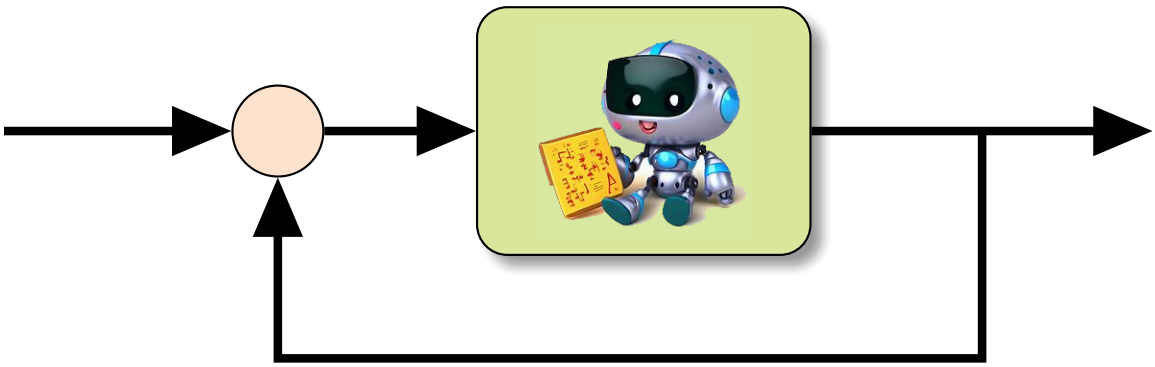
Forelesningsnotater for AIST1001 (versjon: v0-2-1-30112023)

Automatisering, introduksjon

Christian Fredrik Sætre

christian.f.satre@{ntnu.no,gmail.com}



November 2023

Institutt for teknisk kybernetikk, NTNU

NTNU

Innhold

Forord

Dette er forelesningsnotater for emnet AIST1001 (lenke til emnesiden til AIST1001).

Notatene er i stor grad inspirert av og basert på kompendiet 'Kybernetikk, Introduksjon' av Jan Tommy Gravdahl, som brukes i søsteremnet TKK4100 Kybernetikk, introduksjon. Noen deler er også inspirert av forelesningsnotatene til Fredrik Dessen i emnet IELET2101.

En stor takk rettes til det første kullet av studenter høsten 2023, for deres svært nyttige tilbakemeldinger både mtp. notatene og faget som helhet! Spesielt ønsker jeg å takke Ole K. Helgedagsrud for hans detaljerte kommentarer og forslag til forbedringer.

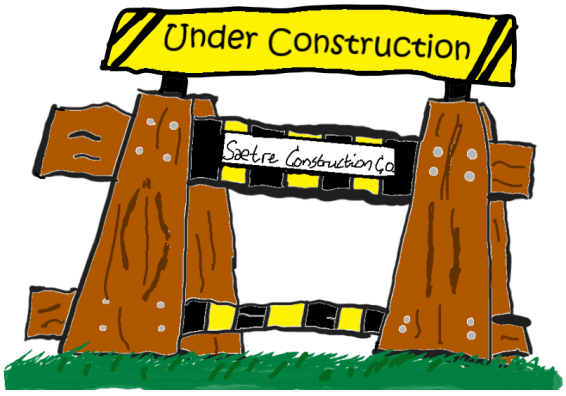
Jeg har som mål at disse notatene skal følge «less is more»-prinsippet, altså få frem idéene, teoriene og metodene med minst mulig blablabla (og per nå er det mye blablabla). I tilegg skal motivasjonen for *hvorfor* du har lyst til å lære et tema (i motsetning til at jeg mener du bør det) komme frem så fort som mulig, og ikke minst bør dette faktisk motivere deg.

Å få til punktene over er ikke så lett uten tilbakemeldinger fra dere, så ikke være redd for å gi meg beskjed (f.eks. via `christian.f.satre@ntnu.no`) hvis du synes jeg har mislyktes noen av disse punktene, eller hvis det noe du synes har behov for mer/bedre forklaringer, etc.

Notatene er forstatt under utvikling; gi derfor gjerne også beskjed hvis du ser noen feil eller mangler, og ikke minst om du har noen spørsmål eller forslag til endringer. Det vil jeg sette stor pris på!

Relevant MATLAB-kode finner du her eller her.

⚠️ **NB!** Seksjoner markert med symbolet * er ikke direkte pensum.



Merk: Disse notatene er fortsatt under utvikling (dette er versjon v0-2-1-30112023). Figuren over indikere at noe ikke er ferdigstilt enda, men vil (forhåpentligvis) komme etterhvert. Ting merkert i rødt er som regel mine egne notater.

Introduksjon og Motivasjon

1. Introduksjon

Som nybakt universitetsstudent kan det være greit å lagre følgende baki hodet et sted:[1]

«*Alt er vanskelig inntil du kan det.*» 🤗

Merk dog at når du først «kan det», så bør du passe deg for Dunning-Kruger-effekten! Og når du virkelig, virkelig kan det, ja da bør du passe deg for bedragersyndromet.

Hovedmålet med dette faget er å gi deg et sterkt automatiseringsteknisk fundament for studiene dine videre, og ikke minst gjøre dette på en måte som får deg til å se magien ved å jobbe med automatisering, intelligente systemer, kybernetikk og robotikk!

Det er jo selvsagt begrenset for mye vi klarer å få dyttet ned i din kanskje helt nye «automatisering-verktøykasse» på ett semester. Selv om det derfor blir en del super-enkle (og ikke nødvendigvis så veldig virkelighetsnære) eksempler, så skal vi uansett prøve å ha et fokus på praktisk og (for det meste) andvendbar automatiseringsteknikk. Håpet er at det vil være lettere for deg å lære flere av metodene og teoriene mer grundig når du ser dem igjen i senere fag, siden du da allerede så vidt har sett det før og dermed ikke er helt nytt for deg.

1.1. Kort om det praktiske

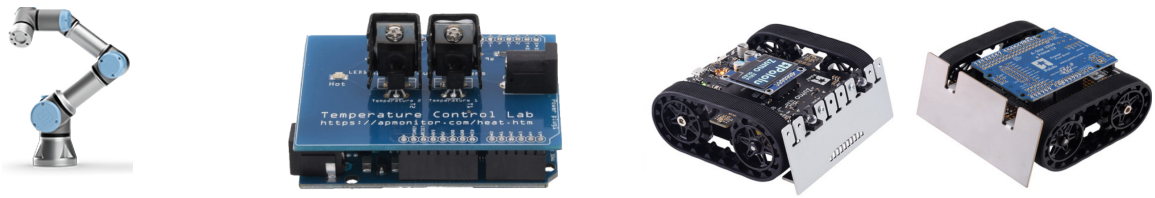
1.1.1 Arbeidskrav

For å gå opp til eksamen må du ha

- 3 av 6 øvinger godkjent.
- 3 av 3 laboratorieoppgaver godkjent (se fig. 1.1).

⚠️ **Achtung!** Det er ditt eget ansvar å passe på at du har dette godkjent!

[1] Fra mitt perspektiv er nok følgende emoji-valg noe mer passende: «*Alt er vanskelig inntil du kan det*» 😈 Men jeg kom fra til at den kanskje ikke var så motiverende...



(a) Robot-lab. (b) Temperaturregulerings-lab. (c) Ståhjuling-lab.

Figur 1.1: Systemene brukt i de forskjellige laboratorieøvingene.

1.1.2 Prosjekt

Det skal også gjennomføres et prosjekt i grupper på 4-5 studenter. Muntlig presentasjon på ca. 20 minutter teller 30% av karakteren i faget. Mer om prosjektet og gruppesammensetning, etc., kommer etterhvert.

1.1.3 Eksamen

Emnet har digital eksamen. Du finner relevant informasjon om dette her: https://i.ntnu.no/wiki/-/wiki/Norsk/Digital+hjemmeeksamen+-+for+studenter.

Dato, tidspunkt og varighet: 30.11 kl. 15:00, 4 timer.

Du får tilgang på følgende tredjeparts programvare: MATLAB og Simunlink. Du vil også kunne bruke tegnebrett (trolig Wacom One) på langsvarsoppgaver.

1.1.4 Pensum, bok og emnesider

Pensum er disse notatene (**NB!** vedleggene samt kapitler/seksjoner merket med " * " er unntak). Merk at notatene vil bli kontinuerlig oppdatert. Dette er versjon v0-2-1-30112023.

Emnesiden finner du her: https://www.ntnu.no/studier/emner/AIST1001.
Du finnes også noe relevant materiell og videoer her: https://folk.ntnu.no/christfs/AIST1001/

1.1.5 Annen litteratur og relevant kilder:

Norske bøker om reguleringsteknikk: (det er noen valgmuligheter her)

- *Kybernetikk Introduksjon* av Jan Tommy Gravdahl.
- *Reguleringsteknikk* av Kåre Bjørvik og Per Hveem.
- *Reguleringsteknikk* av Jens G. Balchen, Trond Andresen og Bjarne A. Foss.
- *Modeling, Simulation and Control* av Finn Aakre Haugen.

Artikler: Teknisk ukeblad har en samling med artikler kalt Praktisk reguleringsteknikk. F.eks. følgende artikler av Christian Svensson er temmelige relevante: PID-regulatoren og Tuning av PID-regulatorer.

Videoer og relevante YouTube-kanaler:

Trond Andresen underviste faget TTK4105 – Reguleringsteknikk i en årrekke, og bygde et rykte for å være en god foreleser og pedagog. I tillegg til boken over, har han tilgjengeliggjort opptak av flere relevante forelesninger og annet materiale; se: https://folk.ntnu.no/tronda/regtek-kurs/.

Brian Douglas er et velkjent navn blant de fleste studenter som studerer reguleringsteknikk grunnet hans fantastiske YouTube videor, både via sin egen kanal og MATLAB sin; se følgende for en samling: https://engineeringmedia.com/videos. Han har også en lettleste bok, se: http://bit.ly/2XLlAKl.

Steve Brunton har videor om alt fra grunnleggende reguleringsteknikk til mer avansert metoder og også en del maskinlæring, se: https://www.youtube.com/c/Eigensteve

Kristin Y. Pettersen, som er emneansvarlig for masteremnet TTK4150 - Ulineære systemer, har noen gode videoer relatert til stabilitet, og lineære- vs ulineære systemer, spesifikt L1.2-serien, se: https://www.youtube.com/c/KYPTeachTech.

1.1.6 Programvare og verktøy for faget og fremtidig studier

- MATLAB og Simulink: Mer om disse straks
- LaTex og Overleaf: Hvordan lage fine vitenskapelige PDF-dokumenter.
- Git og GitHub: Versjonskontroll av kode.
- Visual Studio code: En populær IDE for å skrive kode.

MATLAB og Simulink:

Både i dette og andre fag vil du hyppig bruke programmet MATLAB og tilhørende program Simulink. Det kan være greit å installere og titte litt på disse allerede nå. Du finner mer informasjon via følgende lenke: https://i.ntnu.no/wiki/-/wiki/Norsk/Matlab.

MATLAB og Simulink kan bruke til så mangt, fra reguleringsteknikk og robotikk, til maskinlæring og signalprosessering. Det finnes også en symbolic toolbox som kan hjelpe deg med algebra og kalkelus.

Oppgave 1.1. Ta følgende kurs på ca. 2 timer hver:[a]

MATLAB: https://matlabacademy.mathworks.com/details/matlab-onramp/gettingstarted

Simulink: https://matlabacademy.mathworks.com/details/simulink-onramp/simulink

Kursbeviset fra det første tilsvarer å få godkjent øving 1, så husk å ta vare på det!

[a] Se følgende for flere slike kurs: https://matlabacademy.mathworks.com/

1.2. Viktige begreper og motivasjon for fagets innhold

Men hva er egentlig automatisering? Det skal komme et mer utfyllende svar om litt, men foreløpig kan du nøye deg med følgende:

Hvordan bruke tilgjengelig informasjon om et system for å få det til å gjøre det vi ønsker. 🤓

Relatert til dette er hoveddelene i dette faget:

- **Modellering og Simulering av Dynamiske systemer:** For å få systemer til å gjøre som vi vil, så må vi ha en måte å forstå, analysere og representere disse. Vårt matematiske verktøy her er dynamiske systemer. Hvordan å utlede (matematisk modellere) slike dynamiske systemer for fysiske prosesser fra f.eks fysikkens lover er naturlig nok essensielt. Det er også en stor fordel å kunne analysere og utforske slike systemer i en datamaskin, noe vi kan gjøre ved hjelp av numeriske simuleringer.

- **Reguleringsteknikk**: Innen flere automatiseringsapplikasjoner ønsker man å kunne påvirke og styre dynamiske systemer for å oppnå en ønsket respons. Dette faller inn under grenen *reguleringsteknikk*, som bl.a. omfatter design og analyse av tilbakekoblingssløyfer, stabilitet, PID-regulering, og implementasjon av regulatorer i ekte systemer.

- **Robotikk:** For å kunne få en robot til å bevege seg som ønsket, må man vite hvordan utslag i dens ledd-variabler (tenk skulder, albue og håndledd) påvirker dens konfigurasjon (f.eks. posisjonen til pekefingeren). En slik problemstilling faller inn under kinematikk, som handler om å beskrive et objekts bevegelse og konfigurasjon uten å ta hensyn til årsakene som fører til bevegelsen. Vi skal også se på hvordan man kan styre og planlegge robotbevegelser.

- **Maskinlæring og Optimering:** Maskinlæring handler om utvikling av algoritmer som kan lære fra og gjøre forutsigelser basert på data. Disse teknikkene benyttes til å bygge modeller (f.eks. kunstig nevrale nettverk) som kan lære sammenhenger fra data. Selve læreprosessen tar i bruk en form for *optimering*. Optimering kan også tas i bruk til å fine finne den «beste» løsningen til et problem (f.eks. raskeste vei fra A til B) gitt et sett av begrensninger (f.eks. fartsgrenser for de forskjellig veiene dit).

- **Sekvens- og datatstyring** omhandler organisering og styring av operasjoner i en bestemt sekvens. Dette kan innebære styring av maskiner, prosesser, og andre aspekter av automatiserte systemer på en effektiv og nøyaktig måte.

Vi skal i det som følger ta en nærmere titt på reguleringsteknikk, som er den viktigste delen av dette faget. Men først starter vi med å beskrive de sentrale begrepene kybernetikk og automatisering.

1.2.1 Kybernetikk

Alternative kilder: Store Norske Leksikon.

Propagandaen starter som regel veldig tidlig ved Institutt for teknisk kybernetikk, så du har nok

allerede litt kunnskap om hva «kybernetikk» er fra immatrikuleringsdagen. Studieretningen heter dog 'Kybernetikk og robotikk', så la oss uansett starte med dette begrepet:

Kybernetikk er en tverrfaglig vitenskap som fokuserer på å observere og beskrive hvordan tilstander i tekniske prosesser og levende vesener varierer over tid og samhandler med hverandre. Målet er å påvirke disse prosessene for å oppnå ønsket oppførsel. Tilbakekobling (mer om det om litt) er et sentralt konsept i kybernetikk, som innebærer måling av en størrelse som skal styres, sammenligning med en ønsket referanseverdi, og påvirkning av prosessen basert på avviket mellom den målte og den ønskede verdien. Kybernetikk er også en informasjonsvitenskap, og kybernetikere beskriver hvordan informasjon strømmer gjennom systemer omtrent som elektrisitet gjennom elektriske kretser. Kybernetikk har stor nytteverdi i teknologiske systemer og brukes i stor grad i teknisk kybernetikk, automatisering, robotikk, servoteknikk og reguleringsteknikk.

1.2.2 Automatiseringssystemer

Alternative kilder: Store norske leksikon (SNL); Wikipedia .

Man kan tenke på automatisering som teknikker for å få systemer eller oppgaver til å fungere eller bli gjennomført uten eller med liten grad av menneskelig medvirkning. Automatisering benyttes på alle områder hvor det er ønskelig å erstatte menneskelig arbeidskraft med selvvirkende (autonome) systemer: i industri, handel og kontor, transport, kommunikasjon, administrasjon, helsevesen og i hjemmene. Dette inkluderer alt fra store automatiseringsceller i industrien og selvkjørende biler, til chatboter og minibanker, samt ting vi bruker til daglig som vaske- og oppvaskmaskiner.

Oppbyinngen av et automatiseringssystem: Et teknisk system som skal automatiseres, for eksempel en kjemisk industriprosess, en maskin eller fabrikk, må ha sensorer og pådragsorganer. Sensorene måler forskjellige tilstander i systemet, som trykk, temperaturer og posisjoner. Med pådragsorganene kan man via et pådragssignal påvirke systemet, for eksempel starte eller stoppe motorer eller stille lukningsgraden på ventiler. Denne typen automatisering kalles for *styring*. Tradisjonelt skiller man mellom to hovedtyper av styring: sekvensstyring og regulering.

Sekvensstyring vil si å sørge for at visse operasjoner skjer i en bestemt rekkefølge. Ta for eksempel en mulig prosedyre til en vaskemaskin: 1. Åpne påfyllingskranen. 2. Når en nivåsensor gir signal om at tanken er full, steng kranen. 3. Skru på varmeelementet. 4. Når en temperatursensor gir signal om at ønsket temperatur er oppnådd, skru av varmeelementet. 5. Start trommelmotoren. 6. Etter en viss tid, stopp den. 7. Start avtappingspumpen, etc.

Styring TODO

1.3. Hva er reguleringsteknikk?

Alternative kilder: Store Norske Leksikon; Everything You Need to Know About Control Theory (YouTube-video) av Brian Douglas .

Figur 1.2: Automatiseringssystemer og reguleringsteknikk finnes «over alt»: Eksempel på områder hvor avansert regulerings teknikk er tatt i bruk inkluderer: Kraftsystemer; Prosesskontroll; Luftfart og romfart; Biler og fartøy; Bygninger og trafikksystemer; Robotikk og hvitevarer; Datasystemer og mobiltelefoner; Reklame og økonomi; Kunst og spill; Fysikk og biologi (figur fra [Samad et al., 2020]).

Legge opp litt mer som BR-video.

Reguleringstekniske problemer[2] dukker opp «over alt» (se figur 1.2). Reguleringsteknikk (eng. «control theory») er egentlig et gigantisk felt, bestående av en rekke subdisipliner og metoder slik som vist i figur 1.3.

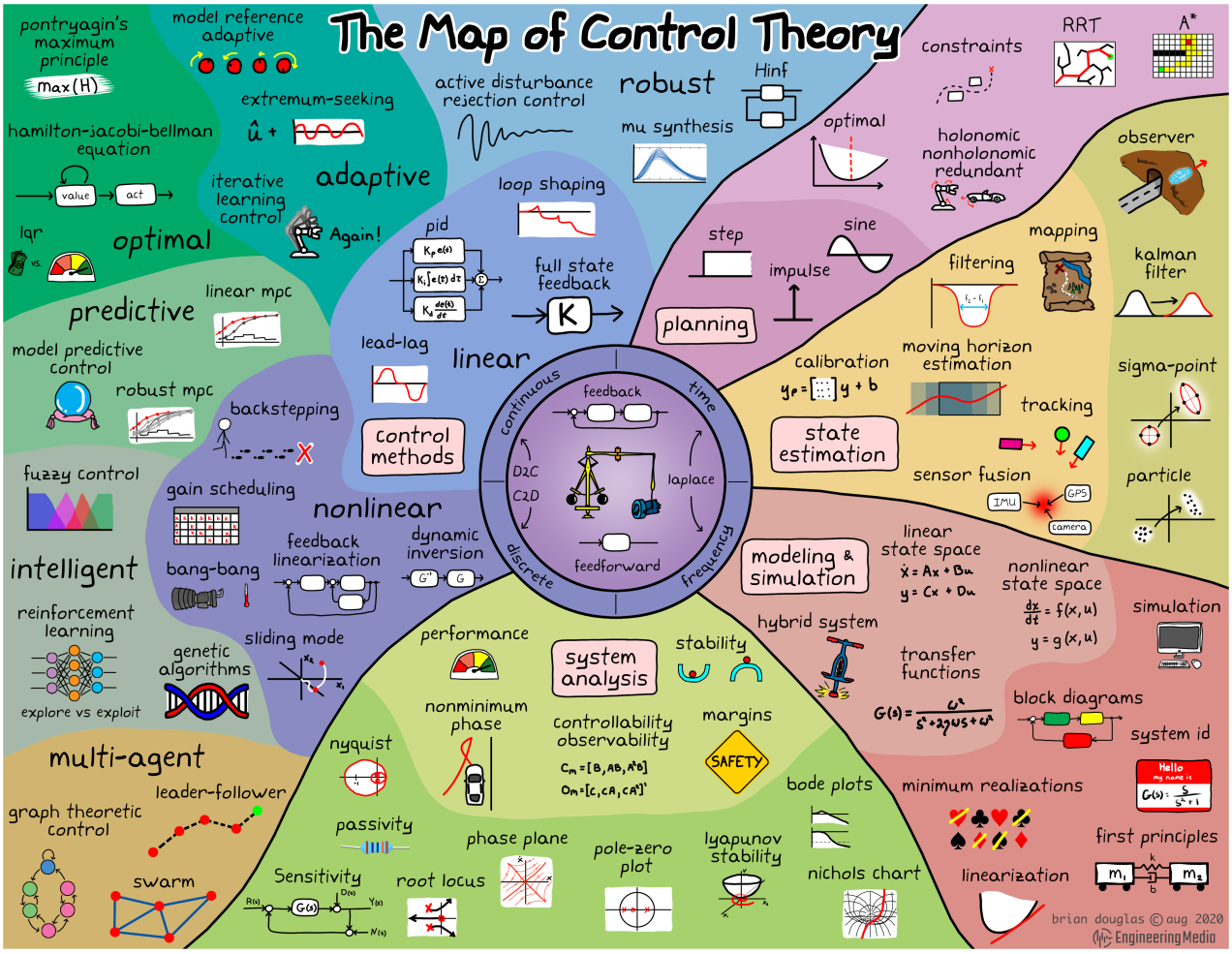
Målet innen reguleringsteknikk er å påvirke såkalte dynamiske systemer/prosesser (f.eks. en bil). Mer spesifikt, så man å styre gitte størrelser kalt prosessvariabler (f.eks. hastigheten til en bil) slik at de følger en ønsket referanseprofil (f.eks. fartsgrensen på 80 km/t) til tross for eventuelle forstyrrelse (f.eks. vind). Dette oppnås ved å kontinuerlig måle gitte størrelser ved hjelp av sensorer, og så bruke disse målingene til å justere systemets pådragsorganer (bilens motor) for å oppnå ønsket referanseverdi. Denne justeringen utføres av en *regulator*.

En viktig karakteristikk ved slike systemer er at man ikke kan få til en umiddelbar (momentant) endring i prosessvariablene. Dette er for eksempel illustrert i figur 1.4, hvor et hopp (sprang) i utgangen til pådragsorganet (f.eks. gasspedalen i en bil) fører til en kontinuerlig endring i utgangsvariabelen (f.eks. hastigheten til bilen).

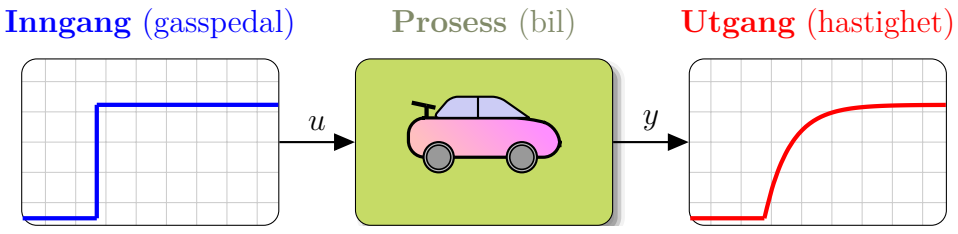
Fun facts, bemerkninger og annet dill dall (you may skip)

For noen år tilbake hadde instituttet besøk av svensken Karl Johan Åstrøm, som er en legende innen feltet. Han holdt en flott presentasjon (på engelsk) om reguleringsteknikk. Det kan være vel verdt å ta en titt på den: https://youtu.be/R-h66PrQ808.

[2]Reguleringsteknikk kan løst deles opp i to problemkategorier: Prosessregulering og Servoteknikk.



Figur 1.3: Reguleringsteknikkens verden er både (farge-)rik og variert, og består av en stor rekke felt og metoder. Vi skal i dette faget for det meste holde oss innen det mørkeblå området «linear». Denne flotte figuren er laget av den reguleringstekniske YouTube-kongen Brian Douglas: https://engineeringmedia.com/map-of-control.

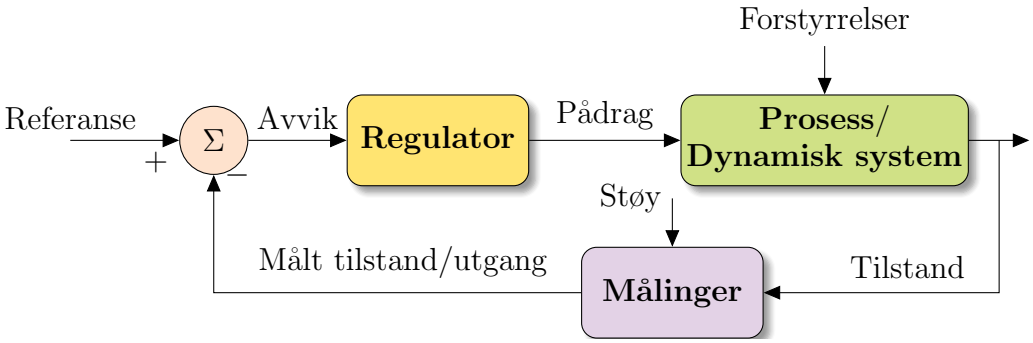


Figur 1.4: Illustrasjonen av et dynamisk system/prosess sin respons til et konstant inngangssignal (pådrag): Utgangen endrer seg ikke momentant, men gradvis mot en stasjonær verdi.

Tilbakekobling:

Et svært sentralt konsept innen reguleringsteknikk er *tilbakekobling*. Dette konseptet, som er illustrert i figur 1.5, går ut på at man måler akkurat den størrelsen man er ute å styre (f.eks. en bils hastighet) ved hjelp av en sensor (speedometer). Man sammenligner så differansen

(avviket) mellom målingen og ønsket referanseverdi/settpunkt (si 80 km/t). Basert på dette avviket, så vil en regulator i et cruisekontroll-system bestemme hva pådragsorganet (motoren) bør gjøre for å oppnå denne referanseverdien.



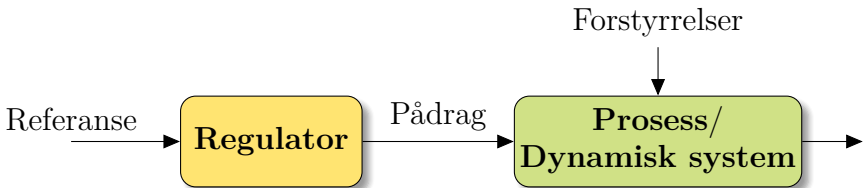
Figur 1.5: Illustrasjon av et enkelt reguleringssystem med tilbakekobling («lukket sløyfe»): Man mater målinger av tilstandene/utgangene man er ute etter styre «tilbake igjen» for å regne ut pådraget (ved hjelp av en regulator) slik at systemet/prosessen følger en ønsket referanse.

Ting man ønsker å oppnå med tilbakekobling:

- Få utgangen til å følge den ønskede referansen;
- Gjøre den lukkede sløyfen lite sensitiv til variasjoner i prosessen;
- Redusere effekten av eksterne forstyrrelser;
- Minimere effekten av målestøy.
- (Bonus) Gjøre systemet "bedre" enn det var uten tilbakekobling!

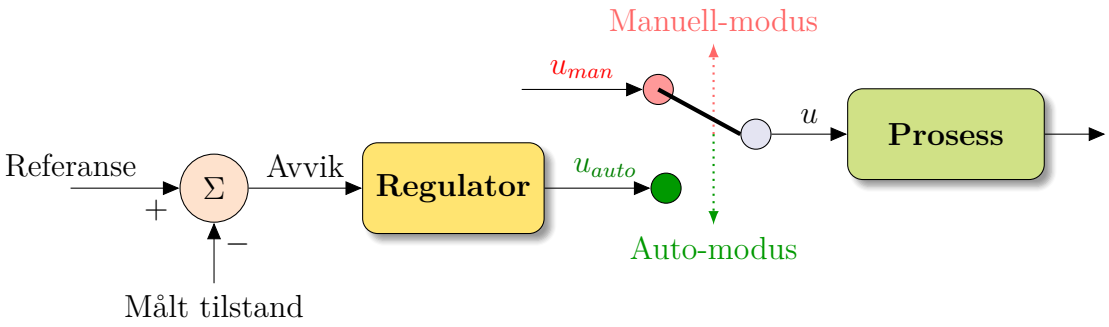
Åpen vs lukket sløyfe

Vi skiller mellom et system *med* tilbakekobling, såkalt **lukket sløfye** (eller «auto-modus» i industrien) slik som det i figur 1.5, og *uten* tilbakekobling, såkalt **åpen sløyfe** (manuell-modus) slik som vist i figur 1.6. I kommersielle regulatorer har man som regel muligheten til å bytte mellom disse modusene (se figur 1.7).



Figur 1.6: Illustrasjon av et enkelt reguleringssystem i åpen sløyfe (uten tilbakekobling).

En god analogi for åpen-sløyfe-regulering er at du skal stoppe en bil ved et veikryss på grunn av et rødt lys. Du kjenner bilen din, du vet hvilken hastighet bilen har, og hvor langt det er til stedet du skal stoppe. Du lukker så øynene og implementerer din «bremsestrategi», krysser fingrene og håper du stopper der du ønsker å stoppe. Siden du ikke bruker øynene dine til å



Figur 1.7: I de fleste kommersielle regulatorer kan man bytte mellom en regulator med tilbakekobling (auto-modus) og uten tilbakekobling (manuell-modus).

korrigere i tilfelle du bremser for mye eller for lite, ender du sannsynligvis opp med å stoppe for tidlig, eller i verste fall, ender opp med å stoppe midt i krysset.

Ved åpen-sløyfe regulering slik som figur 1.6 regner man altså ut pådraget kun basert på ønsket referanseverdi (og muligens den matematiske modellen av systemet). Siden prosessutgangen, som er den størrelsen vi ønsker å styre, ikke tas hensyn til i regulatoren så vil det ikke gjøres nødvendig korrigerende tilltak hvis systemet blir utsatt for eksterne forstyrrelser (f.eks. et vindkast). En slik strategi vil også være svært sårbar i frohold til feil og usikkerhet i den matematiske modellen av systemet.

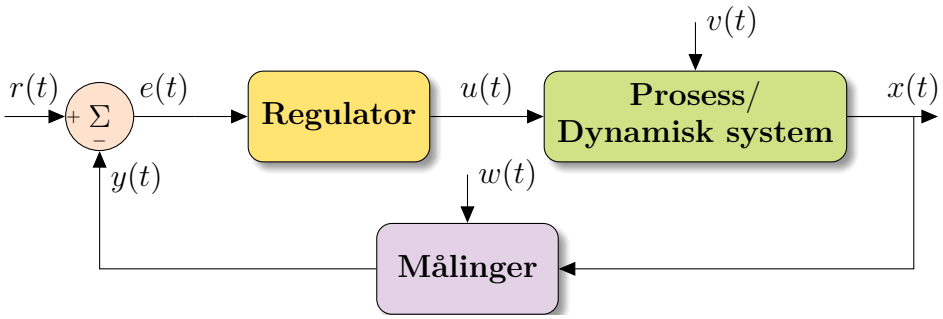
Det går dog an å bruke informasjonen om systemets matematiske modell sammen med den ønskede referansen og/eller målinger av eventuelle forstyrrelser *sammen* med en tilbakekobling. Dette kalles for **foroverkobling**, og er noe vi skal se nærmere på i kapittel 6.

Tabell 1.1: Symboler for vanlige variabler innen reguleringsteknikken.

Bregrep	**Symbol**	**Eksempler**
Tilstand	$x(t)$	Hastigheten til en bil, væskehøyden i en tank, varmen i en ovn.
Pådrag	$u(t)$	Dreiemoment fra en motor, væskestrøm gjennom en reguleringsventil.
Måling	$y(t)$	Turtall fra speedometer, væskehøyde fra flotør, temperatur fra termostat.
Målestøy	$w(t)$	
Referanse	$r(t)$	Ønsket hastighet til en bil, væskenivå i en tank eller temperatur i en ovn.
Avvik	$e(t)$	Differansen mellom ønsket verdi, altså referansen $r(t)$, og målt verdi, $y(t)$.
Forstyrrelse	$v(t)$	Vind, varierende innstrøm i en tank, temperatur utenfor en ovn.

1.3.1 Symboler og variabler som inngår i et reguleringssystem

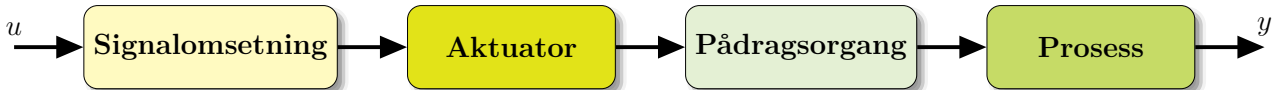
I figur 1.8 har vanlige symboler for variablene bruk i figur 1.5 blitt satt inn. En beskrivelse av disse finner du i tabell 1.1. Disse er alle funksjoner av tiden t.



Figur 1.8: Symbolene som inngår i et enkelt reguleringssystem med tilbakekobling.

1.3.2 Aktuatorer og pådragsorganer

Pådragsignalet u, som blir beregnet av regulatoren i et system, inneholder informasjon som skal tilbakeføres til eller påvirke prosessen vi ønsker å kontrollere. Dette signalet må oftest omformes fra et lav-energi signal i en datamaskin til et høy-energi signal som kan påvirke prosessen direkte. For eksempel, i tilfellet med en automatisk styrbar heisekran, kan motoren påvirke lasten med krefter på mange tusen Newton. Det er derfor et behov for å oversette u til reelle, fysiske verdier som har direkte innflytelse på prosessen. Dette er illustrert i figur 1.9.



Figur 1.9: Signalkjede fra pådragssignal u til påvirking (via pådragsorganet) på prosessen.

Først blir u omregnet i blokken Signalomsetning. Dette kan være en digital-til-analog omforming, omregning mellom strøm og spenning, skalering, eller annen form for signalbehandling. En strøm på 4–20mA er et standard signal som ofte brukes til dette formålet, hovedsakelig på grunn av dens robusthet mot endringer i resistans - en grunn til at spenning sjelden brukes for signaloverføring. Deretter blir signalet sendt til en aktuator, som f.eks. en elektromotor. Motoren påvirker pådragsorganet, f.eks. en ventil, som igjen har direkte effekt på prosessen.

Attenzione! Hver av blokkene i figur 1.9 kan inneholde sin egen dynamikk (dette gjelder for den del også «Målinger»-blokken i figur 1.8). Denne dynamikken vil nødvendigvis påvirke ytelsen til reguleringssystemet som helhet. I dette faget antar vi at denne dynamikken er en del av prosess-blokken.

1.3.3 Ting å tenke på når man skal designe en regulator

Vi nevner også noen viktige betraktninger relatert til regulatordesign før vi hopper videre til noen eksempler:

- Tilbakekobling kan gjør en stabil prosess ustabil; det vil også mate målestøy inn i prosessen.
- Man kan ofte øke ytelsen ved å kombinere tilbakekoling med, f.eks., fremoverkobling eller alternative reguleringsstrukturer (kap. 6).

- Reguleringsteknikk handler ikke bare om regulatorer og innstilling av disse, men også tilstands-estimering, systemidentifikasjon, maskinlæring, etc. (se figur 1.3).
- Reguleringsteknikk kan variere veldig fra felt til felt; f.eks. har prosessregulering som regel ganske annerledes problemer og ytelseskriterier enn bevegelseskontroll relatert til robotikk.
- Det er ofte mye fokus på settpunkt-/referanse-regulering, men ofte er forstyrrelsesresponsen viktigere!

1.4. Reguleringsteknikk 101 med Tom i Tallinjeveien

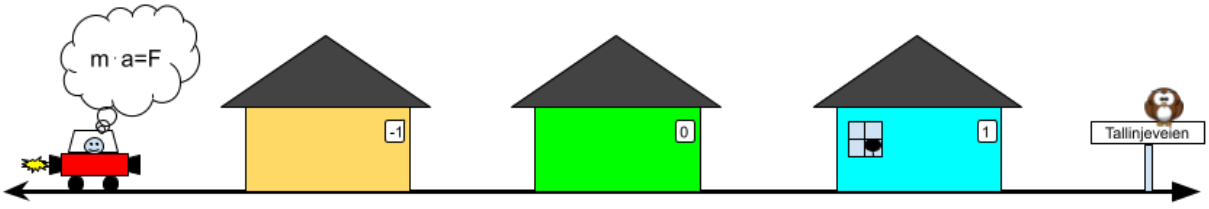
Vi skal tidvis introdusere nye konsepter i dette faget ved hjelp av litt tullete (men enkle!) eksempeler basert på strekfiguren «Tom i Tallinjeveien» (TiT). Tanken er at disse eksemplene er så banale og enkle å forstå at vi kan bruke tiden på et som faktisk er mest viktig for deres læring, nemlig å forstå de grunnleggende ideene og konseptene i dette faget.

I denne seksjonen får du hilse på Tom, samt få se på noen av de mest grunnleggende problemene har trenger vår hjelp med. Tanken er å grovt introdusere flere av konseptene vi skal se nærmere på i de neste kapitlene. Med andre ord: hvis det noe du ikke helt forstår eller syns er vanskelig, så ta det helt med ro, vi skal komme tilbake til det ved et senere tidspunkt og ta det litt grundigere.

1.4.1 Hils på Tom (variabler, funksjoner og tallinjen)

Tom er en strekmann som bor i hus nummer 0 i Tallinjeveien. Tallinjeveien er en gate hvor det er én strekmeter mellom hvert hus. Det også ganske greit å finne frem, siden husene er nummerert monotonisk fra $-\infty$ til ∞; altså til høyre for Tom sitt hus er hus nr. 1, så 2, så 3, etc., mens til venstre er hus nr. -1, så -2, etc. Matematisk kan vi skrive Tallinjeveien som et sett (også kalt mengde) bestående av alle reelle tall, nemlig tallinjen: $\mathbb{R} := (-\infty, \infty)$.

Tom har nylig bygd seg en jetbil som han vil gjøre delvis autonom, og han trenger i den forbindelse vår hjelp til å løse flere grunnleggende regulerings-tekniske problemer.



Figur 1.10: Tom i Tallinjeveien: et sett med enkle reguleringstekniske eksempler.

Symboler og variabler: Vi vil bruke x_p og x_h betegne henholdsvis posisjonen (i strekmeter [m]) og hastigheten (i strekmeter per sekund [m/s]) til jetbilen. Her er x_p et eksempel på en *variabel*, siden det kan ta alle mulige verdier i Tallinjeveien. Dette kan vi skrive som $x_p \in \mathbb{R}$, som betyr «variabelen x_p tar verdier i mengden $\mathbb{R}$».

Funksjoner av tid: På den annen side, så er vi mest interessert i å finne ut hvordan posisjonen (x_p) og/eller hastigheten (x_h) utvikler seg med tiden under påvirkning av jetmotorene. For våre formåle er det derfor mer hensiktsmessig å tenkte på disse som funkjoner av tiden t: $x_p(t)$ og $x_h(t)$. Merk at det f.eks. egentlig er x_p eller $x_p(\cdot)$ som er selve funksjonen, mens $x_p(t)$ betyr verdien denne funksjonen har (altså jetbilens posisjon) ved tiden t. Tiden t varierer selvsagt, men vi kan ikke påvirke den på noen måte, så den er en *uavnhengig variabel*.

1.4.2 Tom har bygd seg jetbil (modellering og differensialligninger)

For å vite hvordan man kan påvirke et fysisk system slik at det gjør slik man ønsker, så er det en stor fordel å kjenne til en matematisk modell av **prosessen** man ønsker å påvirke. Vi kaller utviklingen av en slik modell (f.eks. ved hjelp av fysikkens lover) naturlig nok for **modellering**.

Nåværende mål: Å lage en matematisk modell som beskriver *dynamikken* til jetbilen hans, altså hvordan jetbilens hastighet og posisjon endrer seg over tid når det blir påvirket av kreftene fra jetmotoren.

Denne modellen vi skal lage kan tenkes på som et sett av matematiske «regler», i form av såkalte *differensialligninger*, som vi da kan bruke til å forstå hvordan jetbilen vil utvikle seg med tiden under påvirkning fra jetmotorene. En slik modell er naturlig nok nyttig, siden vi da kan bruke den til å finne ut hva kraften fra jetmotorene skal være for å få systemet til å gjøre som vi vil!

⚠️ **Importante!** Som navnet tilsier, så er en matematisk modell av en prosess akkurat det, en *modell* av prosessen. Med andre ord vil modellene vi lager, uansett hvor nøyaktige og detaljerte vi er, aldri fange opp alle egenskapene og aspektene ved den virkelige prosessen.

Så hva er da vitsen med å utvikle en modell? 🤔 Vel, selv om en matematisk modell aldri er helt nøyaktig, så vil selv en ganske enkel modell som bare grovt fanger visse egenskaper ved prosessen kunne være til stor hjelp når vi skal designe en regulator. Dette er en gjenganger i dette faget:

💡 «*Noe informasjon er som oftest mye bedre enn ingen informasjon!*» 💡

En mer nøyaktig modell vil selvsagt gi oss bedre og mer nøyaktig informasjon som vi kan dra nytte av. Men det vil alltid være et punkt hvor utbyttet av å bruke masse tid, energi og krefter på å utvikle en «bedre» modell blir svært begrenset, i den forstand at vi får nærmest like gode (eller i hvert fall gode nok) resultater med en enklere modell. Man bør derfor alltid finne et passene kompromiss mellom kompleksitet og nytteverdi når man skal utvikle en modell.

Hvordan utvikle en matematiske modell?

Det vi er interessert i nå er å lage en modell som beskriver hvordan hastigheten x endrer seg med tiden under påvirkningen av jetmotorene. Merk at vi ikke skal tenke på hvordan posisjonen til jetbilen endrer seg enda. Det finnes flere måter å utvikle en modell, men den måte vi skal

bruke i dette eksemplet, samt fokusere på i faget for øvrig, er å utlede disse fra lover innen fysikken, som f.eks. Newtons andre lov.

Avgrensninger og antagelser: Vi begynner med å avgrense problemet og å spesifisere antagelsene vi gjør:

- Vi kan måle (uten støy) jetbilens hastighet, x_h, og posisjon, x_p.
- Vekten til jetbilen med Tom i den er $m = 1\,\text{kg}$ og denne endrer seg ikke.
- System kan modelleres ved hjelp av Newtons andre lov.
- Jetmotorene kan umiddelbart påføre en (ubegrenset) kraft F (i Newton (N)) på jetbilen i begge retninger.
- Det virker ingen andre krefter på jetbilen.
- Det er ingen målestøy eller forstyrrelser.

I forhold til variablene i tabell 1.1 har vi da følgende: tilstandene er posisjonen x_p og hastigheten x_h, hvorav begge kan måles ved behov; pådraget er kraften fra jetmotorene, $u = F$; mens det er ingen målestøy eller forstyrrelser, så $w = v = 0$.

Newtons andre lov[3] husker du sikkert fra videregående eller forkurset: massen m ganger akselerasjonen a til et objekt er lik summen av kreftene, $\sum_i F_i$, som virker på det:

$$m \cdot a(t) = \sum_i F_i(t)$$

hvor $\sum_i F_i = F_1 + F_2 + \dots$ betyr at vi summerer alle kreftene. Legger her merke til at akselerasjonen og kreftene er funksjoner av tiden t siden de begge kan endre seg med tiden.

⚠**Pass på enhetene!** Det kan være en god idé å sjekke at enhetene i en ligning «balanserer» hverandre på hver side. Det er også viktig å være klar ovre at det ikke bare er SI-enheter som blir brukt. Selv om feil selvsagt kan skje de beste av oss (se for eksempel denne), så bør man alltid se godt etter i databladet.

Tidsderivat og differensialligning: Du husker kanskje også at tidsderivatet til hastighet ([m/s]) er akselerasjon ([m/s^2]). Vi bruker i dette faget en dott på toppen for indikere at vi snakker om et tidsderivat, såkalt Newton notasjon, altså $\dot{x}_h(t) = \frac{d}{dt}x_h(t)$.

Vi har dermed for jetbilen til Tom at

$$\underbrace{1}_{m} \cdot \underbrace{\dot{x}_h(t)}_{a} = \underbrace{u(t)}_{\sum F}. \tag{1.1}$$

Dette er en såkalt *differensialligning* av første orden. Den sier at endringer i jetbilens hastighet x_h ved tiden t, altså $\dot{x}_h(t)$, tilsvarer kraften $u(t)$ som virker på jetbilen fra jetmotorene ved det tidspunktet. Vi sier at differensialligningen er av første orden siden det bare er ett derivat (en dott) som inngår i den. Merk at jeg også tidsvis kommer til å være litt lat og droppe tidsargumentet, altså f.eks. bare skrive x_h i stedet for $x_h(t)$.

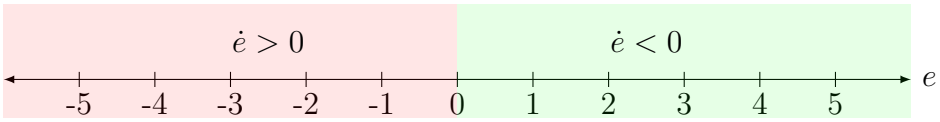
1.4.3 Tom vil ha cruisekontroll (proporsjonal-regulering)

[3]Et annet navn for Newtons andre lov er *kraftbalanse*. De fleste metodene vi skal se på for utlede matematiske modeller i kapittel 3 er såkalte balanselover, som massebalanse, energibalanse, og (rotasjons-)momentbalanse.

Nåværende mål: Designe et aktivt cruisekontroll-system som gjør at jetbilens hastighet x_h holder en konstant ønsket hastighet r (f.eks. $r = 14\,\text{m/s} \approx 50\,\text{km/t}$), altså $x_h \equiv r$.

La oss starte med å oppsummere hvilken informasjon vi har tilgjengelig for å løse dette problemet:

- Referansen/settpunktet: Vi vet at vi ønsker å opprettholde den konstante hastigheten r.
- Måling av hastigheten: Vi kan måle bilens hastighet $y = x_h$.
- Systemets dynamikk: Fra (1.1) har vi at $\dot{y} = \dot{x}_h = u$.



Figur 1.11: Illustrasjon av reguleringsteknikkens kanskje mest fundamentale konsept: hvis du ønsker at en (nøye utvalgt) variabel (eller funksjon), $e \in \mathbb{R}$, skal ha verdien null, sørg for å få absoluttverdien til å synke! Altså at $\frac{d}{dt}|e|^2 < 0$ når $e \neq 0$.

Målet vårt, nemlig at vi skal opprettholde $y \equiv r$, er ekvivalent med å sørge for at **avviket**

$$e = r - y$$

er lik null.[4] Hvis vi dermed kan bruke pådraget u til sørge for at verdien til e går mot null og så forblir der, så har vi løst problemet! Dette tilsvarer følgende, som også er illustrert i figur 1.11:

Veien mot null:[a] For å sørge for at avviket e går mot og så foblir det, så må vi ha at

- Hvis e er positiv ($e > 0$) så må den synke i verdi, altså $\dot{e} < 0$.
- Hvis e er negativ ($e < 0$) så må den øke i verdi, altså $\dot{e} > 0$.
- Hvis e allerede er null ($e = 0$) så skal den ikke endre verdi, altså $\dot{e} = 0$.

[a] Navnet «veien mot null» er bare noe jeg har funnet på for å ha et navn på det. Det er dog en grei måte å tenke på løsningen på mange av problemene vi skal titte på i disse notatene.

Med andre ord: Hvis Toms jetbil kjører raskere enn ønsket så må den redusere farten, mens hvis den kjører saktere enn ønsket så må den øke hastigheten. Mer komplisert er det ikke!

Men hvordan kan vi velge u slik at dette holder? 🤔 Jo, vi kan starte med å finne $\dot{e}$:

$$\dot{e}(t) = \frac{d}{dt}e(t) = \frac{d}{dt}\left(r - y(t)\right) = \underbrace{\dot{r}}_{=0} - \underbrace{\dot{y}}_{=\dot{x}_h} = -u$$

hvor jeg har brukt at $\frac{d}{dt}r = 0$ siden r er konstant og at $\dot{x}_h = u$. Vi kan derfor sørge for at «veien mot null» holder ved å velge u slik at

1) hvis $e = 0$ så $u = 0$,

[4] I stedet for å definere avviket slik som i figur 1.8, altså $e = r - y$, så er også $e = y - r$ mye brukt siden man da har et positivt avvik når y er større enn r, og et negativt når y er mindre enn r.

2) hvis $e \neq 0$ så $e \cdot u > 0$.

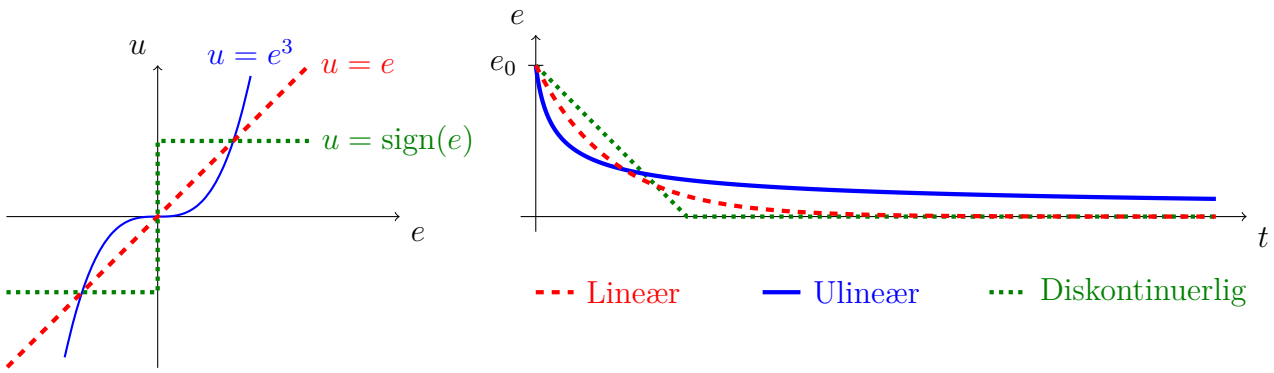
Dette kan vi også lett utlede ved å se at veien mot null tilsvarer at hvis funksjonen $V = \frac{1}{2}e^2$, som alltid er positiv, tilfredsstiller $\dot{V} < 0$ når $e \neq= 0$, altså synker i verdi, så må dette bety at e går mot null! Ved hjelp av kjerneregelen (evt. produktregelen) har vi:

$$\dot{V}(t) = \frac{d}{dt}V(t) = \frac{d}{dt}\left(\frac{e(t)^2}{2}\right) = e(t) \cdot \dot{e}(t) = e(t) \cdot (-u(t)).$$

Fun facts, bemerkninger og annet dill dall (you may skip)

En alltid-positiv funksjon, slik som $V = \frac{1}{2}e^2$, med den egenskapen at hvis den alltid synker i verdi så går avviket mot null, kalles for en Lyapunov funksjon. Det er et nyttig og mye brukt konsept innen viderekommende reguleringsteknikk. Ideen i seg selv er dog ikke veldig avansert og er temmelig lik «veien mot null». Å finne en slik funksjon er dog ofte utfordrende.

Flere kandidater: Det er her ikke mangel på kandidater for u som tilfredsstiller disse to punktene; faktisk så finnes det uendelig mange alternativer!



Figur 1.12: Illustrasjon av forskjellige regulator-kandidater for cruisekontroll til Toms jetbil og tilsvarende respons.

Anta at vi starter med avviket $e(0) = e_0$. For en eller annen $b > 0$, så vil for eksempel følgende kandidater alle gjøre jobben:

- **Lineær:** $u = b \cdot e \quad \overset{\text{eksponentielt stabilt}}{\Longrightarrow} \quad e(t) = e_0 \exp(-bt)$
- **Ulineær:** $u = b \cdot e^3 \quad \overset{\text{asymptotisk stabilt}}{\Longrightarrow} \quad e(t) = e_0/\sqrt{2be_0^2 t + 1}$
- **Diskontinuerlig:** $u = b \cdot \text{sign}(e) \quad \overset{\text{endelig-tid stabilt}}{\Longrightarrow} \quad |e(t)| = |e_0| - bt$

Her er $\exp(\cdot)$ eksponentialfunksjonen (bruker altså $\exp(x)$ i stedet for e^x for å unngå misforståelser med avviket e), mens sign er fortegnsfunksjonen.

Her nevnes også termer som «lineær» og «ulineær», samt typer «stabilitet». Dette er illustrert i figur 1.12.

Den lineære P-regulatoren: For kandidatene over ble også tilsvarende løsning (altså $e(t)$) vist, samt hva slags type *stabilitet* dette tilsvarer. Dette er ting vi skal komme tilbake til i senere kapitler. For vår del nå, så er det den lineære reguleringsstrategien, altså

$$u_P = k_P \cdot e = k_P \cdot (r - y) \qquad \text{(P-regulator)}$$

hvor vi nå har tatt $k_P = b$, som er av mest interesse. Dette kalles for en **proporsjonal-regulator** (evt. «P-regulator»), og er den kanskje viktigste regulatoren i reguleringsteknikken.

Del I

Dynamiske Reguleringssystemer

2. Dynamiske systemer

Automatisering, og reguleringsteknikk spesielt, handler i stor grad om å forstå hvordan vi kan påvirke fysiske prosesser slik at vi oppnår et ønsket resultat. Det kan for eksempel være å få en robotarm til å bevege seg langs en spesifikk bane, få bilens cruisekontroll til å opprettholde en ønsket hastighet, eller holde inflasjonen i en økonomi nær en spesifisert verdi.

Det matematiske verktøyet vi vil bruke for å modellere slike prosesser er **dynamiske systemer**, hovedsakelig representert ved hjelp av **differensialligninger**.

I dette kapittelet skal vi derfor se på hva et dynamisk system er, introdusere enkle og mye brukte eksempler på slike systemer, utforske hvordan vi kan analysere disse systemene og deres respons, introdusere konseptet stabilitet, samt se på alternative representasjonsformer som blokkdiagrammer og overføringsfunksjoner. Vi starter dog med å bygge den matematiske basen, slik som funksjoner og derivering, slik at denne er god og solid før vi beveger oss videre.

⚠️ **Heavy s#!@ incoming:** Dette kapittelet kan nok tidvis virke temmelig abstrakt og vanskelig (tidvis helt gresk, bokstavelig talt) sånn helt i starten. I stedet for å bruke masse tid nå, så anbefales det derfor at du kommer tilbake ved behov utover semesteret (og kanskje også senere år) for en liten oppfrisker. Husk: Alt er vanskelig inntil du kan det! 🤗

2.1. Den Matematiske basen*

Så godt som alle temaene vi vil se på i disse notatene—inkludert reguleringsteknikk, robotikk og maskinlæring—er beskrevet ved hjelp av matematiske metoder. Noe av det meste fundamentale konseptet her er *funksjoner*. Målet med denne seksjoner (og de neste) er derfor at denne fundamentale matematiske basen skal være nogenlunde solid før vi beveger oss videre i den vakre automatiseringsverden. Men først litt motivasjon for hvorfor dette er så viktig:

Fun facts, bemerkninger og annet dill dall (you may skip)

Hvorfor er den matematiske basen så viktig? Du vil i dette faget få se en magisk ting, nemlig hvordan man kan gå fra noe breskrevet ved matematiske metoder til å få noe til å fysisk bevege seg på en ønsket måte i den virkelige verden. Med andre ord: hvis du mener matte er kjedelig[a], så er mitt mål at du skal ha endret mening i løpet av dette semesteret. Skulle dette dog ikke dette være nok til å overbevise deg, så er det nå en gang sånn at matematikk, kort og greit, er et språk en ingeniør må mestre (i hvert fall til en viss grad):

Ingeniørarbeid uten matematikk

Følgende liste illustrer blant annet nytteverdien i forhold til reguleringsteknikk:

- **Kraften i abstraksjon:** Matematikk tillater oss å abstrahere bort komplekse eller unødvendige detaljer i et system og fokusere på dets essensielle atferd. Denne abstraksjonen er det som gjør reguleringsteknikk både mulig og nyttig, da det tillater oss å modellere komplekse systemer ved hjelp av enkle matematiske verktøy.
- **Praktiske anvendelser:** Matematiske funksjoner brukes til å beskrive fysiske størrelser vi ønsker å styre i virkelige systemer, f.eks. hastigheten til en motor eller temperaturen i en ovn, mens differensialligninger beskriver hvordan disse størrelsene vil utvikle seg over tid og hvordan vi kan påvirke (regulere) dem for å få ønsket oppførsel.
- **Elegansen:** Det er en skjønnhet i enkelheten og elegansen til matematiske funksjoner. For eksempel, så dukker basisfunksjoner som eksponentiell-, sinus- og cosinus-funksjonene opp over alt innen reguleringsteknikk (og de fleste ingeniørfaglige felt for øvrig).
- **Problemløsning:** Matematikk er et grunnleggende verktøy for å løse problemer, og da ikke bare i reguleringsteknikk, men også robotikk, maskinlæring og sekvens-styring. Ved å mestre de matematiske metodene i dette kapitelet, vil du utvikle verdifulle problemløsningsevner som vil tjene deg godt i din fremtidig karriere.

[a] Det er mange som sier de ikke liker eller er dårlig i matematikk. Men det er som Edward Frenkel sier så godt i denne YouTube-videoen (Yexc19j3TjE): Man blir neppe glad eller flink i kunst ved å bare male et gjerde, uten å noen gang få se verkene til de store kunstnerne som f.eks. Munch og Leonardo da Vinci.

2.1.1 Funksjoner

Alternative kilder: Khan Academy (kvGsIo1TmsM)

Du vil lære mer om dette i IMAT1002 - Matematikk for ingeniørfag 1.

I mest mulig grove trekk, så kan en matematisk funksjon beskrives som følger: For en hver gyldig verdi den tar inn (dets *argument*), så spytter ut en tilsvarende unik verdi. Et typisk eksempel er $f(x) = 2x$, hvor da argumentet er x og funksjonens verdi er to ganger argumentet.

Merk: når vi snakker om selve funksjonen, så bør vi helst bruke symbolene f eller eventuelt $f(\cdot)$, og ikke $f(x)$ som (for å være litt nøyen på det) betyr verdien til funksjonen f ved x.[1]

En typisk måte å representere en funksjon på er følgende ligningsrepresentasjon:

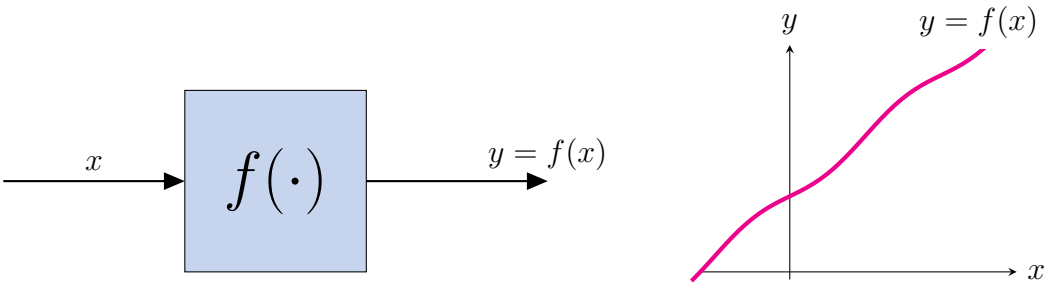
$$y = f(x).$$

Her er x den *uavhengige variabelen*, mens y er den *avhengige variabelen* (den avhenger jo av hva x er). Man kan tenke på denne ligningen som et inngang-utgang-system som vist i figur 2.1 **a)**, hvor utgangsverdien y avhenger av hva funksjonen f er for den gitte inngangen x.

Fun facts, bemerkninger og annet dill dall (you may skip)

Noen funksjoner $f(\cdot)$ har også en såkalt inversfunksjon, $f^{-1}(\cdot)$, med den egenskap at $f^{-1}\big(f(x)\big) = x$. Jeg anbefaler sterkt at du ser denne illustrasjonen av en inversfunksjon.

Funksjoner av tid: I dette faget er vi spesielt interessert i funksjoner av tid, siden vi ofte er interessert i å se og forklare hvordan størrelser som posisjoner, hastigheter og forstyrrelser utvikler seg over tid. Vi bruker da t til å betegne tiden, som da er den uavhengige variabelen. Innen reguleringsteknikken bruker vi f.eks. tidsvarierende funksjonene i tabell 1.1, som tilstanden $x(t)$, pådraget $u(t)$ og målingen/utgangen $y(t)$, etc.



a) Funksjon **b)** Graf

Figur 2.1: Illustrasjon av en matematisk funksjon: **a)** som et inngang-utgang-system, og **b)** ved hjelp av en graf.

Funksjoner i MATLAB

Følgende kodesnutt viser to metoder for hvordan man kan definere enkle funksjoner i MATLAB:

[1] Det går også an å si funksjonen $f(x) = 2x$, mens en litt mer eksotisk variant for samme funksjon er $x \mapsto 2x$.

Kodesnutt 2.1: To metoder for å implementere funksjoner i MATLAB..

```
% Metode 1
kvadratM1 = @(x) x.^2;
kvadratM1(4)
% Metode 2
kvadratM2(4)
function f=kvadratM2(x)
% Funksjonen må defineres i egen .m-fil eller
% i slutten av et script.
    f=x.^2;
end
```

2.1.2 Graf og koordinatsystemer

Det finnes andre måter å representere en funksjon på enn bare ligningsrepresentasjonen: som en numerisk tabell som relaterer inngang- og utgangsverdiene, som et blokkdiagram (se § 2.4), eller til og med bare ved hjelp av ord. En annen måte å representere en funksjon er ved hjelp av dets **graf** (altså en *grafisk* metode).

Grafen til en vilkårlig funksjon f er vist figur 2.1 **b)**, hvor vi måler x om den horisontale aksen, og y om den vertikale. Dette gir oss et **koordinatsystem**, med **koordinater** x (målt langs den horisontale aksen) og y målt (om den vertikale). Grafen til funksjonen f tilsvarer dermed alle par av punkter (x, y) som tilfredsstiller ligningen $y = f(x)$. Hvis vi for eksempel har en partikkel som kan bevege seg i (x, y)-planet, så kan man tenke på funksjonen f som en regel som bestemmer en sti denne partikkelen *må* følge.

Fun facts, bemerkninger og annet dill dall (you may skip)

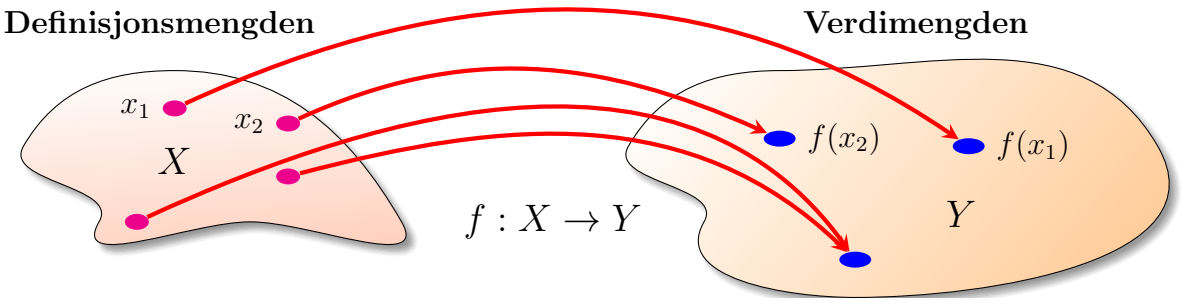
Graf = funksjon? Alle funksjonene vi er interessert i i dette faget har en graf, men det motsatte er ikke nødvendigvis alltid sant. Altså, en gitt graf trenger ikke tilfredsstille én enkelt (unik) funksjon. Et eksempel på dette er enhetssirkelen, som tilsvarer ligningen

$$x^2 + y^2 = 1.$$

Legg merke til at for en gitt x, så vil både $y = f_+(x) = \sqrt{1 - x^2}$ og $y = f_-(x) = -\sqrt{1 - x^2}$ tilfredsstille denne ligningen. På den annen side, så kan kan vi i stedet definere en **multivariabel funksjon**, $h(x, y) = x^2 + y^2 - 1$, slik at enhetssirkelen tilsvarer **nivåkurven** $h(x, y) = 0$.

2.1.3 Definisjonsmengden, verdimengden, og tallinjen

Hva skjerm med verdien til $y = f(x) = \frac{1}{x}$ når verdien til x går mot null (i matematisk notasjon: $x \to 0$, hvor pilen betyr "går mot")? Hvis vi går mot null fra den positive siden ($x \to 0^+$), så får vi at $y \to \infty$, mens fra den negative siden ($x \to 0^-$) får vi $y \to -\infty$. Dette betyr at funksjonen $f(x) = \frac{1}{x}$ ikke er definert ved verdien $x = 0$. Den er dog velldefinert for alle verdier mellom $-\infty$ og opptil med ikke med 0, samt for alle verdier større enn null. Dette kan vi skrive som et såkalt sett/mengde $X = (-\infty, 0) \cup (0, \infty)$, hvor $\cup$ betyr unionen av de to intervallene, mens notasjonen $(a, b]$ betyr alle tall fra a til og med b, men *uten* a selv.



Figur 2.2: En funksjon $f : X \to Y$ tar en verdi x i sin definisjonsmengde X (som er et sett), og «overfører» det til en unik verdi $y = f(x)$ i verdimengden Y (som også er et sett).

For en hver funksjon f følger det derfor med to matematiske mengder/sett:

- **Definisjonsmengden:** Settet av alle mulige argumenter slik at f er velldefinert;
- **Verdimengden:** Settet av alle verdiene f kan ta for argumenter i definisjonsmengden.

En svært viktig mengde er tallinjen (husk Tom i *Tallinje*-veien):

Tallinjen: $\mathbb{R} = (-\infty, \infty)$, altså settet av alle mulige reelle tall mellom $-\infty$ til ∞.

Vi bruker altså $\mathbb{R}$ som et matematisk symbol på dette settet. For eksempel, så betyr $x \in \mathbb{R}$ at *variabelen* x tar verdier ($\in$ betyr "tar verdier i") som ligger på tallinjen. Vi bruk også $\mathbb{R}^2$ og $\mathbb{R}^3$ til henholdsvis det reelle (2D) *planet* og (3D) *rommet*, som er såkalte Euklidiske rom.

Mengder og funksjoner kan dog også være mer abstrakte enn bare tall:

Eksempel 2.1. La $f_{🐘} : E \to N$ hvor E er en mengde bestående av flere elefanter, $E = \{\text{elefant 1, elefant 2, elefant 3,}\dots\}$, mens N er et sett av navn, $N = \{$ Per, Ola, Gudlaug, Bjørg, $\dots\}$. Merk at vi bruker krøllparenteser for mengder av slike diskrete objekter.

Vi har da f.eks. $f_{🐘}(\text{elefant 101}) = \text{Lars Rune}$ og $f_{🐘}(\text{elefant 23}) = f(\text{elefant 91}) = \text{Gunda}$. Altså så returnerer funksjonen f navnene til elefantene.

2.1.4 Derivater og tangentlinjer

Derivatet til en funksjon f (hvis det eksisterer) er en funksjon f' som tilfredsstiller følgende for alle x i f sin definisjonsmengde X:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}. \tag{2.1}$$

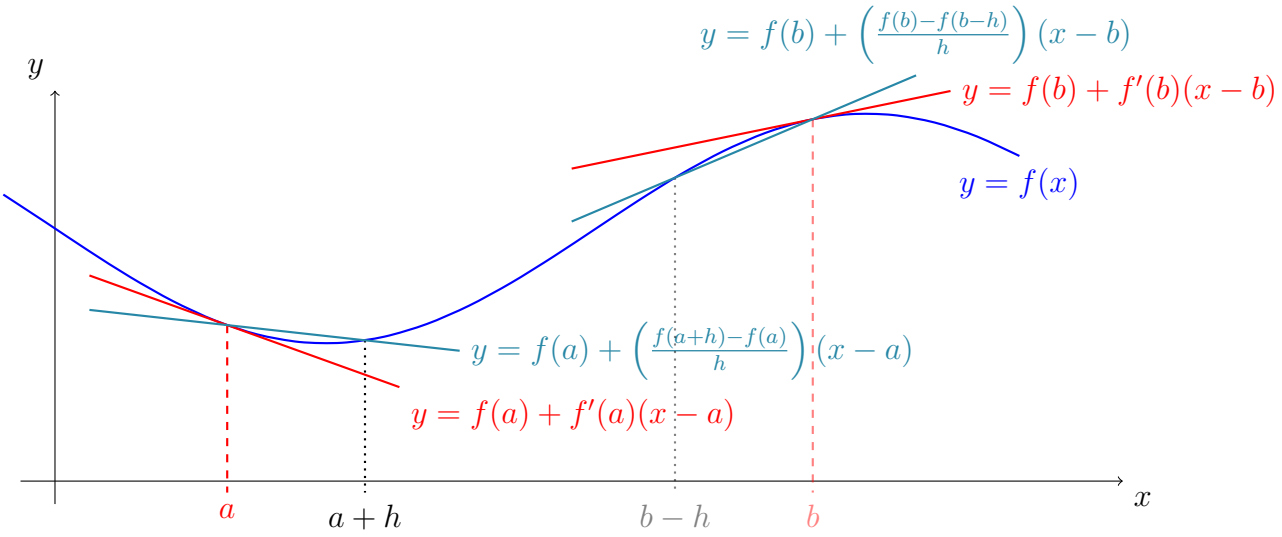
Det er viktig å poengtere dette at f' er en helt ny funksjon, hvor anførselstegnet " ' " brukes til å indikere at denne funksjonen f' har den egenskap at den tilfredsstiller (2.1) og dermed er derivatet til f.

Notasjon: Bruk av anførselstegn, altså f', til å indikere derivatet til en funksjon f kalles for Larange-notasjon. En dott over funksjonen, altså $\dot{f}$, er Newtons notasjon som vi reserverer til funksjoner av tid. Det finnes også en mye brukt operator-basert notasjon, kalt Leibniz-notasjon, for derivatet:

$$\frac{dy}{dx} = \frac{d}{dx}f(x) = \frac{df}{dx}(x) = f'(x)$$

hvor da $y = f(x)$. Man kan intuitivt tenke på dy som en liten endring i y og dx som en liten endringe i x. Ved å multiplisere med dx på begge sider får man $dy = f'(x)dx$, noe som kan tolkes som at en liten endring, dx, fra et punkt x, relateres til en endring, dy, i y via $f'(x)$.

Derivatet definerer tagnentlinjer: En meget nyttig ting med den deriverte av en funksjon er at ved et gitt punkt definerer derivatet hellingen av tangentlinjen til funksjonen ved det punktet. Ta for eksempel $y = f(x)$. Derivatet av denne funksjonen er en ny funksjon, $f'(x)$,



Figur 2.3: Illustrasjon av hvordan derivatet $f'(\cdot)$ til en funksjon f kan brukes til å generere tangentlinjer ved gitte punkter.

som gir hellingen av tangentlinjen til $f(x)$ for hver x-verdi.

Anta at du har en kurve som representerer funksjonen $f(x)$, og du velger et bestemt punkt på denne kurven, si $(a, f(a))$. Tangentlinjen til kurven på dette punktet er en rett linje som «berører» kurven bare på det punktet, og følger dens generelle retning på det punktet slik som det er illustrert i figur 2.3.

Hellingen til denne linjen er definert som endringen i y-verdien (vertikal retning) delt på endringen i x-verdien (horisontal retning). Dette er nøyaktig det derivatet av funksjonen $f(x)$ gir oss ved punktet a, som er $f'(a)$. Tangentlinjen til funksjonen f ved punktet a kan da beskrives med likningen:

$$y = f(a) + f'(a)(x - a).$$

Her er $f(a)$ y-verdien ved punktet hvor tangentlinjen berører kurven, $f'(a)$ er hellingen til tangentlinjen, og $(x - a)$ representerer enhver x-verdi langs tangentlinjen fra punktet a.

Merk at dette er en meget viktig egenskap, siden det blant annet lar oss *linearisere* ulineære systemer. Dette skal vi se nærmere på i kapittel 7. Derivatet er også knyttet til et konsept for

funksjoner med flere variabler som kalles for funksjonens gradient, noe som er ekstremt viktig når det kommer til optimering og maskinlæring.

2.2. Hva er et dynamisk system?

Alternative kilder: Wikipedia; SNL.

La oss begynne med følgende definisjon, som er en litt modifisert forenkling av teksten fra Wikipedia:

Et **dynamisk system** er et system av matematiske ligninger (f.eks. differensialligninger) som beskriver hvordan systemets (indre) tilstandser, som er punkter i systemets tilstandsrom, utvikler seg over tid (deres evolusjon).

Eksempler inkluderer de matematiske modellene som beskriver svingningene av en pendel, strømmen av vann i et rør, bevegelsene til en bil, eller hvordan været utvikler seg.

Til enhver tid har et dynamisk system en tilstand som representerer et punkt i et passende tilstandsrom. Evolusjonsregelen for det dynamiske systemet er en funksjon som beskriver hvilke fremtidige tilstander som følger av den nåværende tilstanden.

Et dynamisk system består av tre ting: dets (interne) tilstander, ytre påvirkninger, og et sett med regler som forklarer hvordan systemets tilstander (dynamisk) vil endre seg med tiden:

Tilstandene til systemet er de variablene man ved et hvert tidspunkt må vite for å 1) kjenne systemets daværende tilstand (duh!), og 2) forklare hvordan disse tilstandene vil utvikle seg med tiden hvis det ikke er noen ytre påvirkninger. Ta for eksempel et tog: Hvis du vil vite hva posisjonen til toget er fra et gitt punkt noen sekunder frem i tid, så er det ikke nok å bare vite nåværende posisjon (tilstand nr. 1), du må også vite togets hastighet (tilstand nr. 2).

Ytre påvirkninger har en effekt på hvordan systemet endrer seg, men er ikke en intern tilstand til systemet. Slike påvirkninger er typisk pådragsorgan, som moteren til et tog eller en væskestrøm inn i en tank. Det kan også være forstyrrelser, som for eksempel en lekkasje i en tank eller vind som påvirker toget.

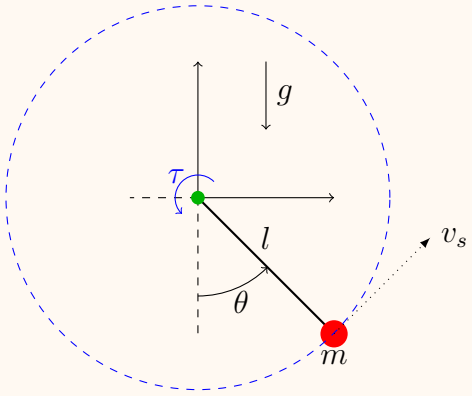
Reglene som bestemmer hvordan tilstandene utvikler seg med tiden kan for eksempel være differensialligninger (2.3), differensligninger (§ 2.10) eller overføringsfunksjoner (§ 2.8). Selv om vi vil se litt på alle representasjonsformene i disse notatene, så er det *differensialligninger* vi vil fokusere mest på.

🇬🇷 **Helt gresk:** De greske bokstavene θ («theta») og τ («tau») dukker opp i det neste eksempelet. Det greske alfabetet er mye brukt på universitet, så det er bare å venne seg til og bli komfortabel med det. Du finner en liste i § A.1; eventuelt kan du se en catchy YouTube-video (3gaeIUsPJ-Y).

Eksempel 2.2. En pendel

En pendel, som vist i figuren under, er et av de enkleste eksemplene på et dynamisk system. Pendelen antas å kunne rotere fritt om et feste (grønn sirkel i figuren), og vi bruker θ (målt

i radianer) til å betegne vinkelen pendelen har rotert fra den nedre, vertikale linjen målt positivt mot klokken. Pendelen antas også å ha en konstant lengde l (målt i meter), slik at enden på pendelen, hvor vi antar all massen til systemet er plassert (se rødfarget sirkel), kun kan bevegene se på den blå, stiplede sirkelen. Den umiddelbare hastigheten til pendelenden er dermed $v_s = l \cdot \dot{\theta}$ ([m/s]), hvor $\dot{\theta}$ er vinkelhastigheten (([rad/s]).



Hvis det virker et deremomentet ved festet, f.eks. fra en motor, så vil **bevegelsesligningene** som bestemmer systemets **dynamikk** være følgende andre-ordens **differensialligning**:

$$\ddot{\theta}(t) = -\frac{g}{l}\sin(\theta) + \frac{\tau}{ml^2}.$$

Her betenger g akselerasjonen fra gravitasjonen til jorden. For dette systemet er dermed **tilstandene** θ og $\dot{\theta}$, mens τ er en ytre påvirkning. Legg også merke til at høyresiden ikke er lineær i variabelen θ, slik at dette faktisk er et *ulineært* dynamisk system.

2.3. Hva er en differensialligning?

Alternative kilder: Wikipedia.

Du vil lære mer om dette i IMAT1002 - Matematikk for ingeniørfag 1.

En differensialligning (diff.ligning) er kort og greit en ligning som gir et gitt forhold mellom derivatene til en funksjon. For eksempel, så er

$$\dot{x}(t) = f(x(t)) \tag{2.2}$$

en **tidsinvariant**, **første-ordens** differensialligning på såkalt **tilsdandsromform** (mer om hva alle disse kule ordene betyr om litt). Dette er også en *ordinær* differensialligning, ofte abbreviert ODE pga. det engelske «ordinary differential equation».

Merk: jeg vil ofte, for enkelhets skyld, droppe t-argumentet og bare skrive $\dot{x} = f(x)$.

Hva betyr en slik ligning?

Betydningen av en slik ligninger er som følger: Vi leter etter en funksjon (av tid) $x(t)$ som har har et tidsderivat, $\dot{x}(t)$, som tilfredsstiller denne ligningen. Tilstanden til det tilsvarende dynamiske systemet ved tiden t er dermed $x(t)$, mens $f(x(t))$ tilsvarer den umiddelbare endringen ("hastigheten") til tilstanden ved tiden t.

En en annen måte å tenke på ligning er som følger: For en hver $x(t)$ kan man finne ut hvordan funksjonen $x(\cdot)$ endrer seg akkurat ved tidspunktet t. Dette kan vi se bedre ved å representere ligningen på en annen vanlig form:

$$\frac{dx}{dt} = f(x(t)).$$

Her kan man intuitivt tenke på *differensialene* (derav *differensial*-ligning), dx og dt, som knøttsmå endringer i henholdsvis x og t fra verdiene $x(t)$ og t. Ved å multiplisere med dt på begge sider og så integrere fra tiden t_0 til t, så finner vi at løsningen på denne differensialligningen er gitt ved

$$x(t) - x(t_0) = \int_{t_0}^{t} f(x(s))ds. \qquad \text{(Løsningen til en første-ordens differensialligning)}$$

Ofte vet man hva **initialverdien** (også kalt startbetingelsene) $x(t_0) = x_0$ er; altså at man vet hvor systemet starter fra. Problemet hvor man ønsker å finne løsningen på differensialligningen fra den gitte initialverdien er da (utrolig nok) kalt et **initialverdiproblem**.

Achtung! Selv om man vet både initialverdien x_0 og funksjonen f, så kan vi som oftest bare approksimere løsningen numerisk (mer om dette senere i notatene). Det finnes dog noen unntak, deriblant når systemet er lineært.

Lysten på et enkelt eksempel? Da kan du f.eks. hoppe tilbake til seksjon 1.4.2.

Mer om linearitet, orden og tilstandsromform

Vi sier at $\dot{x}(t) = f(x(t))$ er en **første-ordens** differensialligning siden det bare inngår ett derivat i ligningen. Siden dette derivatet er isolert på venstresiden, så sier vi at ligningen er på **tilstandsromform** siden det forteller oss direkte hvordan systemets tilstands vil endre/utvikle seg i sitt **tilstandsrom** (altså rommet av alle verdien den kan ta) ved tiden t.

Ligningen $\dot{x}(t) = f(x(t))$ er også **tidsinvariant** (noen ganger også kalt **autonom**) siden høyresiden ikke direkte avhenger av tiden t, bare indirekte via $x(t)$. Vi sier også at systemet er **lineært** hvis funksjonen f er lineær, for eksempel $f(x) = 2x$.

Følgende er et eksempel på en differensialligning av andre orden som hverken er tidsinvariant eller er på tilstandsromform:

$$\ddot{x}(t) = f(t, x(t), \dot{x}(t)). \quad \text{(Tidsavhengig andre-ordens diff.ligning (ikke på tilsdandsromform))}$$

Funksjonen f avhenger av tiden t via sitt første argument, så den er tidsvarierende, ikke tidsinvariant. Den er en andre-ordens differensialligning siden derivatet av høyeste orden, nemlig $\ddot{x}(t)$, er av orden to (dobbel-dott). Den er heller ikke på tilstandsromform siden det ikke er gitt som et såkalt *system av første-ordens differensialligninger*. Som vist under, så er det dog mulig å skrive den om på en slik form.

Tilstandsromform, tilstandsrommet og faseplanet

For andre-orden systemet over, la oss definere[2] tilstandsvariablene/tilstandene

$$x_1 \coloneqq x \quad \text{og} \quad x_2 \coloneqq \dot{x}.$$

Vi kan da representere systemet på tilstandsromform, nemlig som følgende system av første-ordens differensialligninger:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(t, x_1, x_2). \end{aligned} \qquad \text{(Andre-ordens system representert på tilstandsromform)}$$

Denne får vi ved å se at $\dot{x} = \dot{x}_1 = x_2$ og $\ddot{x} = \ddot{x}_1 = \frac{d}{dt}\dot{x}_1 = \dot{x}_2$, samt at $f(t, x, \dot{x}) = f(t, x_1, x_2)$.

Men hvorfor kaller vi denne formen for tilstandsromform? 🤔 Grunnen er av vi har isolert derivatene av hver av tilstandsvariablene, x_1 og x_2, i hver sin ligning, slik at vi kan se på deres momentære endringer individuelt.

Likevektspunkter

Et **likevektspunkt** (også en sjelden gang kalt et stasjonærpunkt) til en tidsinvariant differensialligning $\dot{x} = f(x)$ er et punkt $\bar{x}$ slik at $f(\bar{x}) = 0$. Det betyr at tilstanden x til systemet ikke vil endre seg ved et slik punkt siden man der har at $\dot{x} = 0$. Med andre ord, hvis et system starter i et likevektpunkt, så vil det forbli i det punktet for all fremtid.

2.3.1 Generelle ordinære differensialligninger (ODEer)*

En meget generell form til en differensialligning av orden n er følgende *implisitte form*:

$$F(x(t), \dot{x}(t), \ddot{x}(t), \ldots, \frac{d^n}{dt^n}x(t)) = 0$$

hvor F da er en multivariabel funksjon. Når en differensialligninger avhenger bare av ordinære derivater av én gitt funksjon, så kalles den for *ordinær differensialligning* (abbreviert ODE pga. engelske "equation"). Som regel kan vi isolere derivatet av høyeste ordenen, slik at man kan skrive ligningen (ODEen) på følgende *eksplisitte form*:

$$\frac{d^n}{dt^n}x(t) = f(x(t), \dot{x}(t), \ddot{x}(t), \ldots, \frac{d^{n-1}}{dt^{n-1}}x(t)).$$

I motsetning til den implisitte formen, så kan man ved å introdusere tilstandsvariablene $x_1 = x$, $x_2 = \dot{x}$, $\ldots$, $x_n = \frac{d^n}{dt^n}x(t)$, alltid skrive systemet om på tilstandsromform:

$$\dot{x}_1 = x_2, \tag{2.3a}$$
$$\dot{x}_2 = x_3, \tag{2.3b}$$
$$\vdots \tag{2.3c}$$
$$\dot{x}_n = f(x_1, x_2, \ldots, x_{n-1}). \tag{2.3d}$$

NB! Hvis du synes dette er drøyt, så ta det helt med ro, vi vil for det meste holde oss til første- og andre-ordens differensialligninger i dette faget.

[2] Notasjonen $a \coloneqq 2b$ betyr at a er definert som to ganger b. Både $a \triangleq 2b$ og $a \overset{\text{def}}{=} 2b$ brukes også.

2.3.2 Lineære, tidsinvariante differensialligninger

Første-ordens LTI systemer

Lineære og tidsinvariant differensialligninger (ofte abbreviert LTI) er en viktig klasse av systemer innen reguleringsteknikken. Et første-ordens LTI systemer er har følgende form:

$$\dot{x}(t) = -ax(t), \tag{2.4}$$

hvor a er et konstant tall ($a \in \mathbb{R}$). Som tidligere nevnt, så er den **lineær** fordi høyresiden avhenger lineært av $x(t)$, mens den sies å være **tidsinvariant** siden høyresiden bare indirekte avhenger av tiden t via funksjonen $x(t)$.

Eksempler på ulineære differensialligninger er $\dot{x} = ax^3$ ellr $\dot{x} = a\sin(x)$.
Eksempler på tidsvarierende differensialligninger er $\dot{x} = (1+t)x$ eller $\dot{x} = -t^2$.

Løsning til en første-ordens differensialligning

Løsning: Anta at $x(0) = x_0$. Løsningen på differensialligningen (2.4) er da $x(t) = x_0 e^{-at}$.

La oss utlede dette. Vi begynner med å skrive (2.4) på følgende form

$$\dot{x} = -ax \quad \Longrightarrow \quad \frac{1}{x}\dot{x} = -a$$

Merk at symbolet $\Longrightarrow$ betyr «impliserer». Vi kan da integrere med hensyn på tid på begge sider. Vi starter med venstresiden, altså leddet $\frac{1}{x}\dot{x}$:[3]

$$\int_0^t \frac{1}{x(s)}\dot{x}(s)ds = \Big[\ln(|x(s)|)\Big]_0^t = \ln(x(t)) - \ln(x_0) = ln\left(\frac{x(t)}{x_0}\right).$$

Merk at jeg her i det første steget har brukt (kjerneregelen), hvorfra vi har at $\frac{d}{dt}\big(\ln(x(t)) = \big(\frac{d}{dx}\ln(x)\big)\cdot\frac{d}{dt}x(t) = \frac{1}{x(t)}\dot{x}(t)$. Merk også at vi «fjerner» absoluttverdiene, altså $|\cdot|$, i den naturlige logaritmen siden $x(t)$ og x_0 må ha samme fortegn (det vil vi se om litt).

Ved å så integrere høyresiden finner vi at

$$\int_0^t -a\cdot ds = [-a\cdot s]_0^t = -at.$$

For å fjerne den naturlige logaritmen i leddet i midten gjør vi følgende:

$$e^{\ln(x(t))-\ln(x_0)} = e^{\ln\left(\frac{x(t)}{x_0}\right)} = \frac{x(t)}{x_0} = e^{-at} \quad \Longrightarrow \quad x(t) = x_0 e^{-at},$$

altså vi høynet Eulers konstant e i begge sidene, og brukte at $e^{\ln(y)} = y$.

⚠**Viktig!** I stedet for e^{at} og lignende, vil vi ofte i stedet skrive $\exp(at)$, hvor $\exp(\cdot)$ er eksponentialfunksjonen. Dette er for å unngå notasjonsrot, siden e også brukes til å betegne

[3] Legg merke til at jeg bruker en ny, midlertidig integrasjonsvariabel, s, i integranden siden vi integrerer til tiden t.

et avvik (se tabell 1.1).

Hvor fort $x(t)$ endrer seg vekk fra x_0 avhenger av tallet $\lambda = -a$, som har følgende navn:

Egenverdi: $\lambda = -a$ kalles for egenverdien til (2.4).

Fun facts, bemerkninger og annet dill dall (you may skip)

Gitt et lineært dynamisk system på *matriseform*,

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$$

hvor $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\mathsf{T}$ er en kolonnevektor i tilstandsrommet $\mathbb{R}^n$ og $\mathbf{A}$ er en $n \times n$ matrise. Alle løsningene til et slikt systemt er bygd opp av egenvektorene og egenverdiene til systemet. Løsningen til et slik system er $\mathbf{x}(t) = \mathbf{x}_0\mathbf{exp}(\mathbf{A}t)$ hvor **exp** er matriseeksponentialfunksjonen.

Affine og inhomogene første-ordens differensialligninger:

Noen ganger har man differensialligninger med et konstant ledd, slik som

$$\dot{y} = -ay + b \tag{2.5}$$

hvor $b \in \mathbb{R}$ er en konstant. Slike systemer, med en lineær del gitt av $-ax$ og en konstant del gitt av b, kalles **affine**. Det konstante leddet kan for eksempel tilsvare en konstant lekkasje i en tank eller at systemet jobber mot tyngdekraften, slik som en bil som kjører opp (eller ned) en bakke med konstant helning.

Legger her merke til at den umiddelbare endringen i $y(t)$ ved tiden t, altså dens tidsderivat $\dot{y}(t)$, er lik null når $y = \bar{y} = b/a$, siden $\dot{y} = -a\bar{y} + b = -a\frac{b}{a} + b = -b + b = 0$. Med andre ord . er $\bar{y} = b/a$ systemets eneste likevektspunkt .

Merk at vi kan alltid skrive en differensialligningen på denne formen om til formen (2.4) ved å definere en ny variabel som har sitt nullpunkt i likevektspunktet:

$$x = y - \bar{y} = y - \frac{b}{a},$$

som da også tilsvarer $y = x + \frac{b}{a}$. La oss finne tidsderivatet til den nye variabelen x:

$$\dot{x} = \dot{y} - \underbrace{\frac{d}{dt}\bar{y}}_{=0} = -ay + b = -a\big(x + \frac{b}{a}\big) + b = -ax \qquad \Longrightarrow \qquad \dot{x} = -ax,$$

altså på formen (2.4) slik som jeg hevdet.

Men hvorfor har vi da et eget navn for en ligning på formen (2.5)? 🤔 Grunnen er bare at likevektspunktet for slike systemer, i motsetning til for lineære systemer, ikke er nullpunktet.

Hva med løsningen på den affine ligning? Siden vi vet at løsningen til $\dot{x} = -ax$ er $x(t) = x_0e^{-at}$, så finner vi lett at

$$y(t) = x(t) + \frac{b}{a} = x_0e^{-at} + -\frac{b}{a} = \big(y_0 - \frac{b}{a}\big)e^{-at} + \frac{b}{a} = y_0e^{-at} + (1 - e^{-at})\frac{b}{a}. \tag{2.6}$$

Her har jeg da brukt at siden $x = y - \frac{b}{a}$, så må også $x(0) = x_0 = y_0 - \frac{b}{a}$.

Eksempel/oppgave

Fun facts, bemerkninger og annet dill dall (you may skip)

Ligning (2.5) med $b \neq 0$ er også det enkleste eksempelet på en ikke-homogen (evt. inhomogen) første-ordens differensialligning.[a] Mer spesifikt, så kalles differensialligninger på formen

$$\dot{x} = -ax + f(t)$$

ikke-homogen hvis $f(t) \neq 0$ for alle t.

[a] For første-ordens differensialligninger brukes også «homogenitet» til å beskrive ligninger som kan skrives på formen $\dot{x} = F(x/t)$.

Andre-ordens differensialligninger

Vi skal nå se på differensialligninger av *andre* orden. Et eksempel på en slik ligning er

$$\ddot{x} = -4x.$$

Den er en andre-ordens diff.ligning siden den andre deriverte (mhp. tid) til x dukker opp i ligningen på venstre side, altså $\ddot{x}$. Løsningen på akkurat denne ligningen vil være på formen

$$x(t) = a\sin(\omega t) + b\cos(\omega t), \tag{2.7}$$

noe som impliserer at $x(0) = a$. For å vise at dette faktisk er løsningen, la oss derivere denne funksjonen mhp. tid:

$$\dot{x}(t) = -a\omega\sin(\omega t) + b\omega\cos(\omega t),$$

noe som betyr at $\dot{x}(0) = b\omega$. Ved å derivere nok en gang finner vi at

$$\ddot{x}(t) = -a\omega^2\cos(\omega t) - b\omega^2\sin(\omega t) = -\omega^2\big(a\cos(\omega t) + b\sin(\omega t)\big) = -\omega^2 x(t) \quad \Longrightarrow \quad \ddot{x} = -\omega^2 x.$$

Med andre ord må vi ha at $\omega^2 = 4$, slik at $\omega = \pm 2$, mens vi så igjen finner a og b fra initialverdiene via $a = x(0)$ og $b = \dot{x}(0)/\omega$.

Faktisk så vil en hver andre-ordens differensialligning på formen

$$\ddot{x} = -\omega^2 x \tag{2.8}$$

ha løsningen (2.7) hvor $a = x(0)$ og $b = \dot{x}(0)/\omega$. Et slikt system sies derfor å være **udempet**, siden det ikke er noen form for dempning i systemet, slik at det vil svinge frem og tilbake i det uendelig (gitt at ikke $x(0) = \dot{x}(0) = 0$ vell og merke) siden løsningen bare er bygd opp av en sinus- og consinus funksjon.

Oppgave 2.1. Vis at en andre-ordens diff.ligning på formen

$$\ddot{x} = \lambda^2 x$$

vil ha en løsningen

$$x(t) = ae^{\lambda t} + be^{-\lambda t}. \tag{2.9}$$

Finn også utrykk for a og b gitt av $x(0)$, $\dot{x}(0)$ og λ. Vi sier at en slik ligning tilsvarer et ustabilt system, eller eventuelt at systemet sin løsning er **ustabil**. Hvorfor sier vi det?

Hint: Hva skjer med $x(t)$ nå $t \to \infty$?

En type andre-ordens differensialligning som vil være viktig for oss er følgende:

$$\ddot{x} + 2\omega_0\zeta\dot{x} + \omega_0^2 x = 0. \tag{2.10}$$

Her kalles ω_0 for den *udempede svingefrekvensen* (evt. udempede resonansfrekvensen), mens ζ er den *relative dempningsfaktoren*.

Betydning av dempningsfaktoren: Systemet (2.10) hvor $\omega_0 > 0$ sies å være

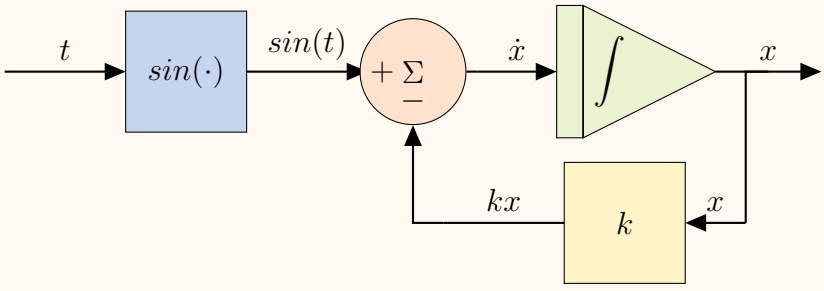
- **Ustabilt** når $\zeta < 0$: utgangen går mot uendelig.
- **Udempet** (marginalt/kritisk stabilt) når $\zeta = 0$: utgangen får stående svingninger
- **Underdempet** når $0 < \zeta < 1$: oscillasjoner om stasjonæreverdien som gradvis dør ut.
- **Kritisk dempet** hvis $\zeta = 1$; returneres så raskt som mulig til likevekt uten oversving.
- **Overdempet** når $\zeta > 1$: returnere gradvis til likevekt uten oversing.

2.4. Blokkdiagrammer

Alternative kilder: §2.i i [Balchen et al., 2016]

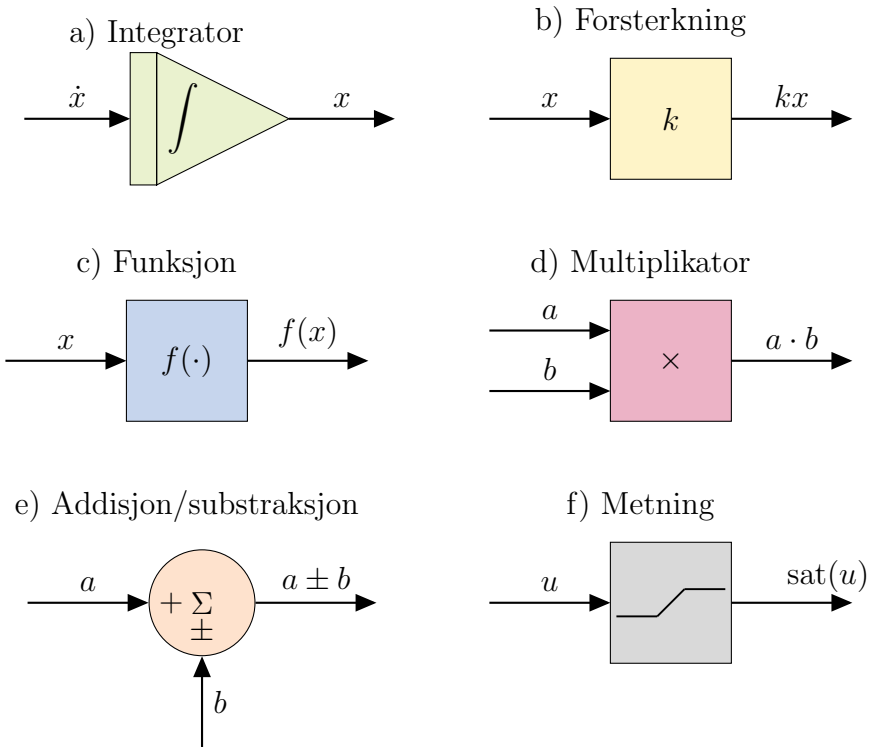
Blokkdiagrammer lar en visualisere flyten i dynamiske systemer ved å representere systemet på en grafisk/skjematisk måte. En utvalg av de vanligste blokkene er vist i figur 2.4 (merk at fargene bare er lagt til for å skille blokkene).

Eksempel 2.3. Blokkdiagrammet til systemet $\dot{x} = -kx + \sin(t)$ er som følger:



Simulink:

Det er mulig å modellere og simulere dynamiske systemer ved hjelp av blokkdiagrammer i **Simulink**. Dessverre bruker Simulink litt andre blokker og notasjon enn de tradisjonelle vi har vist her. Mer spesifikt, så er notasjon der hovedsakelig basert på den såkalte Laplace-variabelen



Figur 2.4: Eksempel på vanlige (klassiske) blokker i et blokkdiagram.

s som vi skal se på i seksjonen 2.8 om overføringsfunksjoner. Spesielt viktig her er at integrator-blokken i Simulink er gitt ved operasjonen $\frac{1}{s}$, altså kan man tenke seg at "$x(t) = \frac{1}{s}\dot{x}(t)$". Vi skal se nærmere på hvordan man kan bruke Simunlink i § 4.2.

2.5. Lineære reguleringssystemer

2.5.1 Første-ordens reguleringssystemer

Et lineært første-ordens reguleringssystem har følgende form:

$$\dot{x} = -ax + bu,$$
$$y = cx + du.$$

Her betegner u inngangen (pådraget) og y utgangen (målingen). For enkelhets skyld ser vi på

$$\dot{y} = -ay + bu \tag{2.11}$$

ved å anta at $c = 1$ og $d = 0$. **Løsningen** til ligning (2.11) er[4]

$$y(t) = e^{-at}\left(y(0) + b\int_0^t e^{a\sigma}u(\sigma)d\sigma\right). \tag{2.12}$$

Fra dette ser vi at hvis $a = 0$ så er $y(t) = y(0) + b\int_0^t u(\sigma)d\sigma$. Vi sier derfor at $\dot{y} = bu$ er en (ren) **integrerende prosess**.

Tvungne likevektspunkter, stasjonær forsterkning og tidskonstant

Legg merke til at for en hver y så finnes det en verdi u slik at $-ay + bu = 0$. Dette betyr at systemet (i teorien) har uendelig mange *tvungne likevektspunkter* (evt. stasjonærpunkter).

Tvunget likevektspunkt/arbeidspunkt:[a] Et par $(\bar{y}, \bar{u})$ slik at $-a\bar{y} + b\bar{u} = 0$.

[a] Når man ønsker at systemet skal operere om et tvunget likevektspunkt, så kalles det et **arbeidspunkt**.

Legger her merke til at vi kan finne hva $\bar{y}$ er fra u_s ved et slikt punkt hvis vi vet a og b. Det vi si at $\bar{y} = \frac{b}{a}\bar{u}$. Vi definerer derfor følgende størrelse:

Stasjonær forsterkning: $K = b/a$.

Merk at den stasjonære forsterkningen bare er av interesse hvis $a > 0$. Grunnet til dette følger fra (2.12), hvor vi kan se at $y(t)$ vil gå mot uendelig for en konstant u hvis $a < 0$. Dette tilsvarer at den åpne-sløyfen er *ustabil* (mer om stabilitet kommer i § 2.7).

Hvis derimot $a > 0$ (og u er konstant), så vil $y(t)$ gå mot en stasjonær verdi når tiden går mot uendelig, som vi derfor betegner y_∞. En viktig størrelse relatert til hvor lang tid det tar å nå en slik verdi er *tidskonstanten* til systemet:

Tidskonstant: Tiden τ det tar å nå ca. 63% av stasjonærverdien y_∞ etter et sprang i u. For et system på formen (2.11) kan man finne denne fra

$$\tau = \frac{1}{a}.$$

Så hvor kommer akkurat 63% fra? Jo, anta at $y(0) = 0$, og la oss ta $\tau = 1/a$ slik at den stasjonære forsterkningen er $K = b \cdot \tau$. Vi kan da skrive (2.11) som

$$\dot{y} = -\frac{1}{\tau}(y - Ku).$$

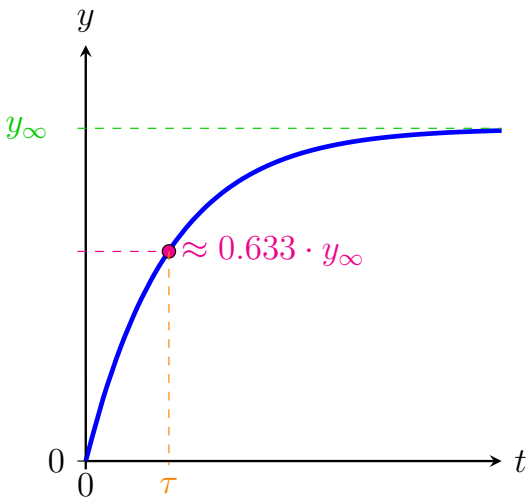
For et sprang i u fra 0 til u_s (= konstant) har dette løsningen

$$\int_0^t \dot{y}(\sigma)d\sigma = \int_0^t \left(-\frac{1}{\tau}(y(\sigma) - Ku_s)\right)d\sigma \quad \Longrightarrow \quad y(t) = \left(1 - e^{-t/\tau}\right)Ku_s.$$

En måte å vise dette på er som følger: Ved å ta $a = 1/\tau$ og $\hat{b} = K \cdot u_s/\tau$, så er systemet (siden u_s er konstant) på den affine formen $\dot{y} = -ay + \hat{b}$, slik at vi kan bruke (2.6) til å finne løsningen.

Fra denne løsningen, altså $y(t) = \left(1 - e^{-t/\tau}\right)Ku_s$, har vi at, når $t = \tau$ så er $y(\tau) = (1 - e^{-1})Ku_s \approx 0.63 \cdot Ku_s = 0.63 \cdot y_\infty$. Med andre ord når vi ca. 63% av y_∞ ved tiden $t = \tau$. Dette er illustrert i figur 2.5 hvor det også er en tabell med relevante tallverdier.

[4] Multipliser begge sider av (2.11) med den integrerende faktoren e^{at}: $e^{at}\dot{y} = -e^{at}ay + e^{at}bu$. Dette kan skrives som $e^{at}\dot{y} + e^{at}ay = \frac{d}{dt}\left(e^{at}y(t)\right) = e^{at}bu(t)$, hvorfra vi får følgende ved å integrere mhp. tid på begge sider: $\int_0^t \frac{d}{d\tau}\left(e^{a\tau}y(\tau)\right)d\tau = \left[e^{a\tau}y(\tau)\right]_0^t = e^{at}y(t) - y(0) = \int_0^t e^{a\tau}bu(\tau)d\tau$. Ved å legge til $y(0)$ på begge sider, samt multiplisere med e^{-at} så får vi da (2.12).



a) Respons etter sprang i u.

t	$1 - e^{-t/\tau}$
0	0
τ	≈ 0.6321
2τ	≈ 0.8647
3τ	≈ 0.9502
4τ	≈ 0.9817
5τ	≈ 0.9933

b) Respons som faktor av tidskonstanten.

Figur 2.5: Sprangrespons og tidskonstant for første-ordens system.

2.5.2 Eksempel (TiT): Hjelp, det blåser (PI-regulering)

Tom i Tallinjeveien har allerede introdusert oss til noen problemer som har ledet oss til P-regulatoren i § 1.4. Han har nå dog et oppfølgingsproblem hvor han trenger vår hjelp til å fjerne et såkalt *stasjonært avvik*:

Nåværende problem: Tom har vært meget fornøyd med cruisekontroll-systemet vi designet for ham i § 1.4 ved hjelp av en P-regulator. Men nå som det har blitt høst, har Tom lagt merke til at det har begynt å blåse en konstant trekk fra ∞ (altså fra høyre). Dette resulterer i at han ikke når den ønskede hastigheten når han kjører mot høyre, mens når han kjører mot venstre vil bilens cruisekontroller få bilen til å kjøre raskere enn ønsket. Tom håper derfor vi kan modifisere P-regulatoren på noe vis slik at dette problemet unngås.

La $y = x_h$ være jetbilens hastighet (målt i (strek-) meter per sekund), hvor $y > 0$ tilsvarer at bilen beveger seg mot høyre, og la r være den ønskede hastigheten. Som før antar vi at massen til bilen med Tom oppi er $m = 1\,\text{kg}$, mens u betegner kraften fra jetmotorene. Ved å ta høyde for en konstant kraft, v, som virker på bilen pga. vinden fra høyre, får vi dermed følgende dynamikk med P-regulatoren $u = k_P(r - y) = k_P e$:

$$\dot{y} = -k_P y + k_P r - v.$$

NB! I realiteten vil kraften v variere med bilens hastighet (grunnet luftmotstanden fra den relative hastigheten til bilen og vinden), men vi antar her at denne er konstant.

Legg merke til at dette er et førte-ordens affint system, tilsvarende (2.5) med $a = -k_P$ og $b = k_P r - v$. Vi har dermed fra (2.6) at løsningen er

$$y(t) = y_0 e^{-k_P t} + (1 - e^{-k_P t})\frac{k_P r - v}{k_P}$$

hvor $y_0 = y(0)$ er starthastigheten.

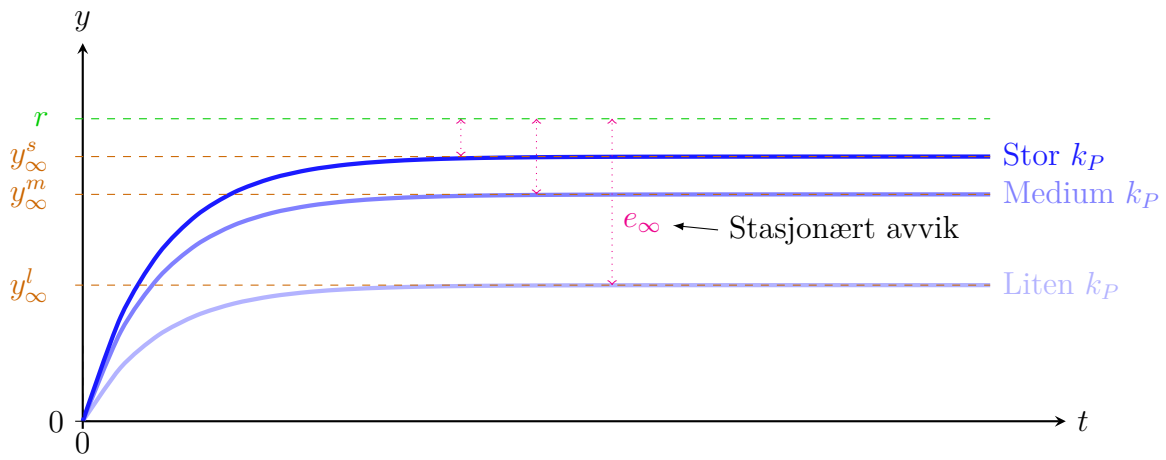
Stasjonært avvik: Hvis $k_P > 0$ har vi derfor at verdien til y når $t \to \infty$, y_∞, er gitt av

$$y_\infty = \lim_{t\to\infty} \left(y_0 e^{-k_P t} + (1 - e^{-k_P t}) \frac{k_P r - v}{k_P} \right) = r - \frac{v}{k_P}$$

siden $e^{-k_P t}$ da går mot null. Dette fører dermed til et endelig, konstant avvik, gitt ved $e_\infty = r - y_\infty = \frac{v}{k_P}$. Vi kaller dette for et **stasjonært avvik**:

Stasjonært avvik: $e_\infty = r - y_\infty$ hvis $y_\infty = \lim_{t-\to\infty} y(t)$ eksisterer og er konstant.

I dette tilfelle er tydelig at vi kan gjøre e_∞ mindre ved å øke k_P; dette er vist i figur 2.6. Men for å fjerne det stasjonære avviket helt, så må k_P tas til å være uendelig stor, noe som igjen vil kreve uendelig pådrag! Reduksjonen i avviket er også omvendt proporsjonal med hensyn på forsterkningen, slik at effekten av å øke k_P også blir mindre og mindre desto større k_P er .



Figur 2.6: Illustrasjon av et stasjonære avvik. Avviket kan bli gjort mindre, men ikke fjernes helt, ved å øke proporsjonal-forsterkningen k_P i en P-regulator.

PI-regulator: Vi kan totalt eliminere det stasjonære avviket ved å leddet til et såkalt *integral(I)-ledd* til regulatoren. Dette gir en **PI**-regulator, som har formen

$$u_{PI}(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau \tag{2.13}$$

hvor k_I er integralforsterkningen. Ideen er som følger: Leddet $\int_0^t e(\tau)d\tau$ integrerer («summerer opp») avviket over tid. Denne akkumuleringen tillater I-leddet å reagere på vedvarende feil, og vil dermed eliminere stasjonære avvik siden størrelsen på I-leddet øker over tid så lenge avviket ikke er null. Dette betyr at så lenge avviket ikke er null så vil virkningen fra I-leddet øke inntil det har eliminert feilen, gitt at det ikke er noen begrensninger (metning) i systemet. Hvor fort leddet øket avhenger av integralforsterkningen k_I.

Ekstra dynamikk: En PI-regulator er et eksempel på en *dynamisk regulator*, siden den har dynamikk (minne) i form av integral-leddet. Dette blir tydelig ved å introdusere den nye «tilstanden» $x_1 = \int_0^t e(\tau)d\tau$, slik at $\dot{x}_1(t) = e(t)$. Ved å også introdusere $x_2(t) = e(t)$, har vi da

$$\dot{x}_1 = x_2 \tag{2.14a}$$

$$\dot{x}_2 = \dot{e} = \dot{r} - \dot{y} = -\left(k_P e + k_I \int_0^t e(\tau)d\tau - v \right) = -k_P x_2 - k_I x_1 + v. \tag{2.14b}$$

Dette er et *andre-ordens system* (siden $\dot{x}_2 = \ddot{x}_1$) med likevektspunkt $(x_1, x_2) = (v/k_I, 0)$. Med andre ord, hvis vi kan velge k_P og k_I slik at tilstandene alltid går til dette likevektspunktet (som da er *asymptotisk stabilt*), så vil avviket $e(t) = x_2(t)$ gå mot null som ønsket, mens $x_1 = \int_0^t e(\tau)d\tau$ vil gå mot verdien v/k_I slik at $k_I x_1 = v$ kansellerer effekten av vinden v.

Det viser seg at vi kan garantere dette ved å ta $k_P > 0$ og $k_I > 0$. For å vise dette, må vi dog første studere andre-ordens reguleringssystemer:

2.5.3 Andre-ordens reguleringssystemer

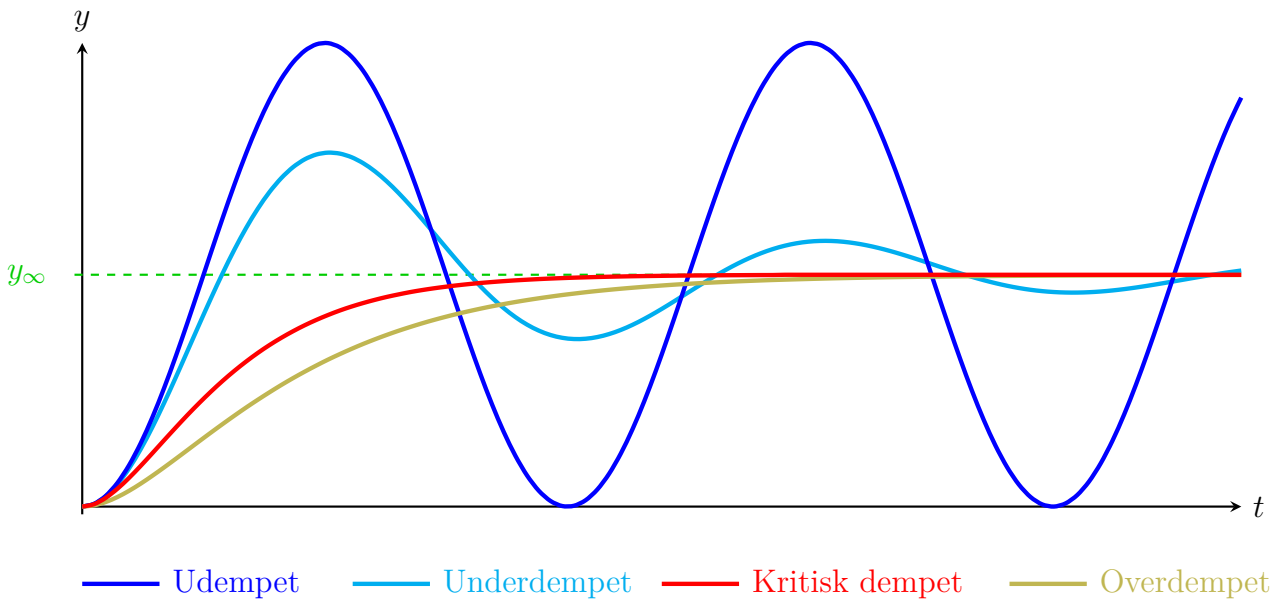
Dynamikken til et lineært andre-ordens reguleringssystem er på formen

$$\ddot{x} = -d \cdot \dot{x} - a \cdot x + b \cdot u,$$
$$y = x.$$

Vi skal dog hovedsakelig holde oss til følgende form som er basert på (2.10):

$$\ddot{x} + 2\omega_0\zeta\dot{x} + \omega_0^2 x = bu, \tag{2.15a}$$
$$y = x. \tag{2.15b}$$

hvor da $a = \omega_0^2$ og $d = 2\zeta\omega_0$, noe som krever at $a > 0$.



Figur 2.7: Sprangresponser til andre-ordens systemer med forskjellig dempningsfaktor.

La oss introdusere tidskonstanten $\tau = 1/\omega_0$ og den stasjonære forsterkningen $K = b/\omega_0^2$.

Anta $y(0) = \dot{y}(0) = 0$. Gitt et sprang i pådraget u ved tiden $t = 0$ fra 0 til en konstant verdi u_s, så er sprangresponsen til andre-ordens systemet (2.15) for forskjellige dempningsfaktorer illustrert i figur 2.7. Disse tilsvarer følgende løsninger for de forskjellige dempningsfaktorene, hvor $\omega_d = \frac{\sqrt{1-\zeta^2}}{\tau} = \omega_0\sqrt{1-\zeta^2}$ kalles den **dempede svinge-/resonans-frekvensen**:

- **Udempet**: $y(t) = u_s K(1 - \cos(\omega_0 t))$.

- **Underdempet**: $y(t) = u_s K \left(1 - e^{-\zeta t/\tau}\left[\cos\left(\omega_d t\right)\right] + \frac{\zeta}{\sqrt{1-\zeta^2}}\sin\left(\omega_d t\right)\right)$;
- **Kritisk dempet**: $y(t) = u_s K \left(1 - (1 + \frac{t}{\tau})e^{-t/\tau}\right)$;
- **Overdempet**: $y(t) = u_s K \left(1 - \frac{\tau_1 e^{-t/\tau_1} - \tau_2 e^{-t/\tau_2}}{\tau_1 - \tau_2}\right)$.

For den overdempede løsningen har jeg brukt $\tau_1 = \tau\zeta + \tau\sqrt{\zeta^2 - 1}$ og $\tau_2 = \tau\zeta - \tau\sqrt{\zeta^2 - 1}$, som er slik at $\tau^2 s^2 + 2\zeta\tau s + 1 = (1 + \tau_1 s)(1 + \tau_2 s)$,

2.5.4 Eksempel (TiT): Tom vil hjem (PD-regulering)

Vi ble i seksjon 1.4 kjent med Tom i Tallinjeveien og de reguleringstekniske utfordringene han har med sin hjemmesnekra jetbil med dynamikk tilsvarende differensialligningen:

$$\underbrace{1}_{m} \cdot \underbrace{\dot{x}_h(t)}_{a} = \underbrace{u(t)}_{\sum F}. \tag{1.1}$$

hvor x_h er jetbilens hastighet. Nå trenger Tom vår hjelp igjen:

Nytt problem: Tom er mye ute å besøker andre i Tallinjeveien, og derfor blitt lei av å kjøre hjem igjen selv tross den nye cruisekontrollen. Han ønsker derfor at vi skal hjelpe ham med å lage en enkel regulerings-strategi som tar ham automatisk hjem når han starter fra et hvilket som helst sted i Tallinjeveien, altså $x_p \in \mathbb{R}$ med hastighet $x_h = 0$.
Viktig! Naboen hans, Fru Møge 👵, som bor i nummer 1 (se figur 1.10), er en skikkelig sladrehank og vil fortelle alle i Tallinjeveien hvis det er noe rart med hvordan bilen til Tom oppfører seg. Vi må derfor sørge for at bilen går direkte til nummer 0, altså uten noe oversving/-skyting. Med andre ord: vi må ha en kritisk- eller overdempet respons.

Naiv strategi: Det kan helt umiddelbart være fristende å gå for en strategi lik den lineære strategien fra cruisekontroll-regulatoren i seksjon 1.4.3, ved å bare bytte ut x_h med x_p:

$$u = -b \cdot x_p.$$

Husk dog at hastighet er tidsderivatet av posisjon. Hvis $x_p(t)$ er jetbilens posisjon ved tiden t, så har vi dermed fra (1.1) at

$$\ddot{x}_p = u,$$

som er en *andre-ordens* differensialligning (vi har to dotter på venstresiden). Ved å nå ta utgangen til systemet som $y = x_p$ får vi dermed

$$\ddot{y} = u. \tag{2.16}$$

Ved å sette inn regulatoren over får vi dermed $\ddot{y} = -by$.

NB! Jeg kaller nå $y = x_p$ for *utgangen*, og ikke *målingen*, siden vi antar at vi også kan måle hastigheten $\dot{y} = x_h$ direkte. Utgangen kan derfor i slike sammenhenger tenkes på som den størrelsen man er mest interessert i å styre.

Det viser seg dog at ved starte ved posisjonen $y(0) = x_p(0) = y_0$ (med $\dot{y}(0) = x_h((0) = 0)$, så får vi følgende løsningen:

$$y(t) = y_0 \sin\left(\sqrt{b} \cdot t\right).$$

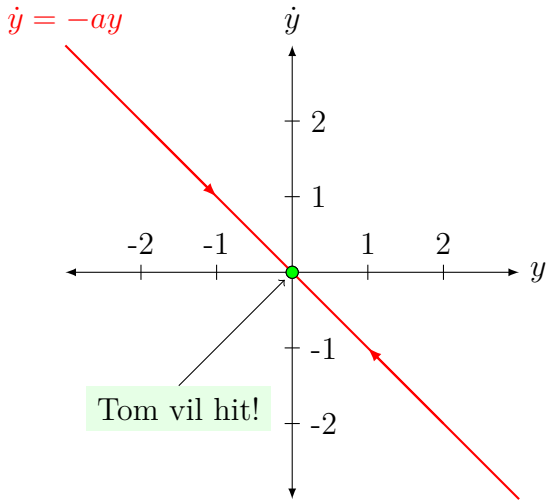
Med andre ord vil Toms bil bare svinge frem og tilbake om huset hans med amplitude y_0! Det vi har gjort er faktisk å ha formet den lukkede-sløyfen til å tilsvare en såkalt harmonisk oscillator som har udempede, stående svingninger.

Bedre strategi: En lærdom fra den forrive naive strategien er at vi også må ta hensyn til bilens hastighet slik at vi ikke ender opp med å overskyte målet. Mer spesifikt, så må vi lage en strategi som faktisk begynner å aktivt bremse *før* vi når målet. Men hvordan gjør vi det? Det er for så vidt flust av muligheter her, men den meste hensiktsmessige for våres del er en såkalt *PD-regulator* (P=proporsjonal, D=derivat) som fører til en *kritisk dempet respons*. Metoden vi skal se på tar i bruk et *reduksjonsprinsipp*, som egentlig er en litt uortodoks avansert strategi.

La oss introdusere en ny variabel som vi ønsker å styre til null:

$$\sigma = x_h + ax_p = \dot{y} + ay.$$

Her er $a > 0$ en positiv konstant, mens σ er den greske bokstaven «sigma». Ideen her er som følger: Hvis vi kan sørge for at $\sigma = 0$ for alle $t \geq t_0$, så har får vi $\dot{y} = -ay$. Dette tilsvarer at $y(t) = x_p(t) = y(t_0)e^{-a(t-t_0)}$, noe som betyr at at punktet $y = x_p = 0$ da er såkalt *eksponentielt stabilt*.

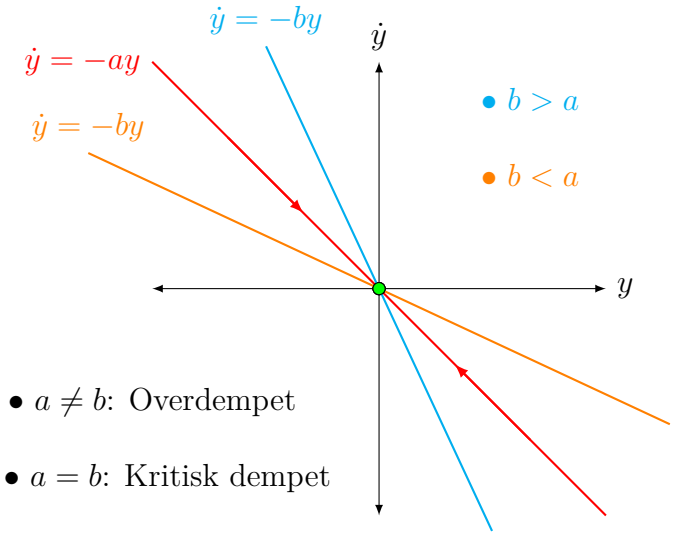


Figur 2.8: Tom kan komme seg hjem som ønsket til nummer 0 ved å følge hastighetsprofilen gitt av den røde kurven i tilstandsplanet.

Hastighetskurve i tilstandsplanet: En måte å tenke på dette er at

$$\sigma = \dot{y} + ay = x_h + ax_p = 0$$

gir oss en ønsket *hastighetsprofil* som er bestemt av bilens posisjon x_p, altså $x_h = f(x_p) = -ax_p$. Dette tilsvarer en kurve i det såkalte *tilstandsplanet* (evt. faseportrett eller faseplan), som vist i figur 2.8. Det vil si, et plan hvor vi måler posisjonen $y = x_p$ langs den horisontale asken og hastigheten $\dot{y} = x_h = \dot{x}_p$ langs den vertikale aksen (se Wikipedia). Tanken er at ved å følge



Figur 2.9: Illustrasjon av forskjellige hastighetskurver i tilstandsplanet for den lukkede sløyfen.

den røde kurven i figur 2.8 som tilsvarer $\sigma = 0$, så vil bilen til Tom ende opp ved huset hans ($x_p = 0$) med null hastighet ($x_h = \dot{x}_p = 0$) uten noen form for oversing, og dermed ingen sladder fra Fru Møge! 🙌

Men hvordan kan vi sørge for å få at $\sigma = 0$? Jo, vi kan ta i bruk «veien mot null» fra figur 1.11. I den forbindelse, så starter vi med å finne tidsderivatet til σ:

$$\dot{\sigma} = \frac{d}{dt}(\dot{y} + ay) = \ddot{y} + a\dot{y} = u + a\dot{y},$$

hvor jeg har brukt (2.16). Leddet $a\dot{x}$ er her litt i veien, så la oss ta

$$u = -a\dot{y} - \hat{u}$$

slik at $\dot{\sigma} = -\hat{u}$. Dette er første-ordens system akkurat slik som det vi så på i seksjon 1.4.3 når vi designet cruise-kontroll for bilen hans! Det betyr at vi f.eks. kan bruke de samme reguleringsstrategiene i figur 1.12 bare vi bytter ut u med $\hat{u}$ og e med σ.

Det er dog den lineære regulatoren, altså $\hat{u} = b\sigma$, som er meste hensiktsmessig i forhold til dette faget her. Ved å sette dette inn i utrykket for u gir det oss den tidligere nevnte PD-regulatoren:

$$u = -a\dot{y} - b\sigma = -a\dot{y} - b(\dot{y} + ay) = -\underbrace{ab}_{k_P} y - \underbrace{(a+b)}_{k_D}\dot{y}. \qquad \text{(PD-regulator)}$$

Her kalles $k_P = ab$ for proporsjonalforsterkningen, mens $k_D = a + b$ er derivatforsterkningen.

Man kan vise at hvis tilstandene starter på én av kurvene $\dot{y} + ay = 0$ eller $\dot{y} + by = 0$, vist i figur 2.9, så vil tilstandene forbli på denne kurven å gå mot origo (altså $(\dot{y}, y) = (0, 0)$). Det viser seg også at systemet har en såkalt *kritisk dempet* respons hvis $a = b$, mens det er *overdempet* ellers. Vi kaller $\lambda_1 = -a$ og $\lambda_2 = -b$ for egenverdiene til dette systemet (det er to egenverdien siden systemet er av andre orden), mens kurvene $\dot{y} + ay = 0$ eller $\dot{y} + by = 0$ tilsvarer egenvektorene.

Oppgave 2.2. Anta at man bruker en PD-regulator for systemet $\ddot{y} = u$. Vis at

a) hvis $\sigma(t_0) = \dot{y}(t_0) + ay(t_0) = 0$ eller $\rho(t_0) = \dot{y}(t_0) + by(t_0) = 0$, så vil henholdsvis $\sigma(t) = 0$ eller $\rho(t) = 0$ for alle $t \geq t_0$. Det vil si, hvis man starter med tilstandene slik at enten $\sigma = 0$ eller $\rho = 0$ med en slik regulator, så vil henholdsvis σ eller ρ forbli lik null til evig tid.

b) systemet er kritisk dempet hvis $a = b$, og overdempet ellers.

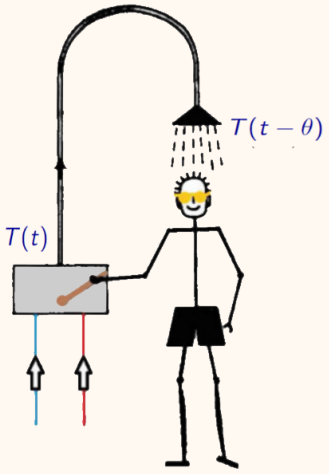
2.6. Effekten av tidsforsinkelser

Alternative kilder: [Bjørvik and Hveem, 2014, §4.1 og §5.6], Brian Douglas video.

Hvorfor skal du lære dette? 🤷‍♀️ Tidsforinsikelser (også kalt dødtid eller transportforsinkelse, avhengig sammenheng) er uunngåelig i del fleste reguleringssystemer. Siden tidsforsinkelser potensielt kan føre til en betraktelig begrensning i ytelsen til et reguleringssystem, er det viktig at man tenker på tidsforsinkelser når man utvikler en regulator.

Noen eksempler på årsaker til tidsforsinkelser er masse-transport i et rør, dødtid i en motor pga. statisk friksjon, aktuator-dynamikk, kommunikasjon over nettverk (kremt, CAN-buss, kremt), samt sampling- og utregningstid. La oss se på et muligens releterbart eksempel:

Eksempel 2.4. (I dusjen, eksempel og figur tatt fra [Fridman, 2014]) Se for deg en dusjende person som ønsker å oppnå ønsket vanntemperatur, T_r, ved vha. blandebatteriet, som vist i figuren til høyre. La $T(t)$ betegne vanntemperaturen i blandebatteriets utgang ved tiden t, og la θ være den konstante tiden som trengs av vannet for å gå fra mikserutgang til personens hode. Anta at endringen av temperaturen er k_P-proporsjonal med rotasjonsvinkelen til håndtaket, mens hastigheten man kan rotere håndtaket er proporsjonal med $T(t) - T_r$. Ved tidspunktet t føler personen vanntemperaturen som forlater mikseren på tidspunktet $t - \theta$, noe som resulterer i følgende ligning med konstant forsinkelse θ:

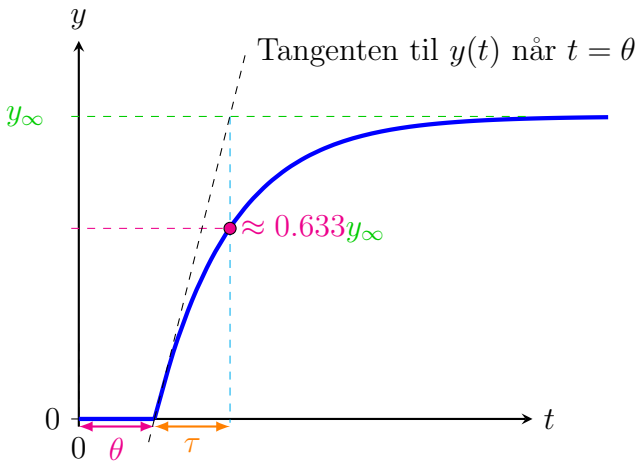


$$\dot{T}(t) = -k_P \left(T(t - \theta) - T_r \right).$$

Hvis vi definerer avviket som $e(t) = T(t) - T_d$, så får vi følgende avviksdynamikk:

$$\dot{e}(t) = -k_P \cdot e(t - \theta).$$

⚠ **Varning:** Uten tidsforsinkelsen kunne vi (teoretisk sett) gjort responsen så rask og responsiv vi bare ville ved å øke k siden løsningen da er $e(t) = e(0)\exp(-k_P \cdot t)$ hvor $\exp(\cdot)$ er eksponentialfunksjonen. Historien er dog en helt annen hvis vi har en tidsforsinkelse, siden for aggressive endringer (stor k_P) da kan ville føre til oversving, og iverste fall *ustabilitet* (mer om stabilitet om litt).



Figur 2.10: Sprangresponsen til et første-ordens-pluss-tidsforsinkelse (FOPTF) system.

Første-ordens-pluss-tidsforsinkelse (FOPTF) systemer

I disse notatene skal vi kun se på såkalte første-ordens-pluss-tidsforsinkelse (FOPTF) systemer (eng. first-ordes plus time delay (FOPTD)). Et slikt system har følgende form:

$$\dot{y} = -ay + bu(t - \theta), \tag{FOPTF}$$

altså er det et første-ordens reguleringssystem hvor inngangen er forsinket med θ seksunder. Merk at systemet i eksempel 2.4 var på denne formen med $a = 0$ og med proporsjonal-regulatoren $u = k_P(T_d - T)$.

Hvis man ved tiden $t = 0$ ønsker å gjennomføre ett sprang på innganen til et slikt system, så fører dette til en respons slik som det vist i 2.10, siden det tar θ sekunder før dette spranget faktisk påvirker systemet.

Oppgave 2.3. Gitt følgende FOPTF-system:

$$\dot{y} = -2y + u(t - 0.1).$$

Dette systemet er ustabilt i åpen sløyfe, men kan bli gjort stabilt ved hjelp av en P-regulator. På den annen side, så vil en for stor proporsjonalforsterkning igjen føre til ustabilitet.

Bruk Simulink til å finne ut hvor stor k_P må være for at systemet blir stabilt, samt den øvre grensen slik at systemet ikke blir ustabilt igjen. Bruk f.eks. initialverdien $y(0) = 1$ og

referansen $r = 0$. Tidsforsinkelsen implementerer du vha. transport delay-blokken, mens initialverdien setter du ved å dobbel-klikke på integratorblokken.

2.7. Stabilitet og responser

2.7.1 Transiente og stasjonære responser

Figur 2.5 og 2.7 viste responsene til henholdsvis et første- og andre-ordens system etter et sprang i pådraget, såkalte **sprangresponser**. Disse responsene kan vi igjen løst dele opp i to faser: den **transiente** fasen og den **stasjonære** fasen.

Stasjonær fasen tilsvarer den faset når systemets utgang (og interne tilstands) har «falt til ro», det vi si at de enten har nådd en konstant eller repeterende (oscillerende) verdi. Dette skjer ofte etter en betydelig tid har gått slik at alle **transienter** har dødd ut.

Transient fasen, på den annen side, refererer til fasen hvor systemets utgang (og interne tilstander) er i endring etter en ytre påvirking som et sprang pådraget eller en forstyrrelse. Selve ordet «transinet» betyr kortlevd og forbigående; denne fasen vil derfor «død ut». Tenk deg for eksempel at du hopper opp i en vannseng. Dette fører til transienter i form av repeterende bølger med amplitude som etterhvert blir mindre og mindre, helt sengen blir rolig igjen og man går over i en nye stasjonær fase.

Det gir dog egentlig bare mening å snakke om stasjonære og transiente responer for *stabile* systemer:

2.7.2 Stabilitet

Hvorfor skal du lære dette? 🤷‍♀️ Stabilitet er et kritisk konsept innen reguleringsteknikken, og det er slik at man aldri bør implementere en regulator uten å ha forsikret seg i forkant om at den lukkede sløyfen er stabil. Grunnen til dette er enkel: Et stabilt system vil etter en forstyrrelse returnere til sin stasjonære tilstand, mens et ustabilt system vil avvike fra den stasjonære tilstanden og potensielt «løpe løpsk», noe man naturlig nok svært sjeldent ønsker eller kan tillate at skjer. For eksempel, hvis et reguleringssystem tilsvarende en bils cruiskontroll er ustabilt, kan dette naturlig nok føre til ukontrollert atferd som kan føre til svært alvorlig ulykker.

Stabilitetsanalyse er derfor et viktig skritt i designprosessen av et reguleringssystem for å sikre at systemet fungerer på en sikker og forutsigbar måte. Men hva er egentlig stabilitet? 🤔

Stabilitet er strengt tatt et ganske vidt begrep, og det finnes mange forskjellige varianter og definisjoner. For lineær, tids-invariant (LTI) systemer, som vi hovedsakelig er interessert i, er begrepet «stabilitet» heldigvis noe mer klart og avgrenset. Spesifikt, så er det to begreper som vi er interessert i: *inngang–utgang stabilitet* og stabiliteten til et punkt i systemets tilstandsrom.

Inngang-utgang-stabilitet

Inngang-utgang-stabilitet refererer til et systems evne til å få begrensede utganger når det mottar begrensede inngangsverdier. Med andre ord, hvis et system er inngang-utgang-stabilt,

betyr det at en begrenset pådrag aldri vil resultere i en uendelig (eller ubegrenset) utgang. Dette er en svært viktig form for stabilitet fordi det sikrer at systemet ikke vil produsere ekstreme eller uforutsigbare utganger i respons til normale eller begrensede inngangssignaler.

Inngang-utgang-stabilitet: La $u(t)$ være et signal med endelig varighet og begrenset amplitude (f.eks. en enhetspuls eller én enkelt firkantpuls). Et lineære, tids-invariant (LTI) system med én utgang, $y(t)$, og én inngang, $u(t)$, sies å være

- **inngang-utgang-stabilt** dersom $|y(t)| < \infty$ for alle tidpunkt $t \geq 0$ (dette kalles også et **marginal inngang-utgang-stabilitet**);
- **asymptotisk inngang-utgang-stabilt** dersom det er inngangs-utgangs-stabilt og $y(t) \to 0$ når $t \to \infty$;
- **ustabilt** dersom det ikke er stabilt.

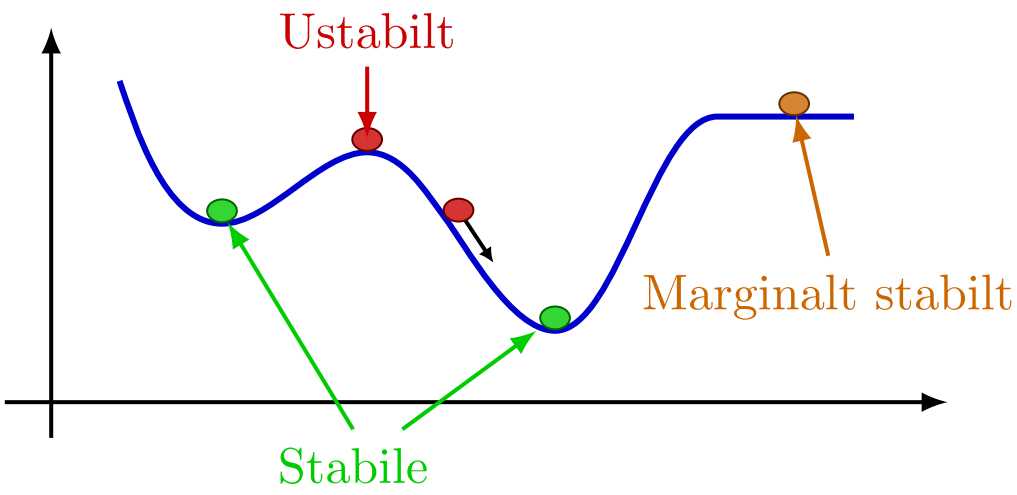
For eksempel, så er systemet (2.11), altså

$$\dot{y} = -ay + bu,$$

asymptotisk inngang-utgang stabilt hvis (og bare hvis) $a > 0$, siden den har løsningen

$$y(t) = e^{-at}\left(y(0) + b\int_0^t e^{a\sigma}u(\sigma)d\sigma\right)$$

hvor leddet $e^{-at} \to 0$ når $t \to \infty$. Med andre ord, hvis vi gir et slikt system en liten kakk med f.eks. en hammer (en impuls) så vil det etterhvert gå tilbake til sin utgangsposisjon.



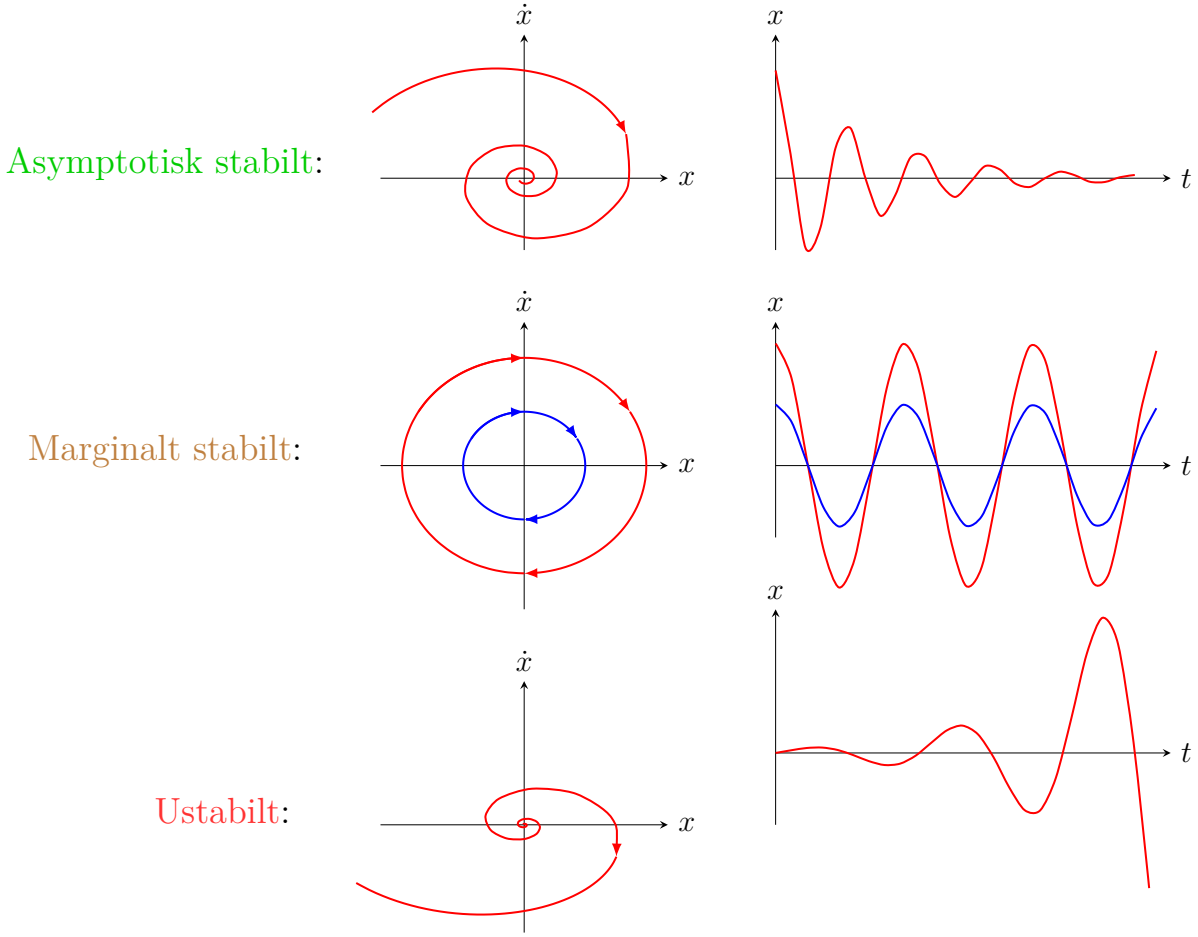
Figur 2.11: Illustrasjon av stabile, ustabile og marginalt stabile (likevekts-)punkter. Hvis du gir ballene plassert ved de stabile punktene en liten kakk (med en hammer), så vil de etter kort tid komme tilbake til disse punktene. Dette er ikke tilfelle or ballen plassert på det ustabile punktet, som etter et lite kakk aldri vil returnere til dette punktet. En kakk på den oransje ballen vil føre til at den flytter seg noe (hvor mye avhenger av kraften i kakket), men den vil trolig stoppe i nærheten av dette punktet. Slike punkter sies derfor å være marginalt stabile.

Stabilitet i tilstandsrommet*

Gitt et dynamisk system med n tilstander på tilstandsromform:

$$\dot{x}_1 = f_1(x_1, x_2, \ldots, x_n), \tag{2.17}$$
$$\dot{x}_2 = f_2(x_1, x_2, \ldots, x_n), \tag{2.18}$$
$$\vdots \tag{2.19}$$
$$\dot{x}_n = f_n(x_1, x_2, \ldots, x_n). \tag{2.20}$$

Anta at $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n$ tilsvarer et likevektspunkt for systemet, det vil si $f_i(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n) = 0$ for $i = 1, 2, \ldots, n$. Det er ofte av interesse å studere stabiliteten til slike punkter[5] i tilstandsrommet. Det kan f.eks. være at det dynamiske systemet tilsvarer en matematisk modell av et fysisk system, eller et reguleringssystem i lukket sløyfe (f.eks. en industrirobot eller en F35), og at vi ønsker å studerer dets stabilitetskarakteristikker.



Figur 2.12: Illustrasjon av forskjellige stabilitets-krakateristikker til punktet $y = 0$; venstre= tilstandsplanet; høyre = tidsplanet; se også: https://youtu.be/KO6sonZb1c0?t=252

Figur 2.12 viser responsen til andre-ordens systemer med forskjellige stabilitet. For enkelhets skyld, så skal vi kun gi definisjonen til denne typen stabilitet, kalt Lyapunov stabilitet, til

[5]Ikke bare punkter, men også stabiliteten til andre invariante mengder (ofte baner), som f.eks. grensesvigninger (eng. limit cycles), kan være interessant å studere innen applikasjoner som f.eks. robotgange.

første-ordens systemet $\dot{x} = f(x)$. Vi skal også bare se på stabiliteten til likevektspunktet $\bar{x} = 0$. Definisjonen blir ikke mindre generell for det, siden vi alltid kan forskyve likevektspunktet til origo ved hjelp av et variabelskifte ikke ulikt det vi gjorde det for det affine systemet (2.5).

Stabilitet til likevektspunkt:[a] Likevektspunktet $\bar{x} = 0$ til systemet $\dot{x} = f(x)$ er

- **stabilt** hvis det, for enhver $\varepsilon > 0$, finnes en $\delta = \delta(\varepsilon) > 0$ slik at $|x(0)| < \delta \implies |x(t)| < \varepsilon$ for alle $t \geq 0$;
- **asymptotisk stabilt** hvis det er stabilt og δ kan velges slik at $|x(0)| < \delta \implies |x(t)| \to 0$ når $t \to \infty$;
- **ustabilt** dersom det ikke er stabilt.

[a] Definisjonen kan generaliserer til systemer med n tilstander, $x_1, x_2, \ldots, x_n$, ved å bytte ut absoluttverdiene med den Euklidiske vektornormen $\sqrt{|x_1|^2 + |x_2|^2 \cdots + |x_n|^2}$.

Oppgave 2.4. Vi ønsker å studere stabiliteten til følgende system sitt likevektspunkt:

$$\dot{y} = -y + 2.$$

Gjør følgende: a) Utfør et bytte av variabler slik at differensialligningen til den nye variabelen har et likevektspunkt i origo. b) Sjekk stabiliteten til likevektspunktet i origo til dette nye systemet.

Hint: Dette er et system på affin form.

Fun facts, bemerkninger og annet dill dall (you may skip)

Hold deg alltid til venstre halvplan: Nullpunktet til et lineært dynamisk system på formen $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$ er asymptotisk stabilt hvis, og bare hvis, alle egenverdiene til systemet er i venstre halvdel av det komplekse plan (har negativ realdel). Dette er det samme som at tilsvarende *overføringsfunksjon* har alle sine poler i venstre halvplan.

2.8. Overføringsfunksjoner

Du vil lære mer om dette i IMAT2012 - Matematikk for ingeniørfag 2.

Vi skal nå se på en annen måte representere dynamiske systemer på, kalt overføringsfunksjoner. Rettere sagt skal vi se en forenklet intuitiv utledning av disse, mens dere i andre fag vil lære en noe mer grundig metode for å utlede disse funksjonene vha. Laplace-transformasjonen.

Hvorfor skal du lære dette? 🤷‍♀️ Overføringsfunksjoner er svært mye brukt i praksis, ikke minst fordi det gjør det lettere å studere reguleringssystemer i frekvensplanet. Mye av reguleringsteknikken dere vil lære i senere fag baserer seg også på overføringsfunksjoner.

Si vi har et lineært system, f.eks. det gitt av følgende andre-ordens differensialligning:

$$\ddot{y}(t) + d\dot{y}(t) + ay(t) = bu(t).$$

Vi har tidligere sett at løsningene til både første- og andre-ordens lineære systemer er bygd opp av eksponential og trigonometriske funksjoner. La oss derfor prøve på noe litt fiffig:

Vi antar nå at for en gitt inngang $u(t) = U(s)e^{st}$ så er løsning på formen $y(t) = Y(s)e^{st}$., hvor $U(\cdot)$ og $Y(\cdot)$ er funksjoner av den nye, mystifistiske variabelen s. Men finnes det faktisk en slik løsning, og hva må den i så fall tilfredsstille? 🤔

For å svare på dette skal vi sette disse løsningene inn i differensialligningen. Men før vi gjør det, så legger vi merke til at følgende holder for $y(t) = Y(s)e^{st}$:

$$\dot{y}(t) = \frac{d}{dt}\left(Y(s)e^{st}\right) = Y(s)se^{st} = s \cdot y(t)$$

Fun facts, bemerkninger og annet dill dall (you may skip)

Legg her merke til at $\dot{y}(t) = sy(t)$, altså kan vi tenke på s som en slags derivator, mens siden vi får $y(t) = \frac{1}{s}\dot{y}(t)$, så kan vi tenke på $\frac{1}{s}$ som en integrator. Sisnevnte brukes jo faktisk som symbol på integrator-blokken i Simulnik .

Ved å gjøre dette nok en gang, får vi

$$\ddot{y}(t) = Y(s)s^2e^{st} = s^2 \cdot y(t).$$

Ved å sette dette inn i differensialligninger over får vi:

$$s^2Y(s)e^{st} + dsY(s)e^{st} + aY(s)e^{st} = bU(s)e^{st}.$$

Ved å dele på e^{st} på begge sider reduseres dette til

$$s^2Y(s) + dsY(s) + aY(s) = bU(s) \quad \Longrightarrow \quad \frac{Y(s)}{U(s)} = \frac{b}{s^2 + ds + a}.$$

Hvis vi derfor definerer

$$P(s) = \frac{b}{s^2 + ds + a}$$

så får vi at $Y(s)$ og $U(s)$ må være relatert via $Y(s) = P(s)U(s)$. Vi kaller derfor $P(s)$ for **overføringsfunksjonen** (også kalt transferfunksjon) fra $U(s)$ til $Y(s)$.

Legg her merke til at vi kan finne den stasjonære forsterkningen ved å sette $s = 0$ i overføringsfunksjonen: $K = P(0) = \frac{b}{a}$.

Oppgave 2.5. Anta at du har funnet overføringsfunksjonen $P(s)$ til en prosess, slik at $Y(s) = P(s)U(s)$. Anta videre at du ønsker å implementere en P-regulator, $u = k_P(r - y)$ for denne prosessen, hvor du også kan anta at $r(t) = R(s)e^{st}$. Vis at overføringsfuksjonen

til den lukkede sløyfen, fra $R(s)$ til $Y(s)$, er gitt ved

$$\frac{Y(s)}{R(s)} = \frac{k_P P(s)}{1 + k_P P(s)}.$$

2.9. Mono-variable vs multivariable systemer

Vårt fokus i disse notatene skal være på såkalte *monovariable* reguleringssystemer, som er systemer med én inngang og én utgang. Eksempler på dette er et første- eller andre-ordens system, for eksempel $\dot{x} = ax + bu$ eller $\ddot{x} = ax + bu$, med utgangsvariabel $y = x$ og inngangsvariabel u.

Multivariable reguleringssystemer, derimot, har to eller flere innganger og/eller utganger. I motsetning til monovariable (eng. Single-Input Single-Output (SISO)) reguleringssystemer, kan multivariable (eng. Multi-Input Multi-Output (MIMO)) systemer dermed ha flere pådragsorgan og flere sensorer.

Et eksempel på multivariabelt-system er følgende sett av differensialligninger:

$$\begin{aligned} \dot{x}_1 &= a_{11}x_1 + a_{12}x_2 + b_{11}u_1 + b_{12}u_2 \\ \dot{x}_2 &= a_{21}x_1 + a_{22}x_2 + b_{21}u_1 + b_{22}u_2 \end{aligned} \tag{2.21}$$

med utgangene $y_1 = x_1$ og $y_2 = x_2$, hvor x_1 og x_2 er systemets tilstander, u_1 og u_2 er inngangssignaler, og a_{ij} og b_{ij} er konstante parametere.

Et viktig aspekt ved multivariable systemer er *krysskoblinger* mellom variablene. Dette betyr at endringer i en inngangsvariabel kan påvirke mer enn én utgangsvariabel, noe som kan komplisere både regulatordesign og analyse av reguleringssystemet.

2.10. Diskretisering og differensligninger

Alternative kilder: Kap. 24 i [Haugen, 2023].

Hvorfor skal du lære dette? 🤷‍♀️ Innen reguleringsteknikk brukes differensligninger i stor grad i praksis siden de fleste reguleringssystemer i dag er digitale. Dette betyr at man kun kan lese av målinger fra sensorer, samt endre verdien til pådragsorganet ved gitte tidspunkter, noe som samsvarer med formen til differensligninger.

Differensligninger er en annen måte å representere dynamiske systemer på som er spesielt godt egnet for digitale reguleringssystemer med en lang tastetid. Følgende er et eksempel på en enkel lineær differensligning med konstante parametere α og β:

$$y[k+1] = \alpha y[k] + \beta u[k], \quad k = 0, 1, 2, \ldots.$$

Her kan $y[k]$ tenkes på som verdien til utganen y ved tiden t_k, altså $y[k] = y(t_k)$. Ligningen over tilsvarer derfor $y(t_{k+1}) = -\alpha y(t_k) + \beta u(t_k)$; altså, vi kan finne y ved tiden $t_{k+1} > t_k$ fra differensligningen hvis vi vet $y(t_k)$ og $u(t_k)$.

Som regel har vi et konstant *tidssteg* $\mathfrak{h}$ mellom hvert tidspunkt, altså er $\mathfrak{h} = t_{k+1} - t_k$ lik for alle heltall k. Siden vi er interessert hovedsakelig i digitale reguleringssystemer, så tilsvarer dette tidssteget som oftest tastetiden for vår del.

Vi kan også a differensligninger som avhenger av flere tidspunkter, f.eks.,

$$y[k+2] + \alpha_1 y[k+1] + \alpha_0 y[k] = \beta_1 u[k+1] + \beta_0 u[k].$$

Vi skal se nærmere hvordan man kan utlede slike ligninger fra differensialligninger ved *diskretisering* om litt, men først litt om stabilitet.

2.10.1 Diskret matematikk og stabilitet

I motsetning til en differensialligning slik som (2.11), hvor man får en kontinuerlig bilde av hvordan y endrer seg, så gir er differensligning bare informasjon om verdien ved de diskrete tidspunktene $t_k, t_{k+1}, t_{k+2}, \ldots$. Det er dermed en form for *diskret matematikk*, hvor kriteriene for blant annet stabilitet er noe annerledes enn i det kontinuerlige tillfellet. F.eks, så husker vi at nullpunktet til $\dot{x}(t) = -ax(t)$ er asymptotisk stabilt hvis, og bare hvis, $a > 0$. For det diskrete systemet $x[k+1] = \alpha x[k]$, derimot, så er nullpunktet asymptotisk stabilt hvis, og bare hvis, $|\alpha| < 1$. Kan du se hvorfor det er slik? 🤔

Fun facts, bemerkninger og annet dill dall (you may skip)

Z-transformen: Ikke ulikt Laplace-transformasjonen for lineære differensialligninger (se § 2.8) hvor man bruke s-operatoren, så finnes det en såkalt *Z-transformasjon* for differensligninger basert på z-operatoren. Denne har egenskapen at $x[k+1]$ tilsvarer $zX(z)$ i det så kalte Z-domene. På samme måte som at $\frac{1}{s}$ blir brukt i Simulink som en integrator, altså "$x(t) = \frac{1}{s}\dot{x}(t)$", så brukes $\frac{1}{z}$ til å representere en tidsforsinkelse (av en gitt lengde), altså "$x[k-1] = \frac{1}{z}x[k]$".

2.10.2 Diskretisering

Prosedyren hvor vi går fra en differensialligning til differensligning kalles for *diskretisering*. For et generelt systemkan man bruke omtrentlige diskretiseringsmetoder, mens for noen systemer kan man bruke eksakte diskretiseringsmetoder hvor da den resulterende differensligningen gir den samme løsningen som den opprinnelige differensialligningen.

La oss ta følgende første-orden differensialligning til å illustrere både en omtrentlig og en eksakt diskretisering:

$$\dot{y} = -ay + bu.$$

Vi antar følgende:

1. Systemet samples med konstant tastetid $\mathfrak{h}$.
2. Det brukes zero-order hold for pådraget, slik at pådraget $u(t) = u_k$ holdes konstant mellom to tastetidspunkter t_k og $t_{k+1} = t_k + \mathfrak{h}$, $k = 0, 1, 2, \ldots$.

Vi bruker derfor naturlig nok $\hbar$ som steglengde. Disse antagelsene betyr at vi har $\dot{y}(t) = -ay(t) + bu_k$ for $t_k \leq t < t_{k+1}$.

Omtrentlig metode: Et eksempel på en omtrentlig metode er Eulers fremovermetode (4.2):

$$y[k+1] = y[k] + h\big(-ay[k] + bu[k]\big) = \underbrace{(1-ha)}_{\alpha} y[k] + \underbrace{hb}_{\beta} u[k],$$

hvor $u[k] = u(t_k)$, $y[k] = y(t_k)$ og $y[k+1] = y(t_k + \hbar)$. Selv om det opprinnelige systemet er stabilt i åpen sløyfe ($u = 0$) så lenge $a > 0$, så er dette diskretiserte systemet stabilt bare hvis $0 < \hbar a < 2$ (husk at stabilitet tilsvarer $|\alpha| < 1$). Dette indikerer at denne diskretiseringen trolig bare vil gi en grei representasjon av systemets dynamikk hvis $\hbar$ er liten, spesielt hvis a er stor (noe som jo indikerer en liten tidskonstant og dermed rask dynamikk).

Eksakt metode: La oss nå se på en eksakt metode som baserer seg på løsningen til differensialligningen. Husk nå fra (2.12) i § 2.5.1 (se også løsningen på den affine diff.ligningen i § 2.3.2) at løsningen til systemet $\dot{y} = -ay + bu$ over et tasteintervall $[t_k, t_{k+1}] = [t_k, t_k + \hbar]$ er

$$y(t_{k+1}) = e^{-a(t_{k+1}-t_k)}\left(y(t_k) + b\int_{t_k}^{t_{k+1}} e^{a(t-t_k)}u(t)dt\right) = e^{-a\hbar}\left(y(t_k) + b\int_{t_k}^{t_k+\hbar} e^{a(t-t_k)}u(t)dt\right).$$

Ved å bruke at $u(t) = u_k = u[k]$ over tasteintervallet $[t_k, t_k + \hbar)$ har vi her at

$$\int_{t_k}^{t_k+\hbar} e^{a(t-t_k)}u(t)dt = u[k]\int_{t_k}^{t_k+\hbar} e^{a(t-t_k)}dt = u[k]\left[\frac{1}{a}e^{a(t-t_k)}\right]_{t_k}^{t_k+\hbar} = \frac{1}{a}u[k]\left(e^{a\hbar}-1\right).$$

Ved å sette dette inn i ligningen over med notasjonen $y[k] = y(t_k)$ får vi

$$y[k+1] = e^{-a\hbar}\left(y[k] + \frac{b}{a}u[k]\left(e^{a\hbar}-1\right)\right) = e^{-a\hbar}y[k] + \frac{b}{a}\left(1-e^{-a\hbar}\right)u[k] = \alpha y[k] + \beta u[k]$$

hvor da $\alpha = e^{-a\hbar}$ og $\beta = \frac{b}{a}\left(1-e^{-a\hbar}\right)$.

Oppgave 2.6. Anta nå at $u(t)$ tilsvarer en rampe på tidsintervallet $[t_k, t_{k+1}]$, altså

$$u(t) = \frac{t-t_k}{t_{k+1}-t_k}(u[k] - u[k-1]) + u[k-1]$$

for $t \in [t_k, t_{k+1}] = [t_k, t_k + \hbar]$. Hva blir da parameterne α og β for det eksakte diskretiserte systemet?

Eksempel 2.5. Omtrentlig diskretisering av PID-regulator: Vi ønsker nå å diskretisere en PID-regulator:

$$u = k_P e + k_I\int_0^t e(\tau)d\tau + k_D\dot{e}.$$

Vi starter med å derivere mhp. tid på begge sider:

$$\dot{u} = k_P\dot{e} + k_I e + \ddot{e}.$$

Vi bruker så at $\dot{f}(t_k) \approx \frac{f(t_k)-f(t_k-\mathfrak{h})}{\mathfrak{h}} = \frac{f[k]-f[k-1]}{\mathfrak{h}}$ for en gitt tastetid $\mathfrak{h} = t_{k+1} - t_k$ og med notasjonen $f[k] = f(t_k)$. Dette gir

$$\dot{u}(t_k) \approx \frac{u[k]-u[k-1]}{\mathfrak{h}} = k_P \frac{e[k]-e[k-1]}{\mathfrak{h}} + k_I e[k] + k_D \frac{\dot{e}[k]-\dot{e}[k-1]}{\mathfrak{h}}.$$

Her har vi igjen at

$$\frac{\dot{e}[k]-\dot{e}[k-1]}{\mathfrak{h}} \approx \frac{\big(e[k]-e[k-1]\big)-\big(e[k-1]-e[k-2]\big)}{\mathfrak{h}^2} = \frac{e[k]-2e[k-1]+e[k-2]}{\mathfrak{h}^2}.$$

Samlet gir dette oss følgende diskretisering:

$$u[k] = u[k-1] + \left(k_P + \mathfrak{h}k_I + \frac{k_D}{\mathfrak{h}}\right) e[k] + \left(-k_P - 2\frac{k_D}{\mathfrak{h}}\right) e[k-1] + \frac{k_D}{\mathfrak{h}} e[k-2].$$

Del II

Modellering og Simulering

3. Modellering

Alternative kilder: Kap. 3 i [Bjørvik and Hveem, 2014].

I dette kapittelet skal vi se på hvordan man kan lage matematiske modeller av dynamiske systemer. Mer spesifikt, så ønsker vi å finne ett sett med ordinære differensiallikninger som beskriver systemets *dynamikk*, altså hvordan systemet (rettere sagt, dets *tilstander*) endrer seg over tid når det blir påvirket av eksterne krefter (f.eks. kontrollpådrag og forstyrrelser).

Slike modeller er selvsagt alltid forenklinger, og vil i varierende grad representere det virkelige systemet. Følgende velkjente (og nærmest obligatoriske) utsagn fanger dette godt:

«*Alle modeller er feil, men noen er nyttige*»

Så hvorfor er slike matematiske modeller så nyttige? Dynamiske modeller spiller en sentral rolle innen reguleringsteknikk, og har en rekke andre bruksområder:[a]

- **Forbedre forståelsen av prosessen:** Dynamisk modeller og datasimuleringer lar en studere prosessatferd uten å måtte «forstyrre» den ekte prosessen.
- **Bygge simulatorer:** Matematiske modeller trengs for å lage realistiske simulatorer. Et godt eksempel på dette er flytrenings-simulatorer som brukes i luft- og romfartsindustrien.
- **Utvikle reguleringsstrategier:** En matematisk modell av en prosess tillater å evaluere mulige reguleringsstrategier.

[a] Matematiske modeller er også viktig for tilstandsestimering, feildiagnose, ytelses- og robusthets-analyser, etc., etc.

Hvordan lage en matematisk modell? Det finnes tre måter å modellere ett system:

- **Teoretisk modellering:** utlede modeller fra fysikkens lover («førsteprinsipper»).
- **Empirisk modellering:** tilpasse modeller fra å eksperimenter/data.
- **Semi-empirisik modellering:** en blanding av de to forrige.

Vi skal hovedsakelig se på teoretisk modellering i dette faget, selv om også empirisk modellering vil dukke opp i form av tilpasning av en modell fra en sprangrespons (se § 5.2.4).

Teoretisk modellering: Hva er vi ute etter? Vi er ute etter å bruke såkalte grunn-/første-prinsipper (lover fra fysikken) til å utlede (ordinære[1]) differensialligninger som beskriver et systems dynamikk på en *god nok* måte. Det vil si at de gir en tilfredsstillende represetasnjon av systemet, med en passende balanse mellom enkelhet og realisme (variasjoner i tyngdekraften kan være viktig for en romrakett, men kanskje ikke fullt så mye for en pendel).

Ved å se bort fra eksterne forstyrrelser, så skal vi utlede modeller på tilstandsromform:

$$\dot{x}_1 = f_1(x_1, x_2, \ldots, x_n, u_1, \ldots, u_m), \tag{3.1a}$$
$$\dot{x}_2 = f_2(x_1, x_2, \ldots, x_n, u_1, \ldots, u_m), \tag{3.1b}$$
$$\vdots \tag{3.1c}$$
$$\dot{x}_n = f_n(x_1, x_2, \ldots, x_n, u_1, \ldots, u_m), \tag{3.1d}$$

hvor $x_i = x_i(t) \in \mathbb{R}$, $i = 1, 2, \ldots, m$, er systemets $n \geq 1$ tilstander. og $u_j = u_j(t) \in \mathbb{R}$, $j = 1, 2, \ldots, m$, $m \geq 0$, er pådragene.

Slike modeller blir som regel funnet ved hjelp av bevaringslover (også kalt konserveringslover) for en eller annen fysisk størrelse (for eksempel masse, energi. bevegelsesmengde (momentum), elektrisk ladning, vinkelmoment, etc.). Vi skal se på noen slike metoder i dette kapitellet.

3.1. Masse- og energi-balanse

Alternative kilder: [Bjørvik and Hveem, 2014].

Masse- og energibalanse baserer seg på prinsippene at enten masse eller energi (i et lukket system) forblir konstant (er bevart) hvis det ikke er noen tilførsel eller tap, samt at raten til en hver endring tilsvarer differansen mellom raten av tilførsel og tap.[2]

Hvorfor skal du lære dette? 🤷🏾‍♀️ Spesielt innen prosessindustrien finner man som oftest teoretiske modeller ved hjelp av enten masse- eller energibalanse.

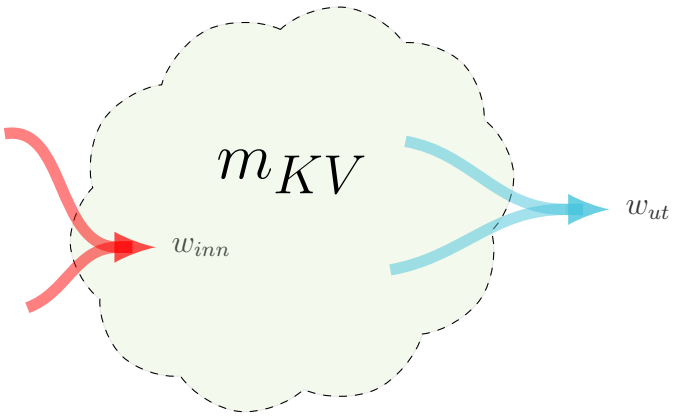
3.1.1 Massebalanse

Massebalansen tilsvarer følgende meget avanserte konsept: Hvis det i løpet av 1 minutt strømmer 10 kg med væske inn i en tank og 2 kg ut, så vil det etter 1 minutt være 8 kg mer væske i tanken i tanken enn før! 🤯

I dette eksempelet var tanken et såkalt kontrollvolum (KV). Det vil si, en (kunstig) avgrenset region med en klart definert, og som regel konstant, ytterkant. Et mer abstrakt kontrollvolum

[1] Dynamikken til en rekke systemer kan kun modelleres ved hjelp av *partielle* differensialligninger, for eksempel, myke (soft) roboter, samt varme- og Navier–Stokes ligningene for henholdsvis varmefordeling og væskestrøm. For reguleringstekniske formål kan man noen ganger approksimere disse vha. ordinære differensialligninger.

[2] For å bevare den psykiske helsen til faglærer så skal vi holde oss til systemer hvor vi ikke trenger å ta høyde for Einsteins spesielle relativitetsteori, og derav hans velkjente formell, $E = mc^2$, som gir en sammenheng mellom masse og energi.



Figur 3.1: Illustrasjon av et kontrollvolum i grønt med ytterkant gitt av den stiplende linjen. Den akkumulerte massen i kontrollvolumet ved et gitt tidspunkt er m_{kV}, mens massestrømmene w_{inn} og w_{ut} flyter henholdsvis inn og ut av kontrollvolumet.

med massestrømmer inn og ut er vist i figur 3.1. Innen dette kontrollvolumet er den akkumulerte massen (målt i kg), altså all masse i volumet, ved et tidspunkt t gitt ved $m_{KV}(t)$. Ved å se på differansen mellom de samlede masstrømmene (målt i kg/s) inn og ut av volumet, betegnet henholdsvis $w_{inn}(t)$ og $w_{ut}(t)$, så kan man finne den momentane endringen i den akkumulerte massen i volumet ved dette tidspunktet ved hjelp av følgende:

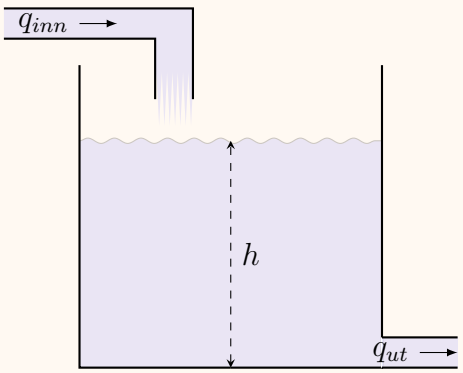
Massebalanse: For et lukket system med et gitt kontrollvolum (KV) har man

$$\underbrace{\frac{d}{dt}m_{KV}(t)}_{\text{Endringsraten til akkumulert masse}} = \underbrace{w_{inn}(t)}_{\text{rate av masse inn}} - \underbrace{w_{ut}(t)}_{\text{rate av masse ut}} . \tag{3.2}$$

La oss se på nok et eksempel relatert til en tank:

Eksempel 3.1. Væskenivået i en tank:

En enkel tank er vist i figuren til høyre. Der er q_{inn} væskestrømmen inn og q_{ut} er væskestrømmen ut, begge med enhet $[\mathrm{m}^3\,\mathrm{s}^{-1}]$. For volumet med væske i tanken (målt i kubikkmeter) bruker vi symbolet V. Det er dermed naturlig å definere tankens indre som kontrollvolumet. Hvis ρ $[\mathrm{kg/m}^3]$ er massetettheten til væsken, så er den akkumulerte massen da $m_{KV} = \rho \cdot V$, mens $w_{inn} = \rho \cdot q_{inn}$ og $w_{ut} = \rho \cdot q_{ut}$. Fra masse-balanse-ligningen (3.2) får vi dermed at



$$\frac{d}{dt}(\rho V) = \rho(q_{inn} - q_{ut}).$$

La oss anta at 1) massetettheten ρ er (tilnærmet) konstant, 2) at avstanden mellom midten av utløpet og tankens bunn er neglisjerbart i forhold til væskehøyden i tanken, og 3) at hvert horisontalt tverrsnitt av tanken har et areal A $[\mathrm{m}^2]$, Endring i høyden, h, til væsken i tanken

målt i meter er dermed

$$\dot{h} = \frac{1}{A}\left[q_{inn} - q_{ut}\right].$$

Dette er et første-ordens system (bare ett derivat) med én enkelt tilstand, nemlig høyden h.

⚠ **Massebevarelse betyr generelt sett ikke bevarelse av volum**. Dette krever antakelsen at massetettheten (til væsken eller gassen) er konstant. I realiteten endres denne f.eks. både ved endring i temperatur og trykk (se f.eks. Avogadros lov for ideelle gasser).

Men: Konstant massetetthet (inkompressibilitet) er dog en god antagelse for de fleste væsker; se f.eks. [Cengel and Cimbala, 2013]. Jobber man med gasser, derimot, er det viktig å vurdere om dette er en fornuftig antagelse eller ikke for prosessen man jobber med siden denne antagelsen, tross alt, i ganske stor grad, forenkler den matematiske modellen.

Oppgave 3.1. I eksempel 3.1 kom vi fram til differensialligningen $\dot{h} = \frac{1}{A}[q_{inn} - q_{ut}]$.

Si at q_{ut} tilsvarer strømmen ut via et åpent rør og ut i friluft. Hvorfor er da ikke

$$q_{ut} = k_v h,$$

for en eller annen kostant $k_v > 0$, en veldig realistisk antagelse?

Den vanligste modellen til en slik utstrøm er (mer om dette straks)

$$q_{ut} = k_v \sqrt{h}.$$

Hvorfor er dette en mer realistisk modell?

Hint: Hva er løsningene på differensialligningene hvis $q_{inn} = 0$?

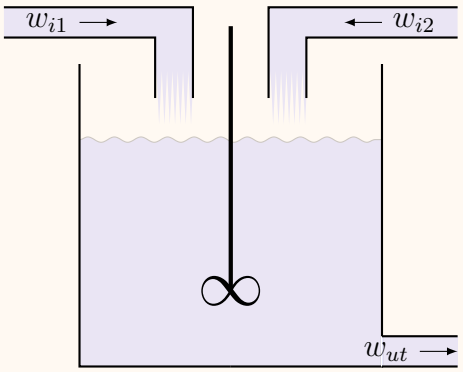
La oss se på et annet eksempel som er hakket mer komplisert:

Eksempel 3.2. Blandetank:

Vi skal nå finne de dynamiske ligningene som beskriver endringen til blandeforholdet mellom to stoffer, stoff F (f.eks. fanta) og stoff R (f.eks. rosévin), i en enkel blandetank. Blandetanken er vist i figuren til høyre, hvor

w_{i1}, w_{i2}, w_{ut} er massestrømmer [kg/s];

b_{F1}, b_{F2}, b_F er blandingsforholdet til stoff F for henholdsvist de to innløpene og utløpet.



La m [kg] betegne den akkumulerte massen i tanken, som igjen tilsvarer $m = \rho V$ hvor

ρ kg m^{-3}] er massetettheten til den blandede væsken og V er volumet (i kubikkmeter). Massebalanse-ligningen (3.2) gir oss dermed

$$\dot{m} = \frac{d}{dt}(\rho V) = w_{i1} + w_{i2} - w_{ut}. \tag{3.3}$$

Spørsmål: Før vi går videre, funder gjerne litt på følgende spørsmål: 🤔

1. Hva er aktuelle tilstandsvariabler? Og hvordan kan vi måle disse?
2. Hva er mulige pådrag?
3. Hva er mulige forstyrrelser?
4. Er konstant massetetthet en fornuftig antagelse?

En «fasit» er gitt i slutten av eksemplet.

Vi er også interessert i å finne hvordan blandingsforholdet b_F til stoff F endrer seg. Vi vet at den totale masseandel for stoff F i tanken er $m_F = V\rho b_F = mb_F$, og dermed

$$\dot{m}_F = \frac{d}{dt}(mb_F) = \dot{m}b_F + m\dot{b}_F = w_{i1}b_{1F} + w_{i2}b_{2F} - w_{ut}b_F,$$

hvor vi har brukt produktregelen. Ved å trekke fra $\dot{m}b_F$ på begge sidene og så sette inn for $\dot{m}$ fra (3.3), så finner vi at

$$m\dot{b}_F = w_{i1}(b_{1F} - b_F) + w_{i2}(b_{2F} - b_F).$$

Dette kan vi igjen skrive som

$$\dot{b}_F = \frac{1}{m}\left[w_{i1}(b_{1F} - b_F) + w_{i2}(b_{2F} - b_F)\right]. \tag{3.4}$$

La oss nå anta følgende:

- tankens tverrsnittsareal, A, er konstant;
- utstrømmen w_{ut} er gitt av $w_{ut} = c_{ut}\sqrt{\rho g h}$, hvor h er væskehøyden i tanken.

Disse antagelsen betyr jo at $m = (\rho V) = \rho A h$ og $w_{ut} = c_{ut}\sqrt{mg/A}$.

Vi har dermed to naturlige kandidater til tilstander, nemlig $x_1 = m$ og $x_2 = b_A$. Deres dynamiske ligninger kan skrives på tilstandsromform som:

$$\dot{x}_1 = w_{i1} + w_{i2} - c_{ut}\sqrt{g/A}\sqrt{x_1} \tag{3.5a}$$

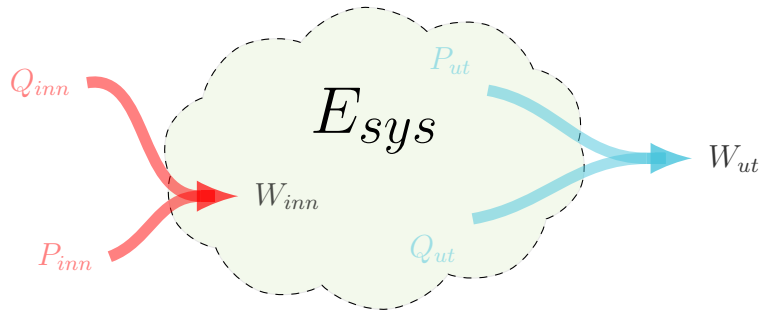
$$\dot{x}_2 = \frac{1}{x_1}\left[w_{i1}(b_{1F} - x_2) + w_{i2}(b_{2F} - x_2)\right]. \tag{3.5b}$$

Fasit: Svar på tidligere spørsmål:

1. Det er mange kandidater for prosessvariabler (og tilhørende målinger): for væskemengden er aktuelle kandidater massen m (veie tanken), høyden h eller volumet V (måle trykket i bunnen); for blandingsforholdet, så er blandingsforholdet b_F kanskje mulig (men hvordan måle det?), eventuelt kan man bruke tettheten ρ hvis denne endrer seg med blandingen (f.eks. via vekt- og volummålinger).
2. Pådragene er relatert til eventuelle reguleringsventiler eller pumper ved inn eller utstrømmene, og kanskje også konsentrasjonene inn. Når det gjelder ventilene, kan vi dermed se på pådraget som Strømningen w ut (evt. ventil åpningen l (evt.evt. spenning tilført reguleringsventilen)).
3. Forstyrrelser kan være blandingsforhold eller en en inn-/utstrøm uten måling.
4. Nei, det er det trolig ikke, siden vi blander to stoff med muligens vidt forskjellige tettheter slik at massetettheten til blandingen kan variere.

3.1.2 Energi-balanse

Energibalanse er egentlig akkurat som massebalanse, bare at vi bytter akkumulert masse i en kontrollvolum (nå også kalt et *lukket system*) med «akkumulert energi», samt bytter massestrømmer med «energi-strømmer» slik som vist i figur 3.2.



Figur 3.2: Illustrasjon av kontrollvolum (KV) for energibalanse: Energien «lagret» i kontrollvolumet (systemet) ved et gitt tidspunkt, $E_{sys}(t)$, har en endringsrate tilsvarende differansen mellom ratene av energi inn og ut.

Energibalanse følger direkte fra termodynamikkens lover (se, f.eks., SNL):

- **Termodynamikkens 1. hovedsetning**: Energi kan ikke forsvinne, men bare gå over fra en form til en annen.
- **Termodynamikkens 2. hovedsetning**: Overføring av varme skjer fra et sted med høyere temperatur til et sted med lavere temperatur («entropi øker med stor sannsynlighet»).

Vi kan bruke energibalanse til å utlede dynamiske modeller ved hjelp av følgende:

Energi-balanse: For et lukket system (SYS) med et gitt kontrollvolum (KV) har man

$$\underbrace{\frac{dE_{sys}}{dt}}_{\text{Endringsraten til et systems energi}} = \underbrace{W_{inn}}_{\text{rate av energi (altså effekt) inn i systemet}} - \underbrace{W_{ut}}_{\text{rate av energi ut av systemet}} \tag{3.6}$$

I boksen over betegner da E_{sys} energien (målt i Joule=J=kgm^2/s^2) «lagret» i kontrollvolumet (se fig. 3.2), mens W_{inn} og W_{ut} betegner energien som er tilført kontrollvolumet (det lukkede systemet) per sekund (målt i Watt = W=J/s).

For eksempel, hvis kontrollvolumet består av et medium med med en konstant masse og uniform (altså likt fordelt) temperatur T og konstant spesifikke varmekapasitet C [J K^{-1} kg^{-1}], så vil en endring i systemets varmeenergi, $\Delta E_{sys} = E_{sys} - E_0$, være relatert til en endring i dets temperatur, $\Delta T_{sys} = T_{sys} - T_0$, via

$$\Delta E_{sys} = C_{sys} \cdot \Delta T_{sys} \tag{3.7}$$

hvor C_{sys} [J/K] er systemets spesifikke varmekapasitet skalert med massen i systemet, altså $\Delta E_{sys} = C \cdot m \cdot \Delta T_{sys}$ hvor m er massen i kilogram. Siden E_0 og T_0 er konstant, gir dette

$$\dot{E}_{sys} = C_{sys} \cdot \dot{T}_{sys}.$$

For vår del vil vi ofte bruke følgende form:

$$\frac{d}{dt}E_{sys} = P_{inn} + Q_{inn} - P_{ut} - Q_{ut} \tag{Energibalanse}$$

hvor

- P_{inn} og P_{ut} er henholdsvis raten av arbeid utført på volumet og av volumet;
- Q_{inn} og Q_{ut} er henholdsvis varmestrømmer inn og ut av kontrollvolumet.

Dette er altså energibalansen for systemet/ kontrollvolumet. Arbeidet representerer en overgang fra og til mekanisk energi. Man kan også definere f.eks. P_{inn} som tilført elektrisk energi som utløser friksjonsvarme i en motstand. Alternativt kan man bare definere denne friksjonsvarmen som tilført varme Q_{inn}.

Relevante størrelser:
Mekanisk effekt (arbeid per tidsenhet):

- $P_{translasjon} = F \cdot v$ hvor F er kraft og v er hastighet;
- $P_{rotasjon} = \tau \cdot \omega$ hvor τ er et dreiemoment ω en vinkehastighet;
- $P_{trykk} = p \cdot q$ hvor p er et trykk og q en volumstrøm.

Elektrisk effekt:

- $P_{elektrisk} = U \cdot I$ hvor U er er spenning og I er strøm.

Varmeoverføring og Newtons kjølelov

Alternative kilder: YouTube-video; Brian Douglas video; Wikipedia.

Energi i form av varme kan overføres på tre måter, nemlig via konveksjon, stråling, og konduksjon/varmledning. Vi skal hovedsakelig fokusere på sistnevnte.

Varmeover-føringen/-ledning mellom kalde til varme deler av et objekt, samt fra et medium til et annet kan beskrives av Fouriers lov[3]. Vi skal se på en enkel variant av denne loven som ofte antas i forbindelse med energy-balanse, nemlig Newtons avkjølingslov:

Newtons avkjølingslov: Varmeoverføring mellom to medium er direkte proporsjonal med temperaturforskjellen:

$$Q = h \cdot A \cdot \Delta T$$

hvor

ΔT : temperaturdifferansen [K].

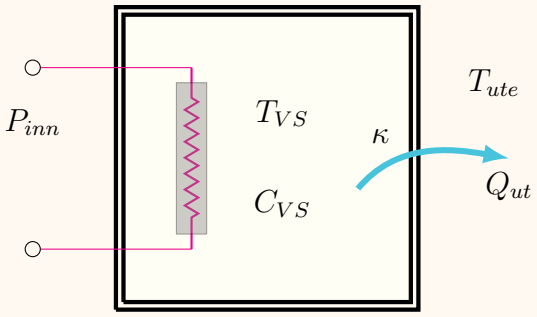
Q : effekten tilsvarende varmeoverføringen [W] ;

h : varmeovergangskoeffisienten [W/m^{2}K];

A : arealet mellom mediumene [m^2];

Eksempel 3.3. Varmeskap:

Et varmeskap (VS) varmes opp elektrisk gjennom et varmeelement; se figuren til høyre. Varme lagres på grunn av skapets varmekapasitet. I tillegg kommer et varmetap til omgivelsene.



En naturlig kandidat for kontrollvolum er naturlig nok innsiden av varmeskapet. Energibalanse gir dermed:

$$\frac{d}{dt}E_{VS} = P_{inn} - Q_{ut}.$$

La T_{VS} betegne temperaturen i varmeskapet, og la T_{ute} betegne temperaturen utenfor, som vi antar holder seg tilnærmet konstant. Fra Newtons kjølelov har vi dermed

$$Q_{ut} = \kappa \cdot (T_{VS} - T_{ute}),$$

hvor $\kappa = h \cdot A$ der h er varmeovergangskoeffisienten og A er skapets overflateareal.

Siden $E_{VS} = C_{VS}T_{VS}$, hvor C_{VS} er varmekapasiteten til mediumet i varmeskapet, har vi

[3]Tilsvarende Navier-Stokes-ligningen for væskeflyt, har man en kjent partiell-differensialligning for varmeoverføring, nemlig varmeligningen.

dermed følgende fra energibalansen:

$$\dot{E}_{VS} = C_{VS}\dot{T}_{VS} = P_{inn} - \kappa \cdot (T_{VS} - T_{ute}) \quad \Longrightarrow \quad \dot{T}_{VS} = \frac{1}{C_{VS}} \left[P_{inn} - \kappa(T_{VS} - T_{ute})\right].$$

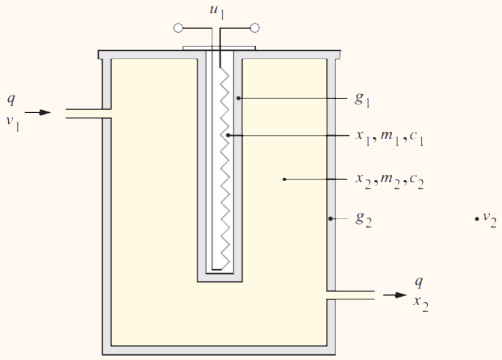
Spørsmål: Hva er naturlige tilstander, utganger, pådrag og forstyrrelser? Og hva er tilstandsromformen?

La oss ta et eksempelet som er hakket mer komplisert.

Eksempel 3.4. Varmtvannsbereder: (Eks. og figur fra 2.1 i [Balchen et al., 2016])

Gitt varmtvannsberederen vist i figuren til høyre, hvor

u_1: elektrisk effekt tilført kolben [kW];
q: væskestrøm [m^3/s];
x_i, v_i: temperaturer [K]
m_i: masser [kg]
c_i: spesifikk varmekapasitet [kJ/kgK];
g_i: Varmeovergangstallet [kW/K];
ρ: massetettheten til vannet [kg/m^3].



Antagelse: lik varmefordeling i tanken.

La oss først se på energibalanse for varmekolben. Tilført energi kommer fra pådraget, u_1, mens vi antar at et "tap" av energi til tanken er proporsjonalt med temperaturforskjellen, altså tapet er $g_1(x_2 - x_1)$. Dette gir os følgende differensialligning for energi-endringen i varmekolben:

$$\dot{E}_1 = u_1 - g_1(x_1 - x_2).$$

Her har vi igjen fra Newtons avkjølingslov at $\dot{E}_1 = m_1 c_1 \dot{x}_1.$, slik at temperaturenendringen i varmekolben er gitt ved

$$\dot{x}_1 = \frac{1}{m_1 c_1} \left(u_1 - g_1(x_1 - x_2)\right).$$

For tanken kan energi flyte til og fra varmekolben og ut av tanken, samt tilføres fra det kalde vannet fra innløpet og tappes fra utløpet. Dette gir oss følgende energibalanse:

$$\dot{E}_2 = g_1(x_1 - x_2) - g_2(x_2 - v_2) + q\rho(c_2 v_1 - c_2 x_2).$$

Tilsvarende varmekolben, har vi at energiendringen i tanken er proporsjonal med temperaturøkningen: $\dot{E}_2 = m_2 c_2 \dot{x}_2$, og dermed

$$\dot{x}_2 = \frac{1}{m_2 c_2} \left(g_1(x_1 - x_2) - g_2(x_2 - v_2) + q\rho c_2(v_1 - x_2)\right).$$

3.2. Elektro-mekaniske systemer

Du vil lære mer om dette i IFYT1002 - Fysikk

Hvorfor skal du lære dette? 🤷 Som navnet tilsier, så er **elektromekaniske systemer** bygd opp av mekaniske og/eller elektriske komponenter. Eksempler på slike systemer inkluderer roboter og elektriske kjøretøy. Nuff said...

3.2.1 Newtons andre lov (kraftbalanse)

Alternative kilder: Wikipedia; SNL; En hvilken som helst fysikkbok.

Newtons andres lov gir en enkel formell for å beskrive dynamikken til én enkelt punktmasse (altså et objekt hvor man antar at all dens masse er lokalisert på ett enkelt punkt). Denne loven kan tenkes på som «kraft-balanse», som igjen egentlig stammer fra en bevaringslov, nemlig bevaring av bevegelsesmengde.

Newtons andre lov: Endringen i *bevegelsesmengden* p til en punktmasse er lik summen av kreftene, $\sum_i F_i = F_1 + F_2 + \cdots$, som virker på objektet: $\dot{p} = \sum_i F_i$. Når massen, m, til objektet ikke endrer seg over tid (hvis den kan endre seg, se denne), tilsvarer dette:

$$ma = m\frac{dv}{dt} = m\dot{v} = \sum_i F_i$$

hvor v er objektetes hastighet og $a = \frac{dv}{dt}$ dets akselerasjon.[a]

[a] **Merk:** p, v og F_i er her normalt sett vektorer i $\mathbb{R}^k$ for $k \in \{1, 2, 3\}$.

For å utlede bevegelsesligningene til flere enkle mekaniske systemer så er følgende størrelser høyst relevante:

Relevante størrelser:

Lineær fjær: $F_k = k \cdot \Delta p$ (Hookes lov) og $E_k = \frac{1}{2}k\Delta p^2$, hvor k [$\mathrm{N\,m^{-1}}$] er fjærkonstanten og Δp [m] er komprimeringen av fjæren fra utgangsposisjonen.

Lineær demper: $F_d = d \cdot v$ hvor d [$\mathrm{N\,s\,m^{-1}}$] er dempningskonstanten og v [m/s] er komprimeringshastigheten til demperen.

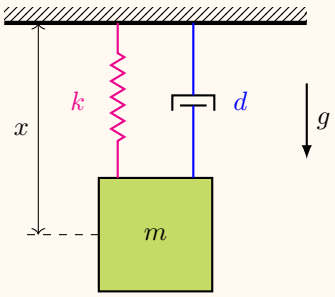
Kinetisk energi: $\mathcal{K} = \frac{1}{2}m \cdot v^2$ hvor m [kg] er massen og v [m/s] er hastigheten til et legeme.

Potensiell energi: $\mathcal{P} = m \cdot g \cdot h$ hvor m [kg] er massen, g ($\approx 9.81\,\mathrm{m/s^2}$) er akselerasjonen fra tyngdekraften , og h [m] høyden et legeme fra et referansepunkt.

La oss ta en titt på en meget klassisk eksempel, nemlig et masse-fjær-demper system:

Eksempel 3.5. Hengende masse-fjær-demper:

Et hengende masse-fjære-demper system er vist i figuren til høyre. Det består av en hengende masse (en boks) som er festet til et tak via en lineær fjær og en lineær demper. Boksen har masse m [kg], fjæren har fjærkonstant k [N/m] og demperen har dempningskonstant d [Ns/m]. Boksens avstand målet fra taket betegnes med x [m]. Utgangsposisjonen til fjæren tilsvarer posisjonen $x = x_*$, det vil si posisjonen hvor systemet ville vært i likevekt hvis det ikke ble påvirket av akselerasjonen fra tyngdekraften, g.



Ved å bruke utrykket for en lineær fjær gitt over (Hookes lov), så vet vi at kraften som virker på kassen fra fjæren er $F_k = -k \cdot (x - x_*)$, mens kraften fra demperen er $F_d = -d \cdot \dot{x}$. Vi må også ta med tyngekraften, $G = mg$, hvor g ($\approx 9.81\,\mathrm{m/s^2}$) er akselerasjonen fra tyngekraften, siden kassens posisjon x ikke er målt relativt til fjærens utgangsposisjon.

Kraftbalanse fra Newtons andre lov gir dermed

$$ma = m\ddot{x} = \sum_i F_i = F_k + F_d + G = -k(x - x_*) - d\dot{x} + mg.$$

Anta at $m = 2$, $k = 10$ og $d = 3$, samt $g = 10$ og $x_* = 1$. Hva er da likevektspunktet til systemet? For å svare på dette, la oss først ta $x_1 = x$ og $x_2 = \dot{x}$ slik at vi kan skrive systemet om på tilstandsromform:

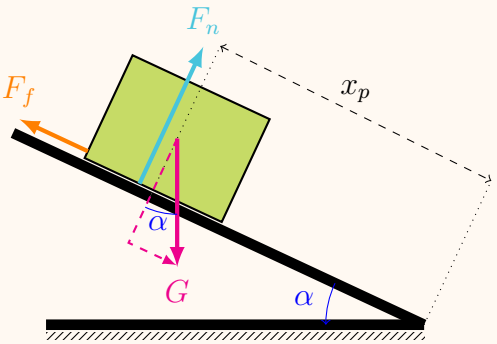
$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -k \cdot (x_1 - x_*) - d \cdot x_2 + mg. \end{aligned}$$

Likevektspunkter tilsvarer punkter $(\bar{x}_1, \bar{x}_2)$ slik at $\dot{x}_1 = \dot{x}_2 = 0$. Dermed har vi $\bar{x}_2 = 0$, mens fra den andre finner vi dermed at $\bar{x}_1$ må tilfredsstille

$$-10 \cdot (\bar{x}_1 - 1) + 2 \cdot 10 = 0 \quad \Longrightarrow \quad \bar{x}_1 = \frac{20 + 10}{10} = 3.$$

Eksempel 3.6. Kasse på en skråning:

Gitt en kasse med vekt m [kg] som sklir ned en skråning med konstant helning α (se figur til høyre). La x_p betegne posisjonen til boksen målt parallelt med den skrå bakken, og la $x_h = \dot{x}_p$ betegne dens hastighet. Tilstandene er dermed x_p og x_h.



Anta at kassen påvirkes av kinetisk friksjon, gitt ved[a] $F_f = \mu \cdot F_n$, hvor F_n er normalkraften mellom bakken og kassen, og hvor μ er friksjonskoeffisienten. Fra Newtons andre lov har vi dermed

$$m\dot{x}_h = G_{\parallel} - F_f = G_{\parallel} - \mu F_n,$$

hvor $G_{\parallel} = G\sin(\alpha) = mg\sin(\alpha)$ er tyngekraften som virker på kassen og $g = 9.81\,\mathrm{m\,s^{-2}}$ er gravitasjons akselerasjonen. Siden normalkraften er lik $F_n = |G_{\perp}| = mg\cos(\alpha)$, er de dynamiske ligningen for systemet som følger:

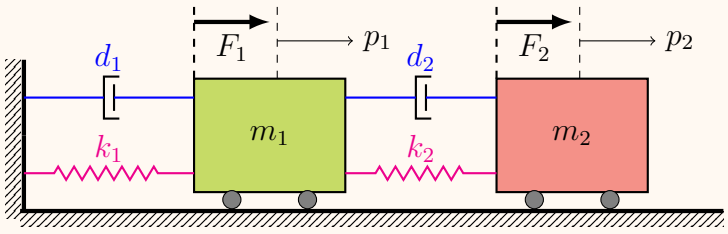
$$\begin{aligned} \dot{x}_p &= x_h \\ \dot{x}_h &= g\left[\sin(\alpha) - \mu\cos(\alpha)\right]. \end{aligned}$$

[a] Merk at vi også bare kan ha $|F_f| \leq |G_{\parallel}|$, eller vil jo friksjonen kunne få kassen til å bevege seg oppover!

Fun facts, bemerkninger og annet dill dall (you may skip)

Friksjon

Eksempel 3.7. Koblede masse-fjær-demper systemer: Gitt to koblede masse-fjære-demper systemer som vist i figuren under.



Trallene har henholdsvis masse lik m_1 og m_2, og påføres hver sin kraft ([N]), F_1 og F_2. Fjærene, som tilfredsstiller Hookes lov med fjærkonstanter k_1 og k_2, er i hvileposisjon for når henholdsvis $p_1 = 0$ og $p_1 - p_2 = 0$.

La $h_i = \dot{p}_i$ og $a_i = \dot{v}_i$. Kraftbalanse gir:

$$\begin{aligned} m_1 a_1 &= F_{k_1} + F_{d_1} + F_1 - F_{k_2} - F_{d_2} \\ m_2 a_2 &= F_2 + F_{k_2} + F_{d_2} \end{aligned}$$

Her er

$$\begin{aligned} F_{k_1} &= -k_1 p_1 \\ F_{d_1} &= -d_1 h_1 \\ F_{k_2} &= k_2(p_1 - p_2) \\ F_{d_2} &= d_2(h_1 - h_2) \end{aligned}$$

Ved å ta $(x_1, x_2, x_3, x_4) = (p_1, p_2, h_1, h_2)$ og $(u_1, u_2) = (F_1, F_2)$ kan vi bruke *matriseformen*:

$$\underbrace{\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{(k_1+k_2)}{m_1} & \frac{k_2}{m_1} & -\frac{(d_1+d_2)}{m_1} & \frac{d_2}{m_1} \\ \frac{k_2}{m_2} & -\frac{k_2}{m_2} & \frac{d_1}{m_2} & -\frac{d_2}{m_2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}}_{\mathbf{Ax}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{m_1} & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}_{\mathbf{Bu}}.$$

3.2.2 Rotasjonsdynamikk (momentbalanse)

Alternative kilder: Wikipedia.

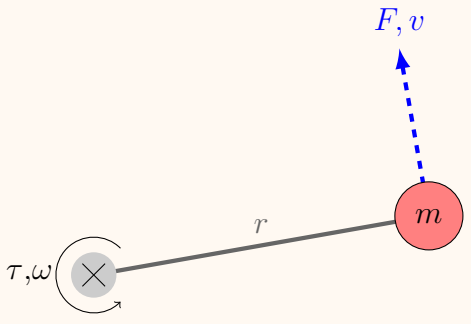
Hva med roterende legemer? Tilsvarende hvordan Newtons andre lov i tar for seg bevaring av et objekts bevegelsesmengde, kan man også se på bevaringen av vinkelmoment for roterende objekter. Mer spesifikt, ved å introdusere det roterende legemets *treghetsmoment* (tenk dets «roterende masse»), leder Newtons andre lov til Eulers rotasjonslingning[4] (se Wikipedia).

Før vi hopper til denne ligningen, så starter vi med et eksempel hvor vi introduserer viktige størrelser.

Eksempel 3.8. Roterende stang:

Vi skal nå se hvordan man kan relatere kraft og hastighet til henholdsvis dreiemoment og vinkelhastighet.

Gitt systemet vist i figuren til høyre, bestående av en masse som er festet til enden av en stang. Stangen kan fritt rotere om en aksling. Anta at massen m [kg] er plassert ytterst på stangen/armen som har lengde r [m] (vi antar at stangen selv ikke har noen vekt).



Kraften F [N] som virker på massen (vinkelrett i forhold til stangen) genererer et **dreiemoment**, τ [N m] (målt positivt mot klokken), gitt ved

$$\tau = r \cdot F.$$

Hvis massen beveger seg med en hastighet v [m/s] som vist i figuren, så er tilsvarende

[4] Denne ligningene predikere faktisk den funky oppførselen du ser i https://youtu.be/1n-HMSCDYtM.

vinkelhastighet. ω [rad/second], gitt ved

$$\omega = \frac{v}{r},$$

eller tilsvarende $v = r \cdot v$. Hvis lengden, r, til armen er kostant har vi dermed

$$\dot{v} = r \cdot \dot{\omega}.$$

Fra Newtons 2. lov vet vi jo at $m\dot{v} = F$ (hvis F er summen av kreftene som virker på massen). Ved å sette inn for $\dot{v}$ og F får vi

$$m\underbrace{r \cdot \dot{\omega}}_{=\dot{v}} = \underbrace{\frac{\tau}{r}}_{=F} \quad \Longrightarrow \quad J\dot{\omega} = \tau,$$

hvor

$$J = m \cdot r^2 \ [\mathrm{kg\,m^2}]$$

er **treghetsmomentet**(også kalt kraftmoment) til systemet/legemet. Den kinetiske energien til systemet er dermed

$$\mathcal{K} = \frac{1}{2}mv^2 = \frac{1}{2}m(r\omega)^2 = \frac{1}{2}(mr^2)\omega^2 = \frac{1}{2}J\omega^2.$$

Man kan derfor tenke på treghetsmomentet J som «massen» ved rotasjonsbevegelser, altså tilsvarende som m brukes for translasjonsbevegelser.

La oss nå ta det i mer detalj, ved å se på Eulers ligning:

Eulers ligning for roterende objekter (momentbalanse): La $\omega \in \mathbb{R}$ betegne rotasjonshastighet til et stivt legeme om én enkelt akse, og la $J \in \mathbb{R}$ være legemets treghetsmoment om rotasjonsaksen. Da har man

$$J\dot{\omega} = \sum \tau, \tag{3.8}$$

hvor $\sum \tau$ er summen av eksterne dreiemoment som virker på objektet.

Relevante størrelser: (holder for små $\Delta\theta$, hvor r [m] er radius/arm fra rotasjonsaksen).[a]

Lineær fjær: $\tau_f \approx k_f \cdot r^2 \Delta\theta$ hvor k_f [$\mathrm{N\,m^{-1}}$] er fjærkonstanten og $\Delta\theta$ [m] er komprimeringen av fjæren fra en referansesvinkel i radianer.

Lineær torsjon: $\tau_t = k_t \cdot \Delta\theta$ hvor k_t [$\mathrm{N\,m\,rad^{-1}}$] er torsjonsonstanten og $\Delta\theta$ [m] er rotasjonen fra et hvilepunkt.

Lineær demper: $\tau_d = d \cdot r^2\omega$ hvor d [$\mathrm{N\,s\,m^{-1}}$] er dempningskonstanten og ω [m/s] er vinkelkomprimeringshastigheten til demperen.

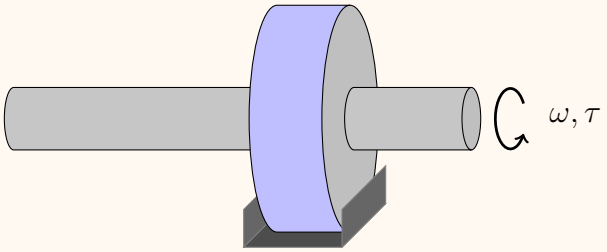
Kinetisk energi: $\mathcal{K}_r = \frac{1}{2}J \cdot \omega^2$ hvor J [$\mathrm{kg\,m^2}$] er treghetsmomentet og ω [rad/s] er vinkelhastiget til det roterende legemet.

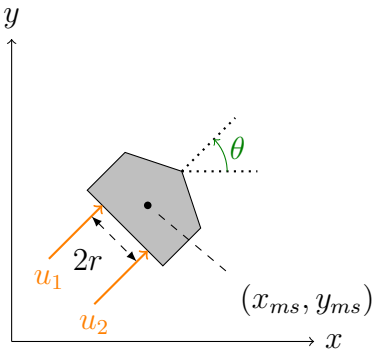
[a] Bruker at et utslag $\Delta p = r\sin(\Delta\theta) \approx r\Delta\theta$ for små $\Delta\theta$.

Eksempel 3.9. Svinghjul med friksjon:

Gitt et svinghjul med treghetsmoment J. Anta at mekanismen er utsatt for viskøs friksjon proporsjonal med vinkelhastigheten ω, altså $\tau_f = d\omega$ hvor d er dempningskonstanten. Anta videre at pådraget vårt (eller forstyrrelsen) er et påført dreiemoment τ. Eulers ligning (momentbalanse) gir dermed

$$J\dot{\omega} = \tau - d\omega.$$





Figur 3.3: Hovercraft i planet.

Oppgave 3.2. Hovercraft: Figur 3.3 viser et hovercraft sett ovenfra. La (x_{ms}, y_{ms}) betegne posisjonen til massesenteret ti farkosten: La dens masse være m og la treghetsmomentet om massesenteret være J. Rettningsvinkelen θ (gitt i radianer) er målt positiv mot klokken fra x-aksen.

Det er to store vifter som kan brukes til å styre farkosten. Disse har samme avstand til massesenteret, og det $2r$ meter mellom dem. Vi bruker u_1 og u_2 til å betegne kraften fra disse viftene som virker på hovercraften (se figuren).

Finn de dynamiske bevegelsesligningene til dette systemet og skriv på dem på tilstandsromform, altså som et sett av førsteordens differensialligninger.

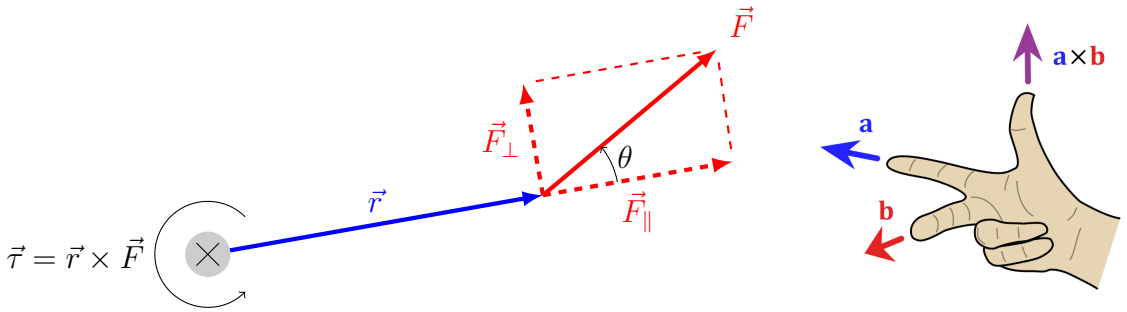
Dreiemoment og treghetsmoment*

La oss se litt nærmere på størrelsene dreiemoment og treghetsmoment.

Dreiemomentet (og kalt kraftmomentet) $\vec{\tau}$ (SI-enheten er N m) er lik kryssproduktet av armen $\vec{r}$ og kraften $\vec{F}$ altså (se også figur 3.4)

$$\vec{\tau} = \vec{r} \times \vec{F}.$$

For å regne ut dreiemomentet, bør man derfor generelt sett tenke på alle størrelser som vektorer i rommet $\mathbb{R}^3$. Ofte kan dette dog forenkles:



Figur 3.4: **Venstre:** En kraft $\vec{F}$ påføres en stang en avstand $\vec{r}$ fra stangens feste. Kraften fører til et dreiemoment $\vec{\tau} = \vec{r} \times \vec{F} = \|\vec{r}\| \left\|\vec{F}_\perp\right\| \vec{n} = \|\vec{r}\| \left\|\vec{F}\right\| \sin(\theta)\vec{n}$ om festet, hvor $\vec{n}$ er en enhetsvektor som er normal på både $\vec{r}$ og $\vec{F}$. **Høyre:** Høyrehåndsregelen for utregning av kryssprodukt; figur fra SNL.

Eksempel 3.10. Gitt en stang av lengde r slik som vist i figur 3.4. Det virker der en kraft $\vec{F}$ vinkelrett på enden av stangen, slik at $\vec{F}_\parallel = 0$. Mer spesfikt, la

$$\vec{r} = \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} \quad \text{og} \quad \vec{F} = \begin{bmatrix} 0 \\ F \\ 0 \end{bmatrix}$$

og dermed

$$\vec{\tau} = \vec{r} \times \vec{F} = \begin{bmatrix} 0 \\ 0 \\ r \cdot F \end{bmatrix}.$$

Det virker derfor et dreiemoment om enhetsvektoren $\vec{n} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ med magnitude $\tau = r \cdot F$.

Treghetsmomentet[5] **J** (SI-enheten er $\mathrm{kg\,m^2}$) kan sees på som det som tilsvarende til masse for rotasjoner. La oss fort vise for denne kvantiteten kommer fra:

Gitt en stang, som den i rødt i figur 3.4, som er $l = \|\vec{r}\|$ lang. Den totale massen til stangen er m, men denne er ikke jevnt fordelt; i stedet kan vi tenke oss at massen til systemet er bygget opp av en begrenset mengde av massepartikler, hver med masse m_i en avstand r_i fra rotasjonsaksen. Dreiemomentet er da gitt ved

$$J = \sum_i r_i \cdot m_i.$$

Enda mer presist: la massen til et hvert tverssnitt langs stangen være gitt av $\rho(\cdot)$, slik at den totale massen er $m = \int_0^l \rho(r)dr$. Anta nå at stagen roterer om sitt feste med en vinkelhastighet ω $[\mathrm{rad\,s^{-1}}]$. Hastigheten, $v_p(r)$ $[\mathrm{m\,s^{-1}}]$, til et punkt på stagen som er en avstand

[5] Du finner en liste med treghetsmomentene til flere vanlige former på Wikipedia.

r fra rotasjonsaksen er $v_p(r) = r \cdot \omega$. Den kinetiske energien tilsvarende dette punktet er dermed $E_p(r) = \frac{1}{2}\rho(r)v_p^2(r)$, slik at den totale kinetiske energien til stangen er

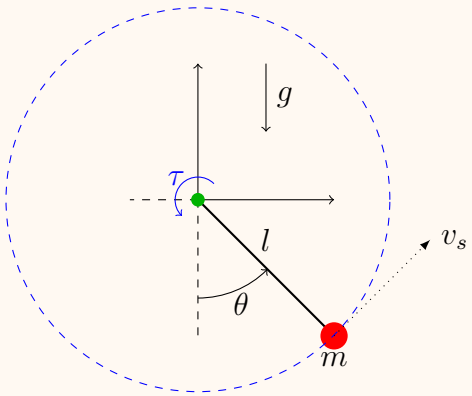
$$E = \int_0^l \frac{1}{2}\rho(r)v_p^2(r)dr = \frac{1}{2}\left[\int_0^l \rho(r)r^2 dr\right]\omega^2 =: \frac{1}{2}J\omega^2.$$

Altså er treghetsmomentet til legemet:

$$J := \int_0^l \rho(r)r^2 dr.$$

Eksempel: Anta at en stang av lengde l har jevn (og dermed konstant) massefordeling, slik at $\rho(r) = (m/l)$. Vi finner da treghetsmomentet til å være $J = \int_0^l \rho(r)r^2dr = \int_0^l (m/l)r^2dr = (m/l)\int_0^l r^2dr = \frac{1}{3}ml^2$.

Eksempel 3.11. (En enkel pendel) Vi skal nå utlede den dynamiske ligningen til den enkle pendelen vist i figuren under.



Metode 1 – Eulers ligning: Anta at all massen til pendelen er lokalisert om et enkelt punkt på enden (merket rødt i figuren). Med andre ord antar vi at stangen ikke har masse. Siden stangen ikke har noen masse, har vi fra parallellakseteoremet at treghetmomentet til pendelen er målt om rotasjonssenteret (se den grønne sirkelen) gitt ved $J = ml^2$ (se også denne listen).

Husk at dreiemoment er kryssproduktet av kraft og arm. Den eneste eksterne kraften som virker på pendel er tyngdekraften, gitt ved $\vec{G} = [0, -mg, 0]^\mathsf{T}$. Armen fra rotasjonssenteret til pendelens massesenter er $\vec{r} = l[\sin(\theta), -\cos(\theta), 0]^\mathsf{T}$. Dermed er dreiemoment forårsaket av tyngdekraften

$$\vec{\tau} = \vec{r} \times \vec{G} = [0, 0, -mgl\sin(\theta)].$$

Sinden vinkelhastigheten til pendellen er $\vec{\omega} = [0, 0, \dot{\theta}]^\mathsf{T}$, har vi fra Eulers ligning at

$$\ddot{\theta} = -\frac{g}{l}\sin(\theta).$$

Metode 2 – Newtons andre lov: La oss vise at Newtons andre lov resulterer i den samme bevegelsesligningen. Vi begynner ved å legge merke til at enden av pendelen bare kan bevege seg langs den blå, stiplede sirkelen vist i figuren. Vi vet derfor at endringen i hastigheten, $v_s = l\dot{\theta}$, til pendelen langs denne sirkelen er gitt av Newtons andre lov, hvor den eneste kraften som virker på den er tyngdekraften. Vi har dermed at

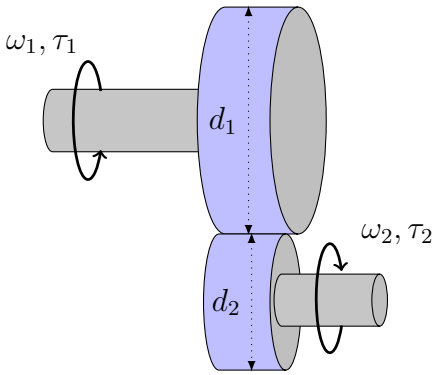
$$m\frac{d}{dt}v_s = (-mg\sin(\theta)) \quad \Longrightarrow \quad \ddot{\theta} = -\frac{g}{l}\sin(\theta).$$

Her har vi brukt at hastigheten til enden av pendelen er $\vec{v}_p = v_s[\cos(\theta), \sin(\theta)]^\mathsf{T}$, slik at effekten av tyngekraften på endringen til v_s er gitt av det indre produktet $\vec{G} \cdot \vec{v}_p / v_s = -mg\sin(\theta)$.

3.2.3 Gir (ideelle)*

Alternative kilder: Wikipedia.

Hvorfor skal du lære dette? 🤷🏾‍♀️ Gir er viktige komponenter i mange mekaniske og elektromekaniske systemer. Det er derfor viktig å kunne modellere disse, vite hensikten med, samt å ha en intuitiv forståelse av den grunnlegende virkemåten.



Figur 3.5: Illustrasjon av et ideelt gir, bestående av to (tann)-hjul med forskjellig diamenter.

Det finnes flere måter å implementere gir på, for eksempel vha. tannhjul. I virkeligheten fører dette til mange tap fra f.eks. friksjon og merkverdige ulineariteter som «backlash». Vi skal dog holde oss til *ideelle* (tapsfrie) gir.

Hensikten med et gir er som regel å enten øke/redusere rotasjonshastigheten og/eller redusere/øke dreiemomentet. Den viktigste intuisjonen her er at for eksempel en økning i rotasjonshastigheten over giret fører til en reduksjon i dreiemomentet, og vice versa:

Giroverføring av vinkelhastighet og dreiemoment:

Gitt et ideelt gir som vist i figur 3.5. Anta at $n = \frac{d_1}{d_2} = \frac{r_1}{r_2}$, hvor r_1 og r_2 er positive heltall.[a]

Vinkelhastighet: Vi har at hastighetene på enden av (tann-)hjulene er $v_1 = r_1\omega_1$ og

$v_2 = r_2\omega_2$. Siden vi antar ideelle gir, hvor det ikke er noen form for tap (de glipper blant annet ikke), så må vi ha $v_1 = v_2$, som igjen betyr $\omega_2 r_2 = \omega_1 r_1$, og dermed

$$\omega_2 = n\omega_1.$$

Dreiemoment: Anta at ω_1 holder seg konstant. Kreftene på enden av (tann-)hjulene er da gitt ved $F_1 = \tau_1/r_1$ og $F_2 = \tau_2/r_2$. På grunn av antagelsen om et ideelt gir, så har vi fra Newtons tredje lov at $F_1 = F_2$. Dermed $\tau/r_2 = \tau/r_1$, som igjen betyr

$$\tau_2 = \tau_1/n.$$

Konklusjon: for en girutveksling $n \geq 1$ (skriver ofte $1:n$) så vil vinkelhastigheten (fra ω_1 til ω_2) øke, mens dreiemomentet (fra τ_1 til τ_2) vil synke.

[a] Gir er som regel basert på tannhjul. Man kan derfor tenke på r_1 og r_2 som antall tenner på tannhjulene.

Men skjer hvis systemet akselererer? 🤔 La oss nå anta at vi har et ideelt slik som i figur 3.5, hvor $\omega_2 = n\omega_1$. Det virker et dreiemoment τ_1 på det venstre hjulet, og et dreiemoment fra en last, τ_l, på det høyre hjulet. Momentbalanse for det første hjulet resulterer i

$$J_1\dot{\omega}_1 = \tau_1 - \hat{\tau}$$

hvor $\hat{\tau}$ er et dreiemoment som tilsvarer alt som skjer på høyresiden av giret pga. av (den holonomske) begrensningen $\omega_2 = n\omega_1$. Tilsvarende gir momentbalansen på høyresiden

$$J_2\dot{\omega}_2 = \tau_2 - \tau_l = \hat{\tau}/n - \tau_l,$$

hvor da $\tau_2 = \hat{\tau}/n$. Siden $\dot{\omega}_2 = n\dot{\omega}_1$, så har vi derfor at

$$J_2 n\dot{\omega}_1 = \hat{\tau}/n - \tau_l \quad \Longrightarrow \quad \hat{\tau} = n\left(J_2 n\dot{\omega}_1 + \tau_l\right).$$

Ved å sette dette inn i ligningen for venstresiden får vi

$$J_1\dot{\omega}_1 = \tau_1 - n\left(J_2 n\dot{\omega}_1 + \tau_l\right) \quad \Longrightarrow \quad (J_1 + n^2 J_2)\dot{\omega}_1 = \tau_1 - n\tau_l.$$

Fra dette kan vi også vise at

$$\left(\frac{J_1}{J_2} + n^2\right)\tau_2 = n\tau_1 + \frac{J_1}{J_2}\tau_l,$$

som viser at hvis treghetsmomentet fra lasten er dominerende, altså $J_2 \gg J_1$, så har vi $\tau_2 \approx \tau_1/n$. Tegne kraft-diagram.

Hvordan (og hvorfor) velge girutveksling? Det er ganske greit å tenke på en bil eller sykkel i denne sammenhengen:

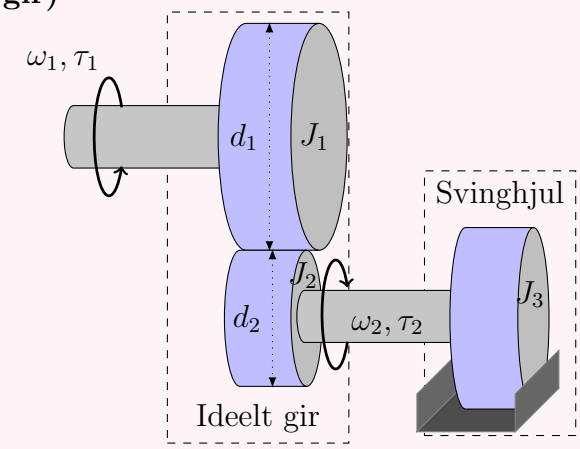
- Tilstrekkelig stor trekkraft og hastighet på utgangen (hjulene).
- Ikke for høyt turtall på motoren.

- Maksimal akselerasjon.
- Drivstofføkonomi.

I en bil kan man jo f.eks. skifte gir, og tilpasse det til situasjonen. I andre sammenhenger—ofte sånn for elektromotorer—bestemmer man girutvekslingen en gang for alle.

Oppgave 3.3. (Svinghjul med ideelt gir)

Et svinghjul med treghetsmoment J_3 skal styres ved hjelp av en motor (se figuren til høyre). Motoren generer et dreiemoment τ_1, og er festet til en aksling som roterer med vinkelhastighet ω_1. Akslingen er igjen festet til et ideelt gir med girutveksling $n = \frac{d_1}{d_2}$, hvor tannhjulene i giret har treghetsmoment henholdsvis lik J_1 og J_2. Det virker et viskøst friksjonsmoment på svinghjulet lik $\tau_f = d_v\omega_2$.



Du skal: a) Vise at de dynamiske ligningene (på tilstandsromform) til systemet med ω_1 som tilstand (både ω_2 og τ_2 **ikke** skal dukke opp i ligningene) er

$$\dot{\omega}_1 = \frac{1}{J}\left[\tau_1 - d_v \cdot n^2 \cdot \omega_1\right]$$

hvor $J = J_1 + (J_2 + J_3) \cdot n^2$.

b) Anta at systemet er i ro ($\omega_1 = \omega_2 = 0$). Det er ønsket å starte det opp med maksimal vinkelakselerasjon $\dot{\omega}_2$ på svinghjulet. Hva skal girutvekslingen være for for å oppnå dette hvis $J_1 = 4$, $J_2 = 1$, $J_3 = 15$ og $d_v = 2$? Merk: dette trenger bare å holde i øyeblikket når $\omega_1 = 0$. [a]

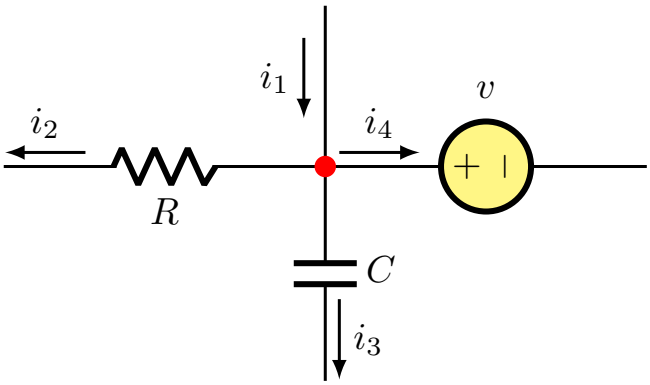
[a] Hint: Husk at en funksjon $f(x)$ har et ekstremum (maksimum eller minimum) ved et punkt a hvis $f'(a) = 0$; girutvekslingen vil være på formen $n = a/b$ for noen positive heltall a og b (altså 2/3, 6/2, for å gi noen eksempler).

3.2.4 Krichhoffs lover for elektriske kretser ("strømbalanse")

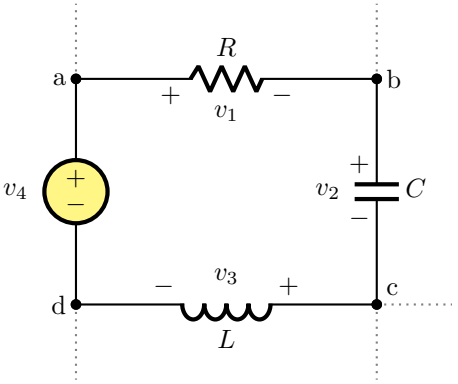
Du vil lære mer om dette i ELET1001 - Elektriske kretser - introduksjon

Alternative kilder: Wikipedia.

Kirchhoffs lover er to grunnleggende prinsipper innen elektrisk kretsanalyse, oppkalt etter den tyske fysikeren Gustav Kirchhoff.



a) Summen av strømmer inn og ute av den rød noden summerer til 0.



b) Summen av spenninger i den lukkede sløyfen a-b-c-d summerer til 0.

Figur 3.6: Illustrasjoner av Kirchhoffs lover for elektriske kretser.

De to «lovene», strømloven (eng. - Kirchhoff's Current Law (KCL)) og spenningsloven (eng. Kirchhoff's Voltage Law (KVL)), er illustrert i henholdsvis **a)** og **b)** i figur 3.6.

Kirchhoffs strømlov: I et forgreningspunkt i en elektrisk krets med n foregreninger er summen av alle inngående strømmer lik summen av alle utgående strømmer: $\sum_{k=1}^{n} i_k = 0$.

Kirchhoffs spenningslov: Summen av spenninger i en lukket strømsløyfe med n komponenter er lik null: $\sum_{k=1}^{n} v_k = 0$.

Disse prinsippsene er viktige verktøy innen elektrisk kretsanalyse og brukes for å beregne strømmer og spenninger i komplekse kretser. For våres del, som er ute etter å lage enkle matematiske modeller av dynamiske systemer vha. differensialligninger, er også følgende idealiserte relasjoner mellom strøm og spenning over basiskomponenter viktige:

La v betegne spenning i Volt [V] og i strøm i Ampere [A]. Vi har da følgende relasjoner:

- **Motstand (Ohms lov):** $v(t) = R \cdot i(t)$ hvor R er motanden i Ohm [Ω].
- **Kondensator:** $i(t) = C \cdot \frac{dv(t)}{dt}$ hvor C er kapasitet/kapasitans i Farad [F].
- **Spole:** $v(t) = L \cdot \frac{di(t)}{dt}$ hvor L er induktansen i Henry [H].

Eksempel 3.12. RLC-krets: Vi skal nå utlede en differensialligning tilsvarende RLC-kretsen vist i figur 3.6-b). Fra Kirchoffs spenningslov har at

$$v_4 = v_1 + v_2 + v_3 = R \cdot i(t) + C \int_0^t i(\tau)d\tau + L \cdot \frac{di(t)}{dt}$$

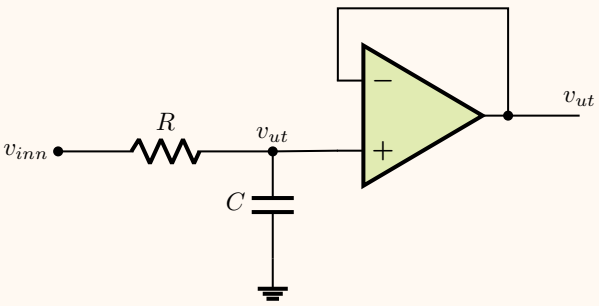
hvor $i(t)$ er strømmen gjennom kretsen (denne må være konstant grunnet strømloven).

Anta nå at spenningstilførslen v_4 holder seg konstant. Ved å da derivere med hensyn på tid på begge sider, og bruke at $\dot{v}_4 = 0$ (siden v_4 er konstant), så får vi følgende andre-ordens differensialligning for kretsen:

$$L \cdot \frac{d^2i(t)}{dt^2} + R\frac{di(t)}{dt} + C \cdot i(t) = 0$$

Eksempel 3.13. Første-orden lavpassfilter fra RC-krets: Et lavpassfilter er en svært vanlig «komponent» med mange bruksområder, ikke minst innen reguleringsteknikken. Denne typen filter, som vi skal se nærmere på i § 10.3, brukes til å «glatte ut» et signal ved å filtrere vekk høyfrekvente deler av signalet.

Vi skal i dette eksempelet utlede differensialligningen tilsvarende et første-ordens lavpass filter gitt av kretsen til høyre. Dette er en RC-krets koblet til en såkalt spenningsfølger (implementert vha. en ideell operasjonsforsteker (opamp)). Spenningsfølgeren har som jobb å sørge for at spenningen v_{ut} er den samme uansett hva som kobles til etter spenningsfølgeren.



Vi trenger derfor bare finne hva spenningen v_{ut} over kondensatoren er. La oss denne gangen bruke strømloven i noden mellom motstanden og kondensatoren:

$$i_C(t) - i_R(t) = C \cdot \frac{dv_{ut}}{dt} - \frac{(v_{inn} - v_{ut})}{R} = 0 \quad \Longrightarrow \quad \dot{v}_{ut} = -\frac{v_{ut}}{RC} + \frac{v_{inn}}{RC}.$$

Dette første-ordens lavpassfilteret tilsvarer derfor en første-ordens differensialligning på formen $\dot{y} = -ay + bu$ med $y = v_{ut}$, $u = v_{inn}$, og $a = b = 1/(RC)$.

Eksempel 3.14. Likestrømsmotor: TODO: slik som i øving 5

4. Simulering

Numerisk simulering omfatter metoder for å finne tilnærmede løsninger til differensialligninger.

Hvorfor skal du lære dette? 🤷‍♀️ Muligheten til å (numerisk) simulere et dynamisk system er nyttig på mange måter, og ikke minst for reguleringstekniske formål: Uten fare for både helse, utstyr og omgivelser, kan man bruke simulering til å blant annet teste ut både reguleringsstrategier og -parametere, undersøke sensitivitet og robusthet, samt få et estimat av ytelse og effekt.

I kapittel 2 så vi jo at vi kunne løse lineære differensialligninger av første og andre orden direkte; det vil si, vi kunne finne hvordan tilstanden $x(\cdot)$ (elle utgangen $y(\cdot)$) utiviklet seg med tiden t, i hvert fall for visse enkle inngangssignaler $u(t)$.

Så hvorfor trenger vi numeriske simuleringsmetoder i det hele tatt? 🤔
Det er dog slik at det generelt sett ikke er mulig å finne en slik *eksplisitt* løsning, altså at $x(t)$ er en funksjon bygd opp av kjente basisfunksjoner som sinus- og eksponentiallfunksjoner, etc. For eksempel er det svært få ulineære differensialligninger som har en eksplisitt løsning, og de fleste «ekte» systemer er unlineære! Det finnes allikevel (som oftest) en løsning på differensialligningen, og denne kan vi approksimere/tilnærme ved hjelp numersike integrasjonsmetoder.

Fun facts, bemerkninger og annet dill dall (you may skip)

Digitale tvillinger (se f.eks. [Sharma et al., 2022]) er en av de store «buzzword»-teknologiene i nyere tid. Den essensielle ideen bak digitale tvillinger er å sette opp en virtuell, simulert «dobbeltgjenger/tvilling» av en virkelig prosess. Enkle målinger av den fysiske prosessen overføres så til simuleringen, hvor disse dataene brukes til å danne grenseverdier og startbetingelsene til simuleringsmodellen. Tidligere ukjent tilstandsinformasjon genereres gjennom simulering og blir deretter levert tilbake til enhetens reguleringssystem. Det er denne toveis datautveksling mellom den virkelige prosessen og dens digital motpart (simuleringen) i sanntid som kjennetegner denne teknologien.

For at dette skal kunne tas i bruke under drift ("in real time"), så innebærer dette naturlig nok at man må ha simuleringsresultater på den virtuelle siden som er minst like raske som den naturlige prosessen ("real time simulation").

Fun facts, bemerkninger og annet dill dall (you may skip)

Sim2Real er et konsept som i disse dager er mye brukt innen maskinlæringsgrenen forsterket læring (eng. reinforcement learning). Ved hjelp av simulatorer kan man trene robotene i stor skala, på en rekke scenarier og forhold som sjelden oppstår i den virkelige verden. Man tar

så i bruk (sjeldent helt direkte) det som har blitt lært (hvilke handlinger som skal gjøres gitt sensordata, og muligens tidligere tilstander og målinger) i den virkelige verden. Eksempler på dette inkluderer: Autonome kjøretøy (bl.a. Tesla), robothender som løser rubiks kube, firbeinte roboter som tar seg en fjelltur (ANYmal), og roboter som spiller bordtennis, etc.

4.1. Numerisk integrasjon

Alternative kilder: Wikipedia.

Hva er vi ute etter? For å kunne simulere et dynamiske system, må vi på en måte «løse» differensialligningene som forklarer systemets dynamikk. Generelt sett (et unntak er lineære systemer, som alltid kan løses direkte) må dette gjøre ved hjelp av **numeriske metoder**. Mer spesifikt, så ønsker vi å finne en løsning (på formen av en (tidsvarierende) funksjon $x(t)$) til et *initialverdiproblem* (IVP) :

Initivalverdiproblem (IVP): Finn $x(t)$ for $0 \leq t \leq T$ gitt

$$\dot{x}(t) = f(x(t), t), \quad x(0) = x_0 \in \mathbb{R}. \tag{IVP}$$

En løsning til IVPen må dermed tilfredsstille:

1) startbetingelsene (initialverdien) $x(0) = x_0$;

2) ODEen (differensialligning) $\dot{x}(t) = f(x(t), t)$ for alle tidspunkt i tidsintervallet $[0, T]$.

Merk: Selv om vi her ikke direkte tar hensyn til noe kontrollsignal $u(t)$ (ei heller forstyrrelse $v(t)$), kan man, hvis nødvendig, anta at dette er en del av funksjonen $f(\cdot)$. Det går selvsagt også an å se på systemer med flere tilstander, men vi holder oss til én tilstand for enkelhets skyld.

Hvorfor trenger vi numerisk integrasjon? Ved å integrere begge sider av (IVP) med hensyn på tid, så er det klart at en hver løsning på initialverdiproblemet må tilfredstille

$$x(t) = x_0 + \int_0^t f(x(\tau), \tau) d\tau.$$

Men, som nevnt tidligere, så har vi følgende problem: vi kan bare i spesielle tilfeller finne en **eksplisitt/analytisk løsning** på integralet $\int_0^t f(x(\tau).\tau) d\tau$, altså en løsning som kan uttrykkes eksplisitt ved hjelp av en bregrenset mengde av elementære basisfunksjoner (altså summer og produker av konstanter, polynomer, trigonometriske-, logaritmiske og eksponentielle funksjoner, etc). La oss ta noen eksempler:

Eksempler:

- $\dot{x}(t) = \cos(t)$, $x(0) = 0$, har løsningen $\int_0^t \cos(\tau) d\tau = \sin(t)$;

- $\dot{x}(t) = \cos(t^2)$, $x(0) = 0$, har ikke en analytisk løsning (fordi $\int_0^t \cos(\tau^2)dt$ ikke har det);
- $\dot{x}(t) = -2 \cdot x(t)$, $x(0) = 3$, har løsningen $x(t) = 3e^{-2t}$;
- $\ddot{x}(t) = -2 \cdot \sin(x(t))$, $x(0) = 3$ og $\dot{x}(0) = 0$, har ikke en analytisk løsning.

NB! Siste eksempelet har samme form som dynamikken til en enkel pendel (se eks. 3.11), så selv om det ikke finnes en *analytisk* løsning, så finnes det jo da en løsning.

For at du skal få den inn med sleiv, har vi derfor følgende «definisjon»:

Numerisk integrasjon: Finne en best mulig *approksimasjon* av en løsning til (IVP) ved hjelp av numeriske metoder. Det er dog viktig at vi bruker metoder som tar høyde for at funksjonen $f(\cdot)$ også kan avhenge av løsningen $x(t)$ (og vi vet jo ikke hva den er enda).

Vi skal se på noen slike metoder i den neste seksjonen, spesifikt noen såkalte *Runge–Kutta-metoder*, deriblant Eulers metoder og trapesmetoden.[1] Vi skal også se på hvordan Runge–Kutta-metoder kan videreutvikles slik at man kan bruke varierende steglengder for å øke nøyaktigheten. I slutten av kapittelet skal vi også så se på hvordan man kan bruke MATLAB og Simulink til å simulere dynamiske systemer ved hjelp av slike metoder.

4.1.1 Numerisk integrasjon uten tilstander

La oss først se på hvordan man numerisk kan integrere en funksjon som bare avhenger av tiden t og ikke tilstanden x, altså:

Nåværende problem: Finn en tilnærmet verdi (en approksimasjon) av integralet

$$\int_a^b f(t)dt.$$

La oss dele tidsintervallet $[a, b]$ i N deler av lengde $\mathfrak{h}$,[2] altså

$$a = t_0 < t_1 < t_2 < \cdots < t_{N-1} < t_N = b$$

hvor, for alle $k = 0, 1, 2, \ldots, N-1$ (evt. kunne vi her ha skrevet $\forall k = 0, 1, \ldots,$),

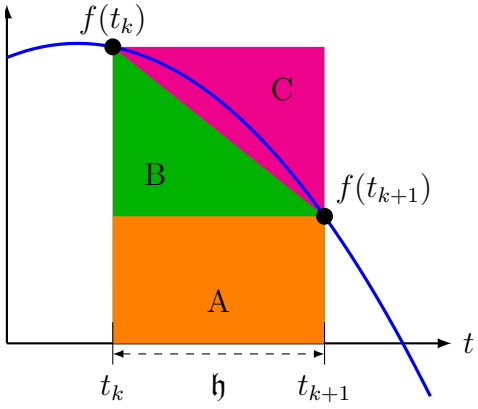
$$\mathfrak{h} = t_{k+1} - t_k.$$

Siden $t_1 = t_0 + \mathfrak{h}$ og $t_2 = t_1 + \mathfrak{h}$, etc., så kan vi dele integralet opp som følger:

$$\int_a^b f(t)dt = \int_{t_0}^{t_0+\mathfrak{h}} f(t)dt + \int_{t_1}^{t_1+\mathfrak{h}} f(t)dt + \cdots + \int_{t_{N-1}}^{t_{N-1}+\mathfrak{h}} f(t)dt = \sum_{k=0}^{N-1} \int_{t_k}^{t_k+\mathfrak{h}} f(t)dt.$$

Hvis $\mathfrak{h}$ er liten nok, kan vi approksimere hvert av disse integralene ved én av følgende metoder, som også er grafisk illustrert i figur 4.1:

[1] Det finnes også en annen familie av metoder, såkalte multistegs-metoder, som vi ikke skal se på.
[2] Bruker gotisk "h" for å skille fra alle andre h-er i disse notatene. Alternativt kan man bruke Δt.



Figur 4.1: Areal som viser numeriske approksimasjoner av integralet $\int_{t_k}^{t_{k+1}} f(\tau)d\tau$: Eulers framover metode=A+B+C=A+2B; Eulers bakover metode=A; Trapesmetoden = A+B.

- **Eulers fremovermetode:** $\int_{t_k}^{t_k+\mathfrak{h}} f(t)dt \approx \mathfrak{h} \cdot f(t_k) =$**A+B+C**;
- **Eulers bakovermetode:** $\int_{t_k}^{t_k+\mathfrak{h}} f(t)dt \approx \mathfrak{h} \cdot f(t_k+\mathfrak{h}) = \mathfrak{h} \cdot f(t_{k+1}) =$**A**;
- **Trapesmetoden:** $\int_{t_k}^{t_k+\mathfrak{h}} f(t)dt \approx \frac{1}{2}\mathfrak{h} \cdot (f(t_k) + f(t_k+\mathfrak{h})) =$**A+B**.

Legg merk til at Trapesmetoden er gjennomsnittet av de to første metodene.

4.1.2 Numerisk integrasjon med tilstander

Nå skal vi også ta hensyn til at dynamikken avhenger av tilstandene:

Nåværende problem: For et IVP,

$$\dot{x}(t) = f(x(t), t), \quad x(0) = x_0 \in \mathbb{R}, \tag{IVP}$$

finn en tilnærming av løsningen $x(t)$ for $0 \leq t \leq T$, gitt ved

$$x(t) = x_0 + \int_0^t f(x(\tau), \tau)d\tau.$$

Antagelser og notasjon: Vi vil anta at $f(\cdot)$ er en glatt funksjon, altså at den kan deriveres uendelig mange ganger. Vil vil også anta en konstant **steglengde** $\mathfrak{h} > 0$.

Vi vil bruke notasjonene $t_k = k \cdot \mathfrak{h}$, $x(0) = x_0$, og $x_k \approx x(t_k)$, $k = 0, 1, \ldots$, det vil si

$$x_{k+1} \approx x(t_k + \mathfrak{h}) = x(t_k) + \int_{t_k}^{t_k+\mathfrak{h}} f(x(\tau), \tau)d\tau. \tag{4.1}$$

I den følgende seksjonen skal vi se på såkalte Runge–Kutta-metoder for å approksimere en løsning til initialverdiproblemet (IVP). Vi begynner med de enkleste metodene, nemlig Euler metodene og trapesmetoden, som vi allerede har sett på for funksjoner som ikke avhenger av tilstandene.

Eulers (eksplisitte) forovermetode

Den enkelste metoden er Eulers (eksplisitte[3]) fremover metode:

Eulers fremovermetode:

$$x_{k+1} = x_k + \mathfrak{h} \cdot f(x_k, t_k). \tag{4.2}$$

Idé: hvis $\mathfrak{h} > 0$ er liten nok, så er $\mathfrak{h} f(x_k, t_k) \approx \int_{t_k}^{t_k+\mathfrak{h}} f(x(\tau), \tau) d\tau$ en grei tilnærming. Vi finner altså x_{k+1} ved å gå én lengde $\mathfrak{h} \cdot \|f(x_k, t_k)\|$ fra x_k i retningen til vektoren $f(x_k, t_k)$.

Fordeler:

- 👍 Rask, enkel og lett å implementere.

Ulemper:

- 💩 Unøyaktig (selv for små $\mathfrak{h}$);
- 💩 Numerisk ustabil hvis $\mathfrak{h}$ ikke er liten nok.

Numerisk ustabilt – hva betyr det? 🤔 Det betyr at differansen mellom estimatet funnet fra Eulers metode og den ekte løsningen til differensialligningen øker, noe som kan resultere i at estimatets magnitude går mot uendelig selv om den ekte løsninger holder.

Følgende eksempel fra Wikipedia illustrerer dette fenomenet:

Eksempel 4.1. Gitt følgende IVP:

$$\dot{x} = -3x, \qquad x(0) = 1.$$

Vi vet at den ekte løsningen er $x(t) = e^{-3t}$, slik at $x(t) \to 0$ når $t \to \infty$.

La oss derfor se hvafår med Eulers fremovermetode med tidsteget $\mathfrak{h} = 1$. VI får da

$$x_{k+1} = (1 - 3)x_k = -2x_k.$$

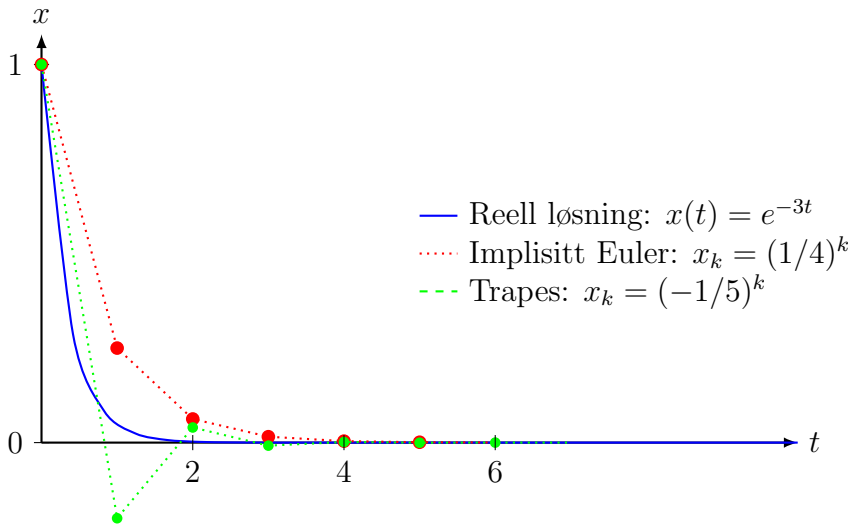
Dette viser tydelig at $|x_k| \to \infty$ når $k \to \infty$ (x_{k+1} har dobbelt så stor magnitude i forhold til x_k). Tar vi derimot $\mathfrak{h} < 2/3$ vil også x_k konvergere til null.

(Videolenke; mulig eksamensoppgave 2023? Husk å FJERNE!)

Implisitte metoder: Eulers bakovermetode og trapesmetoden

I eksempel 4.1 så vi at Eulers fremovermetode, som er en *eksplisitt metode*, med for langt tidsteg førte numerisk ustabilitet, selv for en enkelt lineært system. Implisitte metoder, på den annen side, har som regel bedre numerisk stabilitet enn de eksplisitte metodene, men er til gjengjeld noe mer krevende å regne ut. De to meste elementære implisitte metodene er Eulers bakovermetode og trapesmetoden. Som vist i figur 4.2 er disse ikke ustabile når de brukes på systemet i eksempel 4.1.

[3] Metoden kalles også for den eksplisitte metoden siden høyresiden av (4.2) kun avhenger av x_k, ikke x_{k+1}.



Figur 4.2: Sammenligning av Euler (implisitte) bakovermetode og trapesmetoden for systemet $\dot{x} = -3x$, med $x_0 = \mathfrak{h} = 1$. Merk: Eulers bakover metode har en "overdempet" respons, mens trapesmetoden har en slags "underdempet" respons for dette eksempelet. Eulers forovermetode er ikke vist siden denne er ustabil for dette eksempelet.

Eulers bakovermetode: En annen metode, om enn hakket mer kompleks, med bedre numerisk stabilitet er Eulers (implisitte[4]) bakover metode:

Eulers bakovermetode:

$$x_{k+1} = x_k + \mathfrak{h} \cdot f(x_{k+1}, t_{k+1}). \tag{4.3}$$

Idé: I motsetning til fremovermetoden bruker man tangenvektoren $f(x_{k+1}, t_{k+1})$ i stedet for $f(x_k, t_k)$, noe som gjør det lett å regne ut x_k gitt x_{k+1}, altså *bakover* i tid.

Fordeler:

- 👍 Relativt lett å forstå;
- 👍 Numerisk stabil for alle $\mathfrak{h}$ (såkalt A-stabil)!

Ulemper:

- 💩 Unøyaktig (selv for små $\mathfrak{h}$);
- 💩 Må løse den implisitte ligningen (4.3) (x_{k+1} på begge sider).

⚠️**Achtung!** Eulermetodene introdusere noe man kaller for *numerisk dempning* (denne kan være både positiv eller negativ). Dette kan føre til at et (marginalt) stabilt system ser ustabilt ut når vi simulerer det, eller vice versa. Dere vil se nærmere på dette i en av øvingene.

[4]Metoden kalles også for den implisitte metoden siden høyresiden av (4.3) også avhenger av x_{k+1}.

Eksempel 4.2. (Eulers bakover-metode med MATLABs fsolve)

Mål: løse $\dot{x} = -\sin(x)$, $x(0) = 0$, vha. av Eulers bakovermetode (se (4.3)) i MATLAB med tidssteg $\mathfrak{h} = 0.1$. Ligningen fra bakover-Euler er $x_{k+1} = x_k + h \cdot (-\sin(x_{k+1}))$, som igjen tilsvarer $F(x_{k+1}, x_k) = x_{k+1} - x_k + h \cdot \sin(x_{k+1}) = 0$.

En enkel måte å løse dette på i MATLAB er vist i kodesnutt 4.1. Merk at `g=@(t,x) -sin(x)*t` betyr at `g` er en funksjon av variablene `t` og `x` lik $-\sin(x)$, mens `h=@(x)g(2,x)` gir en ny funksjon `h` som bare har variabelen `x` og tilsvarer da `g` ved når $t = 2$, altså er `h` lik funksjonen `@(x)-2*sin(x)`.

Kodesnutt 4.1: Løse IVP med én tilstand i MATLAB vha. Eulers implisitte metode og fsolve.

```
h     = 0.1;                    % Tidssteg
x0    = 1;                      % Initialverdi
tEnd  = 10;                     % Simuleringstid
N     = tEnd/h;                 % Antall steg
t = linspace(0,tEnd,N+1);% Liste med tidspunkt
x=zeros(N+1,1);                 % Liste hvor vi skal lagre tilstandene
x(1)=x0;
% Systemets dynamikk:
f = @(t,x) -sin(x);
% Funksjon tilsvarende ligning fra Implisitt Euler (z=x_{k+1}):
F = @(t,z,xk)(z-xk-h*f(t,z));
opt = optimset('Display','off','TolFun',1e-8); % Til fsolve
    senere
% For-lokke
for k=1:N
    x(k+1)= fsolve(@(z)F(t(k),z,x(k)),x(k),opt);
end
[tt,xt] =ode45(f,[0,tEnd],x0); % ODE45 til sammenligning
% Lage figur
figure(1);clf(1)
hold on
plot(t,x)
plot(tt,xt,'r:')
xlabel('Tid')
legend({'Implisitt (bakover) Euler', 'ODE45'},'Location','best')
```

Trapesmetoden. En mellomting mellom Eulers fremover- og bakover-metode:

Trapesmetoden:

$$x_{k+1} = x_k + \frac{\mathfrak{h}}{2} \cdot (f(x_k, t_k) + f(x_{k+1}, t_{k+1}))\,. \tag{4.4}$$

Idé: Verdien til x_{k+1} er gjennomsnittet av det man fikk fra Eulers fremover- og bakover metoder. Dette betyr at dette også er en implisitt metode.

Fordeler:

- 👍 OK-ish lett å forstå;
- 👍 Numerisk stabil for alle $\hbar$ (A-stabil).

Ulemper:

- 💩 Ikke veldig nøyaktig (selv for små $\hbar$), men bedre enn Euler-metodene;
- 💩 Må løse den implisitte ligningen (4.4) (x_{k+1} på begge sider).

4.1.3 Aspekter ved numerisk integrasjon og ting som bør vurderes

Huskeregler for valg av ODE-metode og -løsere:

- Høyere-orden: høyere nøyaktighet, men også høyere kompleksitet og flere nødvendige steg.
- Eksplisitt: enkle å implementere men dårligere stabilitet.
- Implisitt: mer kompliserte å implementere (trenger f.eks. noe a la Newtons metode) men A-stabile (og derfor egnet til stive ODEer).
- Adaptiv/variabel: tillater en ønsket balansegang mellom hurtighet og ønsket nøyaktighet.

Høyere-ordens Runge–Kutta-metoder

Både metodene vi har sett på så langt (Eulers fremover- og bakover-metoder, samt trapesmetden) og kjente ODE-løsere som MATLABs `ode45` er (bygd opp av) såkalte *Runge–Kutta-metoder*. Disse metodene kan deles opp i deres orden (høyere orden gir høyere nøyaktighet), og om de er eksplisitte eller implisitte (som `ode45`, som faktisk består av to slike metoder, kan de også være adaptive / ha variable steglenger, mer om dette senere). For eksempel:

- Eulers fremovermetode er en første-ordens eksplisitt metode;
- Eulers bakovermetode er en første-ordens implisitt metode;
- Trapesmetoden er en andre-ordens implisitt metode;
- Dorman–Prince-metoden (`ode45`) består av både en fjerde- og en femte-ordens eksplisitt metode som kjøres parallelt.

Numerisk integrasjon med variable/adaptive steglengder

Nøyaktigheten til en integrasjonsmetode kan alltid økes ved å korte ned steglengden $\hbar$. En kortere steglengde fører dog igjen til flere diskretiseringpunkter, og dermed til en lengre integrasjonstid.

Et alternativ som lar en finne et ønsket kompromiss mellom nøyaktighet og integrasjonstid er å ta i bruk adaptive løsere. Her er det to vanlige strategier:

1. Ta samme løser og sammenlign resultatet for to forskjellige steglengder; f.eks. $\hbar$ vs $\frac{\hbar}{2}$;
2. Ta to forskjellige løsere å sammenlign deres resultatet. Slik kan man approksimere feilen.

Eksempel: Bak MATLABs velkjente ODE-løser `ode45` er Dormand–Prince metoden [Dormand and Prince, 1980]. Denne metoden bruker totalt seks funksjonsevalueringer for å beregne fjerde- og femteordens approksimasjoner. Forskjellen mellom disse brukes som et estimat på feilen til disse approksimasjonene. Dette feilestimatet er veldig praktisk for integrasjonsalgoritmer med adaptive trinnstørrelser.

Stive vs ikke-stive differensialligninger

Intuitivt sett, så er stive differensialligninger bare differensialligninger som det er vanskelig å integrere numerisk. Eksempler på dette er systemer som innehar både veldig treg og veldig rask dynamikk.

Tips: Bruk implisitte ODE-løsere ved stiv differensialligninger.

Kaotiske systemer, kaosteori og kontrollering av kaos

Kaotiske dynamiske systemer kjennetegnes av at evolusjonen til systemets tilstander er meget sensitive i forhold til initialbetingelsene. Små feil vil derfor raskt blåses opp (eksponentiell feilpropergering), noe som gjør at man må ta resultatene man får etter en viss tidsperiode med en klype salt. Læren om slike systemer kalles kaosteori.

Kjente eksempler på kaotiske systemer: værmodeller, en dobbel pendel, og solsystemet vårt.

Merk: Når vi designer en regulator for et kaotisk system (for å f.eks. stabilisere et gitt punkt), så er vi faktisk ute etter å fjerne/eliminere all kaos fra systemet og i stedet gjøre dets oppførsel prediktivt og forutsigbart; med andre ord, for å få det til å høres mest mulig kult ut, så vil vi «kontrollere kaos».

Trunkeringsfeil og numeriske feil

Flyttall og avrundingsfeil. Trunkeringsfeil i fermover Euler.

4.2. Simulering vha. MATLAB og Simulink

MATLABs numeriske ODE-løsere (ODE45 og ODE23)

En liste av alle ODE-løserene i MATLAB er vist i figur 4.3, hvor man også kan se hvilke typer systemer de er egnet til, samt deres nøyaktighet.

Kodesnutt 4.2: Simulering av pendel vha. ode45 i MATLAB.

```
%% Simulere en pendel vha. ode45
%% Init
x0   = [pi/2;0]; % Initialverdier
t0   = 0;        % Starttid for simulering [s]
tend = 20;       % Sluttid for simulering [s]
```

Solver	Problem Type	Accuracy	When to Use
ode45	Nonstiff	Medium	Most of the time. ode45 should be the first solver you try.
ode23		Low	ode23 can be more efficient than ode45 at problems with crude tolerances, or in the presence of moderate stiffness.
ode113		Low to High	ode113 can be more efficient than ode45 at problems with stringent error tolerances, or when the ODE function is expensive to evaluate.
ode78		High	ode78 can be more efficient than ode45 at problems with smooth solutions that have high accuracy requirements.
ode89		High	ode89 can be more efficient than ode78 on very smooth problems, when integrating over long time intervals, or when tolerances are especially tight.
ode15s	Stiff	Low to Medium	Try ode15s when ode45 fails or is inefficient and you suspect that the problem is stiff. Also use ode15s when solving differential algebraic equations (DAEs).
ode23s		Low	ode23s can be more efficient than ode15s at problems with crude error tolerances. It can solve some stiff problems for which ode15s is not effective. ode23s computes the Jacobian in each step, so it is beneficial to provide the Jacobian via odeset to maximize efficiency and accuracy. If there is a mass matrix, it must be constant.
ode23t		Low	Use ode23t if the problem is only moderately stiff and you need a solution without numerical damping. ode23t can solve differential algebraic equations (DAEs).
ode23tb		Low	Like ode23s, the ode23tb solver might be more efficient than ode15s at problems with crude error tolerances.
ode15i	Fully implicit	Low	Use ode15i for fully implicit problems $f(t,y,y') = 0$ and for differential algebraic equations (DAEs) of index 1.

Figur 4.3: Liste over ODE-løserene i MATLAB. Hentet fra lenke.

```
g     = 9.81;       % Gravitasjonsakselerasjonen [m/s^2]
l     = 2;          % Pendelens legnde [m]
params.a =g/l;      % Modellparameter
%% Simulere:
opts = odeset('RelTol',1e-3,'AbsTol',1e-5);
[t,x] = ode45(@(t,x)pendDyn(t,x,params),[t0,tend],x0,opts);
%% Plotte:
figure(1);clf(1);
hold on;
subplot(2,1,1);
plot(t,x(:,1),'r:','LineWidth',2);
ylabel('$x$ [rad]','Interpreter','latex','FontSize',12);
hold on;
subplot(2,1,2);
plot(t,x(:,2));
plot(t,x(:,1),'--o','LineWidth',1);
xlabel('$t$ [s]','Interpreter','latex','FontSize',12);
ylabel('$\dot{x}$ [rad/s]','Interpreter','latex','FontSize',12);
%% Dynamikken: (slutten av scriptet/egen funksjon)
```

```
function dxdt = pendDyn(t,x,params)
a = params.a;
dxdt=[x(2);-a*sin(x(1))];
end
```

Hva med tidsforsinkelser? Løseren dde23 kan brukes ved konstante tidsforsinkelser.

Man kan også håndtere tidsforsinkelser ved hjelp av transport delay-blokken i Simulink.

Innstillinger: Relativ- og absolutt toleranse

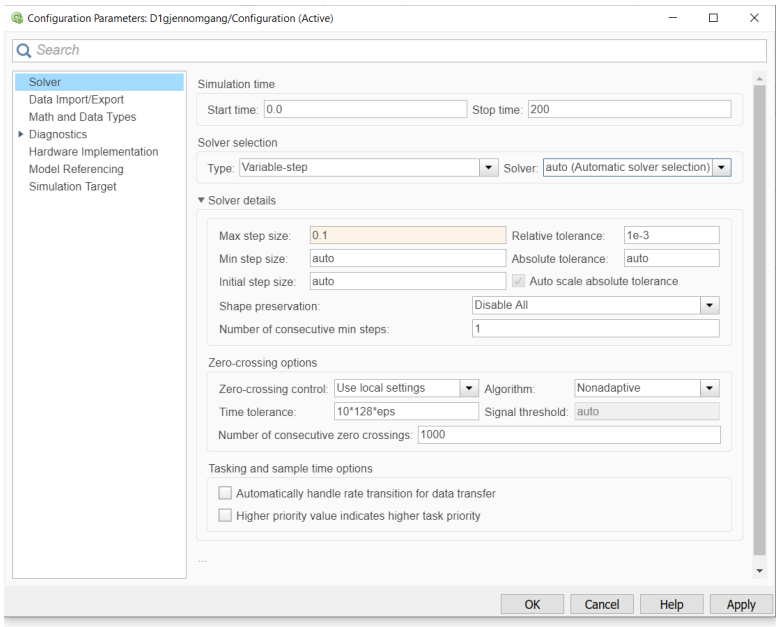
MATLAB: De forskjellige ODE-løserene har mange innstillinger man kan endre vha. odeset-kommandoen, se denne lenken for full oversikt. Følgende tre er det viktig å ha kontroll på:

MaxStep: Den største steglengde ODE-løseren får lov å ta. Med andre ord: selv om løseren mener den kan ta et lengre steg uten at det går utover ønsket nøyaktighet (gitt av størrelsen under), får den ikke lov å gå over verdien til MaxStep.

Relativ toleranse (RelTol): Verdien RelTol spesifiserer den tillatte feiltoleransen i forhold til tilstandsvektoren ved hvert simuleringstrinn. Hvis du setter RelTol til $1e-2$ (=0.01), spesifiserer du at en feil på 1 % i forhold til hver tilstandsverdi er akseptabel ved hvert simuleringstrinn. Intuitivt sett så kontrollerer den antall signifikante sifre i en løsning, bortsett fra når det er mindre enn den såkalte *absolutte toleransen*.

Absolutt toleranse (AbsTol): AbsTol brukes til å bestemme den største tillatte absolutte feilen på ethvert trinn i en simulering. Denne toleransen "tar over for" den relative toleransen når en løsning er liten. Intuitivt sett, når løsningen nærmer seg 0, er AbsTol terskelen hvor du ikke lenger bekymrer deg for nøyaktigheten til løsningen (altså den relative toleransen) siden tilstandsverdiene uansett er veldig små (kanskje tilnærmet lik 0).

Simulink: I Simulink kan du endre på både løser og dens innstillinger via følgende sti: Modelling→Model Settings (tannhjul-icon). Alle innstillingsmuligheter er vist i figur 4.4.



Figur 4.4: Innstillingsmuligheter i Simulink.

Del III

Reguleringsteknikk

5. PID-regulatoren og PID-tuning

Alternative kilder: Sek. 1.4.4 og kap. 11 i [Haugen, 2023] Kap. 2 i [Bjørvik and Hveem, 2014]; WIKIPEDIA; Brian Douglas videoer: https://youtu.be/wkfEZmsQqiA.

Du vil lære mer om dette i IELET2002 - Reguleringsteknikk

I seksjon 1.3 introduserte vi så smått hva reguleringsteknikk er og hva det kan brukes til. Vi har også introdusert, ved hjelp av «Tom i Tallinjeveien»-eksemplene, tre grunnleggende konsepter innen reguleringsteknikk: P-, PI- og PD-regulatorene (se hhv. § 1.4.3, 2.5.2 og 2.5.4). Disse regulatorene er underkategorier av den trolig viktigste reguleringsstrategien som finnes, nemlig PID-regulatoren.

I dette kapittelet skal vi se nærmere på PID-regulatoren og dens «deler», nemlig P-, I- og D-leddene. Vi skal også se på måter for å justere parameterne til slike regulatorer, såkalt «*tuning*», samt nevne noen viktige betraktninger relatert til utfordringer innen regulatordesign og reguleringsteknikk for øvrig.

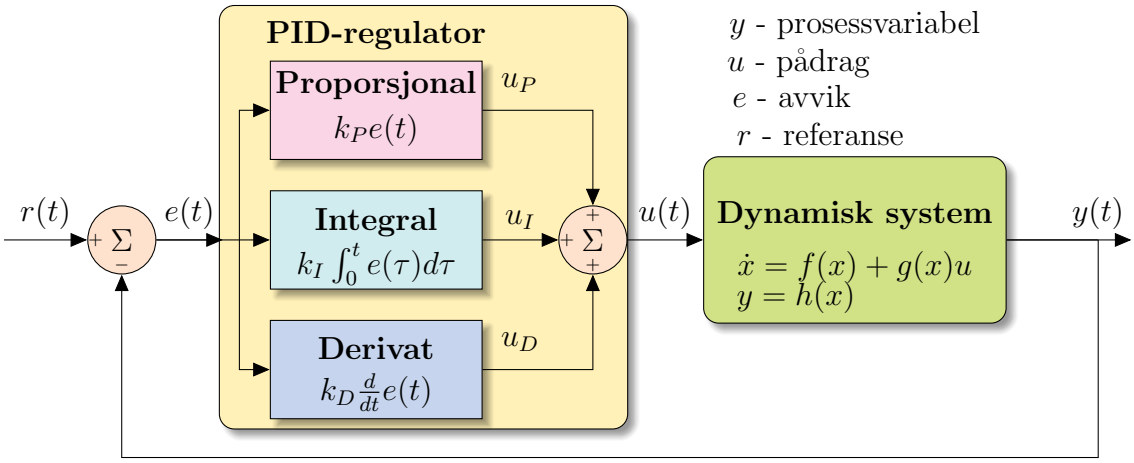
5.1. PID-regulatoren i et nøtteskal

Hvorfor skal du lære dette? 🤷‍♀️ Proporsjonal-integral-derivat-(PID)-regulatoren er enkelt og greit «kongen» av reguleringsteknikken; den er, og vil trolig lenge forbli, den mest brukte reguleringsstrategien brukt i praksis med god margin. Faktisk er trolig over 95% av regulatorene brukt i industrien PID-regulatorer, hvor de fleste er av typen PI [Seborg et al., 2016, Desborough and Miller, 2002]. I følge spørreundersøkelsen i [Samad, 2017] er PID-regulatoren også den teknologien innen reguleringsteknikken med høyest (opplevd) effekt. Så populær er den, at det til og med skrives artikler i (tekniske) ukeblader om den!

Den mest grunnleggende formen til en Proposjonal-Integral-Derivat(PID)-regulator er

$$u_{PID}(t) = \underbrace{k_P e(t)}_{\text{proporsjonal}} + \underbrace{k_I \int_0^t e(\tau)d\tau}_{\text{integral}} + \underbrace{k_D \frac{de(t)}{dt}}_{\text{derivat}}. \tag{PID}$$

Et blokkdiagram av en slik regulator er vist i figur 5.1. Der har vi et mulig unlineært dynamisk system med én inngang, u, og én utgang, y, hvor målet er å få utgangen til å følge den ønskede



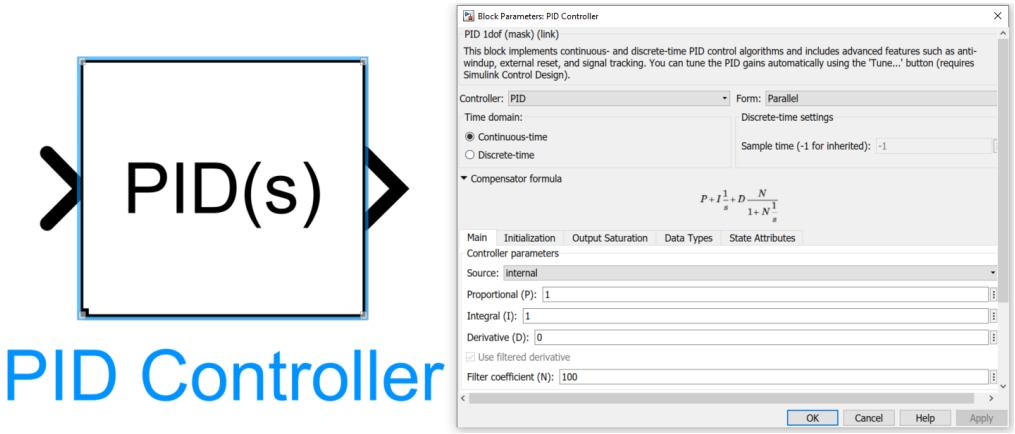
Figur 5.1: Blokkdiagram av en PID-regulator på parallellform.

referansen/settpunktet $r(t)$. Vi bruker som tidligere e (for «error») for å betegne avviket, altså differansen mellom prosessvariabelen og referansen/settpunktet: $e(t) = r(t) - y(t)$.

PID-regulator i Laplace-domenet: Husk nå fra seksjon 2.8 om overføringsfunksjoner at man kan tenke på den såkalte *Laplace-variabelen* s som en derivat-operator, mens $\frac{1}{s}$ kan tenkes på som en integrator (slik som i Simulink). PID-regulatoren representert i *Laplace-domenet*, altså den *overføringsfunksjons-form*, er derfor som følger:

$$U_{PID}(s) = k_P E(s) + k_I \frac{1}{s} E(s) + k_D s E(s) = \left(k_P + \frac{k_I}{s} + s k_D \right) E(s).$$

Grunnen til at jeg også tar med denne formen, er at den er svært mye brukt; f.eks. er det denne formen som brukes i PID-blokken i Simulink (se figur 5.2), selv om den også bruker et såkalt *derivat-filter* i forbindelse med D-leddet (§ 5.1.3).



Figur 5.2: PID-regulatoren i Simulink og dens innstillinger.

5.1.1 P-leddet

Proporsjonal(P)-leddet i en PID-regulator har følgende form:

Tidsdomenet: $u_P(t) = k_P e(t)$. Laplacedomenet: $U_P(s) = k_P E(s)$

Som navnet tilsier, så gir proporsjonal-leddet et bidrag som er proporsjonalt (gjennom forsterknings-parameteren k_P) med avviket, altså ikke ulikt Hookes lov for lineære fjærer hvor kraften er proporsjonal med utslaget fra utgangsposisjonen. P-leddet bør tenkes på som arbeidshesten i de fleste PID-regulatorer, i den forstand at den som regel bør gjøre mesteparten av jobben. Økt proporsjonalforsterkning fører (som oftest) til følgende:

Fordeler:

- 👍 raskere respons og høyere båndbredde (et ord dere vil høre mer om i senere fag);
- 👍 mindre stasjonært avvik (se § 2.5.2).

Ulemper:

- 💩 mindre stabilitetsmargin[1] (derav mer sensitiv til f.eks. tidsforsinkelser; § se 2.6);
- 💩 høyere pådrag, og dermed fare for at pådragsorganet i går *metning* (se § 7.2.1).

5.1.2 I-leddet

Integral(I)-leddet i en PID-regulator har følgende form:

Tidsdomenet: $u_I(t) = k_I \int_0^t e(\tau)d\tau$ eller $\dot{u}_I(t) = k_I e(t)$. Laplacedomenet: $U_I(s) = k_I \frac{E(s)}{s}$

Et integral-ledd er som en hammer: det løser mange problemer, men skaper til gjengjeld mange problemer hvis det brukes uforsiktig, og kan være noe mindre egnet til oppgaver som krever «fin-presisjon» og høy hastighet. Husk fra § 2.5.2 at et I-ledd fører til en *dynamisk* regulator, og ekstra dynamikk er jo lik mer treghet! Økt integralledd fører (som oftest) til:

Fordeler:

- 👍 at stasjonært avvik fjernes raskere (se § 2.5.2);
- 👍 at man kan bruke forenklede regulerings-stukturer (ikke pensum, men verdt å nevne).

Ulemper:

- 💩 tregere respons på grunn av ekstra dynamikk, noe som kan føre til oversving;
- 💩 eventuell forverring av fenomenet windup/oppspoling (mer om dette i § 7.2);
- 💩 potensiell rykk-og-napp oppførsel (og mulige grensesvingninger) pga. statisk friksjon/stiksjon

5.1.3 D-leddet

Derivat(D)-leddet i en PID-regulator har følgende form:

[1] *Stabilitetsmargin* betyr enkelt og greit hvor langt unna man er at systemet blir ustabilt. Desto større marginer, desto bedre i den forstand. Merk dog at større marginer kan bety lavere ytelse, i form av lav responstid!

Tidsdomenet: $u_D(t) = k_D\dot{e}(t)$. Laplacedomenet: $U_D(s) = k_D sE(s)$

Man kan høre litt forskjellig når det kommer til betydningen av D-leddet; noen ganger er det hensiktsmessig å tenke på det som å legge til fiktiv demping i systemet, mens det andre ganger kan bli regnet som et "prediksjons"-ledd som forutser endringer i avviket/prosessvariabelen og korrigerer pådraget deretter. En økning i D-leddet har som regel følgende effekt (avhenger litt av systemets og mulige tidsforsinkelser):

Fordeler:

- 👍 raskere respons (rettere sagt, så tillater det det);
- 👍 økt stabilitet;
- 👍 mindre oversving (bidrar med en enkel prediksjon).

Ulemper:

- 💩 mater forsterket målestøy inn i systemet;
- 💩 mer oversving ved store tidsforsinkelser;
- 💩 sprang i pådrag ved endring i referansen (eng. "derivative kick").[2]

På grunn av den mulige forsterkningen av (hurtig-endrende) støy, implementeres D-leddet nesten alltid sammen med et derivatfilter:

5.1.4 Derivat-filter

Hvorfor skal du lære dette? 🤷‍♀️ Et ideelt derivat-ledd har formen $u_D(t) = k_D\dot{e}(t) = k_D(\dot{r}(t) - \dot{y}(t))$. Problemet er at målingen $y(t)$ i praksis alltid inneholder (høyfrekvent/hurtigendrende) målestøy $w(t)$ (se tabell 1.1). Hvis vi prøver å derivere $y(t)$ for å finne $\dot{y}$ (for derivatleddet) så deriverer vi også denne målestøyen. Siden støy er hurtigendrende, så vil dette forsterke den, slik at D-leddet kan ende opp med å mate forsterket målestøy inn i prosessen! 😭 F.eks., hvis $w(t) = \sin(100t)$, så er jo $\dot{w}(t) = 100\cos(100t)$, altså forsterket 100 ganger! 😱

I realiteten vil et slikt D-ledd også forsterke referanse-endringer, og til og med kvantiseringsfeil. Man må derfor alltid kombinere et slikt ledd med et *lavpassfilter*, som vi kaller et *derivatfilter*, når man skal implementere et D-ledd i praksis.

Et D-ledd med derivat-filter, $u_{DF}(t)$, har som regel følgende form:

$$\dot{z}(t) = \frac{1}{T_f}\left(-z(t) + e(t)\right) \tag{5.1a}$$

$$u_{DF}(t) = \frac{K_D}{T_f}\left(e(t) - z(t)\right). \tag{5.1b}$$

hvor $T_f > 0$ er filterkonstanten ($T_f \ll T_D = k_D/K_P$), mens $z(t)$ er en intern tilstand i regulatoren som man initialiserer som $z(0) = 0$.

[2] For å unngå store sprang i pådraget fra et D-ledd grunnet endringer (sprang) i referansen, er det også vanlig at bare prosessvariabelen blir derivert, altså $u_D(t) = -k_D\dot{y}$.

Ser dette drøyt og uforståelig ut? Ta det med ro, det er bare «chillen»! 😎 Det er ikke så viktig at du skjønner virkemåten via diff.ligningen akkurat nå; det viktige er at du forstår behovet for et slikt filter, samt intuitivt forstår dets virkemåte.

Virkemåten/idéen er roughly som følger:

1. I (5.1a) filtreres avviket vha. av et første-ordens lavpassfilter med tidskonstant T_f. Det vil si at $z(t)$ er en «glattet ut» versjon av avviket, $e(t)$. Desto større T_f, desto mer glatt og fint er $z(t)$, men det blir dog desto mer på etterskudd i forhold til den reelle e-verdien.

2. I (5.1b) regnes først et estimat av tidsderivatet til $e(t)$ ut via leddet $(e(t)-z(t))/T_f$ (husk at derivatet er definert som $\dot{e}(t) = \lim_{h\to 0} \frac{e(t)-e(t-h)}{h}$). Desto mindre T_f, desto mer nøyaktig blir derivatet, men desto mer støy og «uglattheter» blir sluppet igjennom.

3. Til slutt ganges estimatet av e sitt derivatet med derivatforsterkningen k_D i (5.1b).

Overføringsfunksjonsformen tilsvarende ligningene over er

$$U_{DF}(s) = \frac{k_D s}{1+T_f s} E(s). \qquad \text{(Derivat-filter)}$$

Dette filteret er implementert i PID-blokken i Simulink som $U_{DF}(s) = D\frac{N}{1+\frac{1}{s}N} = D\frac{1}{N+\frac{1}{s}} = D\frac{s}{s\frac{1}{N}+1}$, hvor dermed $D = k_D$ og $T_f = \frac{1}{N}$.

Et slikt filter fører dermed til en *dynamisk* regulator, med indre dynamikk gitt av dynamikken til z. Husk at dynamikk fører til treget, og i dette tilfelle blir dynamikken tregere desto større T_f er. Vi vil derfor helst ta T_f så liten som mulig, men ikke så liten at den slipper igjennom for mye målestøy. Følgende gir en grei huskeregel:

Tommelfingerregel: Ta $T_f = \alpha T_D$ med verdier er $\alpha \in [0.05, 0.2]$, hvor $\alpha = 0.1$ er vanlig.

Oppgave 5.1. Vis at overføringsfunksjonen tilsvarende (5.1) fra e til u_{DF} er (Derivat-filter).

5.1.5 Parallellform, integraltid og derivattid

I industrien, så er PID-regulatorer som oftest gitt på følgende *parallellform:*

$$u_{\parallel}(t) = k_P\left(e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\dot{e}(t)\right) \qquad \text{(parallellform)}$$

hvor T_I er *integraltiden* og T_D er *derivattiden*. Dette tilsvarer følgende i Laplace-domenet:

$$U_{\parallel}(s) = k_P\left(1 + \frac{1}{T_I s} + T_D s\right)E(s).$$

Legg merke til at (parallellform) tilsvarer (PID) med $k_P = k_P$, $k_I = k_P/T_I$ og $k_D = k_P T_D$.

⚠️ **Følg med på definisjoner!** I Simulink brukes i stedet *ideal form* for en regulatorform noe mer likt (parallellform) (dog uten integral- og derivattid), mens *parallel form* brukes til

det vi kaller den ekspanderte formen, altså (PID).

Fun facts, bemerkninger og annet dill dall (you may skip)

Hvorfor kaller man det integraltid og derivattid? 🤔
Integraltiden stammer fra et noe hypotetisk scenario hvor avviket får et sprang fra null til en konstant verdi. Dette spranget i avviket vil forårsake et umiddelbart hopp i proporsjonal-delen, altså $u_P = k_P e$, av regulatoren, mens integraldelen starter fra null og begynner å «integrere seg opp» via $\dot{u}_I = \frac{k_P}{T_I} e$. Integraltiden er da tiden det tar før integralleddet har fått samme verdi som proporsjonal-delen (med antagelsen at avviket har holdt seg konstant).

Derivattiden stammer i stedet fra et scenario hvor avviket gradvis starter å øke med en konstant rate fra null. Proporsjonalleddet starter derfor også fra null og øker i verdi, men derivat-leddet forblir konstant. Derivattiden er derfor tiden det tar før proporsjonal-leddet tar igjen derivat-leddet.

5.2. Tuning av PID-regulatorer

5.2.1 Hva er tuning?

Hvorfor skal du lære dette? 🤷‍♀️ Regulatorinnstilling (tuning) er å finne regulatorparametere (f.eks. k_P, k_I og/eller k_D, evt. T_I og T_d) som gjør at systemet (i lukket sløyfe) har ønskede egenskaper. Det er viktig å finne gode regulatorparametere, siden en dårlig innstilt regulator (som det finnes mange av i industrien [Skogestad, 2003]) kan lede til trege responser, lav robusthet og unøyaktig regulering av ønsket settpunkt; i verste fall kan det til og med lede til en ustabil prosess.

Hva er en godt innstilt regulator? Dette avhenger naturlig nok av forskjellige aspekter som problemet, prosessen og bruksområdet. Følgende fire punkter er dog generelt viktige når man skal justere en regulator:

- Stabilitets-design: Den lukkede sløyfen får ønskede stabilitetskarakteristikker.
- Usikkerhets-kompensering: Regulatoren må være *robust* mtp. usikkerhet i modellen.
- Forstyrrelses-fjerning: Effekten av eventuelle forstyrrelse må dempes/minimaliseres.
- Støy-demping: Regulatoren bør ikke mate forsterket målestøy inn i prosessen.

⚠️ **Dessverre** kan en forbedring av ett av disse punktene føre til forverring for ett eller flere av de andre. For eksempel kan man øke stabiliteten til et system ved å øke den kunstige dempingen (noe som ofte tilsvarer D-leddet i en PID-regulator), men dette kan gjengjeld øke systemets sensitivitet til målestøy.

Man må derfor som regel nøye seg med et **kompromiss** mellom det regulerte systemets

- ***ytelse*** (f.eks. hvor fort det responderer); **og**
- ***robusthet*** (f.eks. hvor sensitivt det er til støy, usikkerhet og forstyrelser).

Ytelses-målene, fordelt på utmerket referanseføling og forstyrrelsesavvisning, bør balanseres opp mot robusthetsmålet i forhold til stabil drift over et bredt spekter av forhold og arbeidsområder.

Men hvorfor trenger man tuning-prosedyrer? 🤔 Selv om en PID-regulator kun har tre parametere man kan endre (fire hvis vi tar med et derivat-filter), er det ikke ikke alltid lett å finne gode verdier (innstillinger) for disse ved bare prøving og feiling. En systematisk prosedyre, spesielt en med så få frihetsgrader (parametere) som mulig, er derfor av stor verdi (gitt at den fungerer da!). Fra [Skogestad, 2003] har vi at tuning-regler helst bør:

1. være godt motiverte, og gjerne modellbaserte og analytisk avledet;
2. være enkle og lette å huske;
3. fungere godt på et bredt spekter av prosesser.

Merk også at:

- **Regulatorinnstillinger trenger ikke å være nøyaktig bestemt:** Generelt vil en liten endring i en pregulatorparameterene fra sin «beste» verdi (f.eks. ± 10 %) ha liten effekt på responsen til den lukkede sløyfen.
- **For de fleste prosesser er det ikke mulig å manuelt stille inn hver regulator:** I prosessindustrien kan en ingeniør ha ansvar for så mange som 300 til 1000 reguleringssløyfer, slik at det ikke alltid er mulig å justere hver regulator manuelt. I stedet fokuserer man på reguleringssløyfene som oppfattes å være de viktigste eller problematiske. De andre regulatorene kan vanligvis bare bruke forhåndsinnstillingnene.

Ønsker alltid negativ tilbakekobling: Direkte- vs reversvirkning

Hvorfor skal du lære dette? 🤷‍♀️ Et viktig spørsmål å spørre seg er: skal regulatorparameterne (k_P, k_I, k_D) ha positivt eller negativt forteng? Velger man feil, så får man positiv tilbakekobling, og systemet blir ustabilt! Man bør derfor alltid vite om den kommersielle regulatoren man bruker skal stilles som *direktevirkende* eller *reversvirkende*.

Hvis vi definerer avviket som $e = r - y$, så tilsvarer positive fortegn på parametene (k_P, k_I, k_D) det vi kaller for *reversvirkning*, mens *direktevirkning* betyr negative fortegn.[3] Bruk derfor:

- **Reversvirkning** hvis et (positivt) sprang i pådraget (rettere sagt, signalet inn til pådraget) fører til en *økning* i prosessvariabelen;
- **Direktevirkning** hvis et (positivt) sprang i pådraget (rettere sagt, signalet inn til pådraget) fører til en *reduksjon* i prosessvariabelen.

Merk at når man bruker en PID-regulator på parallellform, så endrer man kun fortegnet til k_P mens man lar integral- og derivattidene, T_I og T_D, være positive.

[3] Noen bøker bytter om på disse definisjonene. Merk også at flere kommersielle regulator har også en egen innstilling for dette, og krever at alle parameterne har positive forteng.

Fun facts, bemerkninger og annet dill dall (you may skip)

Hvorfor kalles det revers- og direktevirking? Anta, for enkelhets skyld, at vi har en P-regulator, og at vi ønsker og regulere om referansen/settpunktet $r = 0$, altså: $u = k_P e = k_P(r - y) = -k_P y$. Hvis k_P er positiv, så vil en positiv endring i utgangen, $\Delta y > 0$, føre til en negativ, eller *revers*, endring i pådraget, $\delta u < 0$. Hvis k_P derimot er negativ vil en positiv endring $\Delta y > 0$ gi en positiv endring $\Delta u > 0$ i pådraget – altså en *direktevirkning*.

Mer generelt, så avhenger dette av to ting:

1) hvordan vi definerer avviket/feilen; og

2) hva som er «positiv retning» for aktuatoren som gir pådraget (fører en økning i signalet som sendes til aktuatoren til en økning eller reduksjon i prosessvariabelen?).

Når det gjelder 1), så bruker vi jo i disse notatene hovedsakelig $e = r - y$ (se fig. 5.1), men egentlig er $e = y - r$ en mer egnet definisjon av et avvik siden da $e > 0$ når $y > r$, og motsatt ellers. For å illustrere 2), kan vi for eksempel se på første-ordens systemet $\dot{y} = bu$ med en konstant b. Si at vi ønsker å styre utgangen y til et konstant settpunkt r. Gitt avviket $e = r - y$, så har vi $\dot{e} = -\dot{y} = -bu$, slik at en proporsjonalregulator gir $\dot{e} = -bk_P e$, og dermed $e(t) = e(0)\exp(-bk_P t)$. Vi må derfor ha *positive* fortegn for positiv b, og negative ellers.

Sjekkliste:

1. Brukes $e = y - r$ eller $e = r - y$ i regulatoren du bruker?
2. Er regulator-typen/-formen av typen du tror/bruker (f.eks. (PID) vs (parallellform))?
3. Har regulatoren mulighet for endre mellom revers- og direktevirkning eller kan du endre parameterfortegn?
4. Hvordan relateres signal fra regulatoren til pådragets respons?
5. Still inn regulatoren slik at du får negativ tilbakekobling (stabil avviksdynamikk).

5.2.2 Ziegler-Nichols Lukket-sløyfe-metode

Alternative kilder: Wikipedia.

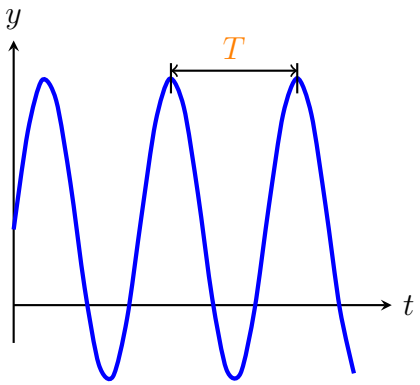
Ziegler og Nichols (ZN) sin lukkede-sløyfe justeringsregel[4] har trolig vært den mest brukte tuning-reglen for PID-regulatorer i over 50 år. Selv om metoden ikke er perfekt (mer om det om litt), så bør man kjenne til den.

Ziegler-Nichols' lukkede-sløyfe-metode for (PID) steg for steg:

For en PID-regulator på parallellform, altså $u(t) = k_P\left(e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\dot{e}(t)\right)$:

Steg 1: Sett $k_P = k_k$ hvor k_k er liten (≈ 0), $T_I = \infty$ (altså $k_P/T_I = 0$) og $T_D = 0$;

[4] Ziegler og Nichols foreslo også en *åpen-sløyfe-metode*.



Figur 5.3: Illustrasjon av stående svingninger for Ziegler-Nichols' andre metode.

Steg 2: Sett et sprang i referansen, altså fra $r = 0$ til $r =$ positiv konstant.

Steg 3: Øk k_k til prosessvariabelen får stående og repeterende oscillasjoner (se fig. 5.3);

Steg 4: Mål periodetiden T til oscillasjonene og sett regulator-parameterne ut fra tabellen:

Regulatortype	k_P	T_I	T_D
P	$0.5k_k$	∞	0
PI	$0.45k_k$	$0.8T$	0
PD	$0.8k_k$	∞	$0.125T$
PID	$0.6k_k$	$0.5T$	$0.125T$

Merk: Den kritiske forsterkningen k_k er forsterkningen som bringer den lukkede sløyfen til stabilitetsgrensen (marginal stabilitet).

Fordeler:

- 👍 Den trenger bare én test for å bestemme parameterene;
- 👍 Testen er gjort i lukket-sløyfe, slik at den også bedre tar hensyn til dynamikk fra aktuatorer og sensorer.

Ulemper:

- 💩 Krever at man kan oppnå stående svingninger (oscillasjoner) med kun en proporsjonalregulator (dette er dog vanlig i prosessindustrien).
- 💩 En prøve-og-feile-strategi for å finne k_k og T kan være veldig tidkrevende, spesielt for systemer med treg dynamikk.
- 💩 Metoden krever at man gjør den lukkede sløyfen marginalt stabilt (stående svingninger). Hvis noe uønsket skjer under innstillinger (f.eks. forstyrrelser eller endringer i prosessen), så kan dette lede til farlige eller uønskede situasjoner.

💩 Regulatoren blir bestemt på grunnlag av to parametere, den kritiske-/ultimate-forsterkningen k_k og -periodetiden T. En FOPTF-modell, derimot, er gitt av tre parametere, (k, τ, θ), noe som betyr at ZN-reglene ikke kan fungere godt på et bredt spekter av slike prosesser.

⚠️ I tilegg til ulempene, er også tre store problemer med ZN-reglene [Grimholt, 2018]:

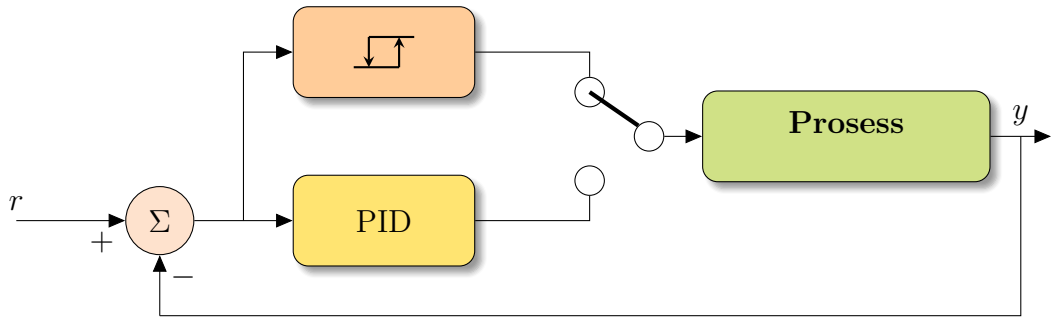
1. Innstillingene er ganske aggressive for de fleste prosesser, og kan føre til svingninger og overskridelser.
2. Regelen inneholder ingen innstillingsparameter for å justere robustheten og gjøre regulatoren mindre aggressiv ved behov.
3. For en ren tidsforsinkelsesprosess, altså $\dot{y} = Ku(t-\theta)$, gir ZN-PID-innstillingene ustabilitet og ZN-PI-innstillingene gir svært dårlig ytelse.

5.2.3 Auto-tuning: Åstrøms relé-metode*

Alternative kilder: [Seborg et al., 2016, §12.5.2]; [Åström and Hägglund, 1984].

Metoden vi skal se på nå kan brukes til til såkalt *auto-tuning*, altså automatisk (selv-)justering av (PID-)regulatorparametere. Vi snakker med andre ord om automatisering av en automatiserings-algoritme! 🤩 Idéen bak metoden er som følger:

Idé: Bruke en relé- (av/på-) regulator med hysterese (kan bare endre verdien (fra av til på eller motsatt) en viss tid etter en endring). Dette skaper stående oscillasjoner (grensesvinginger) i prosessen. Ved å lese av forsterkningen og periodetiden, så kan man bruke tabellen fra Ziegler–Nichols-metoden. Siden man kan skape svingningene uten å måtte ta systemet til stabilitetsgrensen (slik man måtte for ZN-metoden), så er det mulig å automatisere hele prosessen.



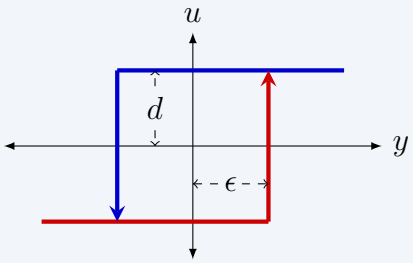
Figur 5.4: Åstrøms (relè) auto-tuning-metode.

Prosedyre for Åstrøms auto-tuning relé-metode:

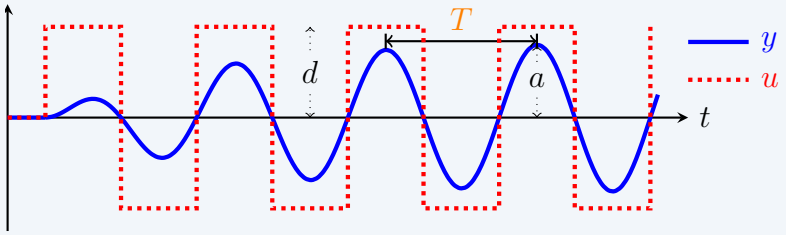
For en PID-regulator på parallellform, altså $u(t) = k_P\left(e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\dot{e}(t)\right)$:

1. *Initialisering:* Prosessen bringes til ønsket arbeidspunkt, enten av operatøren i manuell-modus eller av en tidligere innstilt regulator i auto-modus. Når sløyferesponsen har blitt stasjonær (transienter har dødd ut), så kan auto-tuningen begynne.

2. *Auto-tuning:* PID-regulatoren er midlertidig frakoblet og erstattet med et relé med hysterese, som vist i figuren under. Hysteresens bredde ϵ bestemmes automatisk fra støynivået for å unngå for hyppig endring (fra av til på og motsatt) i pådraget. Under svingningen vil amplituden til reléet, d, justeres slik at prosessvariablen får stående (konstante) svigninger av en ønsket amplitude.



3. *Prameteravlesning:* Regn ut den ultimate forsterkningen og perioden via $k_k = \frac{4d}{\pi a}$ og T, hvor d er relé-amplituden til pådraget, a er amplituden til de stående svingingene i prosessvariabelen, og T er periodetiden, som vist i følgende figur:



4. *Parametersetting:* Gitt k_k og T, regn ut regulator-parameterne vha. (f.eks.) ZN-tabellen:

Regulatortype	k_P	T_I	T_D
P	$0.5k_k$	∞	0
PI	$0.45k_k$	$0.8T$	0
PD	$0.8k_k$	∞	$0.125T$
PID	$0.6k_k$	$0.5T$	$0.125T$

Viktig! Hvis man bruker hysterese, så bør man egentlig ta hensyn til hysterese-bredden når man regner ut den kritiske forsterkningen. Dette er fører dog til noe mer komplekse uttrykk som vi dermed utelater; se [Åström and Hägglund, 1984] for ytterligere detaljer.

Fordeler:

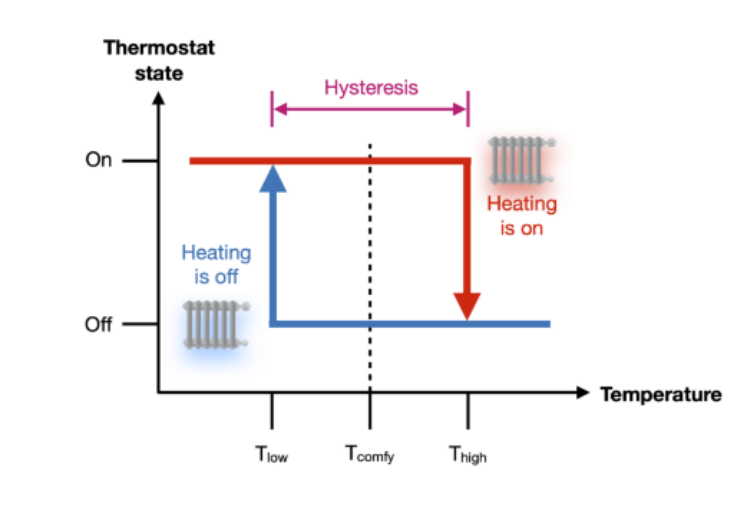
- 👍 Bare én enkelt eksperimentell test kreves i stedet for en prøve-og-feile-prosedyre.
- 👍 Amplituden til prosessutgangen kan begrenses ved å justere relé-amplituden.
- 👍 Prosessen er ikke tvunget til en stabilitetsgrense.
- 👍 Den eksperimentelle testen kan (relativt) lett automatiseres.

Ulemper:

- For langsomme prosesser er det kanskje ikke akseptabelt å utsette prosessen for de to til fire syklusene med svingninger som kreves.
- Hensyn må tas til eventuelle (konstante) forstyrrelser, som kan forårsake asymmetriske oscillasjoner.
- Tilvarende ZN-metoden, blir regulatoren bestemt på grunnlag av to parametere, den kritiske-/ultimate-forsterkningen k_k og -periodetiden T. En FOPTF-modell, derimot, er gitt av tre parametere, (K, τ, θ), noe som betyr at denne metoden heller ikke kan fungere godt på et bredt spekter av slike prosesser.
- Patentbeskyttet (?).

Fun facts, bemerkninger og annet dill dall (you may skip)

Termostater og av-på-regulatorer med hysterese: lage egen figur



5.2.4 SIMC for PI-tuning fra sprangresponser

Alternative kilder: [Grimholt, 2018]; [Balchen et al., 2016, §9.3.3]); §2.7 i [Skogestad and Postlethwaite, 2007].

Vi skal nå se på Skogestads enkle intern-modell kontroll-metode (abbrivert SIMC) for PI-regulatorer; eller bare SIMC-metoden (eng.: simple internal model control) for enkelhets skyld.

Målet er å stille inn en PI-regulator på ved å anta en første-ordens-pluss-tidsforinskelse-prosess (FOPTF). **Men hvor har vi FOPTF-modellen fra?** 🤔 Jo, man kan for eksempel finne/approksimere en slik modell ved å tilpasse en sprangrespons til en FOPTF-modell. Dette skal vi se på om litt, men først tar vi selve SIMC-metoden. Denne er som følger:

Skogestad enkle intern-modell innstillings-regler (SIMC):

Gitt et første-ordens-pluss-tidsforsinkelse system:

$$\dot{y} = \frac{1}{\tau}\left(-y + Ku(t-\theta)\right).$$

Ta parameterne til en PI-regulator[a],

$$u(t) = k_P\left(e(t) + \frac{1}{T_I}\int_0^t e(\sigma)d\sigma\right),$$

som

$$k_P = \frac{1}{K}\frac{\tau}{(\tau_c+\theta)}, \quad T_I = \min\left(\tau, 4(\tau_c+\theta)\right). \qquad \text{(SIMC-regler)}$$

der τ_c er en tuning-parameter (tilsvarer delvis tidskonstanten til den lukkede sløyfen).
Tommelfingerregler:

- $\tau_c = \frac{3}{2}\theta$ for robusthet;
- $\tau_c = \frac{1}{2}\theta$ for ytelse;
- $\tau_c = \theta$ for god balanse mellom disse.

[a]Det finnes også tilsvarende regler for å stille inn en PID-regulator gitt en andre-ordens-pluss-tidsforsinkelses-modell.

Fordeler:

- 👍 Enkel og lett å huske.
- 👍 Det er kun en variabel som må stilles inn, nemlig τ_c; det kan brukes til å få et ønsket kompromiss mellom ytelse og robusthet.
- 👍 Fungerer godt på mange prosesser, med tilnærmet optimal ytelse (til en PID-regulator å være) for visse relevante systemer (se [Grimholt and Skogestad, 2013]).

Ulemper:

- 💩 Oscillatorisk åpen sløyfe prosess kan være vanskelig å håndtere.
- 💩 Ustabile prosesser har ikke fornuftig sprangrespons, og metoden kan ikke brukes.
- 💩 Høyere-ordens prosesser kan noen ganger kreve mer avansert regulering enn PI.
- 💩 Fenomener som ulinearitetet kan gjøre det vanskelig å tune enkle PI-regulatorer.
- 💩 Kan være utfordrende å bruke for multivariable systemer pga. krysskoblinger.

Kort om tilpasning av FOPTF-modell fra sprangresponser

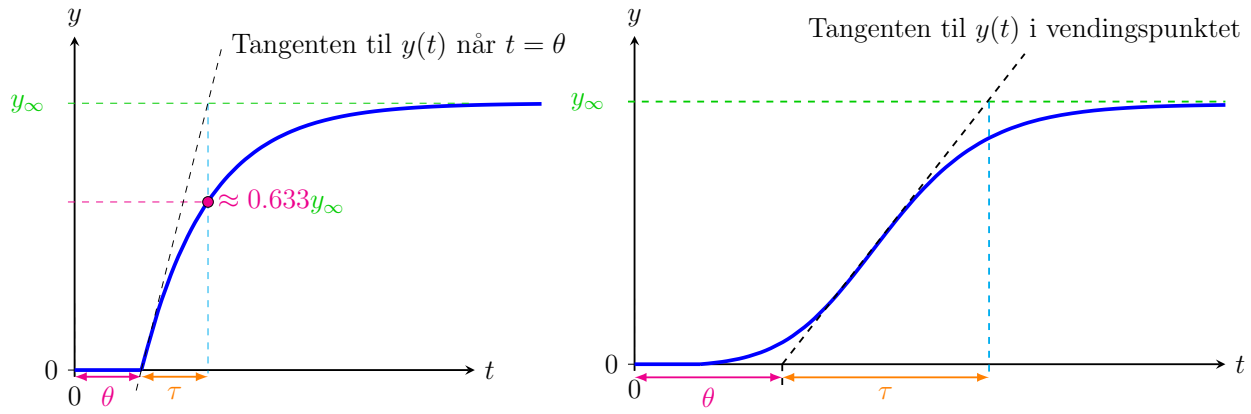
Hvorfor skal du lære dette? 🤷‍♀️ For reguleringstekniske formål trenger vi ikke alltid en global modell (en modell som er gyldig for alle tisltander) slik som de vi utledet i kapittel 3. I stedet er ofte en modell som tilnærmer prosessen/systemet godt nært det ønskede arbeidsområdet bra nok. Den kanskje enkleste måten å tilnærme slik modell er å tilpasse en gitt modell fra data fra en sprangrespons.

Vi ønsker nå å tilpasse et system vha. av en første-ordens-pluss-tidsforsinkelse(FOPTF)-modell:

$$\dot{y} = \frac{1}{\tau}\left(-y + Ku(t-\theta)\right). \tag{FOPTF}$$

Prosedyre for tilpassing av FOPTF-modell fra sprangrespons:

1. Ta utgangspunkt i en enkel sprangrespons (u fra u_0 til u_s) i åpen sløyfe, med antagelsen om at utgangen, y, starter i ro med verdi y_0.
2. Tilpass en førsteordens modell med tidsforsinkelse se (FOPTF) til responsen:
   (a) Bruk figur 5.5a hvis responsen «ser ut som» en FOPTF-respons;
   (b) bruk figur 5.5b hvis det er tegn til høyere-ordens dynamikk;
   (c) ta $K = \frac{\Delta y}{\Delta u}$ hvor $\Delta y = y_\infty - y_0$ og $\Delta u = u_s - u_0$.



(a) FOPTF-karakteristikk.

(b) Tegn på høyre-ordens dynamikk.

Figur 5.5: Sprangsresponger for tilpassing av FOPTF-modell.

5.2.5 Etterjustering og manuell tuning*

Hvorfor skal du lære dette? 🤷‍♀️ Anta én av følgende to scenarier:

1. Du har stilt inn PID-regulatoren med metodene du har lært.
2. Regulatoren ble stilt inn for en tid tilbake, og fungerte bra da.

Men du ikke helt fornøyd med innstillingene, selv om de på et vis fungerer. Kanskje du har brukt en litt unøyaktig matematisk modell, eller det har skjedd en endring i prosessen? Kanskje metoden du brukte i var helt ideell? En mulig løsning på dette er å **etterjustere** regulatoren.

Du har bestemt deg for å etterjustere, hva nå? Etterjustering er ingen eksakt vitenskap, men mer en blanding av intuisjon, forståelse (av både prosessen og regulator-delene), samt kunnskap og erfaring. Følgende huskeregler kan være et utgangspunkt, men er ingen fasit/guide:

Huskeregler:

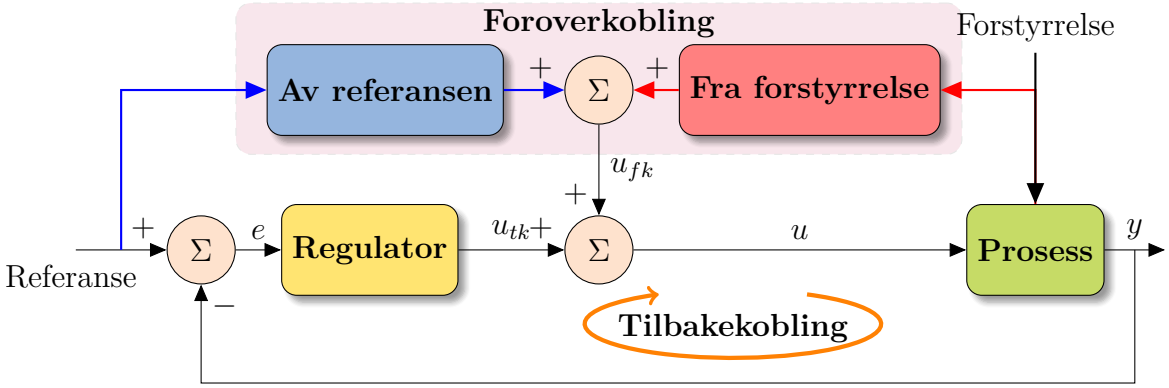
- Er sløyfa urolig (svingete/ oscillerende): reduser k_P og/eller k_I; øk **eller** reduser k_D.
- Er det dynamiske avviket for stort: øk k_I og/eller k_P og/eller k_D.
- Er utgangen eller aktuatoren urolig/"vibrerende": reduser k_D og/eller k_P.

6. Foroverkobling og nominelle pådrag

Alternative kilder: Brian Douglas video; Wikipedia; Kap. 16 i [Haugen, 2023].

Hvorfor skal du lære dette? 🤷 Ofte har vi ekstra kunnskap og informasjon om systemet som vi kan utnytte utover bare målinger av utgangen. For eksempel så kjenner vi jo referansen, samt har vi ofte noe kunnskap om systemets dynamikk (i form av en matematisk modell); kanskje har vi også målinger av noen forstyrrelser? Vi kan bruke slik kunnskap og informasjon til å forbedre regulatoren vår, i form av både bedre referanseføling og forstyrrelsesavisning, noe vi ikke kan få til samtidig med tilbakekobling alene!

I motsetning til tilbakekobling, som baserer seg på avviket, så blir informasjonen vi bruker i en foroverkobling (altså signalene tilsvarende referansen og/eller de målte forstyrrelsene) på et vis «matet fremover» for å regne ut pådraget. Eller sagt på en annen måte: signalene blir *foroverkoblet*. En foroverkobling vil derfor påvirke pådragsverdien, mens pådraget aldri vil påvirke den fremtidige verdien til foroverkoblingen, som bare blir bestemt av referansen og/eller de foroverkoblede forstyrrelsene. Dette er i motsetning til en tilbakekobling, som naturlig nok vil påvirke den fremtidige pådragsverdien siden den både påvirker og er basert på utgangen.



Figur 6.1: Foroverokoblinger av både referansen (i blått) og forstyrrelsen (i rødt).

Vi skal skille mellom de to typene foroverkobling som er vist i figur 6.1:

- **Foroverkobling fra forstyrrelsen:** baseres på måling av forstyrrelse.
- **Foroverkobling av referansen:** baseres seg på kunnskap om referansen.

Designprosedyren for regulator med både tilbakekobling og foroverkobling er som følger:

1. Designe *tilbakekoblingen* for å:
   - minimere sensitiviteten til forstyrrelser;
   - minimere/attenuere effekten av målestøy;
   - øke robustheten mtp. usikkerhet og variasjoner i prosessen.
2. Deretter designe *foroverkobling av referansen* for å oppnå ønsket respons gitt referansen $r(t)$, samt designe *foroverkobling fra forstyrrelsen* for å raskt eliminere effekten av en målt forstyrrelse.

Statisk vs dynamisk foroverkobling: Vi skiller mellom statiske og dynamiske foroverkoblinger. I en statsik foroverkobling vil pådraget u_{fk} kun være en skalering av den nåværende verdien til referansen, $r(t)$, og/eller forstyrrelsen, $v(t)$. Eksempler på statisk foroverkoblinger er dermed $u_{fk}(t) = 2r(t)$, $u_{fk}(t) = \frac{1}{2}v(t)$ eller $u_{fk}(t) = 2r(t) + \frac{1}{2}v(t)$. I en dynamisk foroverkobling vil man ha et *dynamisk* pådrag, f.eks. $\dot{u}_{fk} = -u_{fk} + \dot{r} + 2r$. **Merk:** Vi vil hovedsakelig fokusere på statiske foroverkoblinger i disse notatene, selv om også dynamiske foroverkoblinger, til en viss grad, også blir aktuelt i forohld til referanseføling.

6.1. Foroverkobling fra forstyrrelse

Poenget med en foroverkobling fra forstyrrelsen er å kompensere for effekten av målte forstyrrelser. En forstyrrelse er som sagt en (ekstern/eksogen) størrelse som påvirker prosessen og som vi ikke kan styre (altså ikke et pådragsorgan).

Oppgave 6.1. Gitt et tanksystem hvorden målte prosessvariabelen er væskehøyden i tanken. I et slikt scenario trenger ikke nødvendigvis en lekkasje i bunnen av tanken å regnes som en forstyrrelse. Hvorfor ikke? Er det dog noen scenarier hvor du kan tenke på en slik lekkasje som er forstyrrelse? Og hva er i så fall spesielt i det tilfelle?

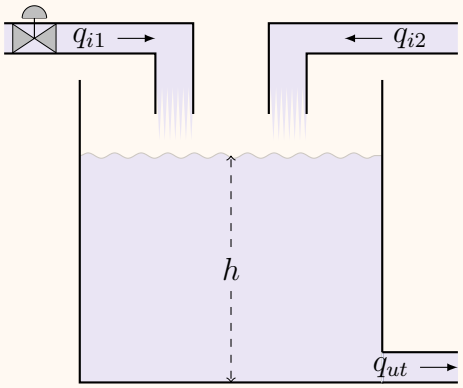
Eksempler på forstyrrelser inkluder ting som varierende luftmotstand for en bil grunnet vind og havstrømmer som påvirker en ubåt, til mer eksotiske ting som trykket fra solvind på en satellitt sine solcellepanel. En ukjent innstrøm i en tank er også som en forstyrrelse å regne:

Eksempel 6.1. (**Ukjent innstrøm i en tank**)

Gitt tanksystemet vist i figuren til høyre. Det har to innstrømmer av samme væske, q_{i1} og q_{i2}, og én utstrøm, q_{ut}. Man kan endre q_{i1} vha. en reguleringsventil; innstrømmen q_{i2} har man derimot ingen kontroll over, men den kan måles.

Det er ønskelige å regulere væskehøyden h i tanken til en ønsket verdi, $h_ø$. Ved å anta at $q_{ut} = c_{ut}h$ (se eks. 3.1), så har vi

- Utgang/tilstand: $y = h$;
- Referanse/settpunkt: $r = h_ø$;
- Pådrag: $u = q_{i1}$;
- Forstyrrelse: $v = q_{i2}$.



Vi antar at væsken har konstant massetetthet ρ og at tanken har konstant tverrsnittsareal A. Fra massebalanse (se 3.2) får vi, ved å ta $a = c_{ut}/A$ og $b = 1/A$, at systemets dynamikk er beskrevet av følgende første-ordens differensialligning (se eksempel 3.1):

$$\dot{y} = -ay + b(u + v). \qquad \text{(Prosessdynamikk)}$$

Scenario 1 – Trenger bare tilbakekobling: Hvis både r og v er konstante eller endrer seg svært sakte, så vil en godt innstilt PI-regulator fungere bra nært et ønsket arbeidsområde.

Scenario 2 – Behov for foroverkobling: Hvis $r(t)$ og/eller $v(t)$ endrer seg (varierer med tiden), så vil ikke en PI-regulator fungere godt på egenhånd ved raske endringer grunnet tregheten i I-leddet. Her kan foroverkobling være en mulig løsning. For å finne ut hvordan en slik foroverkobling kan se ut, definerer vi avviket $e(t) = r(t) - y(t)$ og finner avviksdynamikken:

$$\dot{e} = \dot{r} - \dot{y} = \dot{r} + ay - b(u + v). \qquad \text{(Avviksdynamikk)}$$

Vi ønsker jo at avviket skal bli null, noe vi oppnår hvis vi velger pådraget u slik at $\dot{e}e < 0$ for alle $e \neq 0$. Følgende kadidat oppnår dette når r er konstant (slik a $\dot{r} = 0$):

$$u = \underbrace{-v}_{\text{Foroverkobling fra forstyrrelsen}} \underbrace{+ k_P e + k_I \int_0^t e(\tau)d\tau}_{\text{PI-regulator}}$$

Foroverkoblingen fra forstyrrelsen eliminerer derfor effekten av forstyrrelsen, slik at PI-regulatoren kan justeres utelukkende mtp. å få ønsket ytelse i forholds til referanseføLgingen.

Forstyrrelsen i eksempelet over er hva jeg vil kalle en **matchet forstyrrelse**, siden vi kan bruke pådraget til å direkte kansellere dens effekt. På grunn av ting som aktuatordynamikk og eventuelle tidsforsinkelser er det ofte ikke mulig i praksis. I slike tilfeller kan man allikevel som regel finne en tilnærmet (ofte dynamisk) fra forstyrrelsen som forbedrer regulatoren ytelse. Dette vil du lære mer om i senere fag.

Fun facts, bemerkninger og annet dill dall (you may skip)

Ofte har man ikke direkte målinger av forstyrrelser tilgjengelig. Har man derimot en god modell av det dynamiske systemet, samt måling av tilstandene, så kan man i foroverkoblingen i stedet bruke et estimat av en forstyrrelse som er generert vha. en matematisk modell,

såkalte **forstyrrelses-estimatorer** (eller "disturbance observers" på engelsk).

6.2. Foroverkobling av referansen og nominelt pådrag

I denne seksjonen skal vi se på nominelt pådrag og regulatorer med to frihetsgrader. Man tenke på det nominelle pådraget som et pådrag man kan regne ut i forkant og som perfekt utfører den ønskede oppgaven i åpen sløye under antagelsen om en ideell (perfekt) verden. En regulator med to frihetsgrader kan på den annen side ses på som en metode hvor man «tuner» en foroverkobling av referansen når man ikke nødvendigvis har en god modell av prosessen, samt kun kjenner nåverdien til referansen.

6.2.1 Nominelt pådrag

Hvorfor skal du lære dette? 🤷‍♀️ Et *nominelt pådrag*, u_{nom}, er ment til å gi den pådragsverdien som i teorien vil opprettholde et ønsket arbeidspunkt (tvunget likevektspunkt), noe som kan føre til forbedret ytelse siden det kan tillate en reduksjon i integral-leddet. Flere kommersielle regulatorer tillater en å sette et nominelt pådrag.[a]

[a] Det tilsvarer ofte det manuelle pådraget, u_{man}, som kan endres når regulatoren er i manuell-modus.

Et nominelt pådrag kan tenkes som en standard utvidelse av en PID-regulator:

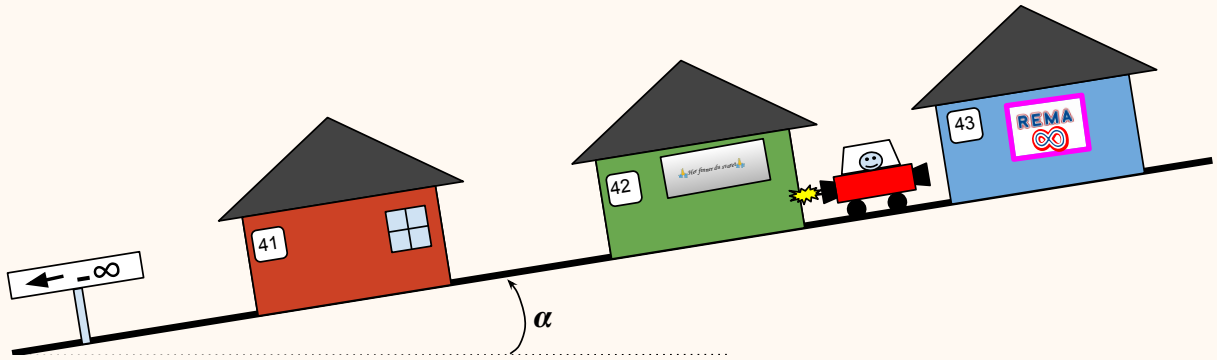
$$u_{PID}(t) = \underbrace{u_{nom}}_{\text{nominelt pådrag}} + \underbrace{k_P e(t)}_{\text{proporsjonal}} + \underbrace{k_I \int_0^t e(\tau)d\tau}_{\text{integral}} + \underbrace{k_D \dot{e}(t)}_{\text{derivat}}. \tag{6.1}$$

Strengt tatt kan man sette verdien til u_{nom} til hva enn man ønsker, og det er derfor også noen ganger (i visse settinger) kalt et manuelt pådrag. Hva som er en «god» verdi å sette det til er ikke alltid like lett å finne ut av; det kan finnes fra en matematisk modell eller eksperimentelt, noe som kan være tidkrevende. Det nominelle/manuelle pådraget er derfor ofte skrudd av i prosessindustrien til fordel for et aktivt integral-ledd. Dette er helt OK, og på ingen måte direkte «feil». Hvis mulig, så bør du allikevel prøve å sette verdien til en hensiktsmessig verdi som hjelper regulatoren din.

⚠️ **Pass på!** Et dårlig innstilt nominelt pådrag vil kunne bli virkende som en konstant forstyrrelse som må kompenseres for ved hjelp av et I-ledd.

La oss ta et eksempel som viser virkemåten til et slikt ledd:

Eksempel 6.2. Tom skal opp en bakke (nominelt pådrag). Tom i Tallinjeveien (se § 1.4) trenger nok en gang vår hjelp. Denne gang ønsker han at vi forbedrer cruisekontroll-systemet til bilen hans slik at det fungere bedre i en bakke i Tallinjeveien.



Bakken det er snakk om er vist i figuren over. Den er ganske lang (fra hus 31 til hus 50141924) og har konstant helning på α grader. Tom har observert at en PI-regulator med et stort I-ledd må til for å fort kunne fjerne det statiske avviket som oppstår, men at dette fører til treghet og oversing når bakken slutter. Han lurer derfor på om vi kan modifisere regulatoren slik at vi forbedrer ytelsen til regulatoren.

Metoden vi skal foreslå til Tom er selvsagt at han bruker et nominelt pådrag, som i dette tilfellet tilsvarer at vi kompenserer for tyngekraften. Du kan selve vise (med antagelse som ingen tap og forstyrrelser) at bevegelsesligningen til Toms bil i bakken er

$$m\dot{x}_h = u - mg\sin(\alpha)$$

hvor m er massen til bilen med Tom oppi (som vi tidligere bare har antatt er lik 1 kg), hvor x_h er bilens hastighet (i strekmeter per sekund), u betgener kraften fra jetmotorene, og hvor g er tyngdekraften (i $\mathrm{m/s^2}$). Hvis vi derfor legger til det nominelle pådraget

$$u_{nom} = mg\sin(\alpha)$$

til PI-regulatoren (altså som (6.1) uten D-leddet), så får vi

$$\dot{x}_h = \frac{1}{m}\left(k_I e(t) + k_I \int_0^t e(\tau)d\tau\right).$$

Dette er akkurat samme form som når bakken ikke er der! 🙌

I praksis er som regel det nominelle pådraget bare satt til en konstant verdi (altså et manuelt pådrag). Det trenger dog ikke være det. For eksempel fungerer det nominelle pådraget i det forrige eksempelet akkurat like godt hvis helningsgraden α varierer så lenge vi kan måle den. I sistnevnte scenario blir dog helningsgraden mer som en forstyrrelse å regne, slik at dette da passer bedre under kategorien «foroverkobling fra forstyrrelsen» enn under «nominelt pådrag». Det er med andre ord tidvis glidene skiller her, og hva som er hva er nødvendigvis ikke alltid så viktig; det viktige er å bruke informasjonen vi har tilgjengelig til å forbedre regulatoren! 🦾

Så hva er et nominelt pådrag, sånn egentlig? 🤔 Følgende er min egen lille hjemmebrygga «definisjon» som fanger hvordan jeg vil at dere alltid skal tenke på dette pådraget, siden dette lar seg generalisere både langt og bredt:

Et **nominelt pådrag** er pådraget som i teorien opprettholder en ønsket (konstant eller varierende) referanse i åpen sløyfe gitt en perfekt matematisk modell og ingen forstyrrelser.

Normalt sett er man ute etter å bruke et nominelt pådrag til å opprettholde et ønsket arbeidspunkt selv uten et (stort) integral-ledd. For et dynamisk system på formen $\dot{x} = f(x, u)$ og et arbeidspunkt, x_s, så må det nominelle pådraget, u_{nom}, da tilfredsstille $f(x_s, u_{nom}) = 0$. Altså er bare $u_{nom} = u_s$ i forhold til et tvunget likevektspunkt/arbeidspunkt slik vi så på i § 2.5.1. Konseptet generaliserer dog som sagt også til visse varierende referanser:

Eksempel 6.3. (Tank med varierende referanse, ukjent innstrøm og ulineær utstrøm) La oss igjen se på tanksystemet fraeksempel 6.1. Vi antar nå dog at utstrømmen er ulineær og på formen $q_{ut} = c_{ut}\sqrt{h}$, samt at referansen er tidsvarierende. Prosess-dynamikken er dermed

$$\dot{y} = -a\sqrt{y} + b(u + v). \qquad \text{(Prosessdynamikk)}$$

Gitt at $\dot{r}(t)$ eksisterer, så er dynamikken til avviket $e(t) = r(t) - y(t)$ gitt ved

$$\dot{e} = \dot{r} - \dot{y} = \dot{r} + a\sqrt{y} - b(u + v). \qquad \text{(Avviksdynamikk)}$$

Vi kan derfor ta

$$u = \underbrace{a\sqrt{r}/b + \dot{r}/b}_{\text{Foroverkobling av ref./nominelt pådrag}} + \underbrace{-v}_{\text{Foroverkobling fra forstyrrelsen}} + \underbrace{k_P e + k_I \int_0^t e(\tau)d\tau}_{\text{PI-regulator}}$$

slik at $\dot{e} = 0$ når $y = r$ selv uten et I-ledd.

⚠**Warnung!** En annen fristende kandidat her er å kansellere det ulineære leddet direkte, altså

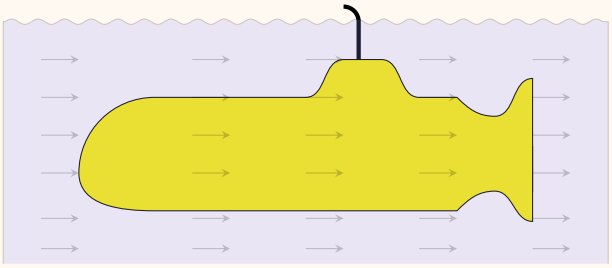
$$u = \underbrace{a\sqrt{y}/b}_{\text{Kansellering av unlineær del}} + \underbrace{\dot{r}/b}_{\text{Foroverkobling av referansen}} + \underbrace{-v}_{\text{Foroverkobling fra forstyrrelsen}} + \underbrace{k_P e}_{\text{P-regulator}}$$

Man får da $\dot{e} = -k_P e$, slik at $y(t) = r(t) + (y(0) - r(0))\exp(-k_p t)$, som er helt superdupert. En slik kansellering av ulineære ledd, såkalt tilbakekoblings-linearisering (eng. "feedback linearization"), er teoretisk sett veldig effektivt, men man skal være forsiktig med dette i praksis. Grunn: ved unøyaktig modell eller måling gjør dette leddet fort mer skade enn nytte. Alternativer: bruke ønsket settpunkt/referansen i stedet for tilstandene (tilsvarende det nominelle pådraget i regulatoren over), eller fjerne hele leddet til fordel for et stort P-ledd eller en PI-regulator.

Et nominelt pådrag kan også baseres på efaringsbasert informasjon, hvor man inkorporerer kunnskapen man har opparbeidet seg om systemet og eventuelle tidligere forstyrrelser man ikke kan måle, etc. Følgende tullete eksempel prøver å illustrere hvordan dette kan tas i bruk.

Eksempel 6.4. Spionubåt i sterk strøm: I forbindelse med sitt meget suksessfulle få-tak-i-rikinger-fra-norge-program, har Sveits 🇨🇭 bedt sin marine sende deres beste (og eneste) ubåt til de norske fjorder for å autonomt innhente informasjon om norske laksebaroner.

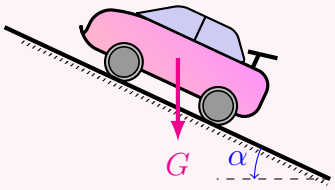
Selv om den autonome ubåten er utstyrt med mekaniske tilbakekoblingssløyfer basert på sveitsisk uverk av ypperste kvalitet, så har de sterke havstrømmene i fjordene gjort det vanskelig å opprettholde en ønsket posisjon.



Heldigvis viser strømmene seg å variere sakte, samt at de svært forutsigbare i forhold til tidevannet (flo og fjære), slik at de over tid kan kartlegges for de forskjellige fjordene. En effektiv løsning er derfor å kompensere for dem vha. et nominelt pådrag som tidsvis blir (automatisk) oppdatert i forhold til lokasjon og tid på døgnet.

Oppgave 6.2. Bil opp en bakke:

En bil med masse $m = 1500\,\text{kg}$ kjører opp en bakke med helningsgrad $\alpha = 25°$. Bilen skal holde en konstant hastighet på $50\,\text{km/t}$. Hjulradiusen er $r = 30\,\text{cm}$. Det er ønskelig å utvikle en cruise-kontroll for dette systemet, i form av en P-regulator med nominelt pådrag.



Spørsmål: Gitt at bilen har firhjulstrekk, hva skal det nominelle pådraget være, i form av et dreiemoment som virker likt på hvert av hjulene, for situasjonen over?

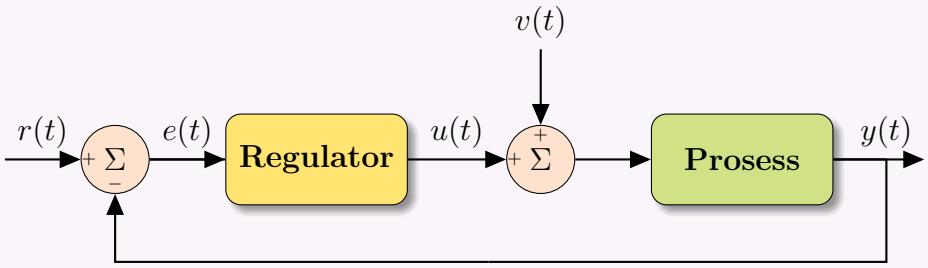
6.2.2 Regulator med to frihetsgrader

Målet med en *regulator med to frihetsgrader* er å tenke på en foroverkobling av referansen som en egen gren av regulatoren (se fig. 6.2) som man individuelt kan «tune», slik at man har

- tilbakekobling for forstyrrelses-fjerning/robushet;
- foroverkobling for ytelse.

Denne strategien er egnet til å løse følgende problem:

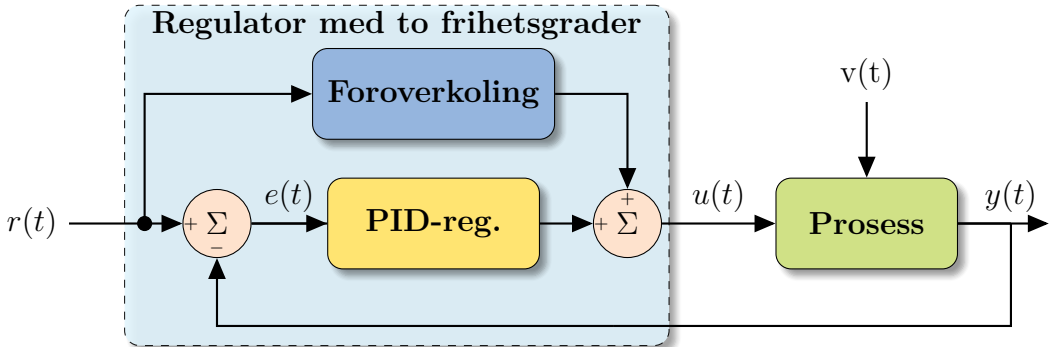
Nåværende problem: Gitt et system som i figuren under



Mål: Vi ønsker både god referanse- og forstyrrelses-respons. **Men**

- vi kan ikke måle forstyrrelsen(e) $v(t)$;
- vi har muligens ingen god modell av prosessen.

⚠ **Ikke mulig med begge deler ved kun ren tilbakekobling:** Ved å optimalisere responsen mtp. sprang i referansen reduserer man ytelsen mtp. forstyrrelsesresponsen hvis vi bare har tilbakekobling, og vice versa.



Figur 6.2: Illustrasjon av regulator med to frihetsgrader.

Følgende eksempel demonstrer påstanden i den siste boksen

Eksempel 6.5. (Eksempel og figur fra [Taguchi and Araki, 2000]

Gitt en FOPTF-prosess:

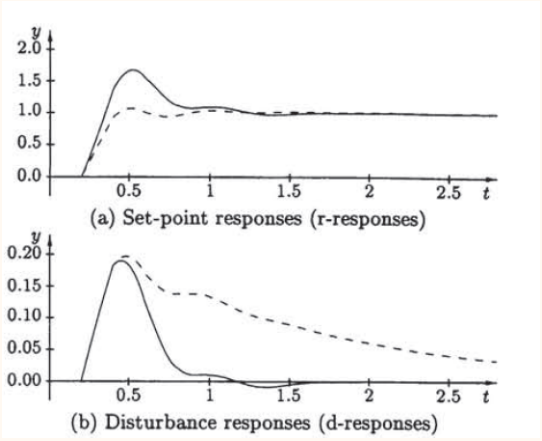
$$\dot{y} = -y + u(t - 0.2) + v.$$

For en PID-regulator på er følgende parametere optimale mhp. responsen etter et sprang i referansen (stiplede linjer):

$$k_p = 4.75, \quad T_I = 1.35, \text{og} \quad T_D = 0.094;$$

mens følgende er optimale mhp. forstyrrelsesresponsen (hele linjer i fig.):

$$k_p = 6.0, \quad T_I = 0.4, \text{og} \quad T_D = 0.084$$



Du kan teste dette selv vha. denne Simulink-modellen.

Vi antar som sagt at vi ikke kan måle forstyrrelsen, og dermed ikke kan legge til en foroverkobling fra forstyrrelsen.

Mulig løsning: Vi justerer PID-regulatoren slik at man får god forstyrrelses-respons. Så legger vi til en foroverkobling av referansen for å øke ytelsen, som (siden vi ikke har en god modell av prosessen) vi «tuner» for å få en god respons mtp. endringer i referansen. Dette gir oss en regulator slik som illustrert i figur 6.2, som ofte sies å ha to frihetsgrader:

Regulator med to frihetsgrader: En modifisert PID-regulator på formen

$$u = k_P(\beta \cdot r - y) + k_I \int_0^t e(\tau)d\tau + k_D(\gamma \cdot \dot{r} - \dot{y}),$$

hvor vi da har to nye parametere vi kan endre, β og γ. [a]

⚠️ Derivatleddet vil her nesten alltid bli implementert med et Derivat-filter i praksis.

Innstillingsprosedyre:

- Juster PID-regulatoren (altså velg k_P, K_I og K_D) for å få ønsket forstyrrelses-respons (ideelt sett tunet vha. kunstige sprang i forstyrrelsen);
- Juster foroverkoblingen (altså β og γ) for ønsket respons fra sprang i referansen.

[a] Denne typen regulator kalles derfor også for en $\beta - \gamma$-PID.

Simulink har egen egen blokk for en PID-regulator med to frihetsgrader, se denne lenken.

6.3. Litt mer om tilbakekobling vs foroverkobling

Tilbakekobling bruker målinger av utgangen/tilstandene man ønsker å regulere (væskehøyden i en tank, hastigheten til en bil, etc.) til å bestemme pådraget for å korrigere for feil.
Fordeler:

- 👍 Regulatoren handler så snart prosessvariabelen avviker fra referansen, uavhengig av årsak.
- 👍 Tilbakekobling krever i utgangspunktet minimalt med kunnskap om prosessen som skal reguleres; f.eks. er ikke en matematisk modell av prosessen nødvendig, selv om det kan være veldig nyttig for både regulator-design og -tuning.
- 👍 PID-regulatorer fungerer tilfredsstillende på de fleste systemer; de er både allsidig og robuste, og kan lett stilles inn på nytt hvis prosessen endrer seg.

Ulemper:

- 💩 Regulatoren reagerer først når et avvik har oppstått. Dermed er perfekt regulering under forstyrrelses- eller settpunkt-endringer teoretisk umulig.
- 💩 Alltid kompromiss mellom ytelse og robusthet.
- 💩 Ikke proaktiv/prediktiv i forhold til kjente eller målbare endringer forstyrrelser og referansen.
- 💩 Kan gi utilfredsstillende regulering for trege prosesser (store tidskonstanter og/eller lange tidsforsinkelser). Ved varierende forstyrrelser er det dermed ikke alltid mulig å få prosessen til ønsket referanse.
- 💩 I noen situasjoner kan man ikke direkte måle prossessvariabelen, slik at tilbakekobling ikke er direkte gjennomførbart.

Foroverkobling bruker målinger/informasjon om variabler/signaler (forstyrrelser eller ønsket referanse) man *ikke* kan/er ute etter å regulere for å bestemme pådraget (i kombinasjon med en tilbakekobling), slik at man ideelt sett oppnår forbedret regulering.
Fordeler:

- 👍 Utnytter ekstra, tilgjengelig informasjon til å forbedre regulatorens ytelse og/eller robusthet.
- 👍 Reagerer umiddelbart på endringer i forstyrrelser og/eller referansen.
- 👍 For lineære systemer påvirker ikke foroverkoblingen stabiliteten til systemet.
  NB! Holder generelt sett ikke for ulineære systemer, altså for de fleste ekte systemer!

Ulemper:

- 💩 Forstyrrelser må måles, noe som krever ekstra sensorer, og som for mange applikasjoner ikke er gjennomførbart.

- 💩 Effektiv bruk av foroverkobling krever en god matematisk modell av systemets dynamikk.
- 💩 Ideelle foroverkoblinger er ofte ikke realiserbare (praktiske tilnærminger er dog mulig).
- 💩 Foroverkobling av forstyrrelsen kan noen ganger komme i «konflikt» med tilbakekoblingen, ved at de ved et sprang i forstyrrelsen begge prøver å løse samme problem.

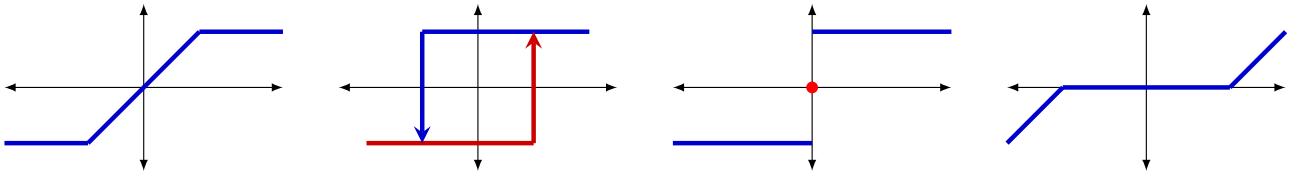
En siste viktig bemerkning:[1] For mange problemer relatert til prosessregulering er lastforstyrrelsesresponsen mye viktigere enn settpunktresponsen. Settpunktresponsen er viktigere i bevegelseskontroll. Få lærebøker og vitenskapelig artikler viser dog dessverre mer enn settpunktbetraktninger.

[1] Sitat av prof. Anders Robertsson, Univ. i Lund; se også Shinskeys bemerkning (f.eks [Shinskey, 2002]).

7. Ulineære systemer

Hvorfor skal du lære dette? 🤷‍♀️ De fleste og mest brukte reguleringsstrategiene bygger på antagelsen om et lineært system. I realiteten er dog «alle» fysiske systemer ulineære. Hvis man derfor ikke tar hensyn til at systemet man implementerer regulatoren på har ulinearitet, så kan det føre til uønsket oppførsel og fare for ustabilitet.

Vi har allerede så vidt sett noen ulinearitet i disse notatene, slik som dynamikken til en pendel i eksempel 3.11 og utstrømmen til en tank i eksempel 6.3. En annen typisk unlinearitet er luftmotstanden som virker på et kjøretøy, som normalt sett modelleres som proporsjonalt med kvadratet til kjøretøyets hastighet. Dette er alle eksempler på kontinuerlige og differensierbare ulinearitet, noe som gjør det mulig å **linearisere** disse om et ønsket arbeidspunkt for å få en lineær modell som tilnærmer dynamikken slik at vi kan bruke den til regulatordesign.



Figur 7.1: Illustrasjon av vanlige ulinearitet.

Det finnes dog tilfeller hvor man ikke (direkte i hvert fall) kan bruke linearisering, og dermed må ta direkte hensyn til ulinearitetene. Dette gjelder spesielt systemer som inneholder såkalte *harde ulinearitet*, som ulineære funksjoner med diskontinuiteter (enten i funkjonsverdi eller den deriverte) eller som ikke er en-entydige, og som av den grunn ikke lar seg linearisere. Harde ulinearitet inkluderer fenomener som ((se fig. 7.1)):

(a) Metning: Alle pådragsorgan har begrensninger. Når de når sin maksimale eller minimale kapasitet oppstår et en ulinearitet siden pådragsverdien ikke lenger endres proporsjonalt, men i stedet flater ut.

(b) Hysterese: Hysterese refererer til et fenomen der en respons ikke bare avhenger av nåværende påvirkning, men også av tidligere påvirkninger. Dette kan føre til en slags forsinkelse, hvor et systemsrespons faktisk kan være forskjellig avhengig om påvirkningene på det (fra f.eks. et pådragsorgang) er økende eller minkende input. Dette kan observeres i systemer som termostater, hvor det er lagt inn en distinkt forskjell mellom på- og avslåingspunktene for å unngå hyppig veksling, noe som gir en slags «dødsone» der systemet ikke reagerer på umiddelbare endringer i inngangen.

(c) Tørr-(Coulomb-)friksjon: Når en overflater beveger seg relativt til en annen flate den er i kontakt med, uten tilstedeværelse av en smørende væske, vil tørr-(Coulomb-)friksjon virke mot bevegelsesretningen med en konstant størrelse. Hvis bevegelsesretningen endres, så denne friksjonskraften umiddelbart motvirke bevegelsene i den nye retningen med samme størrelse, noe som fører til en diskontinuitet.

(d) Dødsoner/dødbånd: Statisk friksjon i f.eks. el-motorer kan resultere i at inngangssignalet er for svakt til å produsere en merkbar respons, noe som betyr at motoren ikke vil reagere eller starte bevegelsen før inngangssignalet overstiger et bestemt nivå. Dette fører dermed til en (død-)sone/bånd hvor motoren ikke reagerer.

Merk at dette er vanlige fenomener, som ofte opptrer i reguleringssystemer og som av den grunn er viktige å ta hensyn til. Vi skal i disse notatene dog kun fokusere på metning, og mer spesifikt på et fenomen dette kan føre til hvis man også har et I-ledd i regulatoren, såkalt «wind-up».

Fun facts, bemerkninger og annet dill dall (you may skip)

Uliniariteter kan også være hensiktsmessige, og noen ganger vil man kunstig «tilføre» uliniariteter i et reguleringssystem ved hjelp av en ulineære regulator.

For eksempel, så finnes det en mengde fenomener som opptrer kun i ulineære systemer og som en linearisert modell dermed ikke vil kunne oppvise. Det kan også i visse tilfeller være ønskelig å designe regulatoren til å være en ulineær funksjon av tilstandene i systemet. Dette kan gjøres på grunn av stabilitetsårsaker, på grunn av et ønske om høyere ytelse (raskere og/eller mer nøyaktig respons), eller for å «fremprovosere» et ulineært fenomen. I mange tilfeller vil også en analyse av det ulineære systemet være enklere enn å først linearisere systemet og så analysere den lineære modellen.

7.1. Linearisering

Du vil lære mer om dette i Du vil lære mer om dette i både IELET2002 - Reguleringsteknikk og IELET2101 - Reguleringsteknikk 2.

Hvorfor skal du lære dette? 🤷‍♀️ Målet med linearisering er å ta et ulineært system og lage et nytt, kunstig lineært system som fungerer som en god tilnærming av det ulineære systemet nært et ønsket arbeidspunkt. Dette gjøre at vi kan ta i bruk det store arsenale av metoder for analyse og regulatordesign for lineære systemer. Dette vil som oftest kun fungere nært det ønskede arbeidspunktet hvor vi lineariserte, men dette er ofte godt nok.

Gitt et ulineært systemt med én tilstand, $x \in \mathbb{R}$, på følgende form:

$$\dot{x} = f(x). \tag{7.1}$$

Anta at

- $f'(x) = \frac{d}{dx} f(x) = \frac{df}{dx}(x)$ er en kontinuerlig funksjon;

- a er et **likevektspunkt** til systemet: $f(a) = 0$.

Det **lineariserte systemet** til (7.1) om punktet a er da

$$\frac{d}{dt}\delta x = A \cdot \delta x \tag{7.2}$$

hvor

- $A = f'(a) = \frac{df}{dx}\big|_{x=a}$ (notasjonen $|_{x=a}$ betyr at vi setter $x = a$ *etter* vi har derivert funksjonen med hensyn på variabelen x);
- δx er «tilstanden» til det lineariserte systemet, hvor man kan tenke at $\delta x \approx x - a$;
- $\frac{d}{dt}\delta x = \frac{d\delta x}{dt}$ betegner tidsderivatet til δx, siden $\dot{\delta x}$ ser litt rart ut.

Eksempel 7.1. Systemet $\dot{x} = f(x) = \sin(x)$ har likevektspunkter som alle er multiplum av π, f.eks. 0, $-\pi$ og π. La oss linearisere om punktet $x_\star = a = 0$:

$$A = \frac{\partial f}{\partial x}\Big|_{x=0} = \cos(0) = 1$$

Dermed er det lineariserte systemet $\frac{d}{dt}\delta x = \delta x$.

Noe som gjør linearisering svært nyttig, er at vi kan finne ut av visse stabilitetskarakteristikker til et likevektspunkt for det ulineære systemet ved å studere stabiliteten til det lineariserte systemet om dette punktet:

Bestemme stabiliteten til et ulineært system fra linariseringen: Hvis nullpunktet til det lineariserte systemet (7.2) er asymptotisk stabilt (hhv. ustabilt), altså $\delta x \to 0$ (hhv. $|\delta x| \to \infty$) når $t \to \infty$, så er også tilsvarende likevektspunkt for det ulineære systemet (7.1) eksponentielt stabilt (hhv. ustabilt) nært likevektspunktet.

⚠️ **Pass på!** Det motsatte er generelt sett ikke sant – altså, det ulineære systemet kan være stabilt selv om det lineariserte systemet ikke er det. Det er heller ikke nødvendigvis sant for systemer med tidsforsinkelser, eller marginalt stabile systemer.

Oppgave 7.1. Følgende system har ett positivt likevektspunkt: $\dot{x} = (x+1)^2 - 4$. Linariser systemet om dette likevektspunktet. Er dette punktet asymptotisk/eksponentielt stabilt for det ulineære systemet?

Hva er motivasjonen bak linariseringsprosedyren? 🤔

Prosedyren baserer seg på det faktum vi så i seksjon 2.1.4, nemlig at den deriverte til funksjonen f evaluerte ved a, altså $f'(a)$, tilsvarer helningen til tangentlinjen til f ved a. Dette gjør at $f(x) \approx f(a) + f'(a)(x - a)$ for x nært a.

Følgende eksempel fra Wikipedia viser på en fin måte litt hvordan linearisering fungerer:

Eksempel 7.2. Gitt funksjonen $f(x) = \sqrt{x}$, så er det jo lett å se at $f(4) = \sqrt{4} = 2$; men hva er $f(4.001)$? Vi kan anta at nærme $a = 4$, så har vi $f(x) \approx f(4) + f'(4)(x-4) = 2 + \frac{1}{4}(x-4)$, slik at $f(4.001) \approx 4 + 0.001/4 = 2.00025$, som jo er temmelig nærme den reelle verdien $f(4.001) = 2.00024998$.

Fun facts, bemerkninger og annet dill dall (you may skip)

Mer spesifikt, så baserer motivasjonen for linearisering om et likevektspunkt seg på Taylors teorem, som sier at, hvis f er kontinuerlig differensierbar ($f'(x)$ er en kontinuerlig funksjon), så har vi for alle punkter a at

$$f(x) = f(a) + f'(a)\Delta x + \gamma(x)\Delta x, \quad \lim_{x \to a} \gamma(x) = 0,$$

hvor $\Delta x = x - a$. Med andre ord, siden $\gamma(x)\Delta x$ normalt sett vil være mye mindre enn $f'(a)\Delta x$ når Δx er liten (unntaket er hvis $f'(a) = 0$, uten at det ødelegger metoden på noen måte), så er $f'(a)\Delta x$ en god approksimasjon av $f(x)$ nært et likevektpunkt a. [a]

⚠ **Achtung!** Man ser ofte at det lineariserte systemet er skrevet som $\frac{d}{dt}\Delta x = A\Delta x$ hvor Δx er definert som $\Delta x = x - a$. Dette stemmer dog bare når $f(x) - f(a)$ allerede er en lineær funksjon. Vi bruker derfor δx i stedet for Δx som «tilstanden» i det lineariserte systemet for å tydelig indikere at dette generelt sett er et fiktivt system.

[a] Hvis du virkelig vil forstå det matematiske maskineriet som lar oss ta i bruk linearisering for reguleringstekniskeformål, så kan du f.eks. ta en titt på Hartman–Grobman teoremet.

Hva med pådrag? La nå systemet i stedet være på formen

$$\dot{x} = f(x, u)$$

altså med én tilstand, $x \in \mathbb{R}$, og ett pådrag, $u \in \mathbb{R}$. Anta at

- $\frac{\partial f(x,u)}{\partial x}$ og $\frac{\partial f(x,u)}{\partial u}$ er kontinuerlige funksjoner;
- $(x_\star, u_\star)$ er et **arbeidspunkt/tvunget likevektspunkt**: $f(x_\star, u_\star) = 0$.

Det lineariserte systemet om $(x_\star, u_\star)$ er

$$\frac{d}{dt}\delta x = A \cdot \delta x + B \cdot \delta u$$

hvor

- $A = \frac{\partial f}{\partial x}\Big|_a$ og $B = \frac{\partial f}{\partial u}\Big|_a$ med notasjonen $\Big|_a = \Big|_{\substack{x=x_\star \\ u=u_\star}}$;
- man intuitivt kan tenke at $\delta x \approx x - x_\star$ og $\delta u \approx u - u_\star$.

Eksempel 7.3. Systemet $\dot{x} = f(x,u) = \sin(x) + \cos(x)u$ har for $u = 0$ likevektspunkter som alle er multiplum av π, f.eks. 0, $-\pi$ og π. La oss linearisere om punktet $(x_\star, u_\star) = (0,0)$:

$$A = \frac{\partial f}{\partial x}\Big|_{\substack{x=0\\u=0}} = \cos(0) - \sin(0)\cdot 0 = 1 \quad \text{og} \quad B = \frac{\partial f}{\partial u}\Big|_{\substack{x=0\\u=0}} = \cos(0) = 1$$

Dermed er det lineariserte systemet $\frac{d}{dt}\delta x = \delta x + \delta u$.

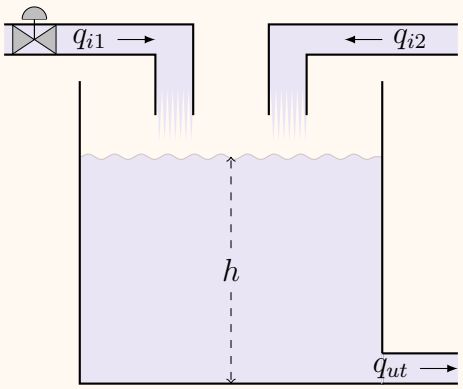
Et mer virkelighetsnært eksempel er følgende:

Eksempel 7.4. (**Linearisering av tankmodell med forstyrrelse**)

Vi ønsker nå å linearisere følgende prosess om arbeidspunktet $(y_\star, u_\star, v_\star) = (1, a/b, 0)$:

$$\dot{y} = -a\sqrt{y} + b(u+v).$$

Prosessen tilsvarer tanksystemet i figuren til høyre (se eks. 6.3) med



- Utgang/tilstand: $y = x = h$;
- Referanse/settpunkt: $r = h_\emptyset$;
- Pådrag: $u = q_{i1}$;
- Forstyrrelse: $v = q_{i2}$.

Den eneste ulineariteten er leddet $-a\sqrt{y}$, hvor vi har at $\frac{d}{dy}\left(a\sqrt{y}\right) = a\frac{1}{2\sqrt{y}}$, som ved $y = y_\star = 1$ tilsvarer $a/2$. Det lineariserte systemet er dermed

$$\frac{d}{dt}\delta y = -\frac{a}{2}\cdot\delta y + b\cdot\delta u + b\cdot\delta v$$

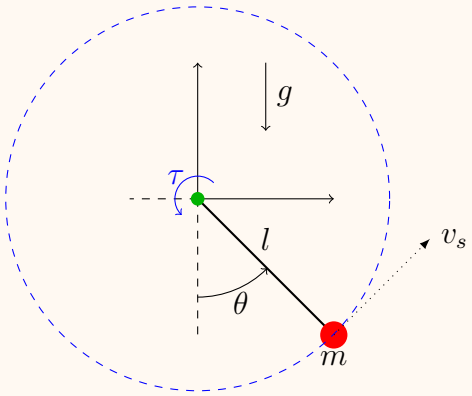
hvor man kan tenke at $\delta y \approx y - 1$, $\delta u \approx u - a/b$ og $\delta v \approx v$ nært arbeidspunktet.

Man kan også linearisere systemer med mer enn bare én tilstand:

Eksempel 7.5. Linearisering av bevegelsesligningen til en pendel

I eksempel 2.2 så vi at bevegelsesligningen til en pendel uten demping, slik som i figuren til høyre, kunne skrives som

$$\ddot{\theta} = -\frac{g}{l}\sin(\theta) + \frac{\tau}{ml^2} = -a\sin(\theta) + b\tau.$$



Ved å introdusere tilstandene $x_1 = \theta$ og $x_2 = \dot{\theta}$, samt ta $u = \tau$, kan dette skrives som et ulineært, andre-ordens system på tilstandsromform:

$$\dot{x}_1 = x_2 =: f_1(x_1, x_2, u)$$
$$\dot{x}_2 = a\sin(x_1) + bu =: f_2(x_1, x_2, u).$$

Vårt mål er å linearisere systemet om det ustabile likevektspunktet tilsvarende når pendelen står rett opp, altså $(\theta_\star, \dot{\theta}_\star, u_\star) = (\pi, 0, 0)$. Den eneste ulinearitet her er $a\sin(x_1)$, slik at

$$\frac{\partial f_1}{\partial x_1} = 0, \quad \frac{\partial f_1}{\partial x_2} = 1, \quad \frac{\partial f_1}{\partial u} = 0,$$

mens

$$\frac{\partial f_2}{\partial x_1} = a\cos(x_1), \quad \frac{\partial f_2}{\partial x_2} = 0, \quad \frac{\partial f_2}{\partial u} = b.$$

Siden $\cos(\pi) = -1$, så har vi dermed at det lineariserte systemet om arbeidspunktet er

$$\frac{d}{dt}\delta x_1 = \delta x_2$$
$$\frac{d}{dt}\delta x_2 = -a\delta x_1 + b\delta u$$

hvor $\delta x_1 \approx x_1 - \pi$, $\delta x_2 = x_2$ og $\delta u = u$ nært arbeidspunktet.

7.2. Metning og integrator-wind-up

Du vil lære mer om dette i IELET2101 - Reguleringsteknikk 2 og IELET2102 - Digitale reguleringssystemer.

Alternative kilder: Wikipedia; §12.5 i [Balchen et al., 2016]; [Tarbouriech and Turner, 2009].

Hvorfor skal du lære dette? 🤷 Metning er uunngåelig i alle fysiske pådragsorgan: en motor kan bare generere så og så mye dreiemoment, man kan ikke føre uendelig med strøm gjennom et varmeelement uten at det brenner opp, en greviling kan sjelden trekke stort mer en 7 poser med kantareller, og en reguleringsventil kan bare være et sted mellom helt lukket og helt åpen.

Det kan være svært problematisk hvis man ikke tar hensyn til at pådraget i et reguleringssystem kan gå i metning. Dette er spesielt tilfelle hvis man har et (dynamisk) I-ledd i regulatoren, siden et fenomen kalt «integrator-wind-up» da kan oppstå; dette kan i beste fall kun redusere ytelsen til systemet, og i verste fall kan wind-up føre til ustabilitet.

7.2.1 Hva er metning?

Metning i forhold til et pådragsorgan betyr bare at det har en øvre grense (ventil helt åpen) og en nedre grense (ventil helt lukket) i forhold til dets påvirkningsgrad på et system. På engelsk er metning «saturation», slik at sat(·) som oftest blir brukt til å betegne metningsfunksjonen:

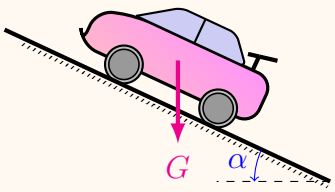
$$\operatorname{sat}_{\underline{u}}^{\overline{u}}(u) = \min\left(\overline{u}, \max(u, \underline{u})\right) = \begin{cases} \overline{u} & \text{når} \quad u \geq \overline{u} \\ u & \text{når} \quad \underline{u} \leq u \leq \overline{u} \\ \underline{u} & \text{når} \quad u \leq \underline{u} \end{cases} \qquad \text{(Metningsfunksjonen)}$$

som bare sier at utgangen (det reelle pådraget) kan være et sted mellom den nedre grensen, $\underline{u}$, og den øvre, $\overline{u}$. For enkelhetsskyld vil vi som regel bare skrive $\operatorname{sat}(u)$, altså droppe metningsverdiene $\overline{u}$ og $\underline{u}$.

Metning setter naturlig nok begrensninger å ytelsen til et reguleringssystem; dette gjelder både hvor godt det kan følge referanser og hvor effektivt man kan kompensere for forstyrrelser. Som nevnt tidligere, så er det dog spesielt metning i kombinasjon med en regulator som har dynamikk (f.eks. I-leddet i en PI-regulator) som kan være problematisk pga. fenomenet «windup» (evt. «overladning» eller «opptvinning» på norsk). La oss ta et illustrativt eksempel:

Eksempel 7.6. (Tafatt bil med cruise-kontroll som skal over en ås)

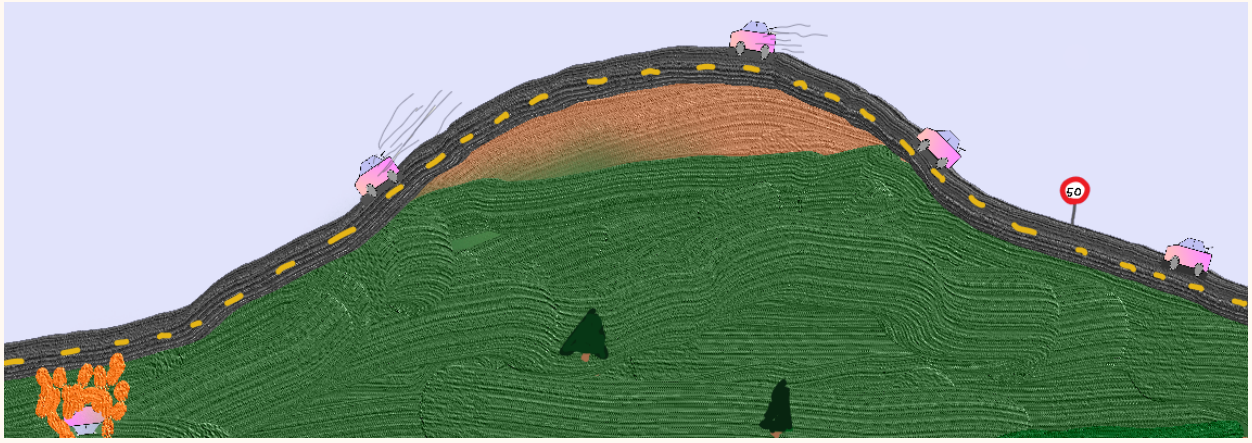
En meget snasen bil skal kjøre over en ås. Bilen har implementert cruise-kontroll ved hjelp av en PI-regulator. Dessverre står ikke bilens motor i stil med dens lekre design. Dette fører til at den ikke klarer å opprettholde den ønskede hastigheten på 50 km/t i de bratteste delene av bakken (se også oppgave 6.2 for hva som kreves av dreiemoment på hjulene her).



Konsekvensene av dette er som følger:

1. Bilens motor klarer ikke å gi nødvendig dreiemoment, den går i metning;
2. Et (stasjonært) avvik oppstår;
3. Avviket vil få regulatorens integral-ledd til å øke tross i at metningsgrensen er nådd;

4. I-leddet vil fortsette å øke til bakken flater ut og man når settpunktet;
5. Man vil måtte ligge en stund over settpunktet for at I-leddet skal "lades/tvinnes ut";
6. Kraftig oversving er mulig, spesielt hvis nedoverbakken begynner før full nedlading;
7. Dette kan potensielt ha katastrofale følger:



7.2.2 Hva er windup?

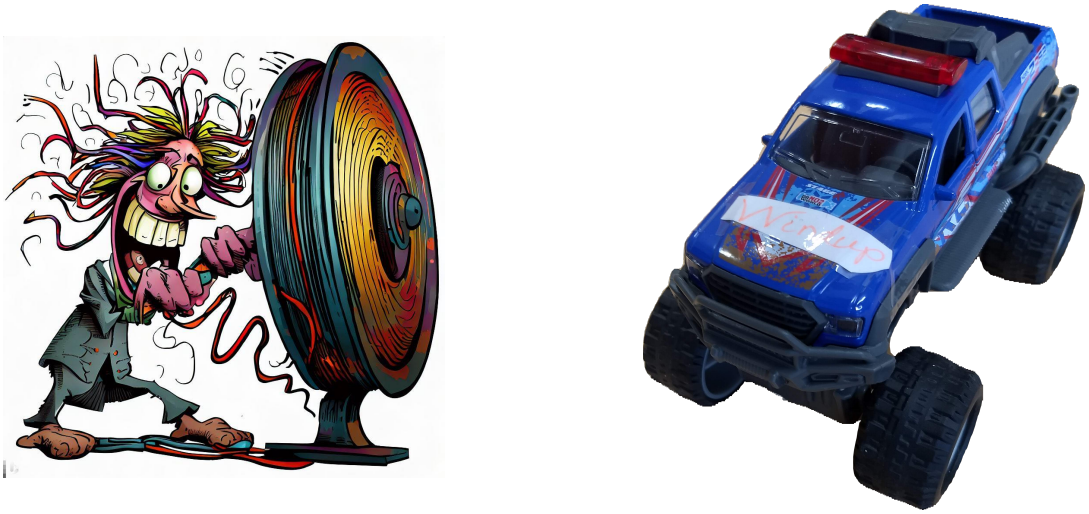
«Wnd-up» (evt. «overlading» eller «opptvinning») kan oppstå gitt følgende «ingredienser»:

1. En regulator med et integral-ledd (eller mer generelt, en dynamisk regulator av noe slag);
2. Pådragsorgan (aktuator) med ratebegrensninger og/eller som kan gå i metning.

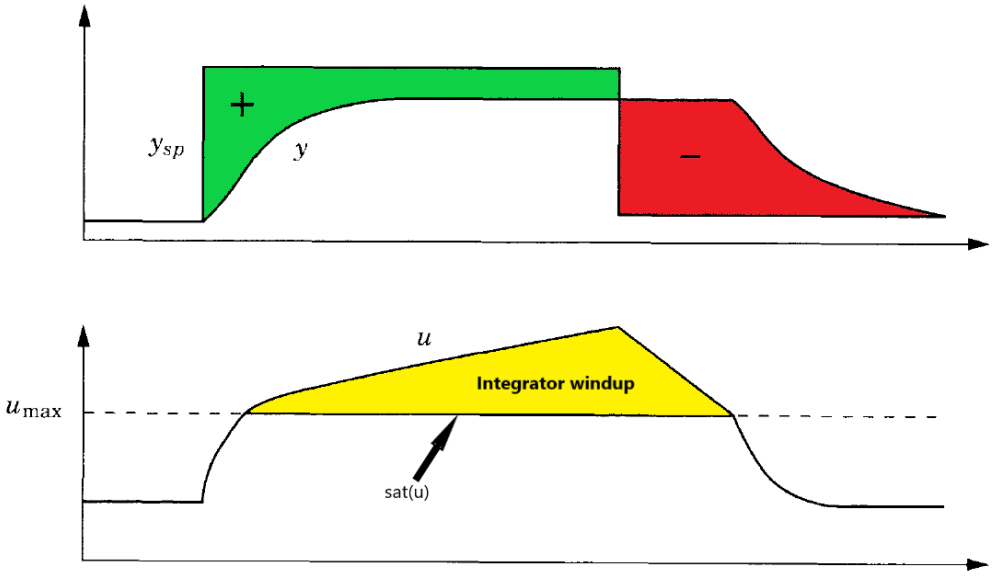
Integral(I)-leddet i en regulator kan man tenke på som en fjær eller spole (evt. trekk-opp-bil) som kan lades eller trekkes opp eller ned (se fig. 7.2). Når man da bruker en aktuator med metnings- eller ratebegrensninger (en ventil kan f.eks. bare være et sted mellom helt åpen og helt lukket), så vil integrator-leddet kunne lades/tvinnes opp (opptvinning=windup) selv om pådragsorganet har gått i metning, slik at det *ønskede* pådraget fortsetter å øke i magnitude uten at dette har noen effekt på systemet. Når man så treffer ønsket settpunkt (hvis man er så heldig) eller det endres, da må det «overflødige» integral-leddet igjen tømmes/lades ut, noe som kan ta tid. Mulige **konsekvenser** av dette er: overskyting av settpunkt, treg respons (forsinkelser og lange transienter), og kanskje til og med ustabilitet.

En illustrasjon av dette fenomenet er vist i figur 7.3. Der ser man at systemet går i metning før man klarer å nå ønsket settpunkt, y_{sp}. Integral-leddet forsetter dog uansett å øke (se **gul** region) på grunn av de vedvarende avviket (se **grønn** region)-. Når man så enderer endrer settpunktet, så tar det lang tid før regulatoren reagerer (se den **røde** regionen).[1]

[1] Det finnes et ferdig eksempel Simulink som illustrerer dette fenomenet. Du finner det her eller ved å skrive følgende i kommandovinduet i MATLAB:
`openExample('simulink_industrial/AntiWindupControlUsingAPIDControllerExample')`



Figur 7.2: Et I-ledd i en PID-regulator fungerer litt som en spole eller en trekk-opp-bil: Når avviket er positivt, «tvinnes» dette leddet opp (det øker), mens det tømmes eller lades ut når avviket er negativt. Når I-leddet allerede er «tvunnet opp» tar det tid å tømme det igjen, noe som kan føre til fenomenet «wind-up» hvis man har metning. Figuren til venstre er generert av DALL-E, mens figuren til høyre viser trekk-opp-bilen «Windup».



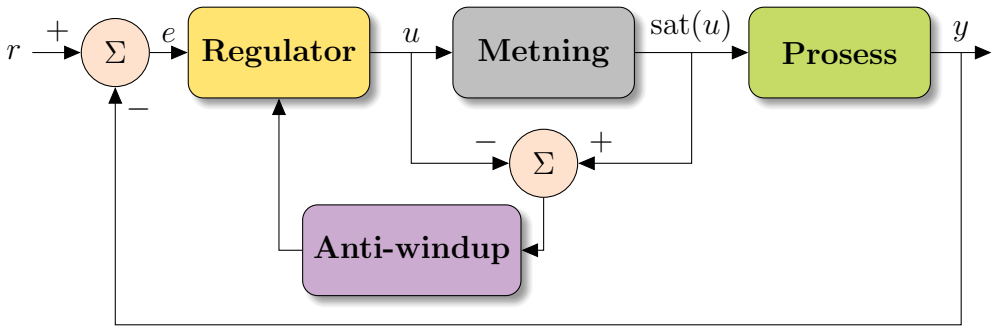
Figur 7.3: Illustrasjon av fenomenet windup; figur fra [Åström and Hägglund, 2006].

7.2.3 Anti-windup-metoder

En god måte å unngå wind-up som oppstår på grunn av sprang i referansen, er å implementere referanse-glatting og -følging (se § 9.1). Er dette gjort på en god måte, så vil pådraget aldri gå i metning eller nå sine ratebegrensninger. På den annen side vil ikke dette ha noen effekt på wind-up som oppstår pga. av en forstyrrelse eller et stasjonært avvik vi ikke kan gjøre noe med. For systemer med metning[2] som er utsatt for store forstyrrelser, og som bruker integralvirking

[2]Som nevnt kan wind-up også oppstå pga. ratebegrensninger, men dette er et mer sjeldent problem enn problemet fra metninger, samt mer krevende å håndtere, så vi skal ikke se på hvordan man håndterer det.

i regulatoren, bør man derfor bruke en **anti-windup metode**/-kompensator.

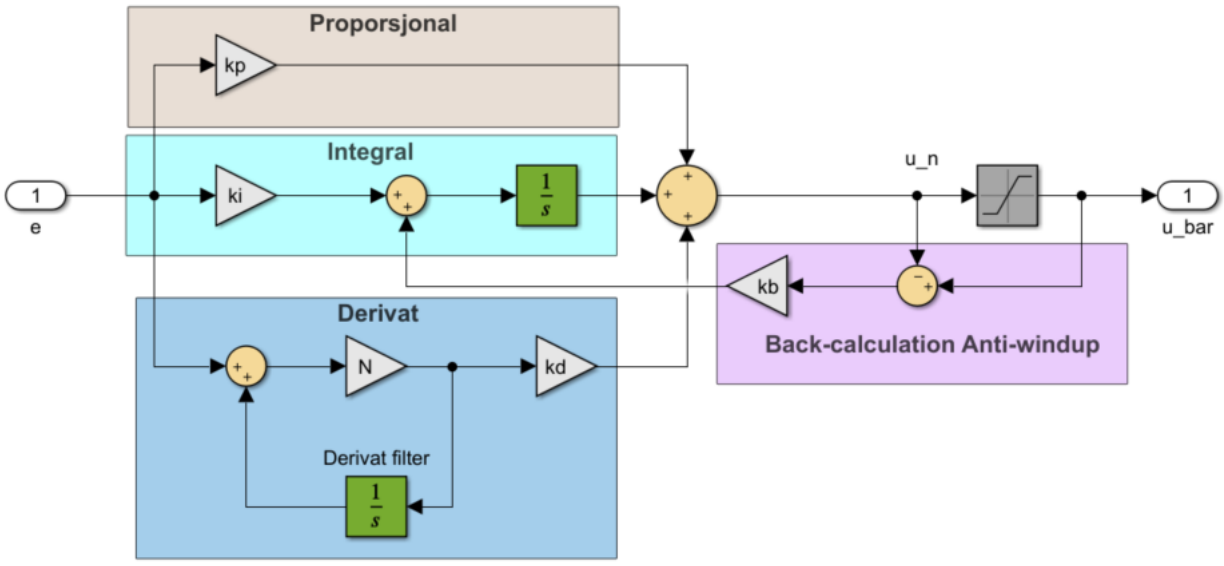


Figur 7.4: Vanlig arkitektur for anti-windup-metoder.

En typisk anti-windup-struktur er vist i figur 7.4, hvor man ser på differansen til pådraget før og etter en metnings-blokk. Hvis differansen ikke er null, så påvirker man regulatoren på en eller annen måte. Selv om det finnes flere anti-windup-metoder, så skal vi kun se på en slik metode, kalt «back-calculation» (grovt oversatt til tilbakekalkulering):

Merk: som oftest måler man ikke direkte om pådragsorganet har gått i metning eller ikke, slik at den grå-blokken i figur 7.4 som oftest er en «fiktiv» metning implementert i koden til regulatoren man har laget.

Back calculation



Figur 7.5: Simulink-diagram av Back-calculation anti-windup.

En Simulink-implementasjon av anti-windup-metoden *back-calculation* er vist i figur 7.5. Back calculation tilsvarer derfor at anti-windup-blokken (den lilla) i figur 7.4 er lik $-k_b \int_0^t (\text{sat}(u)(\tau) - u(\tau))d\tau$. I-leddet vil derfor begynne å bevege seg «motsatt vei» når pådraget har gått i metning, med hastighet gitt av størrelsen på k_b (større = raskere).

Man skulle kanskje tro at man burde ta k_b så stor som mulig, men har man et derivat-ledd skal man være forsiktig med å ta k_b for stor. Grunnen er at derivat-leddet kan føre til raske, kort-levde økninger i pådraget som fører det i metning, noe som da kan "nulle-ut" I-leddet.

Du kan teste back-calculation i dette Simulink-eksempelet.

Relevant oppgave

Del IV

Robotikk og Servoteknikk

8. Robotikk og Kinematikk

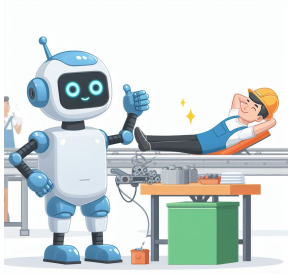
Alternative kilder: Springer handbook of robotics [Siciliano et al., 2008].

Du vil lære mer om dette i IELET2107 - Robotikk

Det finnes i dag mange typer roboter: robotmanipulatorer, forskjellige former for autonome kjøretøy med kule akronymer (ROV, AUV, UAV, AGV osv.), og mekaniske maskiner som imiterer eller er inspirert av biologi (to eller flerbente roboter, slanger, fisker etc.). Mer hverdagslige ting som vaskemaskiner og robotstøvsugere er også som roboter å regne.

Fun facts, bemerkninger og annet dill dall (you may skip)

Ordet robot kommer av det tsjekkiske ordet *robota*, som betyr arbeid, nærmere bestemt kjedelig og hardt arbeid (ordet *robotnik* betyr en eiendomsløs person som utfører slikt arbeid). Ordet ble introdusert av Karel Čapeks i skuespillet «Rossum's Universal Robots» i 1920. Stykket handler om hvordan en vitenskapsmann lager antropomorfiske (menneskelignende) skapninger som etter hvert gjør opprør mot sin skaper.



Fagfeltet robotikk er massivt, og dekker alt fra mekanisk og elektronisk design, instrumentering, matematisk modellering og analyse, sensorikk og persepsjon, regulatordesign, baneplanlegging og -generering, samt programmering, bare for å nevne noen. Robotikkhåndboken til Springer [Siciliano et al., 2008] (som du har gratis tilgang til her via NTNUs nettverk) har for eksempel over 2500 sider.

Vårt mål i disse notatene er å se hvordan vi kan lage enkle *baner* for typiske robotmanipulaoterer man ser i industrien, såkalte *industriroboter*. En bane er grovt forklart bare en (tids-) representasjon som sier hvordan roboten skal bevege seg for å oppnå en ønsket oppgave, for eksempel hvordan få en sveiserobot til å legge sveis der vi ønsker. For å få til dette trenger vi å forstå hvordan verdien til robotens *ledd* (de vi kan styre) tilsvarer hvor sveiseverktøyet er og i hvilken retning det peker. Dette kaller vi for et *kinematikkproblem*, og er det vi skal fokusere på i dette kapittelet. I neste kapittel skal vi se på hvordan vi kan planlegge baner som kobler flere ønskede punkter ved hjelp av interpolering. Vi starter dog først med å gå igjennom oppbyggingen til en typisk industrirobot, samt introduserer noen standard begreper.

8.1. Oppbygningen til en industrirobot

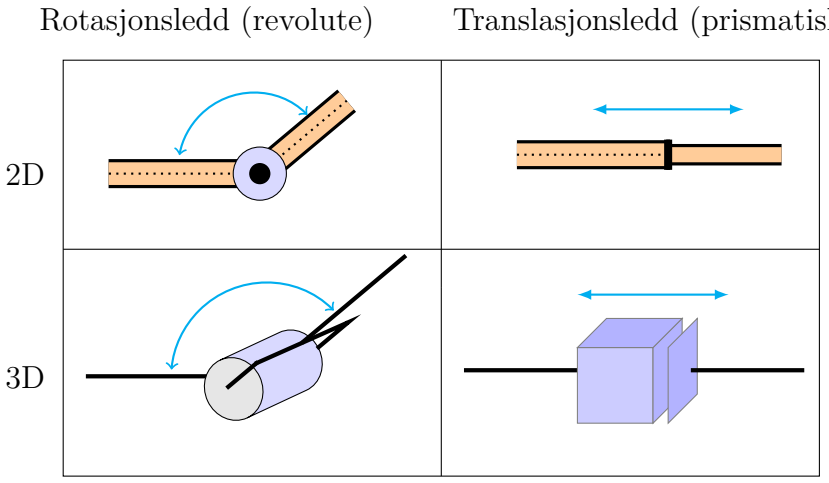
Alternative kilder: SNL; Wikipedia.

Hva er egentlig en industrirobot? 🤔 I henhold til ISO 8373 er en *industrirobot* «en automatisk kontrollert, omprogrammerbar, flerbruksmanipulator programmerbar i tre eller flere akser, som kan være enten fastmontert eller mobil for bruk i industrielle automatiseringsapplikasjoner». I grove trekk, så er derfor en industrirobot en maskin som man kan styre bevegelsene til ved hjelp av en programmerbar datamaskin.

(a) ABB IRB1600 6-akse industrirobot.

(b) Sepro Strong 3-akse (kartesisk) industrirobot.

Figur 8.1: Eksempler på to vanlige typer industriroboter. Figurer fra ABB og Sepro.



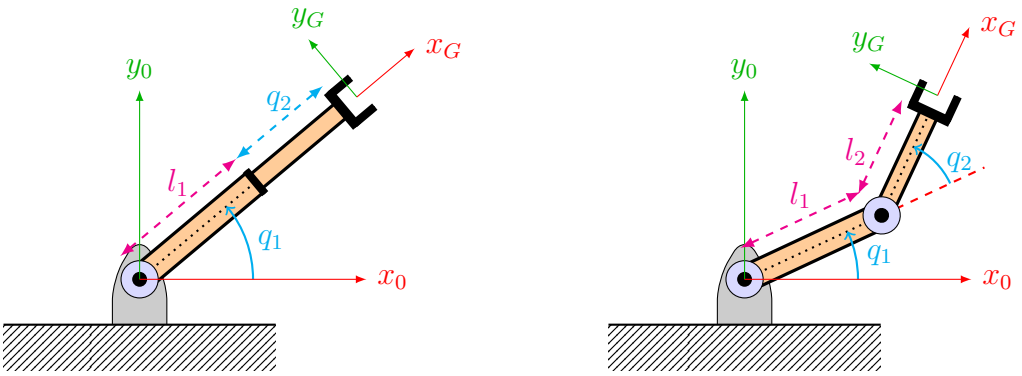
Figur 8.2: Symbolsk representasjon av rotasjons- og translasjonsledd, både 2D og 3D variantene. Figur inspirert av [Spong et al., 2020].

Akser, lenker, leddtyper og frihetsgrader

Eksempler på to vanlige typer industriroboter er vist i figur 8.1. Dette er eksempler på såkalte *seriemanipulatorer*, bestående av *ledd* som kan styres (aktueres) festet sammen via *lenker* av en konstant lengde. Disse robotene (manipulatorene) er seriemanipulatorer siden leddene er koblet etter hverandre (i serie), noe som lager det man kaller for en *åpen kinematisk kjede* (såkalte parallellmanipulatorer lager en *lukket kinematisk kjede*).

ABB roboten i figur 8.1 a) har 6 (aktuerings-)akser, ett for hvert ledd, og like mange *frihetsgrader*, samt består bare av *rotasjonsledd*. Sepro roboten, derimot, har 3 akser, alle tilsvarende et *translasjonsledd*.

Rotasjonsledd, også kalt *revolute ledd*, kan rotere om en gitt akse, mens translasjonsledd, også kalt *prismatiske ledd*, kan utføre en translasjonsbevegelse langs en gitt akse. Grafiske representasjoner av slike typer ledd, både i 2D og 3D, er vist i figur 8.2.



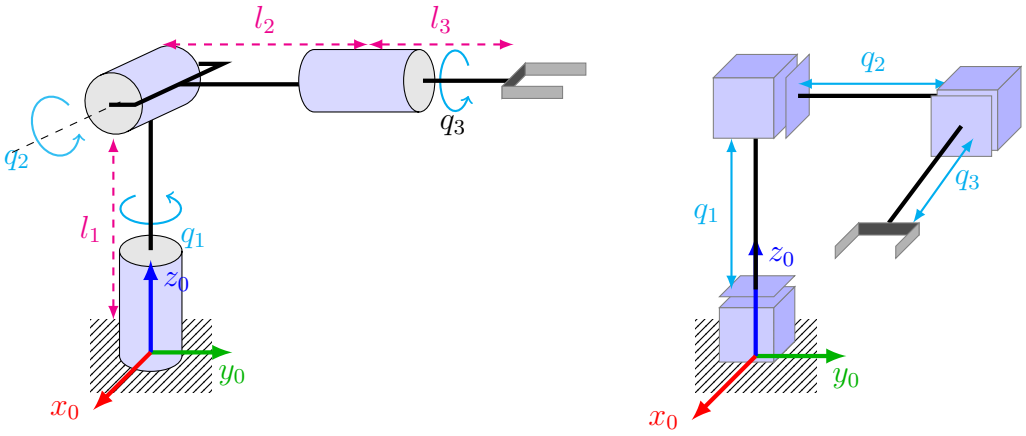
a) RP-manipulator **b) RR-manipulator**

Figur 8.3: Illustrasjon av 2-ledd manipulatorer: a) En manipulator med roterende førsteledd og et translasjonsledd som andreledd (revolut-prismatisk (RP)); b) manipulator med to rotasjonsledd (revolut-revolut (RR)).

Frihetsgradene (eng. degrees of freedom) til en robot sier noe om de ulike bevegelser roboten kan utføre. For et generelt system er antall frihetsgrader hvor mange skalare variabler som er både tilstrekkelig og nødvendig for å unikt beskrive posisjonen og orienteringen til alle komponenter i systemet. For vår del, vil antall frihetsgrader samsvare med antall ledd, altså summen av alle rotasjons- og translasjonsledd. For eksempel, så har 2D (plan) roboten i figur 8.3 a) og b) begge to frihets grader siden de har to ledd (hhv. 1 x revolut + 1 x prismatisk (RP) og 2 x revolute (RR)), mens robotene i figur 8.4 begge har 3 ledd og dermed 3 frihetsgrader.

Verktøy og ende-effektor

Til enden av en robotmanipulator er det som regel festet en eller annen form for vektøy, slik som en mekanisk eller pneumatisk griper, et sveiseverktøy eller et kamera (se fig. 8.5). For å ikke måte spesifisere hvilket verktøy som er festet til robot til en hvert til, vil vi bruke det generelle navnet *ende-effektor* (eng. end effector) til å beskrive det som er feste på enden av roboten. Når det ikke er festet noen verktøy til enden av robotarmen vil ende-effektoren bare samsvare med endepunktet til robotarmen. Koordinatrammen som festes til ende-effektoren kalles enten for ende-effektor-rammen eller bare for verktøy-rammen (eng. tool-frame).



a) RRR-manipulator **b) PPP-manipulator**

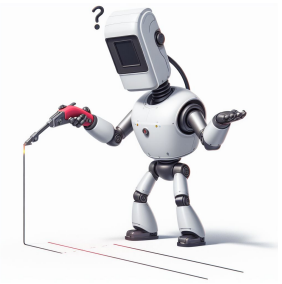
Figur 8.4: Illustrasjon av 3-ledd manipulatorer: a) En manipulator med 3 roterende ledd (derav en RRR-manipulator), ofte kalt et sfærisk håndledd; b) manipulator med tre translasjonsledd (3 x prismatiske, så en PPP-manipulaotr), ofte kalt en kartesisk robot siden den kan bevege seg i kartesiske x-y-z-rommet.

(a) Griper. (b) Vakumgriper. (c) Sveiseverktøy. (d) Freseverktøy

Figur 8.5: Eksempler på forskjellige ende-effektorer/verktøy. Bilder fra hhv. RobotIQ, Universal Robotics, Miller Welds og RONCHINI.

8.2. Kinematikk

Hvorfor skal du lære dette? 🤷‍♀️ Anta at du vil få en robot til å legge en gitt sveis. I enden av robotmanipulatoren er det festet et sveiseverktøy (dets end-effektor). Vi ønsker dermed å styre robotens bevegelse slik at den legger en sveis der vi ønsker. Men siden vi kun kan påvirke robotens leddvariabler, må vi vite hvordan verdien til disse leddvariablene tilsvarer sveiseverktøyets posisjon og orientering. Dette er et kinematikkproblem.



Kinematikk (kan uttales som både ki-nematikk eller skinn-ematikk, hvorav jeg foretrekker den første varianten) stammer fra det greske ordet *kinema* som betyr «bevegelse».

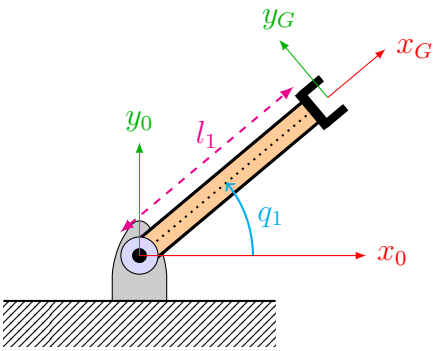
Vi skal her se på foroverkinematikk (også kalt direkte kinematikk) og inverskinematikk. Her ønsker vi å finne forhold mellom en robots leddvariabler og end-effektorens posisjonen og orienteringen. Hastighetskinematikk, hvor man ser på relasjonen mellom hastigheten til end-effektoren og hastigheten til leddvariablene for gitte konfigurasjoner, er også et viktig tema,

men er ikke noe vi vil se på i disse notatene.

8.2.1 Foroverkinematikk

Foroverkinematikkproblemet er som følger:

Foroverkinematikkproblemet: Gitt leddvariablene til en robot, beregn posisjon og orientering til griperen/ende-effektoren i forhold til en gitt koordinatramme.



Figur 8.6: Illustrasjon av en 1-link manipulator med ett rotasjonsledd.

Eksempel 8.1. (Foroverkinematikk for 1-ledd manipulator) Et av de enkleste eksemplene vi kan se for oss er en såkalt 1-link manipulator i planet med ett rotasjonsledd. En slik manipulator er vist i figur 8.6.

Problemet vi er ute etter løse er som følger: finne posisjonen til griperen målt i x_0y_0-rammen, noe som tilsvarer å finne posisjonen til x_Gy_G-rammen sitt nullpunkt (origo).

Enkel trigonometri gir oss posisjonen til griperens senterpunkt:

$$x_G = l_1 \cos(q_1)$$
$$y_G = l_1 \sin(q_1).$$

Vi ser også at griperen peker i samme retning som leddvinkelen q_1, som dermed angir griperens orientering.

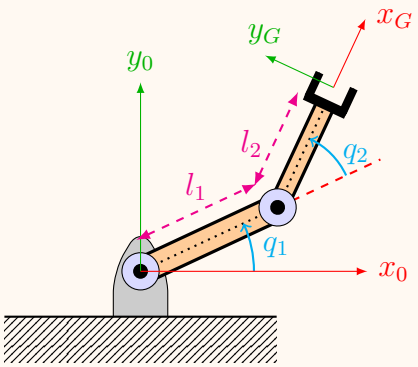
Dette kan ganske lett utvides til en 2-link manipulator i planet som er vist i b) figur 8.3:

Eksempel 8.2. (Foroverkinematikk for 2-ledd RR-manipulator i planet)

Gitt manipulatoren i figuren til høyre med to revolutte ledd. Her vil koordinatene til griperen være gitt av

$$x_G = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2)$$
$$y_G = l_1 \sin(q_1) + l_2 \sin(q_1 + q_2)$$

og retningen til griperen er gitt av summen av leddvinklene $q_1 + q_2$.



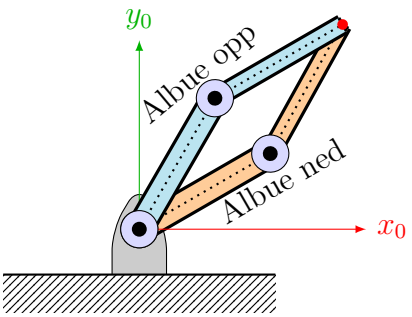
Foroverkinematikk kan også formuleres for roboter med prismatiske ledd, eller kombinasjoner av rotasjonsledd og rotasjonsledd. Uansett vil det være snakk om bruk av geometri og trigonometri for å finne posisjon og orientering til griperen som funksjon av leddvinkelene.

8.2.2 Inverskinematikk

Inverskinematikken er, som navnet hinter til. det motsatte av foroverkinematikkproblemet:

Inverskinematikkproblemet: Gitt ønsket posisjonen og orienteringen til griperen/end-effekotren, finn tilsvarende leddvariabler.

Dette er matematisk sett et mye vanskeligere problem. Basert på geometrier og kompleksiteten til roboten, trenger det heller ikke eksisterer en analytisk løsning. Dessuten er det slik at selv om en løsning eksisterer, er det ikke sikkert at den er unik, det kan eksistere mange løsninger. For eksempel viser figur 8.7 to mulige konfigurasjonser for RR-manipulatoren som oppnår samme posisjon til enden.



Figur 8.7: Illustrasjons av at inverskinematikkproblemet ikke nødvendigvis har en unik løsning: Enden til RR-manipulatoren, som ofte kalles for en *skulder-albue-manipulator*, kan ha samme posisjon for to forskjellige konfigurasjoner av leddvariablene, såkalt «albue opp» og «albue ned».

La oss starte med et av de enkleste 1D eksemplene før vi beveger oss videre til 2D:

Eksempel 8.3. (Inverskinematikk for 1-ledd revolutt manipulator) For 1-link manipulatoren vist i figur 8.6 er inverskinematikkproblemet slik: Gitt (x_G, y_G), finn q_1. Fra figuren, ser vi at

$$q_1 = \text{Atan}(y_G, x_G) \tag{8.1}$$

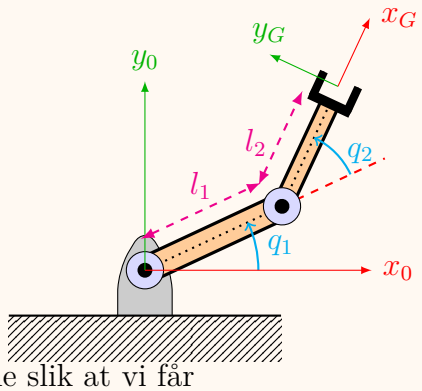
der $\text{Atan}(y, x)$ er en to-arguments invers tangens funksjon; se § A.2.[a]

[a] $\text{Atan}(y, x)$ er definert for alle $(x, y) \neq (0, 0)$ slik at vinkelen alltid kommer i riktig kvadrant. For eksempel så er $\text{Atan}(-1, 1) = \frac{3\pi}{4}$ og $\text{Atan}(1, -1) = -\frac{\pi}{4}$. I Matlab er den tilsvarende funksjonen `atan2(y,x)`.

For 2-link RR-manipulatoren i figur 8.3 vil det vise seg at inverskinematikken blir ganske mye mer komplisert å regne ut:

Eksempel 8.4. (Inverskinematikk for 2-ledd RR-manipulator)

For RR manipulatoren fra eksempel 8.2 vist i figuren til høyre, så ønsker vi nå å løse følgende problem: Gitt en ønsket posisjon til griperen, altså (x_G, y_G), finn q_1 og q_2.
Fra før vet vi jo at



$$x_G = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \qquad (8.2a)$$
$$y_G = l_1 \sin(q_1) + l_2 \sin(q_1 + q_2). \qquad (8.2b)$$

Det vil nå være nyttig å kvadrere disse sammenhengene slik at vi får

$$x_G^2 = l_1^2 \cos^2 q_1 + l_2^2 \cos^2(q_1 + q_2) + 2l_1 l_2 \cos q_1 \cos(q_1 + q_2)$$
$$y_G^2 = l_1^2 \sin^2 q_1 + l_2^2 \sin^2(q_1 + q_2) + 2l_1 l_2 \sin q_1 \sin(q_1 + q_2)$$

Merk at for å forenkle utrykkene i det som følger, så tar vi i bruk følgende notasjon:

$c_1 = \cos q_1$, $s_1 = \sin q_1$, $c_2 = \cos q_2$, $s_2 = \sin q_2$, $c_{12} = \cos(q_1 + q_2)$ og $s_{12} = \sin(q_1 + q_2)$.

Ved å summere x_G^2 og y_G^2 finner vi nå

$$\begin{aligned} x_G^2 + y_G^2 &= l_1^2(c_1^2 + s_1^2) + l_2^2(c_{12}^2 + s_{12}^2) + 2l_1 l_2(c_1(c_1 c_2 - s_1 s_2) + s_1(c_1 s_2 + s_1 c_2)) \\ &= l_1^2 + l_2^2 + 2l_1 l_2(c_1^2 c_2 - c_1 s_1 s_2 + s_1 c_1 s_2 + s_1^2 c_2) \\ &= l_1^2 + l_2^2 + 2l_1 l_2 c_2(c_1^2 + s_1^2) \\ &= l_1^2 + l_2^2 + 2l_1 l_2 \cos q_2 \end{aligned}$$

der vi også har brukt formlene for cos og sin til summen av vinkler (se vedlegg A.2). Vi kan nå finne q_2 fra $x_G^2 + y_G^2 = l_1^2 + l_2^2 + 2l_1 l_2 \cos q_2$ ved sammenhengen

$$\cos q_2 = \frac{x_G^2 + y_G^2 - l_1^2 - l_2^2}{2l_1 l_2} =: \Omega.$$

Det er dog slik at dette utrykket kan bli numerisk unøyaktig for små vinkler, og dessuten vil vi gjerne utnytte fordelen med å automatisk få rett vinkel ved hjelp av Atan-funksjonen. Derfor regner vi også ut

$$\sin q_2 = \pm\sqrt{1 - \cos^2 q_2} = \pm\sqrt{1 - \Omega^2}$$

slik at vi kan finne q_2 ved hjelp av

$$q_2 = \text{Atan}(\pm\sqrt{1 - \Omega^2}, \Omega). \qquad (8.3)$$

Legg merke til at de to løsningene i (8.3) svarer til de såkalte albue opp og albue ned løsningene som er illustrert i figur 8.7. For å finne q_1 tar vi igjen utgangspunkt i (8.2) som kan skrives på følgende ekvivalente form:

$$x_G = l_1c_1 + l_2c_1c_2 - l_2s_1s_2$$
$$y_G = l_1s_1 + l_2c_1s_2 + l_2s_1c_2,$$

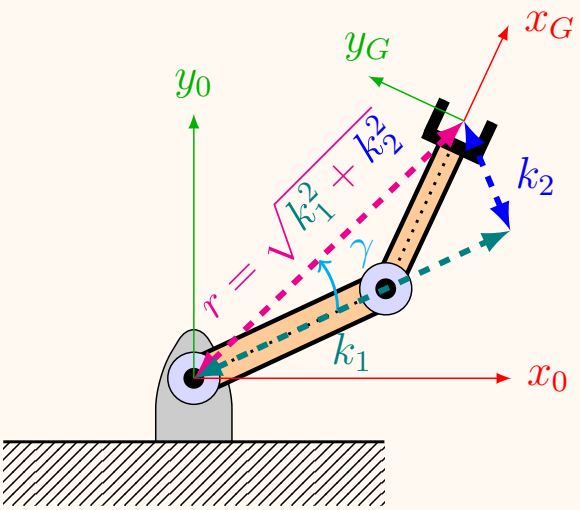
og videre som

$$x_G = k_1 \cos q_1 - k_2 \sin q_1$$
$$y_G = k_1 \sin q_1 + k_2 \cos q_1,$$

der $k_1 = l_1 + l_2c_2$ og $k_2 = l_2s_2$.

Hensikten med å innføre k_1 og k_2 er at disse er begge avhengige av kun q_2 som nå er kjent. Vi innfører nå $r = \sqrt{k_1^2 + k_2^2}$ og vinkelen $\gamma = \text{Atan}(k_2, k_1)$ slik som vist i figuren til høyre. Dette gjør av vi kan skrive $k_1 = r\cos\gamma$ og $k_2 = r\sin\gamma$, slik at uttrykkene for x_G og y_G blir

$$x_G = r\cos\gamma\cos q_1 - r\sin\gamma\sin q_1$$
$$y_G = r\cos\gamma\sin q_1 + r\sin\gamma\cos q_1.$$



Ved å dividere med r får vi

$$\frac{x_G}{r} = \cos\gamma\cos q_1 - \sin\gamma\sin q_1 \quad \text{og} \quad \frac{y_G}{r} = \cos\gamma\sin q_1 + \sin\gamma\cos q_1.$$

Legg her merke til at høyresidene tilsvarer uttrykkene for cos og sin til summer av vinkler, slik at

$$\frac{x_G}{r} = \cos(\gamma + q_1) \quad \text{og} \quad \frac{y_G}{r} = \sin(\gamma + q_1).$$

Fra disse ligningene ser vi igjen at

$$\gamma + q_1 = \text{Atan}\left(\frac{y_G}{r}, \frac{x_G}{r}\right) = \text{Atan}(y_G, x_G)$$

og dermed får

$$q_1 = \text{Atan}(y_G, x_G) - \gamma = \text{Atan}(y_G, x_G) - \text{Atan}(k_2, k_1),$$

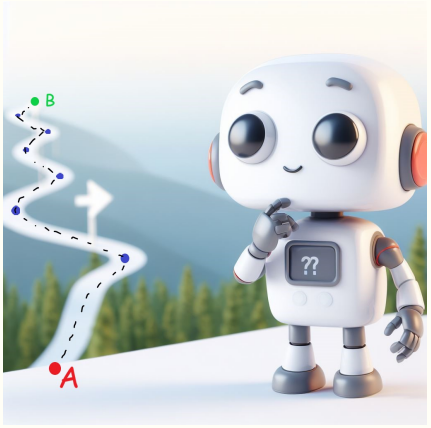
hvorfra vi kan finne q_1 når vi vet q_2 fra $q_2 = \text{Atan}(\pm\sqrt{1-\Omega^2}, \Omega)$. 🥳

Du finner et MATLAB-live skript hvor du kan teste og visualisere resultatene fra dette eksempelet vha. Robotics system toolbox via denne lenken eller denne.

9. Banegenerering og -følging

Du vil lære mer om dette i Du vil lære mer om dette spesielt i IETET2107 - Robotikk, samt i IELET2101 - Reguleringsteknikk 2.

Hvorfor skal du lære dette? 🤷‍♀️ En svært viktig undergruppe av reguleringsteknikken er *servoteknikk*, som blant annet inkluderer posisjons- og hastighetsstyring av motorer. Dette brukes også til *baneføling*, hvor man ønsker at en maskin (ofte en type robot 🤖 eller et autonomt kjøretøy 🚜) skal utføre en ønsket bevegelse. For dette må man naturlig nok vite hvordan denne bevegelsen skal se ut; her kommer *sti-* og *baneplanlegging* inn i bildet. Baneplanlegging er essensielt for å sikre at roboter og autonome kjøretøy beveger seg på en trygg og effektiv måte, hvor de unngår hindringer og er energieffektive.



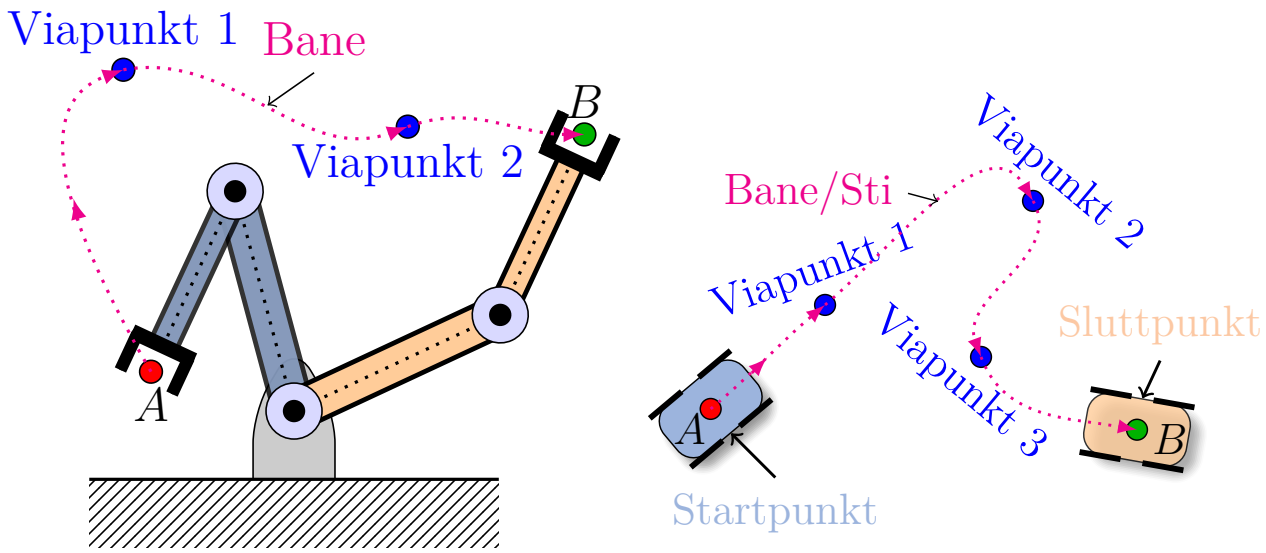
Servoteknikk: I dette kapittelet er vi i hovedsak interessert i å styre kontinuerlig *bevegelser* til mekaniske systemer, fremfor bare sørge for at systemet styrt til et ønsket settpunkt (en konstant referanse). Dette problemet, altså bevegelsesstyring av mekaniske systemer, kalles for *servoteknikk*, noe som for eksempel inkluderer reguleringen av hastigheten og/eller posisjonen til en motor basert på et tilbakekobling.

Baneplanlegging: For å få en robot (f.eks. industri- eller mobile roboter, AUVer eller droner, etc.) til å utføre en ønsket bevegelse, så må vi ha en matematisk beskrivelse av denne bevegelsen. Vi vil her anta at bevegelsen skal «bygges» på en tilsvarende måte som de som er vist i figur 9.1, med en startkonfigurasjon, A, en sluttkonfigurasjon, B, samt et sett med konfigurasjoner systemet må være «innom», såkalte *viapunkter*, som vi f.eks. har funnet fra å løse et inverskinematikkproblem (se § 8.2.2). Vi ønsker å koble disse sammen, først for å få en såkalt *sti*, og deretter en *bane*:

- *Sti* (eng. «path»): En bestemt (kontinuerlig) rekkefølge av konfigurasjoner systemet skal gjennom (i dets konfigurasjonsrom).
- *Bane* (eng. «trajectory»[1]): En fullstendig spesifikasjon av en robots hastigheter og akselerasjoner langs en spesifikk sti, altså en kurve i dets tilstandsrom.

[1]På engelsk kaller man en bane for en *orbit*, mens det vi er ute etter her er det som på engelsk kalles en *trajectory*, noe som også oversettes til bane på norsk. Jeg har sett ordet «trajektor» for dette noen steder

- *Baneplanlegging*: Bestemmelse av ønskede hastigheter og akselerasjoner langs en gitt sti slik at spesifiserte tids-, energi-, moment- og kraftbegrensinger, etc., oppfylles.



(a) Robotmanipulator.

(b) Mobil robot sett ovenfra.

Figur 9.1: Fra A til B: Eksempler på applikasjoner hvor såkalte baner er relevante. Disse kobler samme ønskede konfigurasjoner eller punkter, her punkt A og B, via et sett av andre konfigurasjoner systemet skal «innom» på veien, såkalte *viapunkter*. Noen ganger, spesielt når man kun er interessert i en kurve i f.eks. xy-planet, så brukes ordet «sti» i stedet for «bane».

Banefølging: Når man har en bane man ønsker å følge, så er neste problem på listen banefølging. Dette er et reguleringsteknisk (servoteknisk) problem, hvor man må utvikle en regulator som gjør at roboten følger den ønskede banen selv under påvirkning av visse eksterne forstyrrelser. Siden banen som oftest tilsvarer et tidsvarierende refereansesignal, vi har mao. med *referanseføling* å gjøre, så bør man også bruke en foroverkobling av referansen i tillegg til tilbakekonling.

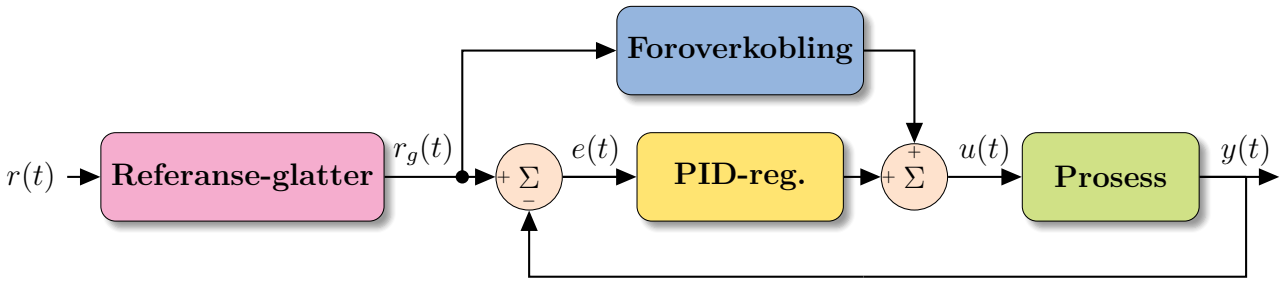
Vi starter dette kapittelet med en enkel form for baneplanlegging, nemlig referanse-glatting, som blant annet kan brukes til mykstarting av motorer og enkle punkt-til-punkt bevegelser.

9.1. Referanse-glatting og myk-starting

Nåværende problem: Et typisk scenario i et reguleringssystemer er at man ønsker å endre fra en konstant referanseverdi (settpunkt), r_0, til en annen ny, konstant referanse, r_1. Dette fører naturlig nok til et sprang i referansen, som kan resultere i flere uønskede fenomener:

(det dukker blant annet opp på en øving i dette faget, som er arvet fra TTK4100), men jeg er usikker på om dette faktisk er et norsk ord eller ikke, så vi bruker bane. Forskjellen på en *orbit* og *trajectory* er for øvrig at førstnevnte bare er en kurve i (tilstands-)rommet, mens det følger med spesifikke tidspunkter (en ønsket «timing») langs denne kurven i en *trajectory*.

1. Spranget i referansen fører til et sprang i avviket, som igjen fører til et sprang i pådraget;
2. Regulatoren vil ikke kunne henge med på spranget;
3. En for aggressiv regulator vil kunne føre til at
   - vi når pådragsorganenes/aktuatorenes ratebegrensninger (hvor fort de kan endre seg);
   - aktuatorene går i metning, slik at fenomenet wind-up potensielt kan oppstå.



Figur 9.2: Illustrasjon av referanse-glatter før regulator med foroverkobling av referansen.

En mulig strategi for å unngå disse problemene er **referanse-glatting**:

Referanse-glatting: I stedet for å endre fra et konstant settpunkt, r_0, til et annet, si r_1, i et sprang (hopp), så generer man et kontinuerlig og tidsvarierende referansesignal som kobler sammen r_0 og r_1; altså, man glatter ut det diskret hoppet til en kontinuerlig funksjon som regulatoren klarer å følge (**referanse-følging**).[a]

[a] Denne strategien egner seg også til såkalt **myk-starting** (av f.eks. elektriske motorer), hvor man prøver å gradvis øke pådraget for å oppnå en ønsket verdi, fremover å gjøre dette som et hopp i en referanse.

En illustrasjon av denne strategien er vist i figur 9.2, hvor en referanse-glatter modifiserer (den potensielt diskontinuerlige) referansen inn, r, til en kontinuerlig referanse $r_g(t)$.

Noen eksempler på slike referanse-glattingsmetoder er illustrert i figur 9.3, hvor man har et sprang ved tiden t_0 fra r_0 til r_1. De forskjellige metodene er som følger:

Første-ordens lavpassfilter (LPF): For en ønsket $a > 0$ (større a = raskere respons), ta

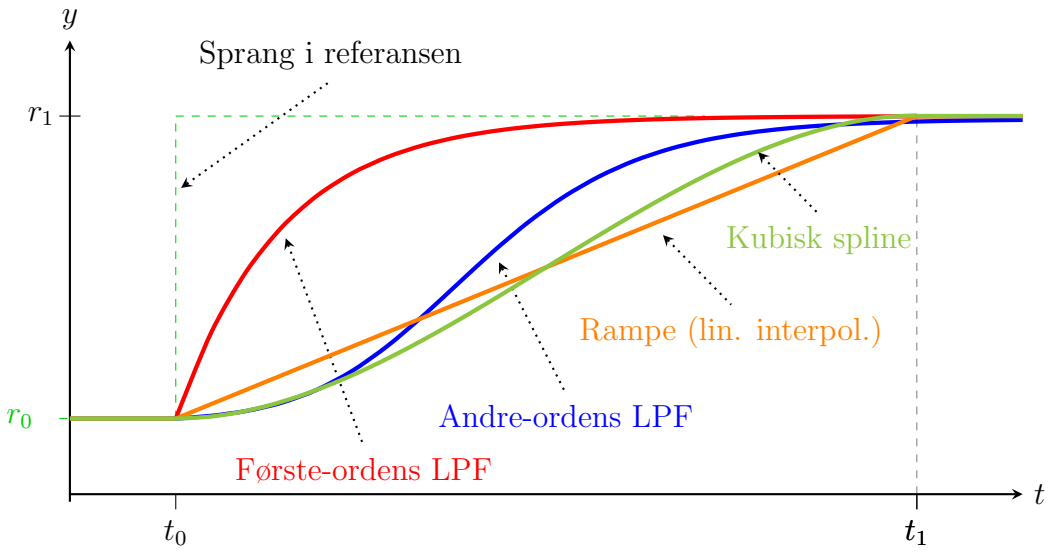
$$\dot{r}_g = a(r - r_g), \quad r_g(t_0) = r_0.$$

Siden dette fører til en knekk ved tiden t_0, egner denne seg best sammen med P(I)-regulator og statisk (konstant) foroverkobling, selv om en mer generell regulator med to frihetsgrader (se § 6.2.2), hvor PID-regulatoren har et stort derivat-filter, også kan brukes.

Andre-ordens LPF: For $a, b > 0$, ta $r_g(t) = z_1(t)$, hvor

$$\begin{aligned} \dot{z}_1 &= z_2, \qquad z_1(t_0) = r_0, \ z_2(0) = 0, \\ \dot{z}_2 &= a(r - z_1) - bz_2. \end{aligned}$$

Her er $\dot{r}_g$ kontinuerlig, så denne metoden er egnet i kombinasjon med PID-regulatorer. Merk at man kan få underdepmpet respons med oversving ved dårlig valg av a og b, slik at $b = 2\sqrt{a}$ (kritisk dempning) anbefales.



Figur 9.3: Illustrasjon av forskjellige referanse-glattings-strategier for et sprang i referansen.

Rampe/ lineær interpolasjon: Lineære interpolering mellom r_0 og r_1:

$$r_g(t) = (r_1 - r_0)\frac{(t-t_0)}{(t_1-t_0)} + r_0 \text{ for } t \in [t_0, t_1]$$
$$r_g(t) = r_1 \text{ for } t \geq t_1.$$

Ulempe at $\frac{dr_g}{dt}$ ikke er velldefinert ved t_0 og t_1, men metoden kan uansett brukes med et nominelt pådrag (analytisk foroverkobling) tross nevnte sprang i $\frac{dr_g}{dt}$ siden den er stykkvis konstant ellers.

Kubisk-spline-interpolasjon: Ved å definere $\hat{t}(t) = \frac{(t-t_0)}{(t_1-t_0)}$, så har man

$$r_g(t) = a\hat{t}^3 + b\hat{t}^2 + c\hat{t} + d \quad \text{for} \quad \hat{t} \in [0,1] \quad \text{(samme som } t \in [t_0, t_1]\text{)},$$
$$r_g(t) = r_1 \quad \text{for} \quad t \geq t_1.$$

Man må her velge (a, b, c, d) slik at $r_g(t_0) = r_0$ og $r_g(t_1) = r_1$, samt tar man som regel $\dot{r}_g(t_0) = \dot{r}_g(t_1) = 0$. Dette krever som oftest at man må løse en matriseligning på formen $\mathbf{Ax} = \mathbf{b}$ mhp. $\mathbf{x}$ for å finne koeffisientene (a, b, c, d). Mer om dette i § 9.2.

Litt mer om disse to typene metoder:

Lavpassfiltere: Ideen her er å sende sende referansen r gjennom et såkalt *lavpassfilter* av ønsket orden. Vi skal se nærmere på lavpassfiltere i § 10.3).

👍 **Fordeler:** Relativt enkle å bruke, ikke veldig numerisk kostbare å implementere, samt lite sensitive til referanse-signalet;

💩 **Ulemper:** Vanskelig å forme responsen akkurat som man ønsker, kan ikke (lett) brukes med et (tidsvarierende) nominelt pådrag, samt kun asymptotisk konvergering til ny referanse (dette er dog ikke et stort problem i praksis).

Interpolasjon: Ideen her er å interpolere mellom den gamle og nye referansen. Vi skal se nærmere på interpolering i § 9.2.

👍 **Fordeler:** Stor frihet i å kunne forme referansen som ønsket, kan brukes i kombinasjon med analytisk foroverkobling;

💩 **Ulemper:** Kan være numerisk kostbare og krevende å implementere.

TODO

9.2. Baneplanlegging og Interpolering

Baneplanlegging går ut på å utvikle en nominell bevegelse, ofte i i form av en bane eller en sti, for en robot eller et autonomt kjøretøy. Det kan for eksempel representere en gitt bevegelse seg fra et startpunkt til et ønsket mål via et sett med ønskede (via-)punkter.

Hvorav *stiplanlegging* bare går ut på å finne rekkefølgen av konfigurasjoner tilsvarende en ønsket bevegelse, så vil baneplanlegging også ta hensyn til hastigheten langs den tilsvarende stien. Vi vil kun se på baneplanlegging, siden dette på mange måter er det enkleste, men vi vil hinte litt til hvordan stiplanlegging også kan fungere.

⚠️ **Forenklinger i sikte:** 1) Vi vil kun se på hvordan man kan lage én enkelt referanse. Hvis vi dog har en robotmanipulator med to ledd, så bør vi jo selvsagt ha to slike referanser. Men metoden vi skal se på her kan da også brukes for hver av disse. 2) Ofte ønsker man at en slik bane skal være best mulig («optimal») mtp. ulike faktorer, samt tilfredsstille visse begrensninger. For eksempel ønsker man ofte å unngå hindringer og kollisjoner (naturlig nok), kanskje vil man at bevegelsen skal ta minst mulig tid, eller eventuelt at den skal være så energieffektiv som mulig. Vi vil dog hoppe over disse delene og i stedet kun fokusere på hvordan vi kan koble sammen (interpolere) gitte punkter vha. splines.

9.2.1 Punkt-til-punkt bevegelse

La oss starte med den enkleste varianten, nemlig å interpolere mellom to punkter slik at vi får et referansesignal tilsvarende en såkalt *punkt-til-punkt bevegelse*. Et eksempel hvor dette er relevant er når et robotledd skal beveges fra å være i ro ved et punkt $a \in \mathbb{R}$ ved tiden t_0 til å kommme til ro ved et punkt $b \in \mathbb{R}$ ved tiden t_f ($> t_0$). Dette kan formuleres matematisk som følger:

Nåværende problem: Generere et tidsvarierende referansesignal, $r(t)$, som sammenhengende (altså uten hopp) kobler sammen to punkter, $a \in \mathbb{R}$ og $b \in \mathbb{R}$, over tidsintervallet $[t_0, t_f]$. Det vil si at $r(\cdot)$ tilfredsstiller følgende:

- $r(t) = a$ og $\dot{r}(t) = 0$ for $t < t_0$;
- $r(t) = b$ og $\dot{r}(t) = 0$ for $t > t_f$;
- $r(\cdot)$ er en kontinuerlig funksjon (har ingen hopp i verdi).

Den enkleste metoden er **lineær interpolering**, hvor vi bare lager en rampe fra a til b:

$$r(t) = \begin{cases} a & \text{når} \quad t < t_0, \\ (b-a)\frac{(t-t_0)}{(t_f-t_0)} + a & \text{når} \quad t_0 \leq t \leq t_f, \\ b & \text{når} \quad t > t_f. \end{cases} \qquad \text{(Lin. interpolering)}$$

Dette er et eksempel på en såkalt lineær *spline*, altså en funksjon av stykkevise lineære polynomer, som blir brukt over forskjellige tidsintervaller (ofte kalt *segmenter*).

Et problem med å bruke lineære splines denne applikasjonen er at hastigheten, altså $\dot{r}$, ikke er kontinuerlig ved overgangen mellom de forskjellige intervallene t_0 og t_f:

$$\dot{r}(t) = \begin{cases} 0 & \text{når} \quad t < t_0, \\ \frac{(b-a)}{(t_f-t_0)} & \text{når} \quad t_0 \leq t \leq t_f, \\ 0 & \text{når} \quad t > t_f. \end{cases}$$

For å unngå et slikt hopp i hastigheten, kan man i stedet bruke et kubisk polynom, noe som kalles for **kubisk-spline-interpolasjon**:

$$r(t) = \begin{cases} a & \text{når} \quad t < t_0, \\ \alpha t^3 + \beta t^2 + \gamma t + \delta & \text{når} \quad t_0 \leq t \leq t_f, \\ b & \text{når} \quad t > t_f. \end{cases} \qquad \text{(Kubisk-spline-interpolasjon)}$$

Her må de fire ukjente koeffisientene $\alpha, \beta, \gamma, \delta$ være slik at følgende begrensninger holder: $r(t_0) = a$, $r(t_f) = b$, og $\dot{r}(t_0) = \dot{r}(t_f) = 0$. Vi kan skrive dette opp på følgende form:

$$\begin{aligned} \alpha t_0^3 + \beta t_0^2 + \gamma t_0 + \delta &= a \\ \alpha t_f^3 + \beta t_f^2 + \gamma t_f + \delta &= b \\ 3\alpha t_0^2 + 2\beta t_0 + \gamma &= 0 \\ 3\alpha t_f^2 + 2\beta t_f + \gamma &= 0. \end{aligned}$$

Dette tilsvarer igjen en matriseligning på formen $\mathbf{Ax} = \mathbf{b}$, hvor

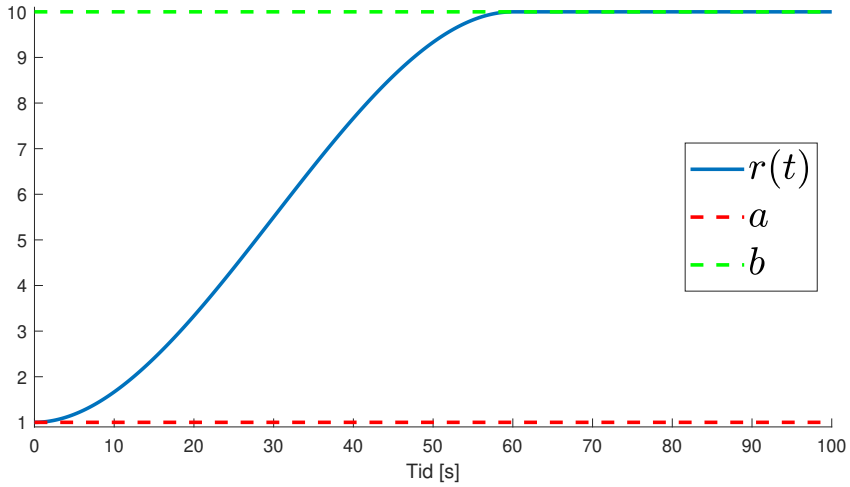
$$\underbrace{\begin{bmatrix} t_0^3 & t_0^2 & t_0 & 1 \\ t_f^3 & t_f^2 & t_f & 1 \\ 3t_0^2 & 2t_0 & 1 & 0 \\ 3t_f^2 & 2t_f & 1 & 0 \end{bmatrix}}_{=:\mathbf{A}} \underbrace{\begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{bmatrix}}_{=:\mathbf{x}} = \underbrace{\begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix}}_{=:\mathbf{b}}.$$

Vi kan derfor finne α, β, γ og δ ved å løse matriseligningen: $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$.

⚠ **Achtung!** Selv om kubiske splines gir kontinuerlige hastigheter, så vil akselerasjonene ikke være det. Dette kan for eksempel være et problem ved posisjonsstyring.

MATLAB-kode som lager en referansen vist i figur 9.4 ved å interpolerer mellom to punkter vha. et kubisk polynom er vist i kodensnutt 9.1. Dette skriptet kan også lastet ned her.

Kodesnutt 9.1: Referanse generert ved å interpolere med et kubsik polynom mellom to punkter.



Figur 9.4: Illustrasjon av referansen for punkt-til-punkt bevegelsen fra kodensnutt 9.1.

```
%% Ønsker å generere en referanse som kobler sammen to punkter,
   a og b. Systemet skal start i ved a ved tiden t0, så komme
   til ro vedb ved tiden t_f
%% Spesifikasjoner
a  = 1;
b  = 10;
t0 = 0;
tf = 60;

%% Finner koeffisientene
A=[  t0^3, t0^2,t0, 1;
     tf^3, tf^2,tf, 1;
   3*t0^2, 2*t0, 1, 0;
   3*tf^2, 2*tf, 1, 0;
   ];
B=[a;
   b;
   0;
   0;
   ];
x=A\B; % Samme som x=inv(A)*B
% alpha = x(1); beta = x(2); gamma = x(3); delta = x(4);
disp(strcat(  "alpha=", num2str(x(1)),...
              ", beta=",  num2str(x(2)), ...
              ", delta=", num2str(x(3)), ...
              ", delta=", num2str(x(4))))
%% Lager referanse funksjon (kubiskPoly definert til slutt)
r = @(t) kubiskPoly(t,a,b,t0,tf,x);
% Plotter så denne:
t = linspace(0,100,1000);
```

```
refSig = zeros(length(t),1);
for i=1:length(t)
    refSig(i)=r(t(i));
end
figure(1);clf(1);
hold on;
plot(t,refSig);
plot([t(1),t(end)],[a,a],'r--');
plot([t(1),t(end)],[b,b],'g--');
ylim([a-0.1,b+0.1]);
legend({'$r(t)$','$a$','$b$'},'Interpreter','latex')
%% Kubisk polynom funksjon
function ref = kubiskPoly(t,a,b,t0,tf,params)
    if t < t0
        ref = a;
        return;
    elseif t > tf
        ref = b;
        return;
    end
    ref = params(1)*t^3+params(2)*t^2+params(3)*t+params(4);
end
```

Tips: For å gjøre det lettere å finne koeffisientene, kan man i stedet bruke polynomet

$$\hat{\alpha}\left(\frac{t-t_0}{t_f-t_0}\right)^3+\hat{\beta}\left(\frac{t-t_0}{t_f-t_0}\right)^2+\hat{\gamma}\left(\frac{t-t_0}{t_f-t_0}\right)+\hat{\delta}.$$

Her må $\hat{\delta}=a$ siden de andre leddene forsvinner når $t=t_0$, mens $\hat{\gamma}=0$ pga. $\dot{r}(t_0)=0$. Videre har vi da $\hat{\alpha}+\hat{\beta}=b-a$ og $3\hat{\alpha}+2\hat{\beta}=0$ når $t=t_f$. Dette gir matriseligningen

$$\begin{bmatrix}1 & 1\\ 3 & 2\end{bmatrix}\begin{bmatrix}\hat{\alpha}\\ \hat{\beta}\end{bmatrix}=\begin{bmatrix}b-a\\ 0\end{bmatrix}\quad\Longrightarrow\quad\begin{bmatrix}\hat{\alpha}\\ \hat{\beta}\end{bmatrix}=\begin{bmatrix}-2 & 1\\ 3 & -1\end{bmatrix}\begin{bmatrix}b-a\\ 0\end{bmatrix}=\begin{bmatrix}2(a-b)\\ 3(b-a)\end{bmatrix},$$

slik at

$$r(t)=\begin{cases}a & \text{når}\quad t<t_0,\\ 2(a-b)\left(\frac{t-t_0}{t_f-t_0}\right)^3+3(b-a)\left(\frac{t-t_0}{t_f-t_0}\right)^2+a. & \text{når}\quad t_0\le t\le t_f,\\ b & \text{når}\quad t>t_f.\end{cases}$$

9.2.2 Interpolering om viapunkter

Anta nå at vi ønsker å lage en bevegelse som starter i ro ved punkt a ved tiden t_0, så gå innom et viapunkt, v_1 ved tiden t_1, for så å komme til ro ved punkt b ved tiden t_f. I tillegg til de samme begrensningen via hadde uten noe viapunkt, altså $r(t_0)=a$, $r(t_f)=b$, og $\dot{r}(t_0)=\dot{r}(t_f)=0$, så har vi nå også $r(t_1)=v_1$, men vi har for så vidt ikke noen begrensning på hva $\dot{r}(t_1)$ skal være.

Nåværende problem: Generere et tidsvarierende referansesignal, $r(t)$, som sammenhengende (altså uten hopp) kobler sammen to punkter, $a \in \mathbb{R}$ og $b \in \mathbb{R}$, over tidsintervallet $[t_0, t_f]$, mens det skal via punktet v_1 ved tiden t_1. Det vil si at $r(\cdot)$ tilfredsstiller følgende:

- $r(t) = a$ og $\dot{r}(t) = 0$ for $t < t_0$;
- $r(t_1) = v_1$;
- $r(t) = b$ og $\dot{r}(t) = 0$ for $t > t_f$;
- $r(\cdot)$ er en kontinuerlig funksjon (har ingen hopp i verdi).

Bruk av en lineær spline er selvsagt det enkleste:

$$r(t) = \begin{cases} a & \text{når} \quad t < t_0, \\ (v_1 - a)\frac{(t-t_0)}{(t_1-t_0)} + a & \text{når} \quad t_0 \leq t < t_1, \\ (b - v_1)\frac{(t-t_1)}{(t_f-t_1)} + v_1 & \text{når} \quad t_1 \leq t \leq t_f, \\ b & \text{når} \quad t > t_f. \end{cases}$$

Denne har dog samme problem som sist, nemlig at hastigheten $\dot{r}$ ikke er en kontinuerlig funksjon. En kubisk spline er derfor generelt sett bedre egnet:

$$r(t) = \begin{cases} a & \text{når} \quad t < t_0, \\ \alpha_1 \left(\frac{t-t_0}{t_1-t_0}\right)^3 + \beta_1 \left(\frac{t-t_0}{t_1 1-t_0}\right)^2 + \gamma_1 \left(\frac{t-t_0}{t_1-t_0}\right) + \delta_1. & \text{når} \quad t_0 \leq t \leq t_1, \\ \alpha_2 \left(\frac{t-t_1}{t_f-t_1}\right)^3 + \beta_2 \left(\frac{t-t_1}{t_f-t_1}\right)^2 + \gamma_2 \left(\frac{t-t_1}{t_f-t_1}\right) + \delta_2 & \text{når} \quad t_1 \leq t \leq t_f, \\ b & \text{når} \quad t > t_f. \end{cases}$$

Oppgave 9.1. Vis at man må ha $\delta_1 = a$, $\gamma_1 = 0$, $\delta_2 = v_1$. Hva med de andre koeffisientene? Hva bør vi også spesifisere for at det skal finnes en unik løsning?

9.2.3 Sti-planlegging*

9.3. Bane- og referanseføLging

Anta nå at vi har løst baneplanleggingsproblemet slik at vi vet en kontinuerlig referanse $r(t)$. Problemet vi skal se på nå er hvordan vi kan lage en regulator som følger denne referansen, derav *referanseføLging*. Hvis denne referansen tilsvarer en bane til f.eks. en robotmanipulator, så kan disse metoden også brukes for *baneføLging*.

Nåværende problem: Gitt en ønsket tidsvarierende referanse (en bane), $r(t)$, til et dynamisk system $\dot{x} = f(x, u)$, design en regulator som får systemets utgang, $y = x$, til følge denne referansen.

Løsningen vi skal se på tar i bruk en såkalt *analytisk foroverkobling*, noe som tilsvarer det nominelle pådraget (se § 6.2.1):

Regulator for referanseføling: Ta pådraget på formen $u = u_{fk} + u_{tk}$ hvor

1. $u_{fk}(t) = u_{nom}(t)$ er det nominelle pådraget;
2. u_{tk} er en stabiliserende tilbakekoblings-basert regulator (f.eks. en PID).

Men hva er det nominelle pådraget? 🤔 Det nominelle pådraget, u_{nom}, er pådraget som (i en perfekt teoretisk verden) vil følge referansen i åpen sløyfe, slik at avviket er og forblir null, altså $e(t) = \dot{e} = 0$. Dette gir

$$\dot{e} = \dot{r} - \dot{y} = \dot{r} - f(r, u_{nom}),$$

hvor jeg har brukt at $y = x = r$ når $e = r - y = 0$. Med andre ord er det nominelle pådraget slik at $\dot{r}(t) - f\big(r(t), u_{nom}(t)\big) = 0$.

La oss demonstrere dette ved hjelp av et enkelt eksempel:

Eksempel 9.1. Sinus-referanse for første-orden system: Gitt et første-ordens system:

$$\dot{y} = -y + u.$$

Vi ønsker at utgangen y skal følge en sinuskurve: $r(t) = \sin(t)$. Merk at vi få svært dårlig følging hvis vi kun bruker en PI-regulator alene. Derfor vil vi også legge til et nominelt pådrag, tilsvarende en analytisk foroverkobling av referansen.

Det nominelle pådraget finner vi ved å anta at $e = \dot{e} = 0$:

$$\dot{e} = \dot{r} - \dot{y} = 0 \quad \Longrightarrow \quad \dot{r} = \dot{y} = -r + u_{nom} \quad \Longrightarrow \quad u_{nom}(t) = \dot{r}(t) + r(t) = \cos(t) + \sin(t).$$

Du kan teste dette i Simulink ved å laste ned denne modellen.

Vi kan også bruke dette for systemer med tidsforsinkelser:

Eksempel 9.2. Predikerende foroverkobling for mykstarting av motor: Rotorhastigheten til en motor skal endres fra $a \in \mathbb{R}$ ved tiden t_0, til $b \in \mathbb{R}$ ved tiden t_f ($> t_0$). Mer spesifikt, så er det ønskelig at hastigheten y skal følge referansesignalet

$$r(t) = \begin{cases} a & \text{når} \quad t < t_0, \\ 2(a-b)\left(\frac{t-t_0}{t_f-t_0}\right)^3 + 3(b-a)\left(\frac{t-t_0}{t_f-t_0}\right)^2 + a & \text{når} \quad t_0 \le t \le t_f, \\ b & \text{når} \quad t > t_f. \end{cases}$$

som er slik at $r(t_0) = a$ og $r(t_f) = b$, samt at $\dot{r}(t_0) = \dot{r}(t_f) = 0$.

Dynamikken til motoren er gitt ved en FOPTF-modell:

$$\dot{y}(t) = \frac{1}{2}\big(-y(t) + u(t - 0.1)\big).$$

Denne har tidskontant $\tau = 2$, stasjonær forsterkning $K = 1$ og tidsforsinkelse $\theta = 0.1$.

Vårt mål er å finne det nominelle pådraget, på formen til den ideelle analytiske foroverkobling av referansen, som en funksjon av tid, altså pådraget som i teorien vil følge referansen i åpen sløyfe. Men siden systemet har en tidsforsinkelse, så må denne faktisk «se fremover i tid»!

Vi starter med å titte på avviksdynamikken:

$$\dot{e}(t) = \dot{r}(t) - \dot{y}(t) = \dot{r}(t) - \frac{1}{2}\big(-y(t) + u(t-0.1)\big).$$

Ved å anta $e = \dot{e} = 0$, og dermed $y = r$, får vi

$$\dot{r}(t) - \frac{1}{2}\big(-r(t) + u_{nom}(t-0.1)\big) = 0.$$

Den ideelle analytiske foroverkoblingen tilsvarende det nominelle pådraget er dermed

$$u_{fk}(t) = u_{nom}(t) = 2\dot{r}(t+0.1) + r(t+0.1).$$

Legg for eksempel her merke til at for $t + 0.1 \leq t_0$, så tilsvarer dette da $u_{fk}(t) = 2\dot{r}(t + 0.1) + r(t + 0.1) = 2 \cdot 0 + a = a$, slik at

$$u_{fk}(t) = \begin{cases} a & \text{når} \quad t < t_0 - 0.1 \\ 2\dot{r}(t+0.1) + r(t+0.1). & \text{når} \quad t_0 - 0.1 \leq t \leq t_f - 0.1 \\ b & \text{når} \quad t > t_f - 0.1 \end{cases}$$

hvor

$$\dot{r}(t) = \begin{cases} 0 & \text{når} \quad t < t_0, \\ 6(a-b)\left(\frac{(t-t_0)^2}{(t_f-t_0)^3} - \frac{(t-t_0)}{(t_f-t_0)^2}\right) & \text{når} \quad t_0 \leq t \leq t_f, \\ 0 & \text{når} \quad t > t_f. \end{cases}$$

For at denne foroverkoblingen skal være realiserbar, altså at vi faktisk kan bruke den, så må vi naturlig nok vite $r(\cdot)$ (i form av a, b, t_0, t_f) minst 100 ms før hastigheten skal endres ved tiden t_0.

Bruk denne lenken til å laste ned et MATLAB-skript og Simulink-modell for dette eksempelet .

Del V

Digital regulering, Filtrering og Estimering

10. Digital regulering og filtrering

De fleste regulatorer er i dag implementert digitalt i en eller annen form for datamaskin (f.eks. en PLS eller mikrokontroller). I motsetning til de kontinuerlige (*analoge*) signalene vi har jobbet med til nå, må vi derfor også ta hensyn til effekten av at de skal bli omgjort til *digitale* signaler i regulatoren. Vi skal derfor i dette kapittelet se på *digital regulering*, som omhandler bruk av digitale signaler til å regulere (analoge) dynamiske systemer. Her spiller også *filtering* av signalene en viktig rolle; filtrering fokuserer blant annet på å forbedre kvaliteten på signaler ved å redusere uønsket støy og forstyrrelser, samt fjerne uønskede effekter som ned-folding/aliasing.

10.1. Digitale reguleringssystemer

Du vil lære mer om dette i IELET2102 – Digitale reguleringssystemer

Alternative kilder: Kap. 23 i [Haugen, 2023].

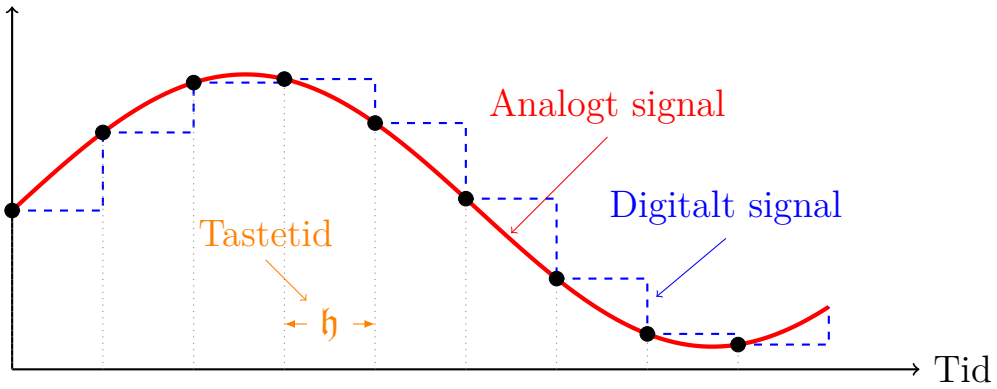
Hvorfor skal du lære dette? 🤷‍♀️ Når man skal implementere en regulator digital så leser man av (måler) verdiene til tilstandene man vil regulere ved gitte tidspunkt. Dette kalles tasting/punktprøving (eng.: sampling), og tiden mellom hvert avlesnings-tidspunkt er det vi kaller tastetiden. Dette kan føre til noen viktig elementer man bør ta hensyn til:

- **Effektiv tidsforsinkelse:** Både filtrering og samplingen av de målte signalene, samt beregningene og diskretiseringen av de ønskede regulatorpådragene fører til tidsforsinkelser som man må ta hensyn til i regulatorsyntesen.
- **Folding/aliasing:** Hvis vi taster for sjelden (tastetiden er for lang) i forhold til de høyeste frekvensene i signalet/tilstanden vi vil måle, så kan disse høy-frekvente delene (lløst forklart) «ødelegge» målingene pga. av et fenomen som kalles *(ned-)folding/aliasing*. Dette problemet kan delvis unngås ved å implementere et analogt lavpass-filter kalt *folding-filter*, før man taster signalet.

10.1.1 Oppbygging og signalomsetning

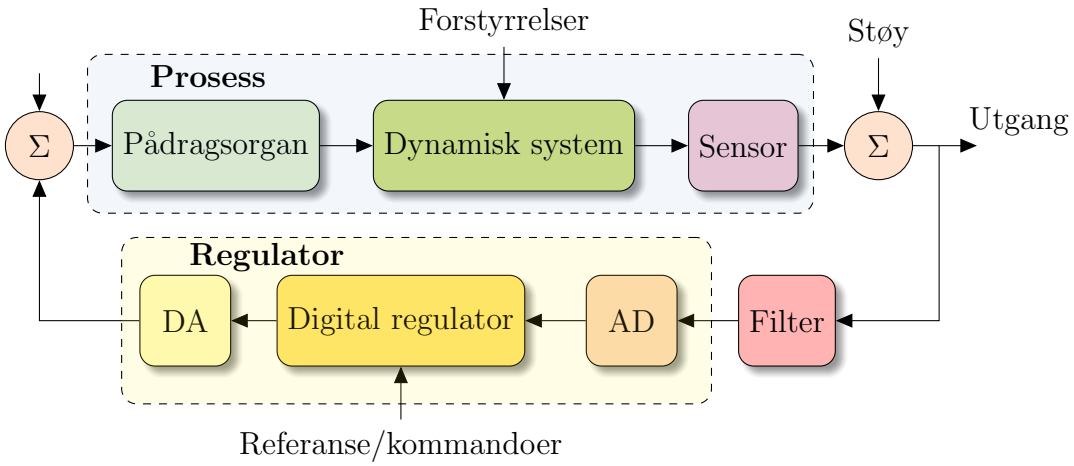
Signaler er informasjonsbærere i reguleringssystemer, og riktig behandling av signalene er en viktig del av reguleringssystemet. Vi skal her skille mellom to typer signaler som begge dukker

opp i digitale reguleringssystemer: analoge og digitale. En hovedforskjell på disse signalene er at analoge signaler er kontinuerlige i tid, mens digitale signaler er det vi kaller diskrete i tid, det vil si de består av øyeblikksverdier. De to typene signaler er illustrert i figur 10.1.



Figur 10.1: Illustrasjon av et analogt og et digitalt signal- Det digitale signalet fås ved å taste det analoge signalet med en tastetid på ɦ sekunder.

En illustrasjon av oppbyggingen til et digitalt reguleringssystem er vist i figur 10.2. De fleste fysiske størrelsene vi finner i slike reguleringsprosess er analoge. Dette kan være nivået i en tank, hastigheten til en bil, posisjonen til en robot eller spenningen over en kondensator.



Figur 10.2: Illustrasjon av et reguleringssytem med en digital regulator.

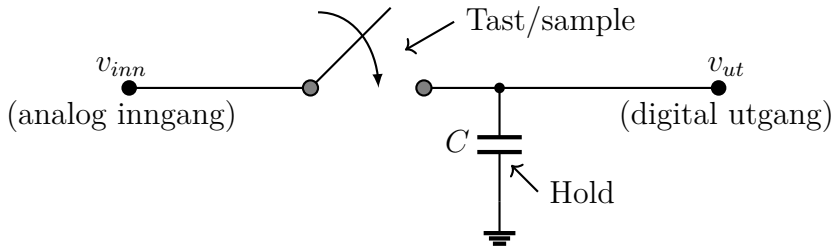
For at en datamaskin som kun opererer med digitale signaler skal kunne kommunisere med disse prosessene må vi kunne gjøre om de analoge signalene vi måler vha. av måleinstrumenter (sensorer) til digitale signaler. Dette gjøres ved hjelp av en *analog-til-digital (AD) omformer*. Overføringen fra et analogt signal fra et måleinstrumenter til et digitalt signal som kan brukes i en datamaskin kalles form AD-omsetning. Overføring fra datamaskin til pådragsorgan, på den annen side, krever at digitale signaler gjøres analoge. Dette er kjent som DA-omsetning og gjøres av en *digital-til-analog (DA) omformer*. Det er noen fordeler og ulemper med både analoge og digitale signaler:

- Støy: 💩 Analoge signaler er sensitive til støy; 👍 Digitale signaler er ikke sensitive.
- Informasjon: 👍 Analoge signaler er tapsfrie; 💩 Digitale signaler kan tape informasjon.

10.1.2 Tasting/sampling og zero-order-hold

La et signal signal bli avlest/tastet/samplet (eng.: sampled) med faste tidsmellomrom $\mathfrak{h}$. Vi kaller $\mathfrak{h}$ for *tastetiden*, mens $f_s := 1/\mathfrak{h}$ (eventuelt $\omega_s := 2\pi/\mathfrak{h}$) er *tastefrekvensen* (eventuelt sampling- eller prøvetakings-rate). Vi bruker enheten Hertz [Hz=1/s] for f_s og [rad/s] for ω_s. For å omsette fra analog til digital brukes ofte en tast-og-hold krets (eng.: sample and hold) som vist i figur 10.3. Bryteren åpnes og lukkes med en frekvens lik $f_s = \frac{1}{\mathfrak{h}}$, og hvor øyeblikksverdier av den analoge inngangsspenningen, v_{inn}, kan måles som utgangsspenningen v_{ut} over kondensatoren C. En tast-og-hold krets opererer i ett av tre modus:

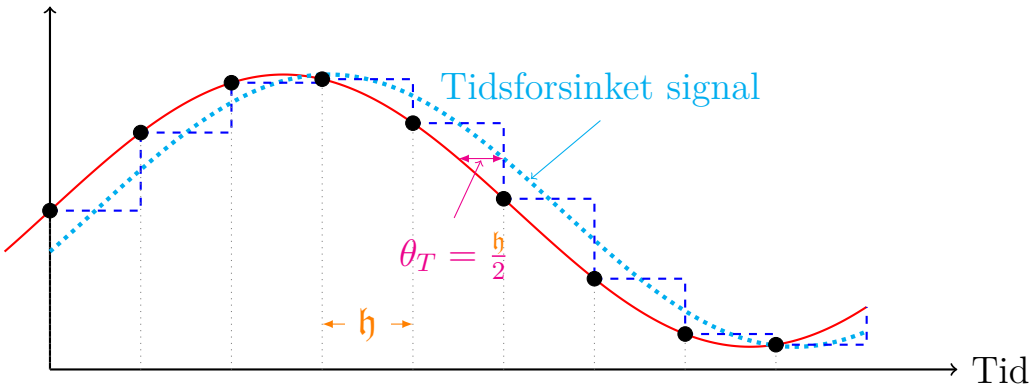
1. **Holdemodus:** Bryteren er åpen; utgangsverdien påvirkes ikke av signalet på inngangen.
2. **Følgemodus:** Bryteren er sluttet; utgangen følger inngangen
3. **Innsvingningsmodus:** Den tiden det tar for v_{ut} å svinge seg inn til riktig verdi.



Figur 10.3: En tastekrets (sample-and-hold) for et spenningssignal.

En slik tastemetode som vi har beskrevet her kalles *nullte-ordens* hold (eng. «zero order hold (ZOH)»). Det finnes også forskjellige typer av 1.ordens hold som også involverer en integrator.

10.1.3 Effektiv tidsforsinkelse ved digital regulering



Figur 10.4: Tasting fører til en tidsforsinkelse.

En effekt av tasting er at hvis vi glatter ut det tastede signalet så får vi et signal som er tidsforsinket med en halv taste-periode. Dette er illustrert i figur 10.4. Ved diskret regulering bør man ta hensyn til denne og andre tidsforsinkelser som oppstår pga. AD- og DA-omformingene.

Anta for eksempel at man taster/sampler et eller flere signaler med tastid $\mathfrak{h}$ [s], og at pådraget/regulatorutgangen bestemmes basert på dette. En slik tidsforsinkelse, θ_D, vil være i innen følgende interval:

$$\frac{\mathfrak{h}}{2} \leq \theta_D \leq \frac{3}{2}\mathfrak{h} \qquad \text{(Effektiv tidsforsinkelse ved digital regulering)}$$

Dette intervallen kommer fra å slå sammen følgende to tidsforsinkelser:

- **Forsinkelse fra hold-element:** Hvis regulatorutgangen blir holdt konstant mellom hver iterasjon («zero-order hold» (ZOH)), så vil dette før til en effektiv tidsforsinkelse på ca . halvparten av tastetiden, altså tilsvarende $\frac{\mathfrak{h}}{2}$ sekunder.
- **Forsinkelse fra beregning:** Tiden fra man sampler (fra AD-konverteren) et nytt regulatorsignal blir bestemt (til DA-omformeren) tar det også en viss tid som bør regnes som en effektiv tidsforsinkelse. Denne forsinkelsen er **minimalt** 0 sekunder, som tilsvarer at regulatorutgangen blir satt umiddelbart; og **maksimalt** $\mathfrak{h}$ seksunder, som tilsvarer at man bruker hele tidsintervallet til å bestemme/regne ut regulatorutgangen.

10.1.4 Nyquist frekvensen og frekvens-folding/aliasing

Alternative kilder: Wikipedia; Steve Bruntion video (FcXZ28BX-xE).

For at vi skal kunne bevare all vesentlig informasjon i et analogt signal når vi taster/sampler det, så må vi ha en høy nok taste-frekvens i forhold til den maksimale (ønskede) frekvensen i signalet. Det er faktisk slik at vi kan gjenskape et analogt signal fra dets målte verdier hvis vi har en tastefrekvens som er høyere enn den såkalte Nyquist-frekvensen:

(Nyquist–Shannon–Kotelnikov samplingsteorem) Et analogt, bånd-bregrenset[a] signal kan gjenskapes fra diskret målinger med konstant tastetid $\mathfrak{h}$ hvis tastefrekvensen $f_s := 1/\mathfrak{h}$ er over dobbelt så stor som den høyeste frekvensen, $f_{\max}$, i signalet:

$$f_s > 2f_{\max}.$$

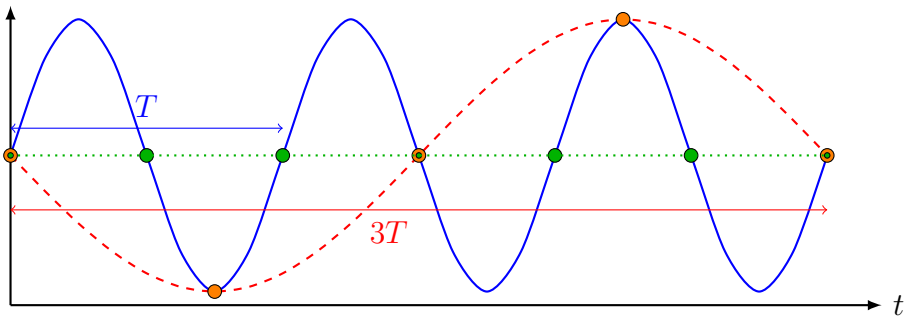
For en viss tastetid $\mathfrak{h}$, så kaller vi grensefrekvensen $f_N := \frac{1}{2}f_s = \frac{1}{2\mathfrak{h}}$ for **Nyquist-frekvensen**, som også kan skrives som

$$\omega_N = 2\pi f_N = \frac{\pi}{\mathfrak{h}}. \qquad \text{(Nyquist-frekvensen)}$$

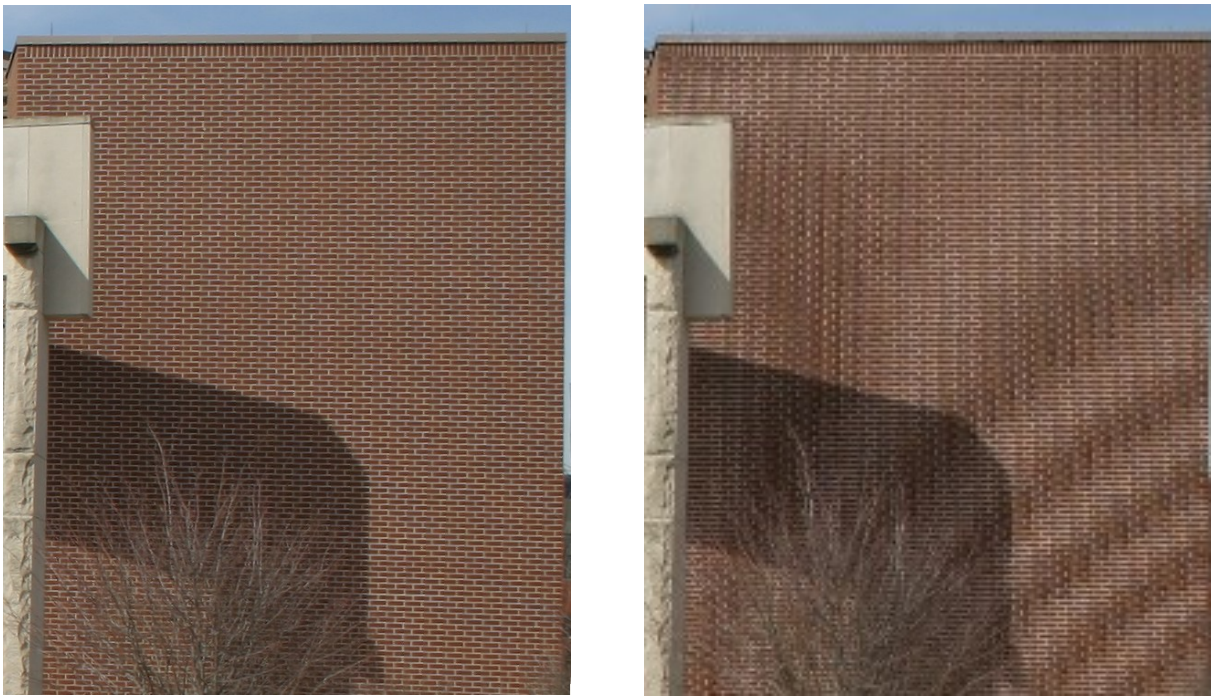
[a] Et signal sies å være **båndbegrenset** hvis det har en endelig maksimal frekvens, og dermed kan representeres med en trunkert Fourier-serie.

Så hva kan skje hvis vi har et signal som har frekvenskomponenter som er høyere enn Nyquist-frekvensen for den gitte tastetiden, utover at vi ikke kan gjenskape signalet digitalt? Jo, det kan føre til et fenomen kalt (ned-)folding/aliasing.

Begrepet folding refererer til symmetrien til et signal om Nyquist-frekvensen i frekvensdomenet. Frekvenser over Nyquist-frekvensen «folde» (brettes) over Nyquist-frekvensen (derfor blir denne også kalt foldingsfrekvensen). Dette fører til at det målte signalet fremstår som at



Figur 10.5: Illustrasjon av folding: Et signal (blått) med periode T blir samplet med tastetid $2T/3$ (se de oransje punktene) $T/2$ (se de grønne punktene). Ut fra disse målingene er disse identiske med hendholdsvis et signal tilsvarende den stiplede linjen (i rødt) som har periodetid $3T$ og den prikkede, grønne null-linjen (begge har altså lavere frekvens).



Figur 10.6: Illustrasjon av folding/aliasing, hvor «bølger» i murveggen oppstår bildet til høyre siden det har lavere oppløsning enn bildet til venstre. Bilder fra Wikipedia (CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=644816).

det har lavere frekvens enn det originale analoge signalet, slik som det er illustrert i figur 10.5. Som vist i figur 10.6, så oppstår også fenomenet når man ønsker å lagre bilder digitalt.

Fra dette kan vi gjøre følgende viktige observasjon:

⚠️ **Oh no!** Hvis vi taster et signal som har komponenter med frekvens høyere enn Nyquist-frekvensen, så vil disse komponentene fremstå som komponenter med lav frekvens (altså lavere enn Nyquist-frekvensen) i det samplede signalet.

Dette er jo itte bra hvis vi vil bruke et slikt signal i en regulator! 😱 Så hva kan vi gjøre? Jo, vi kan prøve å fjerne komponentene med frekvens høyere enn Nyquist-frekvensen *før* signales

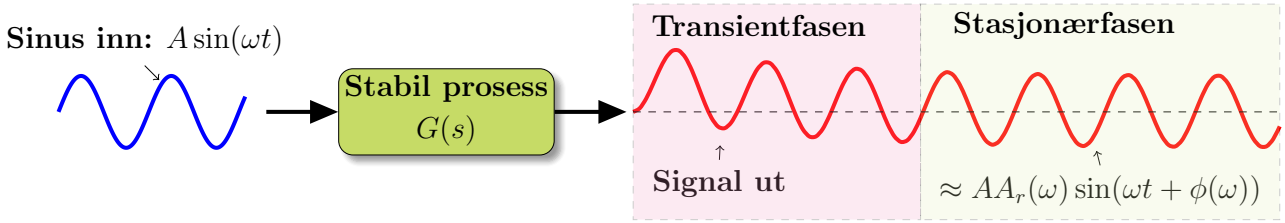
tastes. Dette kan gjøres ved hjelp av et analogt lavpass-filter, et såkalt *folding-filter*, også ofte kalt *anti-aliasing-filter*. Vi skal se mer på dette i § 10.3.

10.2. Frekvensanalyse

Du vil lære mer om dette i IELET2002 – Reguleringsteknikk.

Hvorfor skal du lære dette? 🤷‍♀️ Frekvensanalyse lar en studere hvordan utgang til et systemet endrer seg i forhold til inngangssignal med forskjellig frekvens. Dette er et essensielt vektøy for blant annet filtrering og støyreduksjon, samt kan det brukes til å studere et reguleringssystems stabilitet og ytelse.

Hva er frekvensanalyse? Frekvensanalyse baserer seg på et systems frekvensrespons. Med frekvensrespons menes den statiske responsen (altså responsen etter alle transienter har dødd ut) til et system på en sinusformet inngang med en gitt frekvens, slik som vist i figur 10.7. Denne analysen krever derfor at prosessen er stabil (ellers dør jo ikke transientene ut).



Figur 10.7: Frekvensrespons: Et sinusignal på inngang til et stabilt, lineært, tids-invariant system (her representert ved en overføringsfunksjon $G(s)$, vil, etter en transientfase, føre til et nytt sinussignal ut med samme frekvens, men med en annen amplitude og fase.

Eksempel 10.1. Gitt første-ordens reguleringssystemet $\dot{y} = -ay + bu$ med konstante parametere $a > 0$ og b og initialverdi $y(0) = y_0$. Anta at $u = A\sin(\omega t)$, slik at vi er interessert i løsningen på følgende initialverdiproblem:

$$\dot{y} = -ay + b \cdot A\sin(\omega t), \quad y(0) = y_0.$$

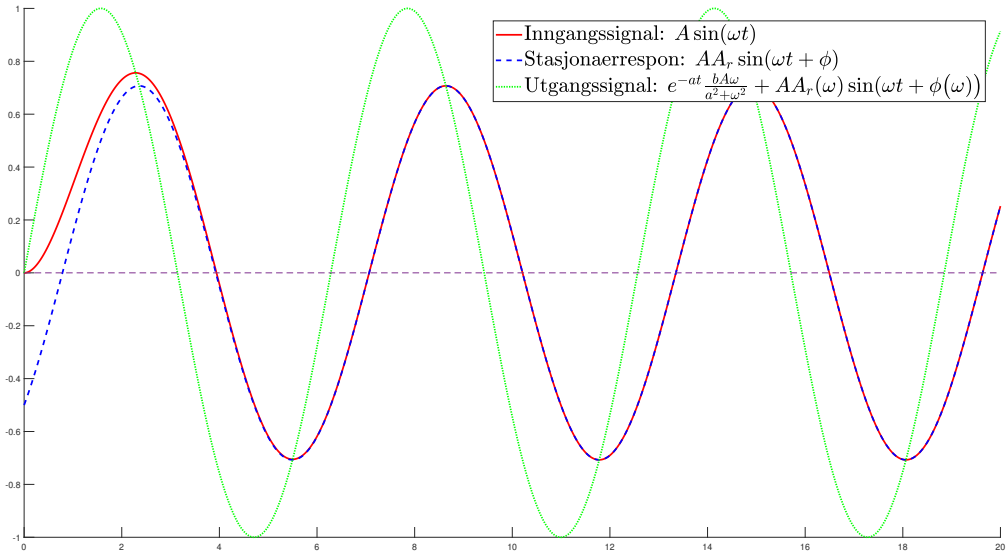
Denne løsningen er (se oppgaven under hvis du har lyst på en utfordring)

$$y(t) = e^{-at}\left(y_0 + \frac{bA\omega}{a^2+\omega^2}\right) + AA_r(\omega)\sin(\omega t + \phi(\omega)) \tag{10.1}$$

hvor $\phi(\omega) = \arctan\left(\frac{-\omega}{a}\right)$ er *faseforskyvingen* og $A_r(\omega) = b/\sqrt{a^2+\omega^2}$ er *amplituderatioen*.

Siden $a > 0$ vil leddet $e^{-at}\left(y_0 + \frac{bA\omega}{a^2+\omega^2}\right)$ etter hvert «dø ut», slik at vi står igjen med et sinussignal på utgangen.

Frekvensresponsen til dette systemet med $a = b = 1$ og $y(0) = 0$ for inngangssignalet $\sin(t)$ er vist figur 10.8, hvor for $\omega = 1$ man har $A_r = 1/\sqrt{2}$, $\phi = \arctan(-1) = -\frac{\pi}{4}$, og at denne faseforskyvingen fører til at utgangssignalet er forsinket med $\theta_\phi = -\phi/\omega = \frac{\pi}{4}$ sekunder.



Figur 10.8: Frekvensresponsen til systemt i eksempel 10.1 med $a = b = A = \omega = 1$.

Vi har da at en sinussignal med amplitude A og frekvens ω, altså $A\sin(\omega t)$, på inngangen gir (etter transientene har dødd ut) et sinussignal $AA_r(\omega)\sin\big(\omega t + \phi(\omega)\big)$ på utgangen, hvor

- $A_r(\omega)$ er **amplituderatioen**, altså amplituden på utgangssignalet delt på amplituden til inngangssignalet (ratioen mellom dem) ved et gitt frekvens ω, slik at $A_r(1) = 2$ betyr at amplituden på utgangssignalet er dobbelt så stor som amplituden til inngangssignalet ved frekvensen $\omega = 1$;
- $\phi(\omega)$ er **faseforskyvingen** til utgangssignalet i forhold til inngangssignalet for en gitt frekvens ω. Dette fører igjen til at utgangssignalet er «tidsforsinket» med $\theta_\phi(\omega) = -\frac{\phi(\omega)}{\omega}$ sekunder i forhold til inngangssignalet.

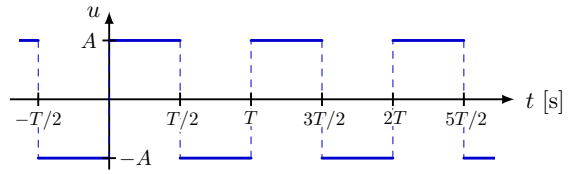
Å vise at illustrasjonen i figur 10.7, altså sinus inn gir ny sinus ut for et stabilt system, er noe mer knotette enn man kanskje skulle tro i tidsomenet, så hvis du har lyst på en utfordring:

Oppgave 10.1. Utled løsningen i eksempel 10.1

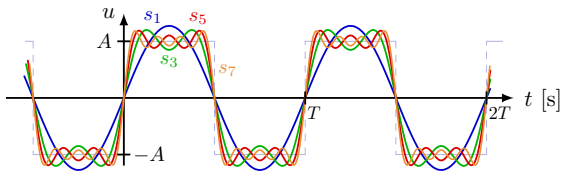
I frekvensresponsmetoder varierer vi frekvensen på inngangssignalet over et visst område og studere den resulterende responsen sin stasjonære fase.
Men hvorfor bryr vi oss om det? 🤔 Jo, på grunn av den såkalte Fourier-transformen!

Fourier-rekker kan brukes til å representere de fleste signaler vi bryr oss om (signalet må være stykkvis kontinuerlig, ha begresent amplitude, og må kunne deles opp i biter som alle «forelenges» som et periodisk signal); se for eksempel figur 10.9 og disse YouYube-videone k8FXF1KjzY0 og YUBe-ro89I4. Dette er viktig, siden det såkalte superposisjonsprinsippet lineære systemer (altså dynamiske systemer beskrevet av en lineære differensialligning) holder for slike systemer: hvis $y_1(t)$ tilsvarer systemets respons for et gitt inngangssignal $u_1(t)$, og $y_2(t)$ er responsen til inngangsignalet $u_2(t)$, så er $(y_1(t) + y_2(t)$ responsen til inngangssignalet

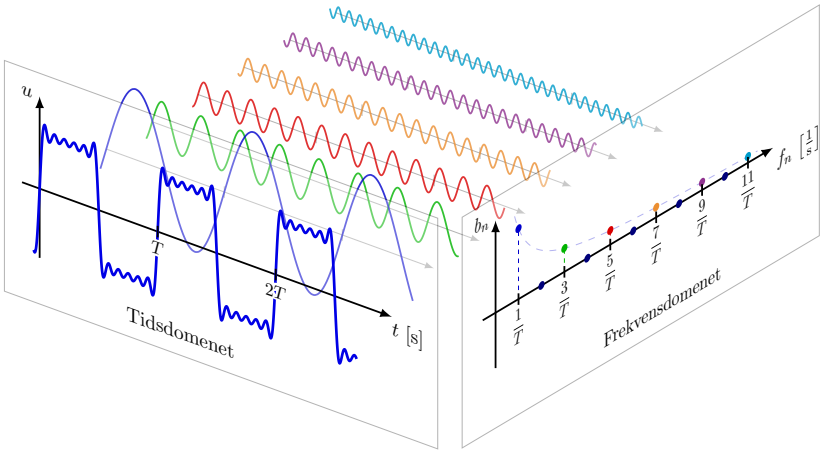
$(u_1(t) + u_2(t))$. I frekvensanalyse (for lineære og stabile systemer) kan vi derfor «dele opp» et inngangssignal i dets frekvenskomponenter (sinussignaler) og studere disse individuelt, og så få den fulle responsen fra superposisjonsprinsippet.



a) Original firkant-puls funksjon.



b) Fourier-rekke approksimasjon.



c) Tidsdomenet vs frekvensdomenet.

Figur 10.9: Fourier-rekke-approksimasjon av firkantspuls-funksjon; figur fra https://tikz.net/fourier_series/.

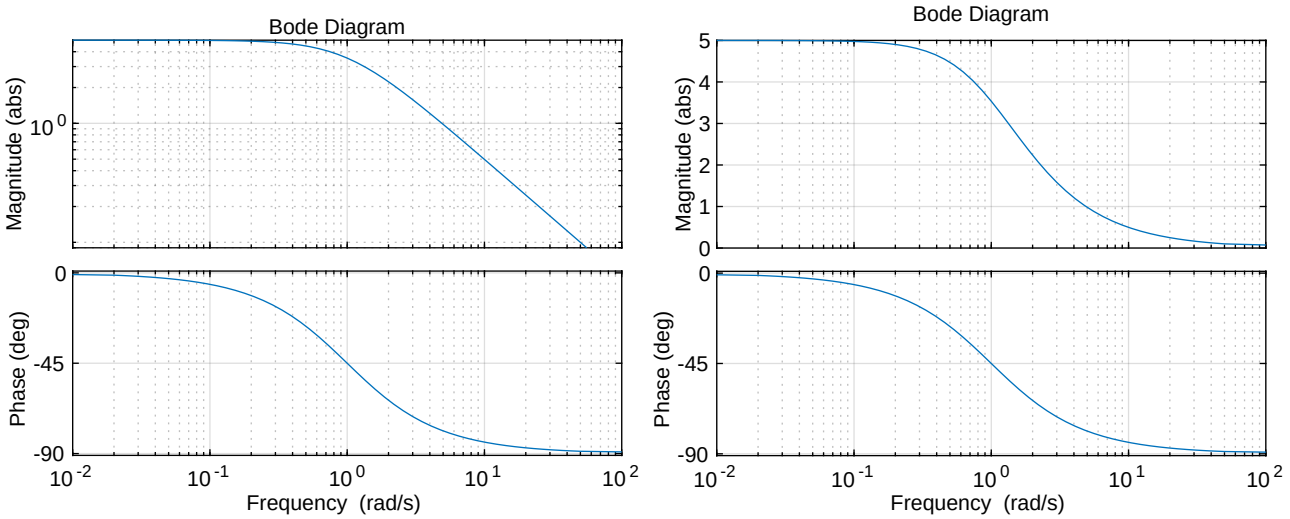
Fun facts, bemerkninger og annet dill dall (you may skip)

For mer om Fourier-transformen og Fourier-serier, se f.eks. https://youtu.be/spUNpyF58BY og https://youtu.be/1JnayXHhjlg.

Analyse av amplitudeforsterkning og faseforskyvning via Bode-plott

Som vi har sett fra figur 10.7 og eksempel 10.1 så kan vi studere frekvensresponsen for et gitt sinussignal på inngangen med frekvens ω. Nærmere bestemt, så kan vi se på amplituderatioen $A_r(\omega)$ og faseforskyvingen $\phi(\omega)$ på utgange. Begge disse er funksjoner av inngangsfrekvensen ω, altså dere verdi vil generelt sett endre seg hvis man endrer ω. En måte å visualisere dette på er ved hjelp av et såkalt *Bode-plott*. Et **Bode-plott** består egentlig av to plott hvor frekvensen, ω, er plottet logaritmisk på den horisontale aksen på begge, mens de vertkiakle aksene til hvert av plottene er som følger:

1. Ett plott med amplituderatioen, $A_r(\cdot)$, normalt sett på en logaritmisk skala slik som i venstre del av figur 10.10;
2. Ett for faseforskyvingen ϕ.



Figur 10.10: Bode-diagram av første-ordens system. Venstre: med log-skala; høyre: uten.

⚠ **Avvikk fra normen!** Det vanligste er å vise amplituderatioen på det ene plottet i enheten desibel (dB), gitt ved $A_r^{dB}(\omega) = 20 \log A_r(\omega)$. Vi skal dog ikke bruke dette her; i stedet skal vi i disse notatene vise amplituden på en logaritmisk-skalert akse (base 10).[a] I MATLAB kan du enkelt konvertere mellom disse vha. kommandoene mag2db og db2mag

[a] Grunnen til at vi ikke skal bruke desibel handler både om at jeg ikke vil at vi skal bruke tid på dette nå, samt at det er min (og andres) mening at desibel ikke akkurat er veldig intuitivt (mennesker sliter som regel med å få en intuitiv forståelse av eksponentielle og logaritmiske representasjoner). Dermed er det ikke ideelt å ta det i bruk i en pedagogisk sammenheng. På den annen side, så brukes desibel veldig ofte innen mange felt (også reguleringsteknikk), så man må også mestre dette og kunne konvertere ved behov.

Hvordan lage et Bode-plott? Hvis vi har en overøfringsfunksjon $P(s)$ (se § 2.8) så kan vi lett lage et Bode-plott i MATLAB ved å bruke en av følgende kommandoer:

MATLAB-kommando: bode eller bodeplot. **NB!** Disse bruker i utgangspunktet en desibel-skala (se kodesnutt 10.1 for hvordan dette kan unngås).

For eksempel, så har systemet $\dot{y} = -y + 5u$ overføringsfunksjonen $P(s) = \frac{Y(s)}{U(s)} = \frac{5}{s+1}$. Et eksempel på hvordan man kan plotte Bode-plottet til denne overføringsfunksjonen er vist i kodesnutt 10.1, med resulterende plott vist i figur 10.10.

Kodesnutt 10.1: Bode-plot av første-ordens system i MATLAB både med og uten at amplituden er på en logaritmisk skala.

```
s = tf('s'); % Initialisere s-operatoren
P = 5/(s+1); % Overforingsfunksjon
frekInt = {0.01,100}; % Frekvensintervall vi skal plotte over
plotoptions = bodeoptions; %Innsillingsobjekt
plotoptions.Grid = 'on'; % Skru paa hjelpelinjer
plotoptions.MagUnits='abs'; % fra desibel til absoluttverdi
% Plott med amplituderatio uten log-skala:
bode(P,plotoptions,frekInt)
% Plott med amplituderatio paa log-skala:
```

```
plotoptions.MagScale='log'; % amplituden paa en log-skala
bode(P,plotoptions,frekInt)
```

Fun facts, bemerkninger og annet dill dall (you may skip)

For en gitt overføringsfunksjon $G(s)$ så baserer et Bode-plott seg på at amplituderatioen tilsvarer $A_r(\omega) = |G(j\omega)|$, samt at $\phi(\omega) = \angle G(j\omega)$ hvor $j = \sqrt{-1}$ er det imaginære tallet, mens $\angle G(j\omega)$ indikerer vinkelen til det det komplekse taller $G(j\omega)$ i det komplekse plan målt positivt mot klokken fra den horisontale reelle aksen.

Men hva kan vi bruke et Bode-plott til? Et Bode-plott kan brukes til så mangt, ikke minst innen reguleringsteknikk. Et relevant bruksområde er å illustrere en reguleringssløyfe sin *båndbredde* , som sier noe om systemets ytelse. La oss ta et eksempel for å illustrere dette:

Eksempel 10.2. Gitt følgende første-ordens system:

$$\dot{y} = -y + u.$$

Anta at vi bruker en P-regulator $u = 10(r - y)$, slik at den lukkede sløyfen tilsvarer

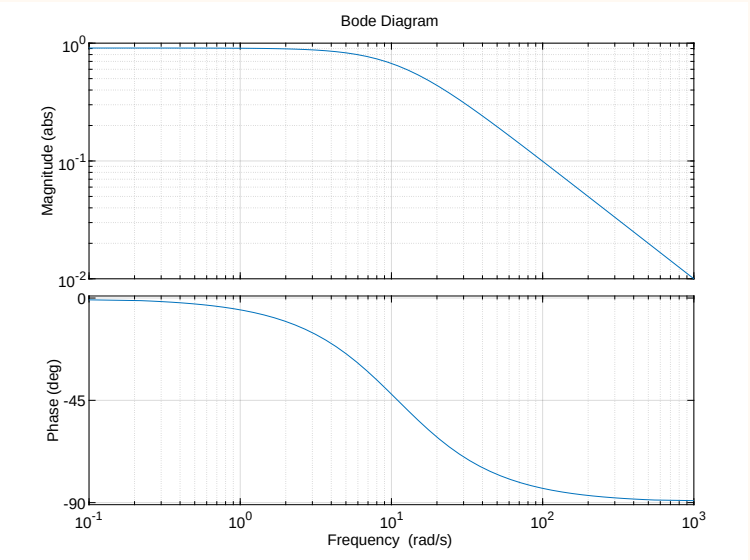
$$\dot{y} = -11y + 10r.$$

Legg merke til at overføringsfunksjonen (se § 2.8) fra r til y er

$$G(s) = \frac{Y(s)}{R(s)} = \frac{10}{s + 11}.$$

Mål: Vi ønsker å finne ut hvor stor frekvens ω vi kan ha på referansesignalet $r(t) = \sin(\omega t)$ for å forstsatt ha god referanseføling, altså at utgangen $y(t)$ er tilnærmet lik $r(t)$ etter transientene har dødd ut. Vi kaller denne frekvensen for *båndbredden* til den lukkede sløyfen, altså hvor bredt frekvensbåndet hvor regulatoren gir god referanseføling.

Ved å bruke kode tilsvarende kodesnutt 10.1 får vi følgende:



Legg merke til at amplituderatioen («Magnitude») er lavere enn $10^0 = 1$ for alle frekvenser («Frequency») større enn $10^{-1} = 0.1$, samt at denne stuper nedover fra ca. $\omega = 10^1 = 10$. Dette indikerer at vi uansett ikke kan få perfekt referanseføIging selv for lave frekvenser som $\omega = 0.1\,\text{rad/s}$, samt at den raskt blir svært mye dårligere fra $\omega = 10\,\text{rad/s}$ og oppover. Legg også merke til at fasen («Phase») faller raskt fra ca. $\omega = 1\,\text{rad/s}$, noe som også indikerer dårlig referanseføIging siden man da har havnet på etterskudd. Med andre ord, kan vi si at vi har grei referanseføIging opp til ca. $\omega = 1\,\text{rad/s}$, som da er den lukkede sløyfens båndbredde.[a]

[a] Det er mer vanlig å si at båndbredden, ω_{bb}, er slik at $A_r(\omega_{bb}) \approx 1/\sqrt{2}$, som her er ca. $\omega_{bb} = 10\,\text{rad/s}$.

10.3. Lavpass- og folding-filtere

Alternative kilder: §11.3 i [Balchen et al., 2016]

Hva skal vi med et lavpass-filter? Et lavpass-filtre har som oppgave å **1)** fjerne (uønskede) høy-frekvente deler av et signal (f.eks. støy); men samtidig **2)** behold de delene som er under en gitt "cutt-off" frekvens, og dermed oppnå ønsket båndbredde. Dette kan brukes til å designe et analogt **foldingfilter** for å unngå folding/aliasing grunnet sampling (se § 10.3.1).

10.3.1 Folding-/anti-aliasing-filtre

Formålet med et folding-filtre (også kalt anti-aliasing-/konvolusjons-filtre) er å forhindre frekvens-innhold nær eller over Nyquist-frekvensen blir med videre i en signalbehandling (for bruk i digital regulering for vår del) etter tasting. Flere kilder [Dessen, 2019, Wittenmark et al., 2002] anbefaler brukene av slike analoge (lavpass-)filtere for digitale reguleringssystemer.

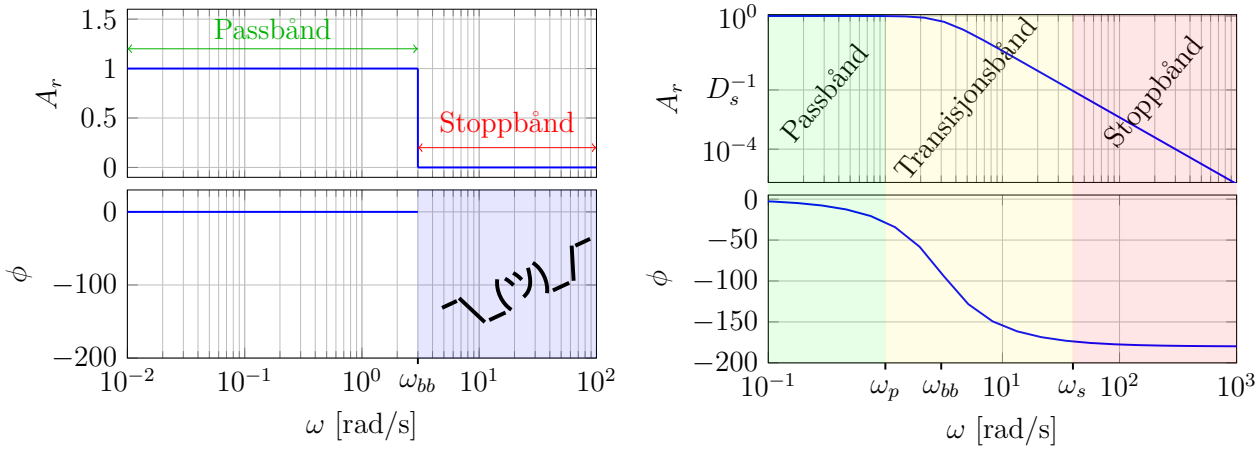
Folding-filtre blir som oftest implementer som analoge elektroniske kretser, noe som gjør dem rimelig enkle. Dette gjelder også designprosedyren, ettersom man normalt sett ikke trenger å ta store hensyn til det digitale reguleringssystemet utover Nyquist-frekvensen.

Hvordan designe og implementere et slikt filter? Man tar utgangspunkt i Nyquist-frekvensen, og spesifiserer ønsket dempning av det filtrerte signalet ved denne frekvensen. Basert på dette, implementerer man et lavpassfilter, som man igjen konstruerer fysisk ved hjelp av en elektrisk krets.

Tips: I en reguleringsløyfe kan det være lurt å også sende referansen gjennom et lavpassfilter tilsvarende det brukte foldingfiltre slik at det får en tilsvarende forsinkelse grunnet den ekstra dynamikken. Dette filteret kan dog implementeres digitalt.

10.3.2 Lavpassfiltere

Som nevnt over, så har et lavpassfilter to viktige oppgaver:



a) Ideelt lavpassfilter.

b) Realiserbart lavpassfilter.

Figur 10.11: Illustrasjon av et ideelt lavpassfilter til venstre og et realiserbart (andre-ordens) filter til høyre.

1. slippe gjennom lavfrekvente komponenter av et signal mest mulig uforstyrret (**passbånd**);
2. blokkere høyfrekvente komponenter i størst mulig grad (**stoppbånd**).

I skillet mellom passbåndet og stoppbåndet har vi det vi kaller **cut-off-frekvensen** (kanskje **avkuttsfrekvens** på godt norsk?). Vi ønsker altså å «kutte» alle frekvenser som er høyere enn denne. **I en perfekt verden**, hvor man kunne ha laget et *ideelt* lavpassfilter slik som vist i se figur 10.11, ville grensen mellom stoppbånd og passbånd være helt presist definert, og det ville ikke være noen endring i fasen i passbåndet. Dette er dog umulig å få til i praksis. Et realiserbart lavbpassfilter har derfor også et **transisjonsbånd** som ligger mellom passbåndet (det frekvensspekteret av signalet som i liten grad blir endret) og stoppbåndet (de delene av signalet som i stor grad blir dempet ut).

Et eksempel på et første-ordens lavpass filter med inngang u og utgang y er

$$\dot{y} = \frac{1}{\tau}(u - y) \quad \text{i tidsomenet, og} \quad \frac{Y(s)}{U(s)} = \frac{1}{\tau s + 1} \quad \text{i s-domenet,} \qquad \text{(FO lavpassfilter)}$$

hvor τ er filterets tidskonstant, mens $\omega_{bb} = 1/\tau$ er filterets «cut-off-frekvens». Vi vet jo fra § 2.5.1 om første-ordens systemer at desto mindre τ er, desto større er ω_{bb} og desto raskere er filterets dynamikk. La oss ta et eksempel:

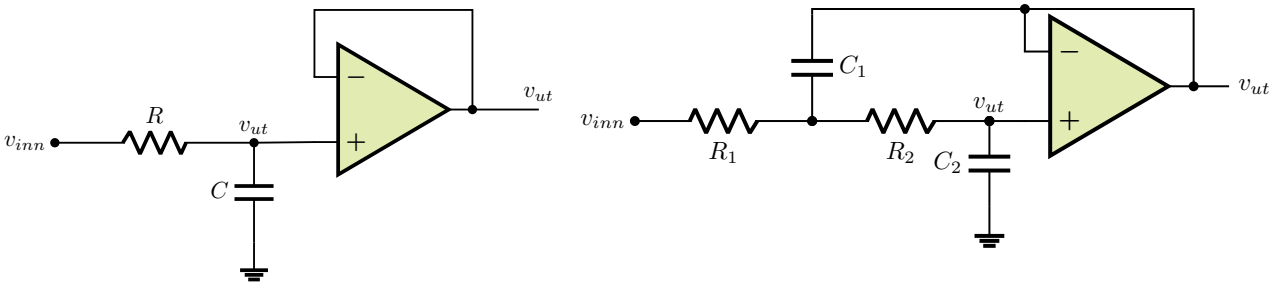
Eksempel 10.3. Vi ønsker å lage et første-ordens lavpass-filter, $\dot{y} = -\frac{1}{\tau}(u-y)$, som demper deler av et inngangssignal u som har frekvens høyere enn $\omega = 10$, for eksempel på grunn av målestøy, med en faktor på minst 100. Vi ønsker dermed å finne en passende τ for dette.

Vi kan gjøre dette ved prøving og feiling med et Bode-plott, eller direkte ved å bruke at $A_r(\omega) = |P(j\omega)| = \left|\frac{1}{\tau j\omega + 1}\right| = \frac{1}{\sqrt{\tau^2\omega^2+1}}$. Siden vi jo ønsker $A_r(10) \leq \frac{1}{100}$, så har vi

$$\frac{1}{\sqrt{\tau^2 10^2 + 1}} \leq \frac{1}{100} \quad \Longrightarrow \quad 100\tau^2 + 1 \geq 100^2 \quad \Longrightarrow \quad \tau \geq \sqrt{(100^2 - 1)/100}.$$

Vi kan derfor for eksempel ta $\tau = 10$.

Modellering av lavpassfiltre implementert ved elektriske kretser



a) Første-ordens lavpassfilter. **b) Andre-ordens lavpassfilter.**

Figur 10.12: Eksempler på elektriske kretser tilsvarende første- og andre-ordens lavpassfiltere.

Eksempler på elektriske kretser som tilsvarer et først- og andre-ordens lavpassfilter (Sallen–Key-topologi) er vist i figur 10.12 (det finnes en rekke andre kretser for dette, såkalte topologier). Merk at man ved å koble disse i serie får et tredje-ordens filter. Legg også merke til at operasjonsforsterkenre ("opampene") antas som ideelle (altså uendelig inngangsmotstand og uendelig forsterkning), slik at de fungerer som spenningsfølgere hvor spenningen ut tilsvarer spenning på +-inngangen.

Ved å tak bruk i bruk Kirchoffs strømlov (se § 3.2.4) ved +-inngangen til opampen i første-ordensfilteret (altså **a)** i fig. 10.12), får vi

$$\frac{(v_{ut} - v_{inn})}{R} + C\frac{dv_{ut}}{dt} = 0 \quad \Longrightarrow \quad \frac{dv_{ut}}{dt} = -\frac{1}{RC}\left(v_{ut} - v_{inn}\right).$$

Tilsvarende overføringsfunksjon fra inngang til utgang med cut-off-frekvens/båndbredde $\omega_{bb} = 1/(RC)$ (ofte brukes ω_c i stedet) er dermed som følger:

$$P(s) = \frac{\omega_{bb}}{s + \omega_{bb}} = \frac{1}{\frac{s}{\omega_{bb}} + 1}.$$

La oss nå finne tilsvarende ligninger for andre-ordens filteret i figure 10.12-**b)**. Vi starter med å igjen bruke Kirchoffs strømlov i forgreningspunktet til høyre for R_1-motstanden hvor vi antar spenningen er lik v_1:

$$\frac{(v_1 - v_{inn})}{R_1} + \frac{(v_1 - v_{ut})}{R_2} + C_1\left(\frac{dv_1}{dt} - \frac{dv_{ut}}{dt}\right) = 0.$$

Ved å også ta i bruk strømloven ved +-inngagen til opampen, får vi $\frac{(v_{ut}-v_1)}{R_2} + C_2\frac{dv_{ut}}{dt} = 0$, hvorfra vi har at $v_1 = v_{ut} + R_2C_2\frac{dv_{ut}}{dt}$, og dermed $\frac{dv_1}{dt} = \frac{dv_{ut}}{dt} + R_2C_2\frac{d^2v_{ut}}{dt^2}$. Ved å sette dette inn i utrykket over får vi

$$\frac{(v_{ut} + R_2C_2\frac{dv_{ut}}{dt} - v_{inn})}{R_1} + \frac{(v_{ut} + R_2C_2\frac{dv_{ut}}{dt} - v_{ut})}{R_2} + C_1\left(\frac{dv_{ut}}{dt} + R_2C_2\frac{d^2v_{ut}}{dt^2} - \frac{dv_{ut}}{dt}\right) = 0.$$

Ved å kansellere leddene markert i rødt og så multiplisere på begge siden med R_1 får man

$$(v_{ut} + R_2C_2\frac{dv_{ut}}{dt} - v_{inn}) + R_1C_2\frac{dv_{ut}}{dt} + R_1C_1R_2C_2\frac{d^2v_{ut}}{dt^2} = 0.$$

11. Estimering og sensorfusjon*

Alternative kilder: Ebook av Brian Douglas; Video-serie av Brian Douglas

11.1. Sensorfusjon

Vi ønsker å måle en prosessvariabel, y. Vi har to sensorer tilgjengelig, som gir oss målingene[a] $y_m^1 = y + m_1$ og $y_m^2 = y + m_2$, hvor m_1 og m_2 er målestøyene til de to sensorene. F.eks. kan y_m^2 være en rask måling hvor m_1 har liten varians, men muligens varierende (drivende) middelverdi; mens y_m^1 er en tregere sensor med stor varians i støyen, men med stabil middelverdi. **Mål:** kombinere (fusjonere) målingene de to målingene til å oppnå en ny, bedre måling.

[a] Sensorene trenger ikke nødvendigvis måle den ønskede prosessvariabelen direkte, en indirekte måling hvorfra man kan estimere den er et vanlig scenario som man også kan dekkes av metodene vi skal se på.

11.1.1 Komplementærfilter

Gitt problemstillingen over (f.eks. y_m^1 er en IMU[1] og y_m^2 e en GPS-måling), så kan et *komplementærfilter* være en mulig strategi:

[1] Et vanlig bruksområde til et komplementærfilte er fusjonering av gyroskopmålinger i en IMU med dens akselerometermålinger.

Komplementærfilter: Gitt to målinger, $y_m^1 = y + m_1$ og $y_m^2 = y + m_2$, hvor m_1 og m_2 målestøyene. Ta, for en positiv τ_m,

$$\hat{y}_m = \frac{1}{\tau_m s + 1} y_m^1 + \frac{\tau_m s}{\tau_m s + 1} y_m^2$$

Dermed en kombinasjon av et lasspassfilter og en høypassfilter: Vi bruke lavpassfilteret til å beholde den stasjonære verdien fra den trege målingen y_m^1 målingen med stor varians, mens vi for den raske, men drivende målingen y_m^2 bare beholder den de høyfrekvente delene.

Hvorfor komplementær? Og forklare filterets orden.

11.2. Tilstandsestimering og Kalmanfilteret

Fun facts, bemerkninger og annet dill dall (you may skip)

Tracking av objekter fra kameraer

Kontinuerlig Kalman–Bucy-filter

Del VI

Maskinlæring og Optimering

12. Maskinlæring

I dette kapittelet skal vi ta en rask titt på et tema som er meget «hot» om dagen, nemlig en gren innen kunstig intelligens kalt maskinlæring. Målet er at du skal få en grunnleggende forståelse for hva det er, og ikke minst hvordan det kan brukes i automatiserings-relevante sammenhenger. Dette er egentlig et gigantsik felt, betsående av en rekke idéer og metoder, så vi skal snevre inn vårt fokus til en familie av metoder som inkluderer kunstig nevrale nettverk.

Merk at metodene vi skal se på i stor grad er basert på numeriske metoder, og ikke minst matematiske felt som lineær algebra og statistikk. I en ideell verden ville du derfor allerede hatt en litt bredere innføring innen disse temaene før dette kapittelet. På den annen side, så håper jeg dette kapittelet kan virke som en motivator for at du skal følge godt med i mattetimene.

12.1. Hva er maskinlæring? Og hva brukes det til?

Alternative kilder: Store norske leksikon.

Hvorfor skal du lære dette? 🤷‍♀️ Kunstig intelligens, og da spesielt maskinlærings-undergruppen *dyplæring*, er i ferdig med å drastisk endre ikke bare hverdagslivene våre, men også hvordan ingeniøryrket utføres. Vil du virkelig «stå igjen på perrongen» uten vite de mest grunnleggende delene av denne teknologien?

Noen problemer lar seg ikke løse vha. håndskrevne regler

Som tidligere nevnt, så kan automatisering tenkes på som problemet om å få systemer og prosesser til å gjøre som man ønsker «på egenhånd» (autonomt) basert på tilgjengelig informasjon. Dette kan for eksempel være et sekvensstyringsproblem som «Start transportbåndet hvis sensor A blir aktiv», altså en regel som bruker informasjon (data) fra sensor A. I kapittelet om PID-regulatoren og PID-tuning så vi også hvordan man ved hjelp av en matematisk modell av en prosess (f.eks. utledet fra førsteprinsipper) kunne bestemme et sett med regler, bestemt av regulatoren, for hva pådragsorganene skal gjøre for gitte målinger av prosessutgangen (informasjonen/dataen).

Det finnes dog en rekke problemer som enten er så komplekse og/eller krever behandling av så store mengder informasjon at det er uhensiktsmessig, kanskje til og med umulig, å sette seg ned og formulere et sett med regler som vil løse den ønskede oppgaven «for hånd».

Figur 12.1: Du ser nok umiddelbart hvor huskatten Tini er på dette bildet, og ikke minst at dette er en katt (🐈) og ikke en hund (🐕). Men klarer du å skrive et dataprogram for hånd som får en datamaskin til å gjøre det samme?

For eksempel, hvordan kan du skrive ned et sett med regler som gjør at man kan finne ut om det er én eller flere katter på et gitt bilde som det i figur 12.1, for ikke å snakke om *hvor* på bildet de er? Informasjonen vi har tilgjengelig er bildet, rettere sagt pikselverdiene, og et bilde fra en mobiltelefon har som regel flere millioner(!) piksler. Så selv om et slikt problem er en triviell sak for et menneske, så er det langt fra trivielt å lage et sett med regler (i kode) som definerer hva en katt er, hva som skiller den fra en hund, og hvordan man kan finne og ramme inn én eller flere eventuelle katter på et bilde.

Maskinlæring: La maskinen «lære» reglene selv fra data

Anta at man har et problem hvor det er vanskelig/praktisk umulig å skrive ned et sett med regler som lar en maskin løse det, men at man har store mengder relevant informasjon i en eller annen form for data[1]. Dette er et scenario hvor den beste løsningen kan være å la maskinen selv «lære» et sett med regler som best mulig løser oppgaven basert på den tilgjengelige dataene. Det er dette vi kaller maskinlæring (ofte forkortet som ML).

Moderne maskinlæring har gjort det mulig å løse problemer som tidligere ble ansett som ekstremt utfordrende (om ikke umulige) å takle ved tradisjonell programmering. De moderne metodene har blant annet evnen til å:

- Håndtere komplekse og store datasett som mennesker ikke kan bearbeide manuelt, ei eller lage tradisjonelle programmer som kan behandle disse på en effektiv måte.
- Generalisere kunnskap og erfaringer den har «lært» og bruke disese til å håndtere nye og ukjente situasjoner.
- Oppdage komplekse sammenhenger og mønstre i data som mennesker kanskje ikke er i stand til å oppdage eller formulere eksplisitt gjennom regler og logikk.
- Lære og forbedre seg kontinuerlig over tid uten behov for manuell omprogrammering.

[1] Terminologien stordata (eng. «big data») er noe man ofte hører i forbindelse med maskinlæring (og moderne teknologi for øvrig). Det er relatert til metoder og teknologier for innhenting, prosessering, lagring, analyse og bruk av store mengder data.

- Lære å løse problemer gjennom trening fra data i tilfeller der vi mennesker kanskje ikke en gang har klar innsikt eller forståelse av hvordan løsningen skal være.

Disse egenskapene gjør at maskinlæring har en rekke bruksområder innen automatisering og robotikk, deriblant:

- **Manipulasjon:** Maskinlæring kan hjelpe roboter med å lære å manipulere og håndtere (gripe) ulike objekter på en effektiv måte. Ved å analysere og forstå formen, størrelsen og materialet til objektene, kan maskinlæring bidra til å plukke og å flytte på myke og fleksible objekter med kompliserte geometrier.

- **Kvalitetskontroll:** Maskinlæring kan brukes til å utføre kvalitetskontroll i produksjonsprosesser. Ved å analysere data fra sensorer (som kameraer), kan roboter identifisere defekte eller avvikende produkter og ta nødvendige tiltak.

- **Feildetektering og vedlikehold:** Maskinlæring kan brukes til å forutsi feil og problemer i roboter og automatiserte systemer. Ved å analysere sensoriske data og historisk informasjon kan maskinlæring bidra til å identifisere mulige feil eller vedlikeholdsbehov, slik at man kan planlegge og iverksette riktig tiltak på riktig tidspunkt.

- **Navigasjon og autonome systemer:** Maskinlæring spiller en avgjørende rolle innen utviklingen av blant annet autonome kjøretøy og farkoster, som selvkjørende biler og droner. Ved hjelp av maskinlæring kan kjøretøyene lære å gjenkjenne objekter i omgivelsene sine som trafikkskilt og personer, unngå hindringer (og dermed kollisjoner), forstå veiforhold, navigere trygt, og ta beslutninger i sanntid basert på sensordata.

- **Adaptive og selvjusterende regulatorer:** Maskinlæring kan brukes til å få regulatorer til å justere seg selv, ved å forbedre sine justeringsvariabler for å bedre ytelsen (i en eller annen forstand) basert på data fra prosessen i lukket sløyfe. Metodene i seksjon 5.2, og da spesielt Åstrøms «auto-tuning» relé-metode, er enkle eksempler på slike strategier.

- **Systemidentifikasjon og data-drevet modellering:** Maskinlæring, spesielt kunstige nevrale nettverk, kan brukes til systemidentifikasjon og data-drevet modellering av dynamiske systemer. Ved å lære sammenhenger mellom inn- og utgangsdata, kan det utvikle en modell som kan forutsi systemets respons på nye innganger eller estimere gitte systemparametere. Dette er nyttig for både analyse og regulering, og da spesielt for ulineære systemer hvor det er utfordrende å lage en god modell fra f.eks. førsteprinsipper.

Legg her merke til at flere av disse applikasjonene er relatert til maskinlærings-basert maskinsyn.

Kunstig intelligens vs Maskinlæring vs Dyplæring

I tillegg til maskinlæring, så har du sikkert hørt om både kunstig intelligens (KI) og såkalt dyplæring. Selv om disse utrykkene ofte brukes synonymt blant "plebsa", så er KI en samlebetegnelse av flere metoder og algoritmer, deriblant maskinlæring, mens dyplæring igjen er en undergruppe av metoder innen maskinlæring. Denne sammenhengen er vist i figur 12.2, hvor notasjonen $A \supset B$ betyr at B er en del-/under-mengde av A.

Kunstig intelligens (KI) er et svært bredt begrep som inkluderer metoder innen rekke felter, med det til felles at de løser (spesifikke) problemer som normalt krever menneskelig

Figur 12.2: Kunstig intelligens (KI) omfatter et svært vidt og mangfoldig spekter av forskjellige felt og metoder, deriblant metodene som faller under kategorien maskinlæring (ML). Innen ML, så er kunstig nevrale nettverk spesielt populære i disse tider, og da spesielt de dype nettverkene som inngår i såkalt dyplæring (DL).

intelligens. Ikke bare maskinlæring, men også PID-regulatoren og PID-tuning og algoritmer relatert til Sekvens- og Datastyring er eksempler på KI. Se for eksempel denne YouTube-videoen (XFZ-rQ8eeR8) for en fin oversikt.

Maskinlæring (ML) er som tidligere nevnt en samlebetegnelsen på metoder som lar (data-)maskiner lære og forbedre seg selv basert på data og erfaringer fra interaksjoner med et miljø.

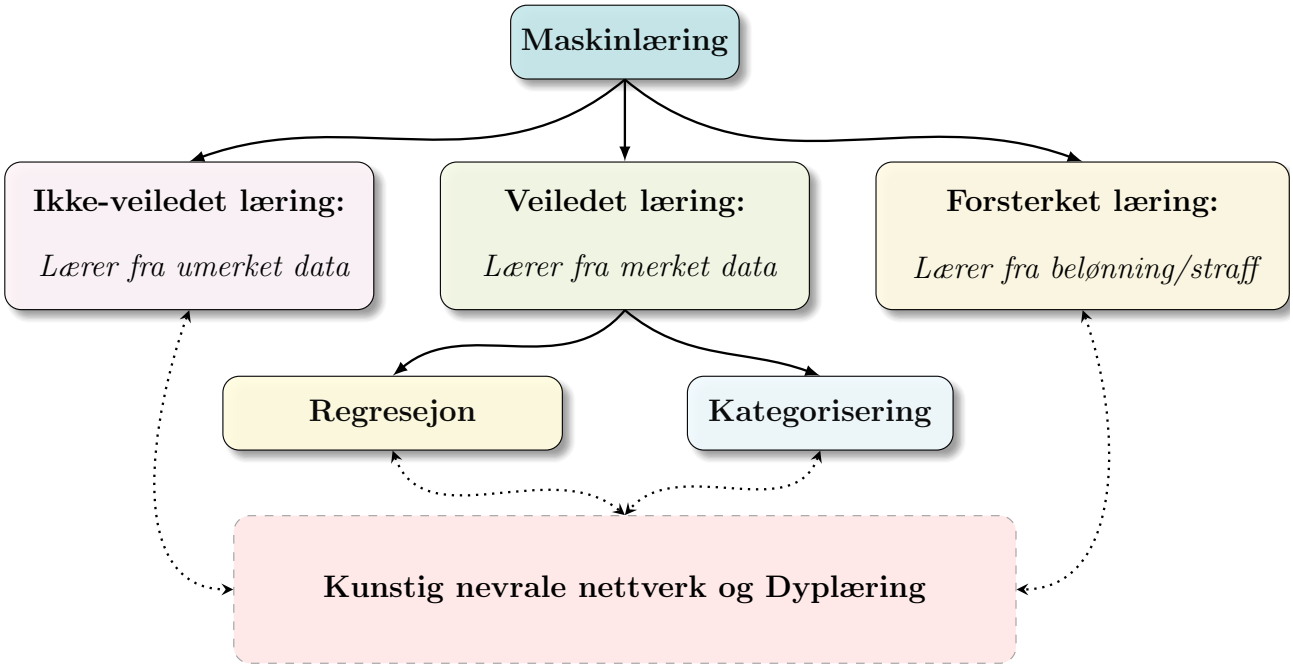
Dyplæring er igjen en gren av maskinlæring som bruker Kunstige nevrale nettverk med flere såkalte skjulte lag for å lære og trekke ut høyere nivåer av abstrakte representasjoner fra data. Dyplæring er en kraftig tilnærming innenfor maskinlæring og har vært avgjørende for mange av de imponerende prestasjonene innen bl.a. talegjenkjenning og naturlig språkbehandling, samt bildegjenkjenning og -generering (se figur 12.3) som har dukket opp de senere årene.

Oppgave 12.1. Spørsmål: I desember 2006 møttes verdens ledende maskinlæringseksperter til en konferanse. Der lagde de sammen en liste over de daværende 10 viktigste og mest innflytelsesrike data-mining- og maskinlærings-algoritmene (se [Wu et al., 2008]). Hvilken algoritme/metode tror du tronet øverst på denne listen? Svaret får du i seksjon 12.3.

Figur 12.3: AI-genererte (DALL-E) illustrasjoner av Tom i Tallinjeveien.

Fun facts, bemerkninger og annet dill dall (you may skip)

Trenger vi å frykte kunstig intelligens? Ja, kanskje. Men trolig ikke pga. de grunnen



Figur 12.4: Et (ikke helt utfyllende) kart over maskinlæring. Det kan til en viss grad deles opp i tre hovedkategorier: ikke-veiledet læring, veiledet læring og forsterket læring. Vi skal i disse notatene hovedsakelig fokusere på veiledet læring, mer spesifikt på underproblemene regresjon og kategorisering. Det finnes en rekke forskjellige maskinlæringsmetoder innen de forskjellige kategoriene, deriblant metoder basert på kunstig nevrale nettverk. Læringsmetoder basert på dype nevrale nettverk, såkalt dyplæring, har (med rette) fått spesielt mye fokus i senere år. Se f.eks. også følgende for et mer detaljert kart: https://github.com/trekhleb/homemade-machine-learning

du tror, slik som denne TED-talken på YouTube (TRzBk_KuIaM) så godt illustrerer.

Typer maskinlæring: veiledet-, ikke-veiledet- og forsterket læring

Som vist i figur 12.4, så kan maskinlæring grovt deles inn i følgende tre hovedkategorier:

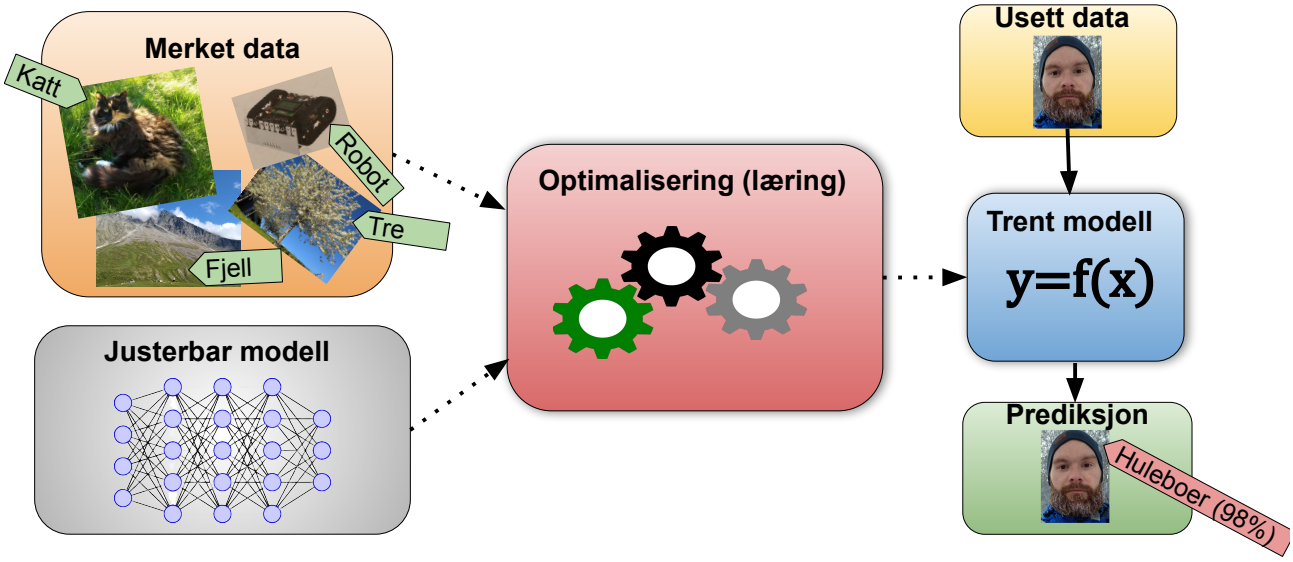
- **Veiledet læring:** (eng. Supervised learning) Basert på tilgjengelig **merket data**, altså kjente sammenhenger hvor gitte innganger tilsvarer gitte utganer, så er målet å lære en modell som kan generalisere til nye, usette innganer og forutsi deres utganger nøyaktig. Eksempler inkluderer bildeklassifisering, naturlig språkbehandling og regresjonsanalyse.

- **Ikke-veiledet læring:** (eng. Unsupervised learning) Gitt (helst store mengder) **umerket data**, så ønsker man å finne strukturer og mønstre i disse. Dette kan gjøres ved å gruppere inngangsverdiene i klynger eller på andre måter. Ikke-veiledet læring kalles derfor også noen ganger for *selv-veiledet læring*, siden dataene som er tilgjengelig fungerer som selve veiledningsmekanismen. I motsetning til veiledet læring, har ikke-veiledet maskinlæring fortsatt en lang vei å gå for å matche menneskelig ytelse på dette området.

- **Forsterket læring:** (eng. Reinforcement learning) En agent (tenk robot 🤖) trener sin beslutningsprosess ved å interagerer med et miljø. Forsterket læring blir derfor ofte ref-

erert til som «learning by doing». Treningen baserer seg på belønning og/eller straff i forhold resultatene av en gitt beslutning, eller en sekvens av beslutninger. Dette styrker ønsket oppførsel, uten å spesifisere hvordan oppgaven skal løses. Forsterket læring fungerer bra når det er mange veier til målet, og man ikke vet hvilken som er den beste. Siden denne læringsformen til en viss grad er basert på tidkrevende prøving-og-feiling, så foregår ofte selve læringsprosessen i en simulert verden. Forsterket læring sies også ofte å være en form for *delvis-veiledet læring*, siden man bruker en form for merket data, nemlig belønning og/eller straff, men disse er ofte svært forsinket i forhold til beslutningen som blir tatt, samt ofte er svært sporadiske og/eller sjeldne. Noen kjente suksesshistorier hvor forsterket læring har blitt tatt i bruk er DeepMinds AlphaGo og AlphaZero, den firbeint roboten ANYmal, samt OpenAIs fingernemme robothånd. Forsterket læring har også dype bånd til feltet optimal regulering innen reguleringsteknikken.

Vårt fokus i disse notatene: Veiledet læring er nok den ekleste å forstå av disse, og det er derfor det vi hovedsakelig skal holde oss til i disse notatene.



Figur 12.5: Illustrasjon av prosessen i veiledet maskinlæring, fra merket data og en bestemet justerbar modell, til en ferdig trent og generaliserende modell etter maskinlæringsprosessen.

12.2. Veiledet læring

Hovedkonseptet bak veiledet læring er illustrert i figur 12.5. Vi ønsker å bruke et (stort) sett med **merket data** til å finne et sett av parametere i en modell (f.eks. et kunstig nevralt nettverk) slik at den trente modellen vi har fått etter læringsprosessen er i stand til generalisere og gjøre gode prediksjoner[2]

[2]Strengt tatt bruker man kun ordet «prediksjon» når man faktisk er ute etter å predikere noe, altså forutsi et «resultat» gitt de kjente forholdene (inngangsdataene). Når man f.eks. ønsker å finne ut om det er en katt

Veiledet læring kan igjen deles inn i to hovedkategorier:

- **Kategorisering/klassifikasjon**: Kategorisering brukes når målet er å forutsi diskrete kategorier eller klasser. Det innebærer å tilordne inngangsvariablene (f.eks. pikslene i et bilde) til forhåndsdefinerte kategorier eller klasser basert på lærte mønstre i treningsdataene. For eksempel kan kategorisering brukes til å finne hunder eller katter i bilder, til å klassifisere e-postmeldinger som spam eller ikke-spam, samt forutsi om en pasient har en bestemt sykdom basert på symptomer og testeresultater.
- **Regresjon/prediksjon:** Regresjon brukes når målet er å predikere kontinuerlige numeriske verdier. Grovt forklart, så innebærer det å finne en sammenheng mellom et sett av inngangsvariabler og utgangsvariabler. Målet er å trene en regresjonsmodell ved hjelp av tidligere kjente forhold mellom inngagenene og utgangene slik at det også gir en god prediksjon for tidligere usett data. For eksempel kan regresjon brukes til å forutsi salgsinntekter basert på reklameutgifter, eller forutsi boligpriser basert på egenskaper som størrelse, beliggenhet osv. Den enkleste formen for dette er lineær regresjon.

12.2.1 Lineær regresjon

Alternative kilder: Wikipedia; Store norske leksikon.

Den formen for maskinlæring som kanskje er enklest å forstå er lineær regresjon, hvor målet er å finne et lineært forhold mellom et sett av kontinuerlige innganger og utganger.

Fun facts, bemerkninger og annet dill dall (you may skip)

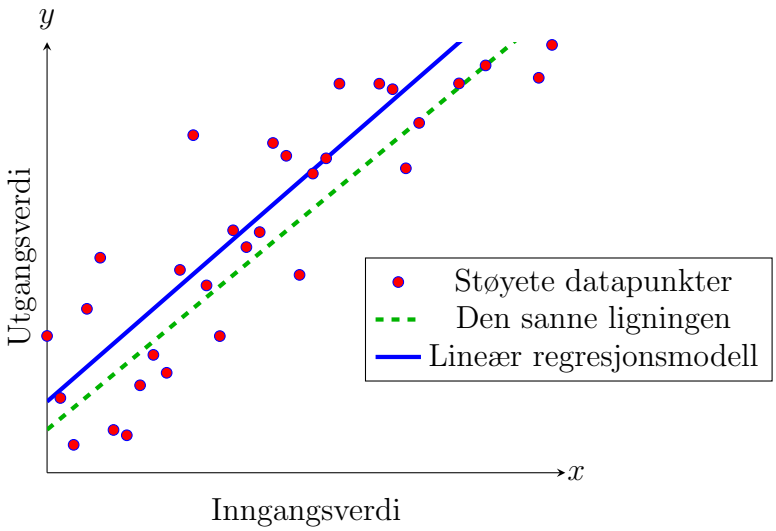
Utrykket "regresjon" stammer fra Francis Galton, som prøvde å finne en modell som predikerte menns høyde basert på deres fedres høyde. Han la merke til at selv om høyden til mennene ofte var ganske nærme fedrenes høyde, så ville de som oftest også gå litt i retning av gjennomsnittshøyden til menn; med andre ord så *regresserte* høyden til mennene mot gjennomsnittet, slik at høye fedre har en tendens til å ha sønner som er kortere enn seg selv men høyere enn gjennomsnittet for øvrig, og vice versa.

Et eksempel på lineær regresjon med én inngangsvariabel, x, og en utgangsvariabel, y, er vist i figur 12.6 (eksempelet ble generert i MATLAB vha. kodesnutt 12.1). Problemet vi der er ute etter å løse er som følger:

Anta at vi vet et sett av diskrete datapunkter, merket rødt i figur 12.6, som gir et visst forhold mellom inngangen x og utgangen y. Vi ønsker så å finne en lineær ligning $y = ax + b$, rettere sagt de to parameterne a og b, som best mulig (i en eller annen forstand) passer disse dataene. En annen måte å tenke på dette problemet er som følger: Vi er ute etter å gjenskape en «sann ligning», som vi tror er på formen $y = ax + b$, fra et sett av n «støyete målinger», gitt som diskrete datapunkter (y_i, x_i), $i = 1, 2, \ldots, n$, der y_i kan tenkes på som å inneholde noe «målestøy». Siden det er litt målestøy for hvert par av datapunkter, så vil ikke den lineære modellen passa alle datapunktene nøyaktig, slik at vi generelt sett kun har

$$y_i = ax_i + b + \epsilon_i$$

på et gitt bilde, så trenger man jo ikke predikere noe (enter er det eller så er det ikke en katt på bildet). I slike tillfeller kaller man det i ML-lingo heller for «inference», som bare betyr å trekke en slutning (katt på bildet, ja eller nei?).



Figur 12.6: Illustrasjon av lineær regresjon: Den stiplede, grønne linjen viser det reelle forholdet mellom inngangen, x, og utgangen, y, via $y = ax + b$. Ved tilgang på støyete målinger, gitt som par (y_i, x_i) merket ved røde sirkler, kan man ved hjelp av lineær regresjon (se kodesnutt 12.1) lag en tilnærmet modell, vist i blått.

hvor ϵ_i er *residualet* (eventuelt feilen eller avviket) til modellen for dette datapunkt paret.

En måte å finne a og b som passer dataene er ved hjelp av minste kvadraters metode:

$$J = \min_{a,b} \sum_{i=1}^{n} (y_i - ax_i - b)^2 = \min_{a,b} \sum_{i=1}^{n} \epsilon_i^2 = \min_{a,b} \left((\hat{y}_1 - ax_1 - b)^2 + (y_2 - ax_2 - b)^2 + \cdots \right)$$

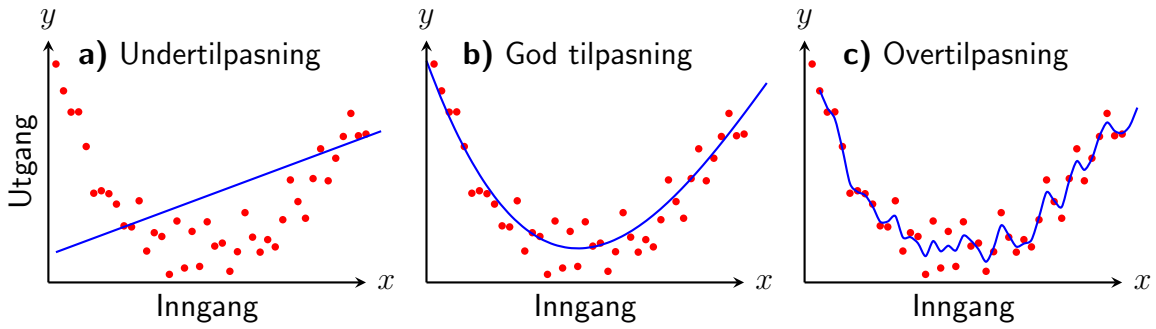
Vi sier at de parameterverdiene a og b som minimerer verdien J—altså som gjør at J har minst mulig verdi—er *optimale*, i den forstand at de minimaliserer summen av kvadratene i til residualene, altså $J = \sum_{i=1}^{n} \epsilon_i^2$. Dette kalles for et (ubegrenset) *optimeringsproblem*, og er noe vi skal se nærmere på i kapittel 13 om optimering.

Kodesnutt 12.1: Eksempel på lineær regresjon i MATLAB.

```
n=40; % Antall datapunkter
X=linspace(0,10,n)'; % Inngangsverdier
Yt=X+1;  % Den sanne ligningen
rng(24); % Sette "seed" til randon number generator (randn)
Y=X+2*randn(length(X),1)+1; % Maalte verdier med stoey
a = regress(Y,[X,ones(length(X),1)]) % Finne koeffisientene
% a =pinv([X,ones(length(X),1)])*Y; % Alternativ maate
Yr=a(1)*X+a(2); % Lineaer regresjonsmodell
```

12.2.2 Over- og undertilpasning

Innen maskinlæring står man ofte overfor to store utfordringer som kan påvirke modellens ytelse: undertilpasning og overtillpasning. Begge disse «tilstandene», som er illustrert i figur 12.7, sier noe om hvor godt modellen kan generere nøyaktige prediksjoner basert på dataene den har



Figur 12.7: Illustrasjon av forskjellige tilpasningsgrader: **a**) En lineær modell er ikke fleksibel nok til å få en god representasjon av de underliggende dataene, noe som fører til undertilpasning; **b)** En kvadratisk modell gir i dette tilfelle god tilpasning; **c)** Fenomenet "overtilpasning" kan oppstå når en fleksibel modell blir trent for lenge slik at den tilpasser treningsdataene i for stor grad, noe som gjør at den trente modellen trolig ikke vil generalisere godt til nye, usette data.

lært fra, samt hvor godt den lar seg generalisere til nye, usette data. I hovedsak handler maskinlæring om å finne den riktige balansen mellom undertilpasning og overtillpasning for å oppnå en modell som kan generalisere godt til nye data.[3]

Undertilpasning (eng. «underfitting») skjer når en modell enten er for enkel til å lære den underliggende strukturen i dataene, eller ikke blir trent tilstrekkelig lenge eller med nok data. For eksempel, hvis vi prøver å passe en lineær modell til data generert av en ulineær prosess slik som i figur 12.7-**a**, vil vi sannsynligvis oppleve undertilpasning, i den forstand at modellen våre ikke er en god representasjon av de underliggende dataene. En undertilpasset modell har rett og slett ikke nok kompleksitet og fleksibilitet til å fange alle mønstre og nyanser i treningsdataene, og som et resultat presterer den dårlig på både treningsdataene og usett data. Undertilpasning kan løses enten ved å øke modellens kompleksitet, for eksempel ved å bruke en mer kompleks modell, og/eller trene modellen lengre vha. mer og/eller bedre data.

Overtilpasning (eng. «overfitting») skjer når en modell er for kompleks og lærer for mye fra treningsdataene. Dette fenomenet er illustrert i figur 12.7-**c**, hvor man kan se at modellen er i overkant tilpasset datapunktene, slik at det faktisk også har tilpasset seg noe av støyen i målingene, samt eventuelle ekstremverdier (eng. «outliers»), altså data som skiller seg vesentlig ut fra resten av datasettet. Overtilpassede modeller har derfor en tendens til å ha veldig lav feil på treningsdataene (treningssettet), men høy feil på nye, usette data (testsettet). Dette skyldes at den overtilpassede modellen har blitt for godt tilpasset til treningsdataene og derfor ikke generelt lar seg generalisere godt til nye data. Overtilpasning kan forhindres ved å introdusere regularisering (straff for modellkompleksitet), samle mer treningsdata, eller ved å bruke teknikker som *kryssvalidering* for å få en mer pålitelig vurdering av modellens prestasjon.

12.2.3 Kryssvalidering

Kryssvalidering (eng. «cross-validation») er en teknikk som brukes innen maskinlæring for å vurdere hvor godt en modell vil generalisere til nye data. Den er spesielt nyttig for å finne en

[3]For å evaluere ytelsen til en modell, spesielt mtp. på under- og overtilpasning, så deler man ofte all tilgjengelige merket data opp i to deler: 1. **treningssettet**, altså den delen som skal brukes i læringsprosessen; og 2. **testsettet**, som da brukes til å se hvor godt modellen yter og generaliserer etter treningen. Det er også vanlig å ta med et (varierende) **valideringssett**.

god balanse mellom under- og overtillpasning.

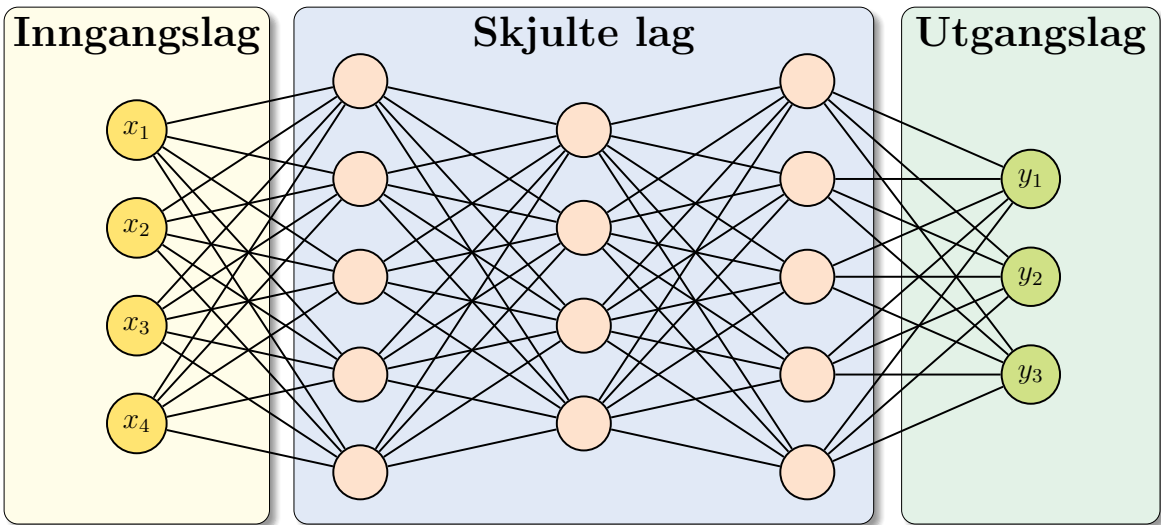
Den mest kjente formen for kryssvalidering er k-fold kryssvalidering. Her er det generelle konseptet at du deler ditt originale datasett inn i k like store deler eller «folder». Du trener så modellen k ganger, hver gang med en annen fold holdt tilbake som et valideringssett, og resten av dataene brukes som treningsdata. Etter at du har gjennomført denne prosessen k ganger, tar du gjennomsnittet av resultatene for å få den endelige modellens ytelse.

Kryssvalidering gir en mer robust vurdering av modellens ytelse, siden den benytter hele datasettet for både trening og validering. Det reduserer også risikoen for at modellen din presterer unormalt godt eller dårlig på grunn av en tilfeldig oppdeling av data. Følgende sitat fra [Brunton and Kutz, 2022] fanger viktigheten av kryssvalidering godt:

«If you don't cross-validate, you is dumb.»

12.3. Kunstige nevrale nettverk

Alternative kilder: YouTube-video av 3Blue1Brown



Figur 12.8: Illustrasjon av et dypt, foroverkoblet kunstig nevralt nettverk, med full sammenkobling mellom lagene.

Du har helt sikkert hørt om kunstige nevrale nettverk. Disse nettverkene er bærebjelken innen generative KI-metoder, som DALL-E og Midjourney, samt innen store språkmodeller, som GPT-serien til OpenAI og den nå allment kjente ChatGPT.

Svaret på oppgave 12.1: I starten av dette kapittelet ba jeg deg gjette hvilken metode som av verdens ledende maskinlæringseksperter ble rangert som den viktigste i en liste på 10 maskinlærings-metoder. Gjettet du kunstig nevrale nettverk? Ta tok du i så fall feil, for kunstig nevrale nettverk var ikke på topp-10 listen[a] en gang!

[a]Listen var som følger: 1. C4.5, 2. k-means algoritmen, 3. Support vector machine (SVM), 4. Apriori algoritmen, 5. EM algortimen, 6 Page rank (altså Google søk), 7. AdaBoost, 8. k-nearest neighbor klassifisering (KNN), 9. Naive Bayes, 10. CART.

I disse dager hadde nok kunstig nevrale nettverk som en generell kategori av metoder toppet en liste slik som den nevnt i boksen over. Så hva har endret seg siden 2006? Det har for eksempel ingenting å gjøre med at kunstig nevrale nettverk er en ny teknologi, for det er det absolutt ikke; det har blitt forsket aktivt på slike nettverk i hvert fall så langt tilbake som 1970-tallet.

Dyplæringseksplosjonen: Grunnene til hva som har forutsaket eksplosjonen i innen denne typen maskinlærings-metoder de senere årene er nok noe sammensatt, men de kanskje tre viktigste faktorene er følgende:

- Bedre tilgang på store mengder data: Om enn en litt rar analogi, så sies det jo ofte at «data er den nye oljen». Standard datasett (som f.eks. de på Kaggle) har også gjort det lett å sammenligne forskjellige ideer og arkitekturer.
- Nye arkitekturer: Nye ideer for hvordan man kan bruke, bygge opp og trene slike nettverk, deriblant konvolusjonelle nevrale nettverk og transformere (se også figur 12.12), har ført til store økninger i ytelse innen rekke felter.
- Mer datakraft (til en «rimelig» penge): For både å håndtere den store mengden data, samt trene nettverk av økende kompleksitet, har både ytelsen og den noe lavere kostnaden til moderne GPUer gjort det mulig for både store selskaper å trene sine gigantiske nettverk og for studenter til å trene mindre, men allikevel imponerende, nettverk vha. av gaming-PCen sin på hybelen. Grunnen er at GPUer tillater parallelle utregninger.

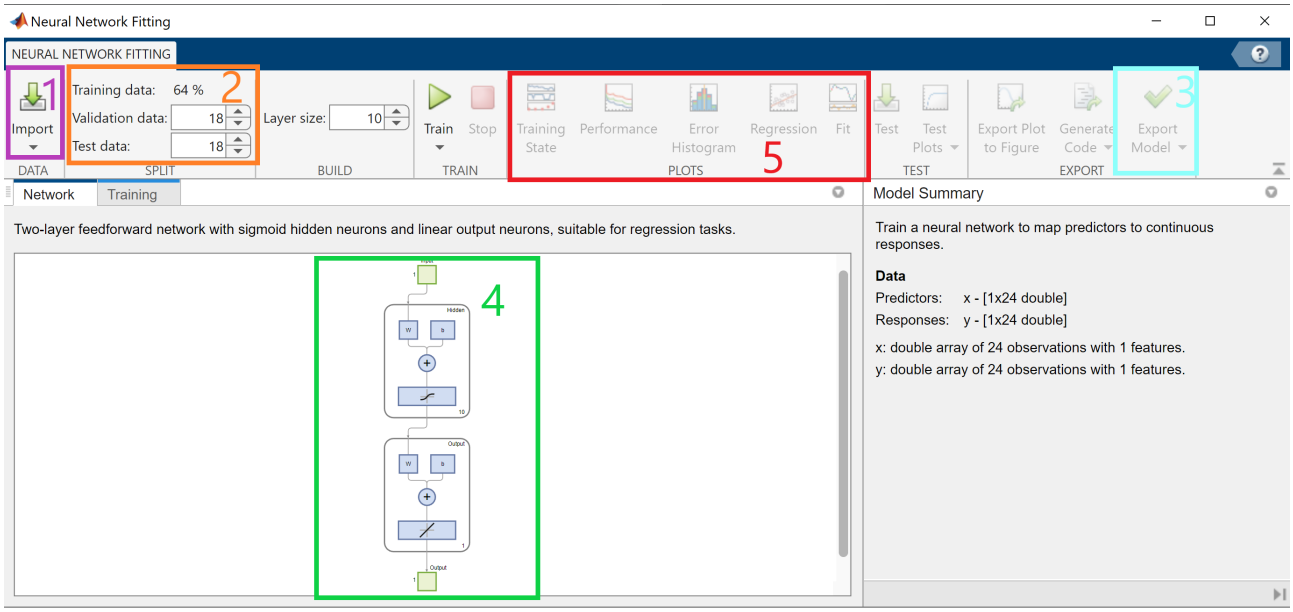
Andre faktorer inkluderer trender innen open-source deling av kode og programvare, samt hyppig open-access deling av forskning via såkalt preprints i arkiver som arXiv. Muligheten for «plug-and-play/off-the-shelves» maskinlæring ved hjelp av programvare som MATLAB, og i enda større grad Python (pga. bl.a. TensorFlow og PyTorch), samt skytjenester (AWS, GCP, Microsoft Azure) og effektive metoder for hyperparameter-tuning har også vært ekstremt viktig.

Tilsammen har dette gjort det mulig å lage og trene langt større nettverk enn det som var mulig bare få år tilbake. For eksempler har OpenAIs sitt bildegenereringsnettverk DALL-E 2 rundt 3.6 milliarder parametere; Goggle Imagen har 4.6 milliarder; mens OpenAIs GPT-4 sies å ha over 100 billioner! 😱

12.3.1 Oppbyggingen til et fremoverkoblet kunstig nevralt nettverk

Den mest grunnleggende *arkitekturen* et slikt nettverk kan ha, er et full- og foroverkoblet nettverk slik som det som er vist i figur 12.8. Det består av et inngangslag som blir sendt videre til et sett med skjulte lag, før det avsluttes med et utgangslag. Vi sier at et slikt nettverk er *dypt*, som i dyplæring, hvis det er mer enn ett skjult lag. Nettet sies å være foroverkoblet/-matet siden alt blir forplantet fremover i nettet (det er ingen tilbakekobling), samt fullkoblet siden hver node i et lag er koblet til alle nodene i det neste laget.

Vi har faktisk allerede såvidt sett på en slik type nettverk, nemlig lineær regresjon. I lineær regresjon avhenger utgangen av et lag lineært på inngangen, noe som betyr at skjulte lag ikke er av noen betydning. La oss ta et enkelt eksempel for å illustrere dette: La inngang til en lineær regresjonsmodell være x, la $z = ax + b$ være et «skjult» lag, og la $y = cz + d$ være utgangen. Vi har jo da at $y = c(ax + b) + d = cax + cb + d = \alpha x + \beta$ med $\alpha = ca$ og $\beta = cb + d$.



Figur 12.9: Neural Network Fitting-appen i MATLAB hvor man blant annet kan: 1. laste inn merket data; 2. fordele datasettet i trenings-, valideringss- og test-sett som ønkset; 3. eksportere det ferdigtrente nettverket til f.eks. Simulink; 4. endre nettverkstrukturen og aktiveringsfunksjoner for et grunt nettverk; og 5 analysere det trente nettverket.

Lineære nettverk er derfor aldri dype, og de har naturlig nok svært begrenset prediksjonskraft for problemer som ikke godt kan modelleres ved hjelp av en lineær sammenheng. Det viser seg dog at hvis man sper på med en liten dråpe ulineæritet, i form av såkalte *aktiveringsfunksjoner*, så kan plutselig slike dype nettverk i teorien være universelle approksimatorer, noe som betyr at de kan approksimere en hvilken som helst kontinuerlig sammenheng mellom inngangene og utgangene! 🦾

Før vi går løs på aktiveringsfunksjoner, la oss demonstrere hvordan man kan bruke denne egenskapen, altså at nevrale nettverk er universelle approksimatorer, til å tilnærme en ulinearitet i et reguleringssystem vha. Neural Net Fitting-appen i MATLAB (se figur 12.9):

Eksempel 12.1. (**Tom vil teste ML: Regulator med ulinearitets-kansellering basert på NN**) Gitt følgende første-ordens system:

$$\dot{x} = u + f(x).$$

Her betegner x hastigheten (i km/t) til en bil, u er pådraget, mens $f(\cdot)$ representerer en ukjent ulinearitet.

Målet er å designe en regulator for aktiv crusiekontroll som gjør at bilen kjører i 50 km/t.

Problem: Med kun en ren P-regulator, $u = k_P(r - x)$, vil ulinearitetene kunne føre til et stort stasjonært avvik. Dessverre er systemet vi skal implementere regulatoren på ganske så kjipt, i den forstand at den ikke kan lagre tidligere tilstands- og pådragsdata. Dette gjør det umulig å implementere *dynamiske* reguleringsstrategier (slik som I-leddet i en PI-regulator).

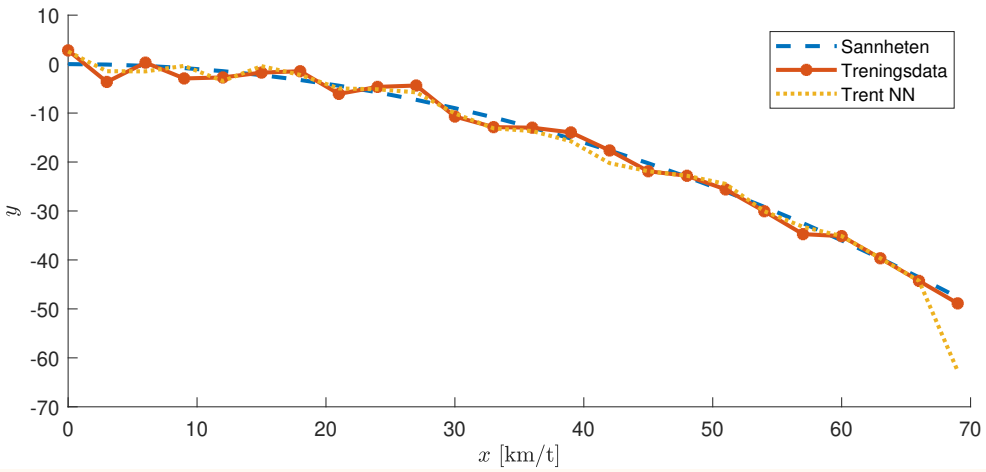
Løsningen vi skal se på er å tilnærme ulinearitetene f vha. et kunstig nevralt nettverk, $\hat{f}$,

slik at vi kan (delvis) kompensere for f med den *statiske* og ulineære regulatoren:

$$u = k_P(r - x) - \hat{f}(x).$$

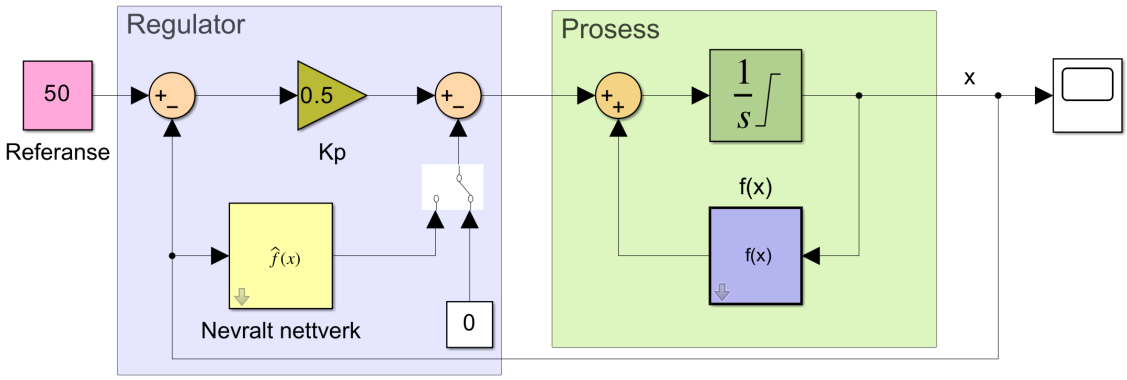
Prosedyren for å gjennomføre dette er som følger:

1. Datainnsamling: Ved å manuelt kjøre bilen slik at den holder forskjellige konstante hastigheter (og dermed $\dot{x} = 0$) kan vi ved å lese av tilsvarende u finne et estimat av $f(x)$ via $f(x) = -u + v$, hvor v tilsvarer eventuelle målefeil og -støy. De røde sirklene i plottet under viser de 24 datapunktene som ble samlet inn:

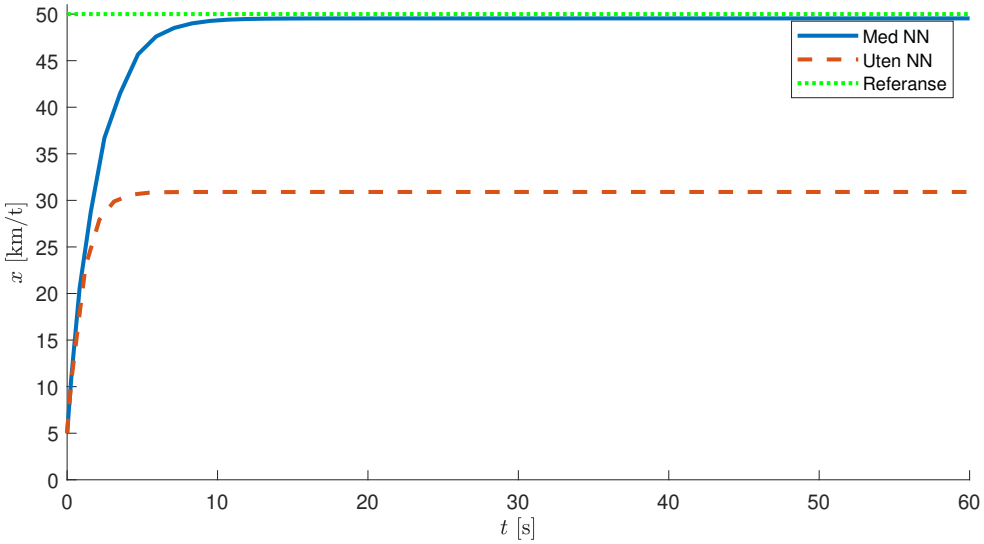


2. Trening av nettverket: Et grunt (altså kun ett skjult lag) kunstig nevralt nettverk ble så trent vha. Neural Net Fitting-appen i MATLAB med innstillingene vist i figur 12.9.

3. Simulering i Simulink: Det trente nettverket ble eksportert til Simulink modellen vist under:



4. Resultater: Responsen med og uten kanselleringen er vist i plottet under:



Ta også å tenk litt på (eller enda bedre, test ut selv) følgende:

1. Hva skjer med regulatoren hvis man beveger seg utenfor området hvor vi samlet data?
2. Nettverket bruker en såkalt *aktiveringsfunksjon* av typen sigmoid (vi skal se mer på dette i neste seksjon). Jeg vil påstå at for dette eksempelet/bruksområdet så er det et mye bedre valg enn f.eks. den kanskje generelt mer populære ReLU. Hva tror du jeg baserer denne påstanden på? Det kan her være lurt å ta en titt på figur 12.11.

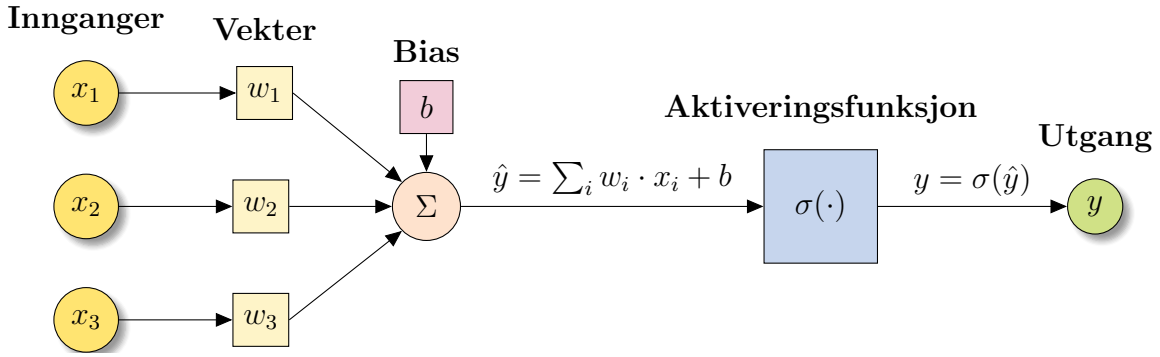
12.3.2 Aktiveringsfunksjoner: En liten dråpe ulinearitet

Alternative kilder: Wikipedia.

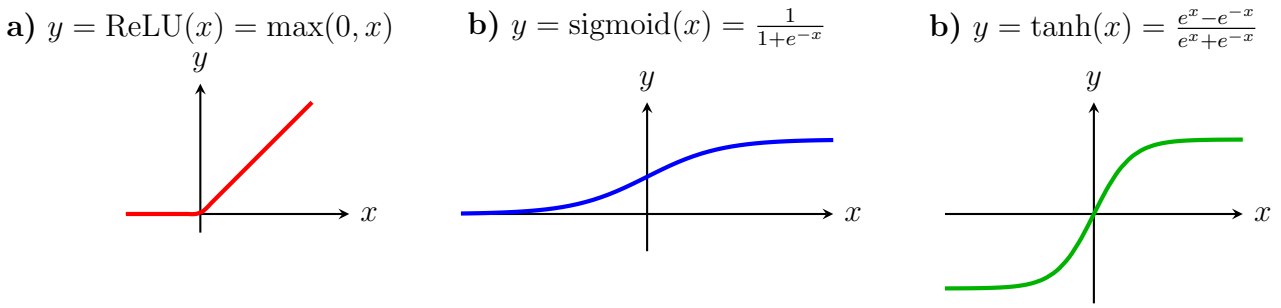
Aktiveringsfunksjoner er en viktig komponent i kunstige nevrale nettverk, da de bestemmer utgangen til en gitt node (nevron) i nettverket. Aktiveringsfunksjoner gir nettverket evnen til å lære og tilpasse seg komplekse ulineære sammenhenger mellom innganger og utganger. Uten en ulineær aktiveringsfunksjon vil et kunstig nevralt nettverk tilsvare lineær regresjon, og vil dermed ikke være i stand til å modellere mer komplekse sammenhenger mellom inn- og ut-dataene. Det er aktiveringsfunksjonen, kombinerte med flere skjulte lag, som gjør slike nettverk stand til å generalisere og håndtere komplekse data.

En illustrasjon av et nevron/node med en aktiveringsfunksjon $\sigma(\cdot)$ er vist i figur 12.10. Hvert slikt nevron i et nettverk tar inngangene og ganger dem med et sett av vekter før man legger dem sammen. Man legger så til et bias, og bruker en aktiveringsfunksjon for å generere nodens utgang. Utgangene i et gitt lag blir innganger i det neste laget, og slik fortsetter det helt til utgangslaget.

Det finnes mange forskjellige aktiveringsfunksjoner som kan brukes i kunstige nevrale nettverk, og valget av aktiveringsfunksjon avhenger ofte av den spesifikke oppgaven og strukturen til



Figur 12.10: Blokkdiagram som illustrerer virkemåten til en enkelt node (nevron) i et kunstig nevralt nettverk: Inngangene blir først vektet, så summert og lagt sammen med en gitt bias. Dette gir et tall, $\hat{y}$, som blir sendt igjennom en aktiveringsfunksjon, $\sigma(\cdot)$, for å få nodens utgang $y = \sigma(\hat{y})$. .



Figur 12.11: Vanlige aktiveringsfunksjoner:

nettverket. Tre populære aktiveringsfunksjoner, vist i figur 12.11, er sigmoid, ReLU (Rectified Linear Unit), og tanh (hyperbolsk tangent):

- Sigmoid: Sigmoid-funksjonen er en klassisk aktiveringsfunksjon som etterligner nevronene i hjernen. Den tar verdier mellom 0 og 1 i form av en S-kurve, og er dermed egnet til problemer relatert til sannsynlighetsfordelinger og logistisk regresjon.
- Hyperbolsk tangent: En annen S-kuve kan genereres vha. den hyperbolske tangentfunksjonen ($\tanh(\cdot)$). Denne er lik sigmoid-funksjonen, men tar verdier mellom om -1 og 1. Dette gjør den egnet til klassifisering; f.eks. er det en katt (🐱=-1) eller en hund (🐶=1)?
- ReLU: ReLU («rectified linear unit») er den aktiveringsfunksjonen som kanskje er brukt mest i disse dager. Det fiffige er at den er stykkvis lineær! Altså, den er null for negative innganger og en lineære linje for positive. En av grunnen til at den er så populær, i tillegg til at den er så enkel da selvsagt, er at den er mindre utsatt for det såkalte «vanishing gradient problem» enn f.eks. Sigmoid og tanh.

Se også Wikipedia for en mer utfyllende liste, som også inkludere de svært mye brukte funksjonene leaky-ReLU (som delvis fikser det såkalte «dying ReLU problem») og softmax.

Tygg litt på denne: ReLU er en svært populær aktiveringsfunksjon, men har (i hvert fall etter min mening) er stort problem. Hva tror du det er? Og kan du tenke deg noen

alternativer som har temmelig lik form som ikke har dette problemet?

12.3.3 Treningsprosessen: Hvordan en maskin «lærer»

Når vi sier at en maskin lærer, så tenker vi egentlig på treningsprosessen. Denne treningsprosessen er en form for optimeringsprosess, hvor vi prøver å finne de «beste» nettverkparameterne, i den forstand at minimaliserer en såkalt tapsfunksjon (eng. «loss function»). Innen mer generell optimering som vi skal se litt mer på i kapittel 13, så er dette et ubegrenset optimeringsproblem hvor tapsfunksjonen da som oftest kalles *målfunksjonen*.

Stegene i en enkel treningsprosess er som følger:

Eksempel på stegene i prosessen for å trene et nevralt nettverk fra scratch:

1. **Initialisering:** Først opprettes det et nettverk med tilfeldige vekter og biaser (eventuelt brukes verdier fra et tidligere trent nettverk, såkalt treningsoverføring). Nettverket består av et inngangslag, ett eller flere skjulte lag (f.eks. lineære, ulineære, konvulosjonslag, pooling, etc.), og utgangslag. Hvert lag består av flere nevroner/noder.

2. **Fremoverpropagering (feedforward):** Inngangsdata mates inn i inngangslaget og propageres gjennom nettverket helt til utgangslaget.

3. **Beregning av tap (loss):** Når inputdata har propageres gjennom nettverket, sammenlignes nettverkets prediksjon med den faktiske verdien ved hjelp av en tapfunksjon (f.eks. minste kvadraters metode for regresjonsproblemer eller kryssentropi for klassifiseringsproblemer). Tapfunksjonen gir et mål på hvor godt (eller dårlig) nettverket presterer (i forhold til treningsdataene).

4. **Tilbakepropagering (backpropagation):** Ved hjelp av kjernereglen finner man gradienten til tapfunksjonen. Denne kan man bruke til å oppdatere vektene og biasene i nettverket. Ved hjelp av f.eks. (stokastisk) gradientnedstigning, finner man ut i hvilken retning hver parameter bør justeres for å senke verdien til tapsfunksjonen.

5. **Oppdatering av vekter og læringsrate:** Vektene og biasene i nettverket oppdateres basert på resultatene fra tilbakepropageringen og en forhåndsdefinert læringsrate. Hvis læringsraten er for høy, kan nettverket «hoppe over» det optimale punktet, mens hvis den er for lav, kan treningen ta veldig lang tid.

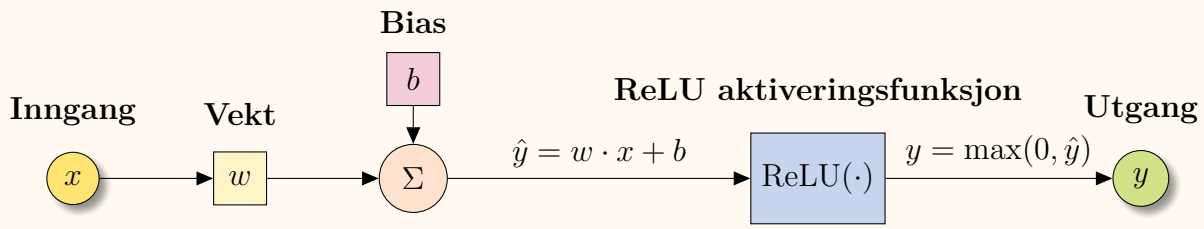
6. **Iterasjon:** Prosessen med fremoverpropagering, tapberegning, tilbakepropagering, og vektjustering gjentas mange ganger—i hver iterasjon eller «epoke»—til nettverket har nådd en tilstrekkelig nøyaktighet, eller til et forhåndsdefinert stopp-kriterium er møtt. Man bør selvsagt også passe på å få en god balanse mellom over- og undertilpasning.

Merk at spesielt begrepene «tilbakepropagering» og «(stokastisk) gradientnedstigning» er her ganske sentrale. Vi skal se litt mer på noen relaterte konsepter i kapittel 13.

Det er også viktig å poengtere at dette er en svært grunnleggende og forneklet forklaring av prosessen, og at det er mange variasjoner og finesser som inngår i treningen av nevrale nettverk, inkludert forskjellige typer nettverk (f.eks. konvolusjonelle nevrale nettverk, rekursive nevrale nettverk), valg av hyperparametere (nettverksdybde, antall noder, læringsrate, etc.), forskjellige

optimeringsalgoritmer (f.eks. gradientnedstigning eller Adam), preprossesering av data, samt mange andre faktorer.

Eksempel 12.2. Trening av én-node-nettverk:

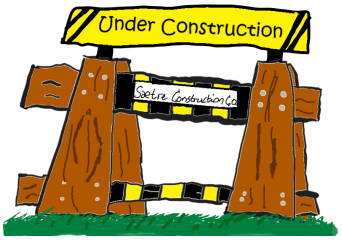


12.3.4 Annnet: trening, alternative lag og arkitekturer*

Trening: Treningsrate og moment, «drop out», hyperparametere

Andre lag Pooling, attention, konvolusjon, softmax.

Andre nettverksarkitekturer Konvolusjonsnetteverk, Tilbake-vendende(/-koblede) nettverk, auto-encodere, residual-nettverk (ResNet), Transformere (Attenion is all you need!)



12.3.5 Dyplæring ved hjelp av MATLAB

MATLAB har innebygd støtte for dyplæring vha. kunstig nevrale nettverk, noe som gjør det til et effektivt verktøy for å bygge, trene og evaluere dype læremodeller.[4]

Her er en grunnleggende veiledning for å bruke MATLAB til dyplæring, samt hvordan du kan implementere forhåndstrente nettverk:

1. **Dyplæringsverktøy:** MATLAB inneholder Deep Learning Toolbox som gir en ramme for design og implementering av nevrale nettverk. Du kan installere og bruke denne verktøykassen for å utføre oppgaver relatert til dyp læring.

2. **Forståelse av nettverksarkitekturer:** Det er viktig å forstå hvordan ulike nettverksarkitekturer fungerer. Du kan lage din egen arkitektur eller bruke en forhåndsdefinert arkitektur.

[4]MATLAB har også innebygd en rekke andre maskinlæringsmetoder.

3. **Treningsprosess:** Når du har valgt en passende nettverksarkitektur eller designet din egen, kan du begynne prosessen med å trene nettverket. I MATLAB kan du bruke 'trainNetwork' funksjonen for å trene nettverket ditt. Denne funksjonen tar i bruk treningsdata, valideringsdata, og ulike andre parametere som læringsrate, antall epoker, batchstørrelse osv.

4. **Evaluering og prediksjon:** Etter trening, kan du evaluere modellens ytelse ved å bruke testdatasettet. Du kan også bruke classify-funksjonen til å lage prediksjoner med den trente modellen.

5. **Bruk av ferdig-trente nettverk:** MATLAB støtter også bruk av forhåndstrente nettverk, som kan være nyttig for overføringslæring.[5] For eksempel kan du laste inn et forhåndstrent nettverk som AlexNet eller VGG-16. Disse forhåndstrente nettverkene kan enten brukes direkte for å lage prediksjoner, eller du kan finjustere de vha. dine egne data.

Et eksempel på hvordan man kan bruke et forhåndstrent nettverk i MATLAB er vist i kodesnutt 12.2.

Kodesnutt 12.2: Eksempel på hvordan man kan laste inn ferdig trente nettverk i MATLAB.

```
% Last inn et forhaandstrent nettverk (AlexNet):
net = alexnet;
% Les inn bildet:
img = imread('mitt_bilde.jpg');
% Endre bildets stoerrelse til nettverkets inngangsstoerrelse:
img = imresize(img, net.Layers(1).InputSize(1:2));
% Klassifiser bildet:
label = classify(net, img);
% Vis bildet og klassifikasjonen:
disp(strcat('Bildet ble kassifisert som:',label);
figure
imshow(img)
text(10,20,char(label),'Color','white')
```

Husk også at dyplæring kan være en tids- og ressurskrevende prosess, og det kan være lurt å bruke GPUer hvis man har det tilgjengelig for å akselerere treningen. MATLAB støtter GPU-beregninger for dyplæring, så lenge du har en kompatibel GPU og de nødvendige driverne installert. Det finnes også muligheter til å eksportere ferdige nettverk til forskjellige programmeringsspråk og til andre kjente ML-verktøy (f.eks. TensorFlow og PyTorch).

Det anbefales at du tar følgende kurs for å lære om hvordan du kan bruke MATLAB til dyplæring. Du vil da lære både hvordan du behandler dataene dine, hvordan du kan bruke eksisterende nettverk direkte eller til overført læring, samt hvordan du kan trene kunstig nevrale nettverk.

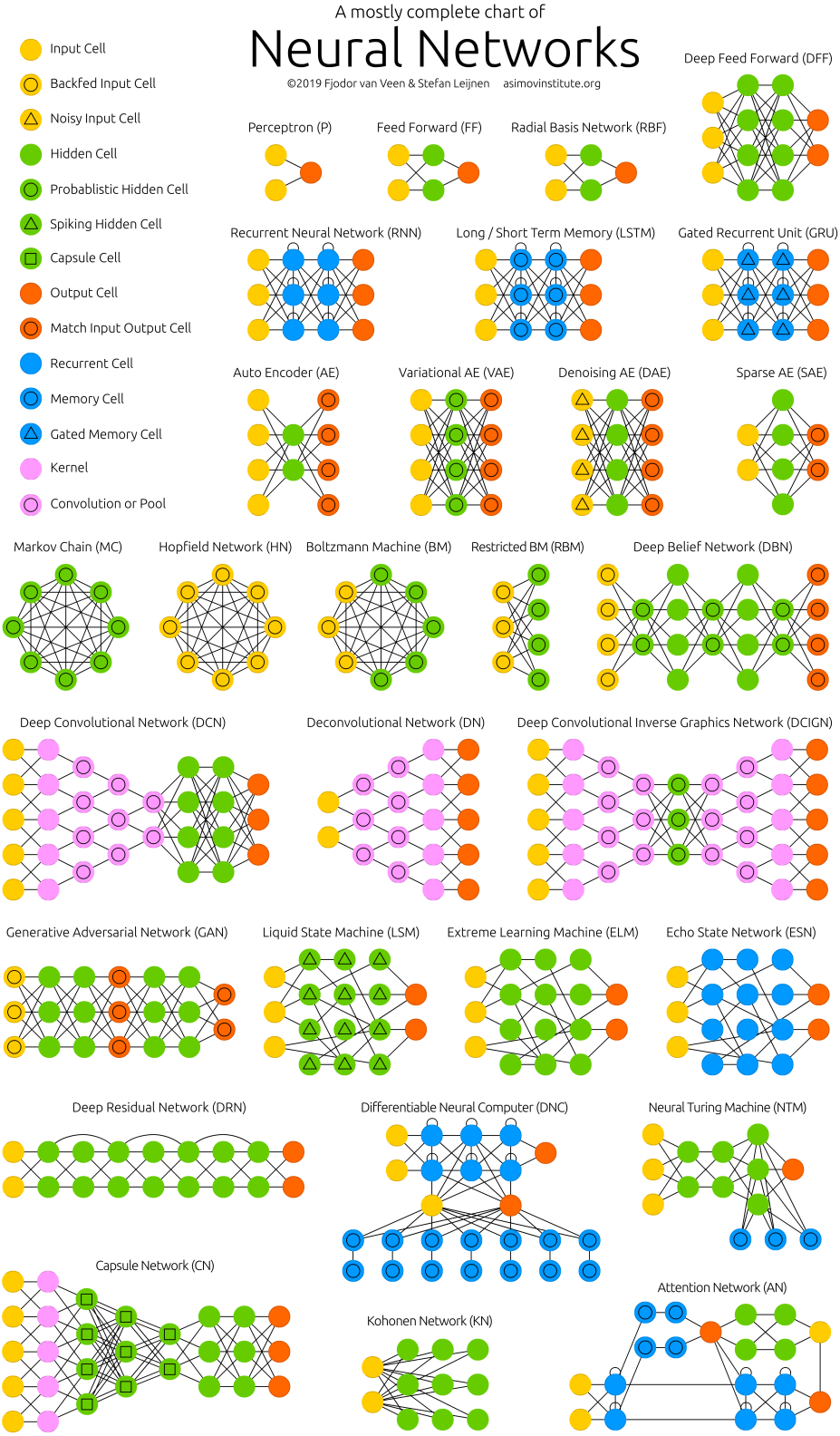
Oppgave 12.2. Ta følgende to-timers kurs:
https://matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning

[5] Overføringslæring involverer finjustering av lagene i et forhåndstrent nettverk. Dette kan f.eks. gjøres ved å erstatte de siste lagene med nye, tilfeldig initialiserte lag, og deretter trene disse lagene på et nytt datasett.

Kursbeviset trengs for å få godkjent øving 6, så husk å ta vare på det!

Noen andre nyttige lenker relatert til maskin- og dyplæringsressurser i MATLAB:

- Design dype nevrale nettverk.
- Applikasjon for å tilpasse data til et enkelt nevralt nettverk.



Figur 12.12: Oversikt over flere kjente kunstig-nevrale-nettverk-arkitekturer. Figur fra https://www.asimovinstitute.org/neural-network-zoo/

13. Optimering

Alternative kilder: Wikipedia.

3D fig av GD fra art

Optimering/optimalisering, også kjent som matematisk programmering, er et sentralt konsept i mange felt, inkludert matematikk, økonomi, fysikk, datavitenskap, maskinlæring og reguleringsteknikk. Den grunnleggende ideen bak optimering er å finne den beste løsningen til et gitt problem innenfor en bestemt gruppe av mulige løsninger og i hendhold til eventuelle begrensninger. Det kan være svært hendig å ha flere optimeringsmetoder liggende i sin automatiserings-verktøykasse, siden disse metodene tillater én å finne løsninger på komplekse problemer på en effektiv og systematisk måte.

Du kan gjerne ta følgende (om enn noe repeterende) MATLAB onramp-kurs allerede nå:

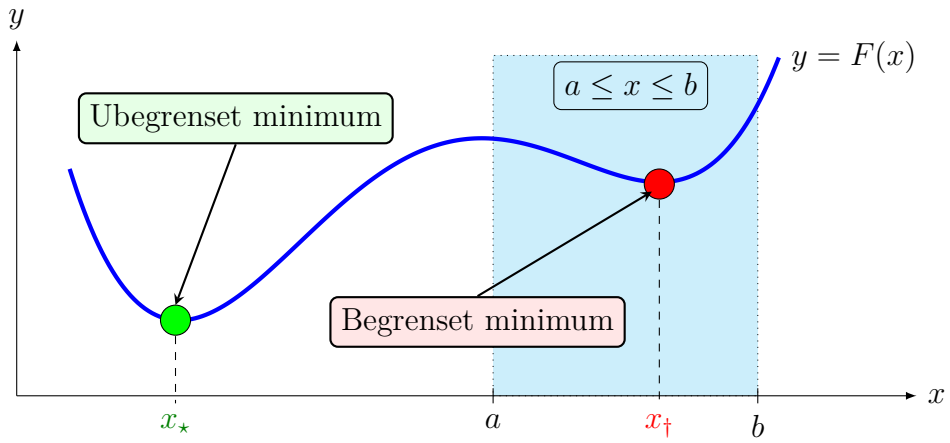
Oppgave 13.1. Ta følgende kurs på ca. 1 time:

https://matlabacademy.mathworks.com/details/optimization-onramp/optim

Husk å ta vare på kursbeviset!

Oppbyggingen: En optimeringsproblem har typisk tre komponenter:

- En funksjon som skal optimaliseres, kalt *målfunksjonen*.
- Et sett med eventuelle begrensninger som løsningen må oppfylle.
- Et sett med variabler som kan justeres for å finne en løsning, såkalte *beslutningsvariabler*.



Figur 13.1: Illustrasjon av både begrenset- og ubegrenset optimering: I det ubegrensede tilfelle er vi ute etter å finne det grønne punktet hvor målfunksjonen $F(x)$ har sin minste verdi, mens i det bregrensede tilfelle ønsker vi det samme innenfor intervallet $a \leq x \leq b$ markert i lyseblått som da tilsvarer det røde punktet som er et *lokalt* minimum.

13.1. Ubegrenset optimering

Vi skal nå se på følgende **ubegrensede optimeringsproblem**:

Ubegrensede optimeringsproblem: Gitt en **målfunksjon** $F(\cdot)$ i bare én variabel x, finn et punkt $x_\star$ slik at $F(x_\star) \leq F(x)$ for alle x nært $x_\star$, altså hvor F har et **minimum**.[a]

[a] Det er også fullt mulig å i stedet lete etter et **maksimum**. F.eks. så tilsvarer et minimum til $F(\cdot)$ et maksimum til $-F(\cdot)$, og vice versa.

13.1.1 Lokale minima og gradienter

Husk at en funksjon F har et *ekstremum* ved et punkt $x_\star$ hvis $F'(x_\star) = 0$.

Newton og gradientnedstigning:

$$x_{k+1} = x_k - \gamma \nabla f(x_k).$$

La $f(x_{k+1})^2 = F(\gamma)$, slik at vi ønsker å minimalisere F:

$$\frac{dF}{d\gamma}(\gamma) = 2f(x_{k+1})\nabla f(x_{k+1})(-\nabla f(x_k)) = 0$$

Om $\delta = 0$ kan ta $f(x_{k+1}) \approx f(x_k) - (\nabla f(x_k)^2\gamma$ og $\nabla f(x_{k+1}) \approx \nabla f(x_k) - \nabla^2 f(x_k)\nabla f(x_k)\gamma$

$$\frac{dF}{d\gamma}(\gamma) = 0 \quad \Longrightarrow$$

13.1.2 Konveksitet, og lokale vs globale løsninger

Lokal vs global optiemring Det er to hovedtyper av optimeringsproblemer: lokal og global optimering. I et *lokal* optimeringsproblem søker vi å finne en løsning som er best innenfor en liten del av løsningsrommet. I et *globalt* optimeringsproblem søker vi å finne den beste løsningen over hele løsningsrommet.

13.1.3 Gradientnedstigning og Newtons metode

Innen både bregrenset og ubegrenset optimering, spiller gradientnedstigning og varianter av Newtons metode nøkkelroller.[1]

Gradientnedstigning

Gradientnedstigning er en iterativ metode som bruker gradienten (eller derivasjonen) av målfunksjonen for å bestemme i hvilken retning vi skal bevege oss i løsningsrommet. Hvis vi har en funksjon $f(x)$ vi ønsker å minimere, kan vi starte med en tilfeldig verdi for x, og iterativt oppdatere x ved å trekke fra en liten brøkdel av gradienten av f ved x.

La oss se på et enkelt eksempel. Anta at målfunksjonen vår er $f(x) = x^2$. Den deriverte av denne funksjonen er $f'(x) = 2x$, så gradient descent vil iterativt oppdatere x som følger: $x_{\text{ny}} = x_{\text{gammel}} - \alpha \cdot 2x_{\text{gammel}}$ hvor α er en liten konstant kjent som læringshastigheten.

Stochastic gradient descent: velg et enkelt eller en batch av *tilfeldige* datapunkt og ta et steg etter det; velg så et nytt tilfeldig punkt og gjenta prosessen. Satser på at dette minimerer sjansen for å havne i et lokalt minima (liten sjanse for at et lokalt minima for et datapunk også er det for et annet; se https://youtu.be/_adhFSH66jc).

Momentum: ”huske” tidlgiere verdier av gradienten

Newtons metode.

Newtons metode, også kalt Newton–Rapshon metoden, lar en finne nullpunkter til en funksjon, noe som har flere bruksområder enn hva man i utgangspuntktet kanskje skulle tro. Metoden fungerer funksjoner av flere variabler, men vi skal for enkelhetsskyld kun se på metoden for bare én variabel.

Finne et punkt z_* slik at $f(z_*) = 0$, hvor $f : \mathbb{R} \to \mathbb{R}$ er en differensierbar funksjon slik at $|f(z)| > 0$ nært z_*.

For $f(z) = z + z^3$ har vi $f'(z) = 1 + 3z^2$, med $z_* = 0$;

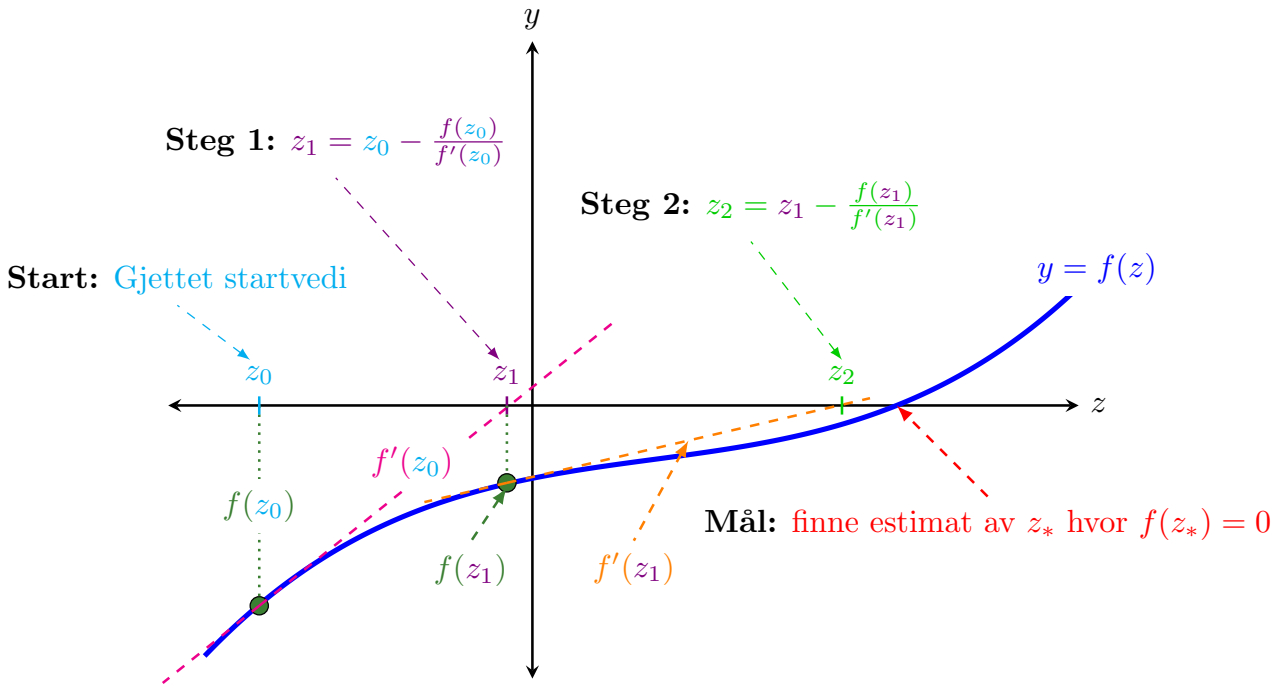
Newton metode for å finne et nullpunkt for en funksjon: Gjett z_0 og sett

$$z_{p+1} = z_p - \frac{f(z_p)}{(f'(z_p))}. \tag{13.1}$$

inntil $|f(z_{p+1}) < \epsilon|$ (evt. $|z_{p+1} - z_p| < \epsilon$) for en ønsket (veldig liten) toleranse $\epsilon > 0$.

Figur 13.2 viser de to første stegene av Netwons metode for å finne nullpunktet $z_* = 2$ til funksjonen $f(z) = (z-2)(z^2+4)$. med startverdi $z_0 = 3$.

[1] Det finnes også andre metoder, som f.eks. simulated annealing for ubegrensede problemer.



Figur 13.2: Illustrasjon av to steg av Newtons metode for å finne estimat av et nullpunkt z_* til en skalar funksjon $f(z)$.

Figur 13.2 viser de to første stegene av Netwons metode for å finne nullpunktet $z_* = 2$ til funksjonen $f(z) = (z-2)(z^2+4)$. med startverdi $z_0 = 3$.

13.1.4 Løse ubegrensede optimeringsproblemer vha. MATLAB

fminsearch

13.1.5 Valg av målfunksjon og sprasommelighet

Konveksitet, Lasso vs minste-kvadraters-metode etc.

Med bare én beslutningsvariabel, si x, så har er valget målfunksjon liten betydning. For eksempel så har $|x|$, x^2, og $e^{|x|}$ det samme minimumet, nemlig $x = 0$, uansett hvile begrensninger man har. Har man derimot to eller flere beslutningsvariable, si x *og* y, så er dette generelt ikke tilfelle lengre tilfellet

Tilpassning av FOPTF modell

13.2. Optimering med begrensninger

Alternative kilder: Video av Brian Douglas (GR4ff0dTLTw)

Gitt følgende problem: minimer (eller maksimer) $J(x)$ uten å bryte begrensningen $c(x) = 0$. Introduserer *Lagrange-funksjonen* $\mathcal{L}(x) = J(x) + \lambda c(x)$ hvor λ er en såkalt *Lagrange-multiplikator* .

13.2.1 Løse begrensede optimeringsproblemer vha. MATLAB

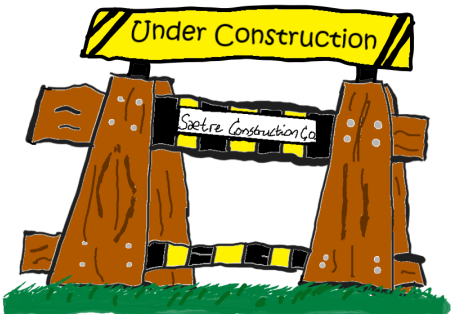
fmincon

13.3. Optimal regulering og modell-prediktiv regulering

Del VII

Sekvens- og Datastyring

14. Sekvens- og Datastyring



14.1. Hva er sekvensstyring?

14.2. Mikrokontrollere og PLS

14.3. Boolsk algebra

Veritasium om Opphavet til datamaskiner: https://www.youtube.com/watch?v=FU_YFpfDqqA

14.4. Algoritmer

14.5. Tilstandsmaskiner og Automata

14.6. Grafalgortimer

Referanser

[Åström and Hägglund, 1984] Åström, K. J. and Hägglund, T. (1984). Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica*, 20(5):645–651.

[Åström and Hägglund, 2006] Åström, K. J. and Hägglund, T. (2006). *Advanced PID control*, volume 461. ISA-The Instrumentation, Systems and Automation Society.

[Balchen et al., 2016] Balchen, J. G., Andresen, T., and Foss, B. A. (2016). *Reguleringsteknikk*. NTNU, Institutt for teknisk kybernetikk.

[Bjørvik and Hveem, 2014] Bjørvik, K. and Hveem, P. (2014). Reguleringsteknikk.

[Brunton and Kutz, 2022] Brunton, S. L. and Kutz, J. N. (2022). *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press.

[Cengel and Cimbala, 2013] Cengel, Y. and Cimbala, J. (2013). *EBOOK: Fluid Mechanics Fundamentals and Applications (SI units)*. McGraw Hill.

[Desborough and Miller, 2002] Desborough, L. and Miller, R. (2002). Increasing customer value of industrial control performance monitoring-honeywell's experience. In *AIChE symposium series*, number 326 in 1, pages 169–189. New York; American Institute of Chemical Engineers; 1998.

[Dessen, 2019] Dessen, F. (2019). Optimizing order to minimize low-pass filter lag. *Circuits, Systems, and Signal Processing*, 38(2):481–497.

[Dormand and Prince, 1980] Dormand, J. R. and Prince, P. J. (1980). A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26.

[Fridman, 2014] Fridman, E. (2014). *Introduction to time-delay systems: Analysis and control*. Springer.

[Grimholt, 2018] Grimholt, C. (2018). *Optimal tuning of PID controllers*. PhD thesis, Norwegian University of Science and Technology (NTNU) Trondheim, Norway.

[Grimholt and Skogestad, 2013] Grimholt, C. and Skogestad, S. (2013). Optimal pid-control on first order plus time delay systems & verification of the simc rules. *IFAC Proceedings Volumes*, 46(32):265–270.

[Haugen, 2023] Haugen, F. A. (2023). Modeling, simulation and control. Tilgjengelig via: http://www.techteach.no/control/.

[Samad, 2017] Samad, T. (2017). A survey on industry impact and challenges thereof [technical activities]. *IEEE Control Systems Magazine*, 37(1):17–18.

[Samad et al., 2020] Samad, T., Bauer, M., Bortoff, S., Di Cairano, S., Fagiano, L., Odgaard, P. F., Rhinehart, R. R., Sánchez-Peña, R., Serbezov, A., Ankersen, F., et al. (2020). Industry engagement with control research: Perspective and messages. *Annual Reviews in Control*, 49:1–14.

[Seborg et al., 2016] Seborg, D. E., Edgar, T. F., Mellichamp, D. A., and Doyle III, F. J. (2016). *Process dynamics and control*. John Wiley & Sons, 4th edition.

[Sharma et al., 2022] Sharma, A., Kosasih, E., Zhang, J., Brintrup, A., and Calinescu, A. (2022). Digital twins: State of the art theory and practice, challenges, and open research questions. *Journal of Industrial Information Integration*, page 100383.

[Shinskey, 2002] Shinskey, F. G. (2002). Process control: as taught vs as practiced. *Industrial & Engineering Chemistry Research*, 41(16):3745–3750.

[Siciliano et al., 2008] Siciliano, B., Khatib, O., and Kröger, T. (2008). *Springer handbook of robotics*, volume 200. Springer.

[Skogestad, 2003] Skogestad, S. (2003). Simple analytic rules for model reduction and pid controller tuning. *Journal of process control*, 13(4):291–309.

[Skogestad and Postlethwaite, 2007] Skogestad, S. and Postlethwaite, I. (2007). *Multivariable feedback control: analysis and design*. Wiley New York, 2nd edition.

[Spong et al., 2020] Spong, M. W., Hutchinson, S., and Vidyasagar, M. (2020). *Robot modeling and control*. John Wiley & Sons.

[Taguchi and Araki, 2000] Taguchi, H. and Araki, M. (2000). Two-degree-of-freedom pid controllers—their functions and optimal tuning. *IFAC Proceedings Volumes*, 33(4):91–96.

[Tarbouriech and Turner, 2009] Tarbouriech, S. and Turner, M. (2009). Anti-windup design: an overview of some recent advances and open problems. *IET control theory & applications*, 3(1):1–19.

[Wittenmark et al., 2002] Wittenmark, B., Åström, K. J., and Årzén, K.-E. (2002). Computer control: An overview. *IFAC Professional Brief*, 1:2.

[Wu et al., 2008] Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., et al. (2008). Top 10 algorithms in data mining. *Knowledge and information systems*, 14:1–37.

Vedlegg

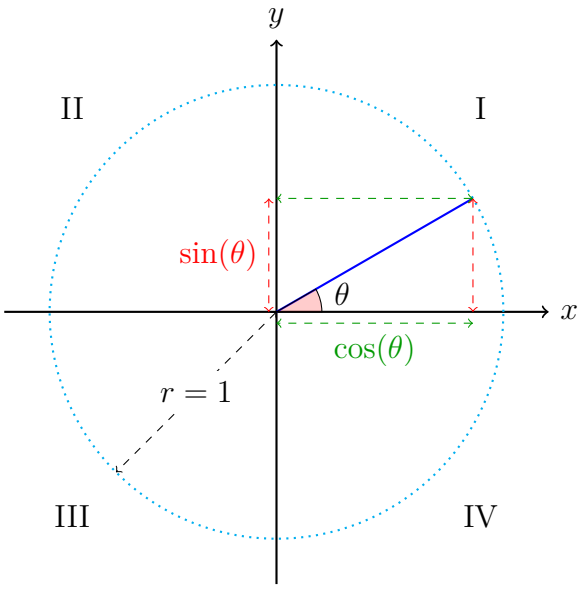
A. Vedlegg

A.1. Det greske alfabetet

Liten	Stor	Navn
α	A	alfa
β	B	beta
γ	Γ	gamma
δ	Δ	delta
ϵ	E	epsilon
ζ	Z	zeta
η	H	eta
θ	Θ	theta
ι	I	iota
κ	K	kappa
λ	Λ	lambda
μ	M	my
ν	N	ny
ξ	Ξ	ksi
o	O	omikron
π	Π	pi
ρ	P	rho
σ	Σ	sigma
τ	T	tau
υ	Υ	ypsilon
ϕ	Φ	phi
χ	X	chi
ψ	Ψ	psi
ω	Ω	omega

A.2. Trigonometriske identiteter

Enhetssirkelen Sinus- og cosinus-funksjonene er definert i forhold til enhetssirkelen (altså en sirkel med radius $r = 1$) slik som vist i figure A.1. Denne er delt opp i kvadrantene I, II, III og IV.



Figur A.1: Enhetssirkelen og tilsvarende kvadranter (I, II, III og IV).

Siden $\sin(\theta)$ og $\cos(\theta)$ angir et punkt på enhetssirkelen, har man at følgende alltid holder:

$$\sin^2(\theta) + \cos^2(\theta) = 1.$$

Tangensfunksjonen er definert som sinus delt på consinus:

$$\tan(\theta) = \frac{\sin(\theta)}{\cos(\theta)}.$$

Denne funksjonen er med andre ord ikke definert for verdier av θ slik at $\cos(\theta) = 0$.

Arktangens-/invers-tangens-funksjonen La $x = r\cos(\theta)$ og $y = r\sin(\theta)$ for en eller annen $r > 0$. Vi har dermed

$$\tan(\theta) = \frac{l\sin(\theta)}{l\cos(\theta)} = \frac{\sin(\theta)}{\cos(\theta)}.$$

Vi kan derfor prøve å finne θ fra x og y via arktangens-/invers-tangens-funksjonen:

$$\theta = \arctan\left(\frac{y}{x}\right).$$

Merk dog at siden $\frac{y}{x} = \frac{-y}{-x}$, så kan vi ikke se forskjell på om x og y ligger kvadrant I (hhv. II) eller III (hhv. IV). I stedet kan vi bruke to-arguments invers-tangens-funksjonen:

$$\text{Atan}(y,x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{hvis } x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{hvis } x < 0 \text{ og } y \geq 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{hvis } x < 0 \text{ og } y < 0 \\ +\frac{\pi}{2} & \text{hvis } x = 0 \text{ og } y > 0 \\ -\frac{\pi}{2} & \text{hvis } x = 0 \text{ og } y < 0 \\ \text{Ikke definert} & \text{hvis } x = 0 \text{ og } y = 0 \end{cases}.$$

Andre trigonometriske identiteter Sum av vinkler:

$$\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y)$$

$$\sin(x+y) = \cos(x)\sin(y) + \sin(x)\cos(y)$$

$$\sin(x-y) = \sin(x)\cos(y) - \cos(x)\sin(y)$$

A.3. Derivasjonsregler: Produkt- og kjerneregelen:

Gitt to differensierbare funksjoner $f(x)$ og $g(x)$, så

$$\frac{d}{dx}\left(f(x)g(x)\right) = f'(x)g(x) + f(x)g'(x), \qquad \text{(produktregelen)}$$

mens

$$\frac{d}{dx}\left(f\big(g(x)\big)\right) = f'(g(x))g'(x). \qquad \text{(kjerneregelen)}$$

A.4. Komplekse tall

A.5. Løsningsforslag til oppgaver

A.5.1 Oppgave 2.1

Vi har for den gitte løsningen at $\dot{x} = a\lambda e^{\lambda t} - b\lambda e^{-\lambda t}$, mens $\ddot{x} = a\lambda^2 e^{\lambda t} + b\lambda^2 e^{-\lambda t} = \lambda^2 x(t)$, som viser at dette er en løsning. Vi harvidere at $x(0) = ae^{\lambda +} + be^{-\lambda 0} = a + b$ og $\dot{x}(0) = a\lambda e^{\lambda 0} - b\lambda e^{-\lambda 0} = \lambda(a-b)$, slik at vi kan finne a og b fra ligningen $a+b = x(0)$ og $a-b = \dot{x}/0)/\lambda$.

Systemet er ustabilt siden leddet $e^{\lambda t}$ går mot uendelig når $t \to \infty$ hvis $\lambda > 0$, mens $e^{-\lambda t}$ går mot uendelig hvis $\lambda < 0$.

A.5.2 Oppgave 2.3

Proporsjonalforsterkningen må være større enn 2, altså $k_P > 2$. Ved prøving og feiling kan man også finne ut at man ikke ha stort mer enn $k_P = 13$ før systemet blir ustabilt.

A.5.3 Oppgave 2.4

a) Systemet har et likevektspunkt for $y = \bar{y} = 2$ i siden da $\dot{y} = 0$. Vi gjør derfor følgende bytte av variabler: $x = y - \bar{y} = y - 2$. Dynamikken til den nye variabelen x er $\dot{x} = \dot{y} = -y + 2 = -(x + 2) + 2 = -x$, altså $\dot{x} = -x$.

b) Vi vet at diff.ligningen $\dot{x} = -x$ har løsningen $x(t) = x_0 \exp(-t)$, hvor $x_0 = x(0) = y(0) - 2$. Siden $\exp(-t) \to 0$ når $t \to \infty$, så må også $x(t) \to \infty$ uansett hva $x(0)$ er. Dermed går også $y(t)$ også til verdien $\bar{y} = 2$ uansett hva $y(0)$ er, noe som indikerer at systemet er (eksponentielt) stabilt.

A.5.4 Oppgave 2.5

Fra $u = k_P(r - y)$ har vi $U(s) = k_P(R(s) - Y(s))$. Ved å sette inn dette i utrykket $Y(s) = P(S)U(s)$ får vi

$$Y(s) = P(s)k_P(R(s) - Y(s)) \implies (1 + k_P P(s))Y(s) = k_P P(s)R(s) \implies \frac{Y(s)}{R(s)} = \frac{k_P P(s)}{1 + k_p P(s)}.$$

A.5.5 Oppgave 3.2

Massesenteret har posisjon (x_{ms}, y_{ms}), mens orienteringen til farkosten er gitt ved vinkelen θ. Systemet har dermed tre frihetsgrader. Tar vi med hastighetene, så får vi systemets seks tilstander: $(x_{ms}, y_{ms}, \theta, \dot{x}_{ms}, \dot{y}_{ms}, \dot{\theta})$.

Vi kan bare kontrollere systemet gjennom kreftene F_1 og F_2 fra viftene, slik at systemet har to pådrag: $(u_1, u_2) = (F_1, F_2)$. Systemets underaktueringsgrad er dermed $3 - 2 = 1$, noe som betyr at vi mangler ett pådrag for ha å full kontroll over systemets akselerasjoner.

La $F := u_1 + u_2$, altså summen av kreftene som virker på farkosten, og la $\tau := r(u_2 - u_1)$, altså momentet om massesenteret (målt positivt mot klokken). Vi har da at bevegelsesligningen til systemet er

$$m\ddot{x}_{ms} = F\cos(\theta) \tag{A.1a}$$
$$m\ddot{y}_{ms} = F\sin(\theta) \tag{A.1b}$$
$$J\ddot{\theta} = \tau \tag{A.1c}$$

hvor de to første ligningene følger fra Newtons andre lov og grunnleggende trigonometri, mens den siste følger fra Eulers lov (momentbalanse).

Ved å ta $z = (z_1, z_2, z_3, z_4, z_5, z_6) = (x_{ms}, y_{ms}, \theta, \dot{x}_{ms}, \dot{y}_{ms}, \dot{\theta})$ og $u = (u_1, u_2) = (F_1, F_2)$ kan

vi skrive de dynamiske bevegelsesligningene på tilstandromform:

$$\begin{aligned}
\dot{z}_1 &= z_4, \\
\dot{z}_2 &= z_5, \\
\dot{z}_3 &= z_6, \\
\dot{z}_4 &= \frac{1}{m}\cos(z_3)(u_1+u_2), \\
\dot{z}_5 &= \frac{1}{m}\sin(z_3)(u_1+u_2), \\
\dot{z}_6 &= \frac{r}{J}(u_2-u_1).
\end{aligned}$$

A.5.6 Oppgave 3.3

a)

Vi kan her både bruke momentbalanse og energibalanse.

Momentbalanse: La oss først ta "venstresiden" av giret, hvor vi fra Eulers ligning får:

$$J_1\dot{\omega}_1 = \tau_1 - \hat{\tau}$$

hvor dreiemomentet $\hat{\tau}$ er et "tap" pga. det som skjer på "høyre side" av giret. La oss nå sette opp momentbalansen på høyresiden:

$$(J_2+J_3)\dot{\omega}_2 = \hat{\tau}/n - d_v\omega_2$$

hvor vi har brukt at $\tau_2 = \tau_1/n$ over et gir, samt at vi må byttet fortegn på leddet med $\hat{\tau}$ pga.Newtons 3. lov. Ved å bruke $\omega_2 = n\omega_1$, og dermed $\dot{\omega}_2 = n\dot{\omega}_1$, får vi

$$(J_2+J_3)n\dot{\omega}_1 = \hat{\tau}/n - d_v n\omega_1 \quad \Longrightarrow \quad \hat{\tau} = (J_2+J_3)n^2\dot{\omega}_1 + d_v n^2\omega_1.$$

Ved å sette dette inn i utrykket for venstresiden finner vi at

$$J_1\dot{\omega}_1 = \tau_1 - \left[(J_2+J_3)n^2\dot{\omega}_1 + -d_v n^2\omega_1\right] \quad \Longrightarrow \quad \dot{\omega}_1 = \frac{1}{J}\left[\tau_1 - d_v n^2\omega_1\right].$$

Energibalanse: Energien "lagret" i systemet ved et gitt tidspunkt tilsvarer den kinetiske energien:

$$E_{sys} = \mathcal{K} = \frac{1}{2}J_1\omega_1^2 + \frac{1}{2}(J_2+J_3)\omega_2^2.$$

Dermed har vi at

$$\dot{E}_{sys} = J_1\omega_1\dot{\omega}_1 + (J_2+J_3)\omega_2\dot{\omega}_2 = (J_1 + (J_2+J_3)n^2)\omega_1\dot{\omega}_1 = J\omega_1\dot{\omega}_1.$$

Dette må tilsvare rate på arbeidet utført på systemet, gitt ved $P_{tilfrt} = \tau_1\omega_1$, minus rate av arbeidet utført av systemet pga. friksjonen: $P_{utfrt} = \tau_f\omega_2 = d_v\omega_2^2 = d_v n^2\omega_1^2$. Altså $\dot{E}_{sys} = P_{tilfrt} - P_{utfrt}$, som, ved å dele på $J\omega_1$ på begge sider, er lik

$$\frac{1}{\omega_1 J}(J\omega_1\dot{\omega}_1) = \dot{\omega}_1 = \frac{1}{\omega_1 J}\left(\tau_1\omega_1 - d_v n^2\omega_1^2\right) = \frac{1}{J}\left(\tau_1 - d_v n^2\omega_1\right)$$

som ønsket (det er mer riktig å fakturere ut $J\omega_1$ på begge og si at det må holde for alle verdier av ω_1, med det å dele gjør det noe mer tydelig).

b)

Vi vet at

$$\dot{\omega}_1 = \frac{1}{n}\dot{\omega}_2 = \frac{1}{J(n)}\left[\tau_1 - d_v \cdot n^2\omega_1\right]$$

hvor $J(n) = J_1 + (J_2 + J_3) \cdot n^2$. Siden vil vil finne maksimal vinkelakselerasjon bare når $\omega_1 = \omega_2 = 0$, betyr dette at vi må finne n slik at (vha. produktregelen)

$$\frac{d}{dn}\left[\frac{n}{J(n)}\tau_1\right] = 0 \quad \Longrightarrow \quad \left[\frac{J(n) - nJ'(n)}{J(n)^2}\right]\tau_1 = 0.$$

Vi trenger her bare ta hensyn til det som er i telleren i klammeparentesene:

$$J(n) - nJ'(n) = J_1 + (J_2 + J_3)n^2 - 2(J_2 + J_3)n^2 = 0 \quad \Longrightarrow \; n = \sqrt{\frac{J_1}{J_2 + J_3}}.$$

Ved å sette inn $J_1 = 4$, $J_2 = 1$, $J_3 = 15$, får vi dermed:

$$n = \sqrt{\frac{4}{16}} = \frac{2}{4} = 1/2.$$

A.5.7 Oppgave 6.2

Fra eksempel 3.6 vet vi at kraften tilsvarende tyngekraften som virker på bilen parallelt med bakken er $G_{\parallel} = mg\sin(\alpha)$. Ved å anta at det ikke virker noen tap på bilen slik som luftmotstand eller friksjon, så har man fra Newtons andre lov at $m\dot{v} = -G_{\parallel} + F$, hvor v er bilens hastighet opp bakken og F er summen av kreftene som virker på bilen fra motoren. Rettere sagt, så tilsvarer kraften F kraften fra hvert hjul på vegen når hjulene ikke spinner. Bidraget fra hvert hjul i forhold til dreiemomentet som virker på det må derfor være $\tau = r \cdot F/4$. Systemet vil ha en konstant hastighet når $\dot{v} = 0$, noe som impliserer $F = G_{\parallel} = mg\sin(\alpha)$, slik at

$$\tau = r \cdot F/4 = rmg\sin(\alpha)/4 = 0.3 \cdot 1500 \cdot 9.81 \cdot \sin(25 \cdot \pi/180)/4 \approx 466\,\mathrm{N\,m}$$

A.5.8 Oppgave 7.1

Vi har at $(x+1)^2 - 4 = 0$ for at $\dot{x} = 0$. Dette holder bare hvis $(x+1) = \sqrt{4} = 2$, slik at systemet har likevektspunktet $x = \bar{x} = 1$. Dette gir oss

$$A = \frac{\partial f}{\partial x}\Big|_{x=1} = 2(x+1)|_{x=1} = 4,$$

slik at $\frac{d}{dt}\delta x = 4\delta x$. Løsningen er dermed $\delta x(t) = \delta x_0 \exp(4t)$, slik at det lineariserte systemet er ustabilt, noe som igjen impliserer at det ulineære systemet er ustabilt.

A.5.9 Oppgave 2.6

Vi har igjen at

$$y[k+1] = e^{-a\mathfrak{h}}\left(y[k] + b\int_{t_k}^{t_k+\mathfrak{h}} e^{a(t-t_k)}u(t)dt\right).$$

Ved å sette inn for $u(t) = \frac{t-t_k}{t_{k+1}-t_k}(u[k]-u[k-1])+u[k-1]$ og bruke at $\mathfrak{h} = t_{k+1}-t_k$ får vi

$$y[k+1] = e^{-a\mathfrak{h}}\left(y[k] + b\int_{t_k}^{t_k+\mathfrak{h}} e^{a(t-t_k)}\left(\frac{t-t_{k-1}}{\mathfrak{h}}(u[k]-u[k-1])+u[k-1]\right)dt\right).$$

Integralet $\int_{t_k}^{t_k+\mathfrak{h}} e^{a(t-t_k)}\left(\frac{t-t_k}{\mathfrak{h}}(u[k]-u[k-1])+u[k-1]\right)dt$ kan vi skrive om som følger:

$$\left(\frac{-t_k}{\mathfrak{h}}(u[k]-u[k-1])+u[k-1]\right)\int_{t_k}^{t_k+\mathfrak{h}} e^{a(t-t_k)}dt + \frac{(u[k]-u[k-1])}{\mathfrak{h}}\int_{t_k}^{t_k+\mathfrak{h}} t\cdot e^{a(t-t_k)}dt$$

Fra før vet vi at

$$\int_{t_k}^{t_k+\mathfrak{h}} e^{a(t-t_k)}dt = \left[\frac{1}{a}e^{a(t-t_k)}\right]_{t_k}^{t_k+\mathfrak{h}} = \frac{1}{a}\left(e^{a\mathfrak{h}}-1\right).$$

Mens

$$\int_{t_k}^{t_k+\mathfrak{h}} t\cdot e^{a(t-t_k)}dt = \left[\frac{(at-1)}{a^2}e^{a(t-t_k)}\right]_{t_k}^{t_k+\mathfrak{h}} = \frac{(a(t_k+\mathfrak{h})-1)}{a^2}e^{a\mathfrak{h}} - \frac{(at_k-1)}{a^2}.$$

Integralet blir dermed

$$\left(\frac{-t_k}{\mathfrak{h}}(u[k]-u[k-1])+u[k-1]\right)\frac{1}{a}\left(e^{a\mathfrak{h}}-1\right) + \frac{(u[k]-u[k-1])}{\mathfrak{h}}\left(\frac{(a(t_k+\mathfrak{h})-1)}{a^2}e^{a\mathfrak{h}} - \frac{(at_k-1)}{a^2}\right).$$

Legg her spesielt merke til at $\left(\frac{t_k}{\mathfrak{h}}(u[k]-u[k-1])\right)\frac{1}{a}\left(e^{a\mathfrak{h}}-1\right)$ dukker opp i begge leddene, men med forskjellige fortegn, slik at det over kan reduseres til

$$\begin{aligned} &u[k-1]\frac{1}{a}\left(e^{a\mathfrak{h}}-1\right) + \frac{(u[k]-u[k-1])}{\mathfrak{h}}\left(\frac{(a\mathfrak{h}-1)}{a^2}e^{a\mathfrak{h}}+\frac{1}{a^2}\right) \\ &= \frac{u[k]}{\mathfrak{h}}\left(\frac{(a\mathfrak{h}-1)}{a^2}e^{a\mathfrak{h}}+\frac{1}{a^2}\right) + u[k-1]\left(\frac{1}{\mathfrak{h}a^2}\left(e^{a\mathfrak{h}}-1\right)-\frac{1}{a}\right). \end{aligned}$$

Dette gir oss diskretiseringen

$$y[k+1] = e^{-a\mathfrak{h}}\left(y[k] + b\left(\frac{u[k]}{\mathfrak{h}}\left(\frac{(a\mathfrak{h}-1)}{a^2}e^{a\mathfrak{h}}+\frac{1}{a^2}\right) + u[k-1]\left(\frac{1}{\mathfrak{h}a^2}\left(e^{a\mathfrak{h}}-1\right)-\frac{1}{a}\right)\right)\right).$$

A.5.10 Oppgave 10.1

Vi ønsker å løse følgende ligning:

$$\dot{y} = -ay + b\cdot A\sin(\omega t), \quad y(0) = y_0.$$

Vi kan løse denne ved å legge til ay og multiplisere med e^{at} på begge sider,

$$e^{at}\dot{y} + ae^{at}x = be^{at}A\sin(\omega t) \quad \Longrightarrow \quad \frac{d}{dt}\left(e^{at}y(t)\right) = be^{at}A\sin(\omega t),$$

for så integrere på begge side:

$$e^{at}y(t) - y_0 = b\cdot A\int_0^t e^{a\sigma}\sin(\omega\sigma)d\sigma \quad \Longrightarrow \quad y(t) = e^{-at}\left[y_0 + b\cdot A\int_0^t e^{a\sigma}\sin(\omega\sigma)d\sigma\right]. \quad (A.2)$$

For å regne ut integralet $I := \int_0^t e^{a\sigma}\sin(\omega\sigma)d\sigma$ på høyresiden bruker vi delvis integrasjon:

$$\begin{aligned} I := \int_0^t e^{a\sigma}\sin(\omega\sigma)d\sigma &= \int_0^t \left[\frac{d}{d\sigma}\left(\frac{1}{a}e^{a\sigma}\sin(\omega\sigma)\right) - \frac{\omega}{a}e^{a\sigma}\cos(\omega t)\right]d\sigma \\ &= \left[\left(\frac{1}{a}e^{a\sigma}\sin(\omega\sigma)\right)\right]_0^t - \frac{\omega}{a}\int_0^t e^{a\sigma}\cos(\omega t)d\sigma. \end{aligned} \tag{A.3}$$

Vi gjør så det samme med med det gjenværende integralet:

$$\begin{aligned} \int_0^t e^{a\sigma}\cos(\omega\sigma)d\sigma &= \int_0^t \left[\frac{d}{d\sigma}\left(\frac{1}{a}e^{a\sigma}\cos(\omega\sigma)\right) + \frac{\omega}{a}e^{a\sigma}\sin(\omega t)\right]d\sigma \\ &= \left[\left(\frac{1}{a}e^{a\sigma}\cos(\omega\sigma)\right)\right]_0^t + \frac{\omega}{a}\underbrace{\int_0^t e^{a\sigma}\sin(\omega t)d\sigma}_{=I}. \end{aligned}$$

Ved å sette dette inn i (A.3) finner vi at integralet $I := \int_0^t e^{a\sigma}\sin(\omega\sigma)d\sigma$ må tilfredsstille

$$\begin{aligned} I &= \left[\left(\frac{1}{a}e^{a\sigma}\sin(\omega\sigma)\right)\right]_0^t - \frac{\omega}{a}\left[\left(\frac{1}{a}e^{a\sigma}\cos(\omega\sigma)\right)\right]_0^t - \frac{\omega^2}{a^2}I \\ \implies (a^2+\omega^2)I &= a\left[e^{a\sigma}\sin(\omega\sigma)\right]_0^t - \omega\left[e^{a\sigma}\cos(\omega\sigma)\right]_0^t \\ \implies \quad I &= \frac{1}{(a^2+\omega^2)}\left[e^{at}\left(a\sin(\omega t) - \omega\cos(\omega t)\right) + \omega\right] \end{aligned}$$

Sette dette inn i (A.2) får vi

$$y(t) = e^{-at}\left(y_0 + \frac{bA\omega}{a^2+\omega^2}\right) + \frac{bA}{\sqrt{a^2+\omega^2}}\left(\frac{a}{\sqrt{a^2+\omega^2}}\sin(\omega t) - \frac{\omega}{\sqrt{a^2+\omega^2}}\cos(\omega t)\right)$$

Ved å ta $\cos(-\phi) = \frac{a}{\sqrt{a^2+\omega^2}}$ og $\sin(-\phi) = \frac{\omega}{\sqrt{a^2+\omega^2}}$ kan vi bruke den trigonometriske identiteten $\sin(\alpha-\beta) = \sin(\alpha)\cos(\beta) - \cos(\alpha)\sin(\beta)$ til å finne at

$$\sin(\omega t+\phi) = \frac{a}{\sqrt{a^2+\omega^2}}\sin(\omega t) - \frac{\omega}{\sqrt{a^2+\omega^2}}\cos(\omega t) = \cos(\phi)\sin(\omega t) - \sin(-\phi)\cos(\omega t)$$

hvor $\phi(\omega) = \arctan\left(\frac{-\omega}{a}\right)$. Ved også ta $A_r(\omega) = b/\sqrt{a^2+\omega^2}$, få vi dermed

$$y(t) = e^{-at}\left(y_0 + \frac{bA\omega}{a^2+\omega^2}\right) + AA_r(\omega)\sin(\omega t + \phi(\omega))$$

som tilsvarer (10.1) slik som ønsket! 🥳

Indeks