TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Rakenduse arendamine automaatjuhtimissüsteemide PID-regulaatori seadistuste arvutamiseks MATLABi abil

RDDR08/17, TOOTMISE AUTOMATISEERIMINE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Natalja Ivleva
Üliõpilaskood: 178691RDDR
Juhendaja: Sergey Chekryzhov, lektor

Kohtla-Järve, 2022

TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Разработка приложения расчёта настроек ПИД регулятора для систем автоматического регулирования с помощью MATLAB

RDDR08/17, TOOTMISE AUTOMATISEERIMINE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Natalja Ivleva
Üliõpilaskood: 178691RDDR
Juhendaja: Sergey Chekryzhov, lektor

Kohtla-Järve, 2022

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

"...." ..... 20.....

Autor: .....
/ allkiri /

Töö vastab rakenduskõrgharidusõppe lõputööle/magistritööle esitatud nõuetele
"...." ..... 20.....

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud
"...." ..... 20.....

Kaitsmiskomisjoni esimees .....
/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS JA REPRODUTSEERIMISEKS

Mina Natalja Ivleva (sünnikuupäev: 14.05.1973)

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose Rakenduse arendamine automaatjuhtimissüsteemide PID-regulaatori seadistuste arvutamiseks MATLABi abil, mille juhendaja on Sergey Chekryzhov.

1.1. reprodutseerimiseks säilitamise ja elektroonilise avaldamise eesmärgil, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ja teistest õigusaktidest tulenevaid õigusi.

TalTech Inseneriteaduskond Virumaa kolledž
LÕPUTÖÖ ÜLESANNE

Üliõpilane: Natalja Ivleva, 178691RDDR

Õppekava, peaeriala: RDDR08/17, Tootmise automatiseerimine

Juhendaja: lektor, Sergey Chekryzhov, sergey.chekryzhov@taltech.ee

Lõputöö teema:

(eesti keeles) Rakenduse arendamine automaatjuhtimissüsteemide PID-regulaatori seadistuste arvutamiseks MATLABi abil

(inglise keeles) Application development to calculate PID regulator settings for automatic control systems using MATLAB

Lõputöö põhieesmärk:

Luua erinevate insenerimetoodikate abil PID-regulaatori arvutamise rakendus MATLABi kasutamisel.

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Teha PID-regulaatoriga automaatjuhtimissüsteemide häälestuse arvutamise insenerimetoodikate võrdlev analüüs ja soovitada praktiliseks kasutamiseks kõige sobivamaid (vastavaid) algoritme.	15.10.2021
2.	Arvutada PID-regulaatori häälestused MATLABis, võttes arvesse soojusenergeetikas kasutatavate siirdeprotsesside kvaliteedi kriteeriume.	30.10.2021
3.	Saada PID-kontrolleri häälestusparameetrite sõltuvused ja välja töötada nende korrigeerimise algoritm.	15.11.2021
4.	Kirjalik osa.	18.12.2021

Töö keel: vene **Lõputöö esitamise tähtaeg:** "04" jaanuar 2022a

/allkiri/

Konsultant: ..... ".....".....20.....a

/allkiri/

Programmijuht: ..... ".....".....20.....a

/allkiri/

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Тема выпускной работы была предложена лектором Вирумааского колледжа Таллиннского Технического Университета Сергеем Чекрыжовым.

Все материалы и данные по изучению выбранной темы были найдены автором самостоятельно. Также использовался учебный материал по предмету RAA0590 Automaatjuhtimissüsteemid, созданный преподавателем Сергеем Чекрыжовым в учебной среде Moodle.

В данной работе проводится сравнительный анализ инженерных методик расчёта настройки систем автоматического управления с ПИД–регулятором и выполняется расчет настроек ПИД –регулятора с помощью MATLAB.

Ключевые слова: ПИД регулятор, метод AMIGO, метод *Skogestad*, метод Циглера-Николса (Ziegler–Nichols), MATLAB, Simulink.

ВВЕДЕНИЕ

В области автоматизации технологических объектов, процессов в настоящее время очень широко используются ПИД-регуляторы (П–пропорционально И–интегрально Д - дифференцирующие регуляторы - *P-proportional I-integrative D-derivative regulator*). Высокая популярность использования объясняется тем, что регулятор позволяет достичь поставленной цели управления для большинства технологических объектов.

Ключевой проблемой в использовании ПИД-регулятора является расчет и настройка его коэффициентов. Существует множество методов расчета и программ, которые различаются по разной степени точности и трудоемкости.

Процесс настройки ПИД – регулятора по экспериментальным правилам, которые использует автор в своей работе, интуитивен и для получения результатов, необходимо настроить регулятор с начальными приближенными коэффициентами, так как без данного расчета коэффициентов попытки настроить ПИД-регулятор могут оказаться безрезультатными.

Для студентов, которые обучаются на специальностях, связанных с программированием контроллеров, проектированием автоматизированных систем, инженеров, работающих в области промышленной автоматизации нужно уметь не только пользоваться готовыми решениями, но и уметь самостоятельно рассчитать, визуализировать, проанализировать полученные данные.

Целью данной выпускной работы является: создать приложения расчёта коэффициентов ПИ регулятора для систем автоматического регулирования с помощью MATLAB на примере практического задания по предмету RAA0590 Automaatjuhtimissüsteemid.

MATLAB – это среда и язык технических расчетов, предназначенный для решения широкого спектра инженерных и научных задач любой сложности в любых отраслях [1]. В MATLAB имеется набор функций, которые позволяют рассчитать различные виды ПИД регулирования и визуализировать результаты вычислений. Также можно использовать среду Simulink, например, PID Tuner для построения моделей системы управления объектов с помощью ПИД регулятора.

Задачи, которые необходимо выполнить для создания приложения:

1. Провести сравнительный анализ инженерных методик расчёта настройки систем автоматического управления с ПИД –регулятором и рекомендовать наиболее пригодные (соответствующие) алгоритмы для практического использования.

2. Выполнить расчет настроек ПИД –регулятора в MATLAB с помощью разных методов.

Выпускная работа состоит из следующих глав:

- В введении описана актуальность данной темы, цели и задачи выпускной работы.
- В главе 1 сделан обзор ПИД регулирования и методы, которые используются для расчета коэффициентов настройки регулятора.
- В глава 2 приведено описание создания скрипта в MATLABe, который вычисляет коэффициенты ПИ регулятора, создает передаточные функции и строит графики переходных процессов замкнутой системы регулирования.
- В главе 3 представлено два примера построения модели с помощью Simulink.
- В главе 4 описано дальнейшее развитие данного направления при создании приложений и применения решения на примере учебного стенда.

В работе использованы данные на примере практического задания расчета коэффициентов ПИ-регулятора по предмету RAA0590 Automaatjuhtimissüsteemid (https://moodle.taltech.ee/course/view.php?id=30553).

1. МЕТОДИКИ РАСЧЕТА ПАРАМЕТРОВ ПИД РЕГУЛЯТОРОВ

1.1 Типы методов настройки регуляторов

На производстве существуют системы, параметры которых должны оставаться постоянными под воздействием внешних факторов. Например, поддержание определенной температуры в резервуаре, давление в газовых трубах и т.п. Для этого применяют системы с обратной связью с использованием регулятора. [2]

Классические ПИ и ПИД-регуляторы составляют основную долю промышленных регуляторов. По сравнению с другими типами регуляторов, они обладают более широкими возможностями для настройки систем с большим транспортным запаздыванием, что характерно для производств химической промышленности в целом. На основании этого одним из актуальных направлений теории автоматического управления является параметрический синтез типовых ПИ-, ПИД-регуляторов. [2]

С точки зрения теории автоматического управления все объекты управления, их можно разделить всего на два крупных класса: объекты, проявляющие линейные свойства, и объекты, обладающие нелинейной характеристикой по каналу управления. Математическое описание таких объектов в большинстве случаев можно представить в виде передаточной функции с запаздыванием. Первые – стационарной передаточной функцией, вторые – передаточной функцией с интервально-заданными параметрами. Запаздывание может быть, как непосредственно в самом объекте управления, так и проявляться за счет аппроксимации переходного процесса динамическим звеном более низкого порядка. [2]

На практике наибольшее применение находят одноконтурные замкнутые системы автоматического регулирования (САР) с типовыми линейными законами регулирования. В цепи такой системы находятся последовательно соединенные регулятор и объект регулирования с указанными передаточными функциями. При синтезе системы управления наиболее сложным и ответственным этапом является выбор параметров настройки регулятора.

Существует большое количество методов настройки параметров регулятора:

- аналитические,
- графические,
- оптимизационные,
- на основе определенных правил. [2]

1.1.1 Аналитические методы

Аналитические методы основаны на обеспечении определенных критериев в настраиваемой системе, таких как: заданный показатель управляемости, заданные запасы устойчивости по фазе и амплитуде, обеспечение максимальной степени устойчивости САУ (Система автоматического управления). [7]

Одним из самых распространенных аналитических методов параметрического синтеза ПИД-регуляторов является метод оптимального модуля. Данный метод обеспечивает высокое быстродействие, минимальное время переходного процесса и малое перерегулирование при довольно больших запасах устойчивости по амплитуде и по фазе. [7]

Наиболее полно данный метод раскрыт в работах: [3-6].

Аналитические методы не требуют понижения порядка передаточной функции обобщенного объекта управления, позволяют охватить широкий класс систем. Недостатком аналитических методов является невозможность задания желаемых показателей устойчивости и качества переходного процесса.

1.1.2 Графические методы

Суть графических методов заключается в построении области устойчивости в координатах параметров регуляторов. Затем производят построение в этих координатах новой области параметров, которые рассчитаны по заранее заданному критерию и далее, основываясь на дополнительных правилах, выбирают оптимальные параметры настройки регуляторов. Таким образом, находят сначала глобальную область устойчивости системы регулятора, а затем объединяют ее с областью, гарантирующей заданные запасы устойчивости по амплитуде и по фазе. [7]

Существенным недостатком графических методов является трудоемкий процесс построения сложных поверхностей областей устойчивости, а также, приближённое нахождение параметров настройки регулятора. [7]

1.1.3 Оптимизационные методы

Оптимизационные методы позволяют осуществить параметрический синтез, гарантирующий оптимальность параметров в смысле заданного функционала, но практическое применение затруднено большими временными затратами на расчет параметров и необходимостью использовать численные методы для поиска экстремума функции. [7]

В литературе есть примеры подходов к настройке регуляторов лишь на основе интегральных показателей [8,9]. Но данные методы сильно зависят от начальных

условий, которые выбираются путем расчета параметров настройки каким-либо другим методом, и, по сути, являются лишь уточняющими средствами. [7]

Гораздо более широкое распространение получили методы, основанные на задании показателей устойчивости. [7]

1.1.4 Методы, основанные на правилах

Существует большое количество методов настройки регуляторов на основе определенных правил. Это обстоятельство объясняется простотой настройки, в связи с чем, эти методы являются самыми используемыми при настройке регуляторов в промышленности. [7]

Согласно последним работам, посвященным настройке ПИД-регуляторов, наибольшее распространение в теоретических и практических работах приобрели методы *AMIGO, SIMC (Skogestad internal model control)*, метод оптимального модуля, метод Циглера-Николса (Ziegler–Nichols). Именно с этими методами целесообразно сравнивать новую методику настройки ПИД-регулятора. [18]

Недостатком всех методов, основанных на правилах, служит необходимость предварительного уменьшения порядка модели, что ведет к ошибке, которая влияет на качество настройки системы управления. [18]

В данной работе автор рассматривает три метода, основанные на правилах, настройки ПИД регулятора.

1.2 Принцип работы ПИД регулирования

В системе управления объектом можно выделить несколько составляющих:

- Объект управления – то чем нужно управлять;
- Управляющее устройство – оно изменяет состояние объекта в зависимости от подданного сигнала;
- Уставка – требуемое значение параметра, задается программой или человеком;
- Регулятор – по определенному алгоритму вычисляет значение сигнала для управляющего устройства;
- Датчик – измеряет реальное значение параметра у объекта;
- Обратная связь – возвращает величину ошибки между реальным значением параметра и требуемым. [10]

Рисунок 1.1. Схема обратной связи [10]

Наибольшее применение в замкнутых системах автоматического управления (САУ) получили регуляторы, в которых используется принцип управления по пропорциональной, интегральной и дифференциальной составляющим сигнала ошибки (ПИД-регуляторы). Популярность ПИД регуляторов объясняется простотой построения, ясностью принципа функционирования и возможностью обеспечить высокие качественные характеристики САУ различными промышленными объектами. [11]

$$u(t) = P + I + D = K_p\, e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de}{dt} \qquad (1.1)$$

K_p, K_i, K_d — коэффициенты усиления пропорциональной, интегральной и дифференциальной составляющих регулятора. Функция зависит от времени, т.к. нужно периодически проверять состояние объекта и оперативно его корректировать.

Передаточная функция PID-регулятора

$$W_{PID}(s) = K_P + \frac{K_i}{s} + K_d \cdot s \qquad (1.2)$$

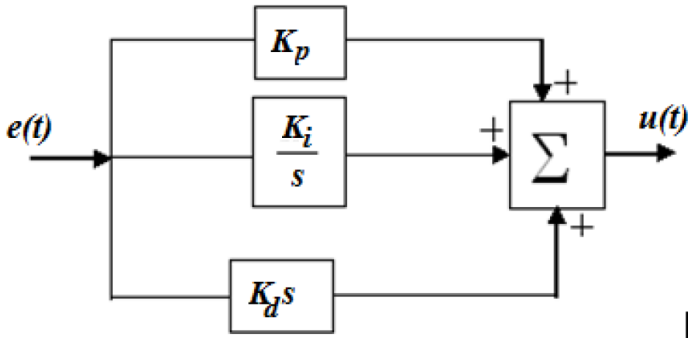


Рисунок 1.2 Схема ПИД регулятора

Входы и выходы регулятора позволяют поддерживать значение измеряемого параметра (PV - *processvalue*) с уставкой (SP - *setpoint*) изменяя величину управляющего воздействия (OP - *output*)

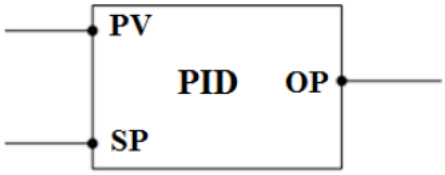


Рисунок 1.3 Схема входных/выходных данных для ПИД регулятора

$$OP = K_P \cdot (SP - PV) + K_d \frac{d(SP - PV)}{dt} + K_i \int (SP - PV)dt \quad (1.3)$$

K_p, K_d, K_i – коэффициенты усиления PID регулятора.

Для расчета параметров PID регулятора, необходимо определить цель, критерии качества регулирования, а также ограничения на значения и скорости изменения переменных.

Оптимизация параметров САР представляет собой трехшаговую процедуру:

- на первом шаге определяется математическая модель объекта регулирования и выбирается соответствующий тип регулятора (П, ПИ, ПИД);
- на втором шаге выполняется оптимизация параметров замкнутого либо разомкнутого контура регулирования, включающего автоматический регулятор и объект (процесс) регулирования;
- на третьем шаге рассчитываются оптимальные параметры регулятора с использованием величин коэффициентов оптимизированного контура автоматического регулирования. [12]

Для ПИД регулирования в работе рассматриваются следующие методы, вычисления которых произведены с помощью среды MATLAB:

- метод Циглера-Николса
- метод AMIGO
- метод Skogestad
- методы и технологии MATLAB

Все три метода (Циглера-Николса, AMIGO, Skogestad) являются экспериментальными.

1.3 Метод Циглера-Николса

Метод Циглера-Николса (Ziegler-Nichols) определения настроек регуляторов используется для быстрой и приближенной оценки значений настроек регуляторов. [12]

Джон Г. Циглер и Натаниэль Б. Николс – это американские ученые-инженеры, которые представили в своей исследовательской работе расчет коэффициентов ПИД-регулятора в 1942 году.

Для его применения используется реакция объекта управления на ступенчатое воздействие. Объекты управления, имеющие апериодическую кривую разгона, аппроксимируются последовательным соединением апериодического и запаздывающего звеньев. [12]

Если объект управления представляет собой инерционное звено с запаздыванием, то описывается передаточной функцией

$$W(s)=\frac{K}{Ts+1}e^{-\tau s}, \quad (1.4)$$

где K – коэффициент усиления, T - постоянная времени, т - запаздывание, то настройки P-, I-, PI- и PID-регуляторов могут быть определены по приведенным в таблице 1.1 формулам. [12]

Таблица 1.1 Расчет коэффициентов усиления по методу Циглера-Николса

Регулятор	Апериодический процесс	Процесс с перерегулированием 20 %	Процесс с минимальным временем регулирования
P	$K_1=\frac{0{,}3\cdot T}{K\cdot\tau}$	$K_1=\frac{0{,}7\cdot T}{K\cdot\tau}$	$K_1=\frac{0{,}9\cdot T}{K\cdot\tau}$

I	$K_0 = \frac{1}{4{,}5 \cdot K \cdot \tau}$	$K_0 = \frac{1}{1{,}7 \cdot K \cdot \tau}$	$K_0 = \frac{1}{1{,}7 \cdot K \cdot \tau}$
PI	$K_1 = \frac{0{,}6 \cdot T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$	$K_1 = \frac{0{,}7 \cdot T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$	$K_1 = \frac{T}{K \cdot \tau}$, $K_0 = \frac{1}{K \cdot \tau}$
PID	$K_1 = \frac{0{,}95 \cdot T}{K \cdot \tau}$, $K_0 = \frac{0{,}4 \cdot T}{K \cdot \tau^2}$, $K_2 = \frac{0{,}38 \cdot T}{K}$	$K_1 = \frac{1{,}2 \cdot T}{K \cdot \tau}$, $K_0 = \frac{0{,}6 \cdot T}{K \cdot \tau^2}$, $K_2 = \frac{0{,}48 \cdot T}{K}$	$K_1 = \frac{1{,}4 \cdot T}{K \cdot \tau}$, $K_0 = \frac{1{,}08 \cdot T}{K \cdot \tau^2}$, $K_2 = \frac{0{,}7 \cdot T}{K}$

Метод Циглера-Николса дает параметры, далекие от оптимальных. Это объясняется упрощенностью самого метода. Судя по медленному затуханию переходного процесса в системе, этот метод дает слишком малый запас устойчивости. [12]

1.4 Метод AMIGO

В последнее время популярностью пользуется метод AMIGO (метод Астера-Хугланда *Karl Johan Åström, Tore Hägglund* – ученые-исследователи из Швеции).

Метод основан на задании объекта управления, в виде инерционное звено первого порядка или интегрирующее звено с запаздыванием.

Правила настройки PI и PID-регуляторов с использованием данного метода представлены в таблице 1.2 [14].

Таблица 1.2 Расчет коэффициентов усиления по методу AMIGO

Передаточная функция объекта управления	K_P	K_i	K_d
$W_o = \frac{K}{Ts+1} \cdot \exp(-\tau \cdot s)$,	$\frac{1}{K} \cdot \left(0{,}2 + 0{,}45 \frac{T}{\tau}\right)$	$\frac{\tau + 0{,}1T}{0{,}4 \cdot \tau^2 + 0{,}8 \cdot T \cdot \tau} K_P$	$\frac{0{,}5 \cdot \tau \cdot T}{0{,}3 \cdot \tau + T} K_P$
$W_o = \frac{K}{s} \cdot \exp(-\tau \cdot s)$,	$\frac{0{,}45}{K}$	$\frac{K_p}{8 \cdot \tau}$	$0{,}5 \cdot K_P \cdot \tau$

1.5 Метод Skogestad

Метод основан на задании желаемого вида переходной характеристики, замкнутой САУ (желаемый вид описывается инерционным звеном первого порядка) путем выбора постоянной времени (Тс). [18]

Таблица 1.3 Параметры настройки PID –регулятора по методу *Skogestad* [12]

Передаточная функция объекта управления	$K_P^{'}$	$K_i^{'}$	$K_d^{'}$,
$\frac{K}{T_1 s+1}\cdot\exp(-\tau\cdot s),$	$\frac{T_1}{K(T_c+\tau)}$	$\min[T_1,4(T_c+\tau)]$	-
$\frac{K}{s}\cdot\exp(-\tau\cdot s),$	$\frac{1}{K(T_c+\tau)}$	$4(T_c+\tau)$	-
$\frac{K}{(T_1 s+1)(T_2 s+1)}\cdot\exp(-\tau\cdot s),$	$\frac{T_1}{K(T_c+\tau)}$	$\min[T_1,4(T_c+\tau)]$	T_2
$\frac{K}{s(T_2 s+1)}\cdot\exp(-\tau\cdot s),$	$\frac{1}{K(T_c+\tau)}$	$4(T_c+\tau)$	T_2

T_c- ожидаемая постоянная времени (desired response time)

Для перехода к стандартному виду PID–регулятора используются формулы:

$$K_P=\frac{K_P^{'}(K_i^{'}+K_d^{'})}{K_i^{'}},\ K_i=\frac{K_P^{'}}{K_i^{'}},\ K_d=K_P^{'}\cdot K_d^{'}, \quad (1.5)$$

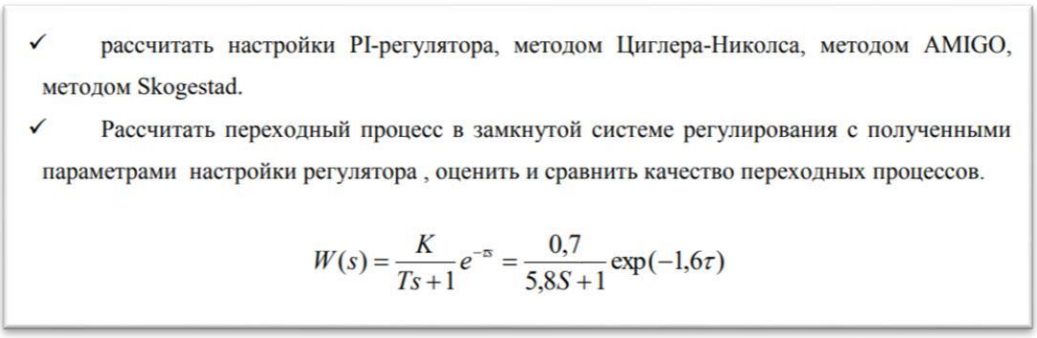
В заключении можно отметить, что недостатком всех экспериментальных методик расчета является неполнота информации о запасе устойчивости и робастности системы. Для учета этих факторов необходим предварительный анализ динамики объекта регулирования и параметров передаточных функций. В этом случае можно смоделировать объект регулирования с учётом диапазона изменения характеристик и далее выйти на параметры настройки, близкие к оптимальным, которые не нуждаются в дополнительной настройке.

2. ПРИМЕНЕНИЕ MATLAB ДЛЯ РАСЧЕТА КОЭФФИЦИЕНТОВ ПИ РЕГУЛЯТОРА

Для изучения возможностей работы MATLAB с ПИД регулированием автор использовал практическое задание, которое представлено как пример в практическом задании по предмету RAA0590 Automaatjuhtimissüsteemid (S.Chekryzhov).

Данный пример был выбран, для того, чтобы можно было сравнить результаты вычислений из практического примера с решением MATLAB.

Пример практического задания, который рассматривается для решения разделен на части и представлен в работе в виде рисунков (рис. 2.1, 2.3, 2.5, 2.9 и 2.12)



- ✓ рассчитать настройки PI-регулятора, методом Циглера-Николса, методом AMIGO, методом Skogestad.
- ✓ Рассчитать переходный процесс в замкнутой системе регулирования с полученными параметрами настройки регулятора , оценить и сравнить качество переходных процессов.

$$W(s) = \frac{K}{Ts+1} e^{-\tau s} = \frac{0,7}{5,8S+1} \exp(-1,6\tau)$$

Рисунок 2.1. Пример задания [13]

В приложении определяются настройки ПИ регулятора методами Циглера-Николса, AMIGO, Skogestad для объекта с запаздыванием, а также те методы, что предлагает MATLAB.

Данная реализация носит ознакомительное представление о проекте, как прототипе приложения, которая демонстрирует функциональные возможности проекта с помощью MATLAB.

2.1 Создание проекта в MATLAB

Для создания проекта были рассмотрены различные возможности создания приложений в MATLAB:

- Script (сценарии) - это программные файлы, которые выполняют последовательность команд MATLAB.
- Function - Функции по сравнению со сценариями гораздо более гибкие и расширяемые. В отличие от сценариев, функции могут принимать входные и выходные данные для вызывающего. Live Script -

интерактивные документы (скрипты). Помимо команд могут содержать графику, форматированный текст.

- Simulink Model - Simulink позволяет инженерам моделировать системы из самых разных областей знаний (таких, как механические, электрические, гидравлические и другие системы) путем создания моделей в интерактивной графической среде. Симуляция, разработка, генерация кода, тестирование и верификация – все это осуществляется в единой среде. [15]
- App (Graphical User Interface) – Desktop приложение с графическим интерфейсом.

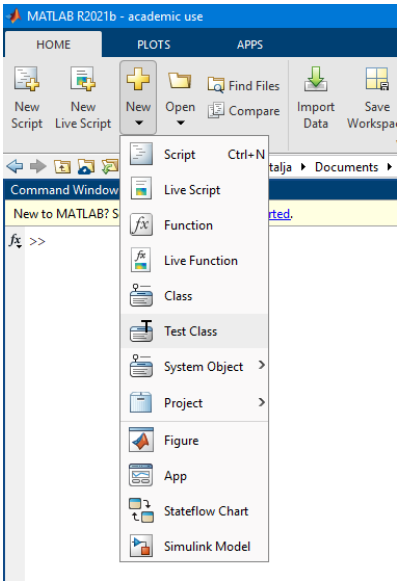


Рисунок 2.2. Меню. Создание приложений

В данной работе создаются приложения: Live Script и Simulink Model.

2.2 Приложение Live Script. Пример из учебного материала

2.2.1 Постановка задачи

Для знакомства с функциями MATLAB и решения задачи было создано Live Script приложение, где ввод данных можно осуществить с помощью диалогового окна ввода (рис 2.4).

✓ Рассчитать переходный процесс в замкнутой системе регулирования с полученными параметрами настройки регулятора , оценить и сравнить качество переходных процессов.

$$W(s) = \frac{K}{Ts+1} e^{-\tau s} = \frac{0,7}{5,8S+1} \exp(-1,6\tau)$$

Рисунок 2.3. Пример задание (продолжение) [13]

```
data = {'Enter K';'Enter time T';
'Enter tau (time delay)'}
answers = inputdlg(data)

K = sscanf(answers{1},'%f')     % 0.7
T = sscanf(answers{2},'%f')     % 5.8
tau = sscanf(answers{3},'%f')   % 1.6

%Transfer function Object
s=tf('s')
Ws = K/(T*s+1)*exp(-tau*s)
```

```
data = 3×1 cell array
    {'Enter K'              }
    {'Enter time T'         }
    {'Enter tau (time delay)'}
answers = 3×1 cell array
    {'0.7'}
    {'5.8'}
    {'1.6'}
K = 0.7000
T = 5.8000
tau = 1.6000
s =

  s

Continuous-time transfer function.
Ws =

                 0.7
  exp(-1.6*s) * ---------
                5.8 s + 1

Continuous-time transfer function.
```

Рисунок 2.4. Установка начальных значений и результат работы скрипта

Результат работы скрипта создал передаточную функцию объекта управления, аналогичную в примере.

s = tf('s') создает специальную переменную s, которую можно использовать для создания модели передаточной функции в непрерывном времени.

Встроенная функция tf создает передаточную функцию модели. В официальной документации (https://www.mathworks.com/help/index.html?s_tid=CRUX_lftnav) по MATLAB можно прочитать подробное описание функции, где представлены примеры с различными входными параметрами и подробным описанием.

2.2.2 Метод Циглера-Николса

1. Расчёт параметров настройки регулятора методом Ziegler-Nichols.

Для апериодического процесса параметры настройки определяются по следующим формулам:

$$K_1 = \frac{0{,}6 \cdot T}{K \cdot \tau} = \frac{0.6 \cdot 5{,}8}{0.7 \cdot 1{,}6} = 3{,}10 \qquad K_0 = \frac{1}{K \cdot \tau} = \frac{1}{0.7 \cdot 1.6} = 0.89$$

Таким образом передаточная функция PI-регулятора имеет вид:

$$W_c(s) = K_1 + \frac{K_0}{S} = \frac{K_1 \cdot S + K_0}{S} = \frac{3.10 \cdot S + 0.89}{S}$$

Рисунок 2.5. Пример задание (продолжение)[13]

Расчет параметров настройки и результат вычислений представлен на рис. 2.6

```
%PI Control
Kp_ZN = (0.6*T)/(K*tau) %K1
Ki_ZN = 1/(K*tau) %K0

c_ZN = pid(Kp_ZN,Ki_ZN)
tf(c_ZN)
```

```
Kp = 3.1071
Ki = 0.8929
C =

  Kp + Ki * 1/s

  with Kp = 3.11, Ki = 0.893

Continuous-time PI controller in parallel form.
ans =

  3.107 s + 0.8929
  ----------------
         s

Continuous-time transfer function.
```

Рисунок 2.6. Результат работы скрипта

Встроенная функция pid – создание ПИД регулятора. Количество передаваемых коэффициентов создает определенный ПИД регулятор.

`c = pid(Kp,Ki)` – создает пропорциональный, интегральный (ПИ) регулятор.

2.2.3 MATLAB и pidTuner

Встроенная функция *pidTuner* – открывает окно приложения алгоритма настройки ПИД-регулятора для линейной модели объекта. Алгоритм для настройки коэффициентов усиления ПИД - регулирования для достижения оптимального баланса между производительностью и надежностью. Можно интерактивно изменять время отклика, полосу пропускания, переходную характеристику или запас по фазе с помощью интерфейса PID Tuner, алгоритм вычисляет новые значения усиления ПИД регулятора.

Если применить данную функцию к созданному ПИ регулятору и передаточной функции объекта управления, то можно увидеть следующие результаты (см. рис. 2.7).

```
pidTuner(Ws,C_ZN)
```

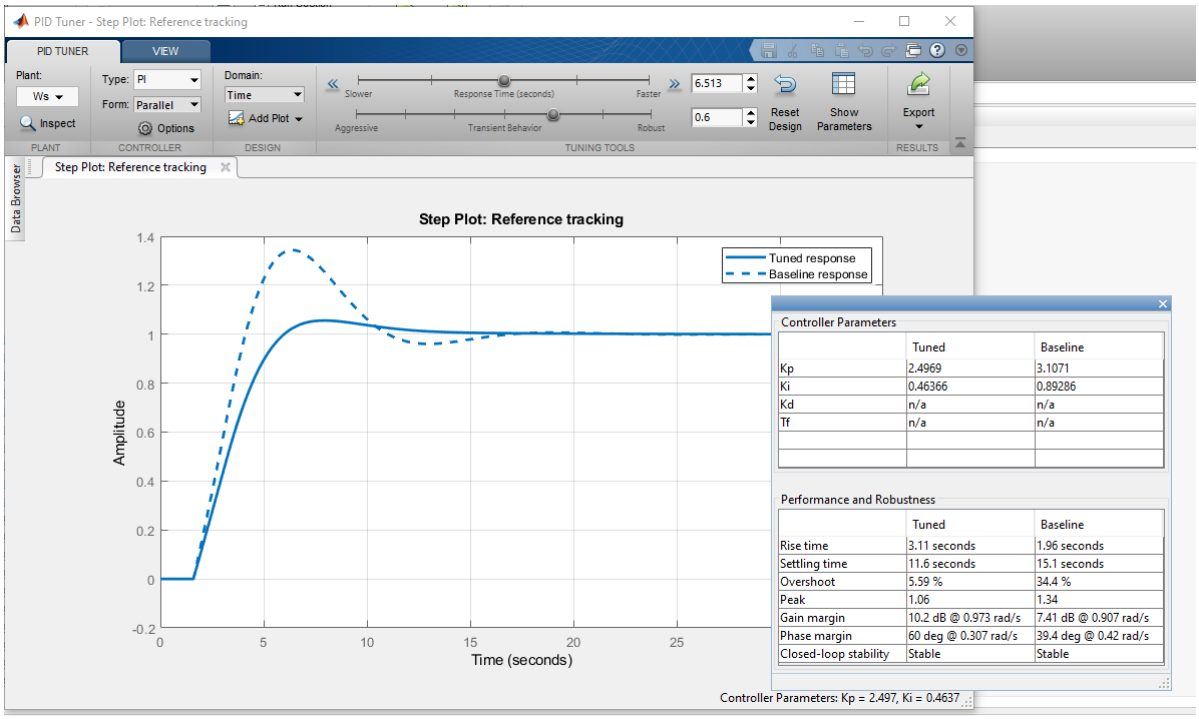


Рисунок 2.7. Результат работы pidTuner

В окне Controller parameters (параметры контроллера) показаны коэффициенты усиления пропорциональной, интегральной и дифференциальной составляющих. В окне Performance and robustness (Производительность и надежность) показаны параметры:

- rise time (время нарастания),
- settling time (время установления),
- overshoot (перерегулирование),
- peak (пик). [16]

Меняя параметр Response time (время реакции), можно менять установившееся время, время нарастания и перерегулирование, исходя из этого меняется и время переходного процесса, и коэффициенты усиления. [16]

Как видно из рисунка pidTuner предложил другие коэффициенты усиления для регулирования, что улучшает модель ПИД регулирования уменьшая время установления, пик и процент перерегулирования.

$K_p = 2.4969$

$K_i = 0.46366$

Далее представлен график, полученный в примере из практической работы (рис. 2.8). Графики практически совпали, есть небольшие расхождения в проценте перерегулирования и времени установления.

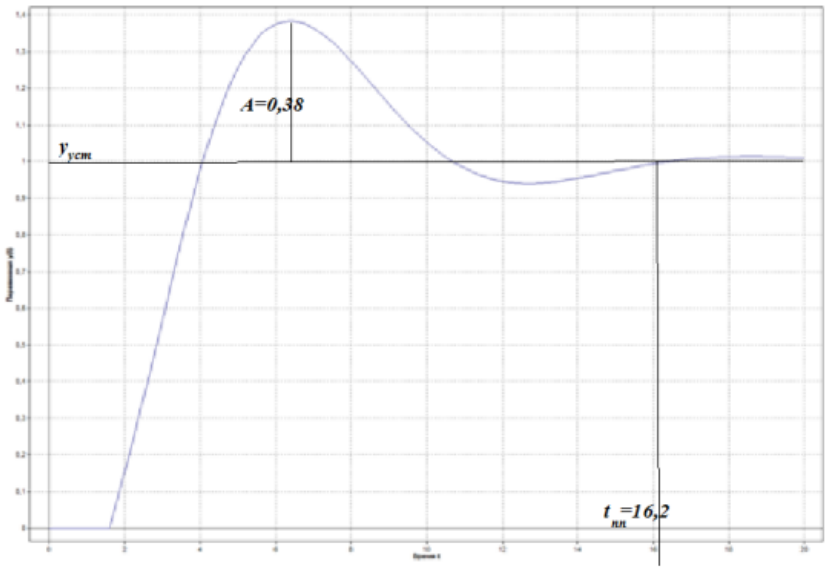


Рисунок 2.8. Результат из практического материала. [13]

2.2.4 Метод AMIGO

Параметры настройки*PID* –регулятора по методу *AMIGO*

Передаточная функция объекта управления	K_P	K_i	K_d
$W_o = \frac{K}{Ts+1}\cdot\exp(-\tau\cdot s),$	$\frac{1}{K}\cdot\left(0{,}2+0{,}45\frac{T}{\tau}\right)$	$\frac{\tau+0{,}1T}{0{,}4\cdot\tau^2+0{,}8\cdot T\cdot\tau}K_P$	$\frac{0{,}5\cdot\tau\cdot T}{0{,}3\cdot\tau+T}K_P$

$$W_o = \frac{K}{Ts+1}\cdot\exp(-\tau\cdot s) = \frac{0.7}{5{,}8\cdot S+1}\cdot\exp(-1{,}6\cdot\tau)$$

$$K_P = K_1 = \frac{1}{K}\cdot\left(0{,}2+0{,}45\frac{T}{\tau}\right) = \frac{1}{0{,}7}\cdot\left(0{,}2+0{,}45\frac{5.8}{1.6}\right) = 2{,}61$$

$$K_i = K_0 = \frac{\tau+0{,}1T}{0{,}4\cdot\tau^2+0{,}8\cdot T\cdot\tau}K_P = \frac{1{,}6+0{,}1\cdot 5{,}8}{0{,}4\cdot 1{,}6^2+0{,}8\cdot 5{,}8\cdot 1.6}\cdot 2{,}61 = 0.673$$

Рисунок 2.9. Пример задания [13]

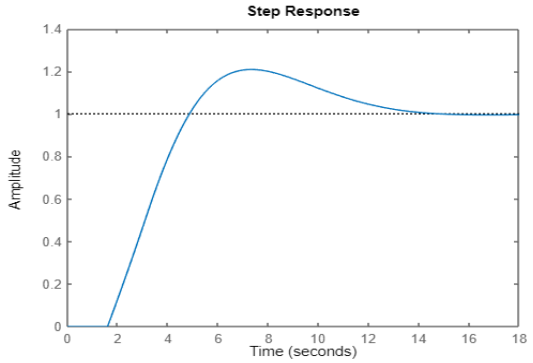
Решение с помощью MATLAB

```
K =     0.7
T =     5.8
tau =   1.6

%Transfer function Object
s=tf('s')
Ws = K/(T*s+1)*exp(-tau*s)
%PI Control
Kp_AMIGO = 1/K*(0.2 + 0.45*T/tau)
Ki_AMIGO = (tau+0.1*T)/(0.4*tau^2+0.8*T*tau)*Kp

c_AMIGO = pid(Kp,Ki)
tf(c_AMIGO)

m_PI_AMIGO = feedback(C*Ws,1)
figure(1)
```

```
stepplot(m_PI_AMIGO)
stepinfo(m_PI_AMIGO)
```

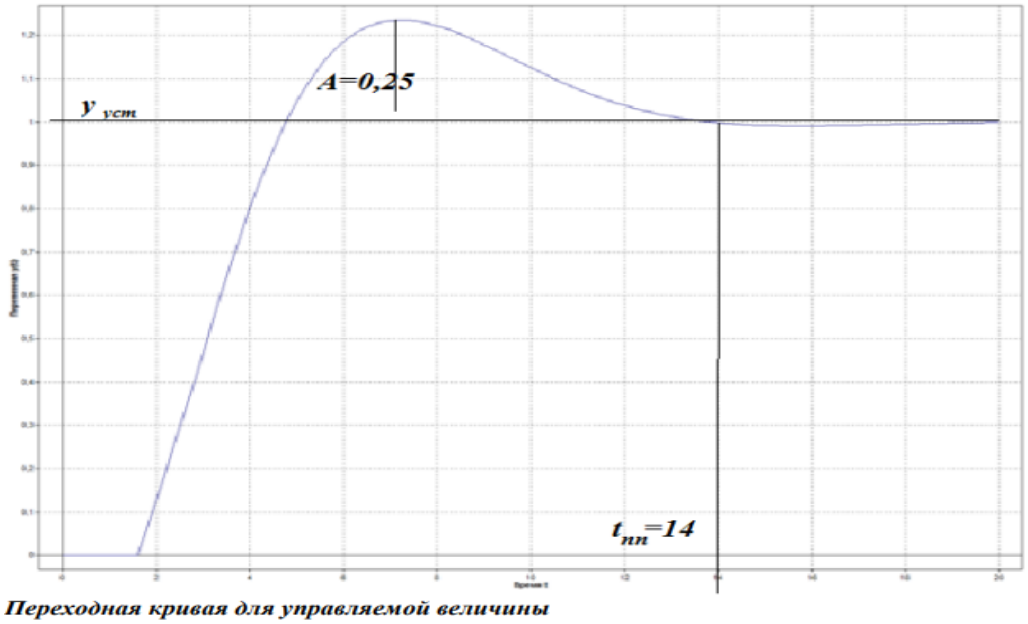
```
Kp = 2.6161
Ki = 0.6751
C =

             1
  Kp + Ki * ---
             s

  with Kp = 2.62, Ki = 0.675

Continuous-time PI controller in parallel form.
ans =

  2.616 s + 0.6751
  ----------------
         s
```

```
Continuous-time state-space model.
```




```
ans = struct with fields:
          RiseTime: 2.4945
     TransientTime: 13.0840
      SettlingTime: 13.0840
       SettlingMin: 0.9015
       SettlingMax: 1.2084
         Overshoot: 20.8384
        Undershoot: 0
              Peak: 1.2084
          PeakTime: 7.3551
```

Рисунок 2.10. Результат работы скрипта



Переходная кривая для управляемой величины

Рисунок 2.11. Результат из практического материала. [13]

Результат работы (рис 2.10) также практически одинаковый с представленным в примере (рис. 2.11). В практической работе перерегулирование и время измерялось вручную.

2.2.5 Метод Skogestad

Параметры настройки *PID* –регулятора по методу Skogestad

Передаточная функция объекта управления	K_P^*	K_i^*	K_d^*,
$\frac{K}{T_1 s+1}\cdot\exp(-\tau\cdot s),$	$\frac{T_1}{K(T_c+\tau)}$	$\min[T_1, 4(T_c+\tau)]$	-

T_c- ожидаемая постоянная времени (desired response time)

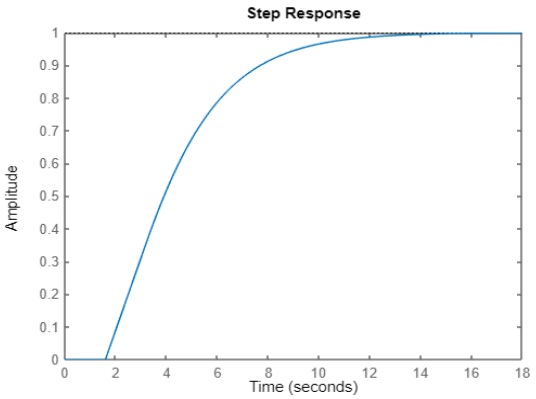
$$K_P^* = \frac{T_1}{K(T_c+\tau)} = \frac{5{,}8}{0{,}7(2{,}9+1{,}6)} = 1{,}84$$
$$K_i^* = \min[T_1, 4(T_c+\tau)] = \min[5{,}8, 4\cdot(5{,}8+1{,}6)] = 5{,}8$$

$$K_P = K_1 = \frac{K_P^*(K_i^*+K_d^*)}{K_i^*} = \frac{1{,}84\cdot 5{,}8}{5{,}8} = 1{,}84$$
$$K_i = K_0 = \frac{K_P^*}{K_i^*} = \frac{1{,}84}{5{,}8} = 0{,}37$$

Рисунок 2.12. Пример задания [13]

Результат работы скрипта см. Приложение 1 выдал следующие результаты (рис 2.13)

```
  with Kp = 1.84, Ki = 0.317

Continuous-time PI controller in parallel form.
ans =

  1.841 s + 0.3175
  ----------------
         s
```

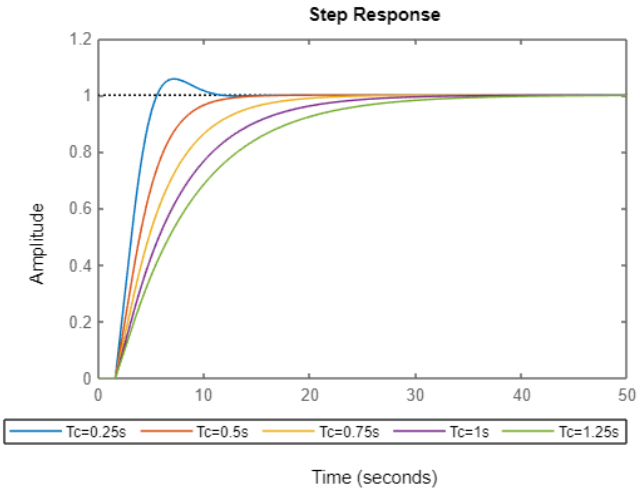



```
ans = struct with fields:
        RiseTime: 5.6540
    TransientTime: 11.1991
     SettlingTime: 11.1991
      SettlingMin: 0.9012
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 26.3457
```

Рисунок 2.13 Результат метода Skogestad

В практическом примере ожидаемая постоянная времени T_c = T*0.5 = 2,9 сек.

Были рассмотрены коэффициенты от 0.25 до 1.25 с шагом 0.25. В результате вычислений с помощью MATLAB были получены следующие результаты



t = 5×7 table

		Kp	Ki	RiseTime	SettlingTime	Overshoot	Peak	PeakTime
1	0.25	2.7166	0.4684	2.8232	9.6439	5.7110	1.0571	7.1260
2	0.5	1.8413	0.3175	5.6540	11.1991	0	1	26.3457
3	0.75	1.3926	0.2401	9.0182	17.6330	0	0.9995	32.6540
4	1	1.1197	0.1931	12.3200	23.5962	0	0.9998	48.5875
5	1.25	0.9362	0.1614	15.5760	29.4254	0	0.9999	68.9801

Рисунок 2.14. Результаты вычислений при различных коэффициентах для ожидаемой постоянной времени

В приложении 2 представлен код данного скрипта.

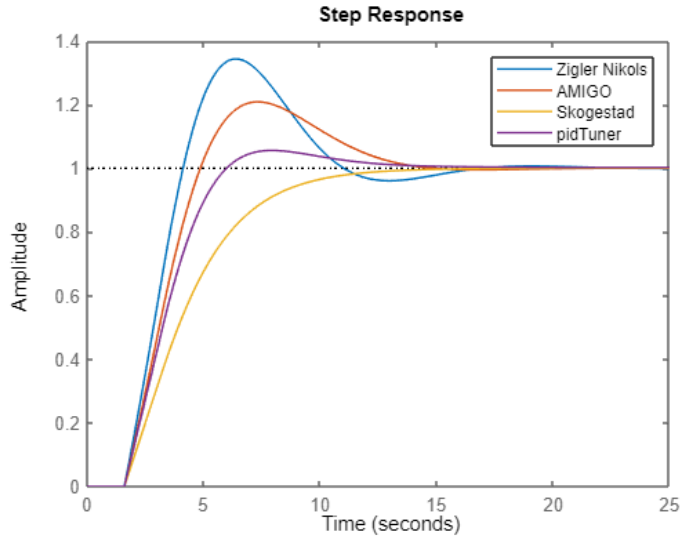
По результатам вычислений можно определить, что ожидаемая постоянная времени при коэффициенте 0.5 (2.9 сек) показала наиболее лучшие результаты:

- rise time (время нарастания) – 5.65 сек,
- settling time (время установления) 11.20 сек,
- overshoot (перерегулирование) 0%,
- peak (пик) - 1

При понижении ожидаемой постоянной времени (меньше 0,5) переходный процесс апериодический с перерегулированием, а при повышении – монотонный с возрастанием времени переходного процесса.

2.2.6 Анализ результатов

Для анализа результата работы также был написан код, который выводит в таблицу основные показатели при расчете коэффициентов различными методами и функциями MATLAB. Также представлен график переходных кривых для объекта управления (см. Приложение 1).



t = 7×4 table

		Zigler Nickols	AMIGO	Skogestad	pidTuner
1	Kp	3.1071	2.6161	1.8413	2.4969
2	Ki	0.8929	0.6751	0.3175	0.4637
3	RiseTime	1.9639	2.4945	5.6540	3.1060
4	SettlingTime	15.0802	13.0840	11.1991	11.5968
5	Overshoot	34.3287	20.8384	0	5.5760
6	Peak	1.3433	1.2084	1	1.0558
7	PeakTime	6.3865	7.3551	26.3457	7.9484

Рисунок 2.15. Итоговая таблица и график методов ПИ регулятора

Вывод. Из рассмотренных 3-х методов и pidTuner наилучшим является метод *Skogestad*, обеспечивающий наименьшее (почти 0) перерегулирование.

3. ПРОГРАММНАЯ СРЕДА SIMULINK

MATLAB предоставляет большое количество инструментов, где можно использовать графическую среду для решения различных задач. *Simulink* является одним из популярных инструментов, который позволяет моделировать, разрабатывать и выполнять программу в виде блоков.

Например, есть блок суммирования, блок константы, блок вывода и осциллограф для того, чтобы отобразить вывод, сгенерированный блоком суммирования. Также существует ряд встроенных примеров, которые можно запустить и узнать о работе различных блоков, присутствующих в *Simulink*.

В *Simulink* ПИД-регулятор может быть разработан с использованием двух разных способов. Первый способ с использованием блоков суммирования, интеграторов, дифференцирования и т.д. и второй метод с использованием встроенного блока pid.

На рисунке представлена схема построения ПИД регулятора (рис. 3.1). По данной схеме будет построена модель первым методом.

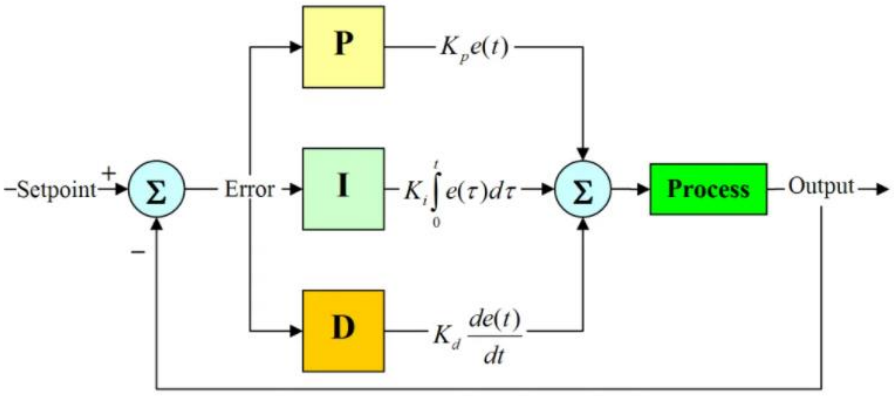
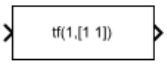


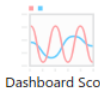
Рисунок 3.1. Схема ПИД регулятора [17]

Для создания схемы используются следующие блоки (см. Таблицу 3.1). Для описания блоков использовалась официальная документация по *Simulink* (https://www.mathworks.com/help/simulink/)

Таблица 3.1 Описание блоков

Блок	Описание
 Step	Step (шаг) обеспечивает между двумя уровнями переход на указанное значение шага в указанное время. (Step Time = 1, Final Time = 1)

	Sum - сложение/вычитание данных. Это могут быть скалярные, векторные, матричные входные данные.
	Gain умножение на константное значение (усиление). Входные данные и коэффициент усиления могут быть скалярные, векторные, матричные. На схеме представлены скалярные значения коэффициентов усиления ПИД регулятора
	Integrator применяется для интегрирования входного сигнала. На схеме используется входной сигнал – ошибка – e(t).
	Derivative – выполняется для численного дифференцирование входного сигнала. На схеме используется входной сигнал – ошибка – e(t).
	Saturation – выполняется для ограничения величины выходного сигнала. На схеме заданы верхний и нижний пределы заданы 5 и 0
	LTI (Linear Time Invariant) system - используется для передаточной функции объекта управления (Process) На схеме используется переменная Ws из полученного скрипта, можно также использовать функцию tf([0 0.7],[5.8 1], 'inputDelay',1.6) в свойстве LTI system variable. В Simulink есть блок Transfer Fcn для применения передаточной функции с использованием коэффициентов числителя и знаменателя. Так как в схеме используется переменная, то применятся блок LTI system
	Mux – применяется для объединения входных сигналов в вектор. На схеме применяется для отображения графика в блоке Scope.
	Scope (осциллограф) – применяется для построения графиков исследуемых сигналов по времени. На схеме это u(t), y(t)

 Dashboard Scope	Dashboard Scope применяется для просмотра исследуемых сигналов во время симуляции на дисплее осциллографа сразу на схеме.

На схеме (рис. 3.2) используются коэффициенты, которые были получены с помощью метода Циглера-Николса.

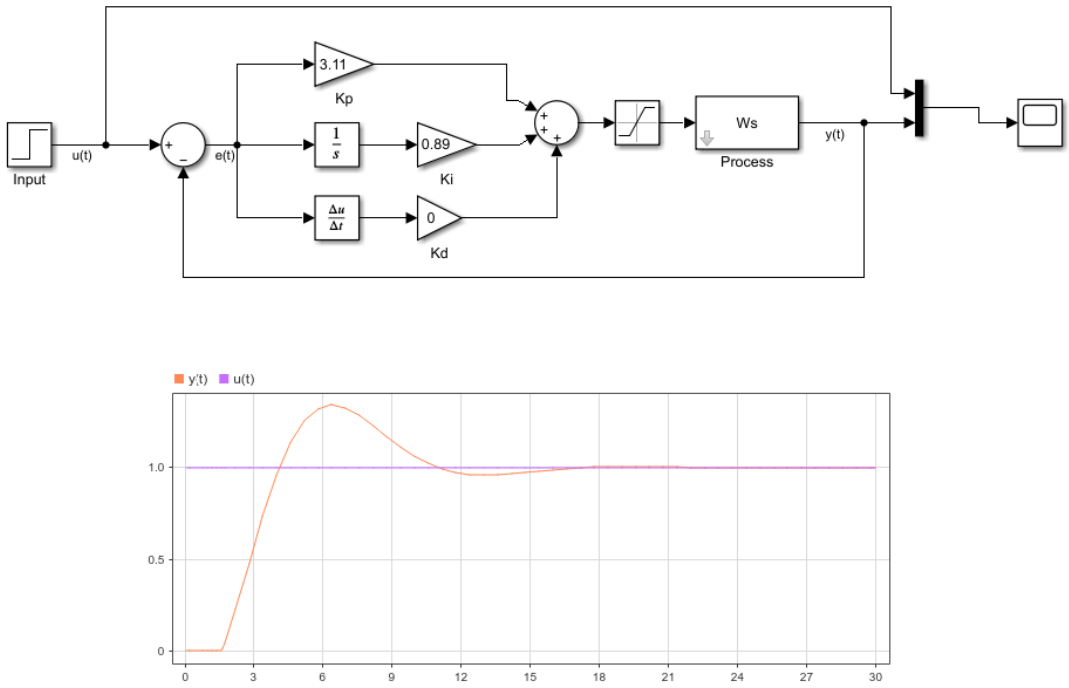


Рисунок 3.2. Схема ПИ регулирования (первый способ)

Второй способ — это создание модели с помощью блока pid, который реализует ПИД-регулятор (PID, PI, PD, P, I). На рисунке 3.3 представлены параметры настройки ПИД-регулятора.

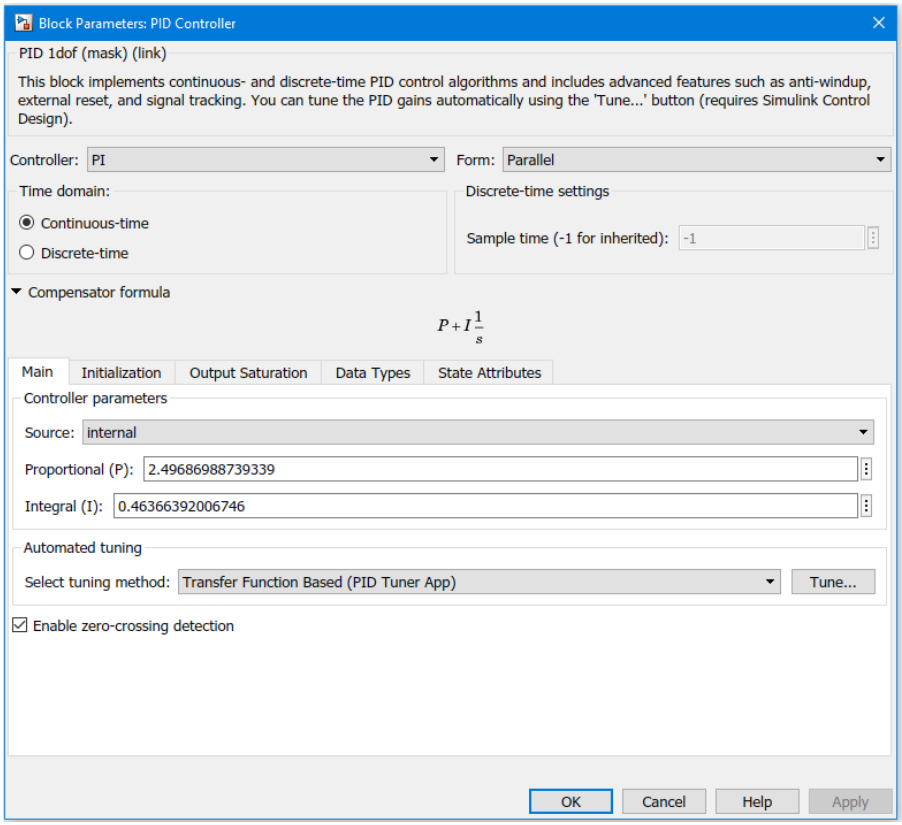


Рисунок 3.3. Окно настройки параметров PID регулятора

Используемые свойства для представленной модели на рис. 3.4:

- Controller – можно выбрать тип PID контроллера
- В зависимости от выбора типа PID контроллера можно установить коэффициенты (Proportional (P), Integral (I))
- Кнопка Tune - для автоматической настройки коэффициентов регулятора. После нажатия были предложены коэффициенты P и I.
- На вкладке Output Saturation – установлены ограничения для выходного сигнала: верхний (5) и нижний предел (0).

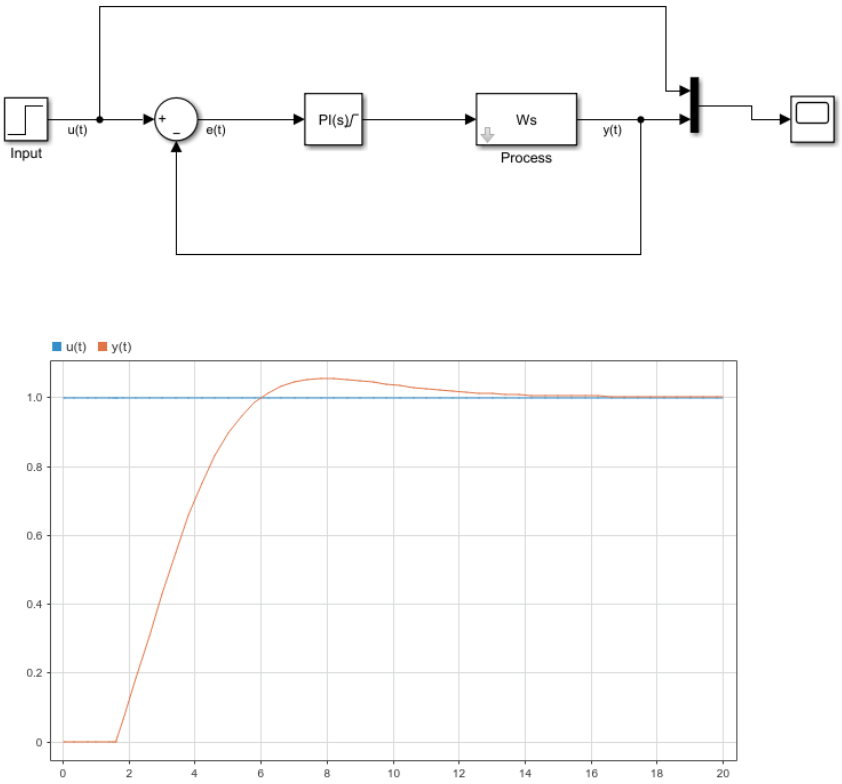


Рисунок 3.4. Модель ПИ регулирования (второй способ - блок pid)

При использовании Simulink можно построить модели для ПИД регулирования без использования программирования. Также можно применить визуализацию, настроить автоматическое вычисление коэффициентов для различных типов ПИД контроллеров. Используя окно параметров блока pid можно настраивать различные параметры и при запуске симуляции сразу увидеть результат.

4. РАЗВИТИЕ ПРИЛОЖЕНИЯ

При изучении возможностей MATLAB и Simulink можно использовать данные технологии на реальном примере.

На данный момент создан учебный стенд, который имеет возможность нагревания воды до определенной температуры (в представленном скрипте до температуры 65^{o} C) и поддержку данной температуры при изменении мощности блоков питания для нагрева ТЭНов (Трубчатый электронагреватель).

Стенд работает на базе микроконтроллера Arduino. Создан скрипт в MATLABe, который считывает температуру воды с помощью цифрового температурного датчика DS18B20 через Serial port. Данные для анализа и построения модели сохраняются в csv файл, каждые 10 секунд до нагрева температуры до 65 градусов. В приложение 3 представлен код для Arduino, в приложение 4 код на MATLAB.

Также создан учебный материал, который основан на решении практических заданий по предмету RAA0590 Automaatjuhtimissüsteemid и содержит решение 5 практических заданий. В дальнейшем учебный материал можно дополнить решениями других практических заданий, связанных с автоматическими системами управления (см. Приложение 5 – представлен Пример 1).

1. Решение дифференциальных уравнений с использованием преобразований Лапласа
2. По заданной передаточной функции записать дифференциальное уравнение
3. Передаточная функция Transfer
4. Идентификация параметров передаточной функции объекта по переходной кривой
5. PID Controller. Метод Ziegler-Nichols

Студенты, которые обучаются на специальности, связанные с изучением систем автоматического управления, могут применить MATLAB для создания ПИД регулятора контроля температуры воды на данном учебном стенде. Решение можно начать от построения математической модели, идентификации объекта до вычисления коэффициентов различных типов ПИД регуляторов (ПИ, П, ПИД и т.д.), анализа полученных данных. Также возможно применить алгоритмы машинного обучения, например, для определения коэффициентов ПИД регулирования с помощью эвристических методов и искусственного интеллекта.

Наличие в MATLAB возможностей работы с массивами и графического вывода данных позволяет эффективно использовать эту среду программирования для решения задач нахождения оптимальных настроек регуляторов.

ЗАКЛЮЧЕНИЕ

Тема выпускной работы: Разработка приложения расчёта настроек ПИД регулятора для систем автоматического регулирования с помощью MATLAB

Ключевой проблемой в использовании ПИД-регулятора является расчет и настройка его коэффициентов.

В работе созданы приложения в среде MATLAB для расчёта коэффициентов ПИД регулятора для систем автоматического регулирования с использованием методов Циглера-Николса, *AMIG*O и *Skogestad*.

В работе демонстрируются расчеты коэффициентов ПИ регулятора с помощью трех методов: Циглера-Николса, *AMIGO, Skogestad*. Приведены результаты расчета настроек ПИ-регулятора для контрольного примера и проведено сравнение переходных процессов при прямом критерии качества переходного процесса-перерегулирование от заданного значения, который является определяющим для тепловых процессов. Наилучшие результаты дает метод *Skogestad*, который подтверждает результаты зарубежных исследований. Проанализированы переходные процессы при выборе разных по величине показателей желаемой постоянной времени процесса, которые позволяют получит переходные процессы от монотонного с различным временем переходного процесса до апериодического с перерегулированием. Также использована встроенная функция pidTuner MATLAB, которая дает оптимальные коэффициенты для ПИД регулирования.

Для решения задачи расчета коэффициентов ПИ регулятора были также построены 2 модели в *Simulink*. При сравнении результаты расчетов параметров оказываются близкими, что позволяет сделать вывод об эффективности работы приложений.

Рассмотрены возможности альтернативной постановки задачи синтеза настроек регуляторов и применение предложенных методов для нахождения настроек регуляторов для студентов, которые обучаются на специальностях, связанных с программированием контроллеров, проектированием автоматизированных систем, инженеров, работающих в области промышленной автоматизации.

Разработанные приложения дают возможность не только пользоваться готовыми решениями, но и уметь самостоятельно рассчитать, визуализировать, проанализировать полученные данные.

KOKKUVÕTTE

Lõputöö teema: Rakenduse arendamine automaatjuhtimissüsteemide PID-regulaatori seadistuste arvutamiseks MATLABi abil

PID-kontrolleri kasutamise põhiprobleem on selle koefitsientide arvutamine ja seadistamine.

Selles töös loodi MATLABi keskkonnas rakendus automaatjuhtimissüsteemide PID-regulaatorite koefitsientide arvutamiseks, kasutades suhteliselt uusi AMIGO ja Skogestadi meetodeid.

Töös näidatakse PI-regulaatori koefitsientide arvutusi kolme meetodi abil: Ziegler-Nichols, AMIGO, Skogestad. Esitatakse PI-regulaatori seadistuste arvutamise tulemused testjuhtumi jaoks ja võrreldakse siirdeprotsesse otsese kvaliteedikriteeriumi juures - antud väärtusest ümberreguleerimine, mis on soojusprotsesside jaoks määratav. Parimad tulemused on saadud Skogestadi meetodi abil, mis kinnitab välismaiste uuringute tulemusi. Siirdeprotsesse analüüsiti, valides soovitud protsessi ajakonstandi erinevad väärtused. Samuti kasutati sisseehitatud funktsiooni pidTuner MATLAB, mis annab PID-reguleerimise optimaalsed koefitsiendid.

PI-kontrolleri koefitsientide arvutamiseks ehitati Simulinkis 2 mudelit. Parameetrite arvutamise tulemused on võrdlemisel lähedased, mis võimaldab järeldada, et rakenduste töö on efektiivne.

Vaadeldakse regulaatorite seadistuste sünteesi ülesande alternatiivse püstitamise võimalusi ja pakutavate meetodite rakendamist regulaatorite seadistuste leidmiseks üliõpilastele, kes õpivad kontrollerite programmeerimise, automatiseeritud süsteemide projekteerimisega seotud erialadel ja tööstusautomaatika valdkonnas töötavatele inseneridele.

Loodud rakendused võimaldavad lisaks valmislahenduste kasutamisele ka iseseisvalt arvutada, visualiseerida ja analüüsida saadud andmeid.

SUMMARY

The topic of the graduation thesis is Application development to calculate PID regulator settings for automatic control systems using MATLAB.

The key problem in using a PID regulator is the calculation and adjustment of its coefficients.

In the work, applications have been created in the MATLAB environment for calculating the coefficients of the PID regulator for automatic control systems using relatively new methods of AMIGO and Skogestad.

The article demonstrates the calculations of the coefficients of the PI regulator using three methods: Ziegler-Nichols, AMIGO, Skogestad. The results of calculating the settings of the PI regulator for a test example are presented, as well as a comparison of transients with a direct criterion for the quality of the transient (overshoot) from a given value, which is crucial for thermal processes. The best results are obtained by the Skogstad method, which confirms the results of foreign studies. Transients were analyzed by selecting different values of the required process time constant. The built-in PID tuner MATLAB function is also used, which gives optimal coefficients for PID regulation.

To solve the problem of calculating the coefficients of the PI regulator, 2 models in Simulink were also built. When comparing the results of the parameters' calculations are close, which allows us to conclude about the efficiency of the applications.

The possibilities of an alternative formulation of the problem of synthesizing regulators' settings and the application of the proposed methods for finding regulators' settings for students who study in specialties related to controller programming, design of automated systems, engineers working in the field of industrial automation are considered.

The developed applications make it possible not only to use ready-made solutions, but also to be able to independently calculate, visualize, and analyze the data obtained.

ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА И ССЫЛКИ

1. MATLAB. Документация на русском языке. [*Online*] https://exponenta.ru/matlab (10.09.2021)
2. ПИД-регуляторы. [*Online*] https://www.bookasutp.ru/Chapter5_1.aspx (10.09.2021)
3. Hu, Z.-R. Analytical design of PID decoupling control for TITO processes with time delays [Text] / Z.-R. Hu, D.-H. Li, J. Wang, F. Xue // Journal of Computers. – 2011. – Vol. 6. – N 6. – P. 1064-1070.
4. Hu, W. An analytical method for PID controller tuning with specified gain and phase margins for integral plus time delay processes [Text] / W. Hu, G. Xiao, X. Li // ISA Transactions. – 2011. – Vol. 50. – N 2. – P. 268-276.
5. Shamsuzzoha, M. Analytical design of PID controller cascaded with a lead-lag filter for time-delay processes [Text] / M. Shamsuzzoha, S. Lee, M. Lee // Korean Journal of Chemical Engineering. – 2009. – Vol. 26. – N 3. – P. 622-630.
6. Isaksson, A.J. Analytical PID parameter expressions for higher order systems [Text] / A.J. Isaksson, S.F. Graebe // Automatica. – 1999. – Vol. 35. – N 6. – P. 1121-1130.
7. Частотный метод параметрического синтеза ПИД регулятора для стационарных, интервальных и многосвязных САУ. С. Михалевич. 2015. http://earchive.tpu.ru/bitstream/11683/21610/1/dis00007.pdf (10.12.2021)
8. Martins, F.G. Tuning PID controllers using the ITAE criterion [Text] / F.G. Martins // International journal of engineering education. – 2005. – Vol. 21. – N 3. – P. 867-873.
9. Sanchis, R. Tuning of PID controllers based on simplified single parameter optimization [Text] / R. Sanchis, J.A. Romero, P. Balaguer // International journal of control. – 2010. – Vol. 83. – N 9. – P. 1785-1798
10. Для чего применяют ПИД-регулятор в промышленности? [*Online*] https://vk.com/@industrial.automation.experts-dlya-chego-primenyaut-pid-regulyator-v-promyshlennosti (10.12.2021)
11. Естественнонаучный журнал «Точная наука» https://t-nauka.ru/wp-content/uploads/v64.pdf (11.12.2021)
12. Практикум по предмету Проектирование технологических процессов, визуализация и контроль. С. Чекрыжов. (moodle.taltech.ee) (10.12.2021)
13. Практическая работа 5. Проектирование систем автоматической регулировки. Использование компьютера в системах автоматической настройки. S.Chekryzhov (moodle.taltech.ee) (10.12.2021)

14. Panagopoulos H., Åström K.J., Hägglund T. Design of PID controllers based on constrained optimization // IEE Procedings – Control, Theory and Applications, Vol. 149, N. 1, 2002. –p. 32-40. (10.12.2021)
15. Моделирование и симуляция в Simulink – это просто! [*Online*] https://www.mathworks.com/videos/simulations-made-easy-with-simulink-82456.html (11.12.2021)
16. Автоматизированная настройка ПИД-регулятора для объекта управления следящей системы с использованием программного пакета MATLAB Simulink. Филиппов А. В., Косолапов М. А., Маслов И. А., Тарасова Г. И. 2015. https://3minut.ru/images/PDF/2015/18/avtomatizirovannaya-nastrojka.pdf (11.12.2021)
17. PID controller design using Simulink MATLAB. [*Online*] https://microcontrollerslab.com/pid-controller-design-simulink/ (11.12.2021)
18. Частотный метод параметрического синтеза пидрегулятора для стационарных, интервальных и многосвязных САУ. Михалевич С.С. http://earchive.tpu.ru/bitstream/11683/21610/1/dis00007.pdf (12.12.2021)

Приложение 1. Live Script MATLAB

Расчет коэффициентов ПИ регулятора. Live Script экспортирован как Microsoft Word документ

Метод Циглера-Никольса

```
% data = {'Enter K';'Enter time T';'Enter tau (time delay)'}
% answers = inputdlg(data)
%
% K = sscanf(answers{1},'%f')     % 0.7
% T = sscanf(answers{2},'%f')     % 5.8
% tau = sscanf(answers{3},'%f')   % 1.6

K =     0.7;
T =     5.8;
tau =   1.6;
```

Передаточная функция объекта

```
%Transfer function Object
s=tf('s');
Ws = K/(T*s+1)*exp(-tau*s)
```

```
Ws =


                 0.7
  exp(-1.6*s) * ---------
                5.8 s + 1


Continuous-time transfer function.
```

```

```

Передаточная функция ПИ регулятора

Регулятор	Апериодический процесс	Процесс с перерегулированием 20 %	Процесс с минимальным временем регулирования
P	$K_1 = \frac{0{,}3 \cdot T}{K \cdot \tau}$	$K_1 = \frac{0{,}7 \cdot T}{K \cdot \tau}$	$K_1 = \frac{0{,}9 \cdot T}{K \cdot \tau}$
I	$K_0 = \frac{1}{4{,}5 \cdot K \cdot \tau}$	$K_0 = \frac{1}{1{,}7 \cdot K \cdot \tau}$	$K_0 = \frac{1}{1{,}7 \cdot K \cdot \tau}$
PI	$K_1 = \frac{0{,}6 \cdot T}{K \cdot \tau}$, ✓	$K_1 = \frac{0{,}7 \cdot T}{K \cdot \tau}$, ✓	$K_1 = \frac{T}{K \cdot \tau}$,

```
%PI Control
Kp_ZN = (0.6*T)/(K*tau) %K1
```

```
Kp_ZN = 3.1071
```

```
Ki_ZN = 1/(K*tau) %K0
```

```
Ki_ZN = 0.8929
```

```
c_ZN = pid(Kp_ZN,Ki_ZN)
```

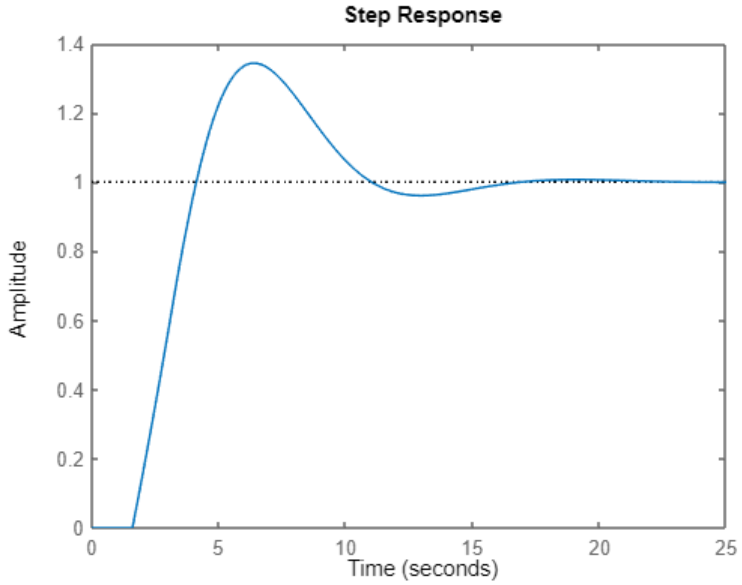
```
c_ZN =

             1
  Kp + Ki * ---
             s

  with Kp = 3.11, Ki = 0.893

Continuous-time PI controller in parallel form.
```

```
tf(c_ZN)
```

```
ans =

  3.107 s + 0.8929
  ----------------
         s

Continuous-time transfer function.
```

Переходная кривая для управляемой величины

```
m_PI_ZN = feedback(c_ZN*Ws,1);
figure(1)
stepplot(m_PI_ZN)
```




```
[info_ZN] = stepinfo(m_PI_ZN)
```

```
info_ZN = struct with fields:

          RiseTime: 1.9639
     TransientTime: 15.0802
      SettlingTime: 15.0802
       SettlingMin: 0.9170
       SettlingMax: 1.3433
         Overshoot: 34.3287
        Undershoot: 0
              Peak: 1.3433
          PeakTime: 6.3865
```

```
% pidTuner(Ws,c_ZN)
```

Метод AMIGO

Передаточная функция ПИ регулятора

Параметры настройки*PID* –регулятора по методу *AMIGO*

Передаточная функция объекта управления	K_P	K_i	K_d
$W_o = \frac{K}{Ts+1} \cdot \exp(-\tau \cdot s),$	$\frac{1}{K} \cdot \left(0{,}2 + 0{,}45\frac{T}{\tau}\right)$	$\frac{\tau + 0{,}1T}{0{,}4 \cdot \tau^2 + 0{,}8 \cdot T \cdot \tau} K_P$	$\frac{0{,}5 \cdot \tau \cdot T}{0{,}3 \cdot \tau + T} K_P$

```
%PI Control
Kp_AMIGO = 1/K*(0.2 + 0.45*T/tau)
```

```
Kp_AMIGO = 2.6161
```

```
Ki_AMIGO = (tau+0.1*T)/(0.4*tau^2+0.8*T*tau)*Kp_AMIGO
```

```
Ki_AMIGO = 0.6751
```

```
c_AMIGO = pid(Kp_AMIGO,Ki_AMIGO)
```

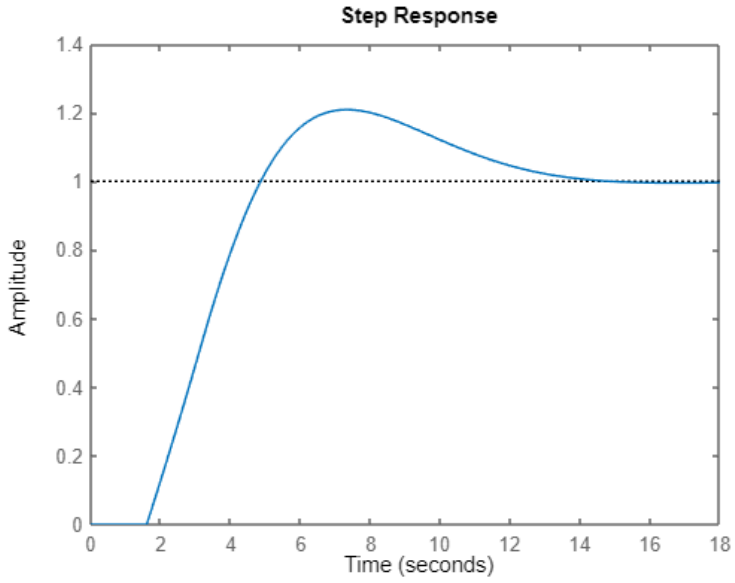
```
c_AMIGO =

             1
  Kp + Ki * ---
             s

  with Kp = 2.62, Ki = 0.675

Continuous-time PI controller in parallel form.
```

```
tf(c_AMIGO)
```

```
ans =

  2.616 s + 0.6751
  ----------------
         s

Continuous-time transfer function.
```

Переходная кривая для управляемой величины

```
m_PI_AMIGO = feedback(c_AMIGO*Ws,1);
figure(1)
stepplot(m_PI_AMIGO)
```




```
[info_AMIGO] = stepinfo(m_PI_AMIGO)
```

```
info_AMIGO = struct with fields:

         RiseTime: 2.4945
    TransientTime: 13.0840
     SettlingTime: 13.0840
      SettlingMin: 0.9015
      SettlingMax: 1.2084
        Overshoot: 20.8384
       Undershoot: 0
             Peak: 1.2084
         PeakTime: 7.3551
```

Метод Skogestad

Передаточная функция ПИ регулятора

Параметры настройки *PID* –регулятора по методу Skogestad

Передаточная функция объекта управления	$K_P^{'}$	$K_i^{'}$	$K_d^{'}$,
$\frac{K}{T_1 s+1}\cdot\exp(-\tau\cdot s),$	$\frac{T_1}{K(T_c+\tau)}$	$\min[T_1,4(T_c+\tau)]$	-

T_c- ожидаемая постоянная времени (desired response time)

$$K_P^{'}=\frac{T_1}{K(T_c+\tau)}=\frac{5{,}8}{0{,}7(2{,}9+1{,}6)}=1{,}84$$
$$K_i^{'}=\min[T_1,4(T_c+\tau)]=\min[5{,}8,4\cdot(5{,}8+1{,}6)]=5{,}8$$

$$K_P=K_1=\frac{K_P^{'}(K_i^{'}+K_d^{'})}{K_i^{'}}=\frac{1{,}84\cdot 5{,}8}{5{,}8}=1{,}84$$
$$K_i=K_0=\frac{K_P^{'}}{K_i^{'}}=\frac{1{,}84}{5{,}8}=0{,}37$$

```
Tc = T*0.5
```

```
Tc = 2.9000
```

```
%PI Control
Kp_temp = T/(K*(Tc+tau))
```

```
Kp_temp = 1.8413
```

```
Ki_temp = min(T, 4*(Tc+tau))
```

```
Ki_temp = 5.8000
```

```
Kp_Skogestad = Kp_temp*Ki_temp/Ki_temp
```

```
Kp_Skogestad = 1.8413
```

```
Ki_Skogestad = Kp_temp/Ki_temp
```

```
Ki_Skogestad = 0.3175
```

```
c_Skogestad = pid(Kp_Skogestad,Ki_Skogestad)
```

```
c_Skogestad =


             1
  Kp + Ki * ---
```

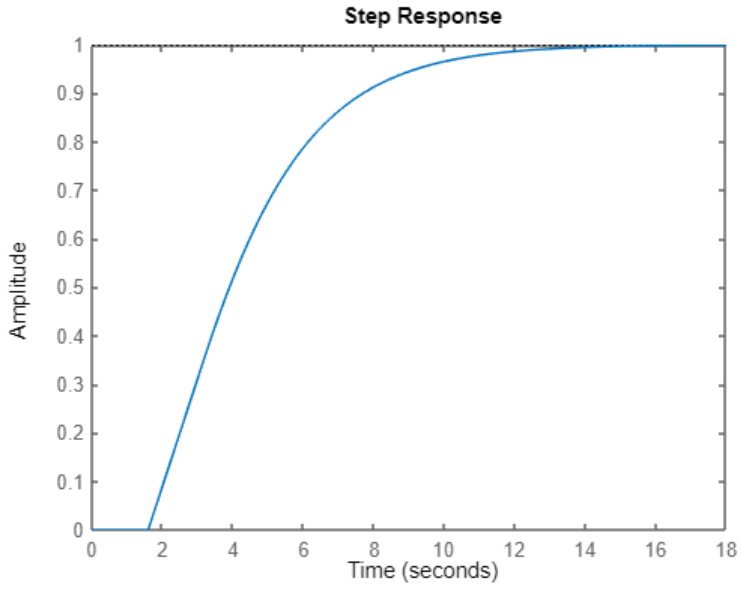
```
            s

  with Kp = 1.84, Ki = 0.317

Continuous-time PI controller in parallel form.
```

```
tf(c_Skogestad)
```

```
ans =


  1.841 s + 0.3175
  ----------------
         s

Continuous-time transfer function.
```

Переходная кривая для управляемой величины

```
m_PI_Skogestad = feedback(c_Skogestad*Ws,1);
figure(1)
stepplot(m_PI_Skogestad)
```




```
[info_Skogestad] = stepinfo(m_PI_Skogestad)
```

```
info_Skogestad = struct with fields:

         RiseTime: 5.6540
```

```
    TransientTime: 11.1991
     SettlingTime: 11.1991
      SettlingMin: 0.9012
      SettlingMax: 1.0000
        Overshoot: 0
       Undershoot: 0
             Peak: 1.0000
         PeakTime: 26.3457
```

pidTuner

```
[c_PI] = pidtune(Ws,'PI')
```

```
c_PI =

            1
  Kp + Ki * ---
            s

  with Kp = 2.5, Ki = 0.464

Continuous-time PI controller in parallel form.
```

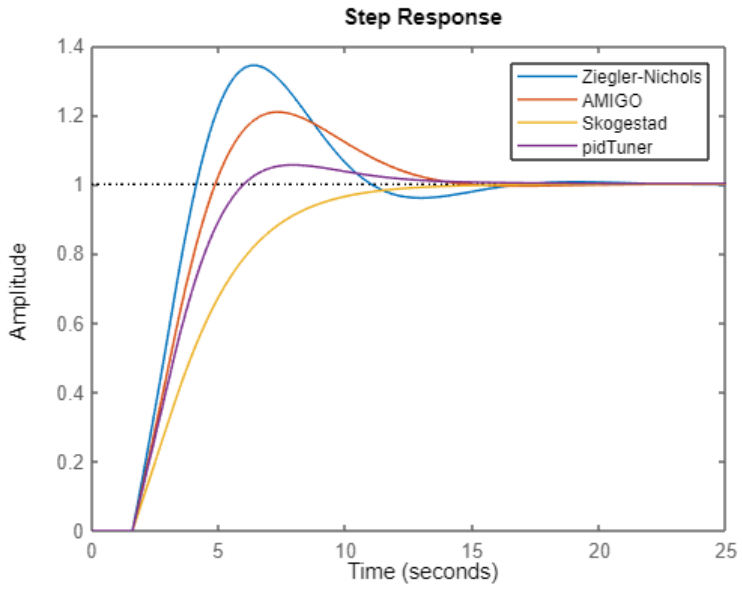
```
t_PI = feedback(c_PI*Ws, 1);
[info] = stepinfo(t_PI)
```

```
info = struct with fields:
         RiseTime: 3.1060
    TransientTime: 11.5968
     SettlingTime: 11.5968
      SettlingMin: 0.9040
      SettlingMax: 1.0558
        Overshoot: 5.5760
       Undershoot: 0
             Peak: 1.0558
```

```
PeakTime: 7.9484
```

Результаты

```
step(m_PI_ZN, m_PI_AMIGO, m_PI_Skogestad, t_PI)
legend('Ziegler-Nichols','AMIGO','Skogestad','pidTuner')
```



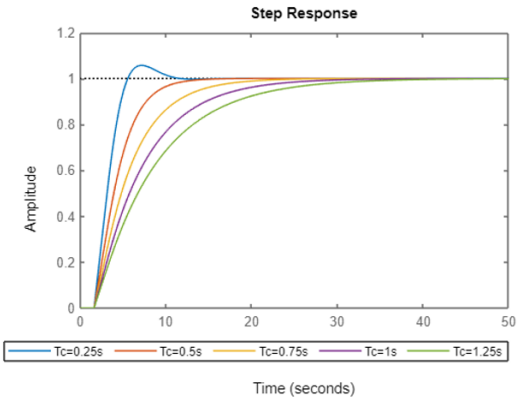

```
info_ZN = [Kp_ZN Ki_ZN info_ZN.RiseTime info_ZN.SettlingTime
info_ZN.Overshoot info_ZN.Peak info_ZN.PeakTime];
info_AMIGO = [Kp_AMIGO Ki_AMIGO info_AMIGO.RiseTime
info_AMIGO.SettlingTime info_AMIGO.Overshoot info_AMIGO.Peak
info_AMIGO.PeakTime];
info_Skogestad = [Kp_Skogestad Ki_Skogestad info_Skogestad.RiseTime
info_Skogestad.SettlingTime info_Skogestad.Overshoot info_Skogestad.Peak
info_Skogestad.PeakTime];
info_pidTuner = [c_PI.Kp c_PI.Ki info.RiseTime info.SettlingTime
info.Overshoot info.Peak info.PeakTime];
t = table(info_ZN(:),info_AMIGO(:),info_Skogestad(:),info_pidTuner(:), ...
    'VariableNames', {'Ziegler-Nichols', 'AMIGO', 'Skogestad','pidTuner'},
...

'RowNames',{'Kp','Ki','RiseTime','SettlingTime','Overshoot','Peak','PeakTim
e'})
```

```
t = 7×4 table
```

	Ziegler-Nichols	AMIGO	Skogestad	pidTuner
1 Kp	3.1071	2.6161	1.8413	2.4969
2 Ki	0.8929	0.6751	0.3175	0.4637
3 RiseTime	1.9639	2.4945	5.6540	3.1060
4 SettlingTime	15.0802	13.0840	11.1991	11.5968
5 Overshoot	34.3287	20.8384	0	5.5760
6 Peak	1.3433	1.2084	1	1.0558
7 PeakTime	6.3865	7.3551	26.3457	7.9484

Приложение 2. Метод Skogestad

```
Tc_array = [0.25 0.5 0.75 1 1.25];
Kp_Skogestad_arr = zeros(1,numel(Tc_array));
Ki_Skogestad_arr = zeros(1,numel(Tc_array));
clear legend
t = table;
for i = 1:numel(Tc_array)
    Kp_temp = T/(K*(Tc_array(i)*T+tau));
    Ki_temp = min(T, 4*(Tc_array(i)*T+tau));
    Kp_Skogestad_arr(i) = Kp_temp*Ki_temp/Ki_temp;
    Ki_Skogestad_arr(i) = Kp_temp/Ki_temp;
    c_Skogestad_arr = pid(Kp_Skogestad_arr(i),Ki_Skogestad_arr(i));
    m_PI_Skogestad_arr = feedback(c_Skogestad_arr*Ws,1);
    step(m_PI_Skogestad_arr)
    legend_text{i} = strcat("Tc=", num2str(Tc_array(i)),'s ');
    [info_Skogestad_arr] = stepinfo(m_PI_Skogestad_arr);
    info_t =
table(Kp_Skogestad_arr(i),Ki_Skogestad_arr(i),info_Skogestad_arr.RiseTime,
info_Skogestad_arr.SettlingTime,info_Skogestad_arr.Overshoot,info_Skogestad_arr
.Peak, info_Skogestad_arr.PeakTime);
    t = [t; info_t];
    hold all
end
legend = legend(legend_text, 'location', 'southoutside', 'orientation',
'horizontal');
hold off

t.Properties.RowNames = strsplit(num2str(Tc_array));
t.Properties.VariableNames =
{'Kp','Ki','RiseTime','SettlingTime','Overshoot','Peak','PeakTime'};
t
```



t = 5×7 table

		Kp	Ki	RiseTime	SettlingTime	Overshoot	Peak	PeakTime
1	0.25	2.7166	0.4684	2.8232	9.6439	5.7110	1.0571	7.1260
2	0.5	1.8413	0.3175	5.6540	11.1991	0	1	26.3457
3	0.75	1.3926	0.2401	9.0182	17.6330	0	0.9995	32.6540
4	1	1.1197	0.1931	12.3200	23.5962	0	0.9998	48.5875
5	1.25	0.9362	0.1614	15.5760	29.4254	0	0.9999	68.9801

Приложение 3. Код для Arduino

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS_PIN       5

OneWire oneWire(ONE_WIRE_BUS_PIN);
DallasTemperature sensors(&oneWire);

float celcius=0;
unsigned long currentMillis;
int seconds;
const int MINUTES = 5;

void setup() {
  Serial.begin(9600);
  sensors.begin();
  DDRD = B00011100;
  PORTD = B00011100; // Heating elements (pins):2,3,4
}

void loop() {
  currentMillis = millis();
  seconds = currentMillis/1000;
  sensors.requestTemperatures();
  celcius=sensors.getTempCByIndex(0);
  //Serial.print(seconds);
  //Serial.print(": ");
  if(celcius <= 65 && seconds <= 60*MINUTES*3){
    //digitalWrite(HEATING_ELEMENT_1_PIN, LOW);
    PORTD = B00011000;
    if(seconds >= 60*MINUTES)
        PORTD = B00010000;
    if(seconds >= 60*MINUTES*2)
        PORTD = B00000000;
  }
  else{
        PORTD = B00011100;
  }
  Serial.println(celcius);
  Serial.println(PORTD);
  delay(10000);
}
```

Приложение 4. Script MATLAB чтение данных с Arduino через Serial port

```
delete(instrfind({'port'},{'COM9'}));
pserial=serial('COM9','BaudRate',9600);
fopen(pserial);
figure(1);
xlabel('time (sec)');
ylabel('T (C)');
t1=fscanf(pserial,'%f');
state = fscanf(pserial,'%d');
hold on
ylim([0 70]);
xlim([0 30]);

drawnow
fileID = fopen('data.csv','a');
sec = 0;
fprintf(fileID,'%s;%s;%.2f;%d;%d\n',datestr(datetime('now'),'dd.mm.yyyy'),dates
tr(datetime('now'),'HH:MM:ss'),t1, sec,state);
while sec <= 3*60*5
    pause(10);
    sec = sec + 10;
    if sec<100
        xlim([0 sec+20]);
    else
       xlim([sec-70 sec+20]);
    end
    t=fscanf(pserial,'%f');
    state = fscanf(pserial,'%d');

fprintf(fileID,'%s;%s;%.2f;%d;%d\n',datestr(datetime('now'),'dd.mm.yyyy'),dates
tr(datetime('now'),'HH:MM:ss'),t,sec, state);
    hold on
    plot([sec-10 sec],[t1 t],'color','blue','LineWidth', 1);
    t1 = t;
    drawnow
end
fclose(fileID);
fclose(pserial);
delete(pserial);
```

data.csv

	A	B	C	D	E
1	Date	Time	Value	sec	State
2	03.12.2021	9:33:32	12.5	0	24
3	03.12.2021	9:33:43	12.94	10	24
4	03.12.2021	9:33:54	13.5	20	24
5	03.12.2021	9:34:05	13.94	30	24
6	03.12.2021	9:34:16	14.13	40	24
7	03.12.2021	9:34:26	14.69	50	24
8	03.12.2021	9:34:37	15.44	60	24
9	03.12.2021	9:34:48	16	70	24
10	03.12.2021	9:34:59	16.56	80	24
11	03.12.2021	9:35:10	16.81	90	24
12	03.12.2021	9:35:20	17.12	100	24
13	03.12.2021	9:35:31	17.44	110	24
14	03.12.2021	9:35:42	17.62	120	24
15	03.12.2021	9:35:53	17.94	130	24
16	03.12.2021	9:36:04	18.31	140	24
17	03.12.2021	9:36:14	18.5	150	24
18	03.12.2021	9:36:25	18.81	160	24

Приложение 5. Учебный материал

Пример 1. Решение дифференциальных уравнений с использованием преобразований Лапласа

Пример. Динамика технологического процесса описывается дифференциальным уравнением

$$\frac{d^2y}{dt^2}+5\frac{dy}{dt}+6y=2\frac{dx}{dt}+12x$$

Входной сигнал имеет форму единичного ступенчатого воздействия, т.е. x(t) = 1. Тогда изображение входного сигнала имеет вид X(s) = 1/s.

$$s^2\cdot Y(s)+5\cdot s\cdot Y(s)+6\cdot Y(s)=2\cdot s\cdot X(s)+12\cdot X(s),$$

$$Y(s)\cdot(s^3+5s^2+6s)=2\cdot s+12$$

```
syms  t s x y y(t) x(t) Y(s) X(s)
dY = diff(y,t);
d2Y = diff(dY,t); %diff(dY,t,t)
dX=diff(x);
% Diff. Equation formulation:
equation=d2Y+5*dY+6*y==2*dX+12*x;
condition1=x(t)==1;
condition2=X(s)==1/x;
% Laplace transform of the Diff. Equation
laplace_equation=laplace(equation,t,s);

laplace_equation=subs(laplace_equation,{laplace(y(t),t,s),subs(diff(y(t), t), t, 0), y(0)},{Y(s), 0, 0});
laplace_equation=subs(laplace_equation,{laplace(x(t),t,s),subs(diff(x(t), t), t, 0), x(0)},{X(s), 0, 0});
laplace_equation=subs(laplace_equation,{X(s)},{1/x});
laplace_equation=subs(laplace_equation,{x(t)},{s})
```

```
laplace_equation =
```

$$6\,Y(s)+5\,s\,Y(s)+s^2\,Y(s)=\frac{12}{s}+2$$

```
Ys(s) = simplify(laplace_equation/Y)
```

$$Ys(s) = Y(s)\neq 0\wedge 2\,s+12=s\,Y(s)\ (s^2+5\,s+6)$$

Оригинал полученной функции отсутствует в таблице оригиналов и изображений. Для решения задачи его поиска дробь разбивается на сумму простых дробей.

Декомпозиция дроби

```
b=[2,12];
a=[1,5,6,0];
[r,p,k] = residue(b,a)
```

```
r = 3×1
    2.0000
   -4.0000
    2.0000
p = 3×1
   -3.0000
   -2.0000
         0
k =
     []
```

```
fraction_split=2/(s+3)-4/(s+2)+2/s;
Ys(s)=partfrac(fraction_split)
```

```
Ys(s) =
```

$$\frac{2}{s+3}-\frac{4}{s+2}+\frac{2}{s}$$

Обратное преобразование Лапласа. Оригинал выходной функции

```
yt=ilaplace(Ys)
```

```
yt = 2 e^(-3 t) - 4 e^(-2 t) + 2
```

График

```
t_min=0;
t_max=3;

fplot(yt,[t_min,t_max],'r','LineWidth', 1, 'LineStyle', '-', 'Marker', '*','MarkerSize',5)
xlabel('t')
ylabel('y(t)')
title(string(yt))
grid on
grid minor
```