Daglogbog

# D. 30/10

Reverse Engineer Circadapt

* 1-Kammers eksperiment anvendt
* Ved add\_component() definition, finds constructor for modellen



* I constructor, finds add\_smart\_component()
  + Dens definition viser, at der findes 4 smart\_components

A screen shot of a computer code

Description automatically generated

* ArtVen undersøges, og bekræftes at være Systemisk/Pulmonær cirkulation

A screen shot of a computer code

Description automatically generated



Reverse Engineer 1-kammers eksperiment

* **add\_component() 1. parameter er basis-objekt, 2. parameter er navn**



* **set\_component() sætter objekterne i relation til hinanden**
  + e.g. hvad er proximalt og distalt til ventilen



* Hvordan relaterer komponenterne sig til hinanden?
  + Fra build\_artven ses, at default argumentet skaber systemisk cirkulation.
  + Der skabes altså en ventil, hvorpå der sidder en blodåre proximalt og distalt
  + Længden af hver blodåre sættes ved inde add\_smart\_component()
    - 
* På samme måde som blodårerne har en længde, har kammeret også en længde. Denne sættes gennem Patch2022 elementet:
  + Først hæftes patch2022 på væggen af kammeret:



* + Så tilføjes en længde til Patch2022 elementet:



* Kan jeg få adgang til differentialligningerne hvorfra beregningerne foretages?
  + A diagram of a mathematical equation

    Description automatically generatedFølgende framework benyttes i udregningerne
  + Det vigtige for mig er så at identificere hvor jeg skal indsætte mit modul
  + Solver modulet undersøges



* + - Solver modulet sætter parametrene for solveren, men giver ikke differentialligningerne

A computer screen shot of a program code

Description automatically generated

* + Run() funktionen undersøges
    - Run\_stable() funktionen undersøges
* Kan jeg tilføje eksternt pressure til modellen?
  + Min søgning stoppede ved .dll filen, som Python wrapperen potentielt set benytter til at lave beregningerne i C++
  + Derfor, overvejes det kraftigt, at skifte over til MatLab udgaven af programmet
  + På <https://github.com/Mheu1991/CircAdaptTL/tree/master/CircAdapt%20AUG2018> er fundet en tilpasset udgave af CircAdapt, der hører til studiet <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6677326/>
  + Forskellene mlm. Matlabkoden fra CircAdapt og den modificerede fra studiet er:

# D. 16/11

Identificere ODE beregningshierarkiet

* Checkede CircAdaptMainP for funktionskald der beregner løsning
  + Sker i CircAdaptP
* CircAdaptP er en abstraktion af løsningsgenerering, hvor der bl.a. også plottes
  + CircAdaptP kører ode113 på SVarDOT
  + Herefter kaldes SVar, som skalerer beregningerne til fysiologiske parametre
* Beregningshierarkiet ligger altså i SVarDOT:

A screenshot of a computer

Description automatically generated

A diagram of a mathematical equation

Description automatically generated

Identificere og manipulere pulmonær trykberegning

* Modificeret CircDisplayP for at sammenligne trykpåvirkning
  + Opdeler plots i volumen, flow og tryk (frem for samlet plot)
  + Tilføjer legends, så beregningerne er tydelige
* pNodeVDot laver node beregninger, for atrium, ventrikel samt pulmonær- og sytemisk cirkulation
* Tilføjet intrathoratisk tryk i pNodeVDot der hvor tryk beregnes (l. 51)
  + P(V) kurver påvirkes ekstremt, mens PPV er stort set identisk

Identificere, manipulere og teste CircAdapt’s transmurale trykberegninger

* BagV2p simulerer Pericardium
  + Her beregnes også et transmuralt tryk
  + Dog er det transmurale tryk baseret på Henry’s lov og volumeændring fra hjertets areal.
    - Altså ikke intrathoracisk tryk
* Det er forsøgt at tilføje en konstant til BagV2p’s transmurale tryk
  + Resultaterne er dog sammenlignelige med resultaterne af at ændre i pNodeVDot
  + Det formodes, at dette kan skyldes de anvendte enheder
    - Enheder i selve beregning er ukendt, og de transduceres til fysiologiske konstanter senere i processen
* Sammenlignelige (men ikke beregnede) resultater opnået ved ChamberV2p’s transmurale tryk
* Det vurderes, at næste skridt må være, at identificere og tilføje de korrekte enheder i modellen

# 17/11

Identificere og tilføje korrekte enheder i modellen

* Syart & PuArt har ekstremt høje trykværdier sammenlignet med andre nodes
* Rettelse: Gns. Er ekstremt højt, men max værdier varier ikke meget
  + Dette skyldes, at værdierne for SysArt og PuArt har en høj baseline der oppuster gnms., mens Lv og Rv periodisk går fra lav minimumsværdi til høj minimumsværdi, så gns. Er lavere
* Der tjekkes om CircDisplayP funktion henter signaler i rigtig rækkefølge
  + Det gør den – manuel sammenligning af kendt rækkefølge fra P og data udtrykket fra CircDisplayP bekræfter dette

# 21/11

Clone og implementer CircOpt

* Circopt findes på dette link
  + <https://github.com/idealab-isu/CircOpt>
* Cloned til sin egen folder
* CircAdapt lagt i de korrekte subfolders
* CircAdapt modificeret iflg. CircOpts krav

Identificier og implementer korrekt CircAdapt version

* Oprindelige CircAdapt: 2015 version
* Fundet og testet 2005 version, som refereres i CircOpt dokumentation
  + Ikke korrekt version
* Fundet og testet 2012 version
  + Korrekt version

Opdater Matlab Licens samt hent nødvendige toolboxes

* Aktiveret Matlab med AAU licens
* Afinstalleret Matlab 2022
* Installeret Matlab 2023
* Kørt installation 2 gange yderligere, for at installere nødvendige toolboxes
  + (Parallel Computing og Global Optimization toolboxes)

Test og tilpas CircOpt

* Par.ODEtime nødvendig i CircOpt
  + Findes kun i ParDog data, og ikke ParRef data
  + Kopieret fra ParDog til ParRef
* Derudover masse små justeringer, som ikke kræver dokumentation
* CircOpt fungerer ikke på trods af tilpasning
  + Løsning: Byg egen optimeringsframework

Byg Egen optimeringsløsning

Byg cyklisk intrathoracisk tryk tilføjelse

* Start med at implementere som sinuskurve
  + Forskellig længde for inspiration og ekspiration

# 22/11

Brainstorm om implementering af Ppl modul

* Hjerteslag beregnes for 0.85sek pr. default
* Respiratorisk cyclus er mlm. 4-5sek
  + ~1.5sek insp, 2.5sek exp
* Insp\_Time & Exp\_Time kan beregnes ud fra dataen, ved at finde tiden mellem max og min
* Hvordan synkroniseres hjertecyklus med respiratorisk cyklus?

**Forslag: Simular ODE for kort tidsinterval (f.eks. 10ms)**

* + Først Hjertecyklus + gem output
  + Så resp cyklus + gem output
  + Opdater hjerteoutput med Ppl
  + Gentag iterativt
* Problem med dette forslag er massiv overhead
* Hvordan optimeres dette forslag?

Brainstorm om implementering af optimeringframework

* Antagelse er, at vi optimerer på det 1. forslag til Ppl modul implementering

A diagram of a flowchart

Description automatically generated

Implementering af Ppl

* Efter gennemgang af CircOpt, finds det at circopt optimerer en række parametre, kører modellen, og herefter beregner fejlfunktion
* Det er ikke en god løsning til mit problem, da Ppl simpelthen skifter løbende
* Derfor, forsøges implementering af 50ms steps (kan varieres)
* UPDATE: Pga. modellens beregniner bliver Ppl nødt til at blive implementeret ”live”
  + Dette skyldes at beregningshastigheden simpelthen bliver for langsom ellers

# 23/11

Ppl modul implementering som optimering integreret direkte i CircAdapt

* Optimeringsframework implementering
  + Kald ønskede optimeringsalgoritme
  + Input er fejlfunktion
  + Fejlfunktion indeholder model samt fejlberegning

# 27/11

Dokumentation af CircAdapt

# 30/11

Krystalliser formål med model

* Ved slut expiration, er Ppl mlm. -3 til -5 cmH2O
  + Dvs. at PPV eller NPV påvirker Ppl ud fra denne baseline
* Skal Pmus (og Pvent) være input til modellen?
  + Pvent er en nødvendig uafhængig variabel
  + Pmus skal gerne udledes fra fysiologiske sammenhænge
* Sammenhæng forstået:
  + Formål med modellen er at studere et fænomen. Dvs., at der er input variabler, afledte variabler og estimerede variabler
  + Dvs. modellen ikke nødvendigvis er et redskab ”at the bedside”
* Breakthrough:
  + Modellen skal simulere PPV udsving for den individuelle patient
  + Vores PPV data, benyttes til at validere modellen (LS cost)
  + Konkret - Ved en specifik PS skal vi kunne justere Pmus, så det passer til data
  + Andre modeller skal bruge Peso målinger for at validere Pmus - Denne model skal bare bruge compliance
* Hvordan valideres korrekt Pmus?
  + Case Lav PS:
  + Negativt Pth får PPV til at falde
  + Start\_insp (max\_insp) og start\_exp (min\_exp) viser ændring i Pth på PPV
  + Her vil Pmus altså være ændringen i pulstrykket
  + Normalt ville det være et mudret signal, men siden vi kender compliance fra PS manøvren, kan vi differntire mlm. PS og Pmus
    - Dvs. at vi i dataen kender den korrekte Pmus, og derved kan vi se om modellen rammer den rigtige PPV ved den samme Pmus
  + Case høj PS:
  + Positivt tryk får Pth til at stige grundet positive ventilation (PV)
  + D3 (slut\_insp-slut\_exp) vil være positiv
  + D4 (start\_insp-start\_exp) vil være negativ
  + PPV vil her udelukkende være grundet PS - vi validerer ved at se om PPV svarer til patientdatas PPV

# 1/12

Breakthrough:

* Har spurgt mig selv hvordan patientens vejrtrækning inkorporeres
  + Svar: JEG simulerer vejrtrækningen! Dvs. at modellen simulerer vejrtrækning over en tidsramme, og de variabler der indgår beregnes derfra.

# 11/12

Debugging af Activation Function

* Activation Function fungerer i et test scenarie
* Den fungerer dog ikke i modellen
* Hypotese: Housekeeping er problemet