NTNU
Kunnskap for en bedre verden

KANDIDAT

Vassnes Geir Arne (10021)

PRØVE

TKP4900 1 Kjemisk prosessteknologi, masteroppgave

Emnekode	TKP4900
Vurderingsform	Oppgave
Starttid	01.05.2024 10:00
Sluttid	14.07.2024 23:59
Sensurfrist	15.08.2024 23:59
PDF opprettet	23.07.2024 14:19
Opprettet av	Anacleta Venturin Andersen

Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Det humanistiske fakultet
Institutt for historiske studier

Student Studentsen

Min oppgave

Masteroppgave i Mitt studieprogram
Veileder: Veileder Veiledersen
Januar 2020

NTNU
Kunnskap for en bedre verden

i

Levering av masteroppgave ved NTNU
Gratulerer! Du skal nå levere din masteroppgave.

Forarbeid - lage PDF med NTNU mal
Masteroppgaven skal leveres som en PDF-fil. Før du leverer, anbefaler vi at PDF-en genereres via NTNU Grafisk senter (lenke) eller Skipnes kommunikasjon (lenke) sine løsninger. Dette gjøres for at oppgaven din får et NTNU-uttrykk. I tillegg blir bildene i oppgaven kvalitetssjekket, slik at de er gode nok dersom oppgaven skal trykkes. Utfyllende veiledning finner du ved å følge denne lenken.

Når du nå leverer oppgaven din må du også:

- Ta et valg om du vil publisere oppgaven din i NTNU Open, som er NTNUs institusjonsarkiv
- Bestemme om eventuelle vedlegg skal publiseres i NTNU Open
- Legge inn nødvendig metadata, som brukes i NTNU Open og på vitnemålet ditt

Utfyllende informasjon, samt brukerveiledninger, finner du ved å følge denne lenken.

- [x] **Vi anbefaler at du kopierer tittelen fra en word-fil, eller skrive det direkte inn i ruta**.
Kopiering fra PDF fører ofte til en kodefeil i NTNU Open, derfor anbefaler vi at du kopierer fra en word-fil, eller skriver direkte inn i feltene her.

Tittel på oppgaven din
Fyll inn tittelen på oppgaven din. Tittelen skal være på språket oppgaven er skrevet på. Denne tittelen vil automatisk legges på vitnemålet ditt, vil ligge i metadata i NTNU Open, og vil være søkbar der.

Tuning of cascade control systems

Ord: 5

Er oppgaven din skrevet på et annet språk enn engelsk, legg inn engelsk tittel
Den engelske tittelen vil legges inn i det engelske tillegget til vitnemålet ditt, samt i NTNU Open. Du trenger ikke å skrive noe i dette feltet hvis du har skrevet masteroppgaven din på engelsk.

Ord: 0

Språket masteroppgaven din er skrevet på

- ○ Norsk - bokmål
- ○ Norsk - nynorsk
- ● Engelsk
- ○ Svensk
- ○ Dansk
- ○ Ingen av disse

Besvart.

Sammendrag

Et sammendrag av oppgaven skal skrives både på norsk, eller et annet skandinavisk språk, og engelsk (ref. Studieforskriften §5-8). Disse sammendragene skal være like de sammendragene som står i PDF-versjonen av masteroppgaven din.

Vi anbefaler at du kopierer sammendraget fra en word-fil, eller skrive det direkte inn i ruta. Kopiering fra PDF fører ofte til en kodefeil i NTNU Open, derfor anbefaler vi at du kopierer fra en word-fil, eller skriver direkte inn i feltene her.

Sammendraget vil være tilgjengelig og søkbart i NTNU Open (NTNUs institusjonsarkiv) **uavhengig om du samtykker til publisering eller ikke.**

Dersom du ber om publisering av oppgaven på et senere tidspunkt (båndlegging), vil sammendraget ikke publiseres før båndleggingsdato er passert. Istedenfor sammendrag, vil følgende tekst vises: *"Abstract will be available on YYYY-mm-dd".*

Sammendrag på norsk, eller et annet skandinavisk språk

Denne oppgåva tek for seg utfordringa med å optimalisera tidsskalaseparasjon i kaskadekontrollsystem, ein mykje brukt industriell kontrollstrategi. Trass i utbreiinga er det framleis mangel på enkle reglar for val av tidskonstant separasjon mellom primære og sekundære sløyfer. Oppgåva tek sikte på å utleia enkle og pålitelege reglar for innstilling av primærkontrollerens tidskonstant (τ_{c1}) ved å optimalisera ein kostnadsfunksjon designet for å halda oppe robustheten til SIMC-PID-innstillingsreglane. Studien evaluerer ulike metodar og historiske tilnærmingar til kaskadekontroll, inkludert arbeidet til McMillan, Huang et al., Padhan og Majhi, og Çakıroğlu et al., og framhevar deira bidrag og avgrensingar. Gjennom MATLAB-simuleringar blir ytinga analysert til ulike innstillingsstrategiar ved å bruka Integrated Absolute Error (IAE) og Total Variation (TV) som nøkkeltal. Resultata indikerer at medan ei nedre grense for tidsskalaseparasjon effektivt kan setjast til 5, krev etablering av ei øvre grense ytterlegare forsking. Den foreslåtte kostnadsfunksjonen kan gi eit balansert mål på yting og robusthet, og antydar potensial for praktisk bruk og framtidig foredling.

Ord: 159

Sammendrag på engelsk

This thesis addresses the challenge of optimizing time scale separation in cascade control systems, a widely used industrial control strategy. Despite its prevalence, there remains a lack of straightforward rules for selecting the time constant separation between primary and secondary loops. The thesis aims to derive simple and reliable rules for tuning the primary controller's time constant (τ_{c1}) by optimizing a cost function designed to maintain the robustness of the SIMC-PID tuning rules. The study evaluates various methods and historical approaches to cascade control, including the work of McMillan, Huang et al., Padhan and Majhi, and Çakıroğlu et al., highlighting their contributions and limitations. Through MATLAB simulations, the performance of different tuning strategies is analyzed using Integrated Absolute Error (IAE) and Total Variation (TV) as key metrics. The results indicate that while a lower bound for time scale separation can be effectively set at 5, establishing an upper bound requires further research. The proposed cost function may provide a balanced measure of performance and robustness, suggesting potential for practical application and future refinement.

Ord: 173

Besvart.

EKSEMPEL

Oppgi navn på forfatter, eller forfattere, av oppgaven: Ditt navn. Dersom du har skrevet oppgaven sammen med andre, skriv inn alle navnene. Informasjonen du legger inn her bli synlig, og søkbar, i NTNU Open. Dette er nyttig, fordi du kan finne igjen oppgaven din. Du kan da vise den frem til framtidige arbeidsgivere, og andre interesserte.

Forfatter(e)s navn skrives på denne formen: Etternavn, Fornavn. Er det flere forfattere, skal disse separeres med linjeskift

Normann, Kari
Doe, Jane Elizabeth
Medel-Svensson, Filip

Oppgi veileders navn. Har du flere veiledere, skal også disse oppgis.

Navn på veileder(e) skrives på formen: Etternavn, Fornavn. Dersom det er flere veiledere, skal disse separeres med linjeskift.

Nilssen, Otto
Modal, Lise Müller

☑ **Oppgi navn på forfatter, eller forfattere, av oppgaven:** Ditt navn. Dersom du har skrevet oppgaven sammen med andre, skriv inn alle navnene. Informasjonen du legger inn her bli synlig, og søkbar, i NTNU Open. Dette er nyttig, fordi du kan finne igjen oppgaven din. Du kan da vise den frem til framtidige arbeidsgivere, og andre interesserte.

Forfatter(e)s navn skrives på denne formen: Etternavn, Fornavn. Er det flere forfattere, skal disse separeres med linjeskift

Vassnes, Geir Arne

Ord: 3

Oppgi veileders navn. Har du flere veiledere, skal også disse oppgis.

Navn på veileder(e) skrives på formen: Etternavn, Fornavn. Dersom det er flere veiledere, skal disse separeres med linjeskift.

Skogestad, Sigurd

Ord: 2

Besvart.

1 Levere masteroppgaven:

Din fil ble lastet opp og lagret i besvarelsen din.

Last ned Fjern Erstatt

Filnavn:	thesis (1).pdf
Filtype:	application/pdf
Filstørrelse:	7.21 MB
Opplastingstidspunkt:	14.07.2024 23:41
Status:	**Lagret**

Besvart.

2 Lever eventuelle vedlegg:

Din fil ble lastet opp og lagret i besvarelsen din.

Last ned Fjern Erstatt

Filnavn:	Files.zip
Filtype:	application/x-zip-compressed
Filstørrelse:	36.63 KB
Opplastingstidspunkt:	14.07.2024 22:37
Status:	**Lagret**

Er du usikker på hvordan du lager en ZIP-fil? Følg denne lenka til veiledning på Innsida.

Besvart.

3 KI-deklarasjon

Vennligst fyll ut KI-deklarasjonen og last opp det ferdig utfylte og signerte dokumentet her.

Din fil ble lastet opp og lagret i besvarelsen din.

Last ned | Fjern | Erstatt

Filnavn:	207214765-Deklarasjon om KI_no_nn_skjema_207221692_1708091642582.pdf
Filtype:	application/pdf
Filstørrelse:	484.37 KB
Opplastingstidspunkt:	14.07.2024 22:42
Status:	**Lagret**

Besvart.

Publisering av masteroppgaver

Når du leverer inn masteroppgaven din må du ta stilling til om oppgaven skal gjøres åpent tilgjengelig, ha utsatt publiseringstidspunkt eller holdes lukket.

NTNU oppfordrer til publisering av alle oppgaver, men valget er opp til deg/dere som forfatter(e).

Åpen publisering

Velger du å gjøre oppgaven din tilgjengelig for allmennheten, kreves godkjenning i henhold til gjeldende publiseringsavtale (versjon 1.0). Oppgaven vil først bli publisert etter sensuren er endelig og alle oppgaver publiseres i **NTNU Open (lenke)**.

Du finner gjeldende publiseringsavtale, samt mer informasjon **ved å følge denne lenken til brukerveiledning om publisering og båndlegging av oppgaver på Innsida**

Reservasjon mot publisering/utsatt publisering

Dersom du ikke ønsker å publisere oppgaven, vil kun metadata (tittel, forfatter, veileder) og sammendrag bli gjort tilgjengelig. Ved avtale om utsatt publisering (båndlegging) vil verken metadata, sammendrag eller oppgaven være tilgjengelig før den angitte dato i avtalen.

Dersom du ikke har levert masteravtale med informasjon om utsatt publisering, men ønsker dette likevel, tar du kontakt med instituttet ditt etter at du har levert.

Masteroppgave med flere forfattere

Dersom oppgaven som leveres har flere forfattere, så gjelder publiseringsavtalen for samtlige forfattere. Alle forfatterne MÅ samtykke til publisering. Dersom en av forfatterne i gruppa ikke ønsker publisering, skal svaret på hvorvidt oppgaven skal publiseres være "Nei".

Tillater du publisering av din oppgave?

- ○ Ja, jeg tillater publisering av masteroppgaven
- ◉ Nei, jeg tillater ikke publisering av masteroppgaven

Publisering av vedlegg

Dersom du leverer vedlegg som en del av masteroppgaven din, kan du velge om du også ønsker å gjøre disse tilgjengelig.

Tillater du publisering av vedlegg?

- ○ Ja, jeg tillater publisering av vedleggene
- ◉ Nei, jeg tillater ikke publisering av vedleggene
- ○ Jeg har ikke levert vedlegg

(Utelater du å hake av for ønsket valg, så vil vedleggene følge valget du har tatt om publisering av oppgaven.)

Besvart.

Geir Arne Vassnes

Tuning of cascade control systems

Master's thesis in Chemical Engineering and Biotechnology
Supervisor: Sigurd Skogestad
Co-supervisor: Krister Forsman
July 2024

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemical Engineering

NTNU
Norwegian University of
Science and Technology

Geir Arne Vassnes

Tuning of cascade control systems

Master's thesis in Chemical Engineering and Biotechnology
Supervisor: Sigurd Skogestad
Co-supervisor: Krister Forsman
July 2024

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemical Engineering

NTNU
Norwegian University of
Science and Technology

Abstract

This thesis addresses the challenge of optimizing time scale separation in cascade control systems, a widely used industrial control strategy. Despite its prevalence, there remains a lack of straightforward rules for selecting the time constant separation between primary and secondary loops. The thesis aims to derive simple and reliable rules for tuning the primary controller's time constant (τ_{c1}) by optimizing a cost function designed to maintain the robustness of the SIMC-PID tuning rules. The study evaluates various methods and historical approaches to cascade control, including the work of McMillan, Huang et al., Padhan and Majhi, and Çakıroğlu et al., highlighting their contributions and limitations. Through MATLAB simulations, the performance of different tuning strategies is analyzed using Integrated Absolute Error (IAE) and Total Variation (TV) as key metrics. The results indicate that while a lower bound for time scale separation can be effectively set at 5, establishing an upper bound requires further research. The proposed cost function may provide a balanced measure of performance and robustness, suggesting potential for practical application and future refinement.

Samandrag

Denne oppgåva tek for seg utfordringa med å optimalisera tidsskalaseparasjon i kaskadekontrollsystem, ein mykje brukt industriell kontrollstrategi. Trass i utbreiinga er det framleis mangel på enkle reglar for val av tidskonstant separasjon mellom primære og sekundære sløyfer. Oppgåva tek sikte på å utleia enkle og pålitelege reglar for innstilling av primærkontrollerens tidskonstant (τ_{c1}) ved å optimalisera ein kostnadsfunksjon designet for å halda oppe robustheten til SIMC-PID-innstillingsreglane. Studien evaluerer ulike metodar og historiske tilnærmingar til kaskadekontroll, inkludert arbeidet til McMillan, Huang et al., Padhan og Majhi, og Çakıroğlu et al., og framhevar deira bidrag og avgrensingar. Gjennom MATLAB-simuleringar blir ytinga analysert til ulike innstillingsstrategiar ved å bruka Integrated Absolute Error (IAE) og Total Variation (TV) som nøkkeltal. Resultata indikerer at medan ei nedre grense for tidsskalaseparasjon effektivt kan setjast til 5, krev etablering av ei øvre grense ytterlegare forsking. Den foreslåtte kostnadsfunksjonen kan gi eit balansert mål på yting og robusthet, og antydar potensial for praktisk bruk og framtidig foredling.

Preface

I would like to thank Professor Sigurd Skogestad for the invaluable help you have contributed to this thesis. Thank you Krister Forsman for the help you provided during this semester.

Table of Contents

1 Introduction

1.1 Motivation

Cascade control is widely used in industry, yet the design methods are not well developed. In particular, it is not clear what the time scale separation between the loops should be. Current knowledge suggests that it should be between 4 and 10, but there are no simple rules for an exact value based on the system.[1]

1.2 Earlier work

Gregory K. McMillan's 1982 study on cascade control highlighted the importance of accurately tuning the outer loop's time constant (τ_c) to improve control loop performance. McMillan developed equations to estimate the ultimate period of the outer loop, which involves adding 90

However, McMillan also noted the necessity for significant detuning of the outer controller's proportional band (PB_o) to prevent instability, especially in integrating and runaway processes. The detuning is quantitatively substantial, as indicated by the corrected outer proportional band equation:

$$PB'_o = PB_o \left(A\frac{T_{Ci}}{T_{Co}} + B\frac{T_{Di}}{T_{Do}} + C \right)$$

where $A = 0.4$ for self-regulating processes and 1.0 for integrating/runaway processes, $B = 0.4$ and 0.0 respectively, and C varies accordingly[2].

Additionally, McMillan pointed out that issues related to control valve rangeability and measurement accuracy were not fully addressed by his method. These limitations suggest the need for further refinements to optimize industrial application, emphasizing the importance of advancements in control valve dynamics and measurement accuracy[2].

In 1998, Huang et al. introduced a streamlined approach for tuning the time constant (τ_c) of the primary controller in cascade control systems. Their method uses a combination of relay feedback tests and first-order plus dead-time (FOPDT) models, simplifying the tuning process with practical equations that link tuning parameters directly to system performance. This technique allows for quick adjustments to τ_c, enhancing the controller's response to disturbances and setpoint changes[3].

Despite its advantages, Huang et al.'s approach faces challenges, particularly the assumption that process dynamics can be accurately modeled by FOPDT models. In real-world applications, this assumption may not hold, leading to potential inaccuracies in tuning. Additionally, the method relies on the accuracy of relay feedback tests, which can be compromised by noise and other disturbances[3].

Huang et al. acknowledge these limitations, proposing that while their method improves efficiency and simplifies tuning, it should be applied with consideration of specific process characteristics and operational conditions. They suggest further research to adapt the method for more complex process dynamics, noting that their simulations demonstrate performance comparable to traditional approaches[3].

In 2013, Padhan and Majhi developed an outer loop controller tuning method for cascade control systems, focusing on integrating processes with significant time delays. They designed the primary controller (G_{c3}) using a combination of PID elements and a lead/lag compensator, aiming to improve response to disturbances while maintaining robust performance. Using the robust Internal Model Control (IMC) methodology, G_{c3} was formulated and approximated using a (1, 2) Pade approximation to handle the time delay $e^{-\tau_m s}$[4].

This design is notable for decoupling setpoint tracking from disturbance rejection, optimizing response times and stability under varying conditions. However, challenges include the complexity of accurately modeling process dynamics and sensitivity to parameter estimations, which can degrade performance if the model does not reflect true dynamics or if there is significant variability [4].

Robustness analysis using Kharitonov's theorem confirmed the theoretical stability of their control strategy over a range of parameter uncertainties. Practical implementation highlighted difficulties in achieving desired performance due to unmodeled dynamics and external disturbances, suggesting a need for ongoing adjustments and potential enhancements [4].

In 2014, Çakıroğlu et al. presented a significant development in tuning τ_c for the primary controller. Their methodology simplifies tuning by decomposing the outer-loop process into first-order transfer functions, each controlled by a tailored controller (proportional for integrator forms, proportional-integral for real stable poles). This approach facilitates precise control and efficient tuning of τ_c, addressing each process segment individually [5].

However, their approach faces challenges, particularly in system design complexity and computational load during setup. The method's effectiveness relies heavily on accurately tuning Smith predictors to compensate for time delays. Inaccurate tuning or mismatched models can affect control performance, stability, and robustness [5].

Despite these limitations, their method is validated through simulations showing superior performance in reference tracking, disturbance rejection, and robustness against parameter variations. By simplifying complex high-order systems into manageable components, their approach reduces the number of tuning parameters, offering a robust framework for advancing cascade control systems [5].

Cascade control tuning has advanced significantly but faces ongoing challenges. McMillan established key principles for tuning the outer loop's time constant and proportional band detuning. Huang et al. simplified tuning with relay feedback tests and FOPDT models, despite real-world limitations. Padhan and Majhi's IMC-based method improved disturbance response for integrating processes with time delays, though it was sensitive to parameter estimations. Çakıroğlu et al.'s decomposition approach enhanced tuning efficiency and performance but required accurate Smith predictor tuning. These studies emphasize the need for precise tuning, accurate modeling, and adaptability to varying process dynamics, highlighting the necessity for ongoing research in cascade control systems for modern industrial applications.

1.3 Aim of this work

The aim of this thesis is to derive simple rules for selecting the time scale separation for cascade control systems. This will be achieved by optimizing a cost function designed to maintain the same robustness as the SIMC-PID tuning rules. The ultimate goal is to develop a straightforward and reliable rule for tuning the primary controller time constant, τ_{c1}, in systems utilizing cascade control.

2 Theory

2.1 Cascade Control

Cascade control is a control system in which one controller is used to manipulate the set point of another. The first controller in the loop, called the primary controller, has an independent set point. The second controller in the loop, called the secondary controller, is given a set point from the output of the primary controller; only this controller has an output to the process. The manipulated value, the secondary controller, and its measurement constitute a closed loop within the primary loop. Figure 1 shows a possible cascade configuration. [6]

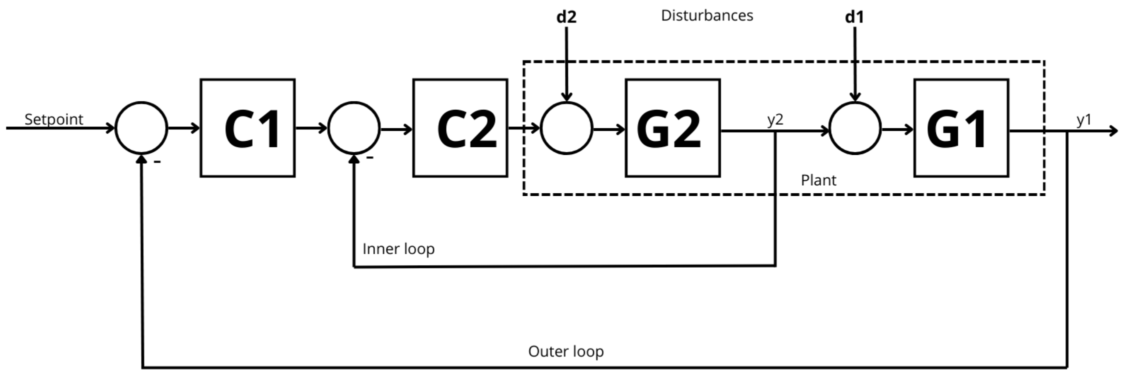

Figure 1: Block diagram of a cascade control system

The principal advantages of cascade control are the following: [6]

1. Disturbances within the secondary loop (d2) are corrected by the secondary controller (C2) before they can influence the primary variable (y1).
2. Phase lag in the secondary part of the process (G2) is greatly reduced.
3. Gain variations in the secondary part of the process (G2) are overcome in the secondary loop.
4. The secondary loop permits an exact manipulation of the flow of mass or energy (y2) by the primary controller.

The tuning of the controllers, C1 and C2, in a cascaded system should be done sequentially. It is recommended to use a design method where closed-loop time constants τ_{c1} and τ_{c2} are used as tuning parameters. The secondary controller, C2, is tuned first based on the inner process model, G2, and with this loop closed, the primary controller, C1, is tuned. [1]

2.1.1 Time scale separation

When using cascade control, the closed-loop time constant τ_{c1} of the primary controller should be larger than the closed-loop time constant τ_{c2} of the secondary controller. The ratio between these values is known as time scale separation, as shown in Equation 1. [1].

$$\text{Time scale separation} = \frac{\tau_{c1}}{\tau_{c2}} \tag{1}$$

A large time scale separation is desired to allow for the independent design of each loop and to avoid potential undesired interactions between the loops. If the time scale separation is too small, typically 3 or less, the loops start interacting and resonance can occur, which degrades performance. Conversely, a large time scale separation provides robustness against process gain variations in both loops.[1]

For instance, a process gain decrease in the inner loop results in a higher value of τ_{c2}, thus reducing the time scale separation and, consequently, robustness. To ensure robustness against such gain variations, it is often recommended to have a time scale separation of 10 or greater. However, a very large time scale separation can also be problematic as it consumes more of the available time window, especially with multiple layers of cascade control.[1]

A rule of thumb is to maintain a time scale separation between 4 and 10.[1]

2.2 PID control

Proportional-Integral-Derivative (PID) control combines proportional, integral, and derivative actions. Together with Proportional-Integral (PI) control, it has been the dominant control technique for many decades. There are several variations of PID control, one of these is the parallel form, which is shown in Equation 2.[7]

$$p(t) = \bar{p} + K_c \left(e(t) + \frac{1}{\tau_I} \int_0^t e(t*)dt * + \tau_D \frac{de(t)}{dt} \right) \tag{2}$$

Where,

$$\begin{aligned} p(t) &= \text{controller output} \\ \bar{p} &= \text{bias (steady-state) value} \\ K_c &= \text{controller gain} \\ e(t) &= \text{error signal} \\ \tau_I &= \text{integral time} \\ \tau_D &= \text{derivative time} \end{aligned}$$

The corresponding transfer function is shown in Equation 3.

$$C(s) = K_c \left(1 + \frac{1}{\tau_I s} + \tau_D s \right) \tag{3}$$

The PID controller has three tuning parameters, K_c, τ_I and τ_D. Derivative action is commonly not used to avoid derivative "kick", resulting in a PI controller with two tuning parameters. For tuning these parameters, it is recommended to base the tuning on a first-order plus delay model of the system, often obtained from an experiment that excites the process.[1]

2.2.1 Half Rule

For tuning higher-order transfer function models, it is useful to approximate these as lower-order models. Low-order models are more convenient for control system design and analysis.[7]

The half rule simplifies the approximation of higher-order dynamic processes by distributing the largest neglected time constant between the effective delay and the smallest retained time constant. Let the original model be as shown in Equation 4.[8]

$$G_0(s) = \frac{\Pi_j (-T_{j0}^{inv} s + 1)}{\Pi_i (\tau_{i0} s + 1)} e^{-\theta_0 s} \tag{4}$$

The effective delay, θ, is calculated by modifying the original delay, θ_0, and considering the contributions of the neglected time constants, τ_{i0}, and inverse response time constants, T_{j0}^{inv}. For digital implementations, the sampling period, h, also contributes approximately $\frac{h}{2}$ to the effective delay. For a first-order model, the effective delay θ and the dominant time constant τ_1 are calculated as shown in Equations 5 and 6. [8]

$$\tau_1 = \tau_{10} + \frac{\tau_{20}}{2} \tag{5}$$

$$\theta = \theta_0 + \frac{\tau_{20}}{2} + \sum_{i \geq 3} \tau_{i0} + \sum_j T_{j0}^{inv} + \frac{h}{2} \tag{6}$$

2.2.2 SIMC-PID tuning rules

To tune PID controllers, it is useful to have a systematic procedure to find good settings for the three tuning parameters. Several objectives guide the design of such procedures: they should be well-motivated, preferably model-based and analytically derived, simple and easy to memorize, and effective across a wide range of processes. [9]

The simple two-step SIMC procedure meets these objectives. The first step involves obtaining a first- or second-order plus delay model of the process. The second step derives model-based controller settings: a first-order model provides PI settings, while a second-order model offers PID settings. [9]

Consider a generic second-order transfer function with time delay, as shown in Equation 7, where k is the process gain, τ_1 and τ_2 are the lag time constants and θ represents the time delay. [9]

$$G(s) = \frac{k}{(\tau_1 s + 1)(\tau_2 s + 1)} e^{-\theta s} \tag{7}$$

The PID-settings for a second-order model when using SIMC are calculated as shown in Equations 8, 9 and 10, where K_c is the controller gain, τ_I is the integral time and τ_D is the derivative time. [9]

$$K_c = \frac{1}{k} \frac{\tau_1}{\tau_c + \theta} = \frac{1}{k'} \frac{1}{\tau_c + \theta} \tag{8}$$

$$\tau_I = min(\tau_1, 4(\tau_c + \theta)) \tag{9}$$

$$\tau_D = \tau_2 \tag{10}$$

The parameter τ_c, introduced in Equations 8 and 9, is the desired closed-loop time constant and is the sole tuning parameter for the controller. For tight control this parameter should be tuned as shown in Equation 11. [9]

$$\tau_c = \theta \tag{11}$$

2.2.2.1 Robustness

Robustness margins, including gain margin (GM) and phase margin (PM), are critical measures of a control system's ability to maintain performance despite system uncertainties and variations. Higher margins typically indicate better robustness, ensuring the system can tolerate greater variations in system parameters without compromising stability. Sensitivity peaks (M_s) and complementary sensitivity peaks (M_t) serve as indicators of the system's capacity to reject disturbances and accurately follow reference signals, respectively. Lower values of M_s and M_t suggest improved performance in rejecting disturbances and tracking desired outputs, highlighting the system's resilience to external and internal changes.[8]

The SIMC (Simple Internal Model Control) settings are tailored to achieve a fast response while maintaining robust performance. These settings provide robustness margins that significantly exceed the typical minimum requirements of $GM > 1.7$ and $PM > 30°$. For first-order processes, the robustness margins are $GM = 3.14$ and $PM = 61.4°$, demonstrating strong robustness. For integrating processes, the margins are slightly lower but still robust, with $GM = 2.96$ and $PM = 46.9°$. These values underscore the effectiveness of the SIMC settings in maintaining stability and performance across different types of processes.[8]

Sensitivity peaks (M_s) and complementary sensitivity peaks (M_t) are crucial for evaluating control system performance. Ideally, these values should be less than 2, as lower values indicate better performance. For first-order processes, the sensitivity peak is $M_s = 1.59$ and the complementary sensitivity peak is $M_t = 1.00$. In the case of integrating processes, the sensitivity peak is $M_s = 1.70$ and the complementary sensitivity peak is $M_t = 1.30$. These values reflect the system's ability to handle disturbances and track setpoints effectively, confirming the robustness and reliability of the SIMC settings.[8]

3 Method

3.1 Plant

As a basis for the simulations performed in MATLAB, a plant was modeled, as shown in Figure 1. where the primary process could be seen as a tank level, while the secondary process was based on a linear valve. The Simulink model for the plant is shown in Appendix B.

The transfer function based on the tank is given in Equation 12 .

$$G_1(s) = k_1 \frac{e^{-\theta_1 s}}{s} \tag{12}$$

With k_1 being the process gain and θ_1 being the process delay.

The transfer function based on the linear valve is given in Equation 13.

$$G_2(s) = k_2 \frac{e^{-\theta_2 s}}{\tau_2 s + 1} \tag{13}$$

With k_2 being the process gain, θ_2 being the process delay and τ_2 being the time constant.

For simulations with a non-integral transfer function for the primary process, a transfer function based on a linear valve was used as shown in Equation 14.

$$G_1(s) = k_1 \frac{e^{-\theta_1 s}}{\tau_1 s + 1} \tag{14}$$

3.1.1 Tuning

The controllers were tuned using the SIMC rules as shown in 2.2.2. The secondary controller, C2, was tuned first with respect to G_2. The desired closed-loop time constant for the inner loop was tuned for tight control with $\tau_{c2} = \theta_2$.

The primary controller, C1, was tuned with regard to the closed-loop transfer function of the inner loop, combined with G_1. Using the half-rule, a new transfer function G_1' is derived, and the controller is tuned with respect to this transfer function. The derivation of G_1' is shown in Equation 15.

$$\begin{aligned} T_2 = \frac{C_2 G_2}{1 + C_2 G_2} \approx \frac{e^{-\theta_2 s}}{\tau_{c_2} s + 1} \\ G_1' = T_2 G_1 = k_1 \frac{e^{-\theta_1 s}}{s} \frac{e^{-\theta_2 s}}{\tau_{c_2} + 1} \\ G_1' = k_1 \frac{e^{-(\theta_1 + \theta_2) s}}{s(\tau_{c_2} s + 1)} \approx k_1 \frac{e^{-(\theta_1 + \theta_2 + \frac{\tau_{c2}}{2}) s}}{s} \end{aligned} \tag{15}$$

For simulations with a non-integral transfer function, the derivation of G_1' is shown in Equation 16

$$
\begin{aligned}
T_2 = \frac{C_2 G_2}{1 + C_2 G_2} &\approx \frac{e^{-\theta_2 s}}{\tau_{c_2} s + 1} \\
G_1' = T_2 G_1 = k_1 &\frac{e^{-\theta_1 s}}{\tau_1 s + 1} \frac{e^{-\theta_2 s}}{\tau_{c_2} + 1} \\
G_1' = k_1 \frac{e^{-(\theta_1 + \theta_2) s}}{(\tau_1 s + 1)(\tau_{c_2} s + 1)} &\approx k_1 \frac{e^{-(\theta_1 + \theta_2 + \frac{\tau_{c2}}{2}) s}}{(\tau_1 + \frac{\tau_{c2}}{2}) s + 1}
\end{aligned}
\tag{16}
$$

The tuning of the controllers using the SIMC-rules was done in MATLAB as shown in Appendix A.1.

3.1.2 Variance

To simulate variances in the processes, G_1 and G_2, different values were used for the process gain and process delay, the tuning of the controllers were always based on the nominal case. The values with respect to the nominal values chosen are shown in Table 1.

Variable	Values
k_2	$[k_{2,\text{nom}},\ 2 \cdot k_{2,\text{nom}},\ (0.25 \text{ or } 0.1) \cdot k_{2,\text{nom}}]$
θ_2	$[\theta_{2,\text{nom}},\ \max(1.25 \cdot \theta_{2,\text{nom}}, 0.5)]$
k_1	$[k_{1,\text{nom}},\ 2 \cdot k_{1,\text{nom}},\ 0.5 \cdot k_{1,\text{nom}}]$
θ_1	$[\theta_{1,\text{nom}},\ 2 \cdot \theta_{1,\text{nom}}]$

Table 1: Variances of the processes

3.2 Cost Function

To have a basis for optimizing the process, a cost function to measure performance is used.

3.2.1 Integrated absolute error

Integrated Absolute Error (IAE) is a performance measurement that is widely used to evaluate the performance of a control structure. IAE is calculated as shown in Equation 17.

$$
IAE = \int_0^\infty |y(t) - y_s(t)| dt \tag{17}
$$

3.2.2 Total variation

Total variation (TV) is a measure of the quality of the signal response, both output (y) and input (u). TV is defined as the total up and down movement of a signal, which should be minimized to ensure a smooth response. TV for a signal y(t) is defined as shown in Equation 18, where v_i represents the differences in the signal values over time. [10]

$$
TV = \sum_{i=1}^{\infty} |v_i| \tag{18}
$$

3.2.3 Combined IAE and TV for SIMC

IAE is a measurement that generally gives a good evaluation of the performance of a control structure, but it does not punish fast oscillations strong enough. Therefore, a cost function that is based on both IAE and TV is used. The cost function is shown in Equation 19.

$$J_0 = \frac{IAE}{IAE_0} + \alpha \frac{TV}{TV_0} \tag{19}$$

With IAE_0 and TV_0 being the IAE and TV of the nominal case under tight control with $\tau_{c1} = \theta_1 + \theta_2 + \frac{\tau_{c2}}{2}$. These values are calculated using the MATLAB code in Appendix A.2. IAE and TV are the measured values of the simulation, while α is a weight value that decides how much TV should affect the cost function.

For j runs that use the same tuning, the cost function is averaged as shown in Equation 20.

$$J = \frac{\sum_{i=1}^{j} \left(\frac{IAE_i}{IAE_0} + \alpha \frac{TV_i}{TV_0} \right)}{j} \tag{20}$$

3.3 Basis for simulation

To simulate the plant, Simulink and MATLAB were used. To test the response of the controllers, a setpoint change was set in at a time of 1, d1 was set to 1 at a time of 500 and d2 was set to 1 at a time of 1000, the simulation ran until 1500. The setpoint change and disturbance d2 had the same weight, while disturbance d1 had $\frac{1}{30}$ of the weight of the two other changes. This was done since the changes of d2 and setpoint are more important for cascade control systems.

4 Results and discussion

4.1 α-value for cost function

To find the value of α that gave the same robustness as the SIMC rules, simulations were performed using G_1 with an integral response to obtain the target τ_{c1} from the SIMC rules. The nominal values for the basis of these simulations are the following: $k_2 = 1$, $\theta_2 = 0$, $\tau_2 = 1$, $k_1 = 1$ and $\theta_1 = 1$. This was done in MATLAB as shown in Appendix A.3, and the results are shown in Figure 2, created in MATLAB as shown in Appendix A.4.

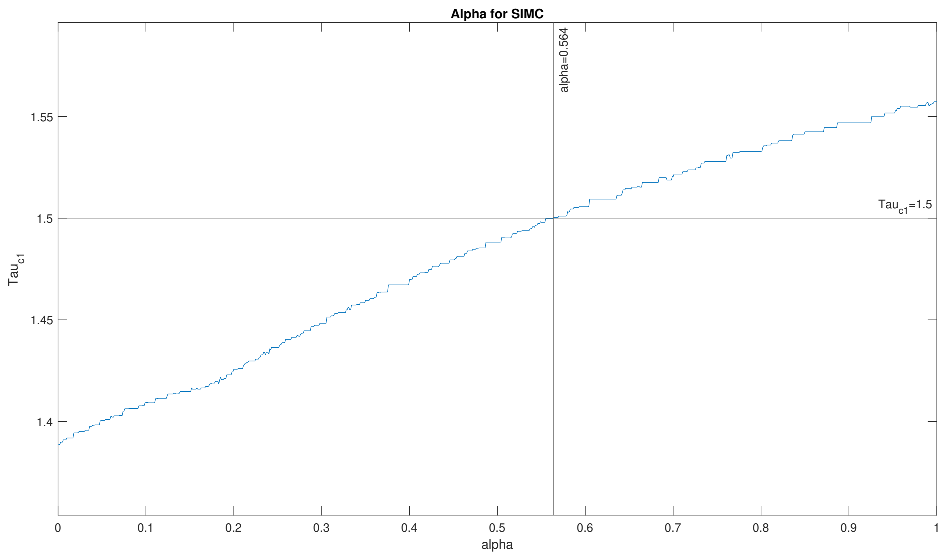

Figure 2: Minimization results for different alpha-values

The code gave a result of $\alpha = 0.564$, which gives the cost function as shown in Equation 21.

$$J = \frac{IAE}{IAE_0} + 0.564\frac{TV}{TV_0} \tag{21}$$

The cost function seems reasonable and seems like it should be a good measure for the performance of the tuning across several variant processes. There is uncertainty if this cost function would hit the SIMC-tuning with other nominal values, as this has not been tested.

4.2 Integral G_1

For the cases mentioned in this section, G_1 with an integral response is used, as shown in Equation 12.

4.2.1 Case 1

For this case, the nominal variables are varied as shown in Table 1, with $0.25k_{2,nom}$ as the lower value of k_2.

The result of the optimization with the nominal case of $k_1 = 1$, $\theta_1 = 1$, $k_2 = 1$, and $\theta_2 = 0$ using the MATLAB code shown in Appendix A.5, is shown in Figure 3. The result of the cost function as shown in Equation 20 was 3.5319, with $\tau_{c1} = 5.479$.

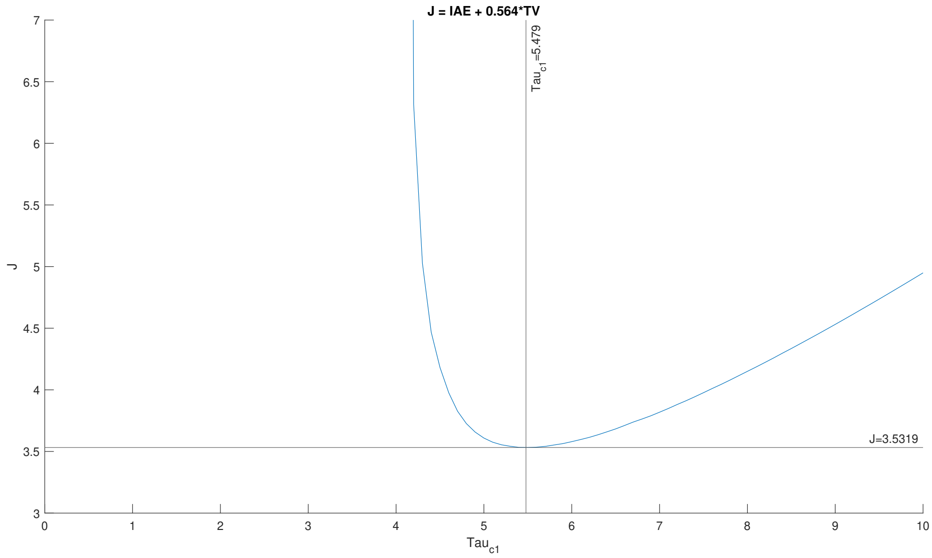

Figure 3: Relation between cost function and τ_{c1} for $K_{2,min} = 0.25K_{2,nom}$

The response on both input and output on the nominal case with $\tau_{c1} = 5.479$ is shown in Figure 4. It is clear that the relationship between y and u is $y = 1 - u$.

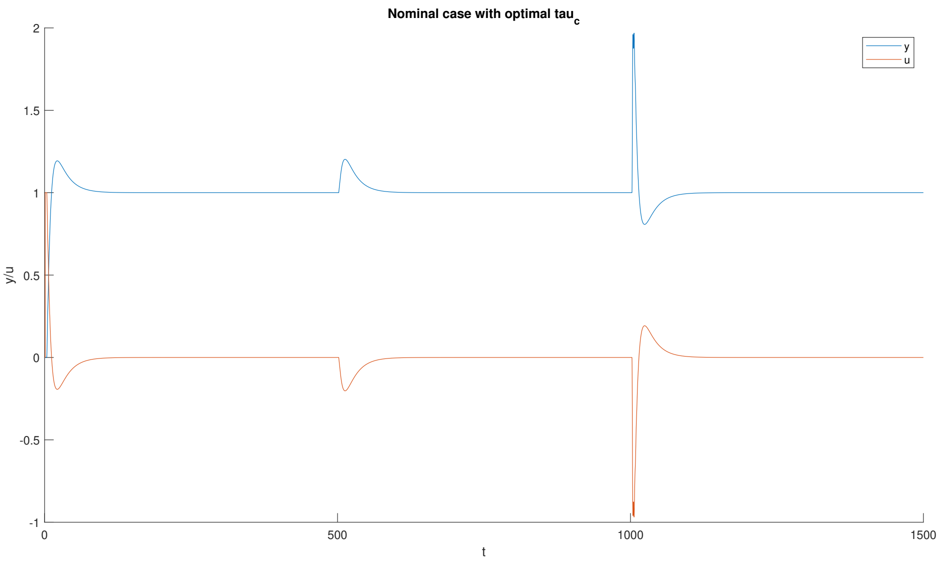

Figure 4: Input and output of the nominal case

The output of all 36 simulated cases is shown in Figure 5.

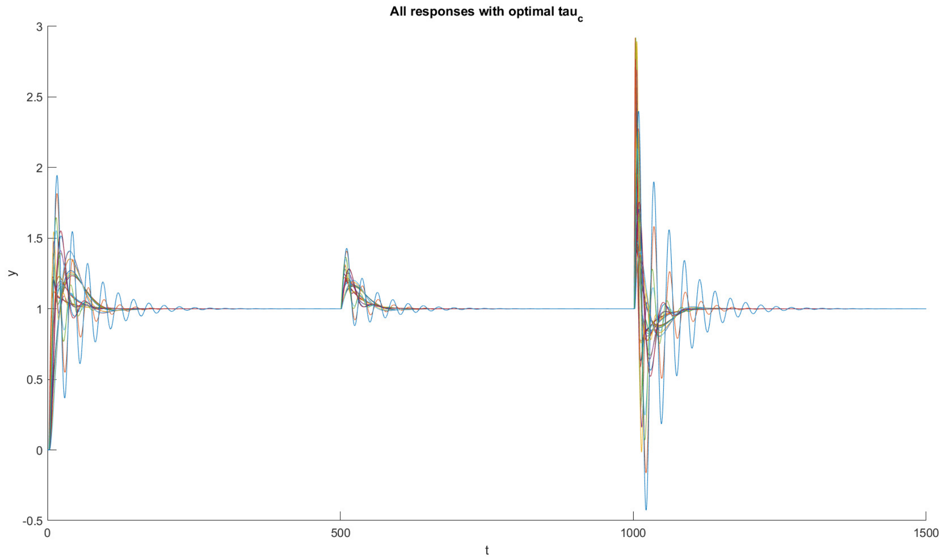

Figure 5: Output of all cases

The output of the worst-performing case and the nominal case is shown in Figure 6.

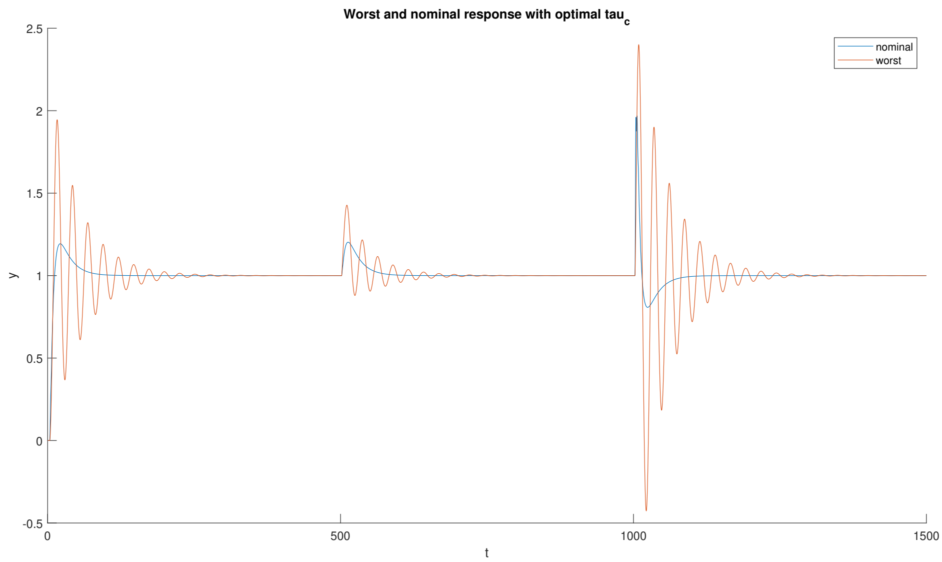

Figure 6: Output of the worst and nominal case

Several other nominal cases were also optimized, the results of these optimizations are shown in Table 2.

Parameter	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7
$K_{1.nom}$	1	2	1	1	1	1	1
$K_{2.nom}$	1	1	2	1	1	1	1
$\theta_{1.nom}$	1	1	1	2	3	3	3
$\theta_{2.nom}$	0	0	0	0	0	1	2
τ_{c1}	5.479	5.455	5.613	8.418	11.172	14.077	18.731
J	3.532	3.664	3.464	3.506	3.536	3.483	4.476
SIMC τ_{c1}	1.5	1.5	1.5	2.5	3.5	4.5	6
τ_{c2}	1	1	1	1	1	1	2
$\frac{\tau_{c1}}{\text{SIMC } \tau_{c1}}$	3.653	3.636	3.742	3.367	3.192	3.128	3.122
$\frac{\tau_{c1}}{\tau_{c2}}$	5.479	5.455	5.613	8.418	11.172	14.077	9.366

Table 2: Results of different nominal cases for $K_{2,min} = 0.25$

As seen in Figure 3, there is a steep increase in the value of the cost function at lower values of τ_{c1}, while the increase is flatter for higher values. It seems to be better to overshoot than undershoot the optimal value. Figure 5 shows that there is a huge difference in the performance of the different cases, Figure 6 shows that the worst case has many oscillations, this could be reduced by increasing α, but this would cause the nominal case to perform worse.

Looking at Table 2, it seems to be an almost linear relationship between τ_{c1} and SIMCτ_{c1}, with all values staying between 3.1 and 3.7.

4.2.2 Case 2

For this case, the nominal variables are varied as shown in Table 1, with $0.1k_{2,nom}$ as the lower value of k_2.

The result of the optimization with the nominal case of $k_1 = 1$, $\theta_1 = 1$, $k_2 = 1$, and $\theta_2 = 0$ using the MATLAB code shown in Appendix A.5, is shown in Figure 7. The result of the cost function as shown in Equation 20 was 5.3419, with $\tau_{c1} = 8.100$.

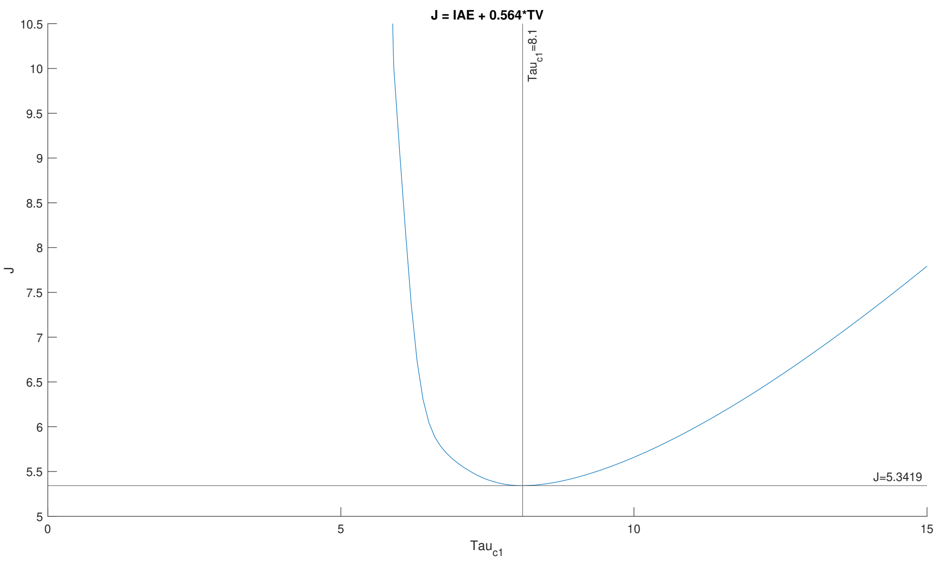

Figure 7: Relation between cost function and τ_{c1} for $K_{2,min} = 0.1K_{2,nom}$

Several other nominal cases were also optimized, the results of these optimizations are shown in Table 3.

Parameter	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7
$K_{1.nom}$	1	2	1	1	1	1	1
$K_{2.nom}$	1	1	2	1	1	1	1
$\theta_{1.nom}$	1	1	1	2	3	3	3
$\theta_{2.nom}$	0	0	0	0	0	1	2
τ_{c1}	8.100	8.008	8.231	11.112	13.922	18.832	18.542
J	5.342	5.566	5.059	4.678	4.475	4.717	6.384
SIMC τ_{c1}	1.5	1.5	1.5	2.5	3.5	4.5	6
τ_{c2}	1	1	1	1	1	1	2
$\frac{\tau_{c1}}{\text{SIMC } \tau_{c1}}$	5.400	5.339	5.488	4.445	3.978	4.185	3.090
$\frac{\tau_{c1}}{\tau_{c2}}$	8.100	8.008	8.231	11.112	13.922	18.832	9.271

Table 3: Results of different nominal cases for $K_{2,min} = 0.1$

As seen in Figure 7, there is a steep increase in the value of the cost function at lower values of τ_{c1}, while the increase is flatter for higher values. It seems to be better to overshoot than undershoot the optimal value.

Looking at Table 3, there appears to be no corresponding relation between τ_{c1} and any other value. The table also shows significantly higher values of τ_{c1} compared to Case 1, showing that a lower value of $k_{2,min}$ gives a higher value of τ_{c1}.

4.3 Non-integral G1

For this case, G_1 with a non-integral response is used, as shown in Equation 14.

For this case, the nominal variables are varied as shown in Table 1, with $0.25k_{2,nom}$ as the lower value of k_2.

The result of the optimization with the nominal case of $k_1 = 1$, $\theta_1 = 1$, $k_2 = 1$, and $\theta_2 = 0$ using the MATLAB code shown in Appendix A.5, is shown in Figure 8. The result of the cost function as shown in Equation 20 was 2.7194, with $\tau_{c1} = 2.8674$.

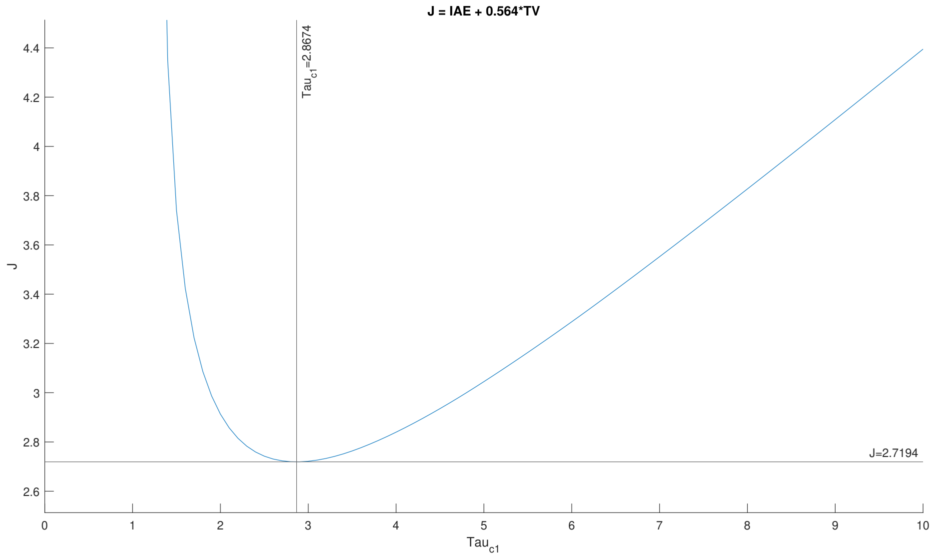

Figure 8: Relation between cost function and τ_{c1} for non-integral G1

Several other nominal cases were also optimized, the results of these optimizations are shown in Table 4.

Parameter	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7
$K_{1.nom}$	1	2	1	1	1	1	1
$K_{2.nom}$	1	1	2	1	1	1	1
$\theta_{1.nom}$	1	1	1	2	3	3	3
$\theta_{2.nom}$	0	0	0	0	0	1	2
τ_{c1}	2.867	3.073	3.075	4.672	6.427	8.125	10.172
J	2.719	2.656	2.650	2.658	2.629	2.809	3.072
SIMC τ_{c1}	1.5	1.5	1.5	2.5	3.5	4.5	6
τ_{c2}	1	1	1	1	1	1	2
$\frac{\tau_{c1}}{\text{SIMC}\tau_{c1}}$	1.912	2.049	2.050	1.869	1.836	1.806	1.695
$\frac{\tau_{c1}}{\tau_{c2}}$	2.867	3.073	3.075	4.672	6.427	8.125	5.086

Table 4: Results of different nominal cases for non integral G1

As seen in Figure 8, there is a steep increase in the value of the cost function at lower values of τ_{c1}, while the increase is flatter for higher values. It seems to be better to overshoot than undershoot the optimal value.

Looking at Table 4, there appears to be no corresponding relation between τ_{c1} and any other value. The table also shows that τ_{c1} is significantly lower than for the integral response cases, showing that the response of G_1 has an effect on τ_{c1}.

5 Conclusion and Further Work

5.1 Simple rule for τ_{c1}

From the first case with an integral response to the primary process, it appeared to be possible to use linear regression to closely model the optimal values τ_{c1}. However, this was found to be neither plausible for the second case nor for the non-integral response.

There does not appear to be any simple rule for finding an optimal τ_{c1} from these findings. There may be mathematical methods that may give a simple rule, and this needs to be further looked into.

5.2 Optimal Time scale separation for cascade control

Looking at the results, a lower bound of 5 seems to fit better, and the proposed upper bound of 10 seems to have cases where this is too small a value to have high robustness. Proposing an upper bound is not possible with the results of this thesis, as there may be higher values for different nominal cases.

5.3 IAE and TV based cost function

The cost function used in this thesis should be a good measure of the performance of a control system. It should in theory give the same robustness as the SIMC-rules, but this is not tested. The proposed cost function may also only be valid for the specific plant it was modeled by.

This method of measuring performance should be further researched to see if it gives the desired robustness and if there are huge differences in the α value for different processes. The method should also be compared to the earlier methods used to see if the results are as good, better, or worse than what has been done before.

Bibliography

[1] Sigurd Skogestad. Advanced control using decomposition and simple elements. *Annual Reviews in Control*, 56:100903, 2023. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2023.100903. URL https://www.sciencedirect.com/science/article/pii/S1367578823000676.

[2] Gregory K. McMillan. Effect of cascade control on loop performance. In *1982 American Control Conference*, pages 363–368, 1982. doi: 10.23919/ACC.1982.4787871.

[3] Hsiao-Ping Huang, I-Lung Chien, Yueh-Chung Lee, and Gow-Bin Wang. A simple method for tuning cascade control systems. *Chemical Engineering Communications*, 165:89–121, 1998. URL https://api.semanticscholar.org/CorpusID:111028925.

[4] Dola Gobinda Padhan and Somanath Majhi. Enhanced cascade control for a class of integrating processes with time delay. *ISA Transactions*, 52(1):45–55, 2013. ISSN 0019-0578. doi: https://doi.org/10.1016/j.isatra.2012.08.004. URL https://www.sciencedirect.com/science/article/pii/S001905781200122X.

[5] Osman Çakıroğlu, Müjde Güzelkaya, and İbrahim Eksin. Improved cascade controller design methodology based on outer-loop decomposition. *Transactions of the Institute of Measurement and Control*, 37(5):623–635, 2015. doi: 10.1177/0142331214536202. URL https://doi.org/10.1177/0142331214536202.

[6] F. G. Shinskey. *Process-Control Systems*. McGraw Hill, 1967.

[7] Dale E. Seborg, Thomas F. Edgar, Duncan A. Mellichamp, and Francis J. Doyle III. *Process Dynamics and Control*. Wiley, 2011. ISBN 978-0-470-64610-6

[8] Sigurd Skogestad. Simple analytic rules for model reduction and pid controller tuning. *Journal of Process Control*, 13(4):291–309, 2003. ISSN 0959-1524. doi: https://doi.org/10.1016/S0959-1524(02)00062-8. URL https://www.sciencedirect.com/science/article/pii/S0959152402000628.

[9] Sigurd Skogestad and Chriss Grimholt. The simc method for smooth pid controller tuning. *Advances in Industrial Control*, pages 147–175, 01 2012. doi: 10.1007/978-1-4471-2425-2_5.

[10] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control - Analysis and Design*. Wiley, 2006. ISBN 978-0-470-01167-6

Appendix

A Matlab code

A.1 Tuning_PID

```
function [Kc,tau_I,tau_D] = Tuning_PID(k,theta,tau_c,tau_1,tau_2)
Kc    = 1/k * 1/(theta + tau_c);
tau_I = min(tau_1,4*(tau_c+theta));
tau_D = tau_2;
end
```

A.2 IAE_SIMC.m

```
function [IAE_val, TV_val] = IAE_SIMC(k1, theta1, k2, theta2, tauc2
    , tau1_2)

s = tf('s');

G1 = k1*exp(-theta1*s)/(s);          %Integrating Process
%G1 = k1*exp(-theta1*s)/(s+1);        %Non-integrating Process
G2 = k2*exp(-theta2*s)/(tau1_2*s+1);

assignin("base","G1",G1);
assignin("base","G2",G2);

[Kc2, tauI2] = Tuning_PID(k2,theta2,tauc2,tau1_2,0);

assignin("base","Kc2",Kc2);
assignin("base","tauI2",tauI2);

tauc1 = theta1+theta2 + 0.5;

[Kc1, tauI1] = Tuning_PID(k1,theta1 + theta2 + 0.5*tauc2,tauc1,inf
    ,0);
%[Kc1, tauI1] = Tuning_PID(k1,theta1 + theta2 + 0.5*tauc2,tauc1
    ,1.5,0);

assignin("base","Kc1",Kc1);
assignin("base","tauI1",tauI1);

out_SIMC = sim('model.slx');

assignin("base","out_SIMC",out_SIMC);

IAE_val = out_SIMC.IAE(end);

TV_val = sum(abs(diff(out_SIMC.y)));

end
```

A.3 alpha_find.m

```
clear;

k1 = 1;
theta1 = 1;
k2 = 1;
theta2 = 0;

t_f = 1500;

ys  = 1;
d1  = 1;
d2  = 1;

tys = 1;
td1 = 500;
td2 = 1000;

s = tf('s');

tau2 = 1;

G1 = k1*exp(-theta1*s)/(s);         %Integrating Process
G2 = k2*exp(-theta2*s)/(tau2*s+1);

tauc2 = 1;

[Kc2, tauI2] = Tuning_PID(k2,theta2,tauc2,1,0);


tauc_val_min = fminsearch(@(tauc1) Comb_find(tauc1,0.563), 0,
   optimset('TolX',1e-27));
tauc_val     = fminsearch(@(tauc1) Comb_find(tauc1,0.564), 0,
   optimset('TolX',1e-27));
tauc_val_max = fminsearch(@(tauc1) Comb_find(tauc1,0.565), 0,
   optimset('TolX',1e-27));


tauc_vals = [tauc_val_min; tauc_val; tauc_val_max];


function Comb_val = Comb_find(tauc1, alpha)
    k1 = 1;
    theta1 = 1;

    assignin("base","tauc1",tauc1);

    [Kc1, tauI1] = Tuning_PID(k1,theta1,tauc1,inf,0);

    assignin("base","Kc1",Kc1);
    assignin("base","tauI1",tauI1);

    out = sim("model.slx");

    IAE = out.IAE(end);

    TV = sum(abs(diff(out.y)));
```

```
    Comb_val = IAE + alpha*TV;

    assignin("base","Comb_val",Comb_val);
end
```

A.4 alpha_graph.m

```
clear;

k1 = 1;
theta1 = 1;
k2 = 1;
theta2 = 0;

t_f = 1500;

ys  = 1;
d1  = 1;
d2  = 1;

tys = 1;
td1 = 500;
td2 = 1000;

s = tf('s');

tau2 = 1;

G1 = k1*exp(-theta1*s)/(s+1);         %non-Integrating Process
G2 = k2*exp(-theta2*s)/(tau2*s+1);

tauc2 = 1;

[Kc2, tauI2] = Tuning_PID(k2,theta2,tauc2,1,0);

tauc_vec = [];

for alpha = 0:0.1:1
    tauc_val = fminsearch(@(tauc1) Comb_find(tauc1,alpha), 1.5,
       optimset('TolX',1e-13));

    tauc_vec = [tauc_vec tauc_val];

    disp(string(alpha*10+1)+'/11');
end

hold on

plot(0:0.1:1,tauc_vec);
yline(1.5,'-','Tau_c_1=1.5');
%xline(-0.525,'-','alpha=-0.525');
title("Alpha for SIMC");
xlabel('alpha');
ylabel('Tau_c_1');
%ylim([0.975*min(tauc_vec) 1.025*max(tauc_vec)]);
```

```
ylim([0 4]);
xlim([0 1]);

hold off

function Comb_val = Comb_find(tauc1, alpha)
    k1 = 1;
    theta1 = 1;

    assignin("base","tauc1",tauc1);

    [Kc1, tauI1] = Tuning_PID(k1,theta1,tauc1,1,0);

    assignin("base","Kc1",Kc1);
    assignin("base","tauI1",tauI1);

    out = sim("model.slx");

    IAE = out.IAE(end);

    TV = sum(abs(diff(out.y)));
    %TV = sum(abs(diff(out.u)));

    Comb_val = IAE + alpha*TV;

    assignin("base","Comb_val",Comb_val);
end
```

A.5 comb_tauc_36.m

```
clear;

t_f = 1500;

ys  = 1;
d1  = 1;
d2  = 1;

tys = 1;
td1 = 500;
td2 = 1000;


% Tauc_val_opt_1 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,1,0),
    2.5, optimset('TolX',1e-13));
% Comb_val_opt_1 = Tauc_comb(Tauc_val_opt_1,1,1,1,0);
% disp("Ended case 1 with tauc = " + string(Tauc_val_opt_1) + "
    with comb_total of " + string(Comb_val_opt_1));
%
% Tauc_val_opt_2 = fminsearch(@(tauc1) Tauc_comb(tauc1,2,1,1,0), 3,
     optimset('TolX',1e-13));
% Comb_val_opt_2 = Tauc_comb(Tauc_val_opt_2,2,1,1,0);
% disp("Ended case 2 with tauc = " + string(Tauc_val_opt_2) + "
    with comb_total of " + string(Comb_val_opt_2));
%
```

```
% Tauc_val_opt_3 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,2,1,0), 3,
    optimset('TolX',1e-13));
% Comb_val_opt_3 = Tauc_comb(Tauc_val_opt_3,1,2,1,0);
% disp("Ended case 3 with tauc = " + string(Tauc_val_opt_3) + "
   with comb_total of " + string(Comb_val_opt_3));
%
% Tauc_val_opt_4 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,2,0),
   4.5, optimset('TolX',1e-13));
% Comb_val_opt_4 = Tauc_comb(Tauc_val_opt_4,1,1,2,0);
% disp("Ended case 4 with tauc = " + string(Tauc_val_opt_4) + "
   with comb_total of " + string(Comb_val_opt_4));
%
% Tauc_val_opt_5 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,3,0), 6,
    optimset('TolX',1e-13));
% Comb_val_opt_5 = Tauc_comb(Tauc_val_opt_5,1,1,3,0);
% disp("Ended case 5 with tauc = " + string(Tauc_val_opt_5) + "
   with comb_total of " + string(Comb_val_opt_5));
%
% Tauc_val_opt_6 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,3,1), 8,
    optimset('TolX',1e-13));
% Comb_val_opt_6 = Tauc_comb(Tauc_val_opt_6,1,1,3,1);
% disp("Ended case 6 with tauc = " + string(Tauc_val_opt_6) + "
   with comb_total of " + string(Comb_val_opt_6));
%
% Tauc_val_opt_7 = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,3,2), 6,
    optimset('TolX',1e-13));
% Comb_val_opt_7 = Tauc_comb(Tauc_val_opt_7,1,1,3,2);
% disp("Ended case 7 with tauc = " + string(Tauc_val_opt_7) + "
   with comb_total of " + string(Comb_val_opt_7));
%
% Results = [Tauc_val_opt_1 Tauc_val_opt_2 Tauc_val_opt_3
   Tauc_val_opt_4 Tauc_val_opt_5 Tauc_val_opt_6 Tauc_val_opt_7;...
%            Comb_val_opt_1 Comb_val_opt_2 Comb_val_opt_3
   Comb_val_opt_4 Comb_val_opt_5 Comb_val_opt_6 Comb_val_opt_7];


% Comb_vals = [];

% for Tauc_val = 0:0.1:10
%
%     comb_val = Tauc_comb(Tauc_val);
%     Comb_vals = [Comb_vals comb_val];
%
%     disp(string((Tauc_val)*10+1)+'/101');
% end


% hold on
% plot(0:0.1:10,Comb_vals);
% yline(Tauc_comb(Tauc_val_opt),'-','J='+string(Tauc_comb(
   Tauc_val_opt)));
% xline(Tauc_val_opt,'-','Tau_c_1='+string(Tauc_val_opt));
% title("J = IAE + 0.564*TV");
% xlabel('Tau_c_1');
% ylabel('J');
% ylim([1 3]);
% xlim([0 10]);
% hold off
```

```
%Tauc_val_opt = fminsearch(@(tauc1) Tauc_comb(tauc1,1,1,1,0),
   5.479, optimset('TolX',1e-13));
%Tauc_comb(Tauc_val_opt,1,1,1,0);

function Comb_total = Tauc_comb(tauc1, k1_nom, k2_nom, theta1_nom,
   theta2_nom)
    % k1_nom = 1;
    % k2_nom = 1;
    % theta1_nom = 1;
    % theta2_nom = 0;

    K2_vec      = [ones(12,1)*2*k2_nom; ones(12,1)*k2_nom; ones
       (12,1)*0.25*k2_nom];

    theta2_vec = [ones(6,1)*theta2_nom; ones(6,1)*max(1.25*
       theta2_nom,0.5);...
                   ones(6,1)*theta2_nom; ones(6,1)*max(1.25*
                      theta2_nom,0.5);...
                   ones(6,1)*theta2_nom; ones(6,1)*max(1.25*
                      theta2_nom,0.5);];

    theta1_vec = [ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom;
       ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom;...
                   ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom;
                      ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom;
                      ...
                   ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom;
                      ones(3,1)*theta1_nom; ones(3,1)*2*theta1_nom
                      ;];

    K1_vec      = [k1_nom/2; k1_nom; 2*k1_nom; k1_nom/2; k1_nom; 2*
       k1_nom; k1_nom/2; k1_nom; 2*k1_nom;...
                   k1_nom/2; k1_nom; 2*k1_nom; k1_nom/2; k1_nom; 2*
                      k1_nom; k1_nom/2; k1_nom; 2*k1_nom;...
                   k1_nom/2; k1_nom; 2*k1_nom; k1_nom/2; k1_nom; 2*
                      k1_nom; k1_nom/2; k1_nom; 2*k1_nom;...
                   k1_nom/2; k1_nom; 2*k1_nom; k1_nom/2; k1_nom; 2*
                      k1_nom; k1_nom/2; k1_nom; 2*k1_nom;];

    vals_vec = [K2_vec theta2_vec theta1_vec K1_vec];

    [simc_IAE, simc_TV] = IAE_SIMC(k1_nom,theta1_nom,k2_nom,
       theta2_nom,1,1);

    Comb_total = 0;

    runs = length(1:36);

    run = 0;

    tau2 = 1;

    s = tf('s');

    for i = 1:36
        k2      = vals_vec(i,1);
```

```
        theta2 = vals_vec(i,2);
        theta1 = vals_vec(i,3);
        k1     = vals_vec(i,4);

        G1 = k1*exp(-theta1*s)/(s);          %Integrating Process
        %G1 = k1*exp(-theta1*s)/(s+1);        %Non-integrating
            Process
        G2 = k2*exp(-theta2*s)/(tau2*s+1);

        assignin("base","G1",G1);
        assignin("base","G2",G2);

        tauc2 = max(1,theta2_nom);

        [Kc2, tauI2] = Tuning_PID(k2_nom,theta2_nom,tauc2,tau2,0);

        assignin("base","Kc2",Kc2);
        assignin("base","tauI2",tauI2);

        [Kc1, tauI1] = Tuning_PID(k1_nom,theta1_nom + theta2_nom +
            0.5*tauc2,tauc1,inf,0);
        %[Kc1, tauI1] = Tuning_PID(k1_nom,theta1_nom + theta2_nom +
             0.5*tauc2,tauc1,1.5,0);

        assignin("base","Kc1",Kc1);
        assignin("base","tauI1",tauI1);


        out = sim('model.slx');

        run = run + 1;
        %disp(string(run)+'/'+string(runs));

        assignin("base","t_"+string(i),out.tout);
        assignin("base","y_"+string(i),out.y);
        assignin("base","u_"+string(i),out.u);

        comb_val = out.IAE(end)/simc_IAE + 0.564*sum(abs(diff(out.y
            )))/simc_TV;

        assignin("base","comb_val_"+string(i),comb_val);

        Comb_total = Comb_total + comb_val;
    end

    Comb_total = Comb_total/36;
    %disp("Ended run with tauc = " + string(tauc1) + " with
        comb_total of " + string(Comb_total));

end
```

B Simulink scheme

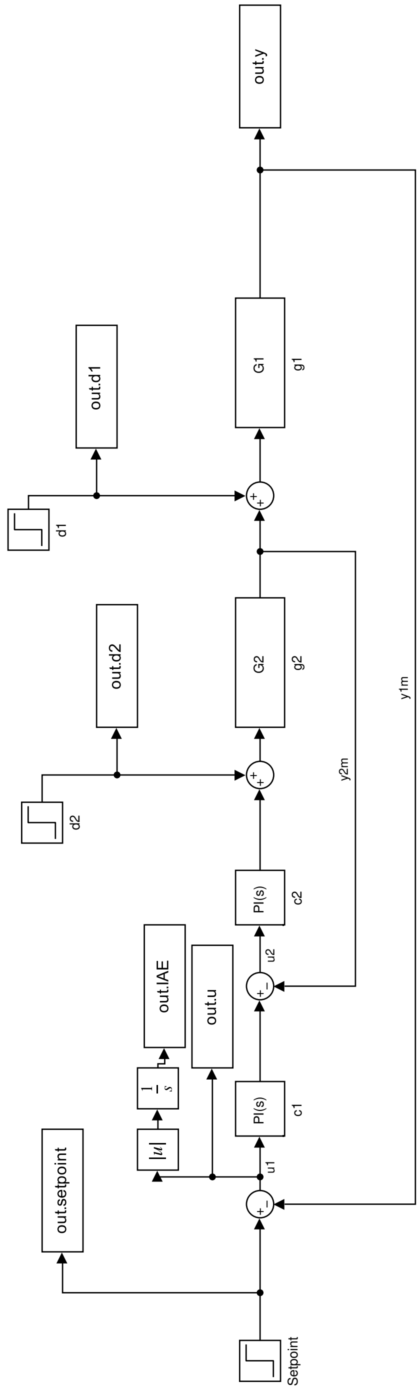

Figure 9: model.slx

NTNU

Norwegian University of
Science and Technology

NTNU

Deklarasjon om KI-hjelpemiddel

Har det i utarbeidinga av denne rapporten/avhandlinga blitt anvendt KI-baserte hjelpemiddel?

○ Nei

◉ Ja

Viss *ja*: spesifiser type av verktøy og bruksområde under.

Tekst

☑ **Stavekontroll.** Er delar av teksten kontrollert av: *Grammarly, Ginger, Grammarbot, LanguageTool, ProWritingAid, Sapling, Trinka.ai* eller liknande verktøy?

☐ **Tekstgenerering.** Er delar av teksten generert av: *ChatGPT, GrammarlyGO, Copy.AI, WordAi, WriteSonic, Jasper, Simplified, Rytr* eller liknande verktøy?

☐ **Skriveassistanse.** Er ein eller fleire av ideane eller framgangsmåtane i oppgåva foreslått av: *ChatGPT, Google Bard, Bing chat, YouChat* eller liknande verktøy?

Viss *ja* til anvending av eit tekstverktøy - spesifiser bruken her:

Writefull og ChatGPT har blitt brukt til å hjelpe med rettskriving og omformulering.

Koder og algoritmar

☐ **Programmeringsassistanse.** Er delar av koden/algoritmane som i) framtrer direkte i rapporten eller ii) har blitt anvendt for produksjon av resultat slik som figurar, tabellar eller tallverdiar blitt generert av: *GitHub Copilot, CodeGPT, Google Codey/Studio Bot, Replit Ghostwriter, Amazon CodeWhisperer, GPT Engineer, ChatGPT, Google Bard* eller liknande verktøy?

Viss *ja* til anvending av eit programmeringsverktøy - spesifiser bruken her:

Bilde og figurar

☐ **Bildegenerering.** Er eitt eller fleire av bilda/figurane i rapporten blitt generert av: *Midjourney, Jasper, WriteSonic, Stability AI, Dall-E* eller liknande verktøy?

Viss *ja* til anvending av eit bildeverktøy - spesifiser bruken her:

Andre KI-verktøy. Har andre typar av verktøy blitt anvendt? Viss *ja* spesifiser bruken her:

☑ Eg er kjend med NTNU sitt regelverk: *Det er ikkje lov å generere svar ved hjelp av kunstig intelligens og levere det heilt eller delvis som eiget svar. Eg har derfor gjort greie for all bruk av kunstig intelligens enten i) direkte i rapporten eller ii) i dette skjemaet.*

Geir Arne Vassnes /
14.07.2024 / Austevoll

Underskrift/Dato/Stad