

Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математической физики

Лытов Дмитрий Андреевич

**Применение Байесовских методов машинного обучения для решения обратной задачи модели** **Лотки — Вольтерры**

КУРСОВАЯ РАБОТА

**Научный руководитель:**

к.ф-м.н. С.Б.Березин

Москва, 2019

Оглавление

[Введение 3](#_Toc9930886)

[Цель работы 3](#_Toc9930887)

[Основная часть 4](#_Toc9930888)

[Модель Лотки – Вольтерры 4](#_Toc9930889)

[Описание данных 5](#_Toc9930890)

[Постановка задачи 5](#_Toc9930891)

[Возможные варианты решения 6](#_Toc9930892)

[Решение 8](#_Toc9930893)

[Код 10](#_Toc9930894)

[Пример работы программы 12](#_Toc9930895)

[Эксперимент 13](#_Toc9930896)

[Оценка 15](#_Toc9930897)

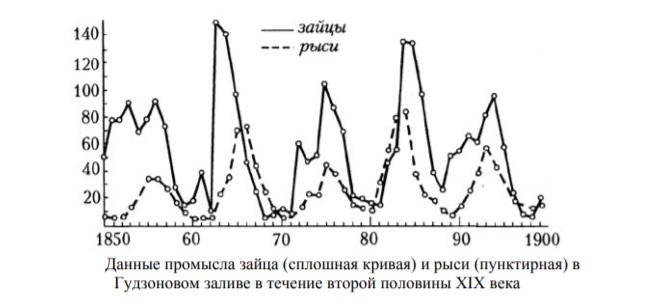
[Заключение 15](#_Toc9930898)

[Список литературы 16](#_Toc9930899)

# Введение

*Система «хищник — жертва» — сложная экосистема, для которой реализованы долговременные отношения между видами хищника и жертвы.*

*На (рис. 1) приведены примеры реальной динамики численности популяции зайцев* и *рысей [1]*

Рис. 1

*Одна из простейших математических моделей для описания динамики численности популяций -* ***Модель Лотки — Вольтерры***. В ней полагаются известными некоторые параметры, характеризующие регуляцию численности популяций, что позволяет смоделировать их динамику. Но что, если наша задача в обратном? Допустим, что уже имеем измерения численности популяций за последние несколько лет наблюдений за видами, но нам хотелось бы по этим данным найти исходные параметры – числовые значения, характеризующие регуляцию их численности?

# Цель работы

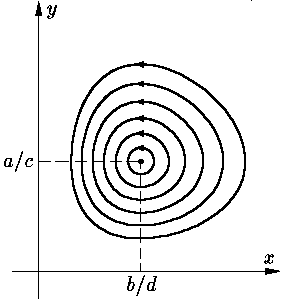
*1. По данным измерениям численности популяций видов хищников и жертв приблизить исходные параметры модели Лотки – Вольтерры.*

*2. Реализовать алгоритм, решающий данную задачу с помощью Байесовских методов машинного обучения и выяснить, насколько точен этот подход в сравнении с другими известными моделями решения обратной задачи для модели Лотки – Вольтерры.*

# Основная часть

## Модель Лотки – Вольтерры

Решением на фазовой плоскости является семейство замкнутых траекторий с центром в равновесной точке:

Координаты равновесной точки находятся [2]

## 

## Описание данных

Данные о численности популяций видов будет моделировать самостоятельно. Для этого был реализован алгоритм для прямого решения поставленной задачи с начальными условиями После того, как я передаю ему на вход некоторые исходные параметры, он моделирует динамику популяций (в виде двух непрерывных функций от времени), и возвращает случайным образом сделанные измерения (берет точки на графике /\*каком\*/ с некоторыми отклонениями, соответствующими погрешности измерения):

## Постановка задачи

На основе данных (2) приблизить исходные параметры системы .

Для оценки точности работы алгоритма используем функционал .

Для визуализации будем строить решение прямой задачи с найденными параметрами и оценивать схожесть с решением задачи с исходными параметрами.

## 

## Возможные варианты решения

Для решения поставленной задачи я использовал Байесовские методы машинного обучения. *Байесовский вывод* — статистический вывод, в котором свидетельство и/или наблюдение используются, чтобы обновить или вновь вывести вероятность того, что гипотеза может быть верной. Название байесовский происходит от частого использования в процессе вывода теоремы Байеса. Вывод производится с помощью метода «Expectation propagation». Но при обучении использовал две различные модели:

1. Разностная модель.

Пусть имеется выборка из n точек , тогда имеем

Аналогично для

1. Модель с использованием первого интеграла

Найдем первый интеграл системы (1):

По определению, значение функции слева принимает одинаковое значение на всех точках наперед взятой фазовой траектории. Воспользуемся этим, чтобы описать модель.

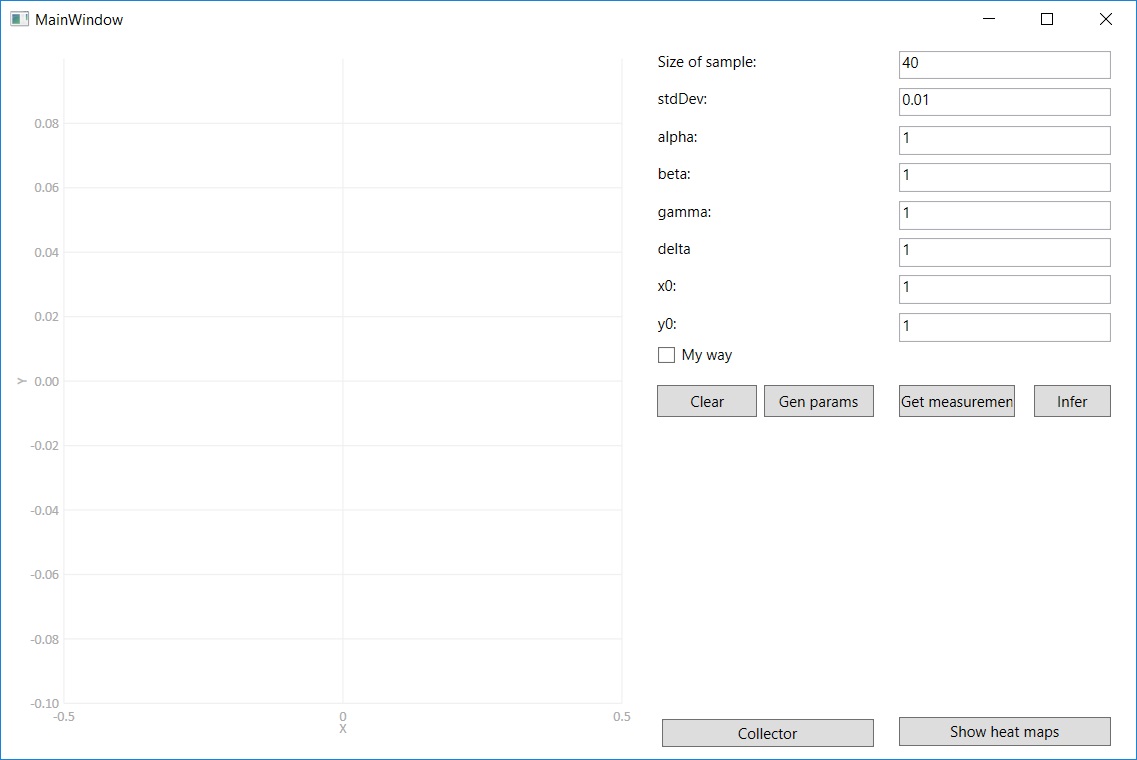
Пусть имеется выборка из n точек , тогда имеем

После обучения получаем значения . Дополнительно, предоставляет данные о конкретной траектории, входные данные не требуют временных отметок, но и параметры получены с точностью до мультипликативной константы (из-за отсутствия данных о времени)

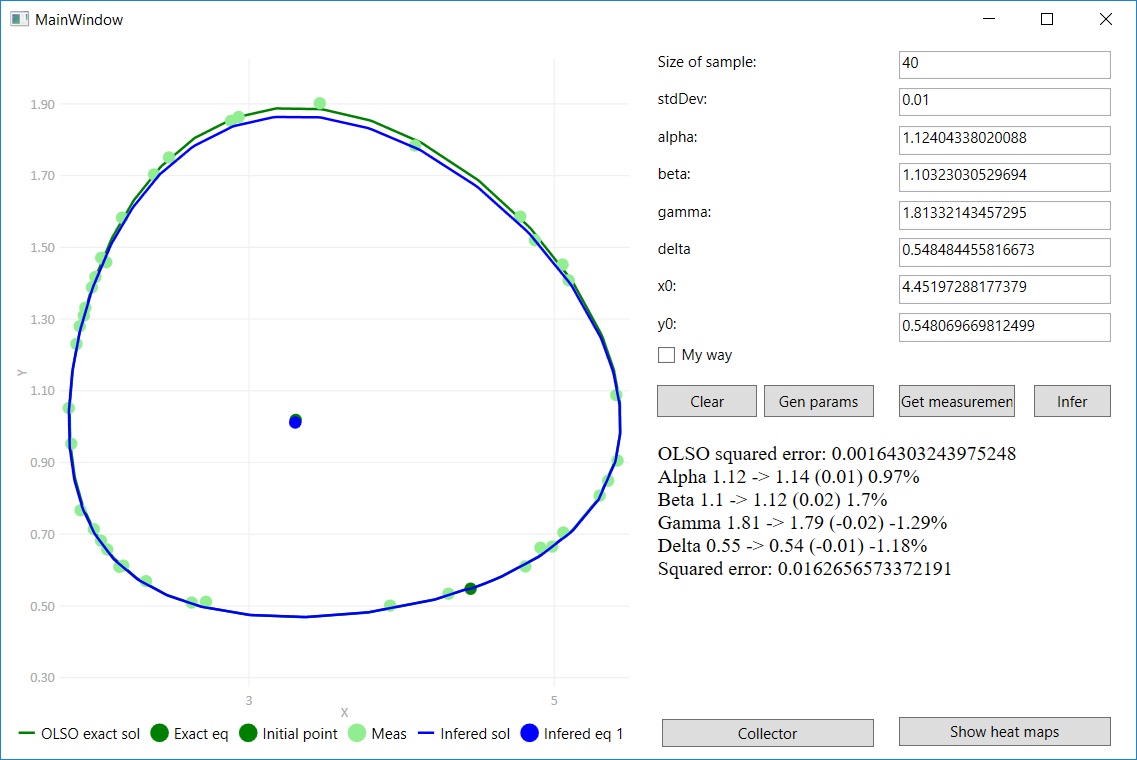
## Решение

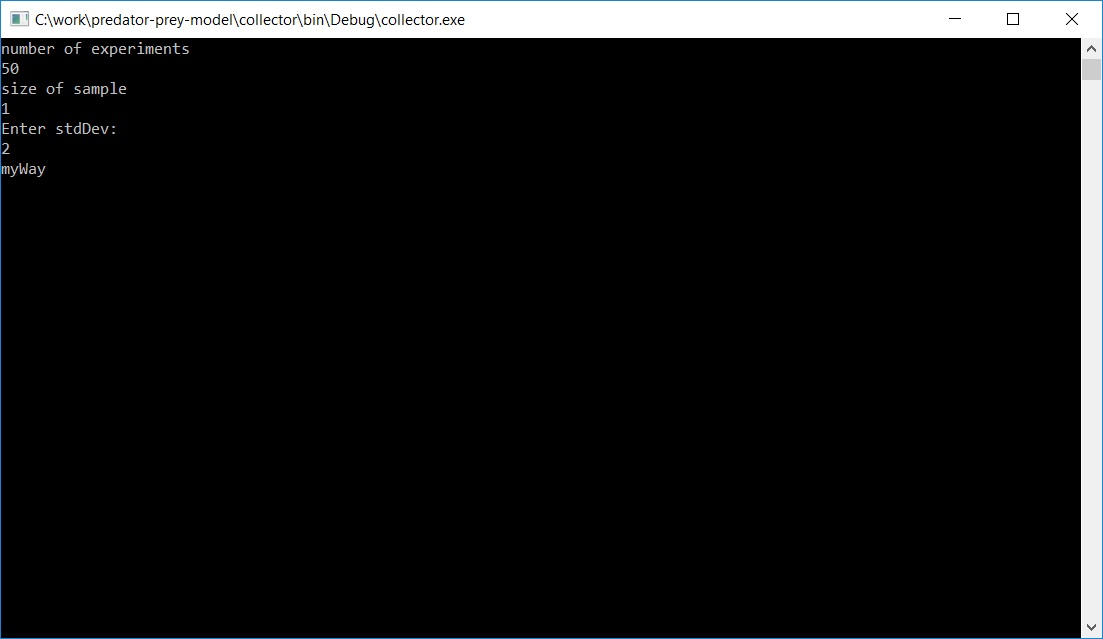
Для описания модели и ее обучения я выбрал программный пакет *Microsoft Infer .NET* – библиотеку, реализующую Байесовские методы машинного обучения, разработанную Microsoft Research. В качестве языка программирования и среды разработки я выбрал C# и Microsoft Visual Studio 2015. Для отрисовки графиков использовал пакет LiveCharts.

В составе решения были написаны 5 проектов:

1. User Interface – пользовательский интерфейс, обеспечивающий возможность использования функционала для пользователя и просмотр графиков;

Пользователю предлагается пронаблюдать эксперимент. Ввести исходные параметры системы или сгенерировать их случайно, решить прямую задачу, снять измерения с выбранной погрешностью, решить обратную задачу одним из двух способов. После чего предлагается оценить визуально сходство решений для исходных и приближенных параметров, а также оценить точность их нахождения:



1. Solver – проект, отвечающий за решение прямой дифференциальной задачи;
2. Randomizer – проект, предоставляющий случайные исходные параметры для экспериментов;
3. Predictor – проект, отвечающий за решение обратной задачи;
4. Collector – проект, запускающий серию экспериментов в автоматическом режиме и собирающий их результаты в выходной файл;

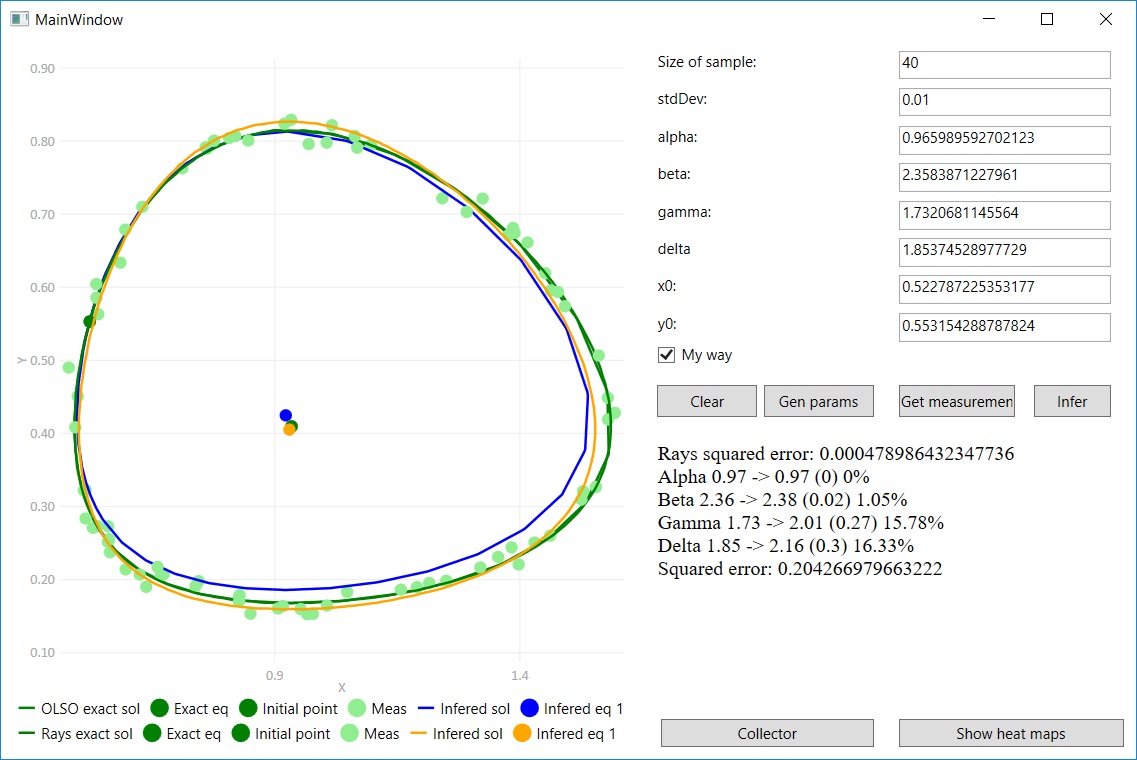
## Код

Ниже приведен код обучения модели и вывода (первый способ):

1. Variable<**double**> alpha = Variable.GaussianFromMeanAndVariance(5, 2).Named("alpha");
2. Variable<**double**> beta = Variable.GaussianFromMeanAndVariance(5, 2).Named("beta");
3. Variable<**double**> gamma = Variable.GaussianFromMeanAndVariance(5, 2).Named("gamma");
4. Variable<**double**> delta = Variable.GaussianFromMeanAndVariance(5, 2).Named("delta");
6. Range dataRange = new Range(points.Count);
7. VariableArray<**double**> dxdt = Variable.Array<**double**>(dataRange);
8. VariableArray<**double**> dydt = Variable.Array<**double**>(dataRange);
9. VariableArray<**double**> x = Variable.Array<**double**>(dataRange);
10. **VariableArray<double> y = Variable.Array<double>(dataRange);**
12. **using** (Variable.**ForEach**(dataRange))
13. {
14. dxdt[dataRange] = (alpha - beta \* y[dataRange]) \* x[dataRange];
15. **dydt[dataRange] = (-gamma + delta \* x[dataRange]) \* y[dataRange];**
16. }
18. **double**[] xObserved = new **double**[points.Count];
19. **double**[] yObserved = new **double**[points.Count];
20. **double[] dxdtObserved = new double[points.Count];**
21. **double**[] dydtObserved = new **double**[points.Count];
23. **for** (**int** i = 0; i < points.Count; i++)
24. {
25. **xObserved[i] = points[i][0];**
26. yObserved[i] = points[i][1];
27. }
29. dxdtObserved[0] = (xObserved[1] - xObserved[0]) / (points[1][2] - points[0][2]);
30. **dydtObserved[0] = (yObserved[1] - yObserved[0]) / (points[1][2] - points[0][2]);**
32. dxdtObserved[points.Count - 1] = (xObserved[points.Count - 1] - xObserved[points.Count - 2]) / (points[points.Count - 1][2] - points[points.Count - 2][2]);
34. dydtObserved[points.Count - 1] = (yObserved[points.Count - 1] - yObserved[points.Count - 2]) / (points[points.Count - 1][2] - points[points.Count - 2][2]);
36. **for** (**int** i = 1; i < points.Count - 1; i++)
37. {
38. dxdtObserved[i] = (xObserved[i + 1] - xObserved[i - 1]) / (points[i + 1][2] - points[i - 1][2]);
39. dydtObserved[i] = (yObserved[i + 1] - yObserved[i - 1]) / (points[i + 1][2] - points[i - 1][2]);
40. **}**
42. x.ObservedValue = xObserved;
43. y.ObservedValue = yObserved;
44. dxdt.ObservedValue = dxdtObserved;
45. **dydt.ObservedValue = dydtObserved;**
47. InferenceEngine engine = new InferenceEngine(new ExpectationPropagation());

## Пример работы программы

Сгенерируем случайные параметры и проведем эксперимент.



Темно-зеленым обозначена исходная траектория, её начальная и равновесная точки.

Светло-зеленым обозначены сделанные по ней измерения.

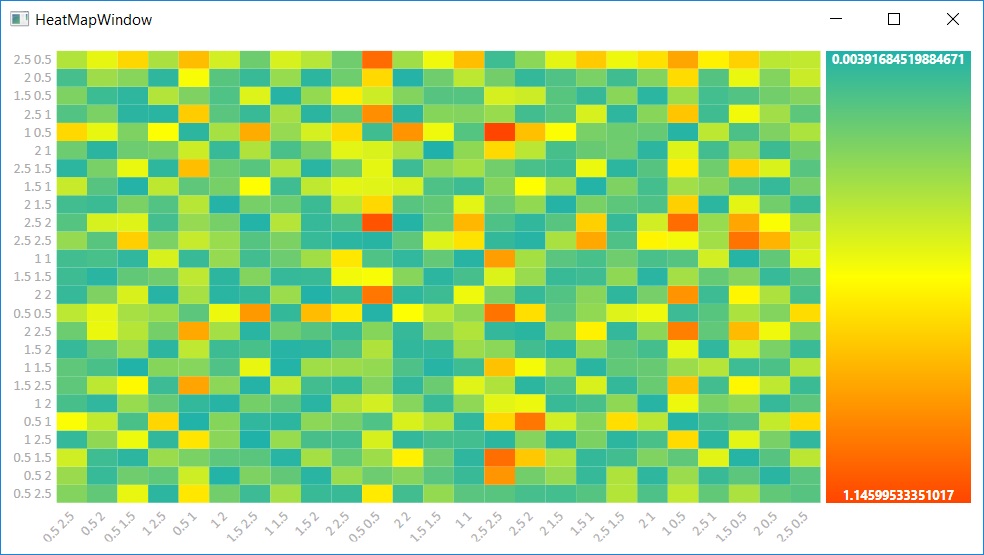
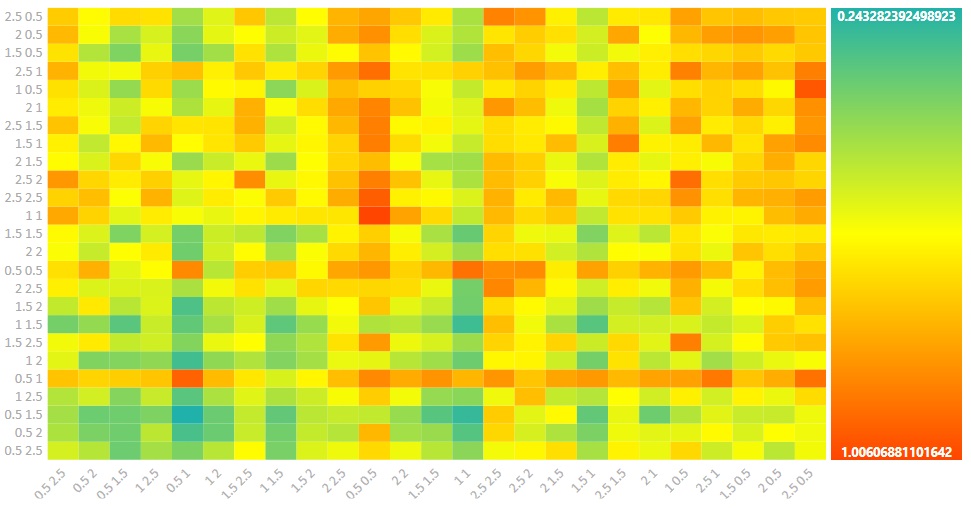
Синим обозначены траектория и её равновесная точка, полученные первым способом.

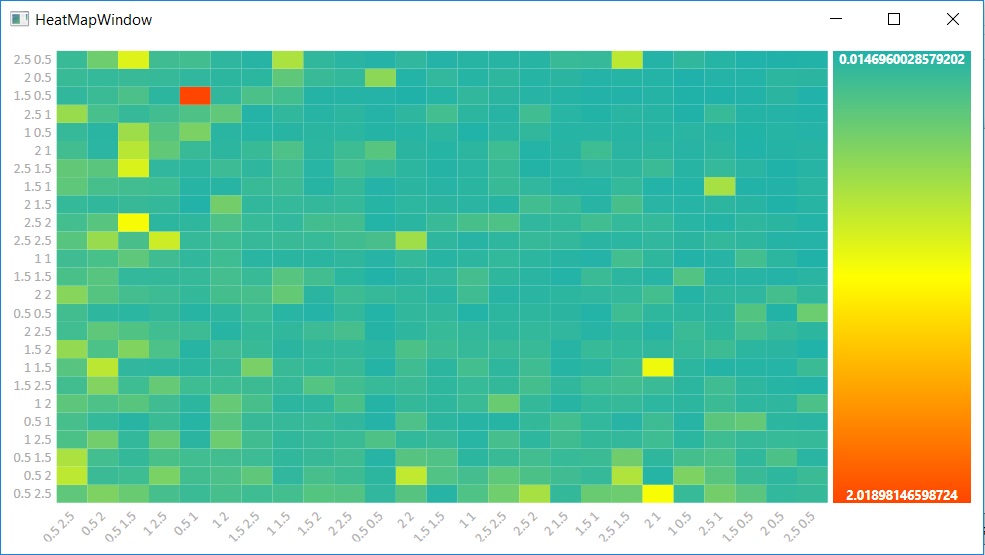
Желтым обозначены траектория и её равновесная точка, полученные вторым способом.

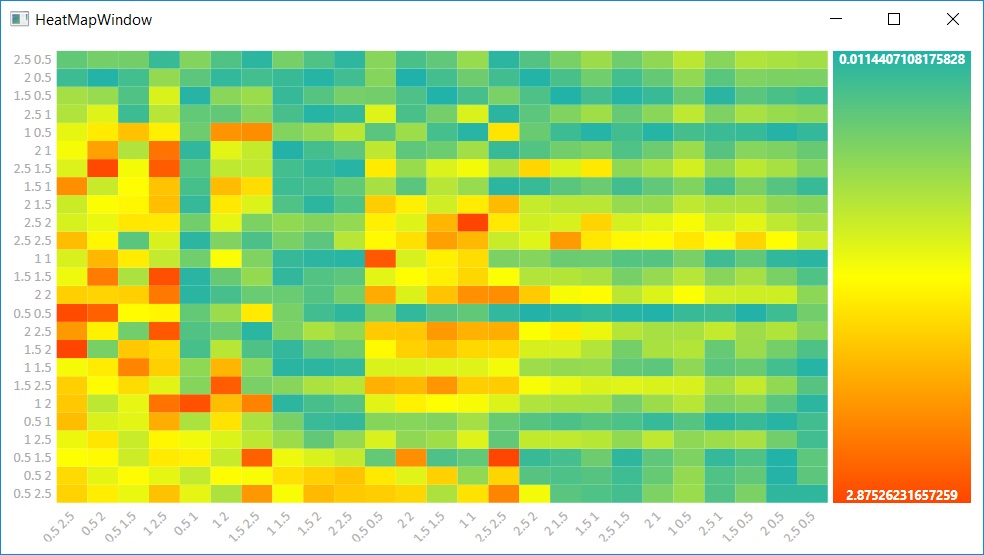
## Эксперимент

Была проведена серия экспериментов, в ходе которых были получены данные о работе каждого из двух методов для выборки из 5 точек и погрешности измерения с нормальным отклонением при и для выборки из 50 точек и погрешности измерения при . Для каждой комбинации было проведено 50 экспериментов. На тепловой карте цвет подобран в соответствии с медианой ошибки.

N = 5, stdDev = 0.001 (первый и второй способы, соответственно):



N = 50, stdDev = 0.01 (первый и второй способы, соответственно): 



## Оценка

Точность приближения параметров напрямую зависит от погрешности измерений. Так при погрешности, представленной нормальным распределением с оба метода показали точность в пределах 1, а при уже в пределах 2. Количество точек не помогает с решением задачи. Модель одинаково работает и при 5, и при 50 точках в выборке.

# Заключение

По данным измерениям численности популяций видов хищников и жертв мне удалось относительно точно приблизить исходные параметры модели Лотки – Вольтерры.

Реализовано два алгоритма, решающих данную задачу с помощью Байесовских методов машинного обучения.

# 

# Список литературы

1. «Хищник и жертва» К. Богданов
2. «Очерки по теории обыкновенных дифференциальных уравнений» Дифференциальные уравнения в биологии, химии, медицине
3. «Model-Based Machine Learning» Christopher Bishop, Thomas Diethe
4. «Байесовские методы машинного обучения» К. Струминский, Д. Молчанов
5. «Bayesian Reasoning and Machine Learning» Barber D.