МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Санкт–Петербургский государственный университет  
аэрокосмического приборостроения»

Кафедра №43 «Компьютерных технологий и программной инженерии»

ОТЧЁТ ПО ПРАКТИКЕ

ЗАЩИЩЁН С ОЦЕНКОЙ

Руководитель

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ст. преподаватель |  |  |  | М.Д.Поляк |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЁТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
| вид практики | Производственная | |
| тип практики | Научно-исследовательская работа | |
| на тему индивидуального задания | | Решение модели с АКАР управлением по методу Эйлера |
|  | | | |
|  | | | |

|  |  |
| --- | --- |
| выполнен | Петровой Екатериной Дмитриевной |
| фамилия, имя, отчество обучающегося в творительном падеже | |

|  |  |  |  |
| --- | --- | --- | --- |
| по направлению подготовки | 09.03.04 |  | Программная инженерия |
|  | код |  | наименование направления |
|  | | | |
| наименование направления | | | |
| направленности |  |  |  |
|  | код |  | наименование направленности |
|  | | | |
| наименование направленности | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Обучающаяся группы № | 4932 |  |  |  | Е.Д.Петрова |
|  | номер |  | подпись, дата |  | инициалы, фамилия |

Санкт–Петербург 2022

Оглавление

[**1.** **Задание** 3](#_Toc109084256)

[**2.** **Описание решения.** 3](#_Toc109084257)

[**3.** **Полученные результаты** 5](#_Toc109084258)

[**4.** **Выводы** 6](#_Toc109084259)

[**5.** **Приложение с исходным кодом** 6](#_Toc109084260)

1. **Задание**

Написать универсальную функцию для Matlab, которая позволяет численно решить любую систему уравнений методом Эйлера с управлением, заданным по методу АКАР.

Возвращать эта функция должна вектор из двух векторов, в котором:

* первый вектор будет хранить значения времени, на котором производилось интегрирование системы дифференциальных уравнений, с шагом, заданным пользователем
* второй вектор будет хранить результаты интегрирования каждого шага.

1. **Описание решения.**

В основе задания лежит метод Эйлера – это простейший численный метод решения систем обыкновенных дифференциальных уравнений. Впервые описан Леонардом Эйлером в 1768 году в работе «Интегральное исчисление». Метод Эйлера является явным, одношаговым методом первого порядка точности. Он основан на аппроксимации интегральной кривой кусочно-линейной функцией, так называемой ломаной Эйлера.

Формула Эйлера:

где

Формула Эйлера с АКАР-управлением:

calc(@calc\_f, @calc\_u, min\_max, init\_vals, dt) - универсальная функция, которая позволяет численно решить любую систему уравнений методом Эйлера с управлением, заданным по методу АКАР.

В этой функции реализован цикл, выполняющий пошаговую обработку входных данных, в нем вычисляются значения векторов f, u и x.

Входные аргументы:

* calc\_f(t,x)

Входные аргументы:

t – время

х – вектор, хранящий значения xi, где i – номер переменной из системы дифференциальных уравнений.

Выходные аргументы:

Вектор, хранящий значения fi, , где i – номер дифференциального уравнения.

Пример:

alfa1 = 0.8;

alfa2 = 0.7;

beta1 = 0.1;

beta2 = 0.3;

vector(1) = alfa1 \* x(1) - beta1 \* x(1) \* x(2);

vector(2) = - alfa2 \* x(2) + beta2 \* x(1) \* x(2);

output = vector';

* calc\_u(t,x)

Входные аргументы:

t – время

х – вектор, хранящий значения xi, где i – номер переменной из системы дифференциальных уравнений.

Выходные аргументы:

Вектор, хранящий значения ui, , где i – номер дифференциального уравнений, для которого будет применяться формула управления.

Пример:

alfa1 = 0.8;

alfa2 = 0.7;

beta1 = 0.1;

beta2 = 0.3;

T1=0.9;

T2 = 1;

x2const = 3;

phi = (alfa2\*x(2) - (1/T2) \* (x(2) - x2const))/(beta2 \* x(2));

f1 = alfa1 \* x(1) - beta1 \* x(1) \* x(2);

f2 = - alfa2 \* x(2) + beta2 \* x(1) \* x(2);

phi2 = (x2const / (T2 \* beta2 \* x(2) \* x(2))) \* f2;

e = - 1/T1 \* (x(1) - phi) - f1 + phi2;

u(1) = e;

u(2) = 0;

output = u';

* min\_max – вектор, хранящий два значения - отрезок, на котором будут проводиться вычисления.

Пример:

[0 1]

* init\_vals – вектор, хранящий начальные значения переменной xi, где i – номер переменной из системы дифференциальных уравнений.

Пример:

[10 7]

* dt – шаг, с которым будут проводиться вычисления.

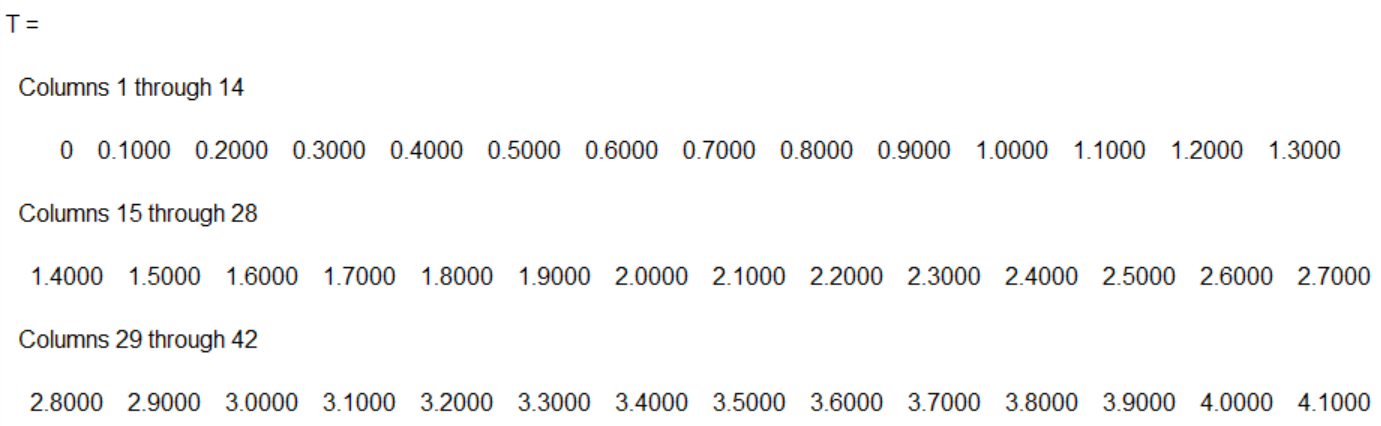
Пример: 0.1

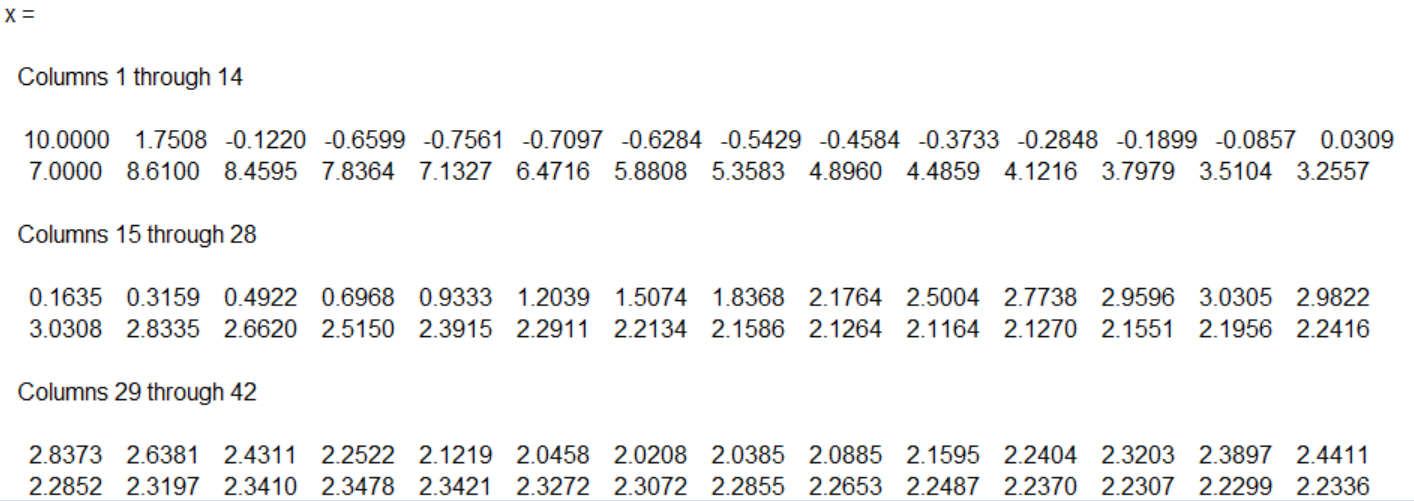
1. **Полученные результаты**

Заполнив calc\_f и calc\_u, из командного окна пользователю необходимо вызвать функцию calc следующим образом (входные аргументы будут использованы из примеров, приведенных выше):

[T, x] = calc(@calc\_f, @calc\_u, [0 10], [10 7], 0.1)

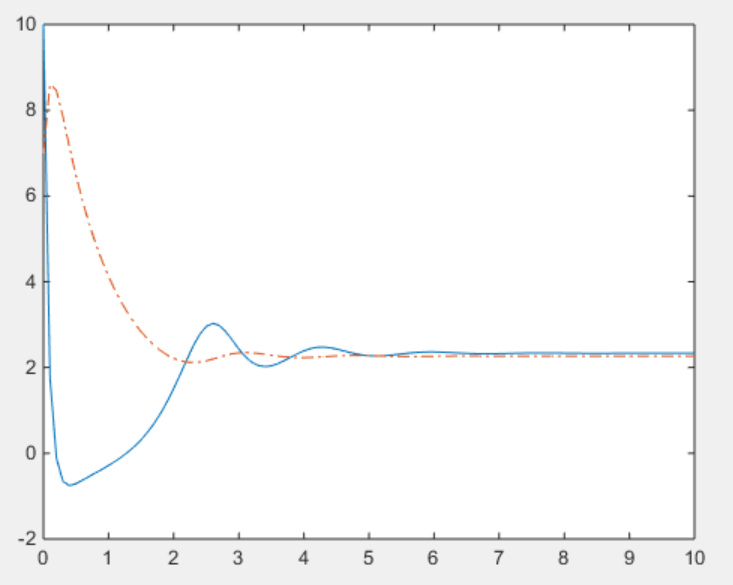
Функция вернет следующие значения:





Теперь можем построить по ним график следующим образом:

plot(T, x(1,:), '-',T, x(2,:),'-.')



В данном примере мы рассматривали модель «Хищник-жертва» с аддитивным управлением по жертвам.

Цель данной системы имеет вид:

где

То есть должно стремиться к 3.

На графике мы можем наблюдать, как оранжевая пунктирная линия с течением времени стремится в 3.

1. **Выводы**

В результате проделанной работы я создала универсальную функцию для численного решения системы дифференциальных уравнений по методу Эйлера с управлением АКАР.

Помимо этого, я ознакомилась с моделью «Хищник - жертва», разными математическими методами решения систем уравнения и впервые написала документацию для функции.

<https://github.com/katenuit/summer-practice-2022>

1. **Приложение с исходным кодом**

calc\_f

function output = calc\_f (t, x)

alfa1 = 0.8;

alfa2 = 0.7;

beta1 = 0.1;

beta2 = 0.3;

vector(1) = alfa1 \* x(1) - beta1 \* x(1) \* x(2);

vector(2) = - alfa2 \* x(2) + beta2 \* x(1) \* x(2);

output = vector';

end

calc\_u

function output = calc\_u (t, x)

alfa1 = 0.8;

alfa2 = 0.7;

beta1 = 0.1;

beta2 = 0.3;

T1=0.9;

T2 = 1;

x2const = 3;

phi = (alfa2\*x(2) - (1/T2) \* (x(2) - x2const))/(beta2 \* x(2));

f1 = alfa1 \* x(1) - beta1 \* x(1) \* x(2);

f2 = - alfa2 \* x(2) + beta2 \* x(1) \* x(2);

phi2 = (x2const / (T2 \* beta2 \* x(2) \* x(2))) \* f2;

e = - 1/T1 \* (x(1) - phi) - f1 + phi2;

u(1) = e;

u(2) = 0;

output = u';

end

calc

% User input:

% function pointer: calc\_f

% function pointer: calc\_u

% vector: min\_max (example: [1 10])

% vector: init\_vals (example: [1 2 3 4 5])

% numeric: dt - точность/шаг

%

% call example:

% >>[T, x] = calc(@calc\_f, @calc\_u, [0 10], [0 0 0 0], 0.1)

%

function [T, x]= calc (calc\_f, calc\_u, min\_max, init\_vals, dt)

x = init\_vals';

T = [min\_max(1):dt:min\_max(2)];

for i = 1 : length(T)-1

t\_i = dt \* i;

f\_i = feval(calc\_f, t\_i, x(:,end));

u = feval(calc\_u, t\_i, x(:,end));

for j = 1:length(f\_i)

h = x(j,i) + dt \* f\_i(j)+u(j);

if (h > 1000)

h = 999;

elseif (h < -1000)

h=-999;

end

x(j, i+1) = h;

end

end

end