Решение моделей в непрерывном и дискретном времени

Гань Чжаолун

20 декабрь, 2024, Москва, Россия

Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

Основной целью работы является освоение специализированных пакетов для решения задач в непрерывном и дискретном времени.

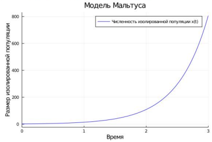
Процесс выполнения лабораторной работы

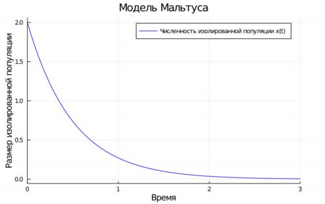
Используя Jupyter Lab, повторите примеры из раздела 6.2

Я повторю все задание 6.2 целиком

Выполните задания для самостоятельной работы

```
plot(sol, label = "Численность изолированной популяции x(t)", color="blue", ls=[:solid], title="Модель Мальтуса",
    xaxis="Время", yaxis="Размер изолированной популяции")
```





Я использовала следующие коэффициенты: a = b – c = 2.0, b = 3.0, c = 1.0 и смоделируем на интервале от 0 до 3. Я решила взять довольно большой коэффициент рождаемости и роста популяции тем самым ожидая на графике очень быстрый рост населения, что в принципе я и получила. Так как за 3 единицы времени размер изолированной популяции с 2 увеличилось до 800 по экспоненте.

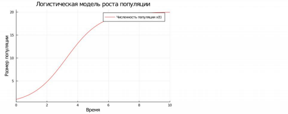
Попробую подобрать другие параметры, чтобы сравнить скорость роста популяции. Допустим в этот раз я возьму довольно маленький коэффициент рождаемости и большой коэффициент смертности. Пусть b = 1, c = 3.

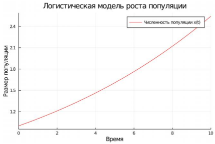
Размер популяции за 3 ед. времени резко снизился до 0. Также реализована анимация графика и представлена вместе с программой в архиве.

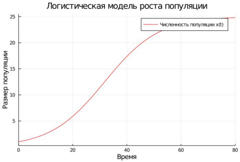
Рисунок 1. Код и результат Задания 1

Выполните задания для самостоятельной работы

```
plot(sol, label = "Численность популяции x(t)", color="red", ls=[:solid], title="Логистическая модель роста популяции",
    xaxis="Время", yaxis="Размер популяции")
```







Задала коэффициенты следующие r = 0.9, k = 20. Таким образом коэффициент роста равен 0.9, а предельное значение численности популяции равно 20, поэтому график на интервале от 0 до 10 должен достичь по оси y значение 20 и выше не подниматься.

Теперь разберем ситуация в которой коэффициент роста популяции намного меньше, допустим 0.1, а предельное значение допустим равно 25 на таком же временном интервале от 0 до 10.

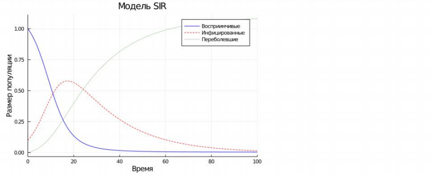
Как мы видим, за 10 ед. времени с таким маленьким коэффициентом роста популяции она не успела дойти до предельного значения, увеличим временной промежуток до 80 и посмотрим результат.

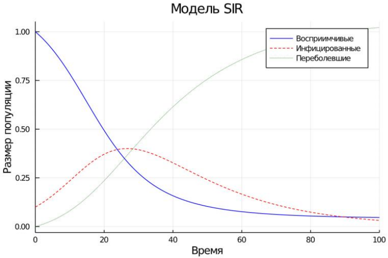
Тем самым можно сделать вывод, что чем меньше мы возьмем коэффициент роста популяции, тем медленнее она будет расти.

Рисунок 2. Код и результат Задания 2

Выполните задания для самостоятельной работы

```
plot(sol, label = ["Восприимчивые" "Инфицированные" "Переболевшие"], color=["blue" "red" "green"], ls=[:solid :dash :dot],
    title="Модель SIR",
    xaxis="Время",yaxis="Размер популяции")
```





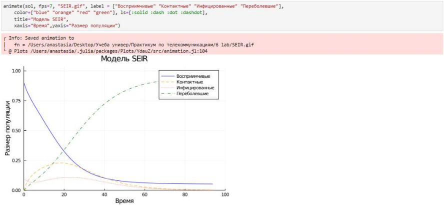
Изначально задам $\beta = 0.25$, $\nu = 0.05$. Таким образом мы имеем достаточно маленький коэффициент интенсивности выздоровления и достаточно большой коэффициент интенсивности контактов индивидов с последующим инфицированием.

Красный график интенсивности эпидемии, показывающей количество одномоментно болеющих индивидов, определяется параметром: $R0 = \beta / \nu = 5$. Попробуем уменьшить данный коэффициент интенсивности и посмотрим на различия. Сделаем это за счет уменьшения коэффициента β, например, 0.15.

Видно, что число инфицированных растет намного медленнее, а также в целом меньшее число людей было инфицировано по истечению времени, нежели в первом случае. Коэффициент интенсивности $R0 = \beta / \nu = 3$.

Рисунок 3. Код и результат Задания 3

Выполните задания для самостоятельной работы



1.Определение системы уравнений SEIR.
Я задал систему дифференциальных уравнений, в которой:
- (s(t) — доля (или количество) восприимчивых,
- (e(t) — доля «контактных», то есть находящихся в инкубационном периоде,
- (i(t) — доля инфицированных,
- (r(t) — доля переболевших (удалённых из эпидемического процесса).

Дополнительно присутствуют параметры:
- beta — скорость контакта (заражения),
- delta — скорость перехода из «контактных» в инфицированные,
- gamma — скорость выздоровления (или удаления).

2.Выбор начальных условий.
Я задал начальные условия так, чтобы в начале почти вся популяция находилась во «восприимчивом» состоянии (s = M - initialInfect), небольшая часть сразу была инфицирована (i = initialInfect), а «контактных» и переболевших изначально нет (e = 0 и r = 0).

3.Задание интервала интегрирования и параметров.
Я выбрал временной промежуток t in [0, 100]. Параметрам beta, gamma, delta присвоил конкретные числовые значения, соответствующие гипотетическому сценарию развития эпидемии.

4.Решение системы ODE.
Я сформировал объект `ODEProblem` из библиотеки `DifferentialEquations.jl`, передав ему описание системы, начальные условия, интервал времени и кортеж параметров. Затем с помощью функции `solve()` получил численное решение.

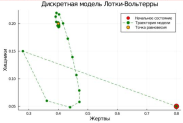
5.Визуализация и анимация.
- Итоговая GIF-анимация иллюстрирует, как SEIR-модель описывает развитие эпидемии с учётом «скрытого» периода состояние (e(t).

Рисунок 4. Код и результат Задания 4

Выполните задания для самостоятельной работы

```
n = 100
anim = @animate for i in 1:n
    modelingTime = (i)
    # задаём значения параметров:
    a, c, d = 2, 1, 5
    # задаём начальное условие:
    u0 = [0.8, 0.05]

    simTrajectory = Array{Union{Nothing, Array}}(nothing, modelingTime)

    for t in 1:modelingTime
        simTrajectory[t] = []
        if(t == 1)
            simTrajectory[t] = u0
        else
            simTrajectory[t] = next(simTrajectory[t-1]...)
        end
    end

    scatter([simTrajectory[1][1]], [simTrajectory[1][2]],
    c=:red, ms=9, label="Начальное состояние")

    plot!(first.(simTrajectory), last.(simTrajectory), color=:green, linestyle=:dash,
    marker = (:dot, 5, Plots.stroke(0)), label="Траектория модели",
    title = "Дискретная модель Лотки-Вольтерры", xlabel="Жертвы", ylabel="Хищники")

    scatter!([balancePoint[1]], [balancePoint[2]], color=:orange, markersize=5,
    label="Точка равновесия",
    xlabel="Жертвы", ylabel="Хищники")
end
gif(anim, "LotkaVolterra.gif", fps=7)
```

```
┌ Info: Saved animation to
│   fn = /Users/anastasia/Desktop/Учеба универ/Практикум по телекоммуникациям/6 lab/LotkaVolterra.gif
└ @ Plots /Users/anastasia/.julia/packages/Plots/YdauZ/src/animation.jl:104
```



1.Вычисление точки равновесия.
Точка равновесия (X_1^*), (X_2^*) получается из условия, что Delta X_1 = 0 и Delta X_2 = 0. Я аналитически нашёл это решение и сохранил в переменную `balancePoint`.

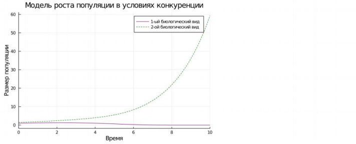
2.Численное решение и построение траектории.
– Я выбрал начальное состояние X_1(0)=0.8, X_2(0)=0.05 и задал количество итераций (`modelingTime`), после чего в цикле по шагам вычислял следующие значения жертв и хищников с помощью функции `next`.
– Для визуализации создал фазовый портрет (ось абсцисс — жертвы, ось ординат — хищники) и отобразил:
– начальную точку,
– траекторию, по которой система перемещается,
– точку равновесия.

3.Анимация.
– Чтобы показать, как траектория «растёт» во времени, я использовал макрос `@animate`. На каждом шаге анимации я пересчитывал систему вплоть до текущего момента и рисовал результат.
– Наконец, собрал все кадры в GIF-файл с помощью `gif(anim, "LotkaVolterra.gif", fps=7)`.

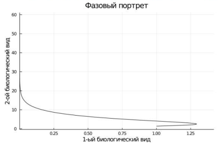
Рисунок 5. Код и результат Задания 5

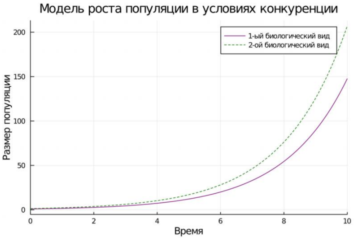
Выполните задания для самостоятельной работы

```
plot(sol, label = ["1-ый биологический вид" "2-ой биологический вид"], color=["purple" "green"], ls=[:solid :dash],
    title="Модель роста популяции в условиях конкуренции",
    xaxis="Время",yaxis="Размер популяции")
```




```
# фазовый портрет:
plot(sol, vars=(1,2), color="black", title="Фазовый портрет",
    xaxis="1-ый биологический вид", yaxis="2-ой биологический вид", legend=false)
```



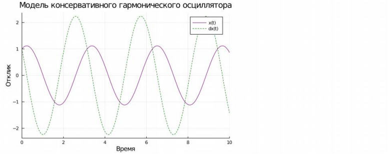


В данной модели мы задаем 2 параметра, насколько я понимаю первый отвечает за рост популяции обеих групп, а второй это коэффициент конкурентности. То есть насколько одна группа конкурирует с другой.
Также в данной модели я заметила, что сильно влияют начальные значения. То есть количество особей в двух моделируемых группах на начало времени. В моем случае 1-я группа имеет начальное значение 1, а вторая группа - 1.4. Следовательно по моим предположениям, второй биологический вид должен выиграть в данной конкурентной среде и начать быстро расти.
вышло так, что второй вид полностью подавил первый биологический вид.

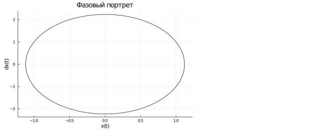
Рисунок 6. Код и результат Задания 6

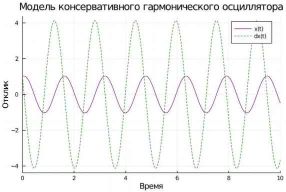
Выполните задания для самостоятельной работы

```
plot(sol, label = ["x(t)" "dx(t)"], color=["purple" "green"], ls=[:solid :dash],
    title="Модель консервативного гармонического осциллятора",
    xaxis="Время",yaxis="Отклик")
```




```
# фазовый портрет:
plot(sol, vars=(1,2), color="black", title="Фазовый портрет", xaxis="x(t)", yaxis="dx(t)", legend=false)
```





1.Начальные условия.
Я задал начальное смещение x(0)=1.0 и начальную скорость dot{x}(0)=1.0. Это позволяет наблюдать «свободные» колебания без дополнительного затухания или внешнего воздействия.

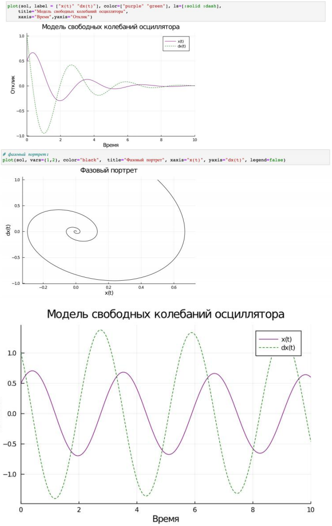
2.Численное решение.
Я оформил задачу как `ODEProblem(lv!, u0, tspan, p)` и решил её с помощью `solve(prob)`. Получил решение в виде временных рядов для x(t) и dot{x}(t).

3.Построение графиков и анимации.
- Построил график x(t) и dot{x}(t) от времени, чтобы увидеть колебания и убедиться, что они действительно гармонические.
- Построил фазовый портрет (ось абсцисс — x, ось ординат — dot{x}, где траектория представляет собой замкнутую эллиптическую кривую, характерную для консервативного осциллятора.
- Создал анимацию, чтобы наглядно показать, как решения меняются во времени, и сохранил её в виде GIF-файла.

В итоге удалось продемонстрировать, что в отсутствии диссипации (затухания) система совершает периодические колебания.

Рисунок 7. Код и результат Задания 7

Выполните задания для самостоятельной работы



1.Выбор параметров и начальных условий
  – Параметр gamma = 0.5 задал для иллюстрации умеренного затухания; omega_0 = 2.0 — для определённой «скорости» колебаний.
  – Начальное смещение x(0) = 0.5 и начальная скорость dot{x}(0) = 1.0 выбраны, чтобы наблюдать свободные затухающие колебания.

2.Численное решение
  – Сформировал задачу `ODEProblem` (из пакета `DifferentialEquations.jl`), передал описание системы, вектор начальных условий и параметры.
  – Функция `solve(prob)` вернула решение, по которому можно строить графики.

3.Построение графиков и анимации
  – Построил во времени функции x(t) и dot{x}(t), чтобы увидеть колебания и их затухание.
  – На фазовом портрете ось x, ось dot{x} затухающие колебания визуально проявляются в виде спиральной траектории, «закручивающейся» к равновесию.
  – С помощью `animate` создал GIF-файл, на котором видно, как решение эволюционирует по времени.

Таким образом, получилась модель свободных (но затухающих) колебаний гармонического осциллятора, позволяющая наглядно проследить динамику как в координате времени, так и на фазовом портрете.

Рисунок 8. Код и результат Задания 8

Выводы по проделанной работе

Вывод

Я освоил специализированные пакеты для решения задач в непрерывном и дискретном времени