МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Д. Х. ИМАЕВ А. М. СИНИЦА

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

по выполнению лабораторных работ

по дисциплине «Теория автоматического управления. Часть 2. Нелинейные системы»

Санкт-Петербург
Издательство СПб ГЭТУ «ЛЭТИ»
2021

УДК 681.5
ББК 32.972.13
В24

Имаев Д.Х., Синица А.М.

В24 Учебно-методическое пособие по выполнению лабораторных работ по дисциплине «Теория автоматического управления. Часть 2. Нелинейные системы» — СПб: Изд-во СПб ГЭТУ «ЛЭТИ», 2021. 48 с.

ISBN 000-0-0000-0000-0

Содержит указания по выполнению лабораторных работ по части «Нелинейные системы» дисциплины «Теория автоматического управления». Предлагаются для изучения нелинейные модели элементов и систем управления, точные и приближенные методы исследования статических характеристик, а также компьютерной имитации нелинейной динамики с помощью пакта программ SciPy, основанного на языке Python, и среды для научно-технических расчетов MATLAB.

Предназначено для подготовки студентов-бакалавров, обучающихся по направлениям подготовки: 27.03.04 – «Управление в технических системах (Компьютерные интеллектуальные технологии управления в технических системах)».

УДК 681.5
ББК 32.972.13

Рецензент – проф. В.В. Цехановский (Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»)

Утверждено
редакционно-издательским советом университета в качестве учебно-методического пособия

ISBN 000-0-0000-0000-0

СОДЕРЖАНИЕ

Предисловие

Нелинейные математические модели, используемые для анализа и синтеза объектов и систем управления, появляются как следствие нелинейного характера законов природы, которым подчиняются исследуемые явления. Нелинейности могут вводиться в системы управления и специально для компенсации нежелательных эффектов, вызванных естественными нелинейностями, или с целью придания системе управления особых свойств, которые принципиально недостижимы линейными средствами. Нелинейные алгоритмы управления могут обеспечить максимальное быстродействие процессов при ограничениях на уровни управляющих воздействий, нелинейности обязательно вводятся при создании генераторов колебаний. В ряде систем управления существенно нелинейные — релейные регулирующие устройства оказываются наиболее простыми, дешевыми и надежными.

Расчет систем управления по нелинейным моделям значительно сложнее, чем по линейным моделям. Это объясняется бóльшим разнообразием движений, описываемых нелинейными уравнениями. Усложнение моделей вызвано необходимостью расширения и углубления знаний о поведении объектов и систем управления.

Большинство задач исследования нелинейных динамических систем решается с помощью инструментов численного решения дифференциальных уравнений. Распространенная в академической среде и практике проектирования систем управления среда MATLAB/ Simulink фирмы The MathWorks, Inc. является весьма объемным пакетом программ для научно-технических расчетов и компьютерного моделирования. К его достоинствам относится удобство использования в самых разных приложениях. Недостатки инструмента заключаются в существенных системных требованиях, высокой стоимости и длительности установки на рабочий компьютер.

Авторы настоящих методических указаний по выполнению лабораторных работ по дисциплине «Теория автоматического управления. Часть 2. Нелинейные системы» [1], [2], наряду с инструментами MATLAB (R2020 - academic use) предлагают экосистему (как называют ее разработчики) SciPy, основанную на языке Python [3]. Знакомство с экосистемой вычислений и моделирования SciPy (Python) будет полезно для будущих инженеров.

Набор программных средств SciPy для математических, научных и инженерных расчетов (как символьных, так и численных) включает пакеты:

NumPy (работа с многомерными матрицами), SciPy library (Базовая библиотека для научных вычислений), Matplotlib (построение графиков), IPython (интерактивная консоль для работы), Sympy (пакет для символьных или аналитических вычислений), pandas (структурирование и анализ данных). Кроме того, существует множество библиотек для Python, разработанных для решения различных научных задач.

Свободное программное обеспечение Anaconda компании Continuum включает большинство необходимых пакетов и имеет сравнительно небольшой объем и системные требования.

Пособие предназначено для подготовки студентов-бакалавров, обучающихся по направлениям подготовки: 27.03.04 – «Управление в технических системах (Компьютерные интеллектуальные технологии управления в технических системах)».

УМП по дисциплине: «Теория автоматического управления», УП № 339, лаб. зан., 6 с., бак.;
каф АПУ, очная

Лабораторная работа 1. Типовые нелинейности в моделях систем управления

Нелинейные модели систем управления в большинстве случаев появляются в результате дополнения линейных моделей нелинейными элементами (НЭ), учитывающими такие естественные факторы, как ограниченность управляющих воздействий, наличие зоны нечувствительности в измерительных элементах и исполнительных механизмах, люфт в кинематических сочленениях и др.

Задача работы: изучение типовых НЭ моделей систем автоматического управления.

Задание на лабораторную работу

Для безынерционных НЭ типа «идеальное реле», «зона нечувствительности», «насыщение»:

- сгенерировать различные тестовые (пробные) сигналы;
- получить реакции НЭ на тестовые сигналы;
- построить статические характеристики (СХ) НЭ;
- по СХ НЭ объяснить разницу в реакциях при разных пробных сигналах.

Выполнение работы в среде SciPy (Python)

Для получения параметров звеньев, укажите номер варианта в поле ниже и выполните его.

```
    variant = 1 # Изменяйте ТОЛЬКО значение варианта

    test_signal_duration = 100
    dt = 0.01
    test_sig_ampl = 1 + variant * 0.1
    test_sig_freq = 1 + variant * 3.5
    non_lin_param_1 = 0.5 + variant * 0.1
    lin_param_k = 0.5 + variant * 0.3
    lin_param_T = 0.1 + variant * 0.2

    print("Вариант номер {}".format(variant))
    print("Период дискретизации сигнала: {:.2} с".format(dt))
    print("Амплитуда тестового сигнала:
{:.2}".format(test_sig_ampl))
    print("Частота тестового сигнала: {:.2}
Гц".format(test_sig_freq))
```

```
    print("Длительность тестового сигнала: {} c".format(test_signal_duration))
    print("Параметр нелинейностей 1: {:.2}".format(non_lin_param_1))
    print("Коэффициент усиления линейного звена: {:.2}".format(lin_param_k))
    print("Постоянная времени линейного звена: {:.2}".format(lin_param_T))
```

Пробные сигналы. В качестве пробных (тестовых) сигналов в данной работе прежлагаются: синусоида (моногармонический сигнал), пилообразный сигнал, меандр.

Необходимо предварительно сформировать вектор времени. В Python для этого принято использовать команду

```
numpy.arange(start, stop  step)
```

или

```
numpy.linspace(T_0, T_k, N_t)
```

В сообществе пользователей Python принято следующее соглашение по импорту библиотеки `numpy`

```
import numpy as np
```

что означает импорт библиотеки с заменой имени на `np`.

Создадим вектор времени от 0 до 100 секунд с периодом дискретизации времени 0.01 с:

```
import numpy as np
t = np.arange(0, test_signal_duration, dt)

print("Размерность массива: {}".format(t.shape))
print("Содержимое массива: {}".format(t))
```

В качестве примера пробного сигнала выберем синусоиду. Для формирования синусоиды в Python используется функция `sin(Phase)` библиотеки NumPy. За аргумент функции принимается угол в радианах, поэтому для корректного вычисления необходимо преобразовать время в радианы с учетом частоты (в герцах):

```
sig_sin = np.sin(t * 2 * np.pi)

print("Размерность сигнала: {}".format(sig_sin.shape))
```

```
print("Содержимое массива сигнала: {}".format(sig_sin))
```

Построение графика. При применении Python для построения графиков принято использовать библиотеку Matplotlib. Пакет библиотеки импортируется следующим образом:

```
`import matplotlib.pyplot as plt`.
```

Названия и работа функций аналогичны функциям MATLAB, за исключением явного указания пакета функций.

Функция matplotlib

```
plt.figure('Name')
```

создает новое окно графика с именем Name. Все последующие команды применяются к созданному окну. Функция

```
plt.subplot(X, Y, N)
```

создает подграфики: X — число полей графиков по горизонтали; Y — число полей графиков по вертикали; N — номер поля.

Другие функции:

```
plt.plot(t, x)
```

строит график; *t* — ось абсцисс, *x* — ось ординат;

```
plt.ylim(yMin, yMax)
```

устанавливает пределы вывода графика по оси *y*;

```
plt.xlim(xMin, xMax)
```

устанавливает пределы вывода графика по оси *x*. Команда

```
plt.grid()
```

включение сетки;

```
plt.title('Name')
```

задание название графика;

```
plt.ylabel('Name')
```

— подпись оси *y*;

```
plt.xlabel('Name')
```

— подпись оси *x*.

Построим график ранее сформированной синусоиды (рис. 1.1).

```
plt.plot(t, sig_sin)
plt.show()
```

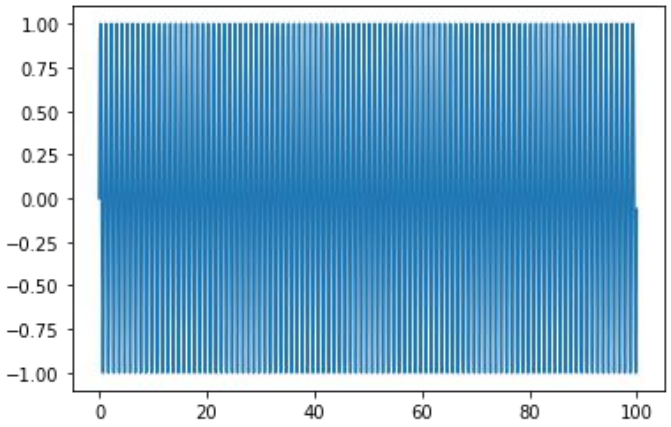


Рис. 1.1. График синусоиды

Построенный график не репрезентативен — на нем мало что видно. Поэтому построим участок графика с использованием среза (slice), чтобы по изображению сигнала было возможно его анализировать (рис. 1.2):

```
plt.plot(t[0:400], sig_sin[0:400])
```

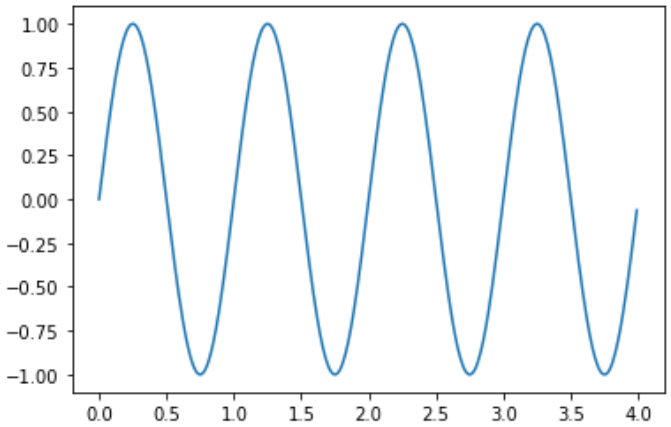


Рис. 1.2. График участка синусоиды

Кроме использования срезов, можно ограничить область видимости сигнала средствами matplotlib (`plt.ylim(yMin, yMax)` и `plt.xlim(xMin, xMax)`).

Добавим на графике сетку и подписи координатных осей (рис. 1.3):

```
plt.grid()
plt.xlabel('Time, s')
plt.ylabel('Ampl')
plt.ylim(-1.5, 1.5)
plt.xlim(0, 4)
plt.plot(t, sig_sin)
```

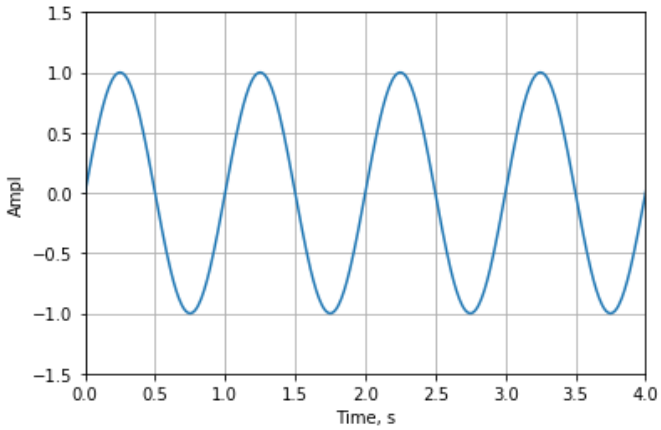


Рис. 1.3. График участка синусоиды с подписями и сеткой

Построение спектра сигнала. Для построения спектра дискретных сигналов берется модуль от дискретного (быстрого) преобразования Фурье сигнала.

В Python для этого используется функция `np.fft.fft()` и `np.abs()` для вычисления спектра и получения амплитуды, соответственно.

```
sig_sin_spec = np.abs(np.fft.fft(sig_sin))
print("Размерность массива спектра: {}".format(sig_sin_spec.shape))
print("Содержимое массива спектра: {}".format(sig_sin_spec))
```

Следующим шагом является построение графика спектра. Попробуем сделать это наивным образом (рис. 1.4):

```
plt.plot(sig_sin_spec)
```

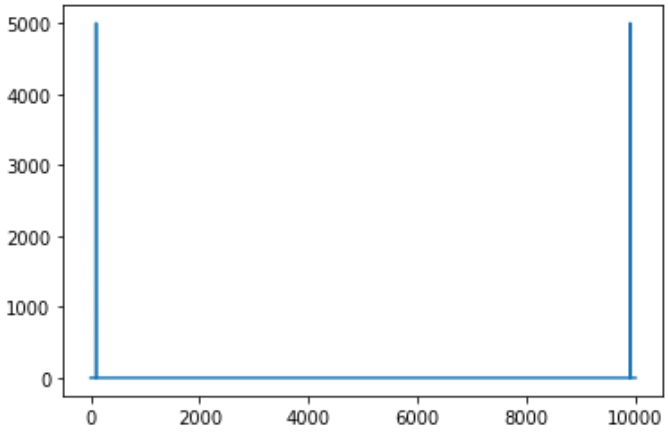


Рис. 1.4. «Наивный» график спектра

Как видно, отсчеты спектра не соответствуют частотам. Для получения соответствий каналов ДПФ и их частот в NumPy используется команда `*np.fft.fftfreq(<Размер вектора сигнала>, <Период дискретизации>)*` (Рис. 1.5):

```
freqs = np.fft.fftfreq(sig_sin.shape[0], dt)
plt.plot(freqs, sig_sin_spec)
```

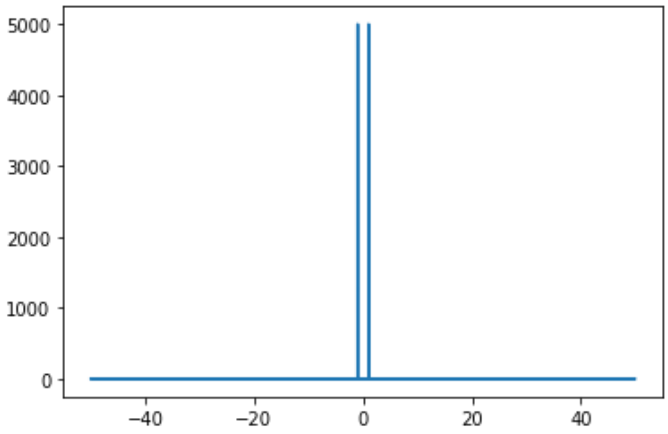


Рис. 1.5. График спектра с учетом значений частот

Ограничим область видимости и подпишем оси (рис. 1.6):

```
plt.grid()
plt.xlabel('Частота, Гц')
plt.xlim(-10, 10)
plt.plot(freqs, sig_sin_spec)
```

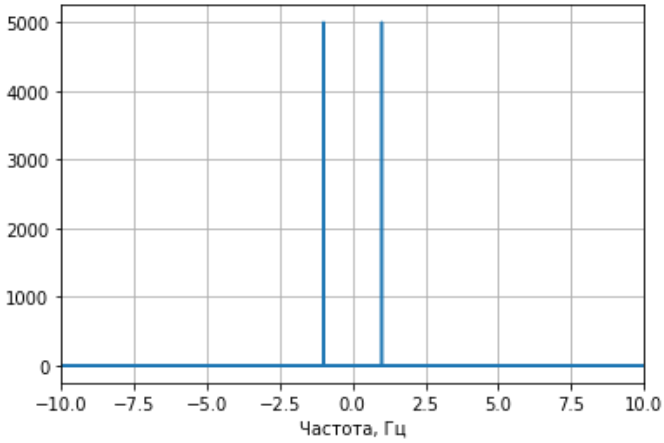


Рис. 1.6. График спектра с подписями и сеткой

Для того, чтобы графики располагались более компактно, можно использовать `plt.subplot` (рис. 1.7).

```
plt.subplot(1, 2, 1)
plt.grid()
plt.xlabel('Time, s')
plt.ylabel('Ampl')
plt.ylim(-1.5, 1.5)
plt.xlim(0, 4)
plt.plot(t, sig_sin)

plt.subplot(1, 2, 2)
plt.grid()
plt.xlabel('Частота, Гц')
plt.xlim(-10, 10)
plt.plot(freqs, sig_sin_spec)
```

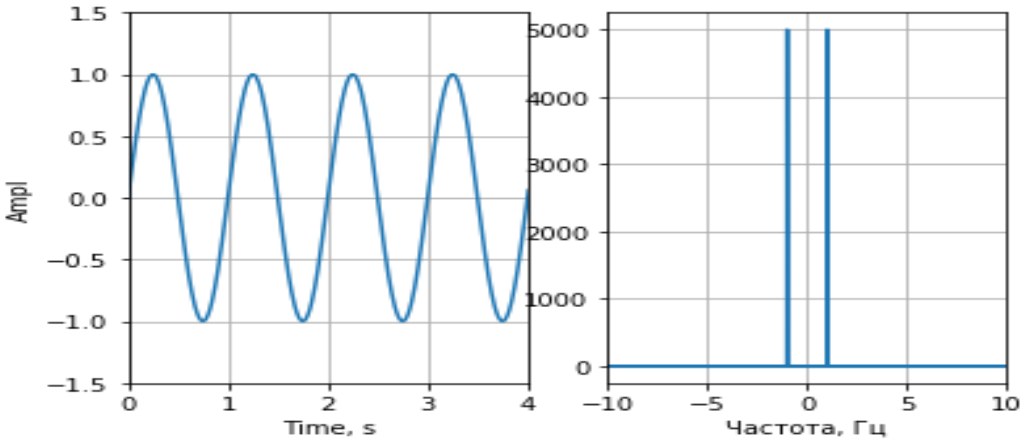


Рис. 1.7. График сигнала и спектра с использованием subplot

Другие сигналы. Для построения других пробных сигналов можно воспользоваться пакетом `signal` из библиотеки `scipy`:

```
from scipy import signal
```

В рамках работы необходимо исследовать пилообразный сигнал и меандр. Синтаксис и способ использования необходимо изучить с использованием документации используемого пакета. На момент публикации методических указаний документация доступна по следующим адресам:

- Пилообразный сигнал:

  https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.sawtooth.html

- Меандр:

  https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.square.html

Типовые НЭ. Для НЭ типа «идеальное реле» имеется готовая функция «знак» (sign)

```
relay = np.sign()
```

Для других типов НЭ необходимо реализовать соответствующую функцию самостоятельно.

Программа NumPy в отличие от MATLAB содержит функционал `np.vectorize` для создания векторизованных функций, что значительно повышает производительность при работе с большими матрицами («длинными» сигналами). Векторизация позволяет создавать быстрые функции, содержащие сложную обработку элементов матрицы. Далее показан пример создания векторизованной функции и ее применение.

НЭ типа «зона нечувствительности»:

```
def dead_zone_scalar(x, width = 0.5):
    if np.abs(x)<width:
        return 0
    elif x>0:
        return x-width
    else:
        return x+width

dead_zone = np.vectorize(dead_zone_scalar,
otypes=[np.float], excluded=['width'])
```

Выход безынерционного звена не зависит от частоты, а зависит только

от амплитуды. Поэтому реакцию типовых НЭ можно получить умножением вектора входного сигнала на функцию нелинейности.

Построим отклик НЭ типа «зона нечувствительности» на синусоиду (рис. 1.8). Для наглядности входной и выходной сигналы построим на одном графике

```
sig_sin_dz = dead_zone(sig_sin, non_lin_param_1)
plt.grid()
plt.xlabel('Time, s')
plt.ylabel('Ampl')
plt.ylim(-1.5, 1.5)
plt.xlim(0, 4)
plt.plot(t, sig_sin, t, sig_sin_dz)
```

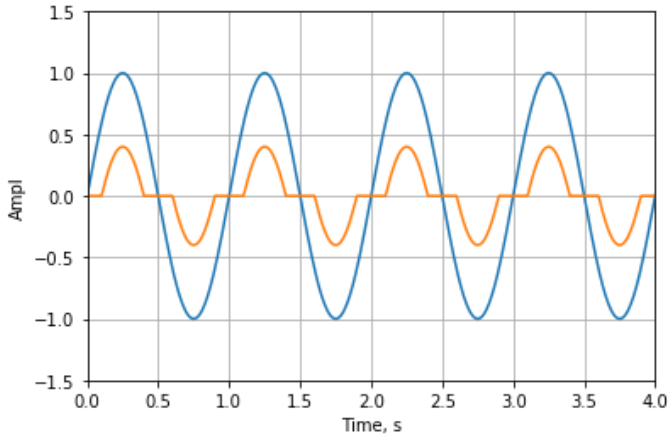


Рис. 1.8. Отклик НЭ типа «зона нечувствительности»

Функцию для НЭ типа «насыщение» предлагается разработать самостоятельно.

```
def saturation_scalar(x, hight = 0.5):
    #Разработайте функцию самостоятельно
    pass

saturation = np.vectorize(saturation_scalar, otypes=[np.float], excluded=['hight'])
```

Построение СХ безынерционного НЭ удобнее в среде MATLAB/Simulink (рис. 1.9). На вход НЭ подается гармонический сигнал от блока Sine Wave. Амплитуда сигнала подбираются таким образом, чтобы в окне блока XY-Graph получилась искомая СХ.

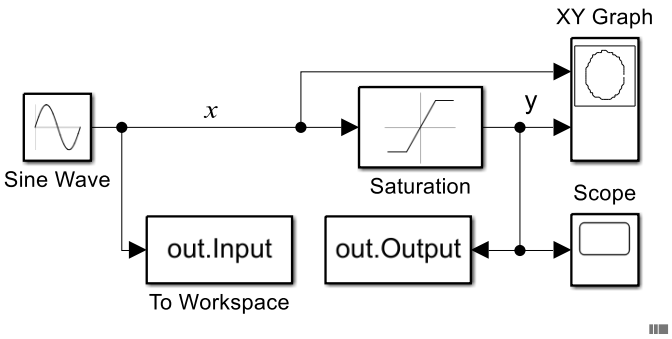


Рис. 1.9. «Исследовательский стенд»

Кроме построения СХ с помощью MATLAB/Simulink в рамках работы требуется самостоятельно построить СХ средствами построения графиков Python и привести исходный код с пояснениями.

Отчет должен содержать: результаты исследований трех типов нелинейностей при трех тестовых сигналах (9 комбинаций сигналов и нелинейностей). Текстовое описание и анализ должны быть для каждого элемента работы. Для сокращения объема отчета рекомендуется совмещать графики сигналов.

ВНИМАНИЕ: отчет по лабораторной работе является иллюстрированным текстом, а не картинками с комментариями.

Контрольные вопросы

1. Нарисуйте СХ НЭ предложенного типа.
2. Нарисуйте отклик НЭ на предложенный сигнал.
3. Нарисуйте отклик предложенной нелинейности на синусоиду или линейно-возрастающее воздействие.
4. Опишите назначение команд в исходном коде.

Лабораторная работа 2. Логические алгоритмы управления

Пример релейной системы автоматического регулирования. Релейные регуляторы — это простые и дешевые устройства, применяемые, например, в термостатах отопительных систем и бытовых холодильников. Принципиальная схема релейной системы стабилизации температуры изображена на рис. 2.1 [4].

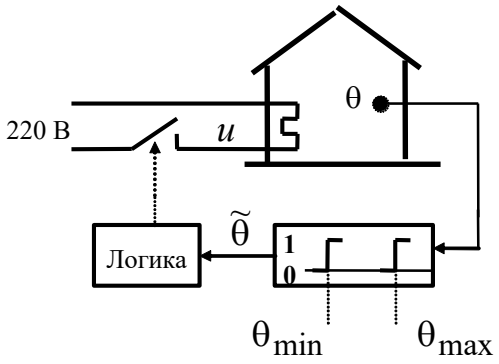


Рис. 2.1. Принципиальная схема релейной системы стабилизации температуры

Датчик температуры θ доставляет информацию о достижении пороговых значений θ_{min}, θ_{max} и, тем самым, выделяет три ситуации (события):

1 — $\theta < \theta_{min}$ («Прохладно»);
2 — $\theta_{min} \le \theta < \theta_{max}$ («Комфортно»);
3 — $\theta \ge \theta_{max}$ («Тепло»).

Управляющее воздействие u — напряжение, приложенное к нагревательному элементу, может принимать два значения 0 В (выключен) и 220 В (включен).

Таким образом, система функционирует в условиях дефицита информации о состоянии объекта и минимального разнообразия управляющих воздействий. Алгоритм принятия решений выразим в форме правил:

1. ЕСЛИ («Прохладно») ТО (Нагреватель включен);
2. ЕСЛИ («Тепло») ТО (Нагреватель выключен);
3. ЕСЛИ («Комфортно») И (Нагреватель включен) ТО (Нагреватель включен);
4. ЕСЛИ («Комфортно») И (Нагреватель выключен) ТО (Нагреватель выключен).

На рис. 2.2 изображена статическая характеристика (СХ) релейного регулятора с гистерезисом для пороговых значений температуры $\theta_{min} = 21$℃,

$\theta_{max} = 23℃$.

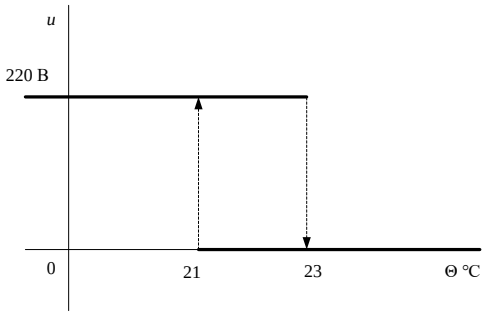


Рис. 2.2. СХ релейного регулятора

Обратим внимание на важное обстоятельство — если значения температуры принадлежат интервалу двузначности СХ, т. е. лежат между двумя пороговыми значениями, воздействие на объект определяется не только входом, но и состоянием объекта до того, как значение температуры вошло в этот интервал. Таким образом, этот безынерционный НЭ обладает памятью — значение его выхода определяется не только значением входа в тот же момент, но также и предысторией.

Неоднозначность кусочно-постоянных СХ создает сложности формального описания релейных и логических алгоритмов управления моделями «вход-выход». Графики СХ приходится дополнять пояснениями о значениях функции в зонах неоднозначности. Необходимы иные формы представления логических алгоритмов.

Механизм вывода в четкой логике (*Crisp-Logic Inference System*) образован последовательным соединением детектора событий, блока логики и декодера (рис. 2.3), исполняющих информационную, алгоритмическую и исполнительную функции [5].

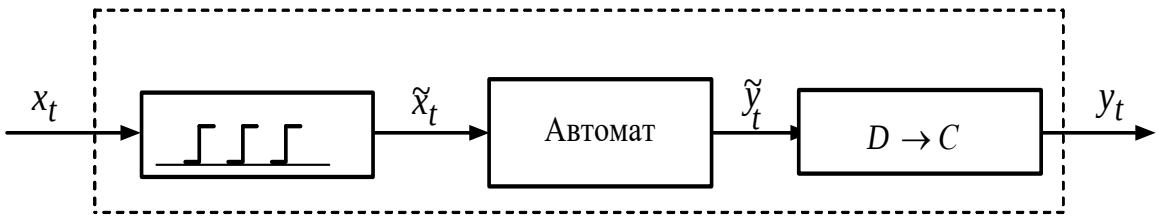


Рис. 2.3. Последовательность вывода в четкой логике

Отображение непрерывного сигнала x_t на множество символов $\tilde{x}_t$ реализуется на пороговых элементах, разбивающих множества действительных значений на конечное число подмножеств. Пороги являются *четкими*, т. е. каждое значение входного сигнала может принадлежать строго одному под-

множеству. Преобразователь $D \rightarrow C$ (декодер) сопоставляет выходным символам автомата $\tilde{y}_t$ действительные значения. Сигнал выхода y_t представляет собой кусочно-постоянную функцию времени, уровень которой может изменяться с появлением нового символа на входе преобразователя.

На рис. 2.3 переменные со знаком «тильда» (волнистая черта сверху) относятся к конечным множествам. Нижний индекс *t* означает, что рассматривается непрерывное время. Следовательно, входы, состояния и выходы автомата могут изменяться в любое время (асинхронный автомат).

Моделью логических устройств (символьных преобразователей), входы, состояния и выходы которых принимают значения из конечных множеств (символов), является *автомат*. Конечный автомат задается пятеркой $< S, Y, X, \delta, \lambda >$, где S, Y, X — множества состояний, выходов и входов автомата; δ— функция переходов; λ— функция выходов. Функционирование автомата можно описать в терминах «вход-состояние-выход»

$$s' = \delta(\tilde{s}, \tilde{x}); \tilde{s}_0,$$
$$\tilde{y} = \lambda(\tilde{s}, \tilde{x}).$$

Здесь $\tilde{s}'$ — последующее состояние, зависящее от предыдущего состояния $\tilde{s}$ и от входа $\tilde{x}$; $\tilde{s}_0$ — начальное состояние автомата.

Механизмы вывода в четкой логике описывают преобразования без временной памяти. Вместе с тем, внутренние состояния автомата позволяет учитывать «пространственную» память, выражаемую в многозначности преобразования — зависимости выхода от предыстории.

Системы вывода в четкой логике — частный случай механизмов логического вывода на нечетких множествах (*Fuzzy-Logic Inference Systems*) [6]. Они позволяют адекватно описать многозначные кусочно-постоянные преобразователи с разрывами первого рода и со сложной логикой переходов между ветвями СХ. Потенциальные возможности логических алгоритмов управления делают перспективными их применение в сложных приложениях.

Гибридная модель релейной системы регулирования температуры. Релейные системы регулирования являются примерами систем, в которых некоторые сигналы принимают значения на конечных множествах. В системе автоматической стабилизации температуры (см. рис. 2.1) сосуществуют непрерывные переменные (температура) и символьные последовательности, представляющие события (ситуации). Непрерывная часть системы описыва-

ется дифференциальными уравнениями, а моделью символьных преобразователей (логической части) являются конечные автоматы. Разнородные модели взаимодействуют посредством интерфейса, состоящего из детектора событий $C \to D$ (измерительной части) и исполнительной части $D \to C$. Такие системы называют *гибридными* [7]. Гибридная модель типа Нероде—Кона (A. Nerode, W. Kohn) системы управления с обратной связью изображена на рис. 2.4.

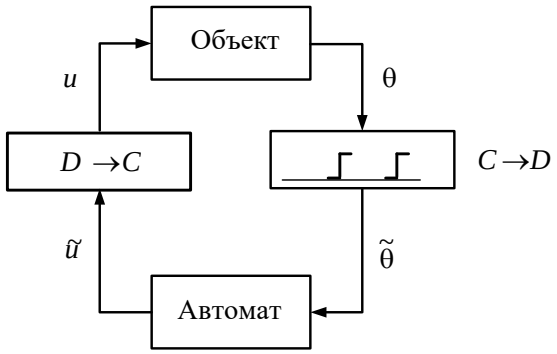


Рис. 2.4. Гибридная модель системы управления

Пусть непрерывная часть системы (см. рис. 2.1) описана системой линейных дифференциальных уравнений второго порядка:

$$\begin{aligned} T_1 \frac{dv}{dt} + v &= u, \\ T_2 \frac{d\theta}{dt} + \theta &= kv, \end{aligned} \tag{2.1}$$

где $T_1 = 20$ мин; $T_2 = 4$ мин; $k = 40 / 220$ °C/В.

Интерфейс $C \to D$ образован двумя пороговыми элементами с выходами в виде нуля или единицы. На выходе этого детектора событий имеем бинарный вектор с двумя компонентами $[\tilde{\theta}_1 \ \tilde{\theta}_2]'$, кодирующий три события. Примем для примера следующие значения: $\theta_{\min} = 21\ ^\circ\text{C}$; $\theta_{\max} = 23\ ^\circ\text{C}$. Эта часть интерфейса реализует следующее отображение: $\tilde{\theta} = [0 \quad 0]'$, если температура ниже 21 °C; $\tilde{\theta} = [1 \quad 0]'$, если температура выше или равна 21 °C, но ниже 23 °C; $\tilde{\theta} = [1 \quad 1]'$, если температура выше или равна 23 °C.

Интерфейс $D \to C$ подает на нагреватель напряжение 0 В или 220 В в зависимости от выхода автомата $\tilde{u}$ — 0 или 1.

Приведенная ранее логика управляющего устройства предполагает, что

автомат имеет память. Для описания состояния автомата достаточно ввести переменную $\tilde{s}$, принимающую два значения: 0, если нагреватель выключен, и 1, если включен (объем памяти автомата — 1 бит). Тогда логику управления можно представить в виде правил:

1. ЕСЛИ ($\tilde{\theta} = [0\ \ 0]'$) ТО ($\tilde{u} = 1$) И ($\tilde{s} = 1$).
2. ЕСЛИ ($\tilde{\theta} = [1\ \ 1]'$) ТО ($\tilde{u} = 0$) И ($\tilde{s} = 0$).
3. ЕСЛИ ($\tilde{\theta} = [1\ \ 0]'$) И ($\tilde{s} = 1$) ТО ($\tilde{u} = 1$) И ($\tilde{s} = 1$).
4. ЕСЛИ ($\tilde{\theta} = [1\ \ 0]'$) И ($\tilde{s} = 0$) ТО ($\tilde{u} = 0$) И ($\tilde{s} = 0$).

Реализующий эту логику автомат Мура можно описать пятеркой — тремя множествами: $S = \{0; 1\}$; $\Theta = \{[0\ \ 0]'; [1\ \ 0]'; [1\ \ 1]'\}$; $U = \{0; 1\}$, а также функциями перехода и выхода (6.3), которые зададим в виде таблицы.

Функции перехода автомата

$\tilde{\theta}_1$	$\tilde{\theta}_2$	$\tilde{s}$	$\tilde{u}$	$\tilde{s}'$
0	0	0	1	1
0	0	1	1	1
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	0	0

В таблице шесть строк вместо четырех правил, так как правило 1 объединяет строки 1 и 2 таблицы, а правило 2 — строки 5 и 6.

При трехмерном бинарном векторе $(\tilde{\theta}_1\ \ \tilde{\theta}_2\ \ \tilde{s})'$ число состояний равно: $2^3 = 8$, однако не все они актуальны. К невозможным ситуациям относятся следующие: [0 1 0]' и [0 1 1]', так как второе пороговое значение температуры не может наступить ранее первого. Соответствующие строки в таблице отсутствуют.

Логика автомата наглядно изображается в виде диаграммы графа; для автомата Мура релейного регулятора диаграмма изображена на рис. 2.5.

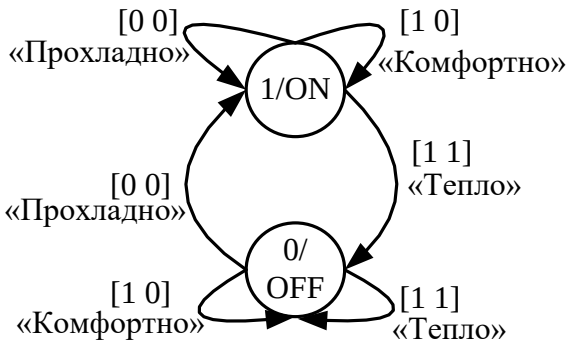


Рис. 2.5. Диаграмма графа конечного автомата типа Мура

Компьютерная имитация релейной системы регулирования температуры. Основным методом исследования систем управления по гибридным моделям является компьютерная имитация.

Для имитации релейной системы воспользуемся наработками из прошлых работ. Преобразуем систему дифференциальных уравнений (2.1) в форму Коши:

$$\frac{dv}{dt} = \frac{u-v}{T_1};$$

$$\frac{d\theta}{dt} = \frac{kv-\theta}{T_2},$$

где: $T_1 = 20$ мин; $T_2 = 40$ мин; $k = 40/220 \frac{℃}{В}$. И промоделируем поведение системы (рис. 2.6).

```
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt

T_1 = 20
T_2 = 40
k = 40/220
state = 0
def ode_sys_1_control(x):
    try:
        ode_sys_1_control.state
    except:
        ode_sys_1_control.state = 0
    c_1 = x >= 21
    c_2 = x >= 23

    if not c_1 and not c_2:
        ode_sys_1_control.state = 1
```

```
            return 1
        elif c_1 and c_2:
            ode_sys_1_control.state = 0
            return 0
        elif c_1 and not c_2 and ode_sys_1_control.state==1:
            ode_sys_1_control.state = 1
            return 1
        elif c_1 and not c_2 and ode_sys_1_control.state==0:
            ode_sys_1_control.state = 0
            return 0

def ode_sys_1(x, t):
    """
    Функция, рализующая систему из первого примера
    """
    v, theta = x

    u = 220 * ode_sys_1_control(theta)
    dv_dt = (u - v)/T_1
    dtheta_dt = (k*v - theta) / T_2
    return (dv_dt, dtheta_dt)


y0 = (0, 0)
t = np.linspace(0, 200, 1001)
sol_ode_sys_1 = odeint(ode_sys_1, y0, t)

plt.plot(t, sol_ode_sys_1[:,1])
plt.grid()
```

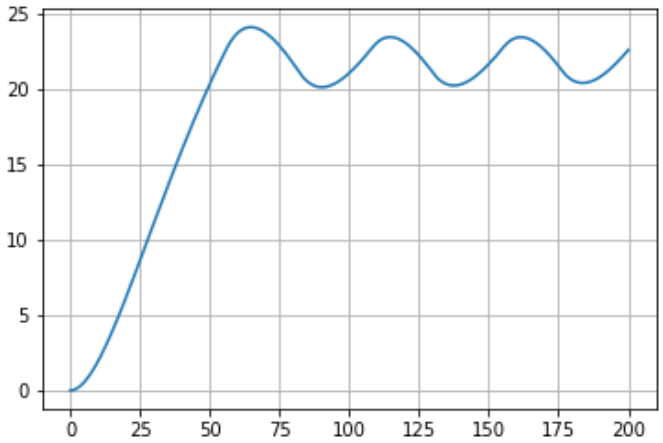


Рис. 2.6. График изменения температуры

Задание:

- определить частоту и амплитуду установившихся колебаний;
- заменить НЭ регулятора на идеальное реле и повторить анализ.

Лабораторная работа 3. Нейросетевая аппроксимация СХ НЭ

Полностью определенные модели многих объектов и элементов систем управления не удается получить аналитическим способом. Тогда привлекаются процедуры идентификации, позволяющие оценивать параметры путем обработки данных экспериментов — активных (специально спланированных и реализованных) или пассивных (в режиме нормальной эксплуатации).

Нейросетевые модели НЭ рекомендуются в ситуации, когда тип нелинейности не известен, но имеются данные о ее входах и выходах. Искусственные нейронные сети (ИНС) представляют собой универсальные подстраиваемые аппроксиматоры безынерционных многомерных НЭ с однозначной СХ.

На рис. 3.1 изображена модель нейрона. Выходная переменная y является нелинейной функцией взвешенной суммы входных переменных

$$y = F(b + \sum_{i=1}^{n} w_i x_i),$$

где: x_i — сигнал на входе нейрона, w_i — вес i-го входа, b — смещение; $F(s)$ — функция активации. Параметры нейрона — смещение b и весовые коэффициенты w_i можно настраивать, добиваясь требуемой зависимости выхода от входов.

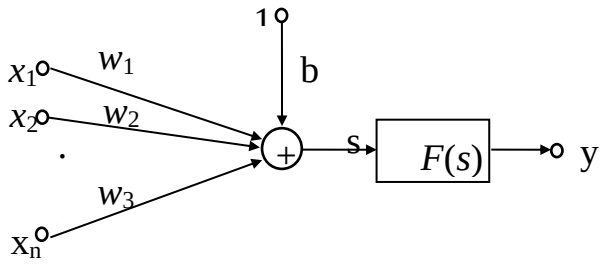


Рис. 3.1. Модель нейрона

Широкие аппроксимирующие возможности достигаются, если нейроны образуют сети определенной архитектуры. На рис. 3.2 изображена однонаправленная двухслойная нейронная сеть с тремя входами и двумя выходами. Весовые коэффициенты и смещения первого (скрытого нелинейного) слоя сети упорядочены в матрицы $\mathbf{W}_1$, $\mathbf{b}_1$, а второго линейного — в матрицы $\mathbf{W}_2$, $\mathbf{b}_2$.

Разработан ряд алгоритмов обучения многослойных сетей — настройки весов и смещений из условия минимизации суммы квадратов отклонений

выхода сети от выхода моделируемого объекта [8].

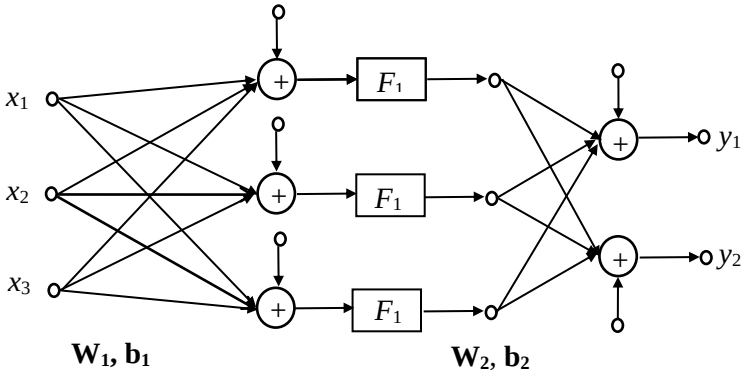


Рис. 3.2. Двухслойная нейронная сеть

Нейросетевая аппроксимация СХ многомерного НЭ на примере ГДХ центробежного компрессора природного газа.

Зависимость степени сжатия газа $\varepsilon = p_{\text{вых}}/p_{\text{вх}}$ от объемного расхода Q и относительной частоты вращения компрессора n

$$\varepsilon = F_1(Q, n) \qquad (3.1)$$

задается так называемыми газодинамическими характеристиками (ГДХ). Функция двух аргументов (3.1) задает СХ НЭ с двумя входами и одним выходом. Построенные по экспериментальным данным графики функции (3.1) приводятся в справочниках заводов-изготовителей (см., например, [9]). В качестве параметра семейства кривых обычно выступает относительная частота вращения ротора компрессора n.

Графики ГДХ наглядно отражают взаимосвязь основных переменных процесса компримирования газа, однако для разработки компьютерных (имитационных) моделей их следует привести к иной форме.

Существует множество способов аппроксимации нелинейных характеристик.

Методика построения нейросетевой модели складывается из следующих этапов:

1 — подготовка обучающих данных;

2 — выбор типа и архитектуры ИНС;

3 — обучение сети;

4 — анализ сети.

1. Подготовка обучающих данных на входе (наборы значений расхода Q и n) и выходе (набор значений степени сжатия ε) требует перевода ГДХ в табличную форму задания, т. е. оцифровки графиков ГДХ. Обычно кривые семейства помечены частотами вращения ротора компрессора n.

Ручная оцифровка кривых ГДХ дает следующие наборы данных.

n = 0.70: Q = [245 280 340 400 440 465]; ε = [1.145 1.140 1.13 1.11 1.09 1.08];
n = 0.75: Q = [265 320 360 400 440 460 495]; ε = [1.166 1.16 1.15 1.14 1.12 1.11 1.08];
n = 0.80: Q = [280 320 360 420 480 520 530]; ε = [1.19 1.185 1.18 1.16 1.13 1.105 1.095];
n = 0.85: Q = [300 360 400 440 480 520 560]; ε = [1.22 1.21 1.20 1.185 1.17 1.144 1.11];
n = 0.90: Q = [315 360 400 440 480 520 560 600]; ε = [1.246 1.24 1.232 1.22 1.205 1.183 1.155 1.12];
n = 0.95: Q = [330 380 420 460 500 540 580 620 630]; ε = [1.276 1.27 1.26 1.25 1.23 1.21 1.18 1.145 1.135];
n = 1.00: Q = [350 420 450 500 540 580 620 665]; ε = [1.31 1.3 1.29 1.27 1.25 1.23 1.195 1.15];
n = 1.05: Q = [370 420 480 525 550 580 620 660 695]; ε = [1.345 1.34 1.32 1.30 1.29 1.27 1.245 1.21 1.165];
n = 1.10: Q = [385 420 460 490 510 560 600 640 680 730]; ε = [1.382 1.38 1.37 1.36 1.35 1.33 1.305 1.275 1.24 1.18]

Построенные по этим данным кривые ГДХ компрессора типа 520-12-1 (рис. 3.3) показывают достаточную корректность ручной оцифровки «альбомных» ГДХ.

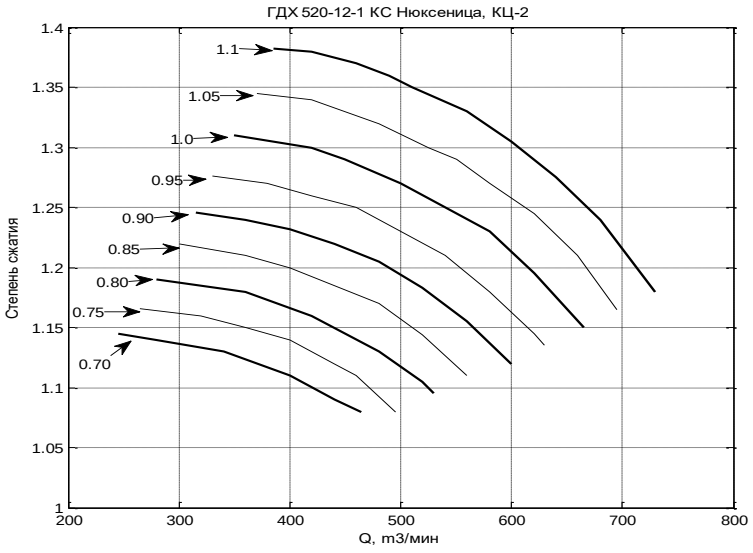


Рис. 3.3. ГДХ компрессора

2. Выберем инструмент обучения ИНС — MATLAB/*NeuralFitting app.* Программа вызывается командой

```
>>nftool.
```

Вводим данные для обучения сети в рабочее пространство MATLAB:

>> n =[0.7 0.7 0.7 0.7 0.7 0.7 0.75 0.75 0.75 0.75 0.75 0.75 0.75 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.85 0.85 0.85 0.85 0.85 0.85 0.85 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.95 0.95 0.95 0.95 0.95 0.95 0.95 0.95 0.95 1 1 1 1 1 1 1 1.05 1.05 1.05 1.05 1.05 1.05 1.05 1.05 1.05 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1]; % частота вращения

>> Q = [245 280 340 400 440 465 265 320 360 400 440 460 495 280 320 360 420 480 520 530 300 360 400 440 480 520 560 315 360 400 440 480 520 560 600
330 380 420 460 500 540 580 620 630 350 420 450 500 540 580 620 665
370 420 480 525 550 580 620 660 695 385 420 460 490 510 560 600 640 680 730]; % объемный расход

Сформируем вектор входа: 2 строки по 71 элементов

>> In=[n; Q];

Выход: строка из 71 элемента

>>Epsilon = [1.145 1.140 1.13 1.11 1.09 1.08 1.166 1.16 1.15 1.14 1.12 1.11 1.08 1.19 1.185 1.18 1.16 1.13 1.105 1.095 1.22 1.21 1.20 1.185 1.17 1.144 1.11 1.246 1.24 1.232 1.22 1.205 1.183 1.155 1.12 1.276 1.27 1.26 1.25 1.23 1.21 1.18 1.145 1.135 1.31 1.3 1.29 1.27 1.25 1.23 1.195 1.15 1.345 1.34 1.32 1.30 1.29 1.27 1.245 1.21 1.165 1.382 1.38 1.37 1.36 1.35 1.33 1.305 1.275 1.24 1.18];

Выбирается двухслойная feedforward сеть с линейным выходным нейроном и семью сигмоидными скрытыми нейронами.

3. Обучение: *Levenberg-Marquardt backpropagation algorithm*.

4. Анализ нейросетевой модели компрессора проводится на компьютерном имитаторе — Simulink-модели обученной сети (рис. 3.4).

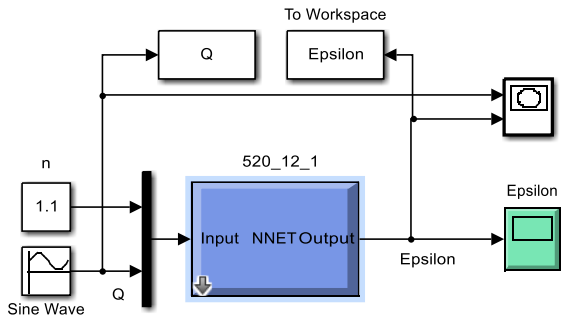


Рис. 3.4. Нейросетевой имитатор (Simulink-модель) компрессора

На первый вход обученной сети подается очередное значение относительной частоты вращения компрессора *n*, а на второй вход — гармонический сигнал со смещением *Q*. Смещение и амплитуда сигнала подбираются таким образом, чтобы в окне блока XY-Graph получилась соответствующая кривая семейства ГДХ. Рекомендуется ориентироваться на кривые исходной ГДХ (см. рис. 3.1). При удачном подборе вводится команда построения гра-

фика кривой

```
>> plot (Q,Epsilon)
```

Кривую следует снабдить меткой, отвечающей частоте вращения.

Примечание: для сохранения на графике предыдущих кривых семейства не забудьте ввести команду

```
>> hold on
```

Лабораторная работа 4. Метод фазового пространства

Рассмотрим математическую модель динамической системы в форме дифференциальных уравнений первого порядка, разрешенных относительно производных (форма Коши)

$$\frac{dv_i}{dt} = \varphi_i(v_1,...,v_n); \quad i = 1,...,n. \tag{4.1}$$

В правые части уравнений (4.1) время t явно не входит, такие системы называют *автономными*. На автономные системы не действуют внешние силы. Автономные системы *стационарны* — их свойства неизменны во времени.

Состояние конечномерной динамической системы (4.1) характеризуется вектором в n-мерном пространстве $\mathbf{v}(t) = (v_1(t),...,v_n(t))^T$. *Начальное состояние* $\mathbf{v}(0)$ автономной системы полностью определяет ее поведение для $t > 0$ независимо от предыстории, т. е. того, каким путем система пришла в это состояние.

Геометрическое место точек конца вектора $\mathbf{v}(t)$ при $t \geq 0$ образует *траекторию состояния* — образ поведения при конкретном начальном состоянии. На рис. 4.1 иллюстрируется траектория трехмерной системы.

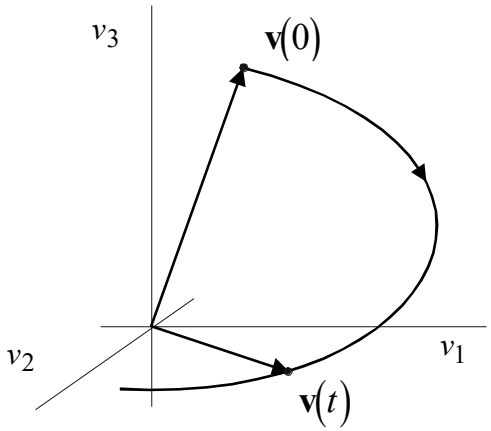


Рис. 4.1. Траектория изменения состояния

Метод фазовой плоскости. Хотя геометрическая интерпретация метода пространства состояний распространяется на системы любого порядка, важное его преимущество — наглядность — наиболее ярко проявляется в случае систем второго порядка, когда состояния системы представляются точками на *фазовой плоскости*. Следует добавить, что нелинейные модели второго порядка позволяют выявлять многие принципиальные особенности поведе-

ния динамических систем; это определяет методическое, теоретическое и практическое значение метода фазовой плоскости. Метод фазовой плоскости дает возможность изобразить качественную картину всей совокупности свободных движений (процессов) для выбранной области начальных условий (состояний).

Пусть заданы уравнения системы второго порядка

$$\begin{aligned} \frac{dv_1}{dt} &= \varphi_1(v_1, v_2); \\ \frac{dv_2}{dt} &= \varphi_2(v_1, v_2). \end{aligned} \tag{4.2}$$

Для построения фазовой траектории с начальными условиями $v_{10} = v_1(0)$ и $v_{20} = v_2(0)$ из решения системы (4.2)

$$v_1(t; v_{10}, v_{20}); \quad v_2(t; v_{10}, v_{20})$$

следует исключить время t, т. е. получить зависимость $v_2(v_1; v_{10}, v_{20})$.

Через каждую точку фазового пространства при условии однозначности функции проходит только одна фазовая траектория. Единственность нарушается в *особых точках*, отвечающих точкам равновесия

$$\varphi_1(v_1, v_2) = 0;$$

$$\varphi_2(v_1, v_2) = 0.$$

На фазовой плоскости существует несколько основных типов особых точек (типов поведения в окрестности положения равновесия):

- устойчивый узел;
- неустойчивый узел;
- седло;
- устойчивый фокус;
- неустойчивый фокус;
- центр.

Особые точки являются частным случаем аттракторов (от англ. to attract — притягивать) — компактных подмножеств фазового пространства динамической системы, к которым стремятся все траектории из некоторой окрестности. Аттракторами могут быть замкнутые траектории (предельные циклы) или ограниченные области с неустойчивыми траекториями внутри (как у «странного» аттрактора).

Пример построения фазового портрета. Рассмотрим систему дифференциальных уравнений математического маятника:

$$\begin{aligned} \frac{d\theta}{dt} &= \omega; \\ \frac{d\omega}{dt} &= -c\ sin(\theta). \end{aligned} \tag{4.3}$$

Воспользуемся библиотеками numpy, scipy и matplotlib языка Python. Блок импорта модели выглядит следующим образом:

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
```

Определим функцию, отвечающую за численное решение обыкновенных дифференциальных уравнений (ОДУ), следующего вида:

```
def ode(y, t, c):
    theta, omega = y
    dydt = [omega, - c*np.sin(theta)]
    return dydt
```

Аргументы функции:

- y — вектор переменных состояния,
- t — время,
- c — параметр ОДУ (может быть любое число).

Функция возвращает вектор производных и является аналогом системы ДУ первого порядка. Далее необходимо реализовать функцию для получения решения ОДУ с заданными начальными условиями:

```
def calcODE(args, y0, dy0, ts = 10, nt = 101):
    y0 = [y0, dy0]
    t = np.linspace(0, ts, nt)
    sol = odeint(ode, y0, t, args)
    return sol
```

Аргументы функции:

- args — параметры ОДУ (см. определение функции с ОДУ),
- y0 — начальные условия для первой переменной состояния,
- dy0 — начальные условия для второй переменной состояния (или в нашем случае ее производной),
- ts — длительность решения,
- nt — число шагов в решении (= время интегрирования * шаг времени).

В 3-й строке формируется вектор временных отсчетов. В 4-й строке вызывается функция решения ОДУ.

Для построения фазового портрета необходимо произвести решения ОДУ с различными начальными условиями (вокруг интересующей точки). Для реализации также напишем функцию:

```
     def drawPhasePortrait(args, deltaX = 1, deltaDX = 1, startX
= 0,  stopX = 5, startDX = 0, stopDX = 5, ts = 10, nt = 101):
         for y0 in range(startX, stopX, deltaX):
                 for dy0 in range(startDX, stopDX, deltaDX):
                     sol = calcODE(args, y0, dy0, ts, nt)
                     plt.plot(sol[:, 0], sol[:, 1], 'b')
         plt.xlabel('x')
         plt.ylabel('dx/dt')
         plt.grid()
         plt.show()
```

Аргументы функции:

- args — параметры ОДУ (см. шаг 1),
- deltaX — шаг начальных условий по горизонтальной оси (переменной состояния),
- deltaDX — шаг начальных условий по вертикальной оси (производной переменной состояния),
- startX — начальное значение интервала начальных условий,
- stopX — конечное значение интервала начальных условий,
- startDX — начальное значение интервала начальных условий,
- stopDX — конечное значение интервала начальных условий,
- ts — длительность решения,
- nt — число шагов в решении (= время интегрирования * шаг времени).

Во вложенных циклах (строки 3 и 4) происходит перебор начальных условий дифференциального уравнения. В теле этих циклов (строки 5 и 6) происходит вызов функции решения ОДУ с заданными НУ и вывод фазовой траектории полученного решения. Далее производятся действия:

- строка 7 — задается название оси X
- строка 9 — задается название оси Y
- строка 10 — выводится сетка на графике
- строка 11 — вывод графика (рендер)

Запустить построение фазового портрета можно следующим образом:

```
    c = 0.25 # Параметр ОДУ (длина маятника)
```

```
    args=(c, )
    drawPhasePortrait(args)
    drawPhasePortrait(args, 1, 1, -5, 5, -4, 4, ts = 5, nt =
301)
```

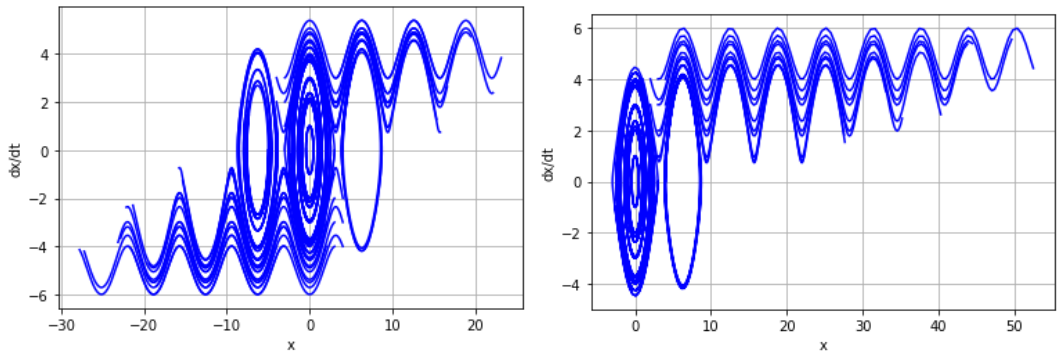


Рис. 4.3. Фазовый портрет математического маятника

В процессе выполнения лабораторной работы необходимо модифицировать функцию построения фазового портрета для получения более наглядного графика.

Фазовый портрет маятника с учетом вязкого трения. Второе уравнение системы (4.3) дополним членом $-b\omega$, учитывающим зависимость момента сопротивления от угловой скорости ω

$$\begin{aligned} \frac{d\theta}{dt} &= \omega; \\ \frac{d\omega}{dt} &= -b\omega - mgl\,sin(\theta), \end{aligned} \tag{4.4}$$

где: m — масса, м; l — длина маятника, м; $g = 9.81$ — ускорение свободного падения, м/с^2; b — коэффициент вязкого трения.

Назначить параметры в соответствии с номером варианта:

- $m = 0.2 * N_{var}$;
- $l = 5 / N_{var}$;
- $b = 0.1 + 0.015 * N_{var}$.

Построить фазовые портреты и сравнить поведение маятников (4.3) и (4.4). Классифицировать особые точки фазовых портретов.

Пример ячейки кода для построения фазового портрета (рис. 4.4) приведен ниже.

```
    def ode(Y, t, b):
        x, y = Y
        dydt = [2*x+y, x-3*y]
        return dydt
```

```
    drawPhasePortrait(args, 1, 1, -5, 5, -4, 4, ts = 0.5, nt =
301)
```

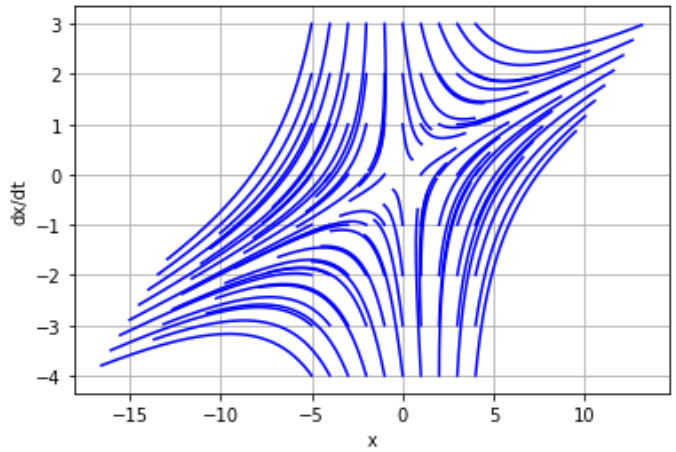


Рис. 4.4. Пример фазового портрета (особая точка типа «седло»)

Построение фазового портрета системы, представленной в форме структурной схемы (рис. 4.5).

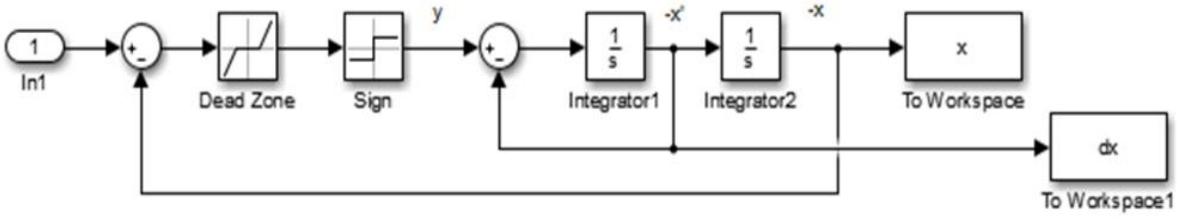


Рис. 4.5. Структурная схема системы

Ширину зоны нечувствительности выбрать в соответствии с номером варианта задания по формуле

$$b = 0.2\,N_{var} + 0.2.$$

Указание: вначале составить систему дифференциальных уравнений по структурной схеме.

Построение фазового портрета осциллятора Ван дер Поля. Дифференциальные уравнения генератора колебаний (осциллятора) предложены голландским инженером и физиком Бальтазаром ван дер Полем. Им были найдены устойчивые колебания, которые были названы релаксационными, известные как «предельные циклы». Уравнение Ван дер Поля применяется в физике и других областях науки и техники. Оно также было использовано в

сейсмологии для моделирования геологических разломов. Например, в биологии создана модель Фитц Хью-Нагумо

$$\begin{cases} \dfrac{dx}{dt} = y; \\ \dfrac{dy}{dt} = \mu(1 - x^2)y - x. \end{cases}$$

Задание. Постройте фазовый портрет с коэффициентами μ равными:

- N_{var},
- $N_{var}/2$,
- $2N_{var}$.

Проанализируйте портрет, оцените влияние параметра μ на динамику системы.

Построение фазового портрета аттрактора Лоренца. Аттрактор Лоренца — компактное инвариантное множество $\boldsymbol{L}$ в трехмерном фазовом пространстве гладкого потока, которое имеет определенную сложную топологическую структуру и является асимптотически устойчивым. Аттрактор устойчив по Ляпунову — все траектории из некоторой окрестности $\boldsymbol{L}$ стремятся к $\boldsymbol{L}$ при $\boldsymbol{t} \to \infty$.

Система дифференциальных уравнений Лоренца записывается так:

$$\begin{cases} x' = \sigma(y - x); \\ y' = x(r - z) - y; \\ z' = xy - bz \end{cases}$$

для параметров: $\sigma = 10$, $r = 28$, $b = 8/3$ и начальных условий $x(0) = 1$, $y(0) = 0, z(0) = 0$.

Для удобства анализа и работы с 3D-графиком, этот пункт рекомендуется предварительно выполнить в отдельном сценарии.

Пример построения 3D фазового портрета в Python:

- загружаем необходимые зависимости

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

- реализуем функцию ОДУ

```
def ode(y, t, sigma, r, b):
    x,y,z = y
    dxdt = sigma * (y-x)
    dydt = x * (r - z) - y
    dzdt = x*y - b*z
    return [dxdt, dydt, dzdt]
```

- реализуем функцию, реализующую запуск вычисления ОДУ

```
def calcODE(args, x, y, z, ts = 10, nt = 101):
    y0 = [x, y, z]
    t = np.linspace(0, ts, nt)
    sol = odeint(ode, y0, t, args)
    return sol
```

- реализуем функцию, реализующую расчет и рендер 3D фазового портрета и его проекций

```
def drawPhasePortrait3D(args,
                        deltaX = 1, deltaY = 1, deltaZ = 1,
                        startX = 0,  stopX = 5,
                        startY = 0, stopY = 5,
                        startZ = 0, stopZ = 5,
                        ts = 10, nt = 101):
    fig = plt.figure()
    ax = fig.add_subplot(2, 2, 1, projection='3d')
    ax.set_title("3D")
    plt.subplot(2, 2, 2)
    plt.title("X-Y")
    plt.grid()
    plt.subplot(2, 2, 3)
    plt.title("X-Z")
    plt.grid()
    plt.subplot(2, 2, 4)
    plt.title("Y-Z")
    plt.grid()

    for x in range(startX, stopX, deltaX):
        for y in range(startY, stopY, deltaY):
                for z in range(startZ, stopZ, deltaZ):
                    sol = calcODE(args, x, y, z, ts, nt)

                    ax.plot(sol[:, 0], sol[:, 1], sol[:, 2])
                    plt.subplot(2, 2, 2)
                    plt.plot(sol[:, 0], sol[:, 1])
                    plt.subplot(2, 2, 3)
                    plt.plot(sol[:, 0], sol[:, 2])
                    plt.subplot(2, 2, 4)
                    plt.plot(sol[:, 1], sol[:, 2])
```

```
    plt.show()
```

- задаем значения параметров и запускаем вычисления

```
sigma = 10
r = 28
b = 8/3
args=(sigma, r, b)
drawPhasePortrait3D(args,
                        deltaX = 4, deltaY = 4, deltaZ = 4,
                        startX = -10,  stopX = 10,
                        startY = -10, stopY = 10,
                        startZ = -10, stopZ = 10,
                        ts = 10, nt = 1001)
```

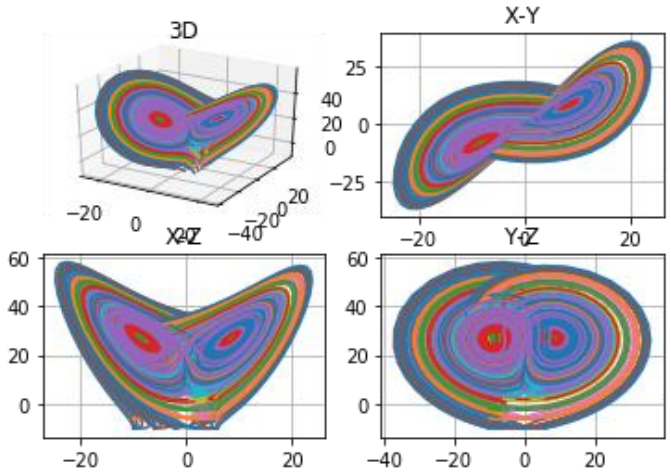


Рис. 2.4. Фазовый портрет аттрактора Лоренца

Модифицируйте код построения фазового портрета для более удобного анализа. Попробуйте изменить параметры и проанализируйте изменения поведения системы.

Варианты задания. По заданию преподавателя построить фазовые портреты нелинейных систем второго порядка. Классифицировать типы особых точек предложенных систем.

Система 1

$$\begin{cases} x' = -4x + 0.1\,N_{var} * x^2 - 4y \\ y' = 1.5x + y - 0.2\ N_{var} * y\text{^}3 \end{cases}$$

Система 2

$$\begin{cases} x' = x + 0.5y - 0.1\ N_{var} * y^2 \\ y' = 0.5x - 0.2\,N_{var}\,x^2 + y \end{cases}$$

Система 3

$$\begin{cases} x' = 2x + 0.2\ N_{var} * x^2 + y - 0.1\ N_{var} * y^2 \\ y' = x - 3y \end{cases}$$

Система 4

$$\begin{cases} x' = -0.1\ N_{var} * x^2 + 2y \\ y' = -3x - y \end{cases}$$

Система 5

$$\begin{cases} x' = 0.1x - 4y \\ y' = 4x - 0.2\ N_{var} * x^2 + 0.1y \end{cases}$$

Система 6

$$\begin{cases} x' = x - 0.1\ N_{var} * x^2 - 4y + 0.3\ N_{var} * y^2 \\ y' = 2x + 0.2\ N_{var} * x^2 - y - 0.3\ N_{var} * y^3 \end{cases}$$

Указание. Задание выполняется в следующей последовательности

- найти положения равновесия,
- линеаризовать уравнения,
- получить характеристический полином
- вычислить корни.

Лабораторная работа 5. Анализ устойчивости положения равновесия

Первый метод Ляпунова позволяет судить об устойчивости изолированного положения равновесия по линеаризованным уравнениям. Метод основан на утверждениях:

- если собственные значения (с.з.) линеаризованной системы имеют отрицательные действительные части (линеаризованная система асимптотически устойчива), то положение равновесия нелинейной системы устойчиво «в малом»;
- если среди с.з. линеаризованной системы имеются «правые», то положение равновесия нелинейной системы неустойчиво;
- если имеются некратные с.з. на мнимой оси, а остальные — «левые», то в этом критическом случае по линеаризованной модели нельзя судить об устойчивости положения равновесия нелинейной системы.

Пример исследования устойчивости положения равновесия осциллятора Ван дер Поля, описываемого дифференциальным уравнением второго порядка:

$$\Phi(y, y', y'') = y'' - \mu(1 - y^2)y' + y = 0.$$

Система имеет единственное положение равновесия $y^* = y' = y'' = 0$. Линеаризованное для малых отклонений уравнение запишется так:

$$a_2 y'' + a_1 y' + a_0 y = 0,$$

где:

$$a_0 = \left.\frac{\partial \Phi}{\partial y}\right|_* = (2\mu\, y' y + 1)\big|_* = 1;$$

$$a_1 = \left.\frac{\partial \Phi}{\partial y'}\right|_* = -\mu(1 - y^2)\big|_* = -\mu;$$

$$a_2 = \left.\frac{\partial \Phi}{\partial y''}\right|_* = 1.$$

Характеристический полином уравнения

$$A(s) = s^2 - \mu s + 1$$

имеет следующие корни

$$s_{1,2} = \frac{\mu \pm \sqrt{\mu^2 - 4}}{2}.$$

Положение равновесия не устойчиво, если $\mu > 0$. При значении $\mu \geq 2$ на фазовой плоскости в начале координат имеется особая точка типа «неустойчивый узел». При значениях $0 < \mu < 2$ имеет место особая точка типа «неустойчивый фокус».

Фазовый портрет осциллятора Ван дер Поля имеет устойчивый предельный цикл, которому соответствуют автоколебания в нелинейной системе (см. Лабораторную работу 4).

Задание 1. Проанализируйте первым методом Ляпунова нелинейный осциллятор, заданный системой дифференциальных уравнений.

$$\frac{dv_1}{dt} = v_2;$$

$$\frac{dv_2}{dt} = -v_1 - v_2^3.$$

Найдите координаты положения равновесия: $v_1 = const;\ v_2 = const$ из условий $\frac{dv_1}{dt} = 0;\ \frac{dv_2}{dt} = 0$.

Составьте линеаризованные уравнения для малых отклонений переменных от положения равновесия (подсказка: для малых отклонений слагаемое v_2^3 во втором уравнении представляет малую третьего порядка и может быть отброшено).

Запишите матрицу состояний линеаризованного уравнения

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

и вычислите ее с.з. с помощью соответствующего инструмента. Можно предварительно записать характеристический полином матрицы

$$A(s) = \det(s\mathbf{I} - \mathbf{A}) = \det\begin{bmatrix} s & -1 \\ 1 & s \end{bmatrix} = s^2 + 1$$

и вычислить его корни. Корни полинома $A(s)$ равны с.з. матрицы $\mathbf{A}$.

Убедитесь, что корни (с.з.) лежат на мнимой оси. В этом критическом случае по линеаризованной модели нельзя судить об устойчивости положения равновесия нелинейной системы.

В данном примере первый метод Ляпунова не позволяет судить об устойчивости положения равновесия. Необходимо обратиться к второму методу Ляпунова [1], [2].

Постройте фазовый портрет нелинейного осциллятора. Что можно сказать об устойчивости положения равновесия?

Задание 2. Проанализируйте первым методом Ляпунова систему, заданную структурной схемой (рис. 5.1).

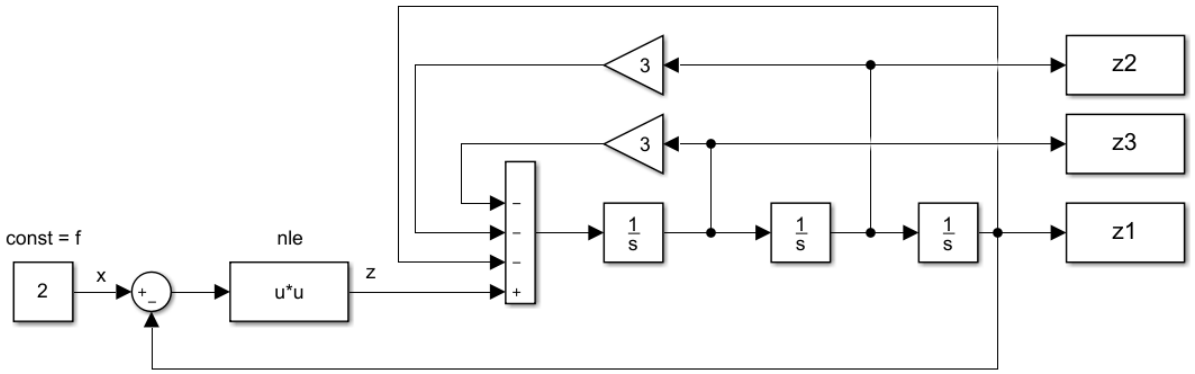


Рис. 5.1. Структура модели

Указание: для анализа устойчивости «в малом» предварительно составьте систему дифференциальных уравнений.

Задание:

- найдите положение равновесия;
- линеаризуйте модель в предположении о малости отклонений от положения равновесия;
- вычислите с.з. матрицы (корни характеристического полинома) линеаризованной системы;
- заключение об устойчивости положения равновесия нелинейной системы.

Проверьте результат исследования по фазовому портрету системы.

Лабораторная работа 6. Метод гармонического баланса

Автоколебания — периодические процессы в автономных системах — практически важные режимы функционирования многих систем автоматического управления и регулирования. Если положение равновесия системы неустойчиво, то колебания реальной системы в силу естественных ограничений расходятся до определенной амплитуды.

Автоколебательные системы успешно применяются для поддержания различных физических переменных технологических процессов, если амплитуда и частота колебаний находятся в допустимых пределах. Релейная система стабилизация температуры, в которой наблюдаются автоколебания (см. рис. 2.6) рассматривается в Лабораторной работе 2.

Точное определение формы и параметров периодических режимов возможно только для некоторых типов нелинейных систем, в частности релейных. Для исследования предельных циклов в системах второго порядка весьма удобен метод фазовой плоскости. На фазовой плоскости автоколебаниям отвечают устойчивые предельные циклы — изолированные замкнутые фазовые траектории (см. Лабораторную работу 3).

Во многих случаях системы управления описываются моделями высоких порядков и имеют сложные характеристики нелинейных элементов. В этих случаях применяют приближенные методы исследования периодических режимов. Важную информацию о существовании периодических режимов в нелинейных системах, их числе и параметрах может дать приближенный *метод гармонического баланса*. По результатам, полученным этим методом, могут быть оценены начальные условия для моделирования систем на ЭВМ с целью последующего уточнения форм и параметров локализованных периодических режимов.

Расчетная структура нелинейных моделей изображена на рис. 6.1.

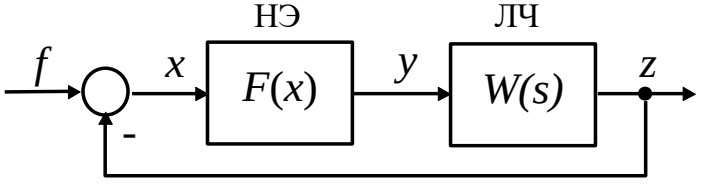


Рис. 6.1. Расчетная структура нелинейных моделей

Если СХ НЭ представляет нечетную функцию, то при отсутствии воз-

действия ($f \equiv 0$) в системе могут возникать симметричные автоколебания.

В методе гармонического баланса принимается гипотеза о форме искомого периодического процесса $x(t) = A\sin\omega t$ на входе НЭ (см. рис. 6.1). Параметризация решения принципиально упрощает задачу — ищется амплитуда A и частота ω решения.

Анализ включает этапы:

- гармоническая линеаризация НЭ;
- локализация параметров периодических решений по гармонически линеаризованной системе;
- анализ устойчивости предельного цикла.

Пример анализа системы (см. рис. 6.1), где НЭ представляет идеальное реле, а ЛЧ системы имеет ПФ

$$W(s) = \frac{1}{s(s+1)^2}. \qquad (6.1)$$

1. Коэффициент гармонической линеаризации идеального реле выражается формулой

$$q(A) = 4C / \pi A .$$

2. Характеристический полином (ХП) гармонически линеаризованной (эквивалентной) системы запишется так:

$$D_э(s,A) = s(s+1)^2 + 4C / \pi A = s^3 + 2s^2 + s + 4C / \pi A . \qquad (6.2)$$

Эквивалентная система должна находиться на границе устойчивости, т. е. ее ХП должен иметь пару мнимых корней (остальные в левой полуплоскости). Для полинома третьей степени с положительными коэффициентами (6.2) условие колебательной границы устойчивости сводится к равенству произведений «средних» и «крайних» коэффициентов (следует из критерия Рауса—Гурвица):

$$2 \cdot 1 = 1 \cdot 4C / \pi A .$$

Откуда следует искомая амплитуда периодического решения $A^* = 2C / \pi$. Подстановка A^* в выражение ХП $D_э(s,A)$ дает

$$D_э(j\omega, A^*) = -j\omega^3 - 2\omega^2 + j\omega + 2 = 0$$

или

$$\mathrm{Re}D_з(j\omega, A^*) = -2\omega^2 + 2 = 0;$$
$$I_m D_з(j\omega, A^*) = \omega - \omega^3 = 0.$$

Решение первого уравнения дает частоту $\omega^* = 1$ периодического режима.

3. Если периодическое решение дифференциального уравнения гармонически линеаризованной системы устойчиво, то в нелинейной системе имеют место автоколебания. Поскольку метод гармонического баланса дает приближенные решения, то и метод исследования устойчивости является приближенным.

Идея метода содержится в следующих рассуждениях. Характеристический полином $D_з(s, A^*)$ гармонически линеаризованной системы, находящейся на колебательной границе устойчивости, имеет пару мнимых корней $\pm j\omega^*$, а остальные — левые (рис. 6.2). При этом колебания переменной *x* на входе НЭ по форме близки к гармонической, их амплитуда близка к A^*, а частота — ω^*.

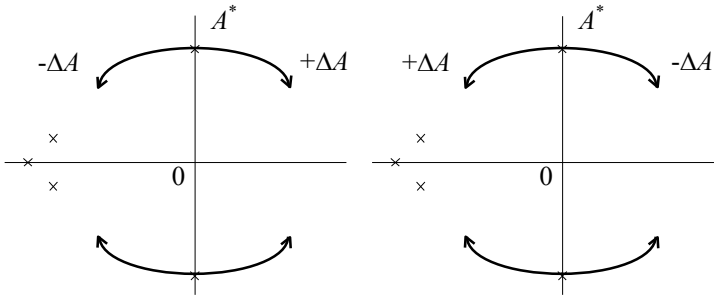


Рис. 6.2. Иллюстрация анализа устойчивости периодического решения

Если возмущения привели к изменению амплитуды $A^* \pm \Delta A$, то характеристические полиномы

$$D_з(s, A^* \pm \Delta A)$$

уже не имеют чисто мнимых корней. Здесь выделим два случая.

Пусть в первом случае полином

$$D_з(s, A^* + \Delta A)$$

имеет все корни в левой полуплоскости, а полином

$$D_з(s, A^* - \Delta A)$$

имеет пару правых корней (рис. 6.2, *а*). Тогда исследуемое периодическое решение ω^*, A^* устойчиво. Действительно, при увеличении амплитуды гармонически линеаризованная система становится устойчивой — колебания затухают до прежнего значения $A = A^*$. При уменьшении этой амплитуды гармонически линеаризованная система становится неустойчивой — колебания расходятся до значения $A = A^*$.

Во втором случае полином

$$D_з(s, A^* + \Delta A)$$

имеет пару правых корней, а у полинома

$$D_з(s, A^* - \Delta A)$$

все корни находятся в левой полуплоскости (рис. 6.2, *б*). Аналогичными рассуждениями приходим к заключению, что в этом случае периодическое решение неустойчиво.

Таким образом, для определения устойчивости периодического решения к полиному $D_з(s, A^*)$ дважды применяют критерий Гурвица (Рауса): для $A = A^* + \Delta A$ и $A = A^* - \Delta A$, что составляет заключительную часть методики Е.П. Попова.

Задание 1. Исследуйте автоколебания системы регулирования температуры, в которой роль регулятора играет реле с гистерезисом.

1. Модуль эквивалентного комплексного коэффициента передачи реле с гистерезисом

$$R_н(A) = \frac{4C}{\pi A};\ A \geq b$$

совпадает с модулем комплексного коэффициента передачи для идеального реле, если амплитуда не меньше ширины гистерезиса. Реле с гистерезисом вносит отрицательный фазовый сдвиг по первой гармонике

$$\varphi_н(A) = -\mathrm{Arctg}\frac{b}{\sqrt{A^2 - b^2}}$$

от $-\pi/2$ при $A = b$ до нуля, при $A/b \to \infty$.

2. В этом примере рекомендуется использовать методику Л.С. Голь-

дфарба [1], [2]. Графическая методика поиска параметров A^*, ω^* основана на условии

$$W(j\omega) = -\frac{1}{W_{\text{н}}(A)} = I(A),$$

где: $W(j\omega)$ — амплитудно-фазовая характеристика ЛЧ, $I(A)$ — инверсная характеристика НЭ. Точка их пересечения отвечает границе устойчивости эквивалентной системы. Частота колебаний локализуется по меткам на АФХ $W(j\omega)$, а амплитуда — по меткам на инверсной характеристике $I(A)$.

Инверсная характеристика реле с гистерезисом выражается так:

$$I_2(A) = -\frac{1}{q(A) + jq'(A)} = -\frac{\pi\sqrt{A^2 - b^2}}{4C} - j\frac{\pi b}{4C}.$$

Поскольку мнимая часть постоянна и не зависит от A, график этой характеристики есть прямая в третьем квадранте, параллельная оси абсцисс.

На рис. 6.3 показаны годографы $W(j\omega)$ и $I_1(A)$ для системы (рис. 6.1), ЛЧ которой имеет ПФ (6.1), а НЭ является идеальным реле. Инверсная характеристика идеального реле

$$I(A) = -\frac{1}{q(A)} = -\frac{\pi A}{4C}$$

совпадает с отрицательной действительной полуосью. Пересечению кривых отвечают параметры:

$$\omega_1^* = 1, \quad A_1^* = 2C / \pi.$$

Построим частотную характеристику средствами Python библиотеки control или средствами MatLab:

```
A = 1
C = 0.5
w_l = tf([1], [1, 2, 1, 0])
w_n = tf([-np.pi*A], [4*C])
plt.axhline(-np.pi*A/(4*C), color='r')
nyquist(w_l)
plt.ylim(-2, 0.5)
plt.legend(['I(A)','W(j omega)'])
plt.plot()
plt.show()
```

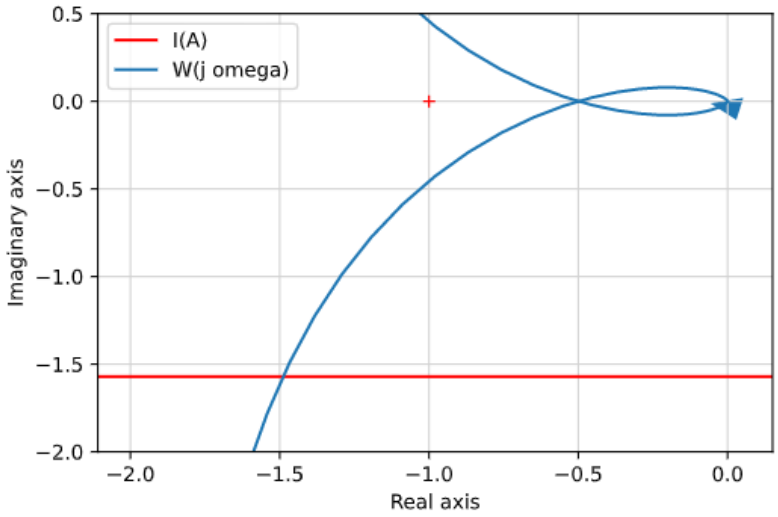


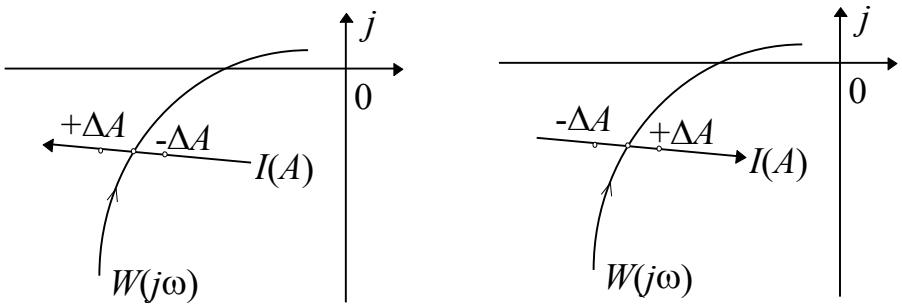
Рис. 6.3. Иллюстрация методики Гольдфарба

3. Анализ устойчивости периодического решения. Методика Л.С. Гольдфарба оперирует годографами на комплексной плоскости. Точки инверсной характеристики НЭ $I(A)$ при конкретных значениях амплитуды A играют роль критических точек в формулировке критерия Найквиста. При $A = A^*$ АФХ ЛЧ $W(j\omega)$ проходит через критическую точку — гармонически линеаризованная система находится на колебательной границе устойчивости.

Если возмущения привели к изменению амплитуды $A^* \pm \Delta A)$, то могут быть две ситуации (рис. 6.4).

В первой ситуации (рис. 6.4, *а*) увеличение амплитуды приводит к тому, что АФХ ЛЧ не охватывает критическую точку $I(A^* + \Delta A)$ — система устойчива, колебания затухают до прежнего значения. При уменьшении амплитуды АФХ ЛЧ охватывает критическую точку $I(A^* - \Delta A)$, а значит, система не устойчива, колебания расходятся. Вывод — периодическое решение устойчиво.

Во второй ситуации (рис. 6.4, *б*) увеличение амплитуды приводит к охвату АФХ ЛЧ критической точки $I(A^* + \Delta A)$ — система становится неустойчивой, колебания расходятся, т. е. амплитуда еще более увеличивается. Уменьшение амплитуды, наоборот, приводит к охвату критической точки, система становится устойчивой — колебания затухают, т. е. их амплитуда становится еще меньше. Вывод — периодическое решение неустойчиво.



а *б*

Рис. 6.4. Анализ устойчивости периодического решения по методике Гольдфарба

Задание 2. Полученные методом гармонического баланса результаты анализа автоколебаний сравнить с результатами компьютерной имитации.

Задание 3. Анализ спектра сигналов в нелинейной системе.

Проанализируем условия прохождения гармонического сигнала по контуру, образованному НЭ, ЛЧ и элементом сравнения (рис. 6.1). Допустим, что несмещенный сигнал на выходе НЭ

$$y(t) = F(x(t)) = F(A\sin\omega t)$$

имеет ту же частоту (тот же период $T = 2\pi/\omega$), что и вход. В силу нелинейного характера преобразования $y = F(x)$ сигнал $y(t)$, кроме первой гармоники, содержит и высшие гармоники. НЭ выступает как генератор высших гармоник.

Пусть линейная часть системы с передаточной функцией $W(s)$ является фильтром низких частот (гипотеза фильтра). Проходя через ЛЧ, высшие гармоники ослабляются в большей степени, чем первая гармоническая составляющая. Поэтому в составе переменной на выходе ЛЧ доля высших гармоник мала:

$$z(t) \approx A_z \sin(\omega t + \varphi_z).$$

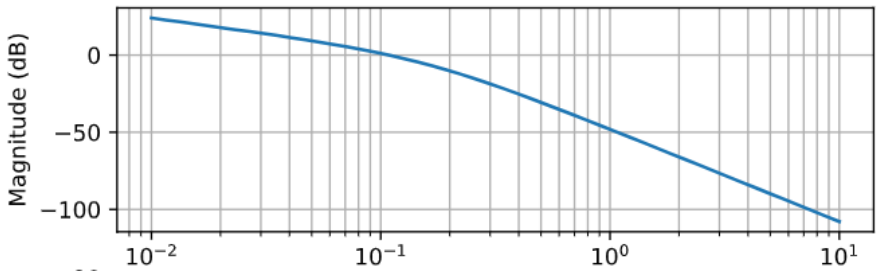


Рис. 6.5. АЧХ (ЛАЧХ) ЛЧХ bodemag

Аналогично первой лабораторной работе проведите анализ спектров колебаний на входе и выходе ЛЧ с ПФ (6.1), т. е. на выходе и входе идеального реле. Сигнал на выходе реле и входе ЛЧ — меандр; известно, что он содержит только нечетные гармоники с амплитудами: $A_3 = A_1/3$, $A_5 = A_1/5$ и т. д.

На рис. 6.5 приведена амплитудно-частотная характеристика ЛЧ. Можно видеть, что теоретические значения при $\omega^* = 1$ $с^{-1}$ равны:

$W(j\omega^*) = -50$Дб; $W(j3\omega^*) = -75$Дб $W(j5\omega^*) = -90$Дб

Аналогично первой лабораторной работе, постройте спектральные характеристики всей системы и оцените влияние нелинейной части на спектральные характеристики системы. Обратите внимание (проведите эксперименты), что спектральных характеристика нелинейного элемента может зависеть от амплитуды входного сигнала.

*Постройте трехмерный спектр рассматриваемой системы (зависимость усиления от амплитуды и частоты входного сигнала)

Список рекомендованной литературы

1. Теория автоматического управления: Учебник для вузов/ С.Е. Душин, Н.С. Зотов, Д.Х. Имаев и др. Под ред. В. Б. Яковлева. Изд. 3-е, стер. — М.: Высш. шк., 2009. С. 567.
2. Теория управления: Учебник/ А.А. Алексеев, Д.Х. Имаев, Н.Н. Кузьмин, В.Б. Яковлев. — СПб: Изд-во СПб ГЭТУ «ЛЭТИ», 1999. С.435.
3. Инструкция по использованию Python/Numpy https://digiratory.ru/1452
4. Имаев Д.Х. Дискретные системы управления. Учебное пособие — СПб: Изд-во СПб ГЭТУ «ЛЭТИ», 2005. С. 148.
5. Имаев Д.Х. Системы вывода в четкой логике и задачи теории управления. В кн. Управление и информационные технологии. — СПб: Изд-во СПб ГЭТУ «ЛЭТИ», 2011. С. 114.
6. Кораблев Ю.А. Шестопалов М.Ю. Системы управления с нечеткой логикой: Учеб. пособие — СПб : СПб ГЭТУ «ЛЭТИ», 1999. С. 60.
7. Nerode A., Kohn W. Models for hybrid systems: Automata, topologies, stability// Hybrid Systems/ Editors R. L. Grossman et al. Lecture Notes in Computer Science. Vol. 736. — Berlin: Springer-Verlag, 1993. P. 317—356.
8. Терехов В. А., Ефимов Д. В., Тюкин И. Ю. Нейросетевые системы управления: Учеб. пособие для вузов. — М.: Высш. шк. 2002. С. 183.
9. Каталог газодинамических характеристик центробежных нагнетателей. https://www.turbinist.ru/11791-katalog-gazodinamicheskih-harakteristik-cbn.html

Учебно-методическое пособие

по выполнению лабораторных работ по дисциплине «Теория автоматического управления». Часть 2. Нелинейные системы

Редактор Г. Г. Петров

Подписано в печать 30.10.20. Формат 60×84 1/16.

Бумага офсетная. Печать цифровая. Печ. л. 3,0.

Гарнитура «Times New Roman». Тираж 72 экз. Заказ 000.

Издательство СПб ГЭТУ «ЛЭТИ»

197376, С.-Петербург, ул. Проф. Попова, 5