NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for naturvitenskap og teknologi
Institutt for kjemisk prosessteknologi

HOVEDOPPGAVE 2004

Dynamisk modellering av LNG-prosess i Matlab

Ingrid Kristine Wold

En dynamisk modell med rigorøs termodynamikk for CO_2-kjøling av naturgass er implementert i Matlab/Simulink.

Faglærer: Professor Sigurd Skogestad, NTNU
Ekstern veileder: Jostein Pettersen, Statoil

Utført i tiden: 15.01.2004-10.6.2004

Antall sider: 105
Hovedrapport: 46
Vedlegg: 59

Jeg erklærer at arbeidet er utført selvstendig og i samsvar med NTNUs eksamensreglement.

Dato og underskrift:

_____

Forord

Rapporten beskriver hovedoppgaven utført våren 2004 ved Norges Teknisk-Naturvitenskapelige Universitet (NTNU), Institutt for Kjemisk Prosessteknologi, i samarbeid med Statoil.

Hovedveileder og medveileder ved NTNU har vært henholdsvis Sigurd Skogestad og Jørgen Bauck Jensen. I tillegg har Tore Haug-Warberg bidratt i forbindelse med termodynamiske beregninger. Espen Storkaas har veiledet i arbeidet med Simulink.

Jostein Pettersen, kontaktperson og veileder i Statoil, har under arbeidet med forprosjektet fra høsten 2003 bidratt til god forståelse av prosessen samt gitt tilgang til nødvendige anleggsdata, som også er benyttet i hovedoppgaven.

Takk til hver av de ovennevnte, som har vært til stor hjelp under arbeidet. Jeg vil spesielt takke Jørgen for hans positive engasjement. Takk også til kontor- og hybelvenner og familie for bidrag på det sosiale plan, samt Andreas Lund Danielsen og Anders A. Wold for gjennomlesning.

Trondheim 10.06.2004

_____

Ingrid Kristine Wold

Sammendrag

Hovedoppgaven inngår i et samarbeid mellom NTNU og Statoil, der målet er å modellere forkjøling med CO_2 i en LNG-prosess i Matlab.

Kjøling av naturgass mot CO_2 i tofaseområdet er modellert dynamisk i Matlab ved hjelp av Simulink. Det er sett på en typisk prosess med muligheter for å endre strømsammensetninger og betingelser i henhold til spesifikke prosesser. Hovedfokus under implementeringen har vært rettet mot termodynamikken i varmeveksleren.

Siden varmeveksleren er en ren fordamper, vil ikke manipulering av gjennomstrømning ha innvirkning på CO_2-temperaturen utover den lille endringen som kommer av at trykket forandres. I en virkelig prosess gjør overheteren at overført varme øker med gjennomstrømningen.

Det modellerte systemet resulterer i et relativt komplisert ODE-sett der substituerte likevektskorreksjoner virker som forstyrrelser og fører til lang regnetid. En implisitt DAE-løser vil sannsynligvis være et bedre alternativ enn den eksplisitte ODE-løseren som er brukt i oppgaven. En slik løser er ikke benyttet fordi den ikke kan velges eksplisitt i Simulink. Dersom et implisitt DAE-system skal løses i Simulink, må initialiserte algebraiske beskrankninger innføres.

To versjoner av modellen med ulik kompleksitet i Simulink og Matlab-rutiner er generert. Økt bruk av grafikk til fordel for tekstbasert kode gir bedre oversikt over løsningsgangen i prosessen. Det fører imidlertid også til at modellen krever større regnekapasitet.

Ved en eventuell videreutvikling av modellen, bør løsning av DAE-likninger i Simulink undersøkes nærmere. Dersom det legges stor vekt på regnetiden, er en ren Matlab-modell trolig den beste løsningen.

Innhold

Figurer

Tabeller

1 Innledning

I forprosjektet til denne hovedoppgaven ble det gjort en studie av CO_2 som forkjølingsmedium ved produksjon av flytende naturgass til havs. Det vil være nødvendig å følge strenge sikkerhetsmessige retningslinjer ved utforming av et flytende anlegg, og kjøling med karbondioksid istedenfor hydrokarboner er et ledd i denne prosessen. The Linde Statoil LNG Technology Alliance har utviklet et konsept med propan/etan i forkjølingskretsen, og dette var utgangspunktet for studiet. Ulike forkjølingsprosesser med CO_2 og hydrokarboner ble sammenliknet ved simulering i Hysys. En CO_2-krets vil ha et høyere kraftforbruk enn en hydrokarbonkrets, og ulike tiltak for å redusere kraftforbruket ble testet. Resultatene viste at karbondioksid kan være et konkurransedyktig kjølemedium på et flytende LNG-anlegg.

Det er ønskelig å se nærmere på CO_2-kjøling blant annet ved å inkludere en detaljert beskrivelse av termodynamikk i varmeveksleren. Programmet Hysys gir ikke full oversikt over beregningsgrunnlaget, og det er derfor noe usikkerhet knyttet til enkelte resultater fra de utførte simuleringene. I Matlab er det mulig å ha kontroll på alle verdier både for tilstandsberegninger og varmekapasiterer. I tillegg vil en modell i Matlab, i motsetning til Hysys, lett kunne omfatte dynamikk i prosessen. Målet for oppgaven er å sette opp en dynamisk modell med rigorøs termodynamikk for CO_2-kjøling av naturgass.

Modellen bør ha et brukervennlig grensesnitt med tanke på evnetuell videreutvikling og endring. Videre legges det vekt på enkel og rask visualisering av simuleringsresultatene. Matlab-pakken Simulink er et aktuelt verktøy i denne sammenheng, i tillegg til at det gir god oversikt over løsningsgang og sammenkobling av ulike deler i prosessen.

Innledningsvis gis det en oversikt over teorien som ligger til grunn for studiet. Videre følger en beskrivelse av den simulerte prosessen i tillegg til selve modelleringen. Resultatene fra modelleringen og simuleringene diskuteres deretter.

2 Bakgrunn

Ved prosessering av naturgass på et flytende anlegg, er karbondioksid et fordelaktig kjølemedium. I det følgende oppsummeres de mest sentrale punktene i denne forbindelse. For mer detaljerte beskrivelser henvises det til forprosjektet[1]. En innføring i nødvendige termodynamiske relasjoner samt dynamisk simulering gis deretter. Aktuelle tilstandslikninger samt programmene brukt i simuleringene nevnes til slutt.

2.1 Prosessering av naturgass

Naturgass består av store deler metan, samt etan, propan, butan og noen tyngre hydrokarboner, og dannes ved at organisk materiale blir utsatt for høye trykk og temperaturer over lengre tid. Omtrent en fjerdedel av all naturgass produsert hvert år omsettes som flytende naturgass (LNG). I 2002 tilsvarte dette 120 millioner tonn.[2][3]

2.1.1 Kondensasjonsprosessen

Naturgass gjøres flytende under trykk, der trykket avhenger av tilstanden i gassreservoaret. Nødvendig energimengde for kondensasjonsprosessen er mindre ved høyt enn ved lavt trykk. På eksisterende anlegg holder naturgassen et trykk på 50 til 70 bar under væskedannelsen. Temperaturintervallet i kondensasjonsprosessen avhenger av molvekten til gassen.

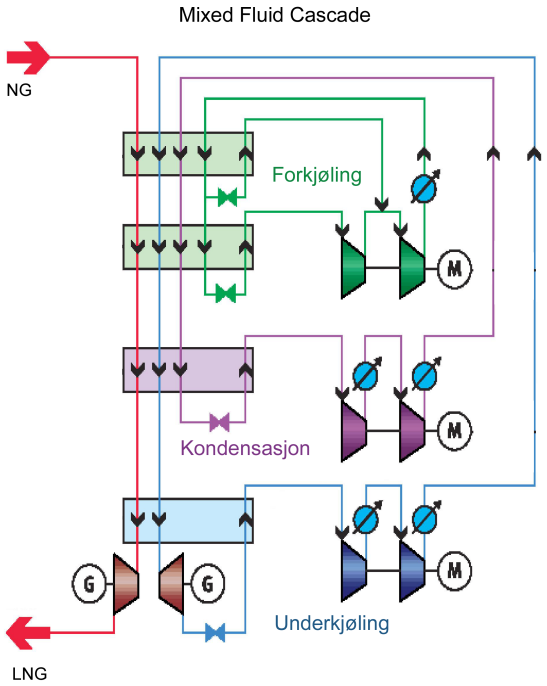
Av økonomiske og sikkerhetsmessige grunner må flytende naturgass transporteres ved atmosfærisk trykk, noe som krever en temperatur lavere enn kokepunktet til metan (-161,49°C). Derfor underkjøles den flytende gassen før ekspansjon til atmosfærisk trykk for å hindre fordamping under transport.

Før LNG kan anvendes til energiformål, må den fordampes/varmes opp. Dette kan kombineres med en prosess med kjølebehov, slik at noe av energien brukt ved kondesering av naturgassen kan utnyttes.[4]

MFC-prosessen (Mixed Fluid Cascade) er grunnlaget for Statoil/Lindes forskning i forbindelse med flytende LNG-anlegg (FLNG). Denne prosessen benytter flere kjølemedier i serie. Kjøling over flere trinn gir lavt eksergitap[1] på grunn av lavere temperaturforskjeller i hvert trinn. I tillegg er muligheten

[1] Eksergi er energiformer som kan omdannes fullstendig til andre energiformer, i dette tilfellet varme som kan tas ut som arbeid. Motsatt er anergi, unyttig energi.[5]

for optimal tilpasning av temperaturprofiler stor når valg av kjølemedium kan varieres i ulike trykknivå. Denne kondensasjonsprosessen tas i bruk i Snøhvit-prosjektet utenfor Hammerfest. En skisse av en MFC-prosess med totrinns forkjøling er vist i figur 2.1.



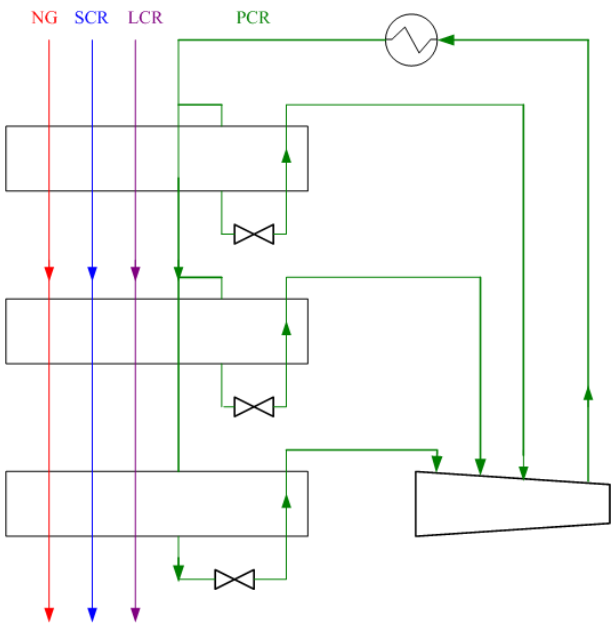
Figur 2.1: MFC-prosessen, utviklet av Linde Statoil LNG Technology Alliance. Forkjølingen er i dette tilfellet inndelt i to trinn.[6]

Prosessen involverer tre sykluser med ulike kjølemedier i kaskade. En blanding av propan og etan (hovedsakelig etan) forkjøler naturgassen i to trinn til ca -50°C. Gassen gjøres deretter flytende under trykk ved ca -80°C ved hjelp av etan i blanding med noe metan og propan, før den underkjøles med metan, etan og nitrogen til -162°C. [7][6]

2.1.2 Forkjølingskretsen

Et flytskjema over en tretrinns forkjølingsprosess med CO_2 som kjølemedium er vist i figur 2.2.

Forkjølingen skjer over tre trinn med varierende trykk på kjølemediet. Naturgassen (NG) samt mediene i kretsene for kondensasjon (LCR) og underkjøling



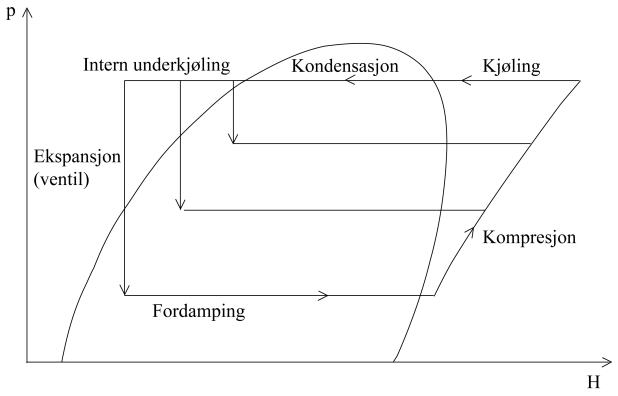
Figur 2.2: Tretrinns forkjølingsprosess med CO_2 som kjølemedium.

(SCR) er varme strømmer i tre seriekoblede plate-finnevekslere. I tillegg underkjøles også deler av forkjølingsmediet (PCR) internt. Resten ekspanderes og sendes gjennom vekslerne som kjølemedium, der det fordampes ved interaksjon med de varme strømmene. Kjølemediet komprimeres deretter og kondenseres og underkjøles ved hjelp av vann før det igjen er klart for intern underkjøling/ekspansjon.

Bare den delen av kjølemediet som går videre til neste LNG-veksler underkjøles. Siden CO_2 består av kun én komponent, har det ingen termodynamisk betydning om forkjølingsprosessen er utformet slik eller som en hydrokarbonprosess, så lenge ventiler benyttes til trykkavspenning. (Ved turbinekspansjon er forholdene annerledes[2]). Valg av struktur har imidlertid betydning med tanke på økonomi og vekt, siden minst mulig areal i varmevekslerne er ønskelig. Et anlegg med CO_2 vil derfor bli utformet med bakgrunn i samme prinsipp som figuren over, der kun deler av kretsen underkjøles fullstendig. [8]

Prosessen kan beskrives termodynamisk i et trykk-entalpidiagram som vist i figur 2.3.

[2]Dette er kommentert nærmere i forprosjektet [1]



Figur 2.3: pH-diagram for en tretrinns forkjølingsprosess.

2.1.3 Flytende LNG-anlegg

En LNG-lekter på størrelse med for eksempel Snøhvit-anlegget vil bli svært kostbar. I tillegg kan det være vanskelig å håndtere et slikt anlegg ved ekstreme værforhold. Det betyr at et flytende anlegg må være svært kompakt i forhold til et stasjonært, noe som medfører økt økonomisk og praktisk sensitivitet overfor følgende to faktorer: Størrelsen på prosessenhetene og andelen av sikkerhetsmessig ufarlige gasser.

Dersom CO_2 benyttes som kjølemedium istedenfor hydrokarboner, kan størrelsen på prosessutstyr som rør og kompressorer reduseres med omtrent 50%. Dette er fordi metningstrykket til CO_2 er omtrent det dobbelte av av en blanding av propan og etan, slik at operasjonstrykket i en kjølekrets fordobles. Volumstrømmen av CO_2 vil derfor halveres i forhold til en hydrokarbonstrøm.[9] Dette vil ha betydning for kostnader og konstruksjon av flytende anlegg, siden modulen for hele kondensasjonsprosessen på en LNG-lekter utgjør omtrent 20% av total vekt.

Det er aktuelt å benytte karbondioksid kun i forkjølingen av LNG-prosessen, siden de påfølgende kjøleprosessene krever lavere temperaturer enn det som er mulig å oppnå med CO_2.

Siden prosessenhetene på et FLNG-anlegg vil stå tett sammen, stilles det strenge krav til sikkerhet. Eksempelvis må boligblokken på anlegget være plassert nærmere prosessen enn på Snøhvit-anlegget på Melkøya, der boligområdet ligger flere hundre meter fra prosessanlegget. Rømningsmulighetene på et flytende anlegg vil være små sammenliknet med et landanlegg. Et gassutslipp vil derfor være en større trussel på et offshore-anlegg enn på land, noe som favorisere bruk av CO_2 til havs: Karbondioksid er en ikke-toksisk gass,

og den er heller ikke brennbar. Propan og etan er derimot en lett brennbare og eksplosive gasser. Både CO_2 og propan/etan er tyngre enn luft, slik at et eventuelt utslipp vil bli liggende langs bakken istedenfor å stige til atmosfæren. Hydrokarboner er imidlertid en kilde til brann, mens CO_2, som delvis vil gå over i fast form (tørris), virker kvelende.[10]

En oversikt over de viktigste fysikalske egenskapene til karbondioksid er gitt i Tabell 2.1.

Tabell 2.1: Fysikalske data for CO_2[11]

Molar vekt M [g/mol]	Frysepunkt T_{fp} [°C]	Kritisk temperatur T_c [°C]	Kritisk trykk P_c [bar]	Asentrisk faktor ω
44,010	-56,6	30,9	79,0	0,239

Det laveste trykket i hydrokarbonkretsen vil være rundt 4 bar, mot tilsvarende 6 bar i en CO_2-krets. Trykkforholdene i en prosess med CO_2 vil derfor være relativt store. Ved høyt trykkforhold vil følgende to problemer forekomme:

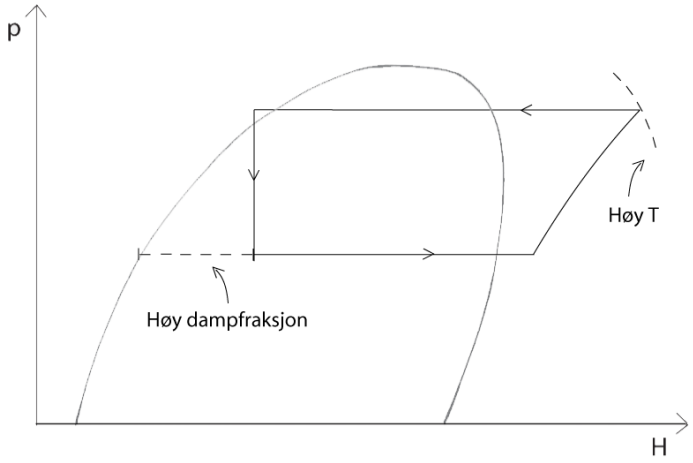
- Ekspansjonen over ventil gir en høy andel gass, noe som er ugunstig med tanke på at gass i liten grad bidrar til kjøling i den påfølgende LNG-veksleren.
- Kompresjonsarbeidet blir høyt og kan i tillegg føre til at den komprimerte strømmen får uønsket høy temperatur.[12]

Dette er vist (for kun ett trinn av prosessen) i figur 2.4.

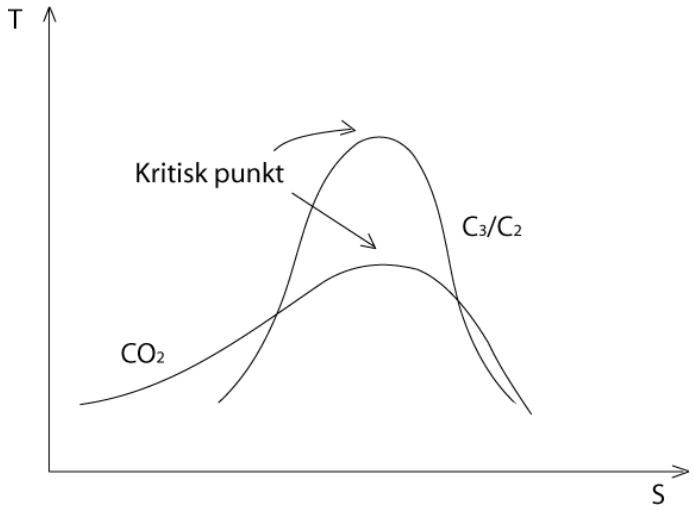
Ved underkjøling før ekspansjon, kan gassandelen etter ventilen reduseres.

Karbondioksid har en lav kritisk temperatur (31°C), noe som fører til at strupetapet i en CO_2-kjølekrets blir høyt i forhold til i en propan/etankrets. Ved å studere et temperatur-entropidiagram for CO_2 og propan/etan, kan denne sammenhengen forklares nærmere. En prinsippskisse i figur 2.5 på side viser formen på tofaseområdet for de to kjølemediene.

Kurven for CO_2 til venstre i diagrammet er ikke så bratt som den for hydrokarbonene. Dette indikerer en relativt høy spesifikk varmekapasitet, C_p, for CO_2 i væskeform.

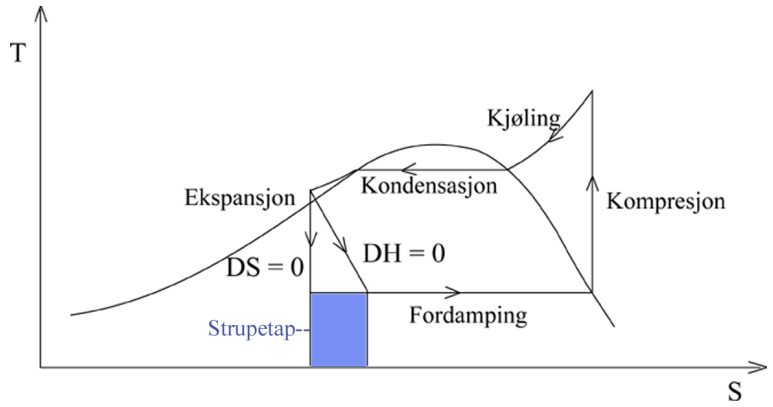


Figur 2.4: Problemer ved høyt trykkforhold



Figur 2.5: Tofaseområdet for CO_2 og propan/etan

Jo høyere varmekapasiteten til væsken er, desto større vil dampandelen etter ekspansjonen bli. Økt gassfraksjon resulterer i økt sirkulert mengde av kjølemediet og dermed økt arbeid, samtidig som kuldeytelsen reduseres. Tapt energi ved struping er vist som det skraverte området i Figur 2.6.[8]



Figur 2.6: Strupetap i forkjølingskretsen

Følgende tiltak kan innføres for å redusere kraftforbruket ved CO_2-kjøling:

- Mellomkjøling ved kompresjon
- Ekstra kjøletrinn
- Ekspansjon i turbin istedenfor ventil
- Ekstern underkjøling

Tiltakene benyttes ikke i denne oppgaven. En nærmere beskrivelse av de ulike effektene kan imidlertid leses i rapporten til forprosjektet[1].

2.2 Termodynamikk

Ulike former for energi er nyttige i forskjellige sammenhenger ut fra hvilke betingelser som er gitt. Energiformene kan utledes fra den generelle energibalansen. For beskrivelse av den termodynamiske oppførselen for ulike fluider, benyttes tilstandslikninger.

2.2.1 Energibalansen og termodynamikkens fundamentale likning

Den generelle energibalansen for et åpent system[3] er gitt av:

$$\Delta E = E_f - E_0 = E_{inn} - E_u t + Q + W \tag{2.1}$$

Indeks 0 og f angir henholdsvis start- og slutt-tilstand for systemet. Q og W er tilført varme og arbeid fra omgivelsene. Energien E består av kinetisk, potensiell og indre energi. For de aller fleste prosesser neglisjeres kinetisk og potensiell energi, slik at energibalansen uttrykt med indre energi U er:

$$\Delta U = U_{inn} - U_{ut} + Q + W \tag{2.2}$$

Dette er termodynamikkens første lov. For en liten endring i et lukket system, kan (2.2) skrives

$$\mathrm{d}U = \mathrm{d}Q + \mathrm{d}W \tag{2.3}$$

De kanoniske variablene til indre energi er entropi S, volum V og moltall $\mathbf{n}$. Det vil si at U uttrykkes på enklest måte når dette variabelsettet er kjent. (2.5) uttrykker det totale differensialet til indre energi.

$$U = U(S, V, \mathbf{n}) \tag{2.4}$$

$$\mathrm{d}U = \left(\frac{\partial U}{\partial S}\right)_{V,\mathbf{n}} \mathrm{d}S + \left(\frac{\partial U}{\partial V}\right)_{S,\mathbf{n}} \mathrm{d}V + \left(\frac{\partial U}{\partial \mathbf{n}}\right)_{S,V} \mathrm{d}\mathbf{n} \tag{2.5}$$

[3] Et åpent system kan utveksle både masse og varme med omgivelsene

Ved å benytte definisjonen på entropi,

$$\mathrm{d}S = \frac{\delta Q_{rev}}{T} \tag{2.6}$$

og at

$$\mathrm{d}W = -p\,\mathrm{d}V \tag{2.7}$$

for en reversibel prosess, ser en ved å sammenlikne (2.3) og (2.5) at $\mathrm{d}U$ kan skrives:

$$\mathrm{d}U = T\mathrm{d}S - p\,\mathrm{d}V + \mu^T\,\mathrm{d}\mathbf{n} \tag{2.8}$$

Dette er den fundamentale likningen for et termodynamisk system. Temperatur, trykk og kjemisk potensial defineres dermed:

$$T = \left(\frac{\partial U}{\partial S}\right)_{V,\mathbf{n}} \tag{2.9}$$

$$-p = \left(\frac{\partial U}{\partial V}\right)_{S,\mathbf{n}} \tag{2.10}$$

$$\mu_i = \left(\frac{\partial U}{\partial n_i}\right)_{S,V,n_{j\neq i}} \tag{2.11}$$

Indeks i angir her komponentnummer.[13][14]

U er Eulerhomogen av første orden[4] med hensyn på tilstandsvariablene (S,V,$\mathbf{n}$), og (2.8) kan dermed integreres til:

$$U = TS - PV + \mu^T\mathbf{n} \tag{2.12}$$

[4] U er proporsjonal med hensyn på S, V og $\mathbf{n}$. For nærmere forklaring av Eulerhomogenitet, se [13]

2.2.2 Energiformer

Med utgangspunkt i uttrykket for dU i (2.8) kan likninger med andre variabelsett enn (S,V,$\mathbf{n}$) utledes. Noen av disse er vist i Tabell 2.2. Utledningene er gjort ved hjelp av Legendre-transformasjoner som vist i Vedlegg A.[13]

Tabell 2.2: Energiformer

Navn	**Differensiell form**	**Integrert form**	**Variable**
Indre energi	$\mathrm{d}U = T\mathrm{d}S - p\mathrm{d}V + \mu^T\mathrm{d}\mathbf{n}$	$U = TS - pV + \mu^T\mathbf{n}$	$(S, V, \mathbf{n})$
Helmholtz' energi	$\mathrm{d}A = -S\mathrm{d}T - p\mathrm{d}V + \mu^T\mathrm{d}\mathbf{n}$	$A = -pV + \mu^T\mathbf{n}$ $= U - TS$	$(T, V, \mathbf{n})$
Entalpi	$\mathrm{d}H = T\mathrm{d}S + V\mathrm{d}p + \mu^T\mathrm{d}\mathbf{n}$	$H = TS + \mu^T\mathbf{n}$ $= U + pV$	$(S, p, \mathbf{n})$
Gibbs energi	$\mathrm{d}G = -S\mathrm{d}T + V\mathrm{d}p + \mu^T\mathrm{d}\mathbf{n}$	$G = \mu^T\mathbf{n}$ $= U - TS + pV$	$(T, p, \mathbf{n})$

Endring i entalpi i et system, ΔH, er lik utvekslet varmemengde mellom system og omgivelser ved konstant trykk. (For prosesser med konstant volum er denne varmemengden lik endring i indre energi). Endring i Gibbs energi, ΔG, og Helmholtz' energi, ΔA, er maksimalt arbeid som kan tas ut av en prosess som går ved henholdsvis konstant trykk og volum.[15]

Med bakgrunn i de andre energiformene, kan balansen for indre energi i (2.2) omformes. Arbeidet W består av strømningsarbeid og volumendringsarbeid (andre former, som akselarbeid, neglisjeres):

$$\begin{aligned} W &= W_{flow,inn} - W_{flow,ut} + W_{\Delta V} \\ &= p_{inn}V_{inn} - p_{ut}V_{ut} - \int_{V_0}^{V_f} p_{ex}\mathrm{d}V \end{aligned} \tag{2.13}$$

I en prosess uten volumendring strykes det siste leddet. Definisjonen på entalpi fra Tabell 2.2 gir dermed følgende uttrykk for energibalansen dersom

volumet er konstant:

$$\Delta U = H_{inn} - H_{ut} + Q \tag{2.14}$$

I prosesser der temperatur og volum er gitt, benyttes ofte Helmholtz' energi i termodynamiske beregninger. For eksempel kan entalpi i henhold til Tabell 2.2 uttrykkes fra S og μ, som sammen med p er gitt av de deriverte til A med hensyn på $(T, V, \mathbf{n})$:

$$-S = \left(\frac{\partial A}{\partial T}\right)_{V,\mathbf{n}} \tag{2.15}$$

$$-p = \left(\frac{\partial A}{\partial V}\right)_{T,\mathbf{n}} \tag{2.16}$$

$$\mu_i = \left(\frac{\partial A}{\partial n_i}\right)_{t,V,n_{j\neq i}} \tag{2.17}$$

2.2.3 Faselikevekt

Likevektstilstanden til et system kan bestemmes ved å maksimere den totale entropien. Termodynamikkens andre lov sier at økning i total entropi S tilstrebes:

$$\Delta S + \Delta S_{sur} \geq 0 \tag{2.18}$$

Ved konstant temperatur har vi fra (2.6) at:

$$\Delta S_{sur} = -\frac{Q_{rev}}{T} \tag{2.19}$$

Dersom trykket også er konstant, og systemet er lukket, er $Q_{rev} = \Delta H$. Dette kan sees fra energibalansen i likning (2.14). Ved å kombinere (2.19) og (2.18) med definisjoner fra Tabell 2.2, fås:

$$\begin{aligned} \Delta S - \frac{\Delta H}{T} &\geq 0 \\ \Delta G &\leq 0 \end{aligned} \tag{2.20}$$

Det vil si at alle naturlige prosesser ved gitt T og p vil søke å oppnå lavest mulig Gibbs energi. Likevektsbetingelsen er dermed $\Delta G = 0$.

I Vedlegg B er det vist at to faser α og β i likevekt vil ha lik temperatur, trykk og kjemisk potensial.

$$T^{\alpha} = T^{\beta} \tag{2.21a}$$
$$p^{\alpha} = p^{\beta} \tag{2.21b}$$
$$\mu_i^{\alpha} = \mu_i^{\beta}; \qquad \forall \quad i \in [1, n] \tag{2.21c}$$

De tre likningene over viser kravene til henholdsvis termisk, mekanisk og kjemisk likevekt.

2.2.4 Tilstandslikninger

En tilstandslikning beskriver forholdet mellom ulike makroskopiske målbare egenskaper til et system. For å beregne en fysisk tilstand til et fluid, relateres trykk, temperatur, volum og antall atomer til hverandre. Den ideelle gasslov er gitt ved

$$pV = nRT \tag{2.22}$$

der p er trykk, V er volum, n er moltall, R er gasskonstanten og T er temperatur.

Reelle gasslover forsøker å beskrive den virkelige oppførselen til en gass bedre enn den idelle gasslov, ved å ta hensyn til tiltrekkende og frastøtende krefter mellom molekylene. Tilstandslikningene er bestemt empirisk eller fra modeller, og det finnes mange ulike typer som er sterke på hver sine områder.[16]

Peng-Robinson

Peng-Robinsons kubiske tilstandslikning er ofte brukt som grunnlag i termodynamiske beregninger for hydrokarboner.

Den generelle formen for kubiske tilstandslikninger er gitt i likning (2.23)

$$p = \frac{RT}{V-b} - \frac{a}{V^2 + ubV + wb^2} \tag{2.23}$$

der $u = 2$ og $w = -1$.

Parametrene a og b er gitt ved:

$$b = \frac{0.07780RT_c}{P_c}$$

$$a = \frac{0.45724R^2T_c^2}{P_c}[1 + f\omega(1 - T_r^{1/2})]^2$$

der

$$f\omega = 0.37464 + 1.54226\omega - 0.26992\omega^2$$

ω er asentrisk faktor:

$$\omega = -logP_r^{sat}(T_r = 0.7) - 1.000$$

Man trenger altså redusert damptrykk ($P^{sat} = P_r^{sat}P_c$) ved $T_r = T/T_c$ for å beregne ω. Asentrisk faktor øker med molekylær vekt og polaritet. [17]

Span-Wagner

Span-Wagners tilstandlikning er av typen empirisk multiparameter og har en optimalisert funksjonell form.

Denne typen tilstandslikning er suveren når det gjelder nøyaktighet, oppførsel i kritiske områder, ekstrapolering og pålitelighet for egenskaper som enten er vanskelig å beskrive eller mangler data.

Span og Wagners tilstandslikning har grunnlag i en optimeringsalgoritme som tar for seg datasett for ulike stoffer simultant. Dermed kan en tilstandslikning utviklet fra for eksempel data for polare fluider, benyttes til å beskrive

oppførselen til et polart fluid som i utgangspunktet har et mangelfullt datasett.

Tilstandslikningen på redusert, trykkeksplisitt form er gitt i likning (2.24).

$$\frac{p}{\rho RT} = \sum_{i=1}^{I_{Pol}} n_i \tau^{t_i} \delta^{d_i} + \sum_{i=I_{Pol}+1}^{I_{Pol}+I_{Exp}} n_i \tau^{t_i} \delta^{d_i} exp(-\gamma \delta^2) \tag{2.24}$$

Her er τ invers redusert temperatur T_c/T, δ er redusert tetthet ρ/ρ_c og R gasskonstanten. I_{Pol}, I_{Exp}, t_i og d_i er funksjonelle former og n_i er substansspesifikke koeffisienter. γ er i de fleste tilfeller lik 1.

En nærmere beskrivelse av tilstandslikningen er gitt i Vedlegg C. Her er i tillegg den generelle formen for polare fluider gitt. For ytterligere informasjon om Span-Wangers tilstandslikning, henvises det til R. Span Multiparameter Equations of State.[18]

2.3 Dynamisk simulering

Dynamisk simulering har hatt en rask utvikling siden 80-tallet og brukes blant annet til prosessdesign, regulering, opplæring og som kontrollromshjelpemiddel.

2.3.1 Differensial-algebraiske likninger

Dynamisk modellering av prosesstekniske systemer leder til et sett med differensial-algebraiske likninger, DAE. Et DAE-sett består av ordinære differensiallikninger, ODE, og algebraiske likninger, AE. Likningene kan skrives på den generelle formen:

$$\begin{aligned} \mathbf{f}(\dot{\mathbf{y}}, \mathbf{y}, \mathbf{z}, \mathbf{u}, t) &= \mathbf{0} \\ \mathbf{g}(\mathbf{x}, \mathbf{y}, \mathbf{u}, t) &= \mathbf{0} \end{aligned} \tag{2.25}$$

Her er $\mathbf{y}$ og $\mathbf{z}$ ukjente og deriverbare variabler, $\mathbf{u}$ er eksplisitt gitte funksjoner og t er tiden. Differensiallikningene kommer normalt fra masse- og energibalanser, mens de algebraiske likningene eksempelvis kan være termodynamiske beskrankninger, volumbalanser og masse-/varmetransport. [19]

På tilstandsromform kan likningene skrives:

$$\begin{aligned} \dot{\mathbf{y}} &= \mathbf{f}(\mathbf{y}, \mathbf{z}, \mathbf{u}, t) \\ \mathbf{0} &= \mathbf{g}(\mathbf{y}, \mathbf{z}, \mathbf{u}, t) \end{aligned} \tag{2.26}$$

2.3.2 Numerisk integrasjon

For løsning av et sett med ODE-likninger, kan flere numeriske metoder benyttes. Løsningteknikker for DAE-sett er utvidelser av metodene for ODE-løsning, og kan medføre problemer med høyere indeks[5] og valg av gode startverdier. Detaljerte beskrivelser kan leses i [19].

[5] Indeks er antall ganger DAE-likninger må deriveres for at de skal transformeres til et ODE-system. Høyere indeks betyr indeks større enn 1. [19]

Tre ulike metoder for numerisk løsning av DAE-systemer er nevnt i [19]:

- **Substitusjon av de algebraiske variablene** i differensiallikningene. For små systemer der de algebraiske variablene lett kan uttrykkes eksplisitt.
- **Eksplisitt ODE-løser.** De algebraiske likningene løses internt for hvert tidssteg. For stive systemer kan metoden medføre lang regnetid dersom løsningen av disse likningene er tidkrevende.
- **Implisitt DAE-løser.** Algebraiske og differensielle likninger løses simultant.

En numerisk metode er eksplisitt dersom man kun trenger opplysninger fra fortiden for beregning av løsning ved tiden t. Dette kan illustreres med Eulers metode, som bruker Taylorrekken til $\mathbf{x}$ om t og utelukker alle ulineære ledd:

$$\mathbf{y}(t) \approx \mathbf{y}(t-h) + \mathbf{f}(\mathbf{y}, t-h) \cdot h \tag{2.27}$$

Her er h steglengden. Dersom $\mathbf{f}$ kalkuleres ved t istedenfor t-h, er metoden implisitt, og det er nødvendig med iterasjon for beregning av løsning i t.

En skiller også mellom enkelt- og multisteg-metoder. En multisteg-løser bruker informasjon fra flere løsningspunkter fra fortiden for å beregne verdien i nåtid, mens en enkeltsteg-løser kun benytter opplysninger fra forrige løsning.

2.3.3 Feil og orden

En lokal numerisk feil oppstår som følge av et iterasjonssteg der det gjøres en avrundings- eller avkuttingsfeil. Avrundingsfeil fås fordi alle tall representeres med et endelig siffer, mens en avkuttingsfeil er synonymt med tilnærming til en Taylorrekke.

En global feil er feilen man står igjen med etter at alle iterasjonsstegene er fullført. Den globale feilen er alltid av en orden lavere enn den lokale. Med orden menes den faktor feilleddet er opphøyd i. [20]

2.3.4 Newtons metode

Newtons metode er en iterativ metode for generering av en approksimasjon $\mathbf{y}$ til en løsning av $\mathbf{g(y)=0}$. Til dette brukes den deriverte av $\mathbf{g}$, som kan uttrykkes ved tilnærming til en Taylorrekke:

$$\mathbf{g}'(\mathbf{y}_k) = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right|_k \approx \frac{\mathbf{g}(\mathbf{y}_k) - \mathbf{0}}{\mathbf{y}_k - \mathbf{y}_{k+1}} \tag{2.28}$$

Løst med hensyn på $\mathbf{y}_k$ gir dette:

$$\mathbf{y}_{k+1} = \mathbf{y}_k - \frac{\mathbf{g}(\mathbf{y}_k)}{\mathbf{g}'(\mathbf{y}_k)} \tag{2.29}$$

Problemet initialieres med hensiktsmessige startverdier $\mathbf{y^0}$, og det itereres inntil avviket mellom $\mathbf{y}_{k+1}$ og $\mathbf{y}_k$ er tilstrekkelig lite.

Metoden er av andre orden fordi differensialene er de samme som første ordensledd i en Taylorrekke.

2.4 Matlab

Matlab står for Matrix Laboratory og er et interaktivt program med tekstbasert grensesnitt. Det brukes til tekniske numeriske beregninger og har mulighet for visualisering av resultater. Beregningene er baserert på matriser og vektorer. Fordelen med et slikt program er muligheten til selv å ha full kontroll over simuleringen, siden alle parametre må defineres av brukeren og alle funksjoner kan bygges opp fra grunnen. Det finnes i tillegg ferdig utviklede pakker som kan forenkle implementasjonen.[21]

2.4.1 ODE-løsere i Matlab

Ode45 er basert på enkeltsteg og benyttes ofte ved første forsøk på løsning av de fleste problemer. Dersom systemet som skal løses består av algebraiske likninger i tillegg til differensielle, eller dersom det er stivt og ode45 er ineffektiv, er det vanlig å bruke ode15s. Dette er en lineær implisitt multistegløser. Metoden er basert på BDF (the Backward Differentiation Formulae)

og er av varierende orden. Ode15s er god for beregning av stive systemer fordi steglengden tilpasses behovet for nøyaktighet i løsningen. Ulemper med ode15s er usikker oppførsel ved ustabile eller oscillerende problemer. Andre ODE-løsere er 23-serien. Et sammendrag av egenskapene til de ulike løserne er gitt i Tabell 2.3.

Tabell 2.3: Ulike ODE-løsere. Opplysningene er hentet fra Matlab[21]

Løser	Type problem	Nøyaktighet	Brukes
ode45	Ikke stivt	Medium	Oftest. Vanlig å prøve først
ode23	Ikke stivt	Liten	Ved høy feil-toleranse/moderat stive probl
ode113	Ikke stivt	Liten-stor	Ved lav feil-toleranse/regnekrevende ode-problem
ode15s	Stivt	Liten-medium	Når ode45 er sen
ode23s	Stivt	Liten	Ved høy feil-toleranse og konstant M-matrise
ode23t	Moderat stivt	Liten	Ved behov for løsning uten numerisk demping
ode23tb	Stivt	Liten	Ved høy feil-toleranse

2.4.2 Simulink

Simulink er en software-pakke i Matlab for modellering og simulering av dynamiske systemer. En modell bygges opp som et blokkdiagram og beskriver grafisk tidsavhengige matematiske sammenhenger mellom systemets pådrag, tilstander og målinger. Det er flere typer ferdigdefinerte bokser i programmet. Eksempler er tilstandsromblokker og Laplace-transformatorer, som er nyttige i forbindelse med regulering av prosesser.[21]

Det er også mulig å velge brukerdefinerte blokker med s-funksjoner til å beskrive deler av systemet. Disse rutinene krever en initialiseringsfil og består av ulike funksjoner blant annet for kalkulasjon av deriverte og utgangsverdier. Ulike typer ODE-løsere kan velges for det dynamiske problemet. Løsning av DAE-systemer krever imidlertid en mer komplisert prosedyre.

En fordel med Simulink er det grafiske grensesnittet, som gjør den modellerte prosessen oversiktlig. Simuleringene er interaktive, slik at det er mulig å endre parametre og straks se resultatene ved hjelp av grafer. Verktøyet er modulbasert, det vil si at et element kan bygges opp èn gang og deretter brukes flere ganger i serie.

Simulink har enkelte begrensninger, for eksempel når det gjelder signalstrømmene. Det er ikke mulig å definere flere innganger til en boks. På denne måten er det relativt enkelt å få oversikt over løsningsprosedyren i et problem, mens det som skjer i selve prosessen ikke direkte framkommer av flytskjemaet. I tillegg må det ofte legges inn rask dynamikk i algebraiske løkker for å bedre den numeriske løsbarheten. Detter er ekstra regneoperasjoner som kan kreve mye tid.

2.5 Hysys

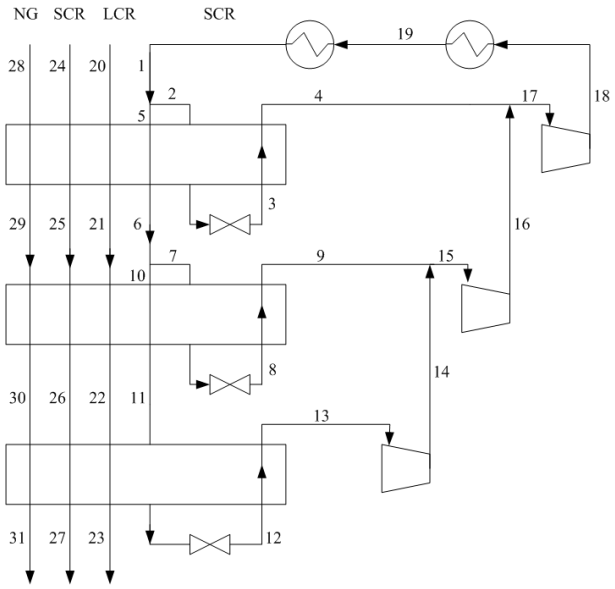
Hysys er et program for modellering og simulering av prosesser. Brukergrensesnittet er grafisk, der forhåndsdefinerte enhetsoperasjoner kan kombineres til ønsket prosess. Det er også mulig å velge ulike termodynamiske modeller som grunnlag for simuleringene.[22] Hysys er brukervennlig på grunn av god dokumentasjon og enkle prosedyrer for komponering av prosesser. Det er imidlertid en ulempe at det ikke er mulig å kontrollere og observere alt som skjer i simuleringene fordi endel parametre er forhåndsdefinerte og utilgjengelige for brukeren.

3 Prosessbeskrivelse

Statoil og Lindes MFC-prosess ligger til grunn for den simulerte delen av forkjølingskretsen. Deler av det øverste trykknivået i forkjølingskretsen er modellert i Matlab ved hjelp av Simulink. Det er lagt vekt på å få en detaljert beskrivelse av varmeveksleren, der termodynamisk flash står i fokus.

3.1 Opprinnelig prosess

Det er tatt utgangspunkt i en tretrinns forkjølingskrets med CO_2 som kjølemedium, som beskrevet i avsnitt 2.1.1. Et eksempel på en slik prosess er vist i Figur 3.1. Data for hver nummererte strøm er gjengitt i Vedlegg F.1. Disse er hentet fra Hysys-simuleringer fra forprosjektet.[1]



Figur 3.1: Tretrinns forkjølingsprosess med CO_2 som kjølemedium.

Kretsen er del av en MFC-prosess der produksjonen er 800 tonn LNG i timen, det vil si i underkant av 7 millioner tonn per år. Naturgassen forkjøles fra 8°C til -30°C under et trykk på 70 bar. I det øverste kjølenivået er CO_2-trykket 48 bar før ventilen. Væsken fordamper ved -11°C og ca 25 bar og overhetes deretter, slik at naturgassen har en temperatur på -10°C før påfølgende kjøletrinn.

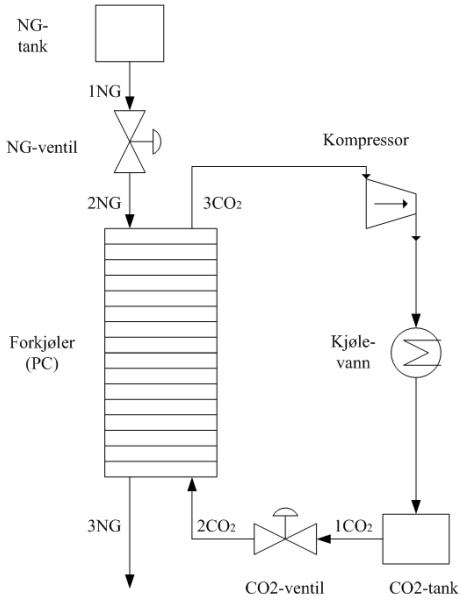
3.2 Endringer og forenklinger

Ekspansjon av kjølemediet samt fordamping av CO_2 ved varmeveksling med naturgass i det øverste trykktrinnet er modellert i Matlab. De to andre varme strømmene er utelatt.

I prosessen inneholder naturgassen ren metan. Det er lagt til muligheter for utvidelse, slik at sammensetningen kan bestå av etan og propan i tillegg. Rutinen for tilstandsberegningene til naturgassen er generert med utgangspunkt i alle tre hydrokarbonforbindelser. Kjølemediet består av CO_2. Siden strømmene inneholder rene komponenter, benyttes heretter N som notasjon for antall mol istedenfor moltallsvektor $\mathbf{n}$.

Modellen gjelder kun tofase på kjølesiden. Derfor er også utstrømmen tofase, i motsetning til en reell prosess, som har overheting av kjølemediet (kompressoren krever ren gass på sugesiden).

Figur 3.2 viser et flytskjema for hvordan kjøletrinnet for den modellerte prosessen ser ut.



Figur 3.2: Modell av øverste kjølenivå.

Forkjøleren er inndelt i m=20 kontrollvolum. Kretsen er modellert slik at volum, temperatur og trykk i tanker før ventilavspenning er gitt (det vil si at entalpi inn på ventil er gitt). Strømmenes baktrykk etter varmeveksling er også gitt.

3.3 Definisjon av variable

Innganger til forkjøleren er massestrøm og entalpi for både naturgass og CO_2. Trykket nedstrøms er nødvendige grensebetingelser, sammen med tanktilstandene. Ventilåpningene (z) for kjølemediet og naturgassen fungerer som pådrag. Tilstandene i forkjøleren er (T, V, N) for naturgass, CO_2-gass og -væske, minus gassvolumet for CO_2 (beregnes ved substitusjon av totalvolum og væskevolum), samt T_{vegg}. Tilsammen er dette $9m$, det vil si 180. En oversikt er gitt i Tabell 3.1.

Tabell 3.1: Definisjon av variable

Navn	Medium	Variable	Antall/delvol	Totalt
Innganger, u	NG:	w, H	2	$5m$
	CO_2:	w^l, w^v, H^t	3	
Grense-	NG:	p_{ut}, $(T, p, V)_{tank}$	4	8
betingelser	CO_2:	p_{ut}, $(T, p, V)_{tank}$	4	
Pådrag	Ventiler	z	-	2
Tilstander, y	NG:	$(T, V, N)_{PC}$	3	
	CO_2:	$(T, V, N)^l_{PC}$, $(T, N)^v_{PC}$	5	$9m$
		T_{vegg}	1	

I Vedlegg F er dimensjonene for enhetene i kjølekretsen vist, sammen med tanktilstander og verdier for varmeoverføringskoeffisienter og ventilkonstanter i forkjøler og ventiler.

4 Modellering

I det følgende gis en beskrivelse av det resulterende likningssettet fra de modellerte enhetene samt løsningsprosedyre.

4.1 Antagelser

I tillegg til å beskrive varmeveksleren rigorøst termodynamisk, er målet for oppgaven å generere en fungerende modell som lett kan gjøres om ved eventuell senere bruk til simulering av spesifikke prosesser. Det er derfor sett på en typisk prosess med CO_2-kjøling med muligheter for å endre strømsammensetninger og betingelser.

Det antas at fluidstrømmene kan beskrives av en ventillikning for væske, som også gjelder for gasser med lite trykkfall.

Hvert delvolum er modellert som en ideell blandetank. Videre er det antatt null trykktap mellom ventiler og varmeveksler. I energibalansene er kinetisk og potensiell energi neglisjert.

En oversikt over antagelsene for modellen er vist under.

- Naturgassen består av metan
- Strømning beskrives med ventillikning for væske
- 20 kontrollvolum modelleres som ideelle blandetanker
- Null trykktap mellom ventiler og forkjøler
- Kinetisk og potensiell energi i energibalansene neglisjeres

4.2 Tilstandslikninger

Fluiders reelle oppførsel nær kritisk punkt er avvikende fra teoretiske beregninger i større grad enn ved lavere trykk og temperaturer. Kjølemediet opereres tett opptil kritisk punkt, og for at de termodynamiske beregningene skal bli mest mulig sikre, er Span-Wagners tilstandslikning benyttet. Denne likningen anses for å være den som best beskriver oppførselen til CO_2.[1]. Peng-Robinsons likning er god for hydrokarboner generelt, og denne er derfor benyttet for naturgassen.

Til grunn for alle tilstandsberegninger ligger de deriverte av Helmholtz' energi med hensyn på det naturlige variabelsettet (T, V, N). Rutinene for beregning av disse er vist i Vedlegg H.3.6 og H.3.7 for henholdsvis CO_2 og naturgass.[23]

Data for CO_2 og metan er hentet fra DIPPR[24]. Vedlegg D viser en oversikt over de deriverte av Helmholtz' energi samt andre nødvendige relasjoner for tilstandsberengingene.

4.3 Modellikninger

I forbindelse med modellering av faselikevekter, er høyere indeks et vanlig problem. Et slikt problem kan oppstå dersom det benyttes balanser for hver fase og likninger for mol og energitransport over fasegrensen. Derfor er modellen basert på felles masse- og energibalanse for begge faser kombinert med faselikevektskrav som formulert i avsnitt 2.2.3.

4.3.1 Balanselikninger

Varmeveksleren er delt inn i m=20 kontrollvolum. For fluidet i hvert kontrollvolum gjelder følgende balanser for energi, volum og masse:

$$\dot{U}^t = \dot{H}^t_{inn} - \dot{H}^t_{ut} + Q^t \qquad [J/s] \qquad (4.1)$$

$$\dot{V}^t = 0 \qquad [m^3/s] \qquad (4.2)$$

$$\dot{N}^t = \dot{N}^t_{inn} - \dot{N}^t_{ut} \qquad [mol/s] \qquad (4.3)$$

Indeks t angir total tilstand, for CO_2-siden vil dette si blandingen av væske og gass.

Massestrømmene mellom kontrollvolumene er beregnet ved hjelp av ventillikningen som beskrevet i avsnitt 4.3.2. Entalpien for en fase beregnes fra:

$$H^s = T \cdot S^s + \mu^s \cdot N^s \qquad [J] \qquad (4.4)$$

der S og μ genereres fra tilstanden (se Vedlegg D for nærmere beskrivelse). $s \in [l, v]$, og l og v betenger henholdsvis væske og gass. Total entalpistrøm er:

$$\dot{H}^t_{ut} = \frac{H^l}{N^l} \cdot \dot{N}^l_{ut} + \frac{H^v}{N^v} \cdot \dot{N}^v_{ut} \qquad [J/s] \qquad (4.5)$$

Overført varme i ett kontrollvolum er gitt av:

$$Q_i = (UA)_i^s \cdot (T_i^{vegg} - T_i^s) \qquad [J/s] \tag{4.6}$$

U er her varmeoverføringskoeffisient og A er totalt areal i varmeveksleren. i betegner kontrollvolum-nummer.

4.3.2 Ventillikning

Fra Bernoullis likning, som gjelder for de fleste væsker samt gasser med lite trykkfall, kan volumstrømmen gjennom en ventil utledes:

$$q = z \cdot C_v \cdot sign(\Delta p) \cdot \sqrt{\frac{|\Delta p|}{\rho}} \qquad [m^3/s] \tag{4.7}$$

C_v er en dimensjonerende ventilkonstant som beregnes fra stasjonær tilstand, og z er ventilåpningen. ρ er fluidets tetthet. [25]

Total massestrøm mellom kontrollvolumene beregnes på samme måte, med ventilåpning lik 1:

$$w^t = C_v \cdot \rho^t \cdot sign(\Delta p) \cdot \sqrt{\frac{|\Delta p|}{\rho^t}} \qquad [kg/s] \tag{4.8}$$

Ved å bentytte at

$$w^s = w^t \cdot \frac{\rho^s V^s}{\rho^t V^t} \tag{4.9}$$

fås følgende uttrykk for massestrømmen til hver av de to fasene på CO_2-siden:

$$w^s = C_v \cdot \rho^s \cdot sign(\Delta p) \cdot \sqrt{\frac{|\Delta p|}{\rho^t}} \tag{4.10}$$

For naturgassen gjelder (4.7) med $z = 1$.

4.4 Algoritmer for varmeveksling

I tilstandsberegningene benyttes rutiner for kalkulasjon av deriverte av Helmholtz' energi, der T, V og $\mathbf{n}$ er det variabelsettet som kreves. Differensiallikningene i systemet uttrykkes med disse variablene.

For tofasestrømmen beregnes de deriverte av T, V^l, V^v, N^l og N^v med Newtons metode, der endringene i (U^t, V^t, N^t) bestemmer retningen på Newtonstegene. Likevektslikningene benyttes som korreksjoner i iterasjonene. Det vil tas ett Newtonsteg for hvert integrasjonssteg, slik at ODE-løseren predikterer en løsning, mens Newtonsteget korrigerer for hvert tidssteg. Denne løsningsformen er en såkalt prediktor korrektor-metode. [23]

4.4.1 CO_2-siden

U, V og N er gitt av balanselikningene. Det er nødvendig å finne en sammenheng mellom disse størrelsene og det termodynamiske rommet T, V, N. Totale differensialer for endringer i U, V og N er gitt av

$$U_k^t + \left.\frac{\partial U^l}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^l + \left.\frac{\partial U^v}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^v = U_{k+1}^t \tag{4.11a}$$

$$V_k^t + \left.\frac{\partial V^l}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^l + \left.\frac{\partial V^v}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^v = V_{k+1}^t \tag{4.11b}$$

$$N_k^t + \left.\frac{\partial N^l}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^l + \left.\frac{\partial N^v}{\partial(T,V,N)^l}\right|_k \delta(T,V,N)^v = N_{k+1}^t \tag{4.11c}$$

der k er iterasjonsindeks. $\delta y = \dot{y} \cdot \delta t$, der δt er et tidssteg.

Likevektskravet vist i avsnitt 2.2.3 kan skrives:

$$T_k^l + \delta T_k^l = T_k^v + \delta T_k^v \tag{4.12a}$$

$$p_k^l + \delta p_k^l = p_k^v + \delta p_k^v \tag{4.12b}$$

$$\mu_k^l + \delta \mu_k^l = \mu_k^v + \delta \mu_k^v \tag{4.12c}$$

Temperaturene i de to fasene settes like. Siden det integreres langs en likevekt der T, V, N er de variablene som kontrolleres, kan ikke nødvendigvis

tilsvarende antas for trykk og kjemisk potensial. Med andre ord settes T_k^l lik T_k^v, mens de to siste likningene i (4.12) fungerer som krav om at $(p_k^l - p_k^v)$ og $(\mu_k^l - \mu_k^v)$ skal gå mot null. De totale differensialene for p og μ kan uttrykkes med hensyn på T, V, N:

$$\delta T_k^l - \delta T_k^v = 0 \tag{4.13a}$$

$$\left.\frac{\partial p^l}{\partial (T,V,N)^l}\right|_k \delta(T,V,N)_k^l + \left.\frac{\partial p^v}{\partial (T,V,N)^l}\right|_k \delta(T,V,N)_k^v = -(p_k^l - p_k^v) \tag{4.13b}$$

$$\left.\frac{\partial \mu^l}{\partial (T,V,N)^l}\right|_k \delta(T,V,N)_k^l + \left.\frac{\partial \mu^v}{\partial (T,V,N)^l}\right|_k \delta(T,V,N)^v = -(\mu_k^l - \mu_k^v) \tag{4.13c}$$

Likevektslikningene over kan sammen med differensiallikningene i (4.11) skrives på matriseform:

$$\mathbf{M}\dot{\mathbf{y}} = \begin{bmatrix} \dot{\mathbf{x}} \\ \Delta\mathbf{b} \end{bmatrix} \tag{4.14}$$

der

$$\dot{\mathbf{y}} = \begin{bmatrix} \dot{T}^l \\ \dot{V}^l \\ \dot{N}^l \\ \dot{T}^v \\ \dot{V}^v \\ \dot{N}^v \end{bmatrix}, \qquad \begin{bmatrix} \dot{\mathbf{x}} \\ \Delta\mathbf{b} \end{bmatrix} = \begin{bmatrix} \dot{U}^t \\ \dot{V}^t \\ \dot{N}^t \\ 0 \\ -(p^l - p^v) \\ -(\mu^l - \mu^v) \end{bmatrix}, \quad \text{og}$$

$$\mathbf{M} = \begin{bmatrix} U_T^l & U_V^l & U_N^l & U_T^v & U_V^v & U_N^v \\ V_T^l & V_V^l & V_N^l & V_T^v & V_V^v & V_N^v \\ N_T^l & N_V^l & N_N^l & N_T^v & N_V^v & N_N^v \\ T_T^l & T_V^l & T_N^l & -T_T^v & -T_V^v & -T_N^v \\ p_T^l & p_V^l & p_N^l & -p_T^v & -p_V^v & -p_N^v \\ \mu_T^l & \mu_V^l & \mu_N^l & -\mu_T^v & -\mu_V^v & -\mu_N^v \end{bmatrix}$$

$$= \begin{bmatrix} U_T^l & U_V^l & U_N^l & U_T^v & U_V^v & U_N^v \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 & 0 & 0 \\ p_T^l & p_V^l & p_N^l & -p_T^v & -p_V^v & -p_N^v \\ \mu_T^l & \mu_V^l & \mu_N^l & -\mu_T^v & -\mu_V^v & -\mu_N^v \end{bmatrix}$$

Her er $\left(\frac{\partial U^l}{\partial T^l}\right)_{V,\mathbf{n}}$ forkortet til U_T^l, og så videre.

For introduksjon av likevekstkorreksjonene til ODE-systemet er det benyttet at $\|\delta\mathbf{y}\| \ll \|\dot{\mathbf{y}}\|$. En nærmere utredning er gitt i Vedlegg E.

Til grunn for den simultane gjennomføringen av Newton- og integrasjonssteg, ligger en antakelse om at den lokale feilen Newtoniterasjonene medfører er mindre enn intergrasjonsfeilen. Newtons metode er som nevnt i avsnitt 2.3.4 av andre orden, slik at den lokale feilen blir av andre orden. Løsningen av ODE-likningene er av varierende orden. Det vil si at det antas at ODE-løseren medfører en feil av orden høyere enn 2, slik at den interne feilen som oppstår ved likevektskorreksjon er av mindre størrelsesorden enn integrasjonsfeilen.

Betydningen av korreksjon for likevekt i hvert tidssteg kan testes ved å sette trykk og kjemisk potensial for de to fasene like.

4.4.2 Naturgass

Den varme strømmen består av ren gassfase, og likningssettet som skal løses for naturgassen kan forenkles til kun å inneholde balanser for indre energi, volum og masse. Endringene i tilstandene T, V og N finnes ved

$$\left(\frac{\partial U^l}{\partial T^l}\right)_{V,\mathbf{n}} \dot{T} + \left(\frac{\partial U}{\partial V}\right)_{T,\mathbf{n}} \dot{V} + \left(\frac{\partial U}{\partial \mathbf{n}}\right)_{T,V} \dot{N} = \dot{U} \tag{4.15}$$

samt balanselikningene for U, V og N. Likningene løses på samme måte som for CO_2, uten likevektsbetningelser:

$$\dot{\mathbf{y}} = \mathbf{M}^{-1}\dot{\mathbf{x}} \tag{4.16}$$

der

$$\dot{\mathbf{y}} = \begin{bmatrix} \dot{T} \\ \dot{V} \\ \dot{N} \end{bmatrix}, \quad \dot{\mathbf{x}} = \begin{bmatrix} \dot{U}^t \\ \dot{V}^t \\ \dot{N}^t \end{bmatrix}, \quad \text{og} \quad \mathbf{M} = \begin{bmatrix} U_T & U_V & U_N \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4.4.3 Skillevegg i varmeveksler

For veggen mellom naturgass og CO_2 i varmeveksleren, benyttes

$$\dot{T}^{vegg} = mC_p^{vegg} \cdot (Q^{NG} - Q^{CO_2}) \tag{4.17}$$

der Q^{NG} er varmen overført fra naturgassen, og Q^{CO_2} er varme overført til kjølemediet.

4.5 Implementering

Det modellerte kjølesystemet utgjør et sett av differensielle og algebraiske likninger. Ved hjelp av uttrykk for de partiellderiverte, substitueres de likevektslikningene inn i de differensielle slik at et nytt ODE-sett genereres. I tillegg beregnes andre algebraiske likninger, som massestrømmene, suksessivt. Det er med andre ord en kombinasjon av den første og den andre metoden nevnt i avsnitt 2.3.2 (substitusjon av AE og eksplisitt ODE-løsning) som benyttes.

Antall differensielle variable i hvert kontrollvolum er lik antall tilstander (se avsnitt 3.3), det vil si $9 \cdot m = 180$ for 20 kontrollvolum.

Trykkdynamikken i en prosess er normalt svært rask sammenliknet med andre endringer som for eksempel forflytning av masse. Siden deler av likevektssystemet består av likninger der trykk inngår, antas det å være relativt stivt. Det betyr at løsere som ode15s trolig er egnet til løsning av likningssystemet.

To ulike løsningsstrategier er testet ut i Simulink.

I Vedlegg H er det gitt en oversikt over matlabrutinene benyttet i modellene. Her er også koden listet i tillegg til at flytskjema fra Simulink er vedlagt.

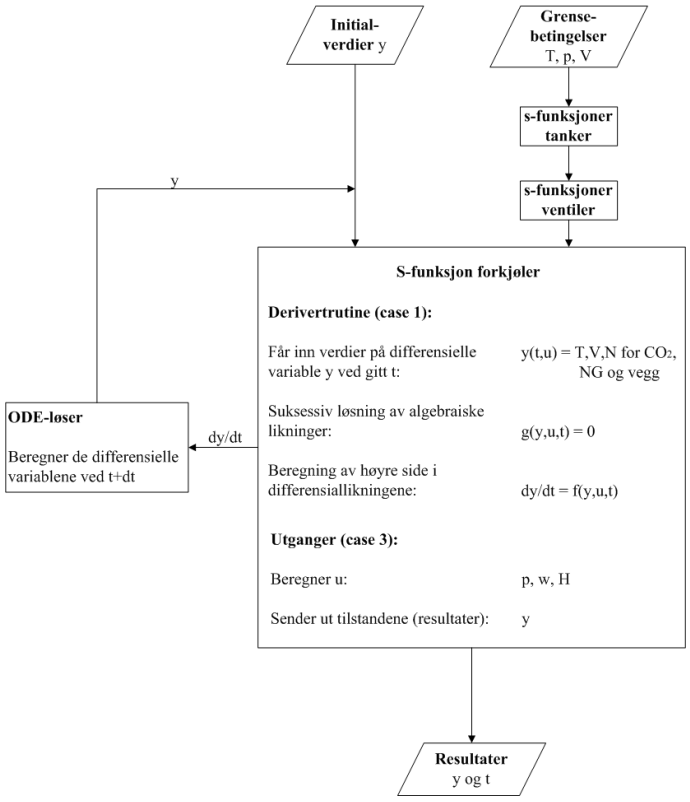
4.5.1 Modell A

I den første modellen består varmeveksleren av kun èn s-blokk i Simulink, som kaller et matlab-funksjon der flash-likningene i alle delvolum løses i ett. Kontrollen på hva som går inn og ut av hvert volum ligger derfor skjult for Simulink. Dette medfører bruk av en for-løkke i skriptene for CO_2 og naturgass.

En skjematisk oversikt over løsningen av modell-likningene i den første modellen er gitt i Figur 4.1. Flytskjema ligger i Vedlegg H.1

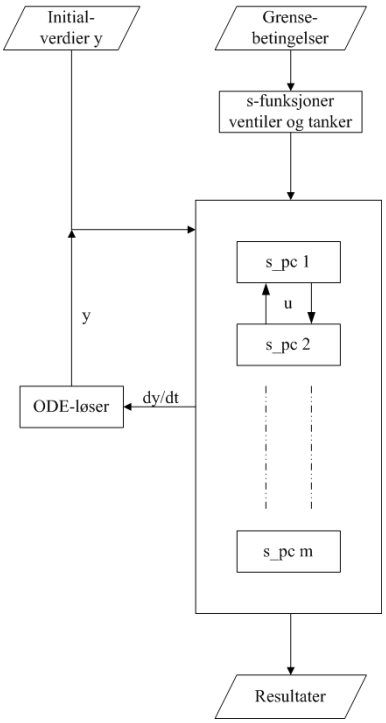
4.5.2 Modell B

Den andre modellen er bygd opp slik at tilstandene i hvert kontrollvolum beregnes separat, som vist i Figur 4.2. Denne modellen benytter seg mer aktivt av muligheten Simulink gir for sammenkobling av blokker, ved at hvert kontrollvolum er linket til en blokk. På denne måten unngås en for-løkke som er nødvendig i Modell A ved beregning av Helmholtz' deriverte for hvert kontrollvolum. For-løkker bruker generelt mye regnetid.



Figur 4.1: Løsningsprosedyre, Modell A

De algebraiske relasjonene mellom delvolumene resulterer i algebraiske løkker. For å bedre den numeriske løsbarheten, er det nødvendig å legge inn rask dynamikk for å bryte disse løkkene (Dette gjøres ved hjelp av initialiserte tilstandsromblokker). I tillegg krever hver s-blokk en ekstra parameter som forteller s-funksjonen hvilket delvolum som behandles. Flytskjema for Modell B er vist i Vedlegg H.2.



Figur 4.2: Løsningsprosedyre, Modell B

4.6 Hysys-modell

En modell i Hysys er generert som sammenlikningsgrunnlag for Matlab-modellene. Peng-Robinsons tilstandslikning er benyttet både for karbondioksid og metan, siden det ikke er mulig å velge Span-Wagners likning som termodynamisk pakke i dette programmet. Denne forskjellen mellom Hysys- og Matlab-modellene gir minimalt utslag, se resultater for Matlab-simuleringene i forprosjektet[1] for nærmere forklaring.

5 Resultater

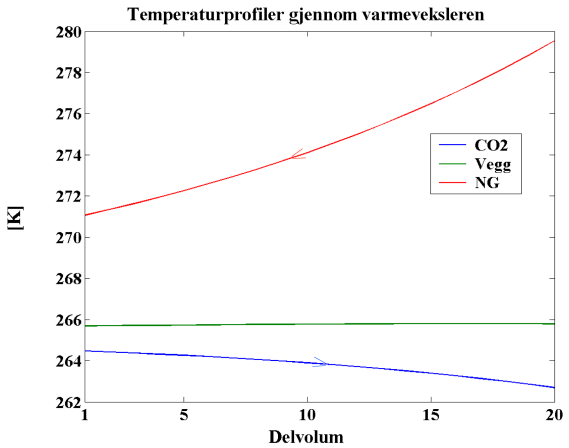
Resultatene fra simuleringer for stasjonær tilstand samt pådrag i ventil med og uten likevektskorreksjon er gitt i det følgende.

5.1 Stasjonær tilstand

Strømdata for stasjonær tilstand er vist i Tabell 5.1. Strømnumrene referer til Figur 3.2 i avsnitt 3.2. Temperaturprofiler for kald og varm side i tillegg til veggen er vist i Figur 5.1. CO_2-temperaturen synker gjennom varmeveksleren på grunn av trykkfall.

Tabell 5.1: Strømdata

Strøm	Flow [kg/h]	Temp [°C]	Trykk [bar]	Gass-fraksjon
1CO2	7,7e4	8,00	42	0
2CO2		-8,00	27,5	0,07
3CO2		-10	26	0,75
1NG	4,5e5	8	71	1
2NG		7,5	69,3	1
3NG		-2	69,0	1



Figur 5.1: Temperaturprofiler gjennom forkjøleren

Data fra Hysysmodellen er gjengitt i Tabell 5.2, der * indikerer hvilke verdier som er satt. En sammenlikning av temperaturer og gassfraksjoner i forhold

til stasjonære data fra Matlab viser at energibalansen over varmeveksleren er samsvarende for modellene fra de to ulike programmene. Hysys-modellen er vist i Vedlegg I.

Tabell 5.2: Strømdata fra Hysysmodellen. Verdier med * er satt

Strøm	Flow [kg/h]	Temp [°C]	Trykk [bar]	Gass-fraksjon
co2_1	7,7e4*	7,5	42*	0*
co2_2		-8,1	27,5*	0,17
co2_3		-10	26*	0,77
NG_1	4,5e5*	8,0*	71*	1
NG_2		7,3	69,3*	1
NG_3		-2*	69,0*	1

5.2 Endring i ventilåpning

CO_2-ventilen ble åpnet 5% etter 100 sekunder under en total simuleringstid på 500 sekunder.

5.2.1 Valg av ODE-løser

Ulike ODE-løsere er testet ut på Modell B. Ingen av løserne for ikke-stive eller moderat stive systemer håndterer steget i ventilåpningen, z.

Ode15s og ode23tb fungerer tilfredsstillende. Regnetiden er imidlertid svært lang. Total tid for simuleringen med steg i z er vist i tabell 5.3.

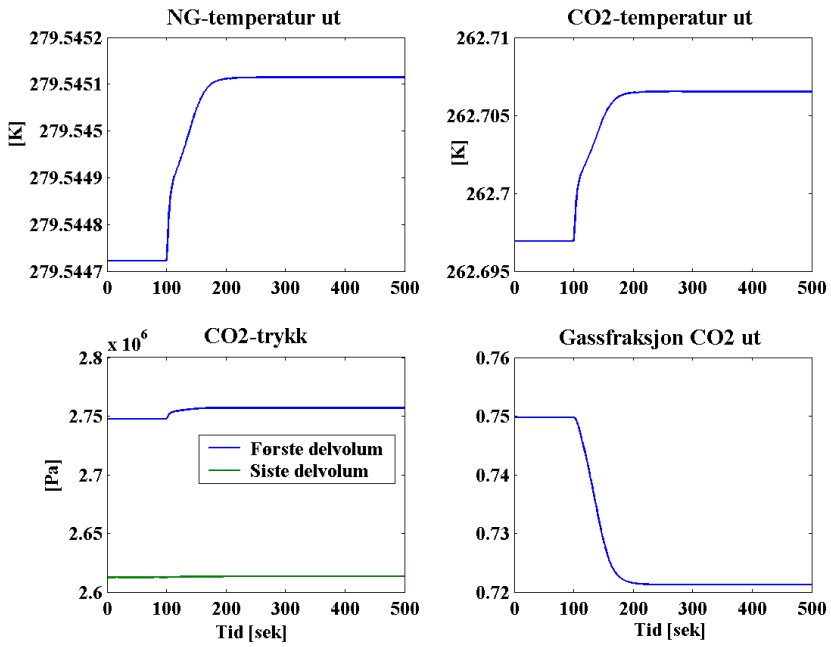
Tabell 5.3: Reelt tidsforbruk for simulert tid 500 sek ved steg i z

Modell	Løser	Tid [sek]
A	ode15s	4,1e3
B	ode15s	7,9e3
B	ode23tb	1,4e4

Siden ode15s viser seg å være den raskeste løseren, er denne benyttet i simuleringene.

5.2.2 Responser

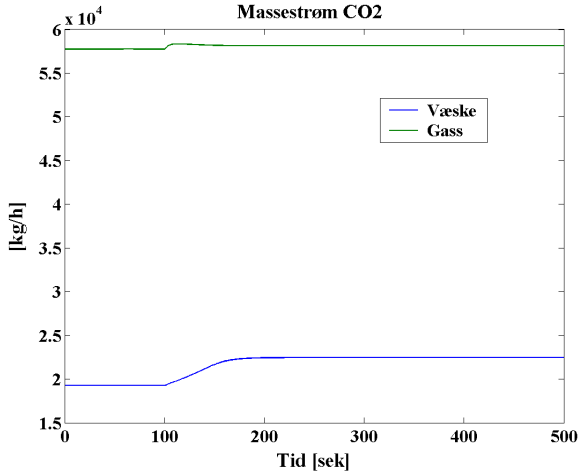
Naturgasstemperaturen påvirkes i svært liten grad som følge av åpningen av ventilen. Dette kan sees fra Figur 5.2, der temperaturen for naturgass og CO_2 ut av varmeveksleren samt CO_2-trykk i første og siste kontrollvolum er vist som funksjon av tid. Det siste plottet i figuren viser at gassfraksjonen ut på tofasesiden påvirkes i større grad enn trykk og temperaturer. Andelen gass synker gjennom hele varmeveksleren når ventilen åpnes.



Figur 5.2: Responsen ved åpning av CO_2-ventilen med 5% etter 100 sekunder. Total simuleringstid er 500 sekunder.

CO_2-trykket i første delvolum øker relativt mye ved åpning av ventilen, mens det i det siste holder seg temmelig konstant. Dette viser at trykkfallet over varmeveksleren øker.

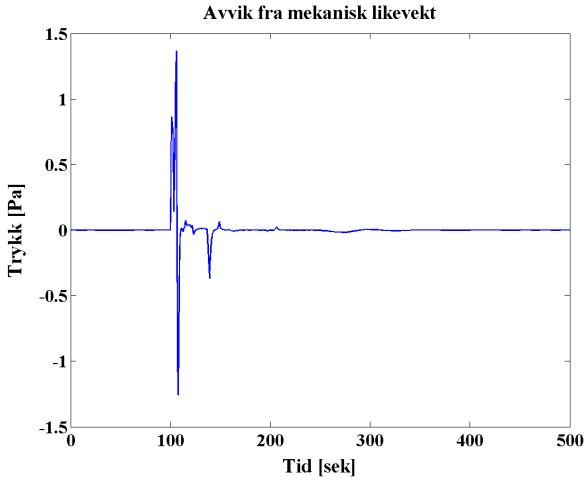
I Figur 5.3 er responsen til massestrømmen for gass og væske i siste delvolum for kjølemediet vist. Fluidet består i virkeligheten av gass og væske i blanding. Derfor kan ikke massestrømmen fysisk deles inn i to separate strømmer. Det betyr at størrelsene i figuren ikke er reelle. De er imidlertid tatt med for å illustrere hvordan væskeandelen øker som følge av trykkøkning bak ventilen.



Figur 5.3: Responsen på CO_2-strømning ved åpning av CO_2-ventilen med 5% etter 100 sekunder.

5.3 Driv i likevektskorreksjon

Antakelsen om at simultan gjennomføring av Newton- og integrasjonssteg ikke medfører avvik fra likevekt kan testes. Et plott med forskjellen i trykk for de to fasene mot tid, indikerer et eventuelt driv fra mekanisk likevekt. Dette er vist i Figur 5.4.



Figur 5.4: Forskjell i væske- og gasstrykk plottet mot tid ved simultant Newton- og integrasjonssteg for beregning av $(\dot{T}, \dot{V}, \dot{N})^s$. Plott fra første kontrollvolum i forkjøleren.

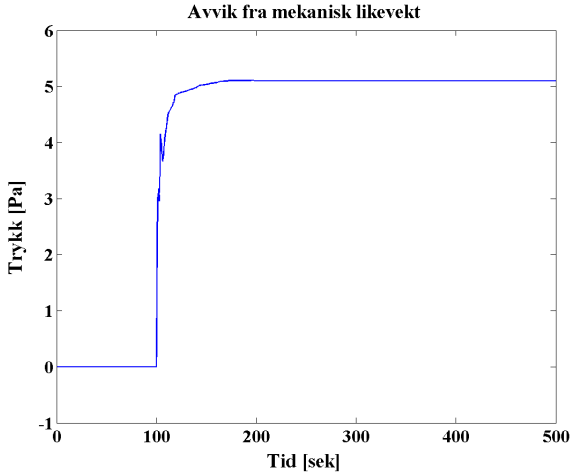
Forskjellen i trykkene til de to fasene gjør seg gjeldene etter 100 sekunder idet steget i ventilåpningen innføres. Avviket svinger imidlertid rundt null, slik at det ikke oppstår driv. I tillegg er det svært lite ($6 \cdot 10^{-4}$ promille). Etter at en ny stasjonær tilstand oppstår (litt etter 200 sekunder), stabiliserer avviket seg med små flukture ringer rundt null.

Dersom trykk og kjemisk potensial antas like, oppstår det et driv i $p^l - p^v$ og $\mu^l - \mu^v$. Dette er tilfelle når likning (4.14) skrives

$$\mathbf{M}\dot{\mathbf{y}} = \begin{bmatrix} \dot{\mathbf{x}} \\ \mathbf{0} \end{bmatrix} \tag{5.1}$$

Det vil si at $\Delta\mathbf{b}$ settes lik en null-vektor.

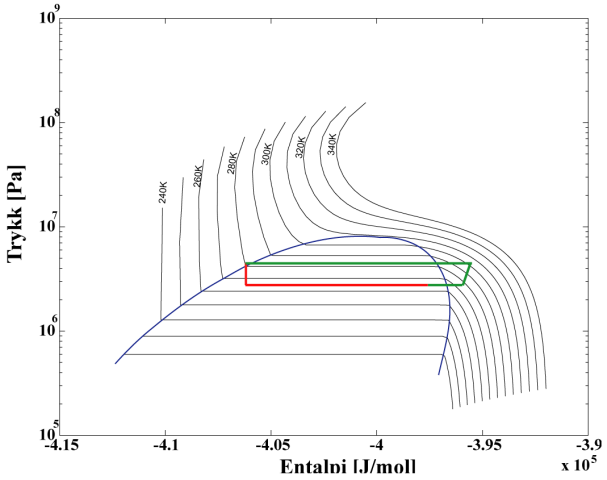
Figur 5.5 illustrerer hvordan avviket mellom trykkene utvikler seg med simuleringstiden. Etter at pertubasjonen oppstår ved 100 sekunder, øker forskjellen til omtrent 5 Pa og holder seg på denne verdien i den nye stasjonære tilstanden. Avviket utgjør kun $2 \cdot 10^{-3}$ promille av trykket i varmeveksleren, det er med andre ord svært lite. Differansen mellom kjemisk potensial for de to fasene er av enda mindre størrelsesorden enn avviket i trykk. Dette kan sees i Vedlegg G.



Figur 5.5: Forskjell i væske- og gasstrykk plottet mot tid ved antatt likevekt. Plottet er fra første kontrollvolum i forkjøleren.

5.4 pH-diagram

Et fasediagram for CO_2 er generert for å illustrere hvordan trykkavspenning og fordamping av gassen foregår. Isotermene er kalkulert med skriptet pH_co2_sw.m, som ligger i Vedlegg H.3. Den blå linjen skisserer tofaseområdet. Den simulerte prosessen er vist med rødt, mens de grønne linjene viser hvordan kompresjon og kondensasjon i en virkelig prosess foregår.



Figur 5.6: pH-diagram for CO_2 med isotermer generert ved hjelp av Span-Wagners tilstandslikning

6 Diskusjon

Modellens virkemåte og oppbygning diskuteres i det følgende. Valg av løsningsstrategi og grad av kompleksitet i Simulink er viktige aspekter med tanke på regnetid og eventuell videreutvikling.

6.1 Stasjonær tilstand

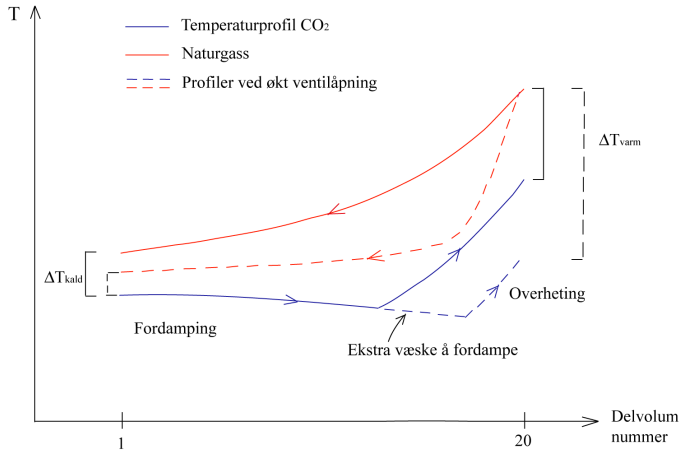
Den stasjonære tilstanden i den modellerte varmeveksleren stemmer overens med resultatene fra Hysysmodellen. Masse- og energibalanser for kjølekretsen er dermed konsistente. Dampfraksjonen etter ventilen er høyere i Hysys enn i Matlab-modellene (0,17 mot 0,07), men denne forskjellen har liten betydning. Den skyldes sannsynligvis at det benyttes ulike tilstandslikninger, eller den kan komme av ulike varmekapasiteter benyttet for CO_2. (Hysys gir som tidligere nevnt ingen mulighet til kontroll av varmekapasiteter)[1]

6.2 Endring i ventilåpning

Ved åpning av ventilen for CO_2 før varmeveksleren, forventes synkende temperatur på naturgass-siden som følge av økt kjølestrøm i den opprinnelige prosessen. Naturgasstemperaturen forblir imidlertid tilnærmet konstant i modellen av fordamperen, og som vist i plottet i avsnitt 5, er den lille endringen som oppstår motsatt av det intuisjonen tilsier. Denne effekten oppnås fordi trykket bak ventilen øker når ventilen åpnes. Dermed stiger også fordampningstemperaturen til CO_2, slik at ΔT blir mindre og naturgassen kjøles i mindre grad enn før. Det fordampes like mye CO_2 som før, mens mer væske strømmer igjennom.

Plottet med massestrømmene på kjølesiden (Figur 5.3) illustrerer hvordan væskestrømmen øker i større grad enn gass-strømmen. Dersom overheting av CO_2 hadde vært med, ville CO_2-gassen her bidratt til å senke temperaturen til naturgassen. Overhetingen foregår langs den horisontale grønne linjen utenfor tofaseområdet i trykk-entalpidiagrammet i Figur5.4. Økt kjøling ved åpning på ventil i en prosess med overheting kan forklares som i Figur 6.1. Økt væskemengde på kjølesiden bidrar til at fordampningsområdet til CO_2 i varmeveksleren forlenges på bekostning av overhetingsdelen. Dette gjør at ΔT ved innløpet til naturgassen blir større, slik at naturgassen her kjøles i større grad. Dermed vil også utløpstemperaturen synke. De stiplede temperaturprofilene i figuren illustrerer tilfellet med økt væskeandel. I en ren

fordamper uten overheting vil CO_2-temperaturen gjennom fordamperen i liten grad styres av gjennomstrømningen.



Figur 6.1: Resultat av økt væskestrøm i en prosess med overheting.

6.3 Løsningsstrategi

Likningssystemet er som antatt stivt, siden ODE-løserne for ikke-stive systemer ikke fungerer tilfredssillende.

De algebraiske likningenes påvirkning under beregningen av $\dot{T}, \dot{V}, \dot{N}$ er skjult for ODE-løseren i modellen. Det vil si at løseren oppfatter hele likningssystemet som et ODE-sett, uten å se de algebraiske likningene (likevektskorreksjonene). Likevektslikningene gir små forstyrrelser i systemet, noe som kan føre til at ODE-løseren får problemer med å velge fornuftige steglengder. Tidsskrittene i iterasjonene blir kortere og total regnetid øker som følge av forstyrrelser. Problemet oppstår altså fordi likevektsbegrensingene benyttes i overgangen fra de opprinnelige balansene, uttrykt ved $\dot{U}, \dot{V}, \dot{N}$, til de deriverte av tilstandene (T, V, N), som er de variablene som sendes til ODE-løseren.

Dersom likningssystemet hadde blitt løst uten denne overgangen, kunne de algebraiske likningene blitt sendt sammen med balanselikningene til ODE-løseren ved å sette høyre side lik null. Løsningsprosedyren vil dermed foregå ved hjelp av en implisitt DAE-løser. Siden systemet viser seg å være langt mer regnekrevende enn antatt, vil trolig denne løsningsformen fungere bedre for iterasjonsprosessen. En beskrivelse av hvordan dette kan gjøres er gitt i avsnitt 6.7.2.

6.4 Avvik fra likevekt

Figur 5.4 i avsnitt 5 viser at avviket fra mekanisk likevekt ved pådrag i ventilen ikke fører til drift fra likevekt. Det betyr at feilen som følge av at Newtonsteget utføres samtidig med utregningen av $(\dot{T}, \dot{V}, \dot{N})$ ikke gjør seg gjeldende. Ode15s er av varierende orden. Newtonsteget gir en feil av andre orden, og dette gir altså høy nok nøyaktighet i kombinasjon med ODE-løseren.

Dersom det antas likevekt, vil som vist avviket mellom p^l og p^v øke når en pertubasjon inntreffer. Forskjellen er ikke mer enn $2 \cdot 10^{-3}$ promille ved 5% åpning av ventilen, noe som gir en forsvinnende liten effekt på likevektsproblemet. Dersom det introduseres hyppige forstyrrelser, vil imidlertid systemet kunne drive av fra likevekt i stadig større grad. Dette viser at likevektskorreksjonen for hvert tidssteg bør være med i likningene, tross i at antatt likevekt fører til et ubetydelig avvik i èn enkelt forstyrrelse.

6.5 De to modellene

Modell A er relativt ukomplisert i Simulink, mens Matlab-skriptene er mer komplekse. Generering av rutinene for varmevekslingen krever kontroll på større sett av likninger og variable enn for Modell B, noe som gjør Modell B mer brukervennlig med tanke på eventuell endring av modell-likninger ved en senere anledning.

I Modell B er signalstrømmene mellom delvolumene flyttet fra Matlab-rutinene til Simulink ved hjelp av algebraiske løkker. Siden varmeveksleren er inndelt i mange kontrollvolum som hver er knyttet til en s-funksjon, er modellen mer kompleks i Simulink enn Modell A. Jo flere blokker, desto flere kilder til feil vil modellen ha utenom koden. For eksempel krever hver tilstandsrom-blokk riktig initialverdi, og hver s-blokk krever en ekstra parameter som forteller s-funksjonen hvilket delvolum som behandles. Disse ekstra operasjonene i Simulink unngås i Modell A, som behandler alle delvolum i ett.

Modell A bruker kortere tid på å løse problemet enn Modell B. Dette tyder på at det ikke er for-løkken i Modell A som tar tid. Kall av rutinene for beregning av Helmholtz' deriverte krever imidlertid relativt mye regnekraft. Dette kan sees ved å benytte funksjonen *profiler* i Matlab, som gir en indikasjon på hvor lang tid de ulike rutinene i Matlab krever. Det er ut fra dette vanskelig å avgjøre hva som gjør Modell B tregere enn A. Derfor er det nærliggende å tro at det er de algebraiske løkkene i Simulink som krever ekstra regnetid.

6.6 Simulink

Generelt vil det være enklere å sette seg inn i grafikkbaserte blokker enn tekstbasert kode. Dette gjør Simulink-modeller generelt mer anvendelige enn rene Matlab-modeller.

Sammenliknet med for eksempel Hysys gir imidlertid ikke flytskjemaet i Simulink spesielt godt innblikk i hvordan selve kjøleprosessen foregår, siden hver blokk bare kan ha èn åpning for inngangene og èn for utgangene.

Det er mulig å endre på det visuelle ved hjelp av maskering av de ulike blokkene med subsystemer, men dette er ekstra operasjoner som gjør programmet mindre enkelt å bruke. Signalstrømmene gir imidlertid en fin oversikt over hvordan de ulike delene i prosessen er knyttet sammen for den matematiske løsningen av problemet.

6.7 Forslag til videre arbeid

Modellen kan utvides til å beskrive hele forkjølingen. Den kan også endres slik at likningssettet løses med en implisitt DAE-løser istedenfor en eksplisitt ODE-løser.

6.7.1 Utvidelse av prosessen

Den modellerte forkjøleren er et godt grunnlag for utvidelse til hele kjølekretsen. Varmeveksleren i prosessen er den enheten der termodynamikken er mest komplisert. Kompressoren og ekstern kondensasjon/underkøling vil kunne modelleres relativt enkelt på lik linje med ventilene i denne modellen. I tillegg vil implementering av regulatorer være nødvendig.

Overhetingen av kjølemediet krever en noe mer komplisert prosedyre idet CO_2 går fra to- til ènfase; Når det oppstår negative moltall (dette skjer før volumet blir negativt), må en stabilitetssjekk kunne fortelle modellen at den har å gjøre med kun èn fase. Et generelt problem i termodynamikken er uvissheten om hvor mange faser som er tilstede ved likevekt. I tillegg medfører if-løkker, som er nødvendig ved en slik stabilitetssjekk, ofte problemer ved simulering.

Stabilitet i en fase kan undersøkes ved hjelp av tangentplantesten, der avstanden mellom tangentplanet og Helmholtz-flaten til den fasen som ikke eksisterer minimeres:

$$\begin{aligned} &min\{d = \mathbf{z}^T \cdot (\mu^o - \mu)\} \\ &\text{begrenset av} \quad \{\mathbf{z}^T \cdot \mathbf{e} = 1 \wedge \mathbf{z} \geq \mathbf{0}\} \end{aligned} \tag{6.1}$$

e er en enervektor, **z** er molbrøk til den ikke-eksisterende fasen, μ er kjemisk potensial til den ikke-eksisterende fasen og μ^o er kjemisk potensial til den eksisterende fasen. Når d er negativ, vil den manglende fasen være ustabil, mens den er stabil (og dermed tilstede) når d er postitiv. d lik null er ekvivalent med likevekt. En nærmere beskrivelse av minimanliseringen er gitt av Haug-Warberg[13][23].

6.7.2 Implisitt DAE-løser

Ved å skrive modell-likningene som et DAE-system, er det ikke nødvendig å omforme energibalansen fra U til T. Både indre energi og temperatur kobles isteden til tilstandsvektoren ved at en ekstra algebraisk likning innføres:

$$0 = U - U^o(T, V, N) \tag{6.2a}$$

I tillegg kommer likevektsbegrensningene samt masse- og volumbalanse med som algebraiske likninger.

Det totale likningssettet skrives på formen

$$M\frac{dy}{dt} = f(y, u) \tag{6.3}$$

der M (mass matrix) er en diagonalmatrise med 1 på diagonalen for differensiallikninger og 0 for algebraiske likninger.[14]

Dette systemet krever en løser som håndterer at enkelte av de tilstandsderiverte ganges med 0. Ode15s er et eksempel på en slik løser.

Dersom likningene skal løses som et implisitt DAE-sett istedenfor et eksplisitt ODE-sett i Simulink, kan blokker for algebraiske beskrankninger benyttes.[21] De algebraiske likningene erstattes på denne måten med algebraiske løkker. Siden det ser ut til at disse løkkene krever ekstra regnetid, vil sannsynligvis

en ren tekstbasert modell i Matlab være den beste løsningen for løsning av den type likningssett som behandles her. Det er imidlertid ikke undersøkt hvordan en Simunlink-modell med algebraiske løkker for både algebraiske likninger og sammenkobling av delvolum vil oppføre seg. Tatt i betraktning fordelen Simulink gir med tanke på brukervennlighet og endringsmuligheter, kan en slik løsning være et godt alternativ.

7 Konklusjon

Kjøling av naturgass mot CO_2 i tofaseområdet er modellert dynamisk i Matlab ved hjelp av Simulink. Det er sett på en typisk prosess med muligheter for å endre strømsammensetninger og betingelser i henhold til spesifikke prosesser. Hovedfokus har vært på termodynamikken i varmeveksleren, der de deriverte til Helmholtz' energi ligger til grunn for tilstandsberegningene.

Siden varmeveksleren er en ren fordamper, vil ikke manipulering av gjennomstrømning ha innvirkning på CO_2-temperaturen utover den lille endringen som kommer av at trykket forandres. I en virkelig prosess gjør overheteren at overført varme øker med gjennomstrømningen.

Når det antas oppfylte likevektskrav, oppstår det et driv i mekanisk og kjemisk likevekt dersom det introduseres pertubasjoner til systemet. Avviket er ubetydelig, men vil kunne øke for hver forstyrrelse som krever stabilisering til en ny likevekt.

To versjoner av modellen med ulik kompleksitet i Simulink og Matlab-rutiner er generert. I Modell A løses tilstandene i alle kontrollvolum simultant. Denne modellen har kortere regnetid enn Modell B, som opererer med signalstrømmer mellom delvolumer i Simulink. Årsaken er trolig at algebraiske løkker i Simulink er regnekrevende. Som resultat av større andel grafikk enn tekst, vil det i Modell B være lettere å få oversikt over løsningsgangen i prosessen enn i Modell A. Dermed er det også enklere å bruke og endre denne modellen.

Det modellerte systemet resulterer i et relativt komplisert ODE-sett der substituerte likevektskorreksjoner virker som forstyrrelser og fører til lang regnetid. En implisitt DAE-løser vil sannsynligvis være et bedre alternativ enn den eksplisitte ODE-løseren som er brukt i oppgaven.

Ved en eventuell videreutvikling av modellen, bør løsning av DAE-likninger i Simulink undersøkes nærmere. Dersom det legges stor vekt på regnetiden, er en ren Matlab-modell trolig den beste løsningen.

Symbolliste

Matriser er markert med uhevet skrift og stor bokstav. Eksempel: **M**.

Vektorer er markert med uthevet skrift. Unntak: μ er en vektor.

Vektorer markert med superskript T er radvektorer, andre kolonnevektorer.

Forkortelser

AE	Algebraiske likninger
DAE	Differensial-algebraiske likninger
FLNG	Flytende LNG-anlegg
HC	Hydrokarboner
LCR	Kondensasjonsmedium
LNG	Flytende naturgass
NG	Naturgass
ODE	Ordinære differensiallikninger
PC	Forkjøler
PCR	Forkjølingsmedium
PR	Peng-Robinson
SCR	Underkjølingsmedium
SW	Span-Wagner

Greske bokstaver

δ	Redusert tetthet	-
γ	Parameter i SWs tilstandslikning	
μ	Kjemisk potensial	J/mol
Ω	Delingsfunksjon	J
ω	Asentrisk faktor	

ρ	Tetthet	kg/m^3
τ	Invers redusert temperatur	-

Latinske bokstaver

$\mathbf{0}$	Nullvektor eller -matrise	
$\mathbf{f}$	Ordinære differensiallikninger (ODE)	
$\mathbf{g}$	Algebraiske likninger (AE)	
$\mathbf{u}$	Eksplisitt gitte funksjoner	
$\mathbf{y}$	Ukjente variable	
$\mathbf{z}$	Ukjente variable	
A	Helmholtz' energi	J
a	Parameter i PRs tilstandslikning	
b	Parameter i PRs tilstandslikning	
C_v	Ventilkonstant	m^2
$f\omega$	Parameter i PRs tilstandslikning	
G	Gibbs energi	J
H	Entalpi	J
h	Steglengde	s
M	Mass matrix	
n	Moltall	mol
n_i	Substansspesifikk koeffisient i SWs tilstandslikning	
p	Trykk	Pa
Q	Varme	J
R	Gasskonstant	J/Kmol
S	Entropi	J/K
T	Temperatur	K

t	Tid	s
U	Indre energi	J
u	Parameter i PRs tilstandslikning	
UA	Varmeoverføring	J/sK
V	Volum	m^3
W	Arbeid	J
w	Parameter i PRs tilstandslikning	
X	Energiform	J
Y	Energiform	J
z	Ventilåpning	

Subskript

c	Kritisk punkt	-
ext	Eksternt/Ytre	
fp	Frysepunkt	
k	Iterasjonsindeks	
r	Redusert	
i	Element i vektor. I matrise: radnummer	
j	Element i vektor. I matrise: kolonnenummer	
sur	omgivelser	

Superskript

$\cdot$	Tidsderivert (feks. $\dot{x}$)
d_i	Funksjonell form i SWs tilstandslikning
I_{Exp}	Funksjonell form i SWs tilstandslikning
I_{Pol}	Funksjonell form i SWs tilstandslikning
s	Gjelder for begge faser

sat	Mettet
t_i	Funksjonell form i SWs tilstandslikning
T	Transponert av vektor

Referanser

[1] I. K. Wold. *Karbondioksid som forkjølingsmedium ved produksjon av LNG på et flytende anlegg. Fordypningsemne TKP 4720 Prosess-Systemteknikk.* 2003.

[2] Wolfgang Förg. Natural gas trade routes and liquefaction processes. *Linde Technology*, (1):4–11, 2003.

[3] http://amchouston.home.att.net/lngc.htm.

[4] The Linde Statoil LNG Technology Alliance.

[5] Jostein Pettersen. *Termodynamisk grunnlag, Termisk Kraft/Varmeproduksjon.* Institutt for Energi og Prosessteknikk, NTNU, 2003.

[6] www.linde.com/en/p0002/p0008a/download/61 a.pdf.

[7] J. M. Coulson og J. F. Richardson. *Chemical Engineering.* Butterworth-Heinemann, 3 edition, 2000.

[8] Statoil Jostein Pettersen.

[9] Jostein Pettersen. *Carbon dioxide as a primary refrigerant.* Bidrag til Institute of Refrigeration. Centenary Conference, 1999, London.

[10] Gustav Lorentzen. Revival of carbon dioxide as a refrigerant. *Int. J. Refrig.*, 17(5):292–301, 1994.

[11] B. Armstrong S. Angus, K. M. De Reuck. *Carbon Dioxide. International Thermodynamic Tables of the Fluid State.* IUPAC Progect Centre, Imperial College, London, 3 edition.

[12] J. Løvland. *Applied Chemical Thermodynamics, part II.* Department of Chemical Engineering NTNU, 2002.

[13] Tore Haug Warberg. *Den termodynamiske arbeidsboken.* 2003.

[14] Sigurd Skogestad. *Prosessteknikk. Masse- og energibalanser.* Tapir, 2 edition, 2003.

[15] Morten Helbek. *Fysikalsk Kjemi.* 1999.

[16] http://www.ccl.net/cca/documents/dyoung/topics-orig/eq state.html.

[17] Poling Reid, Prausnitz. *The properties of gases and Liquids.* McGraw-Hill Book Company, 1 edition.

[18] R. Span. *Multiparameter Equation of State. An Accurate Source of Thermodynamic Property Data.* Springer, 2002.

[19] I. Cameron K. Hangos. *Process Modelling and Model Analysis.* Academic Press, 2001.

[20] D. Kincaid W. Cheney. *Numerikal mathematics and computing.* Brooks/Cole Publishing Company.

[21] www.mathworks.com/products/matlab.

[22] Hysys Process 3.1. Hyprotech.

[23] NTNU Tore Haug Warberg.

[24] www.tds tds.com. *Dippr (Design Institute for Physical Properties Researche)/Technical Database Services.*

[25] Mellichamp Seborg, Edgar. *Process Dynamics and Control.* 1989.

[26] Jørgen Bauck Jensen. *Hovedoppgave. Overordnet reguleringsstruktur for LNG-anlegg.* 2003.

A Legendre-transformasjoner

Integrasjon av den fundamentale likningen for et termodynamisk system er som vist i avsnitt 2.2 (likning 2.12):

$$U = TS - PV + \mu^T \mathbf{n}$$

Ved hjelp av Legendre-transformen er det mulig å bytte ut variabelsettet til en funksjon f_0 med et annet. En uønsket variabel erstattes med den tilhørende funksjonsderiverte. En ny funksjon f_i blir resultatet:

$$f_i\left(\xi_i, x_j, x_k ..., x_n\right) = f_0\left(x_i, x_j, x_k ..., x_n\right) - \xi_i \cdot x_i \tag{A.1a}$$

$$\xi_i = \left(\frac{\partial f_0}{\partial x_i}\right)_{x_j, x_k, ..., x_n} \tag{A.1b}$$

Gjentatte transformasjoner kan utføres på samme måte. For en termodynamisk energifunksjon med $m = dim(\mathbf{n}) + 2$ variable, finnes det 2^m-1 Legendre-transformer. Resultatet av transformasjoner på henholdsvis èn og to variable av gangen gir:

$$A(T, V, \mathbf{n}) \triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial S})_{V,\mathbf{n}} \cdot S \quad = U - TS \tag{A.2}$$

$$H(S, -p, \mathbf{n}) \triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial V})_{-p,\mathbf{n}} \cdot S \; = U + pV \tag{A.3}$$

$$X(S, V, \mu) \triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial \mathbf{n}})_{S,V} \cdot S = U - \mu^T \mathbf{n} \tag{A.4}$$

$$\begin{aligned} G(T, -p, \mathbf{n}) &\triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial V})_{S,\mathbf{n}} \cdot V - (\frac{\partial U}{\partial S})_{V,\mathbf{n}} \cdot S \\ &= U + pV - TS \end{aligned} \tag{A.5}$$

$$\begin{aligned} Y(S, -p, \mu) &\triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial V})_{S,\mathbf{n}} \cdot V - (\frac{\partial U}{\partial \mathbf{n}})_{S,V} \cdot \mathbf{n} \\ &= U + pV - \mu^T \mathbf{n} \end{aligned} \tag{A.6}$$

$$\begin{aligned} \Omega(T, V, \mu) &\triangleq U(S, V, \mathbf{n}) - (\frac{\partial U}{\partial S})_{V,\mathbf{n}} \cdot S - (\frac{\partial U}{\partial \mathbf{n}})_{S,V} \cdot \mathbf{n} \\ &= U - TS - \mu^T \mathbf{n} \end{aligned} \tag{A.7}$$

Her er definisjonene fra avsnitt 2.2.1 begnyttet:

$$
\begin{aligned}
T &= \left(\frac{\partial U}{\partial S}\right)_{V,\mathbf{n}} \\
-p &= \left(\frac{\partial U}{\partial V}\right)_{S,\mathbf{n}} \\
\mu_i &= \left(\frac{\partial U}{\partial n_i}\right)_{S,V,n_{j\neq i}}
\end{aligned}
$$

Ved å transformere U med hensyn på alle tre variable, fås null-funksjonen:

$$
\begin{aligned}
O(T,-p,\mu) &\triangleq U(S,V,\mathbf{n}) - \left(\frac{\partial U}{\partial V}\right)_{S,\mathbf{n}} V - \left(\frac{\partial U}{\partial S}\right)_{V,\mathbf{n}} S - \left(\frac{\partial U}{\partial \mathbf{n}}\right)_{S,V} \mathbf{n} \\
&= U + pV - TS - \mu^T \mathbf{n}
\end{aligned} \tag{A.8}
$$

Ved å derivere denne, fås Gibbs-Duhems likning. Denne likningen er viktig for konsistenssjekk av måledata i termodynamikken, og må være oppfylt for alle faser til enhver tid.

En total oversikt med de ulike transformasjonene og tilhørende navn på energifunksjonene er gitt i Tabell A.

Tabell A.1: Alle Legendretransformasjoner av indre energi

	:					
Indre energi	: $U \triangleq U$			$= TS$	$-pV$	$+\mu^T\mathbf{n}$
	:					
Transformasjon med hensyn på en variabel av gangen						
	:					
Helmholtz' energi	: $A \triangleq U$	$-TS$		$=$	$-pV$	$+\mu^T\mathbf{n}$
	:					
Entalpi	: $H \triangleq U$		$+pV$	$= TS$		$+\mu^T\mathbf{n}$
	:					
Uten navn	: $X \triangleq U$		$-\mu^T\mathbf{n}$	$= TS$	$-pV$	
	:					
Transformasjon med hensyn på to variable av gangen						
	:					
Gibbs energi	: $G \triangleq U$	$-TS$	$+pV$	$=$		$+\mu^T\mathbf{n}$
	:					
Kanonisk potensial	: $\Omega \triangleq U$	$-TS$	$-\mu^T\mathbf{n}$	$=$	$-pV$	
	:					
Uten navn	: $Y \triangleq U$		$+pV$	$-\mu^T\mathbf{n}$	$= TS$	
	:					
Transformasjon med hensyn på tre variable av gangen						
	:					
Nullfunksjon	: $O \triangleq U$	$-TS$	$+pV$	$-\mu^T\mathbf{n}$	$= 0$	

B Faselikevekt

For et lukket system med n forbindelser ved gitt T og p og med to faser α og β i likevekt, vil total endring i Gibbs energi G og masse N være lik null. Tilsvarende er gjeldende for Helmholtz' energi dersom temperatur, totalt volum og totalsammensetning er gitt[13].

$$\begin{aligned}(\mathrm{d}A)_{T,V,\mathbf{n}} &= \mathrm{d}A^{\alpha} + \mathrm{d}A^{\beta} \\ &= -p^{\alpha}\mathrm{d}V^{\alpha} + (\mu^{\alpha})^{\mathrm{T}}\,\mathrm{d}\mathbf{n}^{\alpha} - p^{\beta}\mathrm{d}V^{\beta} + \left(\mu^{\beta}\right)^{\mathrm{T}}\mathrm{d}\mathbf{n}^{\beta} = 0 \end{aligned} \tag{B.1}$$

$$dV^{\alpha} + dV^{\beta} = 0 \tag{B.2}$$

$$dN_i^{\alpha} + dN_i^{\beta} = 0; \qquad \forall \quad i \in [1, n] \tag{B.3}$$

Ved å kombinere de tre likningene slik at $\mathrm{d}V^{\beta}$ og N_i^{β} elimineres i B.1 fås:

$$(dA)_{T,V,\mathbf{n}} = -(p^{\alpha} - p^{\beta})\mathrm{d}V^{\alpha} + (\mu^{\alpha} - \mu^{\beta})^{\mathrm{T}}\mathrm{d}\mathbf{n} = 0 \tag{B.4}$$

$\mathrm{d}V^{\alpha}$ og $\mathrm{d}N_i^{\alpha}$ er uavhengige størrelser i nærheten av likevektspunktet. Med dette må likevektskravet over også kunne formuleres som:

$$\begin{aligned} -p^{\alpha} &= -p^{\beta} \\ \mu_i^{\alpha} &= \mu_i^{\beta}; \qquad \forall \quad i \in [1, n] \end{aligned} \tag{B.5}$$

Det vil si at trykk og kjemisk potensiale for komponent i i tillegg til temperatur er like i de to fasene.

C Span-Wagners tilstandslikning

I takt med de siste 20 årenes utvikling av optimaliseringsalgoritmer, har anvendelsen tilstandslikninger med optimalisert funksjonell form økt.

Utviklingen av den funksjonelle formen er tidkrevende på grunn av at likningene er svært numerisk fleksible og dermed trenger store og konsistente datasett. Dette medfører at denne typen likninger generelt er tilgjengelige kun for et begrenset antall substanser.

Siden Span og Wagners tilstandslikning har grunnlag i en optimeringsalgoritme som tar for seg datasett for ulike substanser simultant, er den anvendbar på en rekke substanser også med mangelfulle datasett.

Span-Wagners likning uttrykt ved redusert Helmholtz energi er gitt i likning (C.1).

$$\frac{a(T,\rho)}{RT} = \frac{a^{\circ}(T,\rho) + a^{r}(T,\rho)}{RT} = \alpha^{\circ}(\tau\delta) + \alpha^{r}(\tau\delta) \tag{C.1}$$

Det første leddet (α°) beskriver oppførselen til en hypotetisk ideell gass ved gitt temperatur og tetthet. $\alpha^{\circ}(\tau\delta)$ kan finnes ved å integrere varmekapasiteten til ideell gass, $c^{\circ}(T)$. Det andre (α^{r}) er residualleddet, det vil si Helmholtz energi til det reelle fluidet.

Ved bruk av Spans simultane optimeringsprosedyre blir uttrykket for α^{r} som vist i liking (C.2).

$$\begin{aligned}\frac{a^{r}(T,\rho)}{RT} = \alpha^{r}(\tau,\delta) &= \sum_{i=1}^{I_{Pol}+I_{Exp}} A_i(\tau,\delta) \\ &= \sum_{i=1}^{I_{Pol}} n_i \tau^{t_i} \delta^{d_i} + \sum_{i=I_{Pol}+1}^{I_{Pol}+I_{Exp}} n_i \tau^{t_i} \delta^{d_i} exp(-\delta^{p_i})\end{aligned} \tag{C.2}$$

Den endelige formen etter optimalisering av de funksjonelle formene I_{Pol} og I_{Exp} er som vist i (C.3).

$$\begin{aligned}\alpha^r(\tau,\delta) = &\sum_{i=1}^{8}\sum_{j=-8}^{12} n_{i,j}\delta^i\tau^{j/8} + \sum_{i=1}^{5}\sum_{j=-8}^{24} n_{i,j}\delta^i\tau^{j/8}e^{-\delta} \\ &+ \sum_{i=1}^{5}\sum_{j=16}^{56} n_{i,j}\delta^i\tau^{j/8}e^{-\delta^2} + \sum_{i=2}^{4}\sum_{j=24}^{38} n_{i,j}\delta^i\tau^{j/2}e^{-\delta^3}\end{aligned} \tag{C.3}$$

Resultatet for polare fluider er som følger:

$$\begin{aligned}\alpha(\tau,\delta) &= \alpha^\circ(\tau,\delta) + \alpha^r(\tau,\delta) \\ &= \alpha^\circ(\tau,\delta) + n_1\delta^1\tau^{0.250} + n_2\delta^1\tau^{1.250} + n_3\delta^1\tau^{1.500} \\ &\quad + n_4\delta^3\tau^{0.250} + n_5\delta^7\tau^{0.875} + n_6\delta^1\tau^{2.375}e^{-\delta} \\ &\quad + n_7\delta^2\tau^{2.000}e^{-\delta} + n_8\delta^5\tau^{2.125}e^{-\delta} + n_9\delta^1\tau^{3.500}e^{-\delta^2} \\ &\quad + n_{10}\delta^1\tau^{6.50}e^{-\delta^2} + n_{11}\delta^4\tau^{4.75}e^{-\delta^2} \\ &\quad + n_{12}\delta^2\tau^{12.5}e^{-\delta^3}\end{aligned} \tag{C.4}$$

D De deriverte av Helholtz' energi

Rutinene co2_sw.m og ng_pr.m (se Vedlegg H.3.6 og H.3.7) tar inn tilstandene T, V, og $\mathbf{n}$ og returnerer de deriverte av Helmholtz' energi på følgende form:

$$A.g = \begin{bmatrix} \left(\frac{\partial A}{\partial T}\right)_{V,\mathbf{n}} \\ \left(\frac{\partial A}{\partial V}\right)_{T,\mathbf{n}} \\ \left(\frac{\partial A}{\partial \mathbf{n}}\right)_{T,V} \end{bmatrix} = \begin{bmatrix} -S \\ -p \\ \mu \end{bmatrix} \tag{D.1}$$

$$A.H = \begin{bmatrix} -\left(\frac{\partial S}{\partial T}\right)_{V,\mathbf{n}} & -\left(\frac{\partial S}{\partial V}\right)_{T,\mathbf{n}} & -\left(\frac{\partial S}{\partial \mathbf{n}}\right)_{T,V} \\ -\left(\frac{\partial p}{\partial T}\right)_{V,\mathbf{n}} & -\left(\frac{\partial p}{\partial V}\right)_{T,\mathbf{n}} & -\left(\frac{\partial p}{\partial \mathbf{n}}\right)_{T,V} \\ \left(\frac{\partial \mu}{\partial T}\right)_{V,\mathbf{n}} & \left(\frac{\partial \mu}{\partial V}\right)_{T,\mathbf{n}} & \left(\frac{\partial \mu}{\partial \mathbf{n}}\right)_{T,V} \end{bmatrix} \tag{D.2}$$

I balanse- og likevektslikningene trengs, i tillegg til sammenhengene over, entalpi og de deriverte av indre energi. Ved å benytte definisjonene på U og H fra Tabell 2.2, fås følgende relasjoner:

$$\begin{aligned} \left(\frac{\partial U}{\partial T}\right)_{V,\mathbf{n}} &= \left(\frac{\partial A}{\partial T}\right)_{V,\mathbf{n}} + S + T \cdot \left(\frac{\partial S}{\partial T}\right)_{V,\mathbf{n}} \\ &= T \cdot A.H(1,1) \end{aligned} \tag{D.3a}$$

$$\begin{aligned} \left(\frac{\partial U}{\partial V}\right)_{T,\mathbf{n}} &= \left(\frac{\partial A}{\partial V}\right)_{T,\mathbf{n}} + T \cdot \left(\frac{\partial S}{\partial V}\right)_{T,\mathbf{n}} \\ &= A.g(2) - T \cdot A.H(1,2) \end{aligned} \tag{D.3b}$$

$$\begin{aligned} \left(\frac{\partial U}{\partial \mathbf{n}}\right)_{T,V} &= \left(\frac{\partial A}{\partial \mathbf{n}}\right)_{T,V} + T \cdot \left(\frac{\partial S}{\partial \mathbf{n}}\right)_{T,V} \\ &= A.g(3) - T \cdot A.H(1,3) \end{aligned} \tag{D.3c}$$

$$\begin{aligned} H &= S \cdot T + \mu \cdot \mathbf{n} \\ &= -A.g(1) \cdot T + A.g(3) \cdot \mathbf{n} \end{aligned} \tag{D.4}$$

E Antagelse for numerisk integrator

ODE-likningene i modellen kan uttrykkes som

$$\mathbf{M}\dot{\mathbf{y}} = \begin{bmatrix} \dot{\mathbf{x}} \\ \mathbf{0} \end{bmatrix} \tag{E.1}$$

Her er $\mathbf{M}$, $\dot{\mathbf{y}}$ og $\dot{\mathbf{x}}$ som beskrevet i avsnitt 4.4.1.

Likevektslikningene kan skrives

$$\mathbf{M}\delta\mathbf{y} = \begin{bmatrix} \mathbf{0} \\ \Delta\mathbf{b} \end{bmatrix} \tag{E.2}$$

$\Delta\mathbf{b}$ er altså likevektskorreksjonene:

$$\Delta\mathbf{b} = \begin{bmatrix} 0 \\ -(p^l - p^v) \\ -(\mu^l - \mu^v) \end{bmatrix}$$

Newtoniterasjoner på tilstanden som kommer fra ODE-løseren tillates ikke, derfor introduseres likevektslikningene i ODE-systemet.

$$\mathbf{M}(\underbrace{\dot{\mathbf{y}} + \delta\mathbf{y}}_{\dot{\mathbf{y}}'}) = \begin{bmatrix} \dot{\mathbf{x}} \\ \Delta\mathbf{b} \end{bmatrix} \tag{E.3}$$

Ved å anta at $\dot{\mathbf{y}}'$ beskriver endringen i tilstandene såvel som $\dot{\mathbf{y}}$, siden $\|\delta\mathbf{y}\| \ll \|\dot{\mathbf{y}}\|$, kan likningssystemet skrives som i likning (4.14):

$$\mathbf{M}\dot{\mathbf{y}} = \begin{bmatrix} \dot{\mathbf{x}} \\ \Delta\mathbf{b} \end{bmatrix}$$

F Data

F.1 Strømningsdata for opprinnelig prosess

Et flytskjema for en tretrinns forkjølingsprosess med karbondioksid er vist i Figur 3.1. Data for strømmene er gjengitt i Tabell F.1 og F.2. Verdiene er hentet fra simuleringer utført i Hysys i forprosjektet[1].

Tabell F.1: Strømdata, forkjølingskrets med CO_2

Strøm nummer	Flow [kg/h]	Temp [°C]	Trykk [bar]	Gass-fraksjon
1	2,76e6	8,00	48,3	0
2	8,01e5	8,00	48,3	0
3	8,01e5	-11,0	25,3	0,19
4	8,01e5	4,95	25,3	1
5	1,96e6	8,00	48,3	1
6	1,96e6	-10,0	48,3	0
7	1,33e6	-10,0	48,3	0
8	1,33e6	-31,0	13,4	0,15
9	1,33e6	-13,0	13,4	1
10	6,32e5	-10,0	48,3	0
11	6,32e5	-30,0	48,3	0
12	6,32e5	-51,0	6,32	0,12
13	6,32e5	-33,2	6,32	1
14	6,32e5	14,2	13,4	1
15	1,96e6	-4,28	13,4	1
16	1,96e6	40,4	25,3	1
17	2,76e6	29,9	25,3	1
18	2,76e6	81,3	48,3	1
19	2,76e6	13,0	48,3	0

Tabell F.2: Strømdata, varme strømmer

Type strøm	Strøm nummer	Flow [kg/h]	Temp [°C]	Trykk [bar]	Gass-fraksjon
LCR	20	6,20e5	8,00	16,4	0
	21	6,20e5	-10,0	16,4	0
	22	6,20e5	-30,0	16,4	0
	23	6,20e5	-50,0	16,4	0
SCR	24	5,50e5	8,00	53,0	1
	25	5,50e5	-10,0	53,0	1
	26	5,50e5	-30,0	53,0	0.82
	27	5,50e5	-50,0	53,0	0.40
NG	28	8,00e5	8,00	70,0	1
	29	8,00e5	-10,0	70,0	1
	30	8,00e5	-30,1	70,0	1
	31	8,00e5	-50,0	70,0	1

F.2 Fysikalske data

Molvekter brukt for CO_2 og naturgass samt varmekapasitet for veggen i varmeveksleren er vist i tabell F.3.

Tabell F.3: Molvekter og varmekapasitet for vegg

Molvekt (MW) CO_2	44e-3 kg/mol
Molvekt NG	16e-3 kg/mol
Varmekapasitet vegg (C_p)	24 J/molK

F.3 Anleggsdata

Dimensjonene på tankene og forkjøleren er gitt i Tabell F.4. Tallene er hentet fra hovedoppgaven til J. B. Jensen [26], der en forkjølingskrets med kapasitet på kjøling av 6,3e5 kg naturgass i timen er simulert. Volumet på CO_2-siden er skalert ned til halvparten av verdien i [26], siden kjøling av kondensasjon- og underkjølingsmediene ikke er simulert i denne oppgaven.

Trykk og temperaturer i reservoarene er vist i Tabell F.5.

Tabell F.4: Volumer

	CO_2	NG
Forkjøler	7 m^3	7 m^3
Tanker	10 m^3	10 m^3

Tabell F.5: T og p i tankene

	T[°C]	p [bar]
CO_2-tank	8	42
NG-tank	8	71

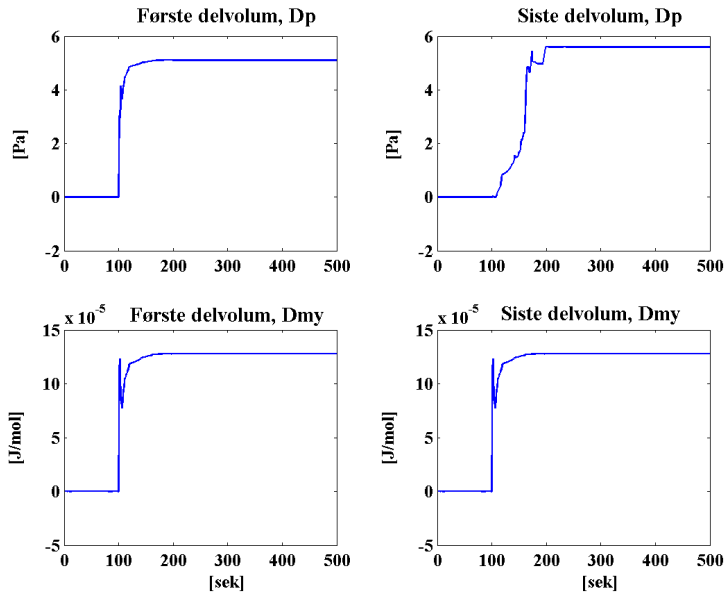
Tabell F.6 viser C_v- og UA-verdier i varmeveksleren og ventilene.

Tabell F.6: Cv og UA

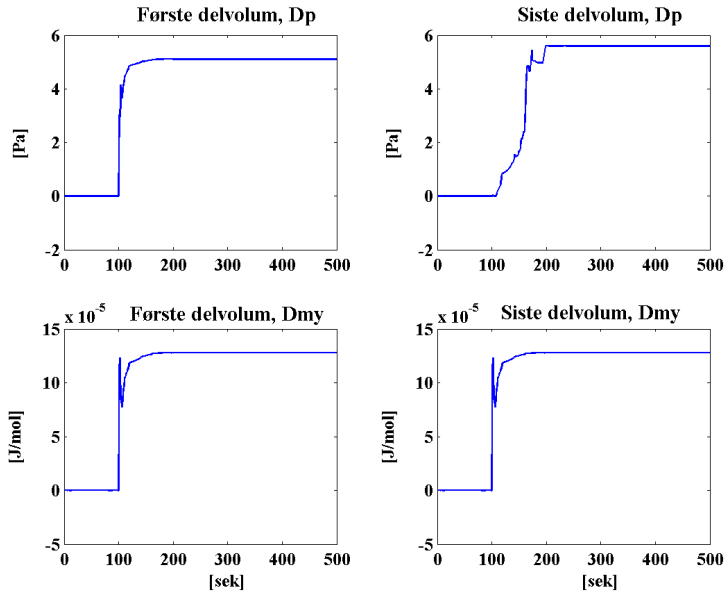
	CO_2	NG	
C_v^{ventil}	0,0012	0,08	m^2
C_v^{PC}	0,02	0,4	m^2
UA	8e4	1.8e4	J/sK

G Avvik fra likevekt

I figur G.1 er avvik i trykk og kjemisk potensial for de to fasene i varmeveksleren vist. Avviket oppstår som følge av endring i ventilåpningen til CO_2-strømmen inn på varmeveksleren når det er antatt likevekt i de termodynamiske beregningene. Når likevektskorreksjoner er tatt med i beregningene ved simultant Newton- og integrasjonssteg, stabiliserer avviket seg rundt null etter introduksjon av diskontinuiteten ved 100 sekunder. Dette er vist i Figur G.2



Figur G.1: Avvik fra mekanisk og kjemisk likevekt plottet mot tid ved antatt likevekt.



Figur G.2: Avvik fra mekanisk og kjemisk likevekt plottet mot tid ved ved simultant Newton- og integrasjonssteg for beregning av $(\dot{T}, \dot{V}, \dot{N})^s$.

H Matlab-rutiner

Følgende matlabrutiner er utviklet:

For hver av de to modellene:

- init.m Avhenging av startverdier.mat og globale_verdier_b.mat
- s_pc.m
- pc.m
- pc_co2.m
- pc_ng.m

Felles for Modell A og B:

- s_co2tank.m
- s_ngtank.m
- s_co2ventil.m
- co2ventil.m
- s_ngventil.m
- co2_sw.m (Utviklet av Tore Haug-Warberg)
- ng_pr.m (Utviklet av Tore Haug-Warberg)

Skript benyttet for første oppstart:

- init_mod.m
- starttilstand.m
- tank.m

Plotting av data:

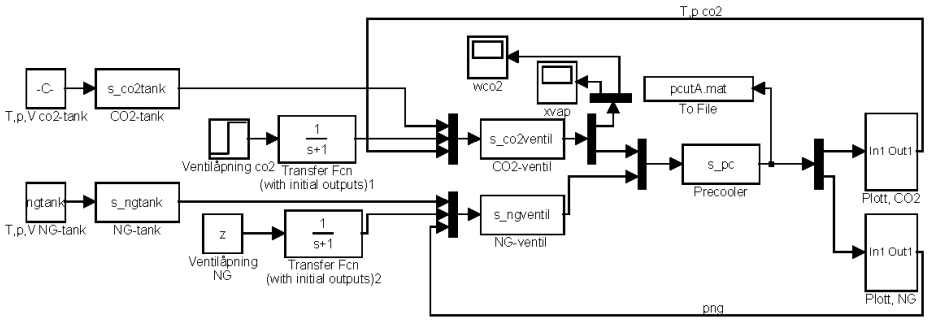
- plotter.m
- pH_co2_sw.m

Alle s-funksjonene har navn som begynner med s_. Disse er direkte koblet til Simulink. De andre rutinene er underordnet s-funksjonene.

Flytskjema fra Simulink samt skriptene for Modell A og B er listet i henholdsvis H.1 og H.2. Rutinene som er felles for modellene er vedlagt i avsnitt H.3.

H.1 Modell A

Denne modellen behandler alle delvolum i varmeveksleren i ett. Flytskjema er vist i Figur H.1.



Figur H.1: Modell A

H.1.1 init.m

```
% init.m

% Initialiserer modell A

load globale_verdier_b  % gir b
b.Cv_pc = [ones(b.n,1)*2e-2 ones(b.n,1)*4e-1];
b.Cv_ventil=1.2e-3;
b.Cv_ngventil=0.08;

b.UA_pc(1) = 8e4;
b.UA_pc(2) = 1.8e4;

tau=0.1;
G=tf(1,[tau 1]);        % Transferfunksjon
G2x2=ss(append(G,G));

for i=1:6
    D1(1+(i-1)*b.n:i*b.n) = b.D1(i)*ones(1,b.n);
    D2(1+(i-1)*b.n:i*b.n) = b.D2(i)*ones(1,b.n);
end
b.D1=D1'; b.D2=D2';      % Radvektorer (120x1)

global b

% Innganger
%-------------------------------------------------------------------------
co2tank = [273+8, b.p_co2tank, 10]';     %T,p,V
ngtank  = [273+8, b.p_ngtank, 10]';

% Regulatorparameter
%-------------------------------------------------------------------------
z = 0.5;                 % Ventilåpning

% Startverdier for PC
%-------------------------------------------------------------------------
load startverdier        % gir s_ut

m=18;
p0l = s_ut(2:m:end,end);
p0v = s_ut(3:m:end,end);
w0ng= s_ut(4:m:end,end);
H0ng= s_ut(5:m:end,end);
p0ng= s_ut(6:m:end,end);
w0l = s_ut(7:m:end,end);
w0v = s_ut(8:m:end,end);

T0l = s_ut(11:m:end,end);
V0l = s_ut(12:m:end,end);
N0l = s_ut(13:m:end,end);
T0v = s_ut(14:m:end,end);
V0v = b.V_pc(1)-V0l;
N0v = s_ut(15:m:end,end);
T0ng = s_ut(16:m:end,end);
V0ng = s_ut(17:m:end,end);
N0ng = s_ut(18:m:end,end);
T0vegg = s_ut(19:m:end,end);

y0 = [T0l; V0l; N0l; T0v; N0v; T0ng; V0ng; N0ng; T0vegg];
save initial y0
```

H.1.2 s_pc.m

```
function [sys,x0,str,ts] = s_pctest(t,x,u,flag)
% t,x,u og flag sendes automatisk til denne s-funksjonen av simulink. u er
% inngangene (pådrag).
% u = wl, wv, Htotco2, wng og Hng
global b
switch flag

  % INITALIZATION
  %--------------
  case 0,
    [sys,x0,str,ts]=Initialize;


  % DERIVATIVES
  %------------
  case 1,
    dy = pc(t,x,u);   % x er tilstandene TVN og p (y el dydt i pc.m)
    sys=dy;

  % OUTPUTS
  %--------
  case 3,
    Tl = x(1:b.n);                % Kolonnevektorer (nedover)
    Vl = x(b.n+1:2*b.n);
    Nl = x(2*b.n+1:3*b.n);
    Tv = x(3*b.n+1:4*b.n);
    Vv = b.V_pc(1)-Vl;
    Nv = x(4*b.n+1:5*b.n);
    Tng= x(5*b.n+1:6*b.n);
    Vng= x(6*b.n+1:7*b.n);
    Nng= x(7*b.n+1:8*b.n);
    Tvegg= x(8*b.n+1:9*b.n);
    for i=1:b.n
        Al = co2_sw([Tl(i),Vl(i),Nl(i)]');
        Av = co2_sw([Tv(i),Vv(i),Nv(i)]');
        Ang = ng_pr([Tng(i),Vng(i),Nng(i),Nng(i)*1e-6,Nng(i)*1e-6]');
        pl(i)  =-Al.g(2); % trykk i hvert kontr.volum
        pv(i)  =-Av.g(2);
        png(i) =-Ang.g(2);
    end
   xv = (Nv)./(Nl+Nv);

    sys = [Tl; Vl; Nl; pl';
           Tv; Vv; Nv; pv'; xv;
           Tng; Vng; Nng; png';
           Tvegg];

%     for i=1:b.n
%         l=isreal(pl(i));
%         if l==0,
%             disp('Komplekse elementer i pl'),disp(Tl(i)),disp(Vl(i)),disp(
    Nl(i))
%
%     end
%     l=isreal(png);if l==0, disp('Komplekse elementer i png'),end

%     l=isreal(Tv); if l==0, disp('Komplekse elementer i Tv'), end
```

```
  case {2,4,9}
    sys=[];

  % UNEXPECTED FLAGS
  %-----------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end

% end sfuntmpl

% Initialiseringsfunksjon:
function [sys,x0,str,ts]=Initialize
global b

sizes = simsizes;

sizes.NumContStates  = 9*b.n; % TVN(liq) TN(vap) TVN(ng) T(vegg)
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 14*b.n; % TVNp(liq) TVNp(vap) xl TVNp(ng) T(vegg)
sizes.NumInputs      = 5; % wl, wv, Htot og wng, Hng
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

load initial
% initialize the initial conditions
x0 = y0; % TVN(liq) TN(vap) TVN(ng) T(vegg)

% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
% end mdlInitializeSizes
```

H.1.3 pc.m

```
% pc.m

% Inn: TVN liq, TN vap, TVN ng, Tvegg. Ut: deriverte til ODE-løser via s_pc
% winn liq = u(1)    [kg/s]
% wut  vap = u(2)
% Hinn co2 = u(3)    [J/s] Totalt for l og v (beregnet i tank.m)
% w NG     = u(4)
% Hinn NG  = u(5)

function dy=pc(t,y,u,ni)
global b

y_co2 = y(1:5*b.n);          %TVNTN
y_ng  = y(5*b.n+1:8*b.n);    %TVN
T_vegg= y(8*b.n+1:end);      %T

Q_co2 = b.UA_pc(1)*(T_vegg-y_co2(1:b.n));   %[J/s]
Q_ng  = b.UA_pc(2)*(T_vegg-y_ng(1:b.n));
```

```
%Q_co2 = zeros(1,b.n);

dy_co2 = pc_co2(y_co2, [u(1:3);Q_co2]);
% dy_co2 = zeros(5*b.n,1);

dy_ng  = pc_ng(y_ng, [u(4:5);Q_ng]);
% dy_ng = zeros(3*b.n,1);

% Qco2 > 0. Ut fra vegg-->minus foran
% Qng  < 0. Inn i vegg-->minus foran:
dT_vegg = b.mCp_vegg*(-Q_ng-Q_co2);
% dT_vegg = zeros(b.n,1);


dy = [dy_co2; dy_ng; dT_vegg];
```

H.1.4 pc_co2.m

```
% pc_co2.m

% Kalles fra pc.m
% Returnerer TVNderivert, tar inn: tilstander, og wl,wv og Htot fra ventil
% samt Q.

function dy=pc_co2(y,u)
global b

wl_ventil = u(1);
wv_ventil = u(2);          %[kg/s]
H_inn = u(3);              %[J/s] Totalt for l og v
Q = u(4:end);              %[J/s] Beregnet i pc.m. For hvert kontrollvolum

Tl = y(1:b.n);             % Kolonnevektorer
Vl = y(b.n+1:2*b.n);
Nl = y(2*b.n+1:3*b.n);
Tv = y(3*b.n+1:4*b.n);
Vv = b.V_pc(1)-Vl;
Nv = y(4*b.n+1:5*b.n);

if Tl<0
    disp('pc_co2.m, Tl=')
    disp(Tl)
end

o = NaN*zeros(b.n,1);
dUdTl=o; dUdVl=o; dUdNl=o; dUdTv=o; dUdVv=o; dUdNv=o;
dpdTl=o; dpdVl=o; dpdNl=o; dpdTv=o; dpdVv=o; dpdNv=o;
dmydTl=o; dmydVl=o; dmydNl=o; dmydTv=o; dmydVv=o; dmydNv=o;
Hl=o; Hv=o; pl=o; pv=o; myl=o; myv=o;
for i=1:b.n
    Al = co2_sw([Tl(i),Vl(i),Nl(i)]'); %eller x?
    Av = co2_sw([Tv(i),Vv(i),Nv(i)]');

    %Elementer til M-matrisen. (Blir radvektorer)
    dUdTl(i) = -Tl(i)*Al.H(1,1); % (= Cvliq)
    dUdVl(i) = Al.g(2) - Tl(i)*Al.H(1,2);
    dUdNl(i) = Al.g(3) - Tl(i)*Al.H(1,3);
    dUdTv(i) = -Tv(i)*Av.H(1,1); % (= Cvvap)
    dUdVv(i) = Av.g(2) - Tv(i)*Av.H(1,2);
    dUdNv(i) = Av.g(3) - Tv(i)*Av.H(1,3);
```

```
    dpdTl(i) = -Al.H(2,1);
    dpdVl(i) = -Al.H(2,2);
    dpdNl(i) = -Al.H(2,3);
    dpdTv(i) = -Av.H(2,1);
    dpdVv(i) = -Av.H(2,2);
    dpdNv(i) = -Av.H(2,3);

    dmydTl(i)= Al.H(3,1);
    dmydVl(i)= Al.H(3,2);
    dmydNl(i)= Al.H(3,3);
    dmydTv(i)= Av.H(3,1);
    dmydVv(i)= Av.H(3,2);
    dmydNv(i)= Av.H(3,3);

    % Elementer til x: (Høyre side av likn)
    Hl(i) = (-Al.g(1)*Tl(i) + Al.g(3)*Nl(i))/Nl(i); % H=TS+myN
    Hv(i) = (-Av.g(1)*Tv(i) + Av.g(3)*Nv(i))/Nv(i); % [J/mol]
    pl(i) = -Al.g(2);                               % [Pa]
    pv(i) = -Av.g(2); % trykk i hvert kontr.volum
    myl(i)= Al.g(3);                                % [J/mol]
    myv(i)= Av.g(3);
end
Dpl = pl - [pl(2:end); b.p_pc_ut(1)];
Dpv = pv - [pv(2:end); b.p_pc_ut(1)];

% I = find(Dpl<0);
% Dpl(I) = 0;
% I = find(Dpv<0);
% Dpv(I) = 0;

rhol = Nl./Vl*b.MW_pc(1);% [kg/m3]
rhov = Nv./Vv*b.MW_pc(1);
rhotot=(Nl+Nv)./b.V_pc(1)*b.MW_pc(1);

wl = b.Cv_pc(:,1).*Vl./b.V_pc(1).*rhol.*sqrt(abs(Dpl)./rhotot).*sign(Dpl);%[
    kgliq/s]
wv = b.Cv_pc(:,1).*Vv./b.V_pc(1).*rhov.*sqrt(abs(Dpv)./rhotot).*sign(Dpv);%[
    kgvap/s]

nl = [wl_ventil; wl]/b.MW_pc(1); % [molliq/s]
nv = [wv_ventil; wv]/b.MW_pc(1); % [molvap/s]

nl_inn = nl(1:end-1);
nl_ut  = nl(2:end);
nv_inn = nv(1:end-1);
nv_ut  = nv(2:end);

Hut = Hl.*nl_ut + Hv.*nv_ut;                % [J/s]
Hinn = [H_inn; Hut(1:end-1)];               % H_inn(1) gitt av u øverst


Udot = Hinn - Hut + Q;                      % [J/s]
Vdot = zeros(b.n,1);
Ndot = (nl_inn+nv_inn) - (nl_ut+nv_ut);     % [mol/s]
deltaT = zeros(b.n,1);
deltap = pl-pv;                             % [Pa]
deltamy= myl-myv;                           % [J/mol]

one = eye(b.n); %diagonalmatrise med enere
zero = zeros(b.n);

x = [Udot; Vdot; Ndot; deltaT; -deltap; -deltamy];
```

```
xx=[x(1),x(21),x(41),x(61),x(81),x(101)]';

M = [diag(dUdTl) diag(dUdVl) diag(dUdNl) diag(dUdTv) diag(dUdVv) diag
    (dUdNv); %Energibalanse
     zero one zero zero one
    zero; %Volumbal
     zero zero one zero zero one
    ; %Massebal
     one zero zero -one zero
    zero; %Templikevekt
     diag(dpdTl) diag(dpdVl) diag(dpdNl) -diag(dpdTv) -diag(dpdVv) -
        diag(dpdNv); %Trykklikev
     diag(dmydTl) diag(dmydVl) diag(dmydNl) -diag(dmydTv) -diag(dmydVv) -
        diag(dmydNv)];%Potensiallikev

M = sparse(M);

l=isreal(M);
if l==0
   disp('Komplekse elementer i M_co2')
end

B=NaN*zeros(size(M));
for i=1:length(M)        % bytt denne til lengde nedover (mer korrekt)
    B(i,:) = b.D1(i)*M(i,:).*b.D2';
end

B = sparse(B);
q = b.D1.*x;
% z er egentlig (skalert) dydt, dvs ydot
z = inv(B)*q; % T,V,N
dy = z.*b.D2;
dy = [dy(1:4*b.n); dy(5*b.n+1:6*b.n)];   % Ønsker ikke å returnere Vv

g=isreal(Udot); if g==0, disp('kompl elem i U'), end
g=isreal(Vdot); if g==0, disp('kompl elem i V'), end
g=isreal(Ndot); if g==0, disp('kompl elem i N'), end
g=isreal(deltaT); if g==0, disp('kompl elem i T'), end
g=isreal(deltap); if g==0, disp('kompl elem i p'), end
g=isreal(deltamy); if g==0, disp('kompl elem i my'), end
```

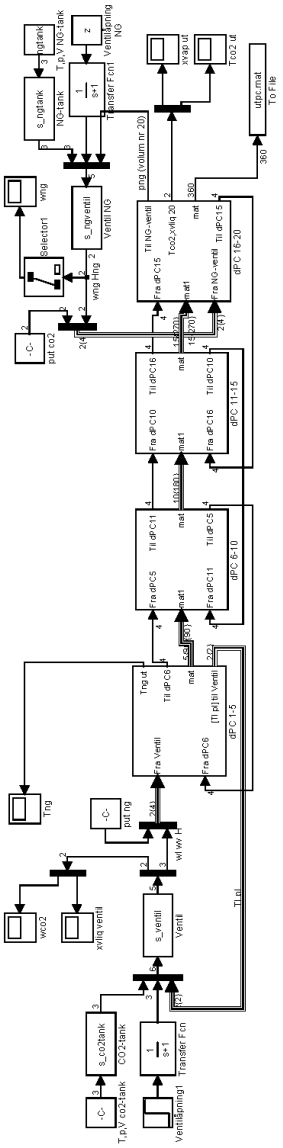
H.1.5 pc_ng.m

```
% pc_ng.m

% Kalles fra pc.m
% Returnerer TVNderivert, tar inn: tilstander, og w og H fra ventil
% samt Q.

function dy=pc_ng(y,u)
global b

w_inn = u(1);       %[kg/m3]
H_inn = u(2);       %[J/s] Totalt for l og v (beregnet i tank.m)
Q = u(3:end);       %[J/s] Beregnet i pc.m. For hvert kontrollvolum

T = y(1:b.n);               % Kolonnevektorer (nedover)
V = y(b.n+1:2*b.n);
N = y(2*b.n+1:3*b.n);
o = NaN*zeros(b.n,1);
dUdT=o; dUdV=o; dUdN=o; H=o; p=o; my=o;
```

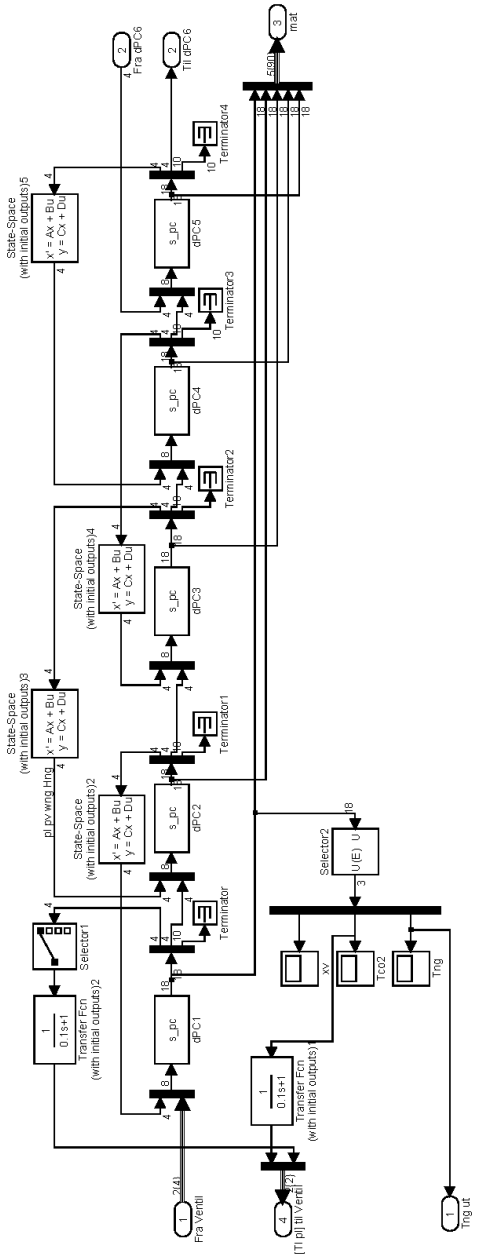
```
for i=1:b.n
    A = ng_pr([T(i),V(i),N(i),N(i)*1e-6, N(i)*1e-6]');

    %Elementer til M-matrisen.
    dUdT(i) = -T(i)*A.H(1,1); % (= Cv)
    dUdV(i) = A.g(2) - T(i)*A.H(1,2);
    dUdN(i) = A.g(3) - T(i)*A.H(1,3);

    % Elementer til x: (Høyre side av likn)
    H(i) = (-A.g(1)*T(i) + A.g(3)*N(i))/N(i); % H=TS+myN [J/mol]
    p(i) = -A.g(2); % trykk i hvert kontr.volum
    my(i)= A.g(3);                                    % [J/mol]
end

Dp = p - [b.p_pc_ut(2); p(1:end-1)];
I = find(Dp<0);
Dp(I) = 0;

rho = N./V*b.MW_pc(2);% [kg/m3]

w = b.Cv_pc(:,2).*rho.*sqrt(abs(Dp)./rho).*sign(Dp);%[kg/s]
w0ng = w;
save massestrom w0ng

n = [w; w_inn]/b.MW_pc(2);        % [mol/s]

n_inn = n(2:end);
n_ut  = n(1:end-1);

Hut = H.*n_ut;                % [J/s]
Hinn = [Hut(2:end); H_inn];% Hinn(1) gitt av u øverst

H0ng = Hut;
save entalpi H0ng;

% Q = Qhatt.*n_ut;
Udot = Hinn - Hut + Q;
Vdot = zeros(b.n,1);
Ndot = n_inn - n_ut;

one = eye(b.n); %diagonalmatrise med enere
zero = zeros(b.n);

x = [Udot; Vdot; Ndot];

M = [diag(dUdT)   diag(dUdV)   diag(dUdN);  %Energibalanse
      zero           one          zero    ;    %Volumbal
      zero           zero         one     ];     %Massebal
M = sparse(M);

l=isreal(M);
if l==0
   disp('Komplekse elementer i M ng')
end

% z er egentlig dydt, dvs ydot
dy = inv(M)*x; % T,V,N
```

H.2 Modell B

I denne modellen er hvert delvolum i varmeveksleren knyttet til en modul i Simulink. Flytskjema er vist i Figur H.2 og H.3.



Figur H.2: Modell B



Figur H.3: Modell B, subsystem dPC 1-5

H.2.1 init.m

```
% init.m

% Initialiserer modell B

load globale_verdier_b  % gir b
b.Cv_pc = [2e-2 4e-1];
b.Cv_ventil=1.2e-3;
b.Cv_ngventil=0.08;

b.UA_pc(1) = 8e4;
b.UA_pc(2) = 1.8e4;

tau=0.1;
G=tf(1,[tau 1]);        % Transferfunksjon
G4x4=ss(append(G,G,G,G));

global b
ni=1;
% Innganger
%-------------------------------------------------------------------------
co2tank = [273+8, b.p_co2tank, 10]';     %T,p,V
ngtank  = [273+8, b.p_ngtank, 10]';

% Regulatorparametre
%-------------------------------------------------------------------
z = 0.5;                  % Ventilåpning

% Startverdier for PC
%-------------------------------------------------------------------------
load startverdier       % gir [pl pv wng Hng png wl wv Htot xv
                        %      Tl Vl Nl Tv Nv Tng Vng Nng Tvegg]'

m=18;
p0l = s_ut(2:m:end,end);
p0v = s_ut(3:m:end,end);
w0ng= s_ut(4:m:end,end);
H0ng= s_ut(5:m:end,end);
p0ng= s_ut(6:m:end,end);
T0l = s_ut(11:m:end,end);

for i=1:b.n
    initB(i).y0 = s_ut(m*(i-1)+11:m*(i-1)+19,end);
end

save initialB initB % til s_pc
```

H.2.2 s_pc.m

```
% s_ngventil.m
function [sys,x0,str,ts] = s_pc(t,x,u,flag,ni)
% u = pl pv wng Hng fra neste, png wl wv Htot fra forrige,

global b
switch flag

  % INITALIZATION
  %--------------
  case 0,
```

```
        [sys,x0,str,ts]=Initialize(ni);

% DERIVATIVES
%------------
case 1,
  dy = pc(t,x,u);   % x er tilstandene TVN og p (y el dydt i pc.m)
  sys= dy;

% OUTPUTS
%--------
case 3,
  Tl = x(1);                % Kolonnevektorer (nedover)
  Vl = x(2);
  Nl = x(3);
  Tv = x(4);
  Vv = b.V_pc(1)-Vl;
  Nv = x(5);
  Tng= x(6);
  Vng= x(7);
  Nng= x(8);
  Tvegg= x(9);

  xv = Nv/(Nv+Nl);

  Al = co2_sw([Tl,Vl,Nl]');
  Av = co2_sw([Tv,Vv,Nv]');
  Ang = ng_pr([Tng,Vng,Nng,Nng*1e-6,Nng*1e-6]');
  pl  =-Al.g(2); % trykk i hvert kontr.volum
  pv  =-Av.g(2);
  png =-Ang.g(2);
  Hl = (-Al.g(1)*Tl + Al.g(3)*Nl)/Nl; % H=TS+myN
  Hv = (-Av.g(1)*Tv + Av.g(3)*Nv)/Nv; % [J/mol]
  Hng = (-Ang.g(1)*Tng + Ang.g(3)*Nng)/Nng; % [J/mol]

  rhol = Nl/Vl*b.MW_pc(1);% [kg/m3]
  rhov = Nv/Vv*b.MW_pc(1);
  rhotot = (Nl+Nv)/b.V_pc(1)*b.MW_pc(1);
  rhong = Nng/Vng*b.MW_pc(2);
  Dpl = pl-u(1);
  Dpv = pv-u(2);
  Dpng= png-u(5);

  I = find(Dpl<0);
  Dpl(I) = 0;
  I = find(Dpv<0);
  Dpv(I) = 0;
  I = find(Dpng<0);
  Dpng(I) = 0;

  wl = b.Cv_pc(1)*Vl/b.V_pc(1)*rhol*sqrt(abs(Dpl)/rhotot)*sign(Dpl);%[
      kgliq/s]
  wv = b.Cv_pc(1)*Vv/b.V_pc(1)*rhov*sqrt(abs(Dpv)/rhotot)*sign(Dpv);%[
      kgvap/s]
  wng = b.Cv_pc(2)*rhong*sqrt(abs(Dpng)/rhong)*sign(Dpng);%[kgNG/s]
  nl = wl/b.MW_pc(1); % [molliq/s]
  nv = wv/b.MW_pc(1); % [molvap/s]
  nng= wng/b.MW_pc(2);

  Htot = Hl*nl + Hv*nv;              % [J/s]
  Hng  = Hng*nng;
```

```
    sys = [pl pv wng Hng png wl wv Htot xv Tl Vl Nl Tv Nv Tng Vng Nng Tvegg
        ]';

  case {2,4,9}
    sys=[];

  % UNEXPECTED FLAGS
  %------------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end


function [sys,x0,str,ts]=Initialize(ni)
global b

sizes = simsizes;

sizes.NumContStates  = 9; % TVN(liq) TN(vap) TVN(ng) T(vegg)
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 18; % pl pv wng Hng png wl wv Htot T(liq),vl,T(ng)
sizes.NumInputs      = 8; % pl pv wng Hng png wl wv Htot
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

load initialB
% initialize the initial conditions
x0 = initB(ni).y0; % TVN(liq) TN(vap) TVN(ng) T(vegg)

% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
% end mdlInitializeSizes
```

H.2.3 pc.m

```
% pc.m

% Inn: TVN liq, TN vap, TVN ng, Tvegg. Ut: deriverte til ODE-løser via s_pc
% put  liq = u(1)
% put  vap = u(2)
% wng fra forrige = u(3)
% Hng fra forrige = u(4)
% pinn ng  = u(5)
% winn liq = u(6)   [kg/s]
% winn vap = u(7)
% Hinn co2 = u(8)   [J/s] Totalt for l og v

function dy=pc(t,y,u)
global b

y_co2 = y(1:5);   %TVNTN
y_ng  = y(6:8);   %TVN
T_vegg= y(9);     %T
```

```
Q_co2 = b.UA_pc(1)*(T_vegg-y_co2(1));    %[J/s]
Q_ng  = b.UA_pc(2)*(T_vegg-y_ng(1));
% Q_ng =0;

dy_co2 = pc_co2(y_co2,[u(1:2);u(6:8);Q_co2]);
% dy_co2 = zeros(5,1);

dy_ng  = pc_ng(y_ng,[u(3:5);Q_ng]);
% dy_ng = zeros(3,1);

% Qco2 > 0. Ut fra vegg--> minus foran
% Qng  < 0. Inn i vegg--> minus foran:
dT_vegg = b.mCp_vegg*(-Q_ng-Q_co2);
% dT_vegg = 0;

dy = [dy_co2; dy_ng; dT_vegg];
```

H.2.4 pc_co2.m

```
% pc_co2.m

% Kalles fra pc.m
% Returnerer TVNderivert, tar inn: tilstander, og pl,pv,wl,wv og Htot fra
% forrige delvolum, samt Q.

function dy=pc_co2(y,u)
global b

pl_neste = u(1);
pv_neste = u(2);
nl_inn = u(3)/b.MW_pc(1);
nv_inn = u(4)/b.MW_pc(1);        %[mol/s]
H_inn = u(5);                    %[J/s] Totalt for l og v
Q = u(6);                        %[J/s] Beregnet i pc.m. For hvert
    kontrollvolum

Tl = y(1);
Vl = y(2);
Nl = y(3);
Tv = y(4);
Vv = b.V_pc(1)-Vl;
Nv = y(5);

if Tl<0
    disp('pc_co2.m, Tl=')
    disp(Tl)
end

Al = co2_sw([Tl,Vl,Nl]');
Av = co2_sw([Tv,Vv,Nv]');

%Elementer til M-matrisen.
dUdTl = -Tl*Al.H(1,1); % (= Cvliq)
dUdVl = Al.g(2) - Tl*Al.H(1,2);
dUdNl = Al.g(3) - Tl*Al.H(1,3);
dUdTv = -Tv*Av.H(1,1); % (= Cvvap)
dUdVv = Av.g(2) - Tv*Av.H(1,2);
dUdNv = Av.g(3) - Tv*Av.H(1,3);

% Elementer til x: (Høyre side av likn)
```

```
Hl = (-Al.g(1)*Tl + Al.g(3)*Nl)/Nl;          % H=TS+myN
Hv = (-Av.g(1)*Tv + Av.g(3)*Nv)/Nv;          % [J/mol]
pl = -Al.g(2);                               % [Pa]
pv = -Av.g(2); % trykk i hvert kontr.volum
myl= Al.g(3);                                % [J/mol]
myv= Av.g(3);

Dpl = pl - pl_neste;
Dpv = pv - pv_neste;

I = find(Dpl<0);
Dpl(I) = 0;
I = find(Dpv<0);
Dpv(I) = 0;

rhol = Nl/Vl*b.MW_pc(1);% [kg/m3]
rhov = Nv/Vv*b.MW_pc(1);
rhotot = (Nl+Nv)/b.V_pc(1)*b.MW_pc(1);

wl = b.Cv_pc(1)*Vl/b.V_pc(1)*rhol*sqrt(abs(Dpl)/rhotot)*sign(Dpl);%[kgliq/s]
wv = b.Cv_pc(1)*Vv/b.V_pc(1)*rhov*sqrt(abs(Dpv)/rhotot)*sign(Dpv);%[kgvap/s]

nl_ut = wl/b.MW_pc(1); % [molliq/s]
nv_ut = wv/b.MW_pc(1); % [molvap/s]

H_ut = Hl*nl_ut + Hv*nv_ut;                  % [J/s]

Udot = H_inn - H_ut + Q;                     % [J/s]
Vdot = 0;
Ndot = (nl_inn+nv_inn) - (nl_ut+nv_ut);      % [mol/s]
deltaT = 0;
deltap = pl-pv;                              % [Pa]
deltamy= myl-myv;                            % [J/mol]


x = [Udot; Vdot; Ndot; -deltaT; -deltap; -deltamy];

M = [dUdTl   dUdVl   dUdNl    dUdTv   dUdVv    dUdNv;    %Energibalanse
       0       1       0        0       1        0;      %Volumbal
       0       0       1        0       0        1;      %Massebal
       1       0       0       -1       0        0;      %Templikevekt
     -Al.H(2,1:3)   Av.H(2,1:3);                          %Trykklikev
      Al.H(3,1:3)  -Av.H(3,1:3)];                         %Potensiallikev


M = sparse(M);

l=isreal(M);
if l==0
   disp('Komplekse elementer i M_co2')
end

B=NaN*zeros(size(M));
for i=1:length(M)
    B(i,:) = b.D1(i)*M(i,:).*b.D2';
end
B = sparse(B);
q = b.D1.*x;

% z er egentlig (skalert) dydt, dvs ydot
z = inv(B)*q; % T,V,N
dy = z.*b.D2;
```

```
dy = [dy(1:4); dy(6)];   % Ønsker ikke å returnere Vv

g=isreal(Udot); if g==0, disp('kompl elem i U'), end
g=isreal(Vdot); if g==0, disp('kompl elem i V'), end
g=isreal(Ndot); if g==0, disp('kompl elem i N'), end
g=isreal(deltaT); if g==0, disp('kompl elem i T'), end
g=isreal(deltap); if g==0, disp('kompl elem i p'), end
g=isreal(deltamy); if g==0, disp('kompl elem i my'), end
```

H.2.5 pc_ng.m

```
% pc_ng.m

% Kalles fra pc.m
% Returnerer TVNderivert, tar inn: tilstander, og w og H fra forrgie
% delvolum, samt Q.

function dy=pc_ng(y,u)
global b

n_inn = u(1)/b.MW_pc(2);          %[mol/s]
H_inn = u(2);                     %[J/s]
p_neste = u(3);
Q = u(4);                         %[J/s]

T = y(1);
V = y(2);
N = y(3);

A = ng_pr([T,V,N,N*1e-6,N*1e-6]');

%Elementer til M-matrisen.
dUdT = -T*A.H(1,1); % (= Cv)
dUdV = A.g(2) - T*A.H(1,2);
dUdN = A.g(3) - T*A.H(1,3);

% Elementer til x: (Høyre side av likn)
H = (-A.g(1)*T + A.g(3)*N)/N; % H=TS+myN [J/mol]
p = -A.g(2); % trykk i hvert kontr.volum
my= A.g(3);                                          % [J/mol]


Dp = p - p_neste;
I = find(Dp<0);
Dp(I) = 0;

rho = N/V*b.MW_pc(2);% [kg/m3]

w = b.Cv_pc(2)*rho*sqrt(abs(Dp)/rho)*sign(Dp);%[m3/s]
n_ut = w/b.MW_pc(2);          % [mol/s]

H_ut = H*n_ut;                % [J/s]

Udot = H_inn - H_ut + Q;
Vdot = 0;
Ndot = n_inn - n_ut;

x = [Udot; Vdot; Ndot];

M = [dUdT   dUdV   dUdN;  %Energibalanse
       0      1      0 ;   %Volumbal
```

```
        0         0         1];       %Massebal
M = sparse(M);

l=isreal(M);
if l==0
    disp('Komplekse elementer i M-ng')
end

% z er egentlig dydt, dvs ydot
dy = inv(M)*x; % T,V,N
```

H.3 Felles rutiner

H.3.1 s_co2tank.m

```
% s_co2tank.m

function [sys,x0,str,ts] = s_pc(t,x,u,flag)

% u = pl pv wng Hng fra neste, png wl wv Htot fra forrige,

global b
switch flag

  % INITALIZATION
  %---------------
  case 0,
       [sys,x0,str,ts]=Initialize;

  % OUTPUTS
  %--------
  case 3,
    T = u(1);
    p = u(2);
    V = u(3);
    rho0 = 1.1e3;
    N0 = rho0*V/b.MW_pc(1);

    % Kun væske i tanken:
    N = N0;
    dN= N;
    while abs(dN)>1e-6*N
        A= co2_sw([T,V,N]');
        dN = (p + A.g(2))/(-A.H(2,3));
        N = N + dN;
    end
    rho = N/V*b.MW_pc(1); % [kg/m3]
    H = (-A.g(1)*T + A.g(3)*N)/N;    % [J/mol]

    sys = [p, rho, H]';

  case { 1, 2, 4, 9 }
    sys=[];

  % UNEXPECTED FLAGS
  %-----------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end


function [sys,x0,str,ts]=Initialize(ni)
global b

sizes = simsizes;

sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 3; % p rho H
sizes.NumInputs      = 3; % T,p,V,rho0
```

```
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

x0 = [];

% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
```

H.3.2 s_ngtank.m

```
% s_ngtank.m

function [sys,x0,str,ts] = s_pc(t,x,u,flag)
% u = pl pv wng Hng fra neste, png wl wv Htot fra forrige,

global b
switch flag

  % INITALIZATION
  %--------------
  case 0,
       [sys,x0,str,ts]=Initialize;

  % OUTPUTS
  %--------
  case 3,
    T = u(1);
    p = u(2);
    V = u(3);
    N0 = p*V/(b.R*T);
    % Kun væske i tanken:
    N = N0;
    dN= N;
    while abs(dN)>1e-6*N
        A= ng_pr([T,V,N,N*1e-6,N*1e-6]');
        dN = (p + A.g(2))/(-A.H(2,3));
        N = N + dN;
    end
    rho = N/V*b.MW_pc(2); % [kg/m3]
    H = (-A.g(1)*T + A.g(3)*N)/N;    % [J/mol]

    sys = [p, rho, H]';

  case { 1, 2, 4, 9 }
    sys=[];

  % UNEXPECTED FLAGS
  %-----------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end

function [sys,x0,str,ts]=Initialize(ni)
```

```
global b

sizes = simsizes;

sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 3; % p rho H
sizes.NumInputs      = 3; % T,p,V
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

% initialize the initial conditions
x0 = []; % TVN(liq) TN(vap) TVN(ng) T(vegg)

% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
% end mdlInitializeSizes
```

H.3.3 s_co2ventil.m

```
% s_co2ventil.m

function [sys,x0,str,ts] = s_co2ventil(t,x,u,flag)
% u = [p rho H]_tank, z, [p T]_1.kontrollvolumPC
global b
switch flag


  % INITALIZATION
  %--------------
  case 0,
    [sys,x0,str,ts]=Initialize;


  % OUTPUTS
  %-------
  case 3,
    sys = co2ventil(u);

  case { 1, 2, 4, 9 }
    sys=[];

  % UNEXPECTED FLAGS
  %-----------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end

% initialisering
function [sys,x0,str,ts]=Initialize
global b

sizes = simsizes;
```

```
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 5; % wl wv Htot
sizes.NumInputs      = 6; % rho,H,pinn, z, p og T etter ventil
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

% initialize the initial conditions
x0 = []; % temp
%
% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
% end mdlInitializeSizes
```

H.3.4 co2ventil.m

```
% co2ventil.m

% Kalles fra s_co2ventil.m
% Tar inn trykk i p, hro, H fra s_co2tank.m, ventilåpning z samt T og p fra
% første delvolum i varmeveksleren. Sender ut wtot og xvap (til plott) samt
% wl, wv og Htot til s_pc.m

function ut=co2ventil(u)
global b

pinn= u(1);
rhoinn= u(2);
Hinn= u(3);      % [J/mol]
z = u(4);
Tut = u(5);
put = u(6);

wtot = z*b.Cv_ventil*rhoinn*sqrt(abs(pinn-put)/rhoinn);

Vl = 10;        % Bare tar et volum..
Nl_init= rhoinn*Vl/b.MW_pc(1);
Nl = Nl_init;
dNl= Nl;
while abs(dNl)>1e-6*Nl
    Al = co2_sw([Tut,Vl,Nl]);
    dNl= (put + Al.g(2))/(-Al.H(2,3));
    Nl = Nl + dNl;
end
Hl = (-Al.g(1)*Tut + Al.g(3)*Nl)/Nl;    % [J/mol]

Vv = 10;
Nv_init= put*Vv/(b.R*Tut);
Nv = Nv_init;
dNv= Nv;
while abs(dNv)>1e-6*Nv
    Av = co2_sw([Tut,Vv,Nv]);
    dNv= (put + Av.g(2))/(-Av.H(2,3));
```

```
    Nv = Nv + dNv;
end
Hv = (-Av.g(1)*Tut + Av.g(3)*Nv)/Nv;    % [J/mol]

wl = wtot*(1-Hinn/Hv)/(1-Hl/Hv);
wv = wtot-wl;

Hut = Hinn*wtot/b.MW_pc(1);       % [J/s]

xvap = Nv/(Nl+Nv);

ut = [wtot; xvap; wl; wv; Hut];
```

H.3.5 s_ngventil.m

```
% s_ngventil.m

function [sys,x0,str,ts] = s_ventil(t,x,u,flag)
% u = [T p] i siste kontrollvolum til PC
global b
switch flag


  % INITALIZATION
  %--------------
  case 0,
    [sys,x0,str,ts]=Initialize;


  % OUTPUTS
  %--------
  case 3,
    pinn= u(1);
    rhoinn= u(2);
    Hinn= u(3);      % [J/mol]
    z = u(4);
%     Tut = u(5);
    put = u(5);

    w = z*b.Cv_ngventil*rhoinn*sqrt(abs(pinn-put)/rhoinn);
    Hut = Hinn*w/b.MW_pc(2);          %[J/s]
    sys = [w Hut]';

  case { 1, 2, 4, 9 }
    sys=[];

  % UNEXPECTED FLAGS
  %-----------------
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end

function [sys,x0,str,ts]=Initialize
global b

sizes = simsizes;

sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 2; % w H
```

```
sizes.NumInputs      = 5; % pinn,rho,H, z, p og T etter ventil
sizes.DirFeedthrough = 1; % 0 hvis utgang ikke direkte funksjon av inngang,
    ellers 1
sizes.NumSampleTimes = 1; % at least one sample time is needed

sys = simsizes(sizes);

% initialize the initial conditions
x0 = []; % temp
%
% str is always an empty matrix
str = [];

% initialize the array of sample times
ts  = [0 0];
%data=initialize;
% end mdlInitializeSizes
```

H.3.6 co2_sw.m

Fra databasen DIPPR96[24] hentes data for CO_2 for å generere en standardtilstand, en ideell tilstandslikning og et reelt residualledd til tilstandslikningen (i dette tilfellet Span-Wagner). Standardtilstanden genereres ved hjelp av kjemisk potensial, dannelsesentalpi og standard entropi for CO_2. I den reelle tilstandslikningen benyttes verdier som kritisk trykk og temperatur. Rutinen er utviklet av Tore Haug-Warberg ved hjelp av følgende: [23]

```
### Fundamental CO2 equation of state (steam table)

n   = ['carbon dioxide']
co2 = Surface.new('CO2') * (
         Helmholtz.new() * (
           StandardState.new() * (
             MuT_cp.new(:ig,:dippr,:dippr96) * (
               MuT_hs.new(:ig,:h0,:dippr96) +
               MuT_hs.new(:ig,:s0,:dippr96).mixture(n)
             )
           ) +
           EquationOfState.new() * (
             ModTVN_ideal.new(:fluid,:idealgas).mixture(n)
           ) +
           EquationOfState.new() * (
             ModTVN.new(:fluid,:sw12polar,:sw03).mixture(n)
           )
         )
       )
```

co2.LaTeX_report!

:sw12polar,:sw03

```
% co2_sw.m

function [S] = co2_sw(x)

[S] = Helmholtz_anonymous_23829804(x);


function [S] = Helmholtz_anonymous_23829804(x)

[S] = StandardState_anonymous_23817360(x);


function [S] = StandardState_anonymous_23817360(x)

[S]       = EquationOfState_anonymous_23031408(x);
i         = [3];
[S_1]     = MuT_cp_dippr_23809380(x(1));
S.g(1)    = S.g(1) + S_1.dmudT'*x(i);
S.g(i)    = S.g(i) + S_1.mu;
S.H(1,1) = S.H(1,1) + S_1.d2mudTdT'*x(i);
S.H(i,1) = S.H(i,1) + S_1.dmudT;
S.H(1,i) = S.H(i,1)';


function [S] = EquationOfState_anonymous_23031408(x)

[S]   = EquationOfState_anonymous_22908308(x);
[S_1] = ModTVN_ideal_idealgas_23028984(x);
S.g   = S.g + S_1.g;
S.H   = S.H + S_1.H;


function [S] = EquationOfState_anonymous_22908308(x)

[S] = ModTVN_sw_0_5_7_0_23818404(x);


function [S] = ModTVN_sw_0_5_7_0_23818404(x)

R               = 8.314511984;
T_c             = 304.1282;
rho_c           = 10624.90627;
tau             = T_c/x(1);
delta           = x(3)/x(2)/rho_c;
N               = x(3);
a_1             = [0.89875108; -2.1281985;-0.06819032; 0.076355306;
    0.00022053253];
t_1             = [0.25;1.25;1.5;0.25;0.875];
d_1             = [1.0;1.0;1.0;3.0;7.0];
a_2             = [0.41541823;0.71335657; 0.00030354234; -0.36643143;
    -0.0014407781; -0.089166707;-0.023699887];
t_2             = [2.375;2.0;2.125;3.5;6.5;4.75;12.5];
d_2             = [1.0;2.0;5.0;1.0;1.0;4.0;2.0];
p_2             = [1.0;1.0;1.0;2.0;2.0;2.0;3.0];
u_2             = d_2 - p_2.*delta.^p_2;
v_2             = exp(-delta.^p_2);
```

```
phir            = a_1'*(delta.^d_1.*tau.^t_1) + a_2'*(v_2.*delta.^d_2.*tau.^
    t_2);
phi_taur        = a_1'*(delta.^d_1.*t_1.*tau.^(t_1 - 1)) + a_2'*(v_2.*delta
    .^d_2.*t_2.*tau.^(t_2 - 1));
phi_deltar      = a_1'*(d_1.*delta.^(d_1 - 1).*tau.^t_1) + a_2'*(v_2.*delta
    .^(d_2 - 1).*u_2.*tau.^t_2);
phi_tautaur     = a_1'*(delta.^d_1.*t_1.*(t_1 - 1).*tau.^(t_1 - 2)) + a_2'*(
    v_2.*delta.^d_2.*t_2.*(t_2 - 1).*tau.^(t_2 - 2));
phi_deltadeltar = a_1'*(d_1.*(d_1 - 1).*delta.^(d_1 - 2).*tau.^t_1) + a_2'*(
    v_2.*delta.^(d_2 - 2).*(u_2.*(u_2 - 1) - p_2.^2.*delta.^p_2).*tau.^t_2);
phi_deltataur   = a_1'*(d_1.*delta.^(d_1 - 1).*t_1.*tau.^(t_1 - 1)) + a_2'*(
    v_2.*delta.^(d_2 - 1).*t_2.*u_2.*tau.^(t_2 - 1));
g_1             = N*R*(phir - tau*phi_taur);
g_2             = -R*T_c*rho_c*delta^2*phi_deltar/tau;
g_i             = R*T_c*(phir + delta*phi_deltar)/tau;
H_11            = N*R*tau^3*phi_tautaur/T_c;
H_21            = -R*delta^2*rho_c*(phi_deltar - tau*phi_deltataur);
H_i1            = R*(phir - tau*phi_taur + delta*(phi_deltar - tau*
    phi_deltataur));
H_22            = R*T_c*rho_c^2*delta^3*(delta*phi_deltadeltar + 2*
    phi_deltar)/(tau*N);
H_i2            = -R*T_c*rho_c*delta^2*(delta*phi_deltadeltar + 2*phi_deltar
    )/(tau*N);
H_ii            = R*T_c*delta*(delta*phi_deltadeltar + 2*phi_deltar)/(tau*N)
    ;
S.g             = [g_1;g_2;g_i];
S.H             = [H_11,H_21,H_i1;H_21,H_22,H_i2;H_i1,H_i2,H_ii];


function [S] = ModTVN_ideal_idealgas_23028984(x)

R    = 8.314511984;
pcirc = [101325.0];
xcirc = [1.0];
T    = x(1);
V    = x(2);
i    = [3];
n    = x(i);
NR   = sum(n)*R;
NRT  = NR*T;
e    = [1];
p    = R*log(R*T*(n./pcirc)/V);
g_1  = n'*p;
g_2  = -NRT/V;
g_i  = T*p;
H_11 = NR/T;
H_21 = -NR/V;
H_i1 = p + R*e;
H_22 = NRT/V^2;
H_i2 = -R*(T/V)*e;
H_ii = R*T*diag(e./n);
S.g  = [g_1;g_2;g_i];
S.H  = [H_11,H_21',H_i1';H_21,H_22,H_i2';H_i1,H_i2,H_ii];


function [S] = MuT_cp_dippr_23809380(T)

t_0      = [298.15];
t_min    = [50.0];
t_max    = [5000.0];
C        = [29.37,34.54,1428.0,26.4,588.0];
c_1      = C(:,1);
```

```
c_2          = C(:,2);
c_3          = C(:,3);
c_4          = C(:,4);
c_5          = C(:,5);
c_p          = c_1 + c_2.*c_3.^2./(T^2*sinh(c_3/T).^2) + c_4.*c_5.^2./(T^2*
    cosh(c_5/T).^2);
h            = c_1*T + c_2.*c_3.*cosh(c_3/T)./sinh(c_3/T) - c_4.*c_5.*sinh(c_5
    /T)./cosh(c_5/T) - c_1.*t_0 - c_2.*c_3.*cosh(c_3./t_0)./sinh(c_3./t_0)
     + c_4.*c_5.*sinh(c_5./t_0)./cosh(c_5./t_0);
s            = c_1*log(T) + c_4.*(log(cosh(c_5/T)) - c_5.*sinh(c_5/T)./cosh(
    c_5/T)/T) + c_2.*(c_3.*cosh(c_3/T)./sinh(c_3/T)/T - log(sinh(c_3/T))) -
    c_1.*log(t_0) - c_4.*(log(cosh(c_5./t_0)) - c_5.*sinh(c_5./t_0)./cosh(
    c_5./t_0)./t_0) - c_2.*(c_3.*cosh(c_3./t_0)./sinh(c_3./t_0)./t_0 - log(
    sinh(c_3./t_0)));
S.mu         = h - T*s;
S.dmudT      = -s;
S.d2mudTdT = -(c_p/T);
i            = [1];
[S_1]        = MuT_hs_h0_23796420(T);
S.mu(i)      = S.mu(i) + S_1.mu;
S.dmudT(i) = S.dmudT(i) + S_1.dmudT;


function [S] = MuT_hs_h0_23796420(T)

hcirc    = [-393510.0];
[S]      = MuT_hs_s0_23790264(T);
S.mu     = S.mu + hcirc;
S.dmudT = S.dmudT + [0];


function [S] = MuT_hs_s0_23790264(T)

scirc    = [213.677];
S.mu     = -(T*scirc);
S.dmudT = -scirc;
```

H.3.7 ng_pr.m

```
### LNG

n = ["metan","ethane","propane"]

gas  = Surface.new(:LNG) * (
         Helmholtz.new() * (
           StandardState.new() * (
             MuT_cp.new(:ig,:poly3,:reid77) * (
               MuT_hs.new(:ig,:h0,:reid87) +
               MuT_hs.new(:ig,:s0,:dippr96).mixture(n)
             )
           ) +
           EquationOfState.new() * (
```

```
            ModTVN_ideal.new(:gas,:idealgas).mixture(n)
          ) +
          EquationOfState.new() * (

ModTVN.new(:gas,:srk,:reid77).
update(:fluid,[],:pr,:dippr96.
update(:fluid,[:a,[:mfac,[],:m_pr,:dippr96]],:pr,:reid87
).mixture(n)
          )
        )
      )
```

```
function [S] = lng(x)

[S] = Helmholtz_anonymous_21985668(x);


function [S] = Helmholtz_anonymous_21985668(x)

[S] = StandardState_anonymous_21977700(x);


function [S] = StandardState_anonymous_21977700(x)

[S]       = EquationOfState_anonymous_21611580(x);
i         = [3,4,5];
[S_1]     = MuT_cp_poly3_21965832(x(1));
S.g(1)    = S.g(1) + S_1.dmudT'*x(i);
S.g(i)    = S.g(i) + S_1.mu;
S.H(1,1) = S.H(1,1) + S_1.d2mudTdT'*x(i);
S.H(i,1) = S.H(i,1) + S_1.dmudT;
S.H(1,i) = S.H(i,1)';


function [S] = EquationOfState_anonymous_21611580(x)

[S]   = EquationOfState_anonymous_21601848(x);
[S_1] = ModTVN_ideal_idealgas_21608604(x);
S.g   = S.g + S_1.g;
S.H   = S.H + S_1.H;


function [S] = EquationOfState_anonymous_21601848(x)

[S] = ModTVN_pr_21599040(x);


function [S] = ModTVN_pr_21599040(x)

R                          = 8.314511984;
d_1                        = 2.414213562;
d_2                        = 0.4142135624;
t_c                        = [190.4;305.4;369.8];
p_c                        = [4600000.0;4880000.0;4250000.0];
```

```
Omega_b                          = 0.0777960739;
T                                = x(1);
V                                = x(2);
i                                = [3,4,5];
n                                = x(i);
e                                = [1;1;1];
c                                = 1/(d_1 + d_2);
NR                               = sum(n)*R;
RT                               = R*T;
NRT                              = NR*T;
dBdn                             = Omega_b*R*(t_c./p_c);
B                                = n'*dBdn;
C_1circ                          = (1/(V + d_1*B) - 1/(V - d_2*B))/B;
C_2circ                          = (-d_1/(V + d_1*B)^2 - d_2/(V - d_2*B)^2 -
    C_1circ)/B;
C_lnd                            = log((V + d_1*B)/(V - d_2*B));
C_lnv                            = log(V/(V - B));
C_1                              = (d_1/(V + d_1*B) + d_2/(V - d_2*B) - C_lnd/B)/B
    ;
C_2                              = (-(d_1/(V + d_1*B))^2 + (d_2/(V - d_2*B))
    ^2 - 2*C_1)/B;
[d2AdTdT,d2AdndT,d2AdndnT] = ModTVN_pr_a_soave_21582048(T,n,R,t_c,p_c);
dAdT                             = d2AdndT'*n/2;
dAdn                             = d2AdndnT*n;
A                                = dAdn'*n/2;
g_1                              = NR*C_lnv - c*dAdT*C_lnd/B;
g_2                              = -NRT*B/(V*(V - B)) + A/((V + d_1*B)*(V - d_2*B)
    );
g_i                              = RT*C_lnv*e + (NRT/(V - B) - c*A*C_1)*dBdn - c*
    C_lnd/B*dAdn;
H_11                             = -c*d2AdTdT*C_lnd/B;
H_21                             = -B*NR/(V*(V - B)) + dAdT/((V + d_1*B)*(V - d_2*
    B));
H_i1                             = R*C_lnv*e + (NR/(V - B) - c*dAdT*C_1)*dBdn - c*
    C_lnd/B*d2AdndT;
H_22                             = -NRT/V^2 + NRT/(V - B)^2 + c*A*(1/(V + d_1*B)
    ^2 - 1/(V - d_2*B)^2)/B;
H_i2                             = -RT*B/(V*(V - B))*e - (NRT/(V - B)^2 + c*A*
    C_2circ)*dBdn - c*C_1circ*dAdn;
H_ii                             = RT/(V - B)*(e*dBdn' + dBdn*e') + (NRT/(V - B)
    ^2 - c*A*C_2)*dBdn*dBdn' - c*C_1*(dAdn*dBdn' + dBdn*dAdn') - c*C_lnd/B*
    d2AdndnT;
S.g                              = [g_1;g_2;g_i];
S.H                              = [H_11,H_21',H_i1';H_21,H_22,H_i2';H_i1,H_i2,
    H_ii];


function [d2AdTdT,d2AdndT,d2AdndnT] = ModTVN_pr_a_soave_21582048(T,n,R,t_c,
    p_c)

e         = [1;1;1];
Omega_a   = 0.4572355289;
K         = [0,0,0;0,0,0;0,0,0];
r_c       = t_c./sqrt(p_c);
F         = 2*Omega_a*R^2*(r_c*r_c'.*(e*e' - K));
[m]       = ModTVN_pr_a_soave_m_pr_21579792;
k         = e + m.*(e - sqrt(T./t_c));
k_t       = -1/(2*T)*(m - k + e);
k_tt      = -1/(2*T)*k_t;
d2AdTdT   = (n.*k_tt)'*F*(k.*n) + (n.*k_t)'*F*(k_t.*n);
d2AdndT   = k_t.*(F*(k.*n)) + k.*(F*(k_t.*n));
d2AdndnT = F.*(k*k');
```

```
function [m] = ModTVN_pr_a_soave_m_pr_21579792

omega_az = [0.0115478;0.099493;0.152291];
e        = [1;1;1];
m        = 0.37464*e + 1.54226*omega_az + -0.2699*omega_az.^2;


function [S] = ModTVN_ideal_idealgas_21608604(x)

R     = 8.314511984;
pcirc = [101325.0;101325.0;101325.0];
xcirc = [1.0;1.0;1.0];
T     = x(1);
V     = x(2);
i     = [3,4,5];
n     = x(i);
NR    = sum(n)*R;
NRT   = NR*T;
e     = [1;1;1];
p     = R*log(R*T*(n./pcirc)/V);
g_1   = n'*p;
g_2   = -NRT/V;
g_i   = T*p;
H_11  = NR/T;
H_21  = -NR/V;
H_i1  = p + R*e;
H_22  = NRT/V^2;
H_i2  = -R*(T/V)*e;
H_ii  = R*T*diag(e./n);
S.g   = [g_1;g_2;g_i];
S.H   = [H_11,H_21',H_i1';H_21,H_22,H_i2';H_i1,H_i2,H_ii];


function [S] = MuT_cp_poly3_21965832(T)

tcirc       = [298.15;298.15;298.15];
t           = [T;T;T];
e           = [1;1;1];
t_1         = t - tcirc;
t_2         = (t.^2 - tcirc.^2)/2;
t_3         = (t.^3 - tcirc.^3)/3;
t_4         = (t.^4 - tcirc.^4)/4;
C           = [19.2451589,0.0521100975,1.1970673e-005,-1.131354165e
    -008;5.4077306,0.178053297,-6.93545635e-005,8.71012955e
    -009;-4.22321995,0.3061729825,-0.0001585904895,3.21366529e-008];
c_p         = sum(C.*[e,t,t.^2,t.^3],2);
s           = sum(C.*[log(t./tcirc),t_1,t_2,t_3],2);
h           = sum(C.*[t_1,t_2,t_3,t_4],2);
S.mu        = h - T*s;
S.dmudT     = -s;
S.d2mudTdT = -(c_p/T);
i           = [1,2,3];
[S_1]       = MuT_hs_h0_21957912(T);
S.mu(i)     = S.mu(i) + S_1.mu;
S.dmudT(i) = S.dmudT(i) + S_1.dmudT;


function [S] = MuT_hs_h0_21957912(T)

hcirc    = [-74900.0;-84740.0;-103900.0];
```

```
[S]      = MuT_hs_s0_21945552(T);
S.mu     = S.mu + hcirc;
S.dmudT = S.dmudT + [0;0;0];


function [S] = MuT_hs_s0_21945552(T)

scirc    = [186.27;229.12;270.2];
S.mu     = -(T*scirc);
S.dmudT = -scirc;
```

H.4 Første initialisering

H.4.1 init_mod.m

```
% init_mod.m
% Førstegangs-oppstart

clear all
close all
global b d

% Generelle parametre
%-------------------------------------------------------------------------
b.n = 20;                           % Antall delvolum i pc
b.R = 8.314;                        % [J/molK] Gasskonst

% Molvekt MW
%-------------------------------------------------------------------------
b.MW_pc  = [44e-3 16e-3 27e-3];  % [kg/mol]

% Varmekapasitet Cp
%-------------------------------------------------------------------------
Cp_pc = [37 36 24];                 % [J/molK]
Cpm_pc = [49.2844 Cp_pc(2)/b.MW_pc(2) Cp_pc(3)/b.MW_pc(3)]; % [J/kgK]
m_pcvegg = 10e3;     % [kg]
b.mCp_vegg = Cpm_pc(3)*m_pcvegg; % [J/K]

% Tetthet rho
%-------------------------------------------------------------------------
d.rho_co2liq = 1.1e3;               % [kg/m3] gjett for CO2 liq..

%-------------------------------------------------------------------------
% ANLEGGSDATA
%-------------------------------------------------------------------------
%
% Trykk
%-------------------------------------------------------------------------
b.p_pc_inn = [27e5 70e5];           % [Pa]
Dp_pc = [1e5 1e5];                  % Trykkfall over hele varmeveksleren
b.p_pc_ut = b.p_pc_inn - Dp_pc;
dp_pc = Dp_pc/(b.n+1);
c.p_pc = [b.p_pc_inn(1)-dp_pc(1):-dp_pc(1):b.p_pc_ut(1)+dp_pc(1);
          b.p_pc_ut(2)+dp_pc(2):dp_pc(2):b.p_pc_inn(2)-dp_pc(2)]';
b.p_co2tank = 42e5;
b.p_ngtank  = 71e5;

% Temperaturer, stasjonære
%-------------------------------------------------------------------------
d.T_pc_inn = [-11 8 -9]+273;        % co2 ng vegg
T_pc_ut = [-11 -7 -5]+273;
dT_pc = (T_pc_ut-d.T_pc_inn)/b.n; % CO2: Temp er konstant under fordampning
    ...
c.T_pc = [d.T_pc_inn(1)*ones(1,b.n);
          T_pc_ut(2):-dT_pc(2):d.T_pc_inn(2)+dT_pc(2);
          d.T_pc_inn(3):dT_pc(3):T_pc_ut(3)-dT_pc(3)]';
d.T_co2tank = 8+273;

% Volumer
%-------------------------------------------------------------------------
b.V_pc = [7/b.n 7/b.n];             % [m3]    co2 ng
```

```
d.V_co2tank = 10;
d.V_ngtank = 10;
c.vvs = .2; %volumfraksjon gass i tank før PC
c.vvf = .9; %siste kontrollvolum

y0start = starttilstander(b,c,d); % startverdier for TVN i pc
% save foerste_y0 y0start

% Stasjonære strømmer
%-------------------------------------------------------------------------
w_ng   = 8e5/3600;                  % [kg/s]
w_co2 = 50;   % [kg/s]

Q_ng = w_ng*Cpm_pc(2)*(T_pc_ut(2)-d.T_pc_inn(2)); %[J/s]

tank(b,d)
load tankverdier            % gir co2_tank og ng_tank = tettheter og
    entalpier inn på pc
rho_co2tank = co2_tank(1);
H_co2tank = co2_tank(2);
rho_ngtank = ng_tank(1);
H_ngtank = ng_tank(2);

% Regulatorparametre
%-------------------------------------------------------------------------
z = 0.5;                % Ventilåpning

% UA-verdier
%-------------------------------------------------------------------------
DT = [c.T_pc(b.n/2,3)-c.T_pc(b.n/2,1) c.T_pc(b.n/2,3)-c.T_pc(b.n/2,2)];
b.UA_pc = [-Q_ng/DT(1) Q_ng/DT(2)]/b.n;                 % [J/sK]
b.UA_pc(1) = 1.0e5;
b.UA_pc(2) = 1.0e4;

% Cv-verdier
%-------------------------------------------------------------------------
b.Cv_ventil =  w_co2/rho_co2tank/sqrt(abs(b.p_co2tank-c.p_pc(1,1))/
    rho_co2tank);
b.Cv_ngventil =w_ng/rho_ngtank/sqrt(abs(b.p_ngtank-c.p_pc(end,2))/rho_ngtank
    );

NV0 = starttilstand(b,c,d);   % startverdier for TVN i pc
N_l = NV0(1:b.n);
V_l = NV0(b.n+1:2*b.n);
N_v = NV0(2*b.n+1:3*b.n);
V_v = NV0(3*b.n+1:4*b.n);
N_ng= NV0(4*b.n+1:5*b.n);

N_co2 = N_l+N_v;            % [mol]
rho_pc = [N_co2/b.V_pc(1)*b.MW_pc(1) N_ng/b.V_pc(2)*b.MW_pc(2)];
b.Cv_pc=[w_co2./rho_pc(:,1)./sqrt(abs(dp_pc(1)./rho_pc(:,1))) w_ng./rho_pc
    (:,2)./sqrt(abs(dp_pc(2))./rho_pc(:,2))];
% Disse gjelder for UT av kontr.vol, totalstrøm (L+V)

% Skaleringsfaktorer:
%-------------------------------------------------------------------------
b.D2 = [c.T_pc(1,1); V_l(1); N_l(1); c.T_pc(1,1); V_v(1); N_v(1)];

b.D1 = [1/(N_co2(1)*b.R.*c.T_pc(1,1));
        1/(V_l(1)+V_v(1));
        1/N_co2(1);
        1/c.T_pc(1,1);
```

```
        b.V_pc(1)/(N_co2(1)*b.R*c.T_pc(1,1));
        1/(b.R*c.T_pc(1,1))];

save globale_verdier_b b
```

H.4.2 starttilstand.m

```
% starttilstand.m
% Finner startverdier i PC

% Id: starttilstand.m,v1.12004/05/2201:05:40IngridExp

function NV0=starttilstand(b,c,d)

% CO2-siden:
T_co2 = c.T_pc(:,1);                % [K] Temperatur i hvert kontr.vol lik
p_co2 = c.p_pc(:,1);                % p settes

vvs = c.vvs;
vvf = c.vvf;
dvv = (vvf-vvs)/b.n;
vv = [vvs+dvv:dvv:vvf]';            % volumfraksj gass
Vv = b.V_pc(:,1).*vv;               % V settes
Vl = b.V_pc(:,1) - Vv;

Nv0 = p_co2.*Vv./(b.R*T_co2);       % [mol] (gjett for gass)

% Naturgass-siden:
T_ng = c.T_pc(:,2);
V_ng = b.V_pc(2);
p_ng = c.p_pc(:,2);

N0_ng= p_ng.*V_ng./(b.R*T_ng);

o=NaN*zeros(b.n,1); Nl=o;Nv=o;N_ng=o;
for i=1:length(p_co2)

    % Newtoniterasjoner, til N ved ønsket trykk finnes:
    Nv(i) = Nv0(i);
    dNv = Nv(i);
    while abs(dNv)>1e-6*Nv(i)
        Av = co2_sw([T_co2(i),Vv(i),Nv(i)]');
        dNv = (p_co2(i) + Av.g(2))/(-Av.H(2,3));
        Nv(i) = Nv(i) + dNv;
    end

    % Bytte ut denne med den forige rho.
    Nl(i) = d.rho_co2liq*Vl(i)/b.MW_pc(1);% (gjett for væske)
    dNl = Nl(i);
    while abs(dNl)>1e-6*Nl(i)
        Al = co2_sw([T_co2(i),Vl(i),Nl(i)]');
        dNl = (p_co2(i) + Al.g(2))/(-Al.H(2,3));
        Nl(i) = Nl(i) + dNl;
    end

    % Er det dumt å sette etan og propan lik null? Er de som nr 2 og 3?
    N_ng(i) = N0_ng(i);
    dN_ng = N_ng(i);
    while abs(dN_ng)>1e-6*N_ng(i)
        %A_ng = ng_pr([T_ng(i),V_ng(i),N_ng(i),0,0]');
        A_ng = ng_pr([T_ng(i),V_ng,N_ng(i),N_ng(i)*1e-6,N_ng(i)*1e-6]');
```

```
        dN_ng= (p_ng(i) + A_ng.g(2))/(-A_ng.H(2,3));
        N_ng(i) = N_ng(i) + dN_ng;
    end
end

% Startverdier for hver tilstand/meidum i hvert kontrollvolum:
%-------------------------------------------------------------------------
NV0 = [Nl; Vl; Nv; Vv; N_ng];
```

H.4.3 tank.m

```
% tank.m
% Gir entalpien inn på PC for CO2 og NG, og tetthet for CO2

function tank(b,d)
% CO2:
T_co2 = d.T_co2tank;
p_co2 = b.p_co2tank;
V_co2 = d.V_co2tank;
N0_co2 = d.rho_co2liq*V_co2/b.MW_pc(1);


% Væske:
N_co2 = N0_co2;
dN_co2= N_co2;
while abs(dN_co2)>1e-6*N_co2
    A_co2 = co2_sw([T_co2,V_co2,N_co2]);
    dN_co2 = (p_co2 + A_co2.g(2))/(-A_co2.H(2,3));
    N_co2 = N_co2 + dN_co2;
end
rho_co2 = N_co2/V_co2*b.MW_pc(1); % [kg/m3]
H_co2 = (-A_co2.g(1)*T_co2 + A_co2.g(3)*N_co2)/N_co2;    % [J/mol]

% Naturgass:
T_ng = d.T_pc_inn(2);
V_ng = d.V_ngtank;
N0_ng= b.p_pc_inn(2)*V_ng/(b.R*T_ng);

N_ng = N0_ng;
dN_ng= N_ng;
while abs(dN_ng)>1e-6*N_ng
    A_ng = ng_pr([T_ng, V_ng, N_ng, N_ng*1e-6, N_ng*1e-6]');
    dN_ng = (b.p_ngtank + A_ng.g(2))/(-A_ng.H(2,3));
    N_ng = N_ng + dN_ng;
end
rho_ng = N_ng/V_ng*b.MW_pc(2); % [kg/m3]
H_ng = (-A_ng.g(1)*T_ng + A_ng.g(3)*N_ng)/N_ng; %[J/mol]

%Resultat:
co2_tank = [rho_co2, H_co2];
ng_tank  = [rho_ng, H_ng];
save tankverdier co2_tank ng_tank
```

H.5 Plotting av data

H.5.1 plotter.m

```
% plotter.m

function grafer = plotter(x,b)

% Gir lokale navn
%-------------------------------------------------------------------------
m=18;
t   = x(1,:);
pl  = x(2:m:end,:);
pv  = x(3:m:end,:);
wng= x(4:m:end,:);
Hng= x(5:m:end,:);
png= x(6:m:end,:);
wl  = x(7:m:end,:);
wv  = x(8:m:end,:);
wco2 = wl+wv;
Htot = x(9:m:end,:);
xv  = x(10:m:end,:);
Tl  = x(11:m:end,:);
Vl  = x(12:m:end,:);
Nl  = x(13:m:end,:);
Tv  = x(14:m:end,:);
Vv  = b.V_pc(1)-Vl;
Nv  = x(15:m:end,:);
Tng= x(16:m:end,:);
Vng= x(17:m:end,:);
Nng= x(18:m:end,:);
Tvegg = x(19:m:end,:);

% Plotter
%-------------------------------------------------------------------------
warning off
n = 1:b.n;

subplot(2,2,1)
plot(t,Tng(end,:))
title('NG-temperatur ut')
ylabel('[K]')

subplot(2,2,2)
plot(t,Tl(end,:))
title('CO2-temperatur ut')
ylabel('[K]')

subplot(2,2,3)
plot(t,pl(1,:),t,pl(end,:))
title('CO2-trykk')
ylabel('[Pa]'), legend('Første delvolum', 'Siste delvolum')

subplot(2,2,4)
plot(t,xv(end,:))
title('Gassfraksjon CO2 ut')



% for i=1:length(Tl(1,:))
```

```
%       Al=co2_sw([Tl(1,i),Vl(1,i),Nl(1,i)]);
%       Av=co2_sw([Tv(1,i),Vv(1,i),Nv(1,i)]);
%       myl(i)=Al.g(3);
%       myv(i)=Av.g(3);
% end
% dp=pv-pl;
% dmy=myv-myl;
%
% subplot(2,2,1)
% plot(t,dp(1,:))
% title('Første delvolum, Dp')
% ylabel('[Pa]')
%
% subplot(2,2,2)
% plot(t,dp(end,:))
% title('Siste delvolum, Dp')
% ylabel('[Pa]')
%
% subplot(2,2,3)
% plot(t,dmy(1,:))
% title('Første delvolum, Dmy')
% ylabel('[J/mol]'), xlabel('[sek]')
%
% subplot(2,2,4)
% plot(t,dmy(end,:))
% title('Siste delvolum, Dmy')
% ylabel('[J/mol]'),  xlabel('[sek]')
```

H.5.2 pH_co2_sw.m

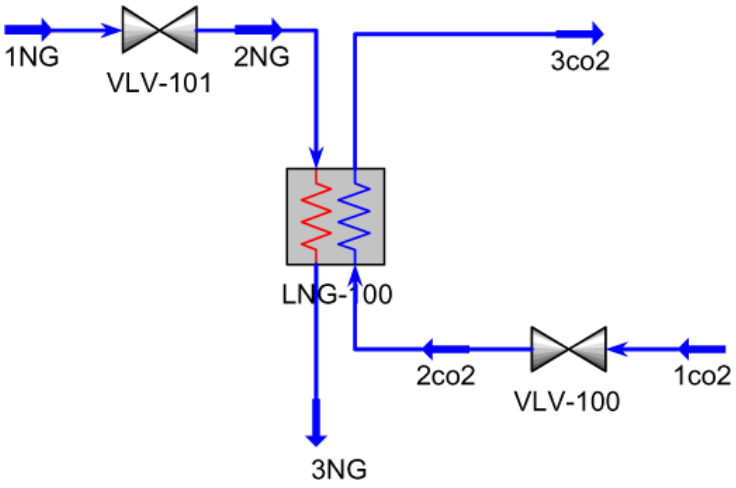
```
% Hp_co2.m
% Kalkulerer p som funksjon av H ved ulike T


format short e;
warning('off');                 % Hvis warning: vil ikke synes

i   = 2:3;                      % Volum og moltall (plassering i matr)
n   = 3;                        % Moltall
t   = 220:10:340;                       % Temperatur
v = logspace(log10(3.9e-5),log10(1e-2),100);     % Volumer
x0 = [NaN NaN 1]';              % Initiell tilstandsvektor
e1 = [1,zeros(size(n))];        % Beskrankningsmatrise (stabilitetssøk)
enthalpy   = [];                % Beregnede entalpier
pressure   = [];                % Beregnede trykk
maxit      = 20;                % Max antall iterasjoner(stabilitetssøk)

for T = t
    phs = 1;                      % Antar 1 fase
    h   = [];                     % Beregnede entalpier
    p   = [];                     % Beregnede trykk
    for V = v
        if phs==1
            x         = x0;                          % Default tilstandsvektor
            x(1:2)    = [T;V];                       % Legger inn T og V
            A         = co2_sw(x);                   % Beregner gradient og
                Hessian
            h(end+1) =-A.g(1)*T + A.g(n)'*x(n);      % Entalpi
            p(end+1) =-A.g(2);                       % Trykk
            H0    = A.H;                             % Trenger denne Hessiske
                hvis
```

```
                                        % en mer stabil fase
                                            finnes
g0    = A.g;                            % Tangentplan
y     = [x(1:2);x(n)/10000];            % Crude estimate for vap
    phase.
                                        % Fortynner massen slik av
                                            vi helt
                                        % sikkert ikke har
                                            væskefase lenger
dy    = y;                              % Initialize update vector
iter = 0;                               % Restrict number of
    iterations
                                        % (stability calculation
                                            may
                                        % oscillate for unstable
                                            phases)
while iter<maxit & norm(dy)>1e-6        % Do a stability check:
    Sjekker at
                                        % vi ikke har væske
    iter = iter + 1;
    A     = co2_sw(y);
    tmp   = inv([e1',A.H(i,i);0,e1])*[g0(i)-A.g(i);0]; %
        Løsningsvektor
    dy    = [0;tmp(i)];
    s     = min(1,-0.8/min([dy./y;-0.3*dy./y]));
    y     = y + s*dy;
end
if iter==maxit
    disp(['Stability test did not converge (T=',num2str(T),',V='
        ,num2str(V),')'])
end
if tmp(1)>1e-6*p(end)                   % Assume positive
    pressure
                                        % i.e. p(end)>0 in this
                                            test
    alpha   = min(0.1,tmp(1)*V/(y(i)'*H0(i,i)*y(i))); % Assume
        alpha>0
                                        % (maybe<0 for unstable
                                            phases)
    y(i)    = y(i)*alpha;               % Scale V and n in vapor
        phase
    x(i)    = x(i) - y(i);              % Apply V and n balances
        for
                                        % liquid phase
    dy      = y;                        % Initialize update
        vector
    while norm(dy)>1e-6                 % Entrer tofase-flash
        Vap = co2_sw(y);              % Forventer å finne to faser
        Liq = co2_sw(x);
        dy   = [0;inv(Liq.H(i,i)+Vap.H(i,i))*(Liq.g(i)-Vap.g(i))
            ];
        s    = min(1,-0.8/min([dy./y;-dy./x]));
        y    = y + s*dy;
        x    = x - s*dy;
    end
    if any(isnan([x,y]))                % Hvis vi har NaN --> noe
         er galt
        h(end) = NaN;
        p(end) = NaN;
        disp(['Phase is metastable but flash did not converge (T
            =',num2str(T),',V=',num2str(V),')'])
    else
```

```
                    phs     = 2;                    % Assume two phases from
                        now on
                    h(end) = -(Liq.g(1)+Vap.g(1))*T + Vap.g(n)'*(x(n)+y(n));
                                                    % Same chem.pot. in both
                                                        phases
                    p(end) = - Vap.g(2);            % Same pressure in both
                        phases
                    disp(['Vapor-liquid 2-phase (T=',num2str(T),',V=',
                        num2str(V),'),V/L=',num2str(sum(y(n))/sum(x(n)))])
                end
            end
        else
            y(1:2) = [T;V-x(2)];
            x(1:2) = [T;V-y(2)];
            dy     = y;
            while norm(dy)>1e-6 & norm(x(i))>1e-6
                Vap = co2_sw(y);
                Liq = co2_sw(x);
                dy  = [0;inv(Liq.H(i,i)+Vap.H(i,i))*(Liq.g(i)-Vap.g(i))];
                s   = min(1,-0.8/min([dy./y;-dy./x]));
                y   = y + s*dy;
                x   = x - s*dy;
            end
            h(end+1) = -(Liq.g(1)+Vap.g(1))*T + Vap.g(n)'*(x(n)+y(n));
                                                    % Same chem.pot. in both
                                                        phases
            p(end+1) = - Vap.g(2);                  % Same pressure in both
                phases
            if norm(x(i))<1e-6
                phs     = 1;                        % Assume one (vapor)
                    phase
                                                    % from now on
                x(i)    = x(i) + y(i);              % Apply V and n balances
                    to
                                                    % vapor phase
                h(end) =-Vap.g(1)*T + Vap.g(n)'*x(n);
                p(end) =-Vap.g(2);
                disp(['Single (vapor) phase (T=',num2str(T),',V=',num2str(V)
                    ,')'])
            end
        end
    end
    enthalpy(:,end+1) = h';
    pressure(:,end+1) = p';
end
semilogy(enthalpy,pressure,'k')
%axis([min(enthalpy), max(enthalpy)])
xlabel('Entalpi [J/mol]')
ylabel('Trykk [Pa]')
return
```

I Hysys-modell

En Hysys-modell tilsvarende Matlab-modellene er vist i Figur I.1. Strømdata er gitt på neste side.



Figur I.1: Hysysmodell

Workbook: Case (Main)

Material Streams

Name	1NG	3NG	3co2	1co2	2co2
Vapour Fraction	1.0000	1.0000	0.7592	0.0000 *	0.1654
Temperature (C)	8.000 *	-2.000 *	-10.06	7.477	-8.117
Pressure (kPa)	7100 *	6900 *	2600 *	4200 *	2750 *
Molar Flow (kgmole/h)	2.805e+004	2.805e+004	1751	1751	1751
Mass Flow (kg/h)	4.500e+005 *	4.500e+005	7.704e+004	7.704e+004 *	7.704e+004
Liquid Volume Flow (m3/h)	1503	1503	93.34	93.34	93.34
Heat Flow (kJ/h)	-2.158e+009	-2.170e+009	-6.994e+008	-7.113e+008	-7.113e+008

Name	2NG
Vapour Fraction	1.0000
Temperature (C)	7.248
Pressure (kPa)	6930 *
Molar Flow (kgmole/h)	2.805e+004
Mass Flow (kg/h)	4.500e+005
Liquid Volume Flow (m3/h)	1503
Heat Flow (kJ/h)	-2.158e+009

* Specified by user.