расчётной сетки, на которой вычисляется линейная аппроксимация тренировочных данных для контроля качества тригонометрической интерполяции.

Эта сетка формируется из сетки .

Дано: сетка ;

Параметры: number - число шагов формируемой сетки ;

stride – количество узлов исходной сетки которые пропускаются для построения сетки

Алгоритм: 1) построение сетки с количеством узлов *number*, путём отбора каждого *m*-го узла сетки , *m=stride*;

[get\_grid\_stride(t\_new, number, stride)]

Выход: сетка ;

* 1. Построение тригонометрических полиномов для аппроксимации сеточных функций – определение коэффициентов полиномов и расчёт значений полиномов на двух сетках (сетке расчёта и сетке контроля)

Дано: сетка ;

сетка расчёта ;

сетка контроля ;

сеточные функции на сетке ;

Параметры: нет

Алгоритм:

1) нахождение коэффициентов тригонометрического полинома по формулам

,

, ,

,.

2) расчёт значений тригонометрических полиномов на сетке расчёта и сетке контроля

, *z* – узел сетки

В случае получим сеточные функции , в случае получим сеточные функции ,

[функция trigonometric\_approximation(f, length, t, tt)]

Выход: сеточная функция значений тригонометрического полинома на сетке ;

сеточная функция значений тригонометрического полинома на сетке ;

* 1. Удаление участка аппроксимированной кривой, соответствующего добавленному линейному участку для замыкания двумерной кривой (тренировочных данных)

Дано: сетка расчёта ;

сеточная функция значений тригонометрического полинома на сетке ;

сеточная функция значений тригонометрического полинома на сетке ;

– скалярное значение времени, начиная с которого удаляются узлы сетки и соответствующие значения сеточной функции

Параметры: нет

Алгоритм: 1) формирование укороченной сетки из сетки путём удаления всех узлов начиная со значения ;

2) формирование сеточной функции на сетке путём удаления значений на отсутствующих узлах;

3) формирование сеточной функции на сетке путём удаления значений на отсутствующих узлах;

del\_linear(t\_new, pol\_x, pol\_y, t1)

Выход: сетка расчёта ;

сеточная функции на сетке

сеточная функции на сетке

* 1. Формирование линейной аппроксимации тренировочных данных на расчётно-контрольной сетке

Дано: сетка контроля ;

сетка ;

сеточные функции

сетка контроля ;

Параметры: нет

Алгоритм: 1) линейная аппроксимация значений сеточных функций на сетку контроля по алгоритму п.п. 3 п. 2.1

[функция linear\_approximation(x\_train, y\_train, t\_train, t\_error)]

Выход: сеточная функции на сетке

* 1. Расчёт ошибок (суммарной и абсолютной) аппроксимации – вычисление суммарной и максимальной модулей разности двух двумерных кривых (абсцисс и ординат), заданных на одной сетке.

Дано: сеточные функции

сеточные функции ,

Параметры: нет

Алгоритм: 1) вычисление суммы ошибок ;

2) вычисление максимальной ошибки

[функция errors\_calculation(pol\_x\_error, pol\_y\_error, z\_x\_linear, z\_y\_linear)]

Выход: error – cуммарная ошибка;

max\_error – максимальная ошибка

Раздел 3. Восстановление матриц полиномиальной правой части СДУ

[функция ode\_approximation\_calculation(curve, t\_new\_1, number, stride)]

* 1. Восстановление матриц полиномиальной правой части СДУ 2 уравнения 2 неизвестных, 3 степень полиномиальной правой части

Дана СДУ

Перепишем её в виде двух уравнений

Введём обозначения

Для нахождения неизвестных коэффициентов матриц A, B и С необходимо решить две системы линейных алгебраических уравнений

каждая из которых имеет 9 неизвестных и 9 уравнений. Правые части этих систем вычисляются путём замены производной на конечную разность.

Дано: сеточные функции значений тригонометрического полинома на сетке ;

сетка расчёта ;

number – число шагов формируемой сетки ;

stride – количество узлов исходной сетки которые пропускаются для построения сетки

Параметры: нет

Алгоритм: 1) формирование матрицы D на сетке (формируется из сетки аналогично п. 2.4)

[функция matrix\_23(z, stride)]

2)формирование правых частей СЛАУ и на сетке (формируется из сетки по сетке по одному узлу из сетки слева и справа от каждого узла сетки из сетки , узлы сетки из сетки в сетку из сетки не включаются). Правые части СЛАУ и рассчитываются как центральные разностные производные

[функция right\_hand\_vector\_calculation(z[:, 0], t, number, stride)]

3) решение двух СЛАУ

с использованием библиотечной функции numpy.linalg.solve(matrix, v)

4) формирование из векторов решений и матриц A, B и С

[функция right\_hand\_23\_calculation(p1, p2)]

Выход: матрицы A, B и С

* 1. Проверка качества восстановления правой части.

1. Расчёт численного решения СДУ с исходной правой частью на некоторой сетке
2. Расчёт численного решения СДУ с восстановленной правой частью на той же сетке
3. Расчёт ошибок