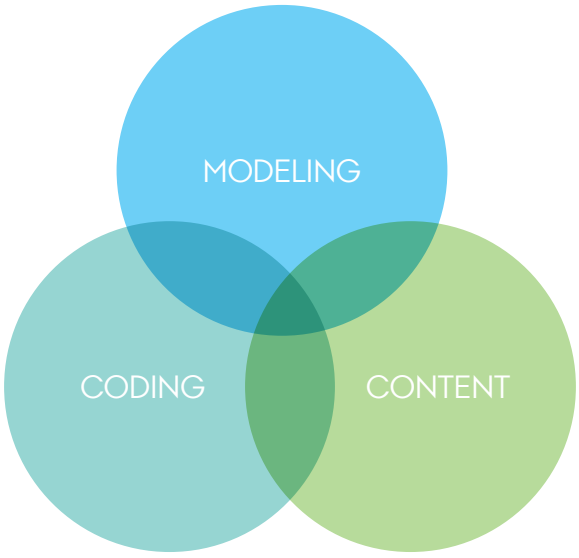
AARHUS
UNIVERSITET

COMPUTATIONAL THINKING
I GYMNASIEFAG

EN MODELBASERET DIDAKTIK

Line Have Musaeus
Jonas Ørbæk Hansen
Morten Damsgaard-Madsen
Keld Nielsen



KOLOFON

“Eleverne lærer faglige begreber, mens de arbejder med CT, og de lærer CT, mens de arbejder med faget.”

Computational Thinking i gymnasiefag. En modelbaseret didaktik. 1. udgave, 2020

Udgives af Center for Computational Thinking & Design, Aarhus Universitet

Forfattere: Line Have Musaeus
Jonas Ørbæk Hansen
Morten Damsgaard-Madsen
Keld Nielsen

Redaktion: Keld Nielsen

Layout: Hreinn Gudlaugsson

UDGIVES I SAMARBEJDE MED
Danske Science Gymnasier
Egaa Gymnasium

UDGIVELSEN ER STØTTET AF
VILLUM FONDEN
Region Midtjylland/Regional Udvikling

TAK TIL
Elisabeth Husum, Egaa Gymnasium, Jesper Buch, Herningsholm Erhvervsskole & Gymnasier og Lars Brian Krogh, VIA, for kritisk gennemlæsning af en tidlige version, værdifulde kommentarer og gode forslag.

Adam Etches, Egaa Gymnasium, for stor indsats som projektleder i skoleåret 2018/2019.

De mange lærere og elever, der har bidraget til de kurser, udviklingsprojekter, og afprøvninger, som ligger bag udgivelsen.

ISBN 978-87-94078-00-9
Gengivelse af tekst fra denne publikation er tilladt såfremt det sker med tydelig henvisning.

Publikationen kan hentes her: cs.au.dk/fileadmin/www.cct.au.dk/cctd.au.dk/PDFs/CT-didaktik2020.pdf

Her gives et bud på, hvordan **Computational Thinking** (CT) kan inddrages i gymnasiets eksisterende faglige undervisning, så eleverne lærer faglige begreber, mens de arbejder med CT, og de lærer CT, mens de arbejder med faget. CT som et selvstændigt fag, som det f.eks. optræder i it-fagene i gymnasiets fagrække, behandles ikke.

Der er tale om et første forsøg på samlet at formulere en række konkrete og afprøvede idéer for undervisning i gymnasiet inden for området Computational Thinking. Vi behandler kort spørgsmålet om, **HVORFOR** der skal undervises i emnet. Vi giver et noget længere svar på spørgsmålet om, **HVAD** der skal undervises i. Vi bruger dog det meste af pladsen på, **HVORDAN** vi mener, der kan undervises. I den forbindelse præsenterer vi en ny tilgang til undervisningen og en række nye didaktiske begreber.

I implementeringen af undervisning, der udvikler elevernes CT-kompetencer, lægger vi stor vægt på anvendelse af computermodeller og modellering. Dels fordi det giver anledning til at inddrage nye metoder og en ny it-faglighed i undervisningen, dels fordi det medvirker til at gøre fagene mere autentiske i en tid, hvor forskning i alle faglige discipliner inddrager computere og computermodeller. Vi foreslår således ikke bare en ny it-faglighed, vi udvider også fagenes eksisterende fagligheder.

OM FORFATTERNE

Line Have Musaeus
Ph.d.-studerende ved Center for Computational Thinking & Design. Projektleder på *CT i gymnasiefag* (se s. 8). Kandidat i biologi. Tidligere lærer på Silkeborg Gymnasium.

Jonas Ørbæk Hansen
Projektleder på *Computational Thinking i Matematik og Naturfag* (se s. 8). Kandidat og Ph.D. i nanoscience. Underviser i fysik og kemi på Silkeborg Gymnasium.

Morten Damsgaard-Madsen
Projektleder på *Modellering og Computational Thinking i Gymnasiet* (se s. 9). Kandidat i samfundsfag. Underviser i informatik, erhvervsøkonomi og samfundsfag på Egaa Gymnasium.

Keld Nielsen
Tidligere lektor og institutleder ved Aarhus Universitet. Kandidat i fysik og Ph.D. i videnskabshistorie. Arbejdede i en årrække ved Center for Science Education, Aarhus Universitet.

FORORD

Dette hæfte henvender sig primært til lærere i gymnasiet, som ønsker at give deres elever opdaterede it-kompetencer gennem at udvikle undervisning, der indeholder centrale elementer af Computational Thinking.

Det kan for eksempel være lærere, som er interesserede i at udvikle deres faglige undervisning ved at inddrage computermodeller i undervisningen, så eleverne får et realistisk indblik i, hvordan computere og modeller anvendes i moderne forskning knyttet til faget. Lærere, som synes, at undervisning i faglige metoder trænger til et løft.

Eller lærere, som underviser i it-fagene og gerne vil knytte deres undervisning tættere til et eller flere andre fag, eller er på udkig efter nye, motiverende måder at introducere programmering på, eller generelt er på jagt efter nye didaktiske idéer.

Eller lærere, som gerne vil arbejde i dybden med modeller og udforske, hvad modeller betyder for elevernes faglige fordybelse. Lærere, som ser en mulighed for at styrke elevers fagligt kreative samtale, tænkning og problemløsning gennem inddragelse af modeller.

Vi tænker også, at hæftet kan læses i forbindelse med kurser, hvor lærere efteruddannes i informatik eller Computational Thinking.

Endelig håber vi, at hæftet vil være interessant for en større gruppe, der i anden sammenhæng interesserer sig for fornyelse af undervisningen i gymnasiet.

Vi har forsøgt at forme teksten, så den kan læses uden forhåndskendskab til computerkode, programmering eller Computational Thinking.

God fornøjelse!
Aarhus, maj 2020,
Forfatterne

→ Se tre minutters introduktion til, hvorfor og hvordan vi foreslår at inddrage CT i undervisningen: **https://www.youtube.com/watch?v=9R9r5w2Gr-w&t=2s**

→ Hvis du har kommentarer, spørgsmål eller forslag i forbindelse med dette bud på en didaktik, så skriv til **CCTD@cc.au.dk**

INDHOLD

1. Indledning 6
2. Baggrund: Idéer og erfaringer 9
3. Computational Thinking, kodning og modeller i fag 11
4. Hvad forstår vi ved Computational Thinking i gymnasieundervisning? 14
5. CMC-tilgangen 18
   Eksempler: Brug af computermodeller i forskning 21
6. Modeller i undervisning 25
   6.1 Mentale modeller 27
   6.2 Computermodeller 27
7. Agent-baseret modellering (ABM) 31
   7.1 NetLogo 32
   7.2 Sammenhæng mellem modellers mikroniveau og makroniveau 33
8. Progression gennem Tinkering og Use-Modify-Create (UMC) 36
9. Modeller og konstruktivisme 38
   9.1 Fagligt enkle modeller 38
10. CMC i praksis – eksempel på et undervisningsforløb 43
11. Læringsmål i forbindelse med CT-i-fag 47
12. Hvordan kommer jeg i gang med NetLogo? 49
   12.1 God kode i NetLogo 49
13. Referencer 53

1. INDLEDNING

Så godt som alle udviklede lande arbejder på at få computere og digitalisering ind i som emner i deres uddannelsessystemer. Bredt og på nye måder, fra de mindste klasser til universitetsniveau.

I Danmark anbefalede Vækstrådet i 2016, at *Computational Thinking (CT)* gøres til en obligatorisk del af uddannelser på alle niveauer. I gymnasiereformen samme år blev *informatik* indført som nyt fag, og styrkelse af elevernes digitale dannelse og digitale kompetencer blev et fokusområde på tværs af alle fag. I grundskolen afprøves nu det it-baserede fag *teknologiforståelse*, som må forventes at blive obligatorisk efter en treårig forsøgsperiode.

Baggrunden er de store samfundsændringer, som er knyttet til computere og digitalisering. Arbejdsliv, underholdning, sociale relationer, indkøb, adgang til nyheder og information, innovation, forskning, politisk beslutningstagning, sundhed og behandling – ja, hele hverdagen som borger – er vokset sammen med brug af computersystemer, automatisering, data og software.

Verden bliver i stigende grad digitaliseret. Digitale teknologier er det centrale element i en radikal og transformerende ændring af professioner, videnskabelige discipliner og vores sociale liv i almindelighed. Det er vigtigt at forstå de nye muligheder, som informatikken giver, allerede fra en tidlig alder. I konsekvens heraf bør alle elever, studerende og lærere ikke blot have digitale brugerkompetencer, men have viden, færdigheder og kompetencer inden for informatik og 'Computational Thinking' (CT) (Caspersen et al., 2018). CT-kompetencer er nødvendige på lige fod med sproglige kompetencer og almene matematiske kompetencer. Dermed er de en grundbetingelse for at kunne begå sig i det 21. århundrede.

Det er ikke tilstrækkeligt bare at være 'bruger'. Bæredygtig anvendelse af computere kræver, at man ved noget om computeres funktion, potentialer og begrænsninger – ikke mindst hvordan software udvikles og kan bruges på nye måder og på nye områder. Så man som borger kan forholde sig kritisk, vurderende og kreativt til de nye muligheder.

Det beskrives og diskuteres derfor i disse år livligt, hvordan og på hvilke områder det danske uddannelsessystem kan imødegå de enorme udfordringer, der er knyttet til denne udvikling - fra 0'te klasse til 3.g. (Se f.eks. Bundsgaard et al., 2019; Caspersen, 2917; Caspersen et al., 2018; Danske Gymnasier, 2019; Iversen et al., 2019).

Diskussionen er væsentlig og nødvendig som grundlag for nuværende og kommende initiativer med relation til undervisning på it-området. Med dette bud på en didaktik forholder vi os selvfølgelig til denne diskussion. Vi bidrager med et konkret og ret håndfast bud på, hvordan nogle af de krav og udfordringer, der diskuteres, kan imødekommes. Vi forestiller os ikke, at vort forslag er relevant ved løsningen af samtlige udfordringer knyttet til den digitale udvikling. Men nok ved løsningen af nogle af dem. Styrken, som vi ser det, er netop, at forslaget er konkret og fokuseret.

Vi håber, at det af det følgende vil fremgå, at de elevkompetencer, der opøves gennem den beskrevne undervisning, tilgodeser både et almendannende perspektiv og et mere snævert professionelt perspektiv.

Det almendannende, fordi eleverne får en helt essentiel baggrund for at forstå, hvordan og hvorfor modellering og brug af computermodeller dominerer megen tænkning og beslutningstagen. Altså får indsigt i, hvorfor computermodellering er så stærkt et redskab i forbindelse med forskning og

problemløsning. Men også hvilke faldgruber, det indebærer: At modelresultater skal fortolkes kritisk og med varsomhed, at en model må være tilpasset opgaven, at nogle modeller i en konkret sammenhæng er bedre end andre, og at det er muligt at lære at udtrykke sig kritisk-konstruktivt over for modeller og modelleringer. Lidt forenklet sagt, er det ikke muligt i et moderne samfund, hvor vi er omgivet af computermodeller, at være almendannet, med mindre man har et vist minimum af forståelse af den omfattende rolle modeller og modellering spiller i politik, økonomi, produktion, underholdning, forskning og udveksling af information.

Det professionelle, fordi eleverne får udbygget deres faglige indsigt – herunder indsigt i fagets metoder – samtidig med at de opøver nogle essentielle generiske kompetencer, knyttet til brug af computere og computerkode.

I det europæiske initiativ *Informatics for All* fremhæves behovet for en tostrenget strategi for CT på alle uddannelsesniveauer (Caspersen et al., 2018). En tostrenget strategi betyder, at der på den ene side skal være fokus på CT som del af en fundamental og selvstændig it-disciplin (CT-som-fag). Den tilgang har i Danmark indtil videre resulteret i faget *informatik* i gymnasiet, samt i det nye fag i grundskolen (digital) *teknologiforståelse*, der bl.a. testes som et selvstændigt fag. På den anden side skal der arbejdes med CT som en tværgående disciplin, der er integreret i andre fag (CT-i-fag). Som det for eksempel sker på andre af de grundskoler, der deltager i afprøvningen af (digital) *teknologiforståelse*. Det er CT-i-fag, som er i fokus i denne didaktik.

Behovet for at inddrage CT i undervisningen er ikke opstået udelukkende på grund af de nye teknologisk betingende udfordringer, man i fremtiden vil møde som borger og på arbejdsmarkedet. Lige så vigtigt er det, at computerbaserede forskningsmetoder nu spiller en voksende – på nogle områder ligefrem dominerende – rolle i så godt som alle forskningsdiscipliner. For fagene i gymnasiets fagrække gælder, at metoder til skabelse af den viden, som undervisningen bygger på, i de sidste årtier har undergået epokegørende forandringer.

I alle fag i gymnasiet indgår læringsmål knyttet til fagenes metoder: Hvordan skabes der ny faglig viden, og på hvilket grundlag kan man argumentere for, at den faglige viden er valid? Den computerbaserede, revolutionerende udvikling af forskningsmetoder skal selvfølgelig afspejles i den måde, der undervises i fagene. Vores opfattelse er, at gymnasiets fag og undervisning er i fare for at komme bagefter på dette område. Også det skal der rådes bod på.

Der er – som antydet ovenfor – ikke tale om, at vi giver et bredt, teoretisk funderet bud på, hvordan undervisning i informatik/CT kan gribes an. Tværtimod giver vi et ret snævert, men fokuseret, konkret og funktionelt bud på, hvordan man kan undervise i CT i forbindelse med den eksisterende faglige undervisning. Vi har valgt denne tilgang, fordi generelle formuleringer af, hvad man bør forstå ved CT er vanskelige at didaktisere, altså at omsætte til specifikke bud på, hvad man kan undervise i, og hvordan man kan undervise. Desuden plages CT-diskussionen af, at der endnu ikke er nogen tydelig konsensus om, hvad man i undervisningssammenhæng skal lægge i begrebet CT, ud over at det er noget med computere og data (Se f.eks. Caspersen et al., 2018).

Løsningen for os har derfor været at anvende en beskrivelse af CT, som er ret konkret. I stedet for at forsøge at beskrive feltet ud fra den akademiske disciplin datalogi ('computer science'), tager den udgangspunkt i, hvad mennesker, der anvender computere i deres arbejde, rent faktisk gør med dem. I forbindelse med undervisning på gymnasieniveau anser vi det således for frugtbart at anskue CT

som et sæt praksisser, der kan identificeres på baggrund af det, f.eks. moderne forskere rent faktisk gør, når de anvender computere (se afsnit 4).

Desuden har vi – igen for at reducere kompleksitet og skabe fokus – valgt at koncentrere os om det agentbaserede programmeringsmiljø NetLogo. Ikke fordi vi tror, at brug af NetLogo kan løse alle CT-problemer, eller at NetLogo i et længere perspektiv kan stå alene. Men i forbindelse med nogle udviklingsprojekter har vi set, at NetLogo har en række pædagogiske kvaliteter (se afsnit 7). Samtidig mener vi, at det vil være en stor fordel, hvis udviklingen af CT-undervisning på de enkelte gymnasier, i denne diffuse opstartsperiode, kan finde en fælles platform og et fælles sprog, så lærere kan arbejde sammen og støtte hinanden på tværs af fag og tidligere erfaringer med undervisning i it-fagene. Senere, når den enkelte skole har fundet sit fokus og sin vej frem i udviklingen af CT, kan brugen af NetLogo kombineres eller suppleres med andre sprog (Scratch, Python, ...), eller andre tilgange (robotter, Arduino, Micro:Bit, ...).

De tanker om CT og undervisning, der fremlægges her, bygger delvist på en række resultater fra udlandet. Især det omtalte programmeringsmiljø NetLogo, som er udviklet ved Center for Connected Learning ved Northwestern University i Chicago. Centeret ledes af Uri Wilensky, som er elev af Seymour Papert, pioneren, der udviklede sproget Logo og samarbejdede med Lego om udviklingen af Lego Mindstorms.

Men en stor del af det følgende præsenterer idéer, som er udviklet af medarbejdere ved Center for Computational Thinking & Design ved Aarhus Universitet. Som det fremgår af afsnit 2, er idéerne dels udviklet på et teoretisk grundlag, dels – i praksis – gennem tre større udviklingsprojekter, hvoraf de to ikke er afsluttet endnu.

Nogle af de grundlæggende idéer i dette bud på en didaktik er således udviklet i samarbejde med danske gymnasielærere med henblik på at indføre CT i den faglige undervisning i gymnasiet. De udviklede metoder er originale, ikke blot i dansk sammenhæng. Andre CT-didaktikere har arbejdet med lignende idéer, men ikke under inddragelse af lærerne i skabelse af undervisningsforløb, og ikke med så stor vægt på brug af computationelle modeller og koden bag dem. Så også set med internationale øjne er der tale om en ny måde at tænke CT-undervisning og udvikling af CT-undervisning på. CMC-tilgangen har vakt international interesse (Musaeus & Musaeus, 2019).

I forbindelse med projekterne er mange idéer diskuteret, implementeret og afprøvet af de deltagende kursister. Vi vil gerne takke de kursister og de skoler, der har bidraget. Også en tak til Villum Fonden og Region Midtjylland, som støtter projekterne økonomisk, og til vore kolleger på CCTD for samarbejde og værdifulde diskussioner.

Dette bud på en didaktik er en første formulering af de indgående idéer. Der er pt. kurser i gang, hvor der arbejdes videre med idéer og didaktik. Når du læser, beder vi dig være opmærksom på, at der er tale om en didaktik i udvikling.

2. BAGGRUND: IDÉER OG ERFARINGER

Dette hæfte bygger på idéer og erfaringer, der er udviklet og opsamlet gennem en længere årrække.

De første danske idéer til, hvordan man kan undervise i CT i forbindelse med faglig undervisning, blev beskrevet i 2013 (Caspersen & Nowack, 2013) og uddybet året efter (Nowack & Caspersen, 2014). Forfatterne beskrev to mulige strategier for CT-undervisning: CT-som-fag og CT-i-fag. Muligheden for at integrere CT i fag ved hjælp af modelbetragtninger og brug af agentbaseret programmering blev foreslået og diskuteret. Idégrundlaget og forholdet til den internationale akademiske debat om it-undervisning uddybes i afsnit 4.

Idéerne er efterfølgende afprøvet i praksis og videreudviklet på baggrund af erfaringer fra en række udviklingsprojekter. Idéerne og erfaringerne er undervejs blevet præsenteret for og diskuteret med danske og internationale it-didaktikere. Der er således tale om et udviklingsarbejde, som også internationalt er banebrydende.

Afprøvning i praksis og erfaringsopsamling har især været knyttet til følgende projekter:

CT I GYMNASIEFAG var et pilotprojekt, der blev gennemført i 2017 med deltagelse af ni midtjyske gymnasier samt Center for Computational Thinking & Design ved Aarhus Universitet. I projektet blev for første gang afprøvet den tilgang til undervisning af CT-i-fag, der her omtales som CMC-tilgangen. Den beskrives i afsnit 5. Pilotprojektet leverede proof-of-concept for de grundlæggende didaktiske idéer (CCTD, 2018a; CCTD, 2018b).

I projektet blev der udviklet seks computermodeller med tilhørende undervisningsforløb: Tre i samfundsvidenskab, to i biologi/biokemi og et i kemi. Forløbene blev afprøvet og evalueret af 15 lærere og omfattede 210 elever. Evalueringen viste, at det didaktiske design fungerede godt for både lærere og elever. Elevene fik øget forståelse af sammenhængen mellem kodning, modellering og faglige fænomener, samtidig med at de fik udbygget deres faglige indsigt. Lærerne følte sig trygge ved at arbejde med programmeringsmiljøet og fik mulighed for at arbejde med kendt stof på en ny måde samt med nye idéer til modeller til brug i undervisningen. De udviklede undervisningsforløb er tilgængelige på **https://library.ct-denmark.org**.

Pilotprojektet blev støttet af Region Midtjylland og gennemført i samarbejde med it-vest. Du kan læse en kort beskrivelse af projektet her:
https://cctd.au.dk/projects/computational-thinking-in-high-school-subjects.
Der er udgivet en evalueringsrapport (CCTD, 2018b). Projektet er beskrevet, og resultaterne er publiceret internationalt (Musaeus & Musaeus, 2019).

COMPUTATIONAL THINKING I MATEMATIK OG NATURFAG (CTiMNAT) er et efteruddannelsesprojekt, som fortsætter der, hvor pilotprojektet stoppede. CTiMNAT strækker sig fra august 2018 til juni 2022. I projektet gentages tre et-årige kursusforløb, hvor nye forløb forbedres og udbygges på grundlag af en løbende opsamling af resultater og erfaringer.

Projektet gennemføres i samarbejde med Danske Science Gymnasier og er støttet af VILLUM FONDEN. Du kan læse om projektet her:
https://cctd.au.dk/projects/ctimnat-computational-thinking-in-math-and-science.

Det første af de tre kursusforløb blev gennemført i skoleåret 2018/2019. Der deltog 29 kursister fra 17 gymnasier. Der blev udviklet 29 computermodeller med tilhørende undervisningsforløb i fagene matematik, fysik, kemi og biologi/bioteknologi. Ca. 970 elever deltog i afprøvningen. Vurderet ud fra kursisternes evalueringer samt de producerede computermodeller og elevaktiviteter var kurset succesfuldt, idet alle kursister var i stand til at producere og afprøve computermodeller med tilhørende elevaktiviteter. I det efterfølgende skoleår anvendte mere end halvdelen af kursisterne computational thinking i deres undervisning. Der er udgivet en evalueringsrapport (Buch et al., 2019). De udviklede undervisningsforløb er tilgængelige på: **https://library.ct-denmark.org.**

Det andet af de tre kursusforløb gennemføres i skoleåret 2019/2020. Der deltager 40 kursister fra fagene matematik, fysik, kemi, geografi og biologi/bioteknologi. Det forventes, at i alt ca. 1000 elever vil deltage i afprøvning af de udviklede undervisningsforløb. (På grund af skolelukningerne i foråret 2020 er projektet forlænget til november 2020).

Det forventes, at det tredje kursusforløb i projektet CTiMNAT vil blive gennemført i skoleåret 2021/2022.

MODELLERING OG COMPUTATIONAL THINKING I GYMNASIEFAG (MCTiG) bygger også på idéerne fra pilotprojektet og udnytter de erfaringer, der blev høstet gennem det første år i CTiMNAT. Der arbejdes med modeller i samfundsfag, dansk, historie, mediefag, international økonomi og musik. Projektet støttes af Region Midtjylland og gennemføres fra december 2018 til november 2020. Du kan læse om projektet her:
https://cctd.au.dk/projects/mctig-computational-thinking-in-humanities-arts-and-social-sciences.
Der deltager 20 lærere fra 11 gymnasier. Det forventes, at mere end 400 elever vil deltage i afprøvning af de udviklede modeller med tilhørende undervisningsforløb. (På grund af skolelukningerne i foråret 2020 er projektet forlænget til november 2020).

Således er der p.t. gang i to lange kursusforløb, som først afsluttes i efteråret 2020. Når de to forløb er afsluttet og evalueret, vil denne version af didaktikken blive opdateret på grundlag af de indhøstede erfaringer. Didaktikken vil blandt andet blive udvidet med flere erfaringer fra samfundsfag og de humanistiske fag. Vi forventer, at en opdateret udgave kan være færdig i januar 2021.

Om elevernes arbejde med kode i den undervisning, der ligger bag didaktikken:

"Da programmering er helt nyt for de fleste elever – og mange af eleverne desuden udtrykker en forventning om, at det er meget svært at programmere – er programmeringsaktiviteterne planlagt med henblik på at give eleverne nogle hurtige mestringsoplevelser."
(Hansen, 2020, s. 25)

3. COMPUTATIONAL THINKING, KODNING OG MODELLER I FAG

Modeller og modellering er centrale for den tilgang til CT-kompetencer, den tilhørende didaktik og for den forståelse af læring, der anvendes i dette hæfte. Begrebet CT uddybes i afsnit 4. Modeller og den tilknyttede forståelse af læring uddybes i afsnit 5 og 6.

Modeller og modellering er ikke kun centrale begreber i naturvidenskab og matematik, men i en lang række fag, herunder musik-, samfunds- og sprogfag. Det gælder både i den forskning og videnskabelse, der finder sted i fagene, og i faglige undervisningssammenhænge. Flere internationale forskere har som konsekvens heraf påpeget, at områder som modellering og algoritmetænkning er centrale i de CT-kompetencer, der med fordel kan arbejdes med i en uddannelsesmæssig sammenhæng (Guzdial, 2019; Wilensky & Rand, 2015).

En lang række af kompetencer knyttet til modellering og algoritmer kan til en vis grad opøves gennem implementering af CT-læringsaktiviteter i de allerede eksisterende fag i uddannelsessystemet. Men arbejdet med at øve CT-relaterede kompetencer afhænger kritisk af elevernes generelle holdning til computere og til programmering.

I vores udviklingsprojekter har vi mødt lærere, der fortæller, at de har elever, som udtrykker aversion mod at arbejde med computerkode. Der er altså elever – vi ved ikke hvor mange – som helt enkelt ikke bryder sig om at programmere.

Derfor er det vigtigt, at eleverne bringes i situationer, hvor de oplever kodning som ufarlig og som et redskab, de kan få kontrol over. Undervisningen skal give eleverne mod på at turde lege og eksperimentere med kode og med algoritmer. Samtidig er det vigtigt for elevernes motivation, at de oplever arbejdet med koden og algoritmerne i computermodellen som en faglig relevant aktivitet - som et redskab til at forstå og udforske faget.

Det er mål, som blandt andet kan opnås ved at gøre eleverne til medskabere af computermodellerne, ikke blot brugere af dem. At være medskaber betyder ikke nødvendigvis, at eleverne skaber nye modeller fra grunden af. Eleverne er medskabere, når de finder ud af, at computermodeller er noget, de selv kan ændre og forbedre – og gør det i praksis. I afsnit 8 om progression i CT-undervisningen behandler vi, hvordan eleverne kan arbejde med at ændre og forbedre faglige modeller. Det sker gennem princippet om at arbejde 'udefra og ind' og forholde sig til modellen og fænomenet, før elementer og sammenhænge i modellen defineres (Nowack & Caspersen, 2014). Dette foregår gennem en fase af udforskning og '*tinkering*' i en bottom-up tilgang (Resnick & Rosenbaum, 2013). Progressionen i læringsaktiviteterne sikres gennem princippet om '*use-modify-create*' (Lee et al., 2011). *Modify*-fasen kan varieres og uddybes, og den indeholder den type arbejde med kode, og dermed model, som fører til en følelse af ejerskab – og dermed en stærk motivation – hos eleverne.

At foretage meningsfulde ændringer i koden bag en computermodel kræver faglig indsigt. I nogle aktiviteter vil eleverne være blevet klædt fagligt på inden aktiviteten. I andre aktiviteter lægges der op til, at den nødvendige faglige viden udvikles undervejs i aktiviteten – i arbejdet med computermodellen (f.eks. gennem en induktiv tilgang). I alle tilfælde er det målet, at arbejdet med koden bag computermodellen skal bidrage til at øge elevernes forståelse af det modellerede fænomen. Det kan f.eks. være i kraft af en dybere forståelse for de kausale mekanismer i modellen eller en mere deltaljeret forståelse af koblingen mellem mikroniveau og makroniveau.

Når elever diskuterer modellens opførsel og egenskaber med læreren, eller – mindst lige så vigtigt – med klassekammerater, tvinger det eleverne til at tænke detaljeret og meget konkret over det faglige fænomen, som er modelleret. Eksempelvis hvilke faglige elementer, modellen indeholder, og hvordan de påvirker hinanden og helheden. Når eleverne arbejder med at forstå de enkelte dele af koden, og hvordan de styrer modellens opførsel, styrkes deres forståelse af fænomenet og deres forståelse for, hvorfor de forskellige dele deri ser ud og opfører sig, som de gør. Koderepræsentation af fænomenet – den algoritmiske opbygning af koden med trinvise instrukser til de forskellige elementer i modellen – kan således hjælpe eleverne til at tænke over fænomenet på en ny måde.

Når eleverne bliver stillet overfor en opgave om at ændre eller forbedre en model, vil det naturligt lede til refleksioner og en vurdering af modellens styrker og svagheder. Det kan være på forskellige niveauer. Fra det simple *'det er ikke hensigtsmæssigt at reaktanter og produkter ser ens ud'* eller *'det er dumt, at man ikke kan lave om på halveringstiden'* til det avancerede *'hvorfor tager vores økonomimodel ikke højde for inflation?'* eller *'molekylernes hastighed er ikke medtaget i modellen, selvom vi ved, at det har betydning for reaktionshastigheden'*.

Et krav om at vurdere modellen fører altså til faglig tænkning og anvendelse af begreber, og dermed til faglige abstraktioner og i sidste ende tilbage til modellen og dens algoritmer. For eksempel ved at eleven selv finder en algoritme i koden, eller fordi eleven selv kopierer eller reformulerer en algoritme i en form, der indgår som del af et stykke modelkode (se også afsnit 8).

Når eleven bliver bedt om at vurdere styrker og begrænsninger ved en foreliggende faglig computermodel, er eleven nødt til at forholde sig konkret og kritisk-konstruktivt til det, modellen viser på skærmen. Her vil eleven måske i første omgang sammenligne computermodellen med en eller flere tegnede modeller i bogen, en faglig tekst, hverdagserfaringer eller resultatet fra et eksperiment. Men da eleven er anbragt i en situation, hvor han/hun er nødt til at forholde sig direkte og på egen hånd til det, der vises på skærmen, er eleven nødt til at bringe sin egen forståelse (eller mentale model) af det faglige fænomen i spil.

Eleven kan ikke foreslå ændringer i modellen og dermed forholde sig kreativt til fænomenet uden at indlede en indre dialog, hvor eleven selv sammenligner computermodellens opførsel med sin egen mentale model. Eleven er altså nødt til at forholde sig eksplicit til sin egen forståelse af det modellerede fænomen. I sideløbende processer arbejder eleven med at justere/forbedre sin mentale model, eller computermodellen, til der opnås en eller anden form for sammenfald. I en konkret undervisningssituation vil eleverne arbejde to og to eller i grupper eller i situationer, hvor det er god opførsel at kontakte kammeraten ved siden af for at indlede en faglig samtale om det, der foregår på skærmen. En samtale som tager udgangspunkt i elevens arbejde med at opnå overensstemmelse mellem sin mentale model og computermodellen.

For læreren – og for den fagligt reflekterende elev – bliver elevens sproglige formuleringer i arbejdet med at forstå modellen, og på det grundlag at foreslå ændringer i modellen, et vindue til elevens mentale model af fænomenet, og dermed til elevens faglige forståelse.

En styrke ved at inddrage computere i arbejdet med modellering er, at der er meget hurtig respons på elevernes løsning af opgaven og deres faglige forståelse af det modellerede fænomen. Når eleverne har lavet ændringer i koden og starter simuleringen, kan de hurtigt forholde sig til, om ændringerne førte til de forventede ændringer i modellens opførsel. Er resultaterne fornuftige? Hvis ikke, skyldes det

så en fejl i måden, de har ændret koden eller fejl i den faglige forståelse af fænomenet? Eleverne kan spejle deres mentale modeller med computermodellen og lynhurtigt få en ide om, om der er overensstemmelse.

Det er klart, at det kan give frustration, hvis vejen hen mod, at computermodellen opfører sig som forventet, er alt for lang. Men i det mindste er eleven blevet bevidst om, at der enten er en mangel i forståelse af det faglige fænomen eller i forståelsen af, hvordan modelkoden er bygget op. Og i de positive tilfælde, som der heldigvis er flest af, bliver eleverne hurtigt bekræftet i, at deres forståelse af fænomenet er korrekt, hvilket også er et vigtigt element for motivationen i undervisningen og for at arbejde med at ændre koden. Denne måde at inddrage computermodeller, som eleverne selv kan tilrette, i den faglige undervisning, stimulerer mange elevers nysgerrighed og lyst til at udforske det faglige fænomen i stor detalje. Desuden gør det det muligt for eleven at arbejde med og undersøge/erkende aspekter ved fænomenet, som ellers ikke er umiddelbart tilgængeligt (f.eks. pga. svær matematik).

Denne måde at arbejde med modellering i undervisningen gør det muligt for eleverne at arbejde aktivt, konstruktivt og kreativt med modellering, så de får en tydeligere opfattelse af, hvorfor modeller er så vigtige i forbindelse med at forstå og beskrive faglige fænomener.

4. HVAD FORSTÅR VI VED COMPUTATIONAL THINKING I GYMNASIEUNDERVISNING?

I denne didaktik arbejder vi med essensen af CT, som er at formulere problemer på en sådan måde, at de kan modelleres – og dermed principielt set løses – computationelt. Derved holder vi fokus på, at computere er redskaber til problemløsning og til at opnå ny erkendelse, og at det kræver særlige teknikker og kompetencer – som eleverne skal lære – at udnytte de kolossale muligheder, som brugen af computere giver os. Samtidig lærer eleverne at forstå og bearbejde de modeller, og dermed store udfordringer, brugen af computere stiller os overfor.

Begrebet Computational Thinking (CT) blev introduceret i 1980 af Seymour Papert i forbindelse med særlige kompetencer knyttet til den digitale udvikling (Papert, 1980). Begrebet blev senere reintroduceret af Jeannette Wing (Wing, 2006). De havde forskellige visioner for CT: Papert mente, at kernen i CT er at udvikle elevernes evne til at lære og skabe digitalt, mens Wing fokuserer på, at eleverne skal lære basale datalogiske kompetencer såsom at programmere.

De to visioner udelukker dog ikke hinanden. Dele af dem begge er inkluderet i de fleste nyere beskrivelser af CT, der især lægger vægt på, at problemløsning ved hjælp af computere kræver, at eleverne lærer:

- Algoritmisk tænkning
- Dekomposition af problemer
- Generalisering i form af mønstergenkendelse
- Abstraktion, refleksion og evaluering
- Kodning/programmering
- Design, analyse og anvendelse.

Nogle af disse kompetencer er tæt knyttet til det akademiske fag datalogi. Andre forskere har gjort opmærksom på, at CT bør ses i et bredere perspektiv, da det både kræver og giver kompetencer i forbindelse med problemanalyse og problemløsning, som rækker ud over datalogi som fag (diSessa, 2018; Guzdial et al., 2019).

Men problemet ved mange af de eksisterende definitioner af CT er, at de nok kan være nyttige i et overordnet, læringsfilosofisk perspektiv, men set fra et undervisningssynspunkt er de for ukonkrete og abstrakte i deres forsøg på at være så brede som muligt (Denning, 2017; Tedre & Denning, 2016).

Det har derfor været et aktivt valg i denne didaktik at anvende en beskrivelse af CT, der er så konkret som muligt. Udgangspunktet har været modellering som en central og samlende kompetence, der optræder både i CT, i mange fag og i datalogi (Lee et al., 2020; Grover & Pea, 2013; Aho, 2012; Denning, 2017).

Modellering og 'model-based thinking' har været præsenteret af flere fremtrædende forskere og styrker elevers forståelse af ikke bare modeller i et givet fag, men også de sammenhænge og de strategier i problemløsning, der eksisterer i relation til et fænomen i faget (Sun et al., 2006).

Flere forskere har arbejdet med at beskrive og afprøve modellering og 'model-based thinking' i en uddannelsesmæssig sammenhæng. Senest er modellering beskrevet af ledende computing education forskere, bl.a. Alan Kay og Marc Guzdial, som et oplagt udgangspunkt for at introducere CT i uddannelsesmæssige sammenhænge (Guzdial et al., 2019).

Specielt har danske forskere opnået international interesse ved at pege på, hvordan modellering baseret på computere kan implementeres i eksisterende fag i ungdomsuddannelserne (Caspersen & Nowack, 2013; Nowack & Caspersen, 2014; Caspersen et al., 2018). Modellering er således en konkret praksis, som kan anvendes i mange fag, også på gymnasieniveau.

Til at udfolde begrebet bruger vi en taksonomi af CT, som er beskrevet af Weintrop et al. (2016). Taksonomien tager udgangspunkt i anvendelsen af computere til erkendelse af ny viden inden for naturvidenskabelige fag, samt i en kortlægning af, hvad indholdet var i den eksisterende CT-undervisning ved en række amerikanske skoler og universiteter. Ved at engagere eleverne som innovative fagpersoner åbnes potentialet for dybere læring i faget (Lee et al., 2020). Herved repræsenteres den autentiske udgave af CT og modellering, hvilket er helt essentielt for, hvordan vi præsenterer fag i nutidens almendannende gymnasieskole.

Weintrop et al.'s overordnede kategorier inkluderer: Brug af data, brug af simuleringer og modeller, problemløsning og tænkning i systemer (se fig. 1).

Den tilgang til undervisning i CT, vi har valgt, dækker ikke alle niveauer i alle kategorier. I det følgende gør vi rede for, hvilke praksisser vi mener der er centrale, og hvilke der i øvrigt kan inddrages i den beskrevne CT-undervisning.

Data Practices
Collecting Data
Creating Data
Manipulating Data
Analysing Data
Visualizing Data

Modelling & Simulation Practices
Using Computational Models to Understand a Concept
Using Computational Models to Find and Test Solutions
Assessing Computational Models
Designing Computational Models
Constructing Computational Models

Computational Problem Solving Practices
Programming
Choosing Effective Computational Tools
Assessing Different Approaches/Solutions to a Problem
Developing Modular Computational Solutions
Creaing Computational Abstractions
Troubleshooting and debugging

System Thinking Practices
Investigating a Complex System as a Whole
Understanding the Relationships within a System
Thinking in Levels
Communicating Information about a System
Defining Systems and Managing Complexity

Figur 1. CT-praksisser (Weintrop et al., 2016)

MODELING AND SIMULATION PRACTICES
Ved at gøre modeller centrale i de indgående læringsaktiviteter trækker vi især på kategorien 'Modeling & Simulation Practices' (se fig. 1), som handler om at bruge, designe og skabe computermodeller til at forstå og beskrive fænomener, og om kritisk at vurdere fordele og ulemper ved en given model.

Tre års erfaringer med udviklingsarbejde og efteruddannelse af lærere har bekræftet os i, at denne måde at introducere CT-praksisser fag i gymnasiet er både praktisk gennemførlig, fagligt styrkende, og didaktisk inspirerende (Musaeus & Musaeus, 2019).

I en undervisning, der skal nå alle elever, er det især de tre øverste niveauer i 'Modeling & Simulation Practices', der er relevante. De to nederste niveauer – 'designing' hhv 'constructing' computational models – må, i hvert fald i en undervisning, der er rettet mod alle, opfyldes ved at 'designing' og 'constructing' modificeres til 'modifying', som der er gjort rede for i afsnit 7.

Også praksisser knyttet til de tre andre kategorier kan inddrages i den her beskrevne tilgang.

SYSTEMS THINKING PRACTICES
Ved at fokusere på faglige fænomener gennem modeller hjælper vi eleverne til at forstå disse fænomener som interagerende systemer, der indeholder kausalkæder og kausalsammenhænge på forskellige niveauer af kompleksitet. I nogle modeller får eleverne konkrete oplevelser af, at ændringer på ét niveau (f.eks. individniveau i en samfundsfaglig model) får modellen – og dermed systemet – til at ændre opførsel på et aggregeret niveau.

I andre modeller vil de se, at et makroskopisk niveau – f.eks. trykket i en gas – afhænger af de egenskaber, modellen tilskriver de enkelte molekyler på et mikroskopisk niveau. Eleverne er nødt til at overveje, hvordan et komplekst system – f.eks. en celle – på ét niveau beskrives af modellen som en helhed, samtidig med at modellen – gennem modellering af enkeltdelenes egenskaber og opførsel – indeholder andre egenskaber, der stammer fra de underliggende niveauer.

Arbejdet med modellering kan således udvide fagligheden ved at inddrage systemforståelse. Senere eksempler vil vise, hvordan der gennem arbejde med computermodeller kan etableres sammenhænge mellem modellers opførsel på mikro- og makroniveau. Andre erfaringer peger på, hvordan arbejde med modellers niveauer f.eks. i kemi kan supplere praktisk arbejde med laboratorieøvelser, hvor eleverne iagttager og måler på makrofænomener (pH, temperatur osv.) mens de i lærebøgerne finder forklaringer, der tager udgangspunkt i størrelser på mikroniveau.

DATA PRACTICES
Nogle af de modeller, som eleverne arbejder med, kan generere og præsentere data. Derved får eleverne dels mulighed for selv at producere data (f.eks. antal smittede og/eller døde i en epidemi-model), som er autentiske i den forstand, at eleverne selv har modificeret modellen til at producere netop disse data, som der senere kan arbejdes videre med i form af analyse eller repræsentation. Eller eleverne kan arbejde med at præsentere mikroskopiske eller makroskopiske data i realtid, når modellen kører.

COMPUTATIONAL PROBLEM SOLVING PRACTICES

I traditionel it-undervisning er denne sidste kategori ironisk nok ofte den sværeste at håndtere bredt i undervisningen, da den bl.a. kræver, at eleverne arbejder med computerkode. Vores erfaring er nu, at med den tilgang til at læse og ændre computerkode, vi beskriver her, kan det lykkes at få alle elever til at arbejde med og ændre i computerkode.

Vi inddrager specifikt og målrettet praksisser hørende til denne kategori ved:

- At arbejde med et agentbaseret programmeringssprog, NetLogo, som er specifikt designet til at være nemt at lære for nybegyndere (Wilensky, 1999). NetLogo er beskrevet i afsnit 7.
- At bruge en tilgang til undervisningen, hvor arbejdet med computerkode kombineres med en computermodel, der modellerer et fagligt fænomen, som eleverne allerede kender, eller som de alligevel skal lære, fordi det indgår i det faglige pensum (CMC-tilgangen. Se afsnit 5). På denne måde kan eleverne reflektere over forholdet mellem – på den ene side – deres viden om det faglige fænomen og modelleringen af det, og – på den anden side – computerkoden bag modellen. Denne konkretisering af kodens forhold til både det faglige fænomen og til dets repræsentation gennem modellen giver eleverne optimale muligheder for at forstå og eksperimentere med, hvad koden gør.

Da CT omfatter kompetencer, som eleverne forventes at kunne gøre brug af i løbet af deres opvækst og hele liv, herunder deres fremtidige karrierevalg, har det været vigtigt for os ikke at fokusere entydigt på f.eks. programmeringsaspektet af CT. Vi mener, at udgangspunktet i modellering, giver et godt fundament for at introducere CT til eleverne, samtidig med at det er klart, at vi gennem CMC-tilgangen ikke dækker alle de omtalte praksisser.

Tre lærere, der har kombineret elevers arbejde med computermodeller og laboratorieforsøg, melder tilbage:

"Udbyttet i forhold til at få en forståelse af de dynamiske processer, der ligger til grund for kemiske processer, er højt."
(Mandlig kemilærer, 2019)

"Det våde laboratorie og computersimuleringen i samspil bidrager til at øge elevernes fagfaglige udbytte af øvelsen."
(Kvindelig kemilærer, 2019)

"Afprøvningen af simuleringen supplerede litteratur og eksperimenter. Og den legende del af afprøvningen virkede motiverende."
(Kvindelig fysiklærer, 2019)

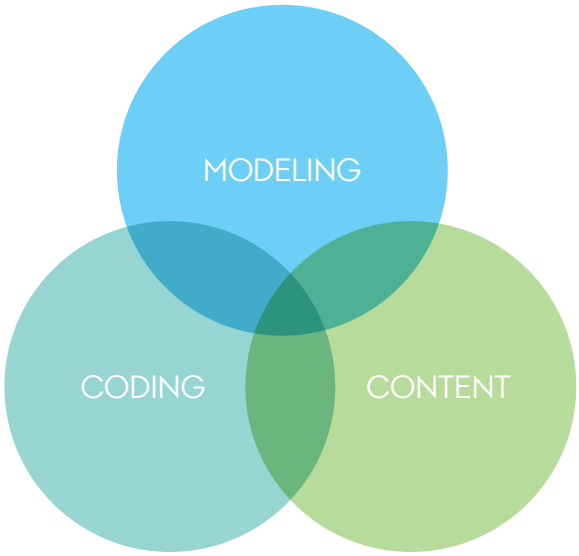
---

5. CMC-TILGANGEN

Vi har valgt at inddrage CT i undervisningen gennem det, vi kalder CMC-tilgangen (Musaeus & Musaeus, 2019). I CMC-tilgangen integrereres faglig viden (Content) med modellering (Modeling) og programmering (Coding) i et samlet læringsforløb (se fig. 2).

I planlægningen af et CMC-undervisningsforløb vælger læreren et fagligt fænomen, som eleverne skal arbejde med. Læreren skriver selv en simpel computermodel eller finder en allerede udarbejdet model, der kan bruges direkte, eller som læreren kan tilpasse, så den passer med lærerens ideer til det kommende undervisningsforløb. Koden bag modellen tilstræbes at være simpel og let at overskue (se afsnit 9.1 og 12.1). Derefter planlægger læreren sin undervisning (målsætning, beskrivelse, rammesætning, sammenhæng med anden undervisning osv.) og udarbejder læringsaktiviteter.

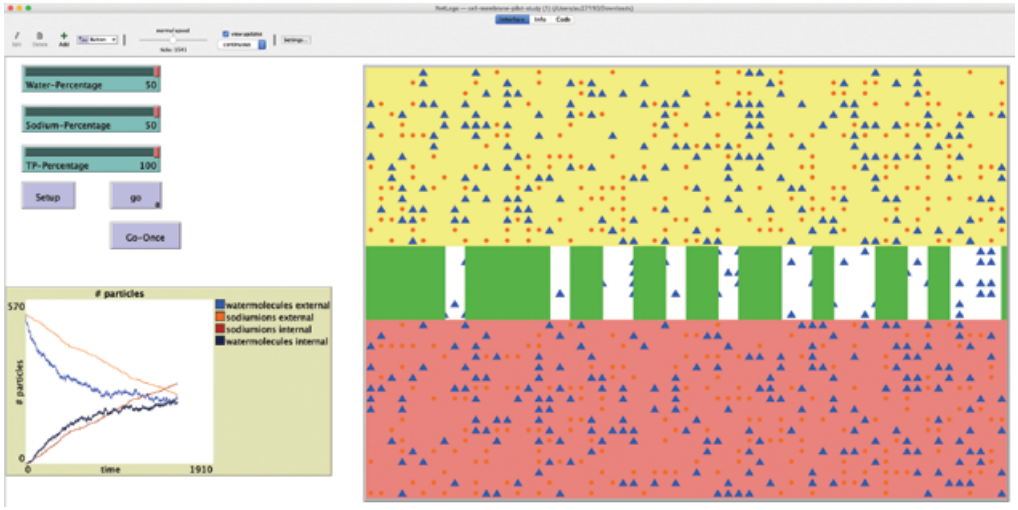
I selve undervisningen arbejder eleverne med fagligt indhold, kodning og modellering på samme tid: Eleverne installerer et program på deres computer, indlæser kodefilen og starter modellen. Undervisningen er i gang. Det forudsættes ikke, at eleverne på forhånd kender til computerkode eller programmering. De starter 'forudsætningsløst' ved at lære at bruge computermodellen gennem dens interface (se fig. 3). Modellen er tilstræbt simpel i sin udformning og indeholder få elementer, så eleverne kommer ret hurtigt frem til at forstå både fænomenet, som modellen repræsenter, samt koden bag modellen. Dette gør det muligt at designe læringsaktiviteter, som både træner elevernes læring i faget og modellering og kodning. Ofte vil læringsaktiviteterne tage udgangspunkt i det faglige indhold ('Content' - feltet i fig. 2) og i første omgang blot bede eleverne arbejde med den visuelle repræsentation af fænomenet. Men gradvist kan læringsaktiviteterne lægge op til, at eleverne forholder sig til modelleringen af fænomenet og dermed også den beskrivelse, der optræder i koden bag modellen, f.eks. hvilke regler objekterne/aktørerne i modellen bevæger sig efter, og hvilket udseende de har.



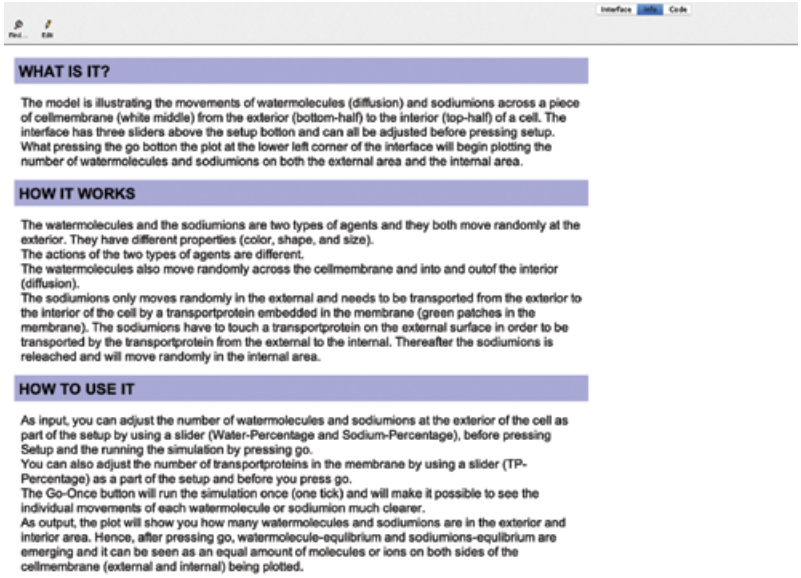
Figur 2: CMC-tilgangen integrerer Content (fagligt indhold) med Coding gennem Modeling.
(Musaeus & Musaeus, 2019)

I NetLogo har eleverne adgang til tre forskellige skærme:

INTERFACE
Viser en simulering af det modellerede fænomen. Evt. – som på figuren – med tilhørende kurver, der i realtid viser udviklingen af en eller flere parametre.



INFO
En kommentarskærm, som beskriver, hvad der er modelleret, og hvordan.



WHAT IS IT?

The model is illustrating the movements of watermolecules (diffusion) and sodiumions across a piece of cellmembrane (white middle) from the exterior (bottom-half) to the interior (top-half) of a cell. The interface has three sliders above the setup botton and can all be adjusted before pressing setup. What pressing the go botton the plot at the lower left corner of the interface will begin plotting the number of watermolecules and sodiumions on both the external area and the internal area.

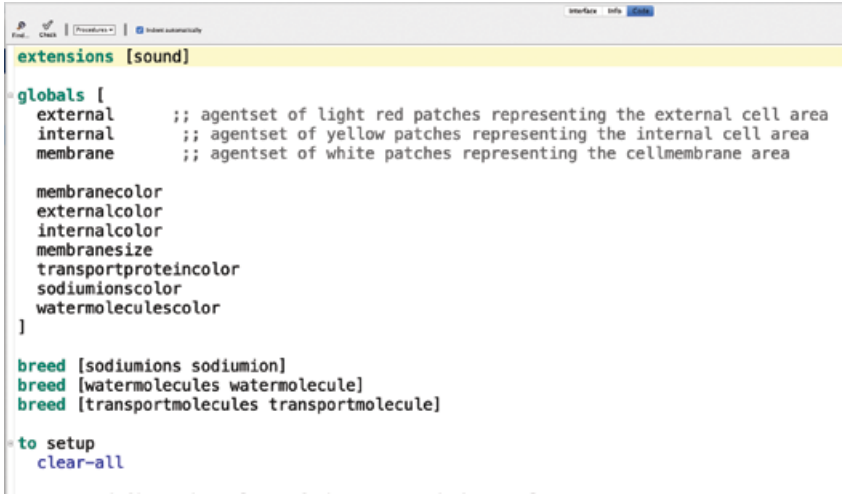
HOW IT WORKS

The watermolecules and the sodiumions are two types of agents and they both move randomly at the exterior. They have different properties (color, shape, and size).
The actions of the two types of agents are different.
The watermolecules also move randomly across the cellmembrane and into and outof the interior (diffusion).
The sodiumions only moves randomly in the external and needs to be transported from the exterior to the interior of the cell by a transportprotein embedded in the membrane (green patches in the membrane). The sodiumions have to touch a transportprotein on the external surface in order to be transported by the transportprotein from the external to the internal. Thereafter the sodiumions is releached and will move randomly in the internal area.

HOW TO USE IT

As input, you can adjust the number of watermolecules and sodiumions at the exterior of the cell as part of the setup by using a slider (Water-Percentage and Sodium-Percentage), before pressing Setup and the running the simulation by pressing go.
You can also adjust the number of transportproteins in the membrane by using a slider (TP-Percentage) as a part of the setup and before you press go.
The Go-Once button will run the simulation once (one tick) and will make it possible to see the individual movements of each watermolecule or sodiumion much clearer.
As output, the plot will show you how many watermolecules and sodiumions are in the exterior and interior area. Hence, after pressing go, watermolecule-equilibrium and sodiumions-equilibrium are emerging and it can be seen as an equal amount of molecules or ions on both sides of the cellmembrane (external and internal) being plotted.

CODE
Viser koden bag modellen. Eleverne kan ændre direkte i koden og teste den nye kode ved at køre simuleringen igen. De kan gemme den ændrede kode, men de kan ikke overskrive det oprindelige program. Går noget galt, begynder de forfra med den oprindelige kodefil.



```
extensions [sound]

globals [
  external      ;; agentset of light red patches representing the external cell area
  internal       ;; agentset of yellow patches representing the internal cell area
  membrane       ;; agentset of white patches representing the cellmembrane area

  membranecolor
  externalcolor
  internalcolor
  membranesize
  transportproteincolor
  sodiumionscolor
  watermoleculescolor
]

breed [sodiumions sodiumion]
breed [watermolecules watermolecule]
breed [transportmolecules transportmolecule]

to setup
  clear-all
```

Figur 3: NetLogo. Modellerne i undervisningsforløbene blev skrevet i programmeringsmiljøet Netlogo. (se også afsnit 6.1).

Et CMC-undervisningsforløb kombinerer således aktiviteter og læringsmål fra både det indgående fag og fra CT. Forløbet er bygget op omkring modellering af en central faglig problemstilling eller fænomen. Ud fra denne computermodel arbejder eleverne ved hjælp af læringsaktiviteter med fænomener, begreber og principper fra både faget og fra CT.

Det betyder, at:
- De faglige fænomener, begreber og principper anvendes til at lære CT
- De CT-faglige fænomener, begreber og principper anvendes til at lære faget.

Centralt for CMC-tilgangen er, at eleverne arbejder på tre forskellige niveauer samtidigt (se fig. 3 om Netlogo) og tilegner sig viden og udvikler kompetencer i de forskellige dele sideløbende. De forskellige dele integreres og understøtter hinanden undervejs. Kodning gør det muligt at arbejde på en anden måde med modellering og det faglige indhold og åbner faget op på nye måder. Samtidig motiverer den faglige relevans arbejdet med at lære at programmere. I et CMC-undervisningsforløb udvikler eleverne centrale CT-kompetencer.

I CMC-tilgangen er begrebet 'modellering' centralt. Det at kunne modellere et fænomen og forholde sig kritisk til modellens rækkevidde (muligheder og begrænsninger) er centralt i mange fag. Det er en pointe at gøre elever opmærksom på dette og formidle, at ved at kunne modellere er der mange andre sammenhænge både indenfor et bestemt fænomen i et fag, men også som tilgang til andre fænomener og problemer på tværs af fag.

CMC-tilgangen gør det muligt for elever at arbejde aktivt og konstruktivt med modellering i undervisningen, så de på egen hånd kan opleve modellen som et fagligt redskab til at teste hypoteser og udforske faget, komme med forudsigelser og afprøve deres egen forståelse. Samtidig kan de erfare, at en model ikke er hele sandheden om virkeligheden, men har begrænsninger, som stammer fra de valg, fravalg og forsimplinger, der blev gjort, da modellen blev konstrueret og efterfølgende kodet.

Tilbagemelding fra lærer, der har gennemført et CMC-forløb:

"Eleverne lærer ikke at programmere. De lærer, at simuleringer afhænger af de kommandoer, som indtastes i koden."
(Kvindelig biologilærer, 2019)

EKSEMPLER: BRUG AF COMPUTERMODELLER I FORSKNING

Computermodeller spiller en stadig større rolle i den naturvidenskabelige forskning. I takt med at computerkraften er steget, er computerbaserede modeller blevet en stadig mere væsentlig metode til at opnå nye erkendelser.

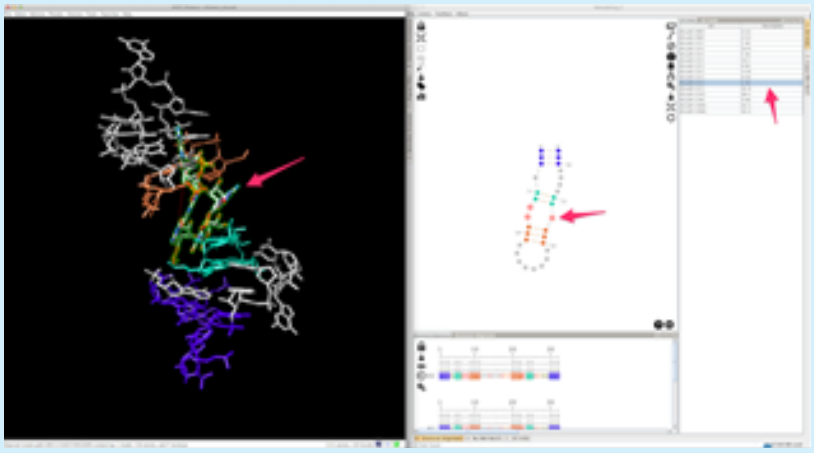
For at illustrere, hvordan anvendelse af computere – og dermed anvendelse af computermodeller – har fået en central rolle i forskningen, giver vi en række korte eksempler, knyttet til fag fra gymnasiets fagrække. Pointen er, at det ikke længere er rimeligt at undervise i fagenes identitet og metoder uden at omtale og illustrere computernes og modellernes rolle. At anvende computermodeller i undervisningen giver samtidig eleverne en realistisk opfattelse af fagene, så man undgår, at de forlader 3.g med en forældet opfattelse.

BIOLOGI
Computermodeller af biologiske systemer gør det i dag muligt at forstå, hvordan komplekse sygdomme hos mennesker og dyr opstår, udvikler sig og behandles. Det drejer sig f.eks. om kræft, hjerte-kar-sygdomme, sukkersyge samt neurologiske og immunologiske sygdomme såsom Alzheimers sygdom og allergi.

I takt med, at vores viden om udviklingen af sygdomme forbedres, bliver det stadigt mere komplekst at beskrive, hvordan kroppens enkelte dele bidrager til det samlede sygdomsbillede. Derfor er der udviklet en ny type biologi, der anvender computere og ligninger til at modellere sammenhænge mellem data fra laboratoriearbejde, feltarbejde og teoretisk arbejde. Med en kvantitativ biologi kan de forskellige dele af et biologisk system sættes sammen til en overskuelig computermodel, der kan eksperimenteres med.

F.eks. kan proteiners struktur og foldning nu forudsiges af komplicerede computermodeller som Rosetta https://www.rosettacommons.org/, der beregner energiniveauerne for forskellige proteinfoldninger og angiver den mest sandsynlige foldning af et bestemt protein.

Sådanne computationelle modeller bruges også som en del af udviklingen af ny medicin og til f.eks. at forstå, hvordan muterede proteiner kan være ophav til sygdomme eller vacciner.
(Kilder: Nature, June 2019. https://www.nature.com/articles/d41586-019-02251-x)
(https://studier.ku.dk/bachelor/kvantitativ-biologi-og-sygdomsmodellering/)

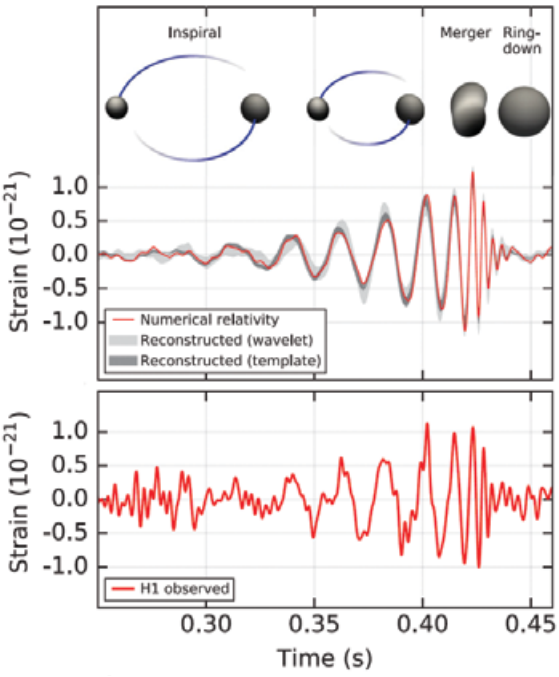


KEMI
I kemisk forskning bruges computermodellering meget. F.eks. i modellering af RNA-molekylets 3D-struktur, hvor modellen bruges til at fortolke og forstå eksperimentelle data.

At bestemme RNA-molekylets 3D-struktur udelukkende ud fra laboratorieeksperimenter er både langsommeligt, svært og dyrt. Derfor kendes strukturen af mange RNA-molekyler stadig ikke. Men nyligt udviklede computationelle metoder bruger de strukturer, der allerede kendes, som et læresæt, hvorudfra ukendte RNA-strukturer kan forudsiges. Derved tages der bl.a. højde for komplekse systemer såsom evolution og funktionalitet af molekylet.
(Kilder: Bioscience Report, 2019. https://portlandpress.com/bioscirep/article/39/2/BSR20180430/110908/Computational-modeling-of-RNA-3D-structure-based),
(http://www.bioinformatics.org/assemble/howto_construct3d.html)

FYSIK
Et eksempel på fysisk forskning, der ikke ville være mulig uden massiv anvendelse af computere, er den sensationelle tolkning af gravitationsbølger i 2015. Her kunne man ved at sammenligne det målte signal med signalet fra over 250.000 modelbaserede simuleringer fastslå, at signalet kom fra to sammensmeltende sorte huller med masser på hhv. 36 og 29 solmasser (Pedersen, 2016).

Ved at inddrage computermodeller kunne man altså opnå en dybere indsigt i fænomenet og give en fysisk forklaring (en mikroskopisk model) på de indsamlede data i form af det målte signal.



Figur3. *Øverst:* Model (simulering) af tyngdebølgerne fra sammensmeltning af to sorte huller, sammen med tegninger af de tre faser. *Nederst:* Det målte signal fra Hanford-detektoren efter en række filtreringer, der undertrykker baggrundsstøj uden for detektorens optimale frekvensområde på 35-350 Hz.

(Kilde: http://www.kvant.dk/upload/kv-2016-1/kv-2016-1-JOPP-Tyngdeboelger.pdf)

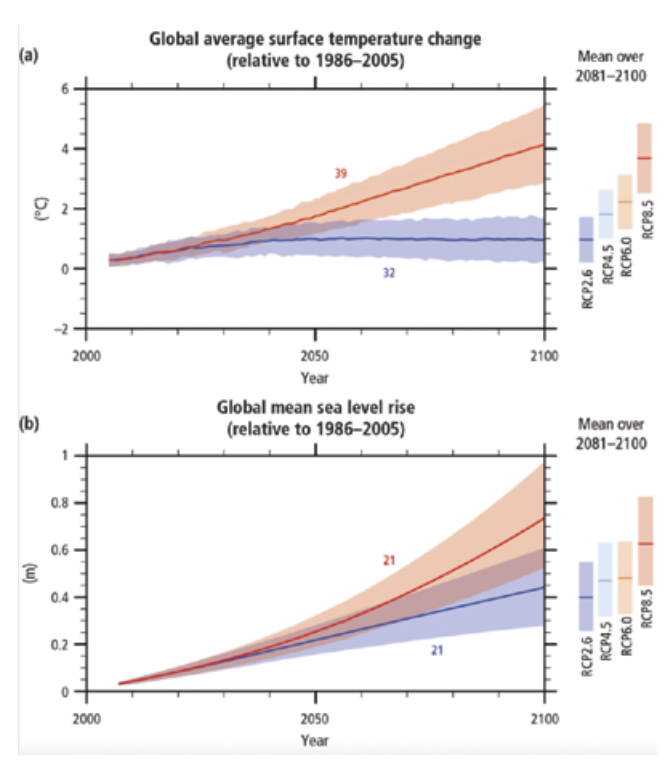
KLIMAFORSKNING

Undersøgelser af, hvordan Jordens klima vil udvikle sig de kommende årtier og århundreder, er et eksempel på forskning, som slet ikke ville være mulig uden brug af computermodeller.

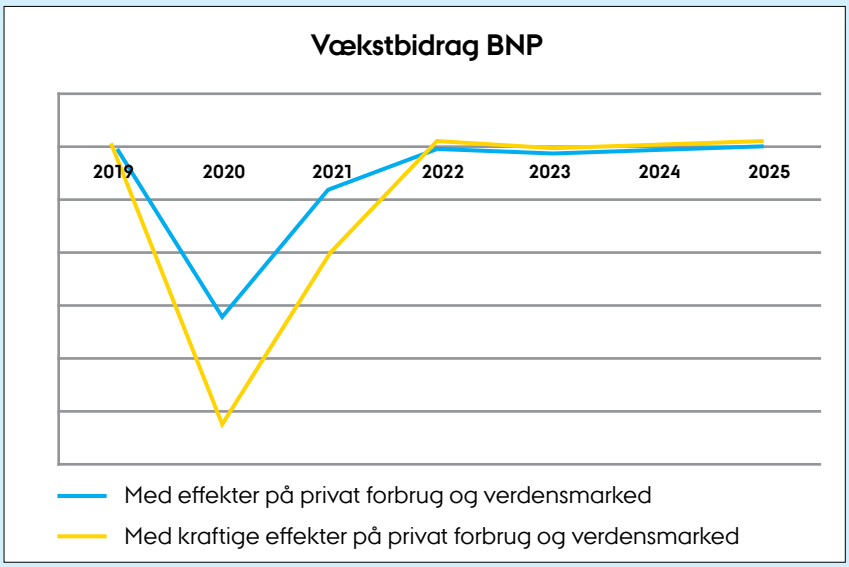
Ligesom vejrprognoser kræver klimamodeller omfattende datamængder og stor computerkraft. En klimamodel er opbygget af mange individuelle undermodeller, der afspejler naturvidenskabelige love samt anslåede værdier for kritiske parametre; f.eks. størrelsen på CO_2-udledningen i fremtiden, fremtidens energibehov, havenes CO_2-absorption osv.

En klimamodel er således et eksempel på en computermodel, som anvendes for at finde ud af, hvordan et stort komplekst system, der er en interaktion af mange enkelte processer, opfører sig. Den kan hjælpe os med at forstå helheder og sammenhænge, som vi ikke ellers ville kunne adressere. Ved at sammenligne resultater fra simuleringen kan vi vurdere, hvilke processer i klimaet der spiller en stor rolle for udviklingen, og hvilke der spiller en mindre rolle. Og med gode modeller kan vi forudsige fremtidens klima.

(Kilde: https://www.ipcc.ch/report/ar5/syr/synthesis-report/)



Øverst: Gennemsnitstemperatur for Jordens overflade i perioden 2006-2100, beregnet med multi-model simuleringer for to forskellige scenarier (RCP8.5 og RCP2.6). Udgangspunktet er perioden 1986-2005. De røde og blå områder angiver usikkerheden på modelberegningerne. Til højre for diagrammet er angivet modelberegnede gennemsnit med tilhørende usikkerhedsintervaller for perioden 2081-2100 i fire forskellige scenarier.
Nederst: Tilsvarende multi-modelberegninger af den gennemsnitlige globale havstigning.



SAMFUNDSFAG
Modeller anvendes i stor udstrækning i samfundsfag, specielt inden for økonomi.

Illustrationen viser et bud på udviklingen i BNP i de kommende år, når man indregner effekterne af corona-krisen. Modellen (ADAM) bruges altså til at lave en fremskrivning ud fra vores viden om nutiden.

Modellerne anvendes også til at regne på fordelingspolitiske konsekvenser, altså om et politisk tiltag er til fordel for bestemte befolkningsgrupper.
(Kilde: https://www.dst.dk/da/Statistik/ADAM)

6. MODELLER I UNDERVISNING

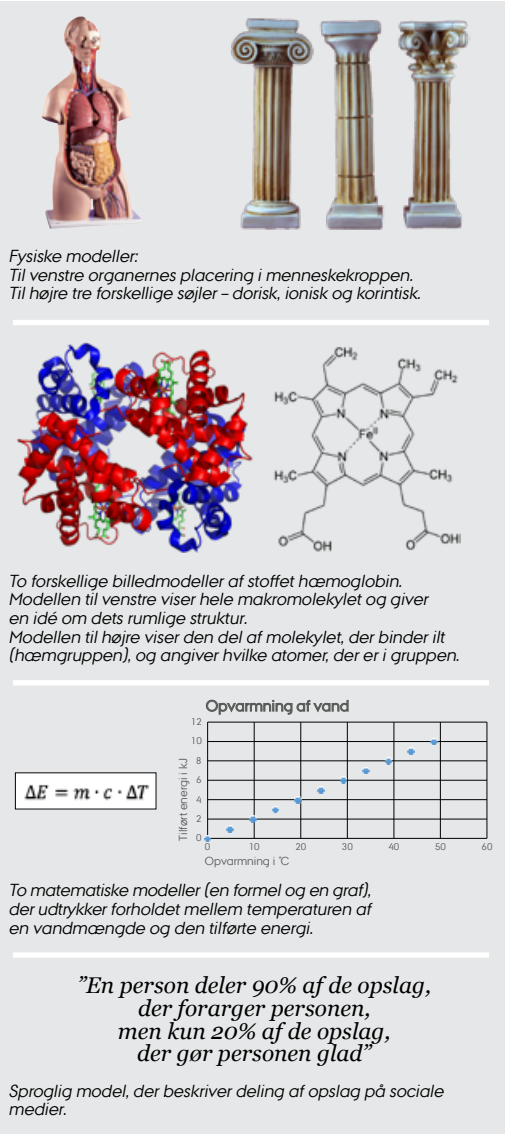
Modeller er redskaber til at udforske og opnå ny viden om den verden og det samfund, vi er en del af, og til at forklare, diskutere og forstå den viden, vi allerede har - eller er lige ved at få – i samarbejde med andre.

En model er en menneskeskabt repræsentation af et udsnit af virkeligheden: En menneskelig konstruktion, der skal hjælpe os til at forstå, afprøve og kommunikere om fænomener og forhold i den virkelige verden.

Modeller kommer i mange former. De kan for eksempel være fysiske, billedlige, matematiske eller sproglige (fig. 4).

Det samme udsnit af virkeligheden kan repræsenteres med forskellige modeller. Afhængigt af anvendelsen vil nogle modeller være mere velegnede end andre i bestemte situationer. Figur 5 viser to forskellige modeller af Jorden. Den ene model er ikke mere korrekt end den anden. De beskriver blot forskellige sider af samme fænomen og er begge forsimplede repræsentationer af den virkelige Jord.

En model er en forenklet repræsentation af virkeligheden, og alle modeller har derfor begrænsninger i deres anvendelsesområde, rækkevidde og præcision. I eksemplet med Jorden er modellernes begrænsninger åbenbare, men for andre modeller (f.eks. den matematiske formel for opvarmning af vand vist i figur 4[1]) kan disse begrænsninger være vanskeligere at få øje på. Statistikeren George E.P. Box udtrykte det således: 'Essentially, all models are wrong, but some are useful' (https://en.wikiquote.org/wiki/George_E.P._Box).



Fysiske modeller:
Til venstre organernes placering i menneskekroppen.
Til højre tre forskellige søjler – dorisk, ionisk og korintisk.

To forskellige billedmodeller af stoffet hæmoglobin.
Modellen til venstre viser hele makromolekylet og giver en idé om dets rumlige struktur.
Modellen til højre viser den del af molekylet, der binder ilt (hæmgruppen), og angiver hvilke atomer, der er i gruppen.

To matematiske modeller (en formel og en graf), der udtrykker forholdet mellem temperaturen af en vandmængde og den tilførte energi.

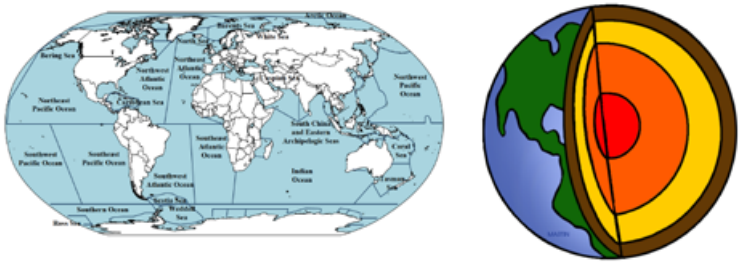
Sproglig model, der beskriver deling af opslag på sociale medier.

Figur 4: Modeller kan være fysiske, billedlige, matematiske eller sproglige.

På grund af forenklingerne har man ofte brug for flere forskellige modeller for at forstå/beskrive et fænomen nogenlunde til bunds. F.eks. både en partikel- og en bølgemodel til at beskrive lys.

Modelbegrebet er mere centralt i nogle fag end i andre. Det gælder både på gymnasialt niveau og i forskningssammenhæng. I gymnasiefag som matematik, samfundsfag og de naturvidenskabelige fag arbejdes der aktivt med modelbegrebet, mens modellerne ofte er sværere at få øje på i fag som dansk, musik, historie og engelsk. Ikke desto mindre er der adskillige fænomener i disse fag, som kan modelleres, og hvor arbejdet med modeller kan bruges til at afprøve forhold og sammenhænge, der har faglig relevans.

[1] Begrænsningerne består for eksempel i, at den kun gælder, hvis vandet er helt isoleret fra omgivelserne, og kun i et temperaturinterval, hvor der ikke sker faseovergange.



Figur 5: To forskellige modeller af Jorden. En model, som den til højre, der viser de forskellige lag af Jorden, vil måske være en god model for en geofysiker, mens et landkort er en mere anvendelig model for kaptajnen på et skib.

Naturvidenskabelig viden består for størstedelen af modeller (matematiske, billedelige, sproglige, computerbaserede). At forstå et naturvidenskabeligt fænomen er i det store hele synonymt med, at man har udviklet en model, der kan simulere eller repræsentere det. Naturvidenskabelig forskning søger hele tiden at udfordre og teste de eksisterende modeller og nå frem til nye, mere omfattende og præcise modeller af virkeligheden. Det er selve hjertet af naturvidenskab. Modeller kan testes i relation til den fysiske verden, f.eks. kan astrofysiske modeller bruges til at undersøge, hvordan egenskaberne og vekselvirkningerne af stjerner skal være for at danne de galakser, som vi ser i Universet i dag. Eller en model kan bruge til at lave forudsigelser om, hvordan et eksperiment vil opføre sig.

Modeller anvendes også i stor udstrækning i samfundsfag, specielt inden for økonomi. For eksempel anvender regeringen og Folketinget en række økonomiske modeller (DREAM, ADAM, SMEC) til at lave konsekvensberegninger af forskellige politiske tiltag. Ud over de makroøkonomiske konsekvenser (ledighed, betalingsbalance, vækst etc.) regner man også på fordelingspolitiske konsekvenser, altså om et tiltag i særlig grad er til fordel for bestemte befolkningsgrupper.

I de senere år, har der været en stigende kritik af de økonomiske modeller. En kritik, som har været rejst i mange år er, at modellerne ikke kun bygger på faktuelle empiriske data, men at der også ligger ideologiske antagelser bag nogle af udregningerne. Et eksempel er den effekt, som en nedsættelse eller afskaffelse af topskatten vil have. Her er fagøkonomer uenige om, hvordan effekterne opstår og deres størrelse, og modellen må derfor nødvendigvis bygge på nogle antagelser, som ikke alle økonomer er enige i (Jespersen, 2018).

En nyere kritik går på, at modellerne ikke indregner en række andre vigtige forhold. Når det gælder miljøbelastning, er der f. eks. ikke taget højde for de skadevirkninger, som industriproduktionen giver, og der er heller ikke indregnet, at man må forvente en udgift til oprydning og udbedring af skader på naturen. Det samme gælder værdien af offentlige goder som uddannelse og offentlige daginstitutioner. Her er der en tendens til, at de eksisterende modeller kun ser disse poster som udgifter og ikke medregner de positive effekter i form af øget produktivitet og muligheden for at begge køn kan have erhvervsarbejde.

Også samfundsfaglige modeller kan bruges til at afprøve forhold i den virkelige verden og lave forudsigelser. F.eks kan man studere, hvordan kriminalitet udvikler sig i et samfund ud fra forskellige forudsætninger. Det kan også være, hvordan en nyhed spreder sig på de sociale medier ved at afprøve forskellige antagelser for de enkelte individers adfærd.

Resultatet af en sådan afprøvning eller forudsigelse afhænger af den eller de anvendte modeller for fænomenet, samt de tal, som modellen fodres med i form af input. På den måde kan en model testes og holdes op imod den virkelige verden, så man kan vurdere modellens gyldighed. Stemmer modellens forudsigelser for dårligt med virkeligheden, må den forkastes eller justeres.

6.1 MENTALE MODELLER

Modellerne beskrevet ovenfor kan vi kalde ydre modeller. Overfor de ydre modeller står mentale modeller. Mentale modeller er mentale repræsentationer – en slags analoge repræsentationer af specifikke fænomener i omverdenen – som hvert individ genererer igennem kognitiv aktivitet. De kan også beskrives som interne beskrivelser af objekter, ideer eller kausale sammenhænge, som er unikke for den enkelte, og som opstår og udvikles igennem interaktion med omgivelserne.

Når elever læser i deres grundbøger, ser en video eller lytter til lærerens forklaringer, konfronteres deres mentale modeller med ydre visuelle eller verbale modeller. Der opstår en interaktion mellem de mentale og de ydre modeller. På den ene side arbejder eleverne med at rette deres mentale modeller til og korrigere dem i forhold til de modeller, de bliver præsenteret for. På den anden side bliver aflæsningen og tolkningen af de ydre modeller præget af de mentale modeller, som eleverne bringer med sig.

Mentale modeller udfordres ikke alene i mødet med de ydre modeller, men også i mødet med den omgivende virkelighed – for eksempel i mødet med naturen, samfundet, teknologien, kunsten og relationer (Rapp, 2005).

Mentale modeller er altså konstant i udvikling. Mentale modeller behøver ikke at være teknisk korrekte, men skal være funktionelle i den forstand, at den enkelte føler, at de bidrager til hans/hendes forståelse af verden. Herunder forståelse af fagenes indhold.

Mentale modeller er i høj grad personlige og dynamiske, men også meget vanskelige at få indsigt i for andre, da de kun eksisterer i menneskers hjerner.

Mentale modeller er psykologisk konstruerede modeller af virkeligheden. Ved at tale med en kammerat eller en lærer om det fænomen, som modellen er konstrueret over, kan eleven udtrykke sin model verbalt. Eller eleven kan udtrykke den skriftligt/billedligt gennem udarbejdelsen af fysiske modeller – f.eks. tegninger, stop-motion-film, figurer – eller ved at arbejde med computermodeller. Det er lærerens møde med en elevs mentale modeller, der giver mulighed for at få indblik i elevens hverdagsforestillinger eller forståelse af et fagligt fænomen (Guy-Gaytán et al., 2019).

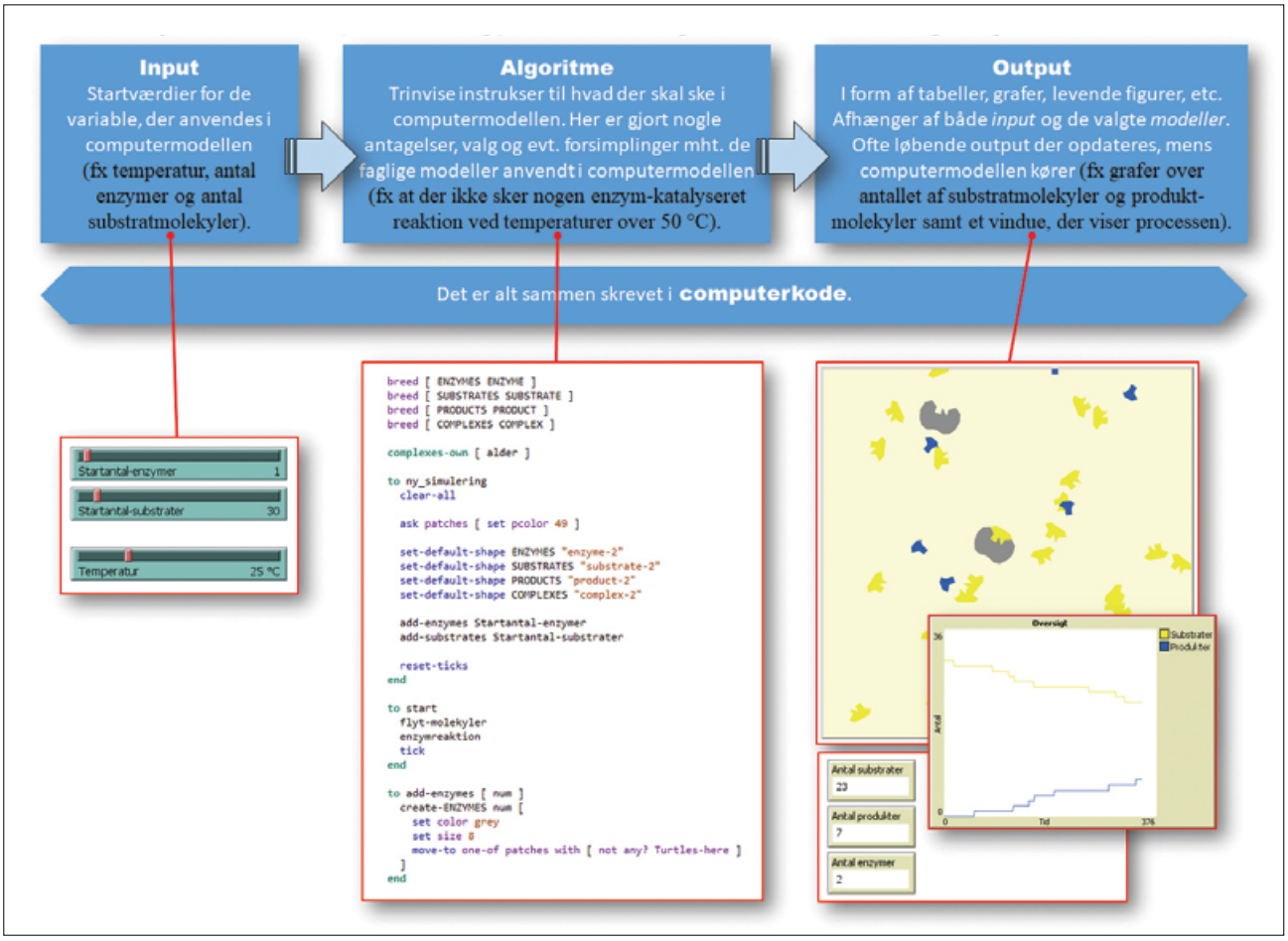
6.2 COMPUTERMODELLER

I sin simpleste form er en computermodel en hvilken som helst faglig model, der er omsat, så en computer kan behandle den.

Fig. 6 viser elementer fra en computermodel, lavet i programmeringsmiljøet NetLogo (NetLogo omtales særskilt i afsnit 7.1). Computermodellen består groft sagt at tre elementer: Et *input*, en *faglig model*, som er skrevet i kode, der kan eksekveres af en computer og et *output*.

Afhængig af det faglige fænomen og det program, som er anvendt til at lave computermodellen, kan interfacet se meget forskelligt ud. I mange af de computermodeller, der typisk anvendes i undervisningssammenhænge, har lærer og elever ikke adgang til at se eller ændre i de anvendte faglige modeller, og dermed heller ikke til at se, hvordan de konkret er implementeret (algoritmerne). Det er tilfældet for rigtigt mange af de simuleringer, der findes på nettet.

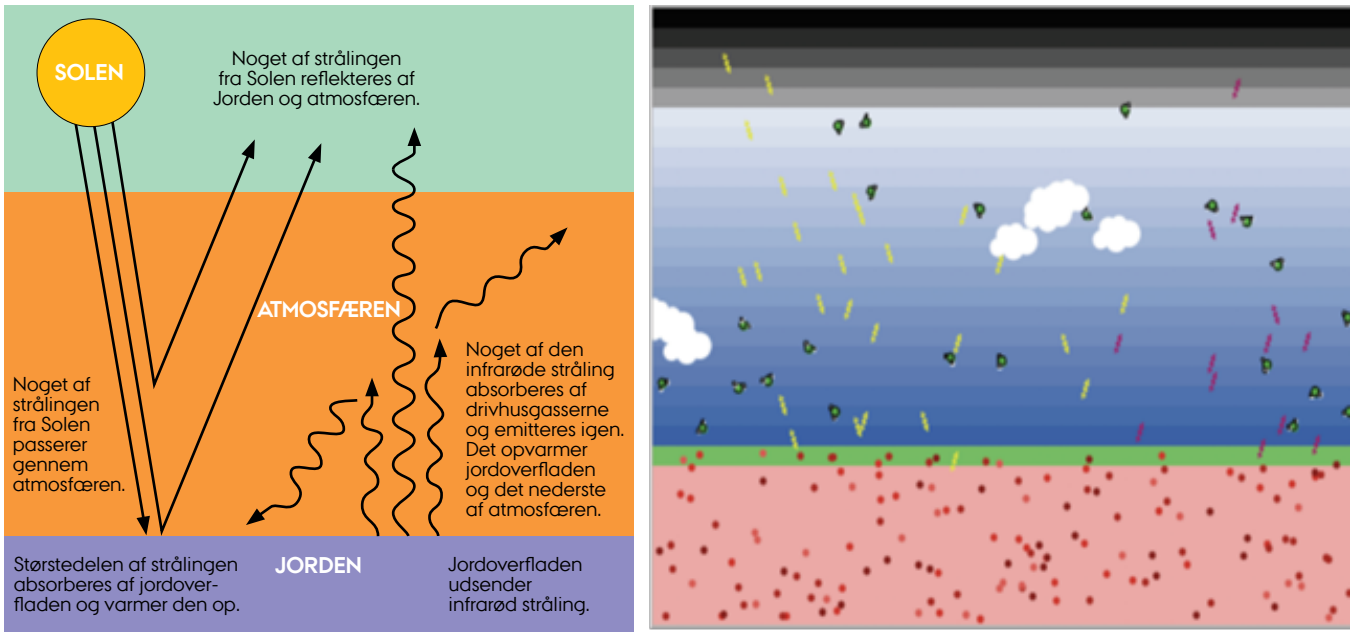
Uden adgang til information om computermodellens opbygning (hvad er udeladt, hvad er taget med, og hvordan er det gjort), er det svært – ofte umuligt – at vurdere en computermodels rækkevidde



Figur 6: Computermodel (simulering) til undersøgelse af en biologisk proces.

og begrænsninger. Det kan lede til den uheldige misforståelse hos eleverne, at computermodeller/simuleringer er en præcis gengivelse af, hvordan virkeligheden faktisk er, og altid giver korrekte og nøjagtige resultater.

Modeller er altid forsimplinger af virkeligheden, og det gælder naturligvis også for computermodeller. Der kan være stor forskel på kompleksiteten af en computermodel, men når et udsnit af virkeligheden omsættes til en computermodel, vil der – uanset hvor kompleks den er – altid være nogle aspekter af virkeligheden, som forsimples eller ikke tages med.



Figur 7: Sammenhæng mellem traditionelt fagligt stof og en computermodel. Til venstre en statisk model (en tegning), der viser træk ved fænomenet 'global opvarmning'. Til højre et øjebliksbillede af en simulering baseret på en computermodel af det samme fænomen, programmeret i NetLogo (Wilensky, 1999). I begge modelleringer optræder de samme mekanismer: Indstråling af solenergi, refleksion, absorption og genudsendelse af stråling fra jordoverfladen, absorption i atmosfæren osv. Det er den samme faglige forståelse, eleverne møder, men i NetLogo-modellen kan de selv ændre modellens opførsel. (Computermodellen findes i NetLogos bibliotek. Se tekstboksen side 49).

En styrke ved en computermodel er naturligvis, at den er i stand til at udføre mange handlinger/beregninger på kort tid. Hvor det i undervisningen ofte kun er muligt at arbejde med en enkelt faglig model – eller et enkelt aspekt af et fagligt fænomen – ad gangen, kan computermodeller håndtere mange fænomener og faglige (under)modeller på én gang. Det bliver dermed muligt at undersøge, hvordan komplekse systemer, der påvirkes af flere forskellige (under)modeller, udvikler sig. Derfor ser man ofte, at flere faglige modeller inkorporeres i en enkelt computermodel.

Et eksempel på dette kunne være den computermodel over klimaet, som er gennemgået i afsnit 10. Her undervises eleverne direkte i sammenhængen mellem 'undermodeller' og en 'hovedmodel'.

Da computermodeller er bygget op omkring faglige modeller, gælder der ligesom med disse, at computermodeller har nogle styrker og begrænsninger, som er vigtige at kende, hvis man skal anvende computermodellens resultater.

Som det er eksemplificeret i tekstboksen 'Brug af computermodeller i forskning' (s. 21-24), spiller computermodeller en uundværlig og voksende rolle i videnskabelig forskning. I nogle discipliner er computermodeller dominerende som redskab for den videnskabelse, der finder sted. Observationer, forsøg og teoridannelse spiller stadig en rolle for de videnskabelige discipliners udvikling, men store dele af det, der før blev omtalt som (felt)observationer eller empiri, er nu et spørgsmål om indhentning af store mængder af data, der opbevares i omfattende databaser. Databankerne er så store, at databehandling 'i hånden' er helt umulig, men foretages ved hjælp af computere. Også mange eksperimenter – f.eks. partikelkollisioner ved CERN – genererer så uhyre mængder af data, at det kræver computerbaseret datavidenskab at finde hoved og hale i datamængderne. Derfor er klassisk teoridannelse i høj grad afløst af modellering ved hjælp af computere, der kan sortere, analysere, teste og præsentere essensen af de opsamlede data.

7. AGENT-BASERET MODELLERING (ABM)

Agentbaseret modellering kan være nemmere for eleverne at gå til end andre programmeringsparadigmer, da det i kraft af sine *agenter* er mindre abstrakt og mere håndgribeligt end traditionelle programmeringstilgange. I vores arbejde med at integrere CMC-tilgangen i fagene, har vi fundet programmeringsmiljøet NetLogo meget velegnet (se afsnit 7.1).

Vigtigt for vores valg af NetLogo har blandt andet været, at det er *agentbaseret*. Det betyder, at udgangspunktet for konstruktionen af en computermodel er en eller flere typer af *agenter*. Afhængigt af, hvilket fag der arbejdes med, og hvad det er for et fænomen, der skal modelleres, kan agenter f.eks. være atomkerner, ioner, fugle, får, borgere, molekyler, penge, etc.

Det centrale i at programmere en agentbaseret model består i at definere agenterne og udstyre dem med egenskaber og adfærd, der definerer, hvordan de ser ud og opfører sig, samt hvordan de interagerer med hinanden og deres omgivelser.

Egenskaber kan f.eks. være agentens størrelse, udseende, farve og position. *Adfærd* kan f.eks. være et bevægelsesmønster, hvordan der træffes beslutninger, frastødning eller tiltrækning eller henfald, hvis agenten er en atomkerne. Herved skabes måske to nye typer af agenter. Det hører f.eks. også med til agenternes adfærd, at det defineres, hvad der sker, hvis de møder en agent af samme slags – måske er det to brintatomer, der møder hinanden og danner en ny type agent, et molekyle. Eller

```
1  breed [får et-får]
2  breed [ulve en-ulv]
3
4  to setup
5    clear-all
6    create-ulve 1 [
7      setxy 0 0
8      set size 2
9      set color gray
10     set shape "wolf"
11   ]
12   create-får 10 [
13     setxy random-xcor random-ycor
14     set size 2
15     set color white
16     set shape "sheep"
17   ]
18   reset-ticks
19 end
20
21 to go
22   ask ulve [ forward 1 ]
23   ask får [ forward 1 ]
24   ask får [ møder-ulv? ]
25   tick
26 end
27
28 to møder-ulv?
29   if any? ulve in-radius 1 [ set color red ]
30 end
```

Figur 8: Programlinjer fra NetLogo-kode. Linjerne er fra indledningen til en model, der viser får og ulve i en rovdyr-byttedyr-cyklus. Selv om man aldrig har set den slags kode før, kan man hurtigt danne sig en forestilling om, hvad der sker i de fleste af linjerne.

I linje 6-9 skabes én ulv (find et-tallet) med størrelse, farve og form. I linje 7 får ulven en position på skærmen.

Dernæst (linje 12-16) skabes 10 får (find 10-tallet) med tilfældige positioner osv. Hvad sker der i linje 21-26?

Det er ikke vanskeligt at ændre i koden, så der skabes flere får, eller at give fårene en anden farve osv. Så er eleven i gang. Eleven lærer ikke at programmere, men finder ud af, hvad et program er for noget.

En rovdyr-byttedyr-cyklus beskrives traditionelt med to koblede differentialligninger (Lotka-Volterra-ligningerne).

(Fra en model skrevet af mandlig fysiklærer, 2019)

to forskellige agenttyper kan interagere, som hvis et får møder en ulv; fåret forsvinder, og måske bliver ulven mere energisk eller tungere.

ABM har den styrke, at programmeringen lægger op til intuitiv identifikation mellem agenterne på den ene side, og den, der skaber eller modificerer agentens udseende eller opførsel, på den anden. Denne egenskab ('Jeg forstår godt, hvorfor agenterne opfører sig, som de gør') er med til at give NetLogo (se afsnit 6.1) en lav indlæringstærskel med hensyn til at bruge, forstå, ændre og skabe nyt i modellen, og dermed opbygge et tilhørsforhold til den.

Agenterne er håndgribelige og konkrete, og det er ofte lettere for eleverne at forholde sig til og arbejde med end f.eks. objekter og klasser i objektorienteret programmering.

Med ABM kan store, komplekse fænomener altså reduceres til et spørgsmål om, hvordan de enkelte agenter agerer i bestemte situationer. Det gør arbejdet med og programmeringen af modeller konkret og let at gå til for eleverne og giver eleverne mulighed for at arbejde med relativt komplekse fænomener, som ellers ville være vanskelige at behandle i undervisningen.

NetLogo

NetLogo er udviklet af Uri Wilensky (Center for Connected Learning ved Northwestern University i Chicago), som var elev af Seymour Papert, pioneren der udviklede sproget Logo og samarbejdede med LEGO om udviklingen af LEGO Mindstorms. (Wilensky, 1999)

7.1 NETLOGO

Der findes flere forskellige værktøjer til at arbejde med CT-kompetencer i gymnasieundervisningen (f.eks. Scratch, micro:bit, Arduino, Raspberry Pi og Python). NetLogo har nogle fordele, der gør det meget velegnet til gymnasiebrug, ikke mindst til CMC-tilgangen, hvor modellering og kodearbejde spiller en central rolle.

NetLogo er et tekstbaseret programmeringssprog, der er designet, så det er forholdsvist simpelt at bruge for både elever og lærere. Det er muligt at komme i gang med at programmere uden at kunne overskue hele sproget og syntaksen i detaljer. Efter en forholdsvis kort indføring kan eleverne arbejde med koden og lave om i modellerne i NetLogo, selvom de ikke er professionelle programmører.

NetLogo er et integreret, multiagent programmeringsmiljø og -sprog til simulering af komplekse fænomener. Det egner sig især godt til modellering af fænomener, der udvikler sig over tid. Det er designet til både forskning og uddannelsesbrug og bruges i en bred vifte af discipliner og uddannelsesniveauer.

NetLogo er velegnet til modellering af fænomener med mange agenter og agenttyper, der interagerer med hinanden. I koden kan der gives instruktioner til hundrede eller tusinde uafhængige agenter på samme tid, så alle udvikles samtidig. Det gør det muligt at undersøge sammenhænge mellem 'mikroskopisk' opførsel af individuelle agenter og 'makroskopiske' mønstre, som optræder på systemniveau pga. agenternes individuelle opførsel (emergerende fænomener, se næste afsnit, se også (Tisue & Wilensky, 2004)).

Programstrukturer og syntaks er relativt let at forstå, og det kræver en minimal mængde kode at lave grafer og visuel repræsentation af det modellerede. Det gør det nemt for eleverne at forstå, hvad de forskellige linjer i koden gør. Vil man for eksempel have nogle molekyler til at flytte sig, kan det gøres med linjen:

'ask molecules [forward 2]'

Til trods for den relativt lave indgangstærskel til NetLogo (sammenlignet med andre programmeringssprog), er det alligevel lavet, så det ligner andre programmeringssprog tilpas meget til, at dét, eleverne lærer i NetLogo – f.eks. mht. et programs opbygning, programstrukturer, kodesyntaks og algoritmer – kan overføres til andre programmeringssprog.

Det blokbaserede programmeringssprog Scratch, som mange nok kender, er også agentbaseret (her kaldes agenterne 'sprites'). Modsat Scratch tillader NetLogo, at eleverne arbejder med den tekstbaserede kode bag computermodellen og ikke blot 'blokke'. Desuden er det betydeligt lettere i NetLogo at skabe mange – rigtig mange – agenter i sin model.

7.2 SAMMENHÆNG MELLEM MODELLERS MIKRONIVEAU OG MAKRONIVEAU

Agentbaseret modellering er velegnet til at undersøge sammenhængen mellem mikroniveau og makroniveau i en model. Altså at undersøge, hvilke mønstre der opstår (emergerer) på et aggregeret niveau på baggrund af mange agenters individuelle opførsel på mikroniveau.[2]

I de naturvidenskabelige fag og samfundsfag har man ofte med systemer at gøre, der består af et meget stort antal agenter (fra hundredvis til trilliardvis). Det er vigtigt for elevernes dybere forståelse af et fænomen, at de kan koble agenternes individuelle egenskaber og opførsel på mikroskopisk niveau med systemets egenskaber og opførsel på det makroskopiske niveau.

Det kan være hensigtsmæssigt at dele modeller op i to kategorier:

- *Modeller på mikroniveau (agent-niveau)*: Modeller/antagelser for de individuelle agenters opførsel. Det er på mikroniveauet, vi finder den dybereliggende forklaring på det, vi iagttager på makroniveau. F.eks. hvordan den enkelte fugl i figur 9 orienterer sig og bevæger sig i forhold til de andre fugle.

- *Modeller på makroniveau (system-niveau)*: Modeller for systemets opførsel. Systemet er en samling af mange agenter og/eller agenttyper, som betragtes som en samlet helhed. F.eks. den 'sorte sky' på figur 9.

I naturvidenskab er det ofte – men ikke udelukkende – makroniveauet, vi kan måle og observere på. Naturvidenskabelige hypoteser, love og teorier (modeller) findes på begge niveauer. Henfaldsloven og ædelgasligningen beskriver et system på makroniveau, mens en atomkernes sandsynlighed for at henfalde og molekylers individuelle hastighed, rotation og vibration er en beskrivelse af opførsel på mikroniveau. I samfundsfag er det typisk mikroniveauet, der iagttages.

Tabellen nedenfor giver eksempler.

---

[2] Der er ikke enighed om, hvordan man bør definere fænomenet 'emergens'. Her bruger vi udtrykket i en forholdsvis bred betydning, i overensstemmelse med følgende opfattelse: *'In the recent years the notion of emergence has been studied extensively, but often without making precise what emergence means. ... In the study of complex systems one often sees that a collection of interacting systems shows collective behaviour. This is what we understand by emergence'* (Baas & Emmeche, 1997).

SYSTEM-NIVEAU	AGENT-NIVEAU. HVEM ER AGENTER?
En klump radioaktivt materiale. Strålingen er givet ved henfaldsloven (makroskopisk model).	De enkelte radioaktive kerner. Hver kerne har en bestemt sandsynlighed for at henfalde hver tidsenhed (mikroskopisk model).
Skattelettelser øger landets BNP (makroskopisk model).	Borgerne, der får forøget deres disponible indkomst. Dermed stiger deres forbrug og virksomhedernes omsætning (mikroskopisk model).
En opløsning med enzymer og substrater. Enzymaktiviteten afhænger af temperaturen (makroskopisk model).	De enkelte enzymer og substrater. Deres rumlige struktur afhænger af temperaturen (mikroskopisk model). Molekylernes hastighed afhænger af temperaturen (mikroskopisk model).
Fake news og postfaktuelle nyheder bliver mere udbredte (makroskopisk model).	Borgerne deler sig i højere grad efter værdier og holdninger, der ikke nødvendigvis hviler på fakta (mikroskopisk model).
Et kammer med en gas. Trykkets afhængighed af temperaturen er givet ved ædelgasligningen (makroskopisk model).	De enkelte atomer eller molekyler i gassen. De bevæger sig hurtigere, jo varmere det er, og rammer tryksensoren oftere og hårdere (mikroskopisk model).
En galakse. Rotationen afhænger af den samlede masse og størrelse af galaksen (makroskopisk model).	Stjernerne i en roterende galakse. De har en masse og hastighed og påvirker hinanden med massetiltrækningskraft (mikroskopisk model).

Et eksempel på en model i naturfag kan være en gas, der er indespærret i en beholder. Gassen er sammensat af molekyler, der bevæger sig tilfældigt rundt, mens de støder ind i hinanden og beholderes vægge. Derved opstår det observerbare makrofænomen, som vi kalder tryk, og som alle, der har blæst en ballon op, har oplevet. Vi iagttager altså makrofænomenet, som skabes i modellen ved at udstyre nogle agenter (molekylerne) med egenskaber (position og hastighed/energi), som vi ikke kan iagttage i virkeligheden. Da modellen gennem emergens skaber den kraft på beholderes vægge, som vi kalder tryk, formoder vi, at antagelserne på mikroniveauet er korrekte. Således kan en computermodel anvendes til at teste en mikroskopisk models/antagelses gyldighed eller til at finde frem til det sæt af mikroskopiske modeller/parametre, der giver det rigtige resultat på makroniveau.

I samfundsfag kan et eksempel være det økonomiske kredsløb. Her antages, at en samfundsøkonomi kan modelleres som et samspil mellem en række indbyrdes afhængige størrelser. Men bag modellernes opførsel på makroniveau ligger en række antagelser på mikroniveau om husholdninger og borgere. I økonomisk teori antager man normalt, at borgerne er rationelle og nyttemaksimerende. Så når vi tror, at øgede offentlige udgifter eller skattelettelser øger beskæftigelsen, er det altså baseret på disse antagelser om borgernes opførsel på mikroniveau.

Det makrofænomen, der modelleres, er altså samfundsøkonomien som et system med mange agenter og mange interaktioner mellem agenterne og systemet. Dette giver mulighed for at inddrage forskellige interaktionsformer. Hvis vi i samfundsfag ser bredt på forholdet mellem borgere og stat kunne disse interaktionsformer være:

- Agent-agent-interaktion: Nogle agenter (borgere) møder andre agenter og ændrer deres adfærd eller ønsker, f.eks. holdning til, om mænd går på barsel
- System-agent-interaktion: Regeringen ændrer på skattesystemet eller dagpengeniveauet, og agenterne (borgerne) indretter deres opførsel efter det
- Agent-system-interaktion: Borgerne presser på for at få indført mere klimavenlig lovgivning, og dette skaber nye rammebetingelser for virksomhederne.

Agenter kan blandt andet være personer, organisationer, virksomheder eller stater. Den grundlæggende modellering af agent-egenskaber følger de samme principper, uafhængigt af hvad det er for agenter.

Med henblik på fagenes metoder er det værd at fokusere på nogle forskellige tilgange, hvor computermodeller anvendes til at skabe ny viden:

1. *Teste en mikroskopisk models rigtighed.* Resultaterne fra en computermodel, lavet på baggrund af antagelser på mikroniveau, sammenlignes med virkeligheden. Stemmer de overens, er de mikroskopiske modeller blevet bestyrket. Passer de ikke sammen, må de mikroskopiske modeller forkastes; helt eller delvis. Denne metode kan bruges til at undersøge, hvordan de mikroskopiske modeller skal laves, så computersimuleringens resultater stemmer overens med virkeligheden, og derigennem opnå viden om, hvad der foregår på et lavere 'skjult' niveau. Et eksempel er vist på side 22 (sorte huller).

2. *Forudsige udviklingen af et komplekst system og undersøge betydningen af forskellige parametre og effekter på udviklingen.* Kan man forklare, hvordan et system over tid udvikler sig ud fra kendte modeller, kan man også forudsige, hvilke parametre man kan ændre eller påvirke for at opnå en bestemt effekt eller et bestemt resultat i virkeligheden. Eksempel på side 23 (klimamodeller).

3. *Studere sammenspillet mellem mikro- og makroniveau.* Modeller kan hjælpe os med at studere, hvilke mønstre og fænomener der opstår på makroniveau ud fra agenternes interaktion på mikroniveau. Ofte har man i samfundsfag modeller for mikroniveauet (f.eks. agenters arbejdsudbudsfunktion) og for makroniveauet (f.eks. den aggregerede indkomst i samfundet målt ved BNP).

Figur 9: Fugle, der flyver i flok og laver 'sort sol', er et klassisk eksempel på emergens.

En video på Youtube viser, hvordan den enkelte fugl ikke forholder sig til 'flokken', men kun til de nærmeste naboer.

I NetLogo-sprog vil man sige, at fuglen er en 'agent'. Se https://www.youtube.com/watch?v=9pRC-IPuhuU&feature=youtu.be

8. PROGRESSION GENNEM TINKERING OG USE-MODIFY-CREATE (UMC)

TINKERING
Som nævnt i afsnit 4 og 5 indeholder CMC-tilgangen bl.a. design af læringsaktiviteter for eleverne. I design af disse tages der udgangspunkt i visualiseringen af det faglige fænomen, som computermodellen bruges til at repræsentere. Ved først at arbejde med repræsentationen af modellen i 'Interface' (se fig. 3) kan eleverne umiddelbart forstå store dele af modellen og fænomenet, samt udforske de muligheder, de indeholder. For eksempel kan eleven ændre visse modelparametre direkte fra interfacet, hvis der optræder 'sliders' eller 'on-off'-knapper på Interface-skærmen.

Denne form for aktivitet har en lav struktur i form af bevidst tilbageholdenhed fra lærerens side, men er karakteriseret ved brug af præcise arbejdsspørgsmål. Elevernes arbejde i denne form for aktivitet er blevet karakteriseret som 'tinkering' (Resnick & Rosenbaum, 2013). Ved tinkering forstås i denne sammenhæng, at eleverne afprøver muligheder i interface og kode uden at reflektere systematisk over sammenhænge og uden at ekstrahere principper og regler ud fra deres afprøvning. Tinkering repræsenterer den konstruktivistiske tilgang, som CMC-tilgangen læner sig op ad. Forskning har vist, at ved at designe læringsaktiviteter med meget lav struktur i starten af et forløb om CT, for derefter at anvende højere struktur i læringsaktiviteterne senere i forløbet, øges elevernes faglige læring (Jacobson et al., 2015).

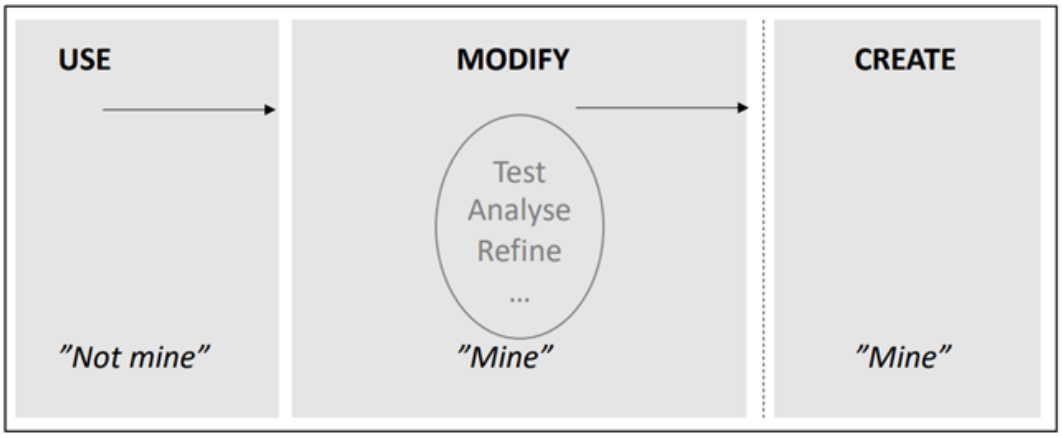
USE-MODIFY-CREATE
Der kan opnås en tydelig progression i undervisningsaktiviteterne ved at anvende et didaktisk princip, hvor der skelnes mellem 'brug', 'modifikation' og 'skabelse' af computermodellen. Denne skelnen omtales ofte som 'Use-Modify-Create'-princippet (UMC). Princippet er beskrevet generelt i forhold til arbejdet med CT i undervisning i Lee et al. (2011).

USE: Eleverne arbejder med visualiseringen af modellen i 'interface'. Dette kan bl.a. være for at forholde sig til, om modellen repræsenterer fænomenet korrekt i forhold til elevernes faglige viden og mentale model. Det kan også være for at se sammenhænge mellem input og output. F.eks. gennem at ændre startparametre, som tastes ind i modellens interface eller ændres gennem brug af en 'slider'. Eleverne kan også arbejde fagligt med output (data) fra modellen.

MODIFY: Elevene arbejder med den computerkode, der ligger bag modellen. De læser og forstår de regler og beskrivelser, der er i koden, og som bestemmer agenternes opførsel og udseende. Eleverne foretager mindre eller større ændringer i koden og ser, hvorledes det ændrer interfacet, modellens opførsel og eventuelle output.

Man kan lade eleverne lave få og simple ændringer i en eksisterende kode, f.eks. ændringer i en enkelt linje eller i agenternes form, farve eller hastighed. Man kan også udfordre eleverne lidt mere og designe læringsaktiviteter, der indebærer, at eleverne skal flytte eller kopiere en kodestump fra ét sted til et andet og evt. rette den til. Endelig kan eleverne arbejde med at skrive et større stykke kode fra bunden af og indsætte det i koden for den eksisterende model. Alle disse aktiviteter er en del af 'Modify', hvilket illustrerer, at denne kategori er meget bred (se figur 10). Fælles for aktiviteter i denne kategori er dog, at eleverne går fra at prøve sig frem til at lave mere velovervejede og fagligt begrundede ændringer i modellen. Elever bevæger sig altså fra at være udforskende til at være kodende og eventuelt til at være modellerende.

CREATE: Eleverne skriver selv 'original' kode. Enten ved at tilføje eller ændre grundlæggende i eksisterende modelkode eller ved at skrive ny kode helt fra bunden eller ultimativt ved at konstruere en helt ny model med ny kode.



Figur 10: Princippet om UMC (efter Lee et al., 2011). I figuren er angivet, hvordan elevernes opfattelse af – og følelse af ejerskab for – computermodellen ændres undervejs i læringsaktiviteterne, hvis princippet følges. Dette uddybes yderligere i afsnit 9.

Elevernes aktiviteter i forbindelse med use-modify-create kan give anledning til progression i arbejdet, der omfatter følgende overgange:

- Eleverne laver gradvist mere omfattende og komplekse ændringer i koden
- Eleverne bliver bevidste om den læring, der ligger i en trial-and-error-tilgang. De forstår, at det er OK at fejle, og at det endda kan være befordrende for ny læring i faget
- Eleverne træner deres vedholdenhed i forhold til opståede småproblemer undervejs, som skal løses. F.eks. at programmet ikke vil køre eller stopper med en fejlmeddelelse. Eleverne presses til at udtænke og afprøve forskellige løsninger eller problemløsningsstrategier
- Eleverne lærer noget fagligt, som de ikke kunne uden denne model
- Læreren kan gøre eleverne opmærksomme på, at de er i gang med at lære gennem en induktiv tilgang
- Læreren kan pointere, hvordan modellen understøtter forståelse af kobling mellem mikroniveau og makroniveau (agent og system)
- Eleverne oplever meget stor motivation og ejerskab over den endelige model og den faglige viden, der er indbygget i den.

Udsagn fra lærer, som har afprøvet CMC-tilgangen:

"Elevernes diskussioner af simuleringens og modellens muligheder og begrænsninger blev brugt til at arbejde mod NV's faglige mål om modellering."

"Ideer til - og gennemførelse af - ændringer i koden med henblik på at forbedre simuleringen sætter tanker i gang om kritiske vinkler til modellen, især hvis de diskuteres."

(Kvindelig fysiklærer, 2019)

9. MODELLER OG KONSTRUKTIVISME

At arbejde med overgange mellem fag (Content), modeller (Models) og programkode (Code) falder fint i tråd med en konstruktivistisk forståelse af læring. At eleven så at sige skal forholde sig til sin egen mentale model af det fænomen, der simuleres af programkoden, er en direkte invitation til, at eleven overvejer sin egen opfattelse, eventuelt diskuterer den med andre, og undervejs rekonstruerer sin egen forståelse.

Dertil kommer, at eleven ser det samme fænomen fra flere sider, så én forståelse bliver udfordret og justeret af den anden. Flere af de lærere, der har testet undervisningsforløb, hvor de bruger CMC-tilgangen, rapporterer om tydelige effekter på elevernes læring og motivation, når de supplerer arbejdet med NetLogo-modellen med (traditionelle) laboratorieforsøg, der omhandler det samme fænomen.

Tilgangen fremmer tilsyneladende evnen til at overføre viden og læring fra én sammenhæng til en anden. Noget som er vanskeligt at opnå i traditionel undervisning (National Research Council, 2014, s. 147).

UMC-tilgangen er også i overensstemmelse med undervisning, der udnytter konstruktivistisk tænkning:

- Undervisningsforløbet indeholder meningsfulde, realistiske opgaver. Eleverne kan med det samme komme i gang med modellen, der relaterer til ting, de allerede kender, og derfor ser realistisk ud. Modellen giver altså lige fra begyndelsen mening, også i undervisnings- og i læringssammenhæng
- I undervisningsforløbet reduceres mængden af passiviserende opgaver. Det betyder, at:
  - omfanget af læsning og andre 'passive' opgaver i de indledende faser kan skæres ned til et minimum, så man hurtigt kommer frem til at arbejde med det faglige indhold
  - eleverne kan hurtigt gå i gang med at være aktive og prøve sig frem ('tinkering').
- Fejl bliver produktive og meningsfulde. Eleverne kan ikke ødelægge modellen eller computeren. Fra spil og lignende er de vant til, at man arbejder sig fremad ved at lave fejl. At begå fejl i klasseværelset med efterfølgende fejlretning bliver mindre traumatisk og læringsmæssigt produktivt.

9.1 FAGLIGT ENKLE MODELLER

Som det for eksempel er illustreret i figur 5, kan man ikke give et allround svar på, hvornår en model er god. Det afhænger af, hvad modellen skal bruges til. En model er nemlig altid et redskab, som man kan bruge til at finde rundt, til at forudsige, til at forstå eller til at forklare. Eller bare til at tænke kreativt med.

De naturvidenskabelige fag i gymnasiet kan se tilbage på en udvikling, hvor flere og flere fænomener er blevet undersøgt, beskrevet og forstået ved hjælp af modeller. For undervisere i disse fag er det oplagt at tænke, at en god model i undervisningen balancerer mellem to yderpunkter. På den ene side skal modellen være fagligt korrekt. Det fordrer kompleksitet i den forstand, at modellen omfatter og opsummerer så mange faktorer og kausale mekanismer som muligt. På den anden side skal modellen være overskuelig for eleverne og derfor have et strøg af enkelthed. Men for de fleste fagligt ambitiøse undervisere er det nok rimeligt at sige, at idealet er at præsentere omfattende og komplicerede modeller, der simulerer virkelige fænomener på en måde, som har en maksimal sandhedsværdi – men dog ikke er så komplicerede, at de kommer over en grænse, hvor de ikke kan forstås og anvendes af eleverne.

I forbindelse med de modeller, der kan bruges i CMC-tilgangen, kommer de samme typer af overvejelser i spil. Der skal findes den rette balance mellem enkelthed og kompleksitet i de faglige modeller, der bringes i spil. Dog er der i ambitionen om, at eleverne skal arbejde med modellen på kodeniveau et ekstra lag af kompleksitet for eleverne, som skal tages med i de didaktiske overvejelser. Så det er vigtigt, at modellen er så simpel, at eleverne – på baggrund af deres faglige viden – umiddelbart kan forstå og arbejde med modellens interface. Og lige så vigtigt er det, at koden bag modellen er let tilgængelig for eleverne. Også for dem, der aldrig før har set computerkode. Begge dele er en forudsætning for, at eleverne kan finde ud af, hvordan de enkelte dele af koden styrer modellens opførsel.

En god NetLogo-model, skal derfor ikke nødvendigvis indfange alle kendte aspekter af et fagligt fænomen. Der kan opnås høj læring ved at lave en computermodel, der fokuserer på nogle få, centrale aspekter ved fænomenet, som eleverne rent faktisk har mulighed for at forstå og arbejde kreativt og konstruktivt med (også på kodeniveau). Denne tilgang har den styrke, at eleverne selv kan nå frem til erkendelsen, at computermodellen har nogle begrænsninger/mangler i forhold til det virkelige fænomen – f.eks. ved at blive bedt om at sammenligne computermodellen med en figur i bogen, en faglig tekst, hverdagserfaringer eller resultatet fra et eksperiment. Det kan give anledning til en sund faglig diskussion i klassen om modellens begrænsninger. En diskussion, der gerne skulle føre eleverne omkring de aspekter ved det faglige fænomen, som computermodellen ikke har med. I mange tilfælde vil det være muligt for eleverne efterfølgende at forbedre modellen i forhold til diskussionen af dens begrænsninger.

En diskussion af computermodellens begrænsninger kan også munde ud i den vigtige pointe, at den slags valg træffes i forbindelse med al modellering – også i andre fag og discipliner – men at det kan være svært at gennemskue, eller man glemmer det måske, når man præsenteres for en flot model, der fremstår professionel og kompleks.

For nogle lærere virker det paradoksalt, at man kan fremme elevernes faglige forståelse ved at præsentere dem for modeller, der er overdrevent forenklede. Men det er en (næsten) entydig erfaring fra flere hundrede afprøvninger af CMC-tilgangen i klasseværelset, at lærerne bagefter udtrykker, at eleverne kan komme frem til dyb faglighed ad den vej. Dermed bliver det, der set med faglige øjne i første omgang kan være problematisk ved en model – at den er overdrevent forenklet – pædagogisk set en styrke.

En let tilgængelig kode opnås ved, at koden enten er meget kort, indeholder få forskellige typer agenter og enkel adfærd, eller – hvis koden er længere – ved at organisere koden, så de første kodelinjer er nemme at læse og ændre i, mens mere komplicerede definitioner og relationer 'gemmes' længere nede i koden. I afsnit 12.1 giver vi nogle erfaringsbaserede råd og anvisninger til, hvordan man skriver og organiserer god kode. Generelt gælder det, at jo simplere det modellerede fænomen er, desto mere tilgængelig kan koden gøres for eleverne. Første trin mod at opnå en enkel kode, der er tilgængelig og til at forstå for eleverne, er altså at indsnævre det faglige fænomen tilpas meget.

For den, der undersøger de modeller, som ligger i NetLogos bibliotek (se afsnit 12), er det vigtigt at vide, at i de oprindelige ideer til, hvordan man kan bruge NetLogo i undervisningen, er det IKKE meningen, at eleverne selv skal forstå og ændre i koden. Derfor er mange af modellerne ret komplicerede. Også for komplicerede til, at man – som i CMC-tilgangen – kan sætte eleverne til at arbejde med koden. Det indgår simpelthen ikke i brugen af NetLogo på amerikanske skoler, at eleverne arbejder med koden.

Lærer i samfundsfag i fokusgruppeinterview, februar 2020:

"Jeg synes, det [dvs. denne måde at inddrage modeller i undervisningen] fungerer godt i forhold til det med, at eleverne får en kritisk sans over for modellering af verden.

Specielt som samfundsfagsunderviser, hvor der er alle mulige, der poster prognoser om, at hvis vi gør sådan her, så sker der det her. For eksempel den her rygermodel. Så skruer vi lidt på afgiften her, og hvad sker der så? Der er nogen, som er kommet med en hel masse forudsætninger i modellen, og journalisterne æder det råt, og det er det, som bliver historien. Men der er jo nogen, der har lagt de forudsætninger ind. Eller i de økonomiske modeller; er det en meget liberal model? Jo, det var det så. Det er sjovt nok, så har det ikke nogen særlig effekt, hvad man gør med uddannelserne.

Det er det, jeg i virkeligheden synes er det vigtigste ved det her. Det med at jeg faktisk kan gå ind og ændre i de her variabler, og det betyder noget for, hvordan det er, at den her model virker.
[...]
Alle de andre modeller [vi bruger i samfundsfag] er som en lukket boks, og det er det, der er forskellen ved det her. For her har eleverne selv mulighed for at sige: Ok, vi ændrer bare en lille bitte smule på det her tal, og så får det en anden effekt."

Det giver selvfølgelig et problem for den lærer, der henter en model i NetLogo-biblioteket. En måde at omgå dette problem er at tage en kompliceret model fra biblioteket og lave en forenklet udgave eller rydde op i programkoden (refaktorering) og gøre den mere 'elevegnet' (se afsnit 12.1).

I langt de fleste af de NetLogo-modeller, der er udviklet i forbindelse med CMC-kurser, er det tilstræbt, at koden skal være simpel og direkte tilgængelig for eleverne. Modellerne med tilhørende undervisningsforløb kan findes på https://library.ct-denmark.org.

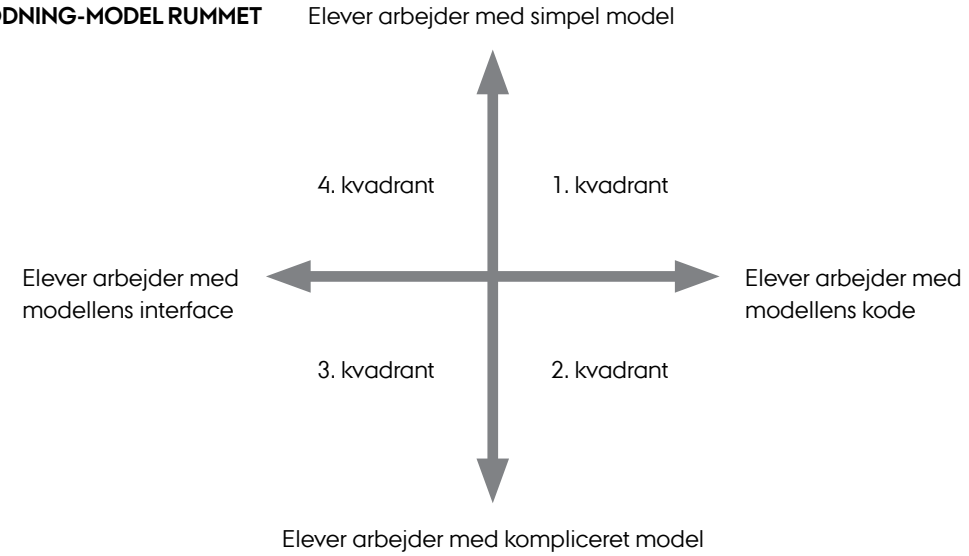
Introduktioner til faglige modeller skrevet i NetLogo:
- Transport gennem en cellemembran: https://www.youtube.com/watch?v=wvLQm_zXXmA&t=2s
- Reaktionshastighed i den kemiske proces: https://www.youtube.com/watch?v=LV4SR1X8uLo
- International handel: https://www.youtube.com/watch?v=yqlJV7qsbxg

Krav til en velegnet model:
Det skal være muligt for eleven at afstemme den interne, kognitive model med den eksterne computermodel gennem meningsfuld manipulation af den eksterne model ved at:
- åbne koden og læse den som et sprog, de næsten genkender
- tilegne sig forståelse for mekanismerne/agenternes adfærd
- forstå, hvordan mekanismerne interagerer dynamisk med hinanden
- forstå, hvordan manipulationer af systemet har indflydelse på systemets adfærd gennem disse dynamikker
- væve forståelserne sammen i kontinuerligt forbedrede kausalforklaringer på fænomenet.

KODNING-MODEL-RUMMET SOM PLANLÆGNINGSVÆRKTØJ
I forbindelse med CMC-tilgangen er der udviklet et redskab, der kan beskrive/kortlægge samspillet mellem, hvor simpel en model med tilhørende modelkode er, og så de måder, læreren instruerer eleverne i at arbejde med modellen. Vi kalder dette redskab til didaktisk kortlægning for kodning-model-rummet.

Den lodrettet akse henviser til **hvad,** eleverne arbejder med: Hvor simpel/kompleks er repræsentationen af det valgte faglige fænomen (modellen)? Den vandrette akse henviser til, **hvordan** eleverne arbejder: Bruger de NetLogos interface, eller arbejder de med modellens kode? (se figur).

KODNING-MODEL RUMMET



Ved en simpel model forstås her en model, som helt bevidst er lavet, så den kun omfatter et enkelt eller få aspekter af det modellerede fænomen.

1. kvadrant: Simpel model, eleverne arbejder med modelkoden
2. kvadrant: Kompleks model, eleverne arbejder med modelkoden
3. kvadrant: Kompleks model, eleverne arbejder med modellens interface
4. kvadrant: Simpel model, eleverne arbejder med modellens interface.

I sin planlægning kan læreren bruge kodning-model-rummet til at overveje og illustrere, hvordan progressionen i et undervisningsforløb kan/bør være. En oplagt mulighed er at starte undervisningsforløbet med at lade eleverne arbejde i NetLogos interface, hvor eleverne ser modellen simulere det aktuelle fænomen. I kodning-model-rummet svarer det til, at eleverne er til venstre for den lodrette akse. I det øjeblik, læreren beder eleverne åbne NetLogos skærm med kode, rykker de til højre. I afsnittet om Use-Modify-Create (se afsnit 8) beskriver vi, hvordan man trinvis kan rykke elevernes aktiviteter længere mod højre.

Afhængigt af hvad lærerens mål med undervisningsaktiviteterne er, kan læreren vælge at lade elevernes aktiviteter forblive over den vandrettet akse, eller – måske i et længere forløb – lade dem gå sydpå.

Fagligt vil læreren nok ønske hurtigt at komme sydpå: Eleverne skal præsenteres for modeller, som er 'gode' i den forstand, at de modellerer et givet fænomen så troværdigt og realistisk, og derved også ofte komplekst, som muligt.

Vi bemærker, at i traditionel undervisning i datalogi vil der være en tendens til at arbejde direkte med læring om programmering, altså at begynde til højre for den lodrette akse, og blive der.

Tilbagemeldinger fra kursisterne tyder på, at man med CMC-tilgangen opnår de mest tilfredsstillende resultater i sin undervisning, hvis man begynder i 4. kvadrant og planlægger sin undervisning, så eleverne kan arbejde sig nedad mod 2. kvadrant (mest vægt på CT-aspektet i undervisningen) eller mod 3. kvadrant (mest vægt på de faglige elementer) eller begge dele.

OPSUMMERING
Netlogo er velegnet til modellering, da det har følgende kvaliteter:

- Der er en meget lav forståelsestærskel for nybegyndere
- Programmet har visualisering indbygget (i Interface-skærmen)
- Programmet anvender agentbaseret modellering, hvilket gør det muligt at modellere komplekse fænomener ved hjælp af relativt simple antagelser om agenter og deres adfærd
- Programmet har indbygget feedback. Så snart der laves ændringer i koden, kan konsekvenserne iagttages i modellen
- Netlogo tilbyder et fælles sprog, så lærere kan afprøve hinandens materialer og udvikle et dansk praksisfællesskab.

NetLogo er dermed et værktøj til at arbejde med modellering og til at undersøge, hvilke interessante *makroskopiske fænomener der opstår ud fra agenternes mikroskopiske adfærd* (CCTD, 2018b), og dermed belyse sammenhængen mellem mikroskopiske og makroskopiske modeller (emergerende fænomener).

10. CMC I PRAKSIS – EKSEMPEL PÅ ET UNDERVISNINGSFORLØB

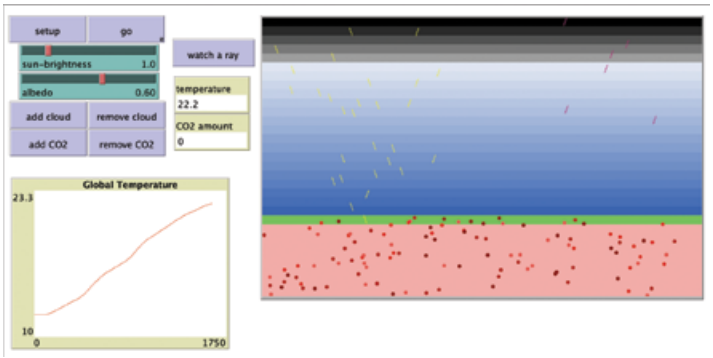
Vi gennemgår her et undervisningsforløb, hvor der arbejdes med en NetLogo-model af drivhusfænomenet og global opvarmning. I forløbet bruges CMC-tilgangen, og der er faglig og CT-mæssig progression gennem brug af use-modify-create. Forløbet indeholder også elementer, der omhandler meta-overvejelser om modeller og perspektivering fra den model, eleverne arbejder med, til andre typer af modeller, f.eks i samfundsøkonomi. Forløbet er beskrevet i større detalje i (Skadhauge, 2020).

Undervisningsforløbet hedder *Albedo, drivhusgas og strålingsbalance* og blev udviklet af to lærere i forbindelse med et kursus i *Computational Thinking i Matematik og Naturfag*, som blev gennemført i skoleåret 2018/19. Forløbet blev blandt andet brugt på lærernes egen skole i forbindelse med NV. Forløbet findes her: library.ct-denmark.org/albedo-drivhusgas-og-straalingsbalance.

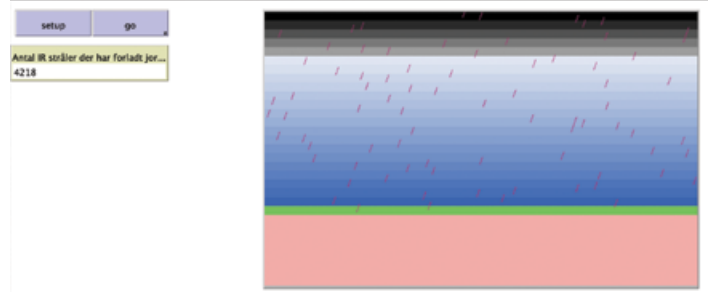
Forløbet demonstrerer, hvordan man som lærer kan udnytte de modeller, der ligger færdige i NetLogos bibliotek (se tekstboks om NetLogo s. 49).

Ca. halvdelen af eleverne valgte at bruge modellen med tilhørende simuleringer til eksamen i NV. De to lærere tolker det som et bevis på, at eleverne følte sig fortrolige med stoffet. Desuden brugte ca. otte af 14 NV-hold forskellige af simuleringerne i forløb op til en fællestime om FN's klimamodeller.

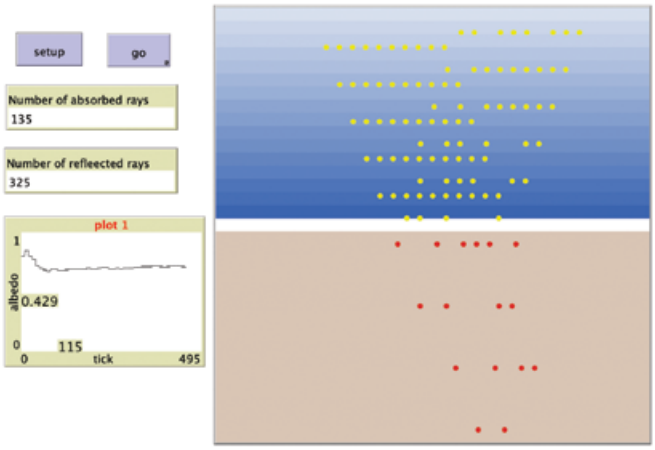
I forløbet arbejdede eleverne med tre forskellige modeller. Eller rettere, de arbejdede med tre forskellige udgaver af den samme model, hvoraf de to er forenklede udgaver af den oprindelige model, der hedder Climate_Change.nlogo. Modellen stammer fra NetLogos bibliotek, men er en dansk version, hvor bl.a. Info-filen er oversat til dansk.



Climate_Change.nlogo:
Modellen viser det samlede drivhusgas-fænomen. Den inkluderer solens indstråling, stråling/absorption i atmosfæren og Jordens albedo. I modellens interface kan man med skydere variere Jordens albedo og sollysets intensitet. Med knapper kan man tilføje eller fjerne skyer og CO_2. En graf viser jordoverfladens temperatur i modellens realtid.



Drivhusgas_model_Den__simple.nlogo:
Modellen viser kun Jordens udstråling gennem atmosfæren, fordi en række kommandolinjer i koden for den samlede model – altså modellen, som ses ovenfor – er gjort inaktive (slukket) ved at sætte ;; foran koden.



Albedo_model.nlogo:
Modellen viser 'kun' refleksion hhv. absorption af indfaldende lys. I interfacet tælles, hvor mange stråler der absorberes af jorden (og bliver til røde varmeprikker), og hvor mange de reflekteres. En graf viser variationen i Jordens albedo. Igen er der tale om, at forenklingen i modellen opnås ved, at nogle kommandolinjer i den samlede model er gjort inaktive.

Figur 11: I undervisningsforløbet arbejder eleverne med tre forskellige versioner af den samme model.

Inden forløbet går i gang, er eleverne blevet bedt om at installere NetLogo på deres computere. Eleverne arbejder sammen i parvise grupper.

Eleverne bliver først bedt om at hente filen *Climate_Change.nlogo* og parvis sammen læse noterne på Info-skærmen. Dernæst skal eleverne overveje og diskutere en række faglige spørgsmål om lys, bølgelængder, absorption osv.

Når eleverne har læst Info-skærmen, bliver de bedt om at skifte til Interface-skærmen i NetLogo og lege med modellen og gennemføre nogle af de aktiviteter, der er beskrevet i Info. Det indebærer bl.a., at nogle af eleverne prøver at lave små ændringer i koden ved at vælge Code-skærmen.

En lærer refererer fra en klasses arbejde med klima-modellerne:

"De [eleverne] sætter sig, og jeg tager den store model op igen, og vi snakker om, at jordoverfladen 'selvfølgelig' har en forkert farve. Jeg spørger dem, hvordan vi skal ændre farven. En af de dygtigste piger i klassen kommer op på projektorlyset og udpeger, hvor i koden der kan ændres.

Alle prøver det selv af, og en stor kreativitet udfolder sig nu, og mange forskellige farver kommer ind i jordens atmosfærer, flere end jeg havde forestillet mig. Jeg er rundt hos folk, og der opstår en sjov udveksling af meninger om de forskellige 'kunstværker'.

Enkelte 'crasher' deres kode; intet virker. Jeg sætter dem til at hente koden ned igen fra Lectio og starte forfra med 'små slag'; små ændringer, som de prøver af på farven."

(Mandlig lærer i fysik og matematik, 2019)

På et tidspunkt har eleverne prøvet så mange småting i Interfacet og diskuteret så meget parvist, at det giver mening at tage en samlet snak i klassen om spørgsmål knyttet til elevernes arbejde med modellen, såsom:

- Hvem fik den højeste temperatur?
- Hvem fik temperaturen stabiliseret til 37 grader? Hvorfor svinger den lidt op og ned?
- Hvem fik den flotteste farvesammensatte jord?
- Hvordan er modellen opbygget i overordnet forstand?
- Hvordan fungerer simuleringen af modellen?
- Hvad skal den vise?
- Hvad kan – og kan den ikke?
- Kender du andre simulerede modeller, som vi som medborgere bør kende opbygningen af?
- Kan man sammenligne den her modelsimulering med den, der bruges til at rådgive politikere om økonomi?
- Kan brugen, små ændringer og udviklingen af modellen her bidrage til, at du bliver kritisk og kreativ på en bedre måde i forhold til modelsimuleringen?

Eleverne bliver nu bedt om at åbne filen *drivhusgas_modellen_Den_simple.nlogo*. Modellen er forenklet så meget, at den kun viser pile (varmestråling) på vej op fra Jorden. Eleverne bliver bedt om at trykke på Code og finde de steder i filen, hvor koden er gjort inaktiv. En simpel opgave for eleverne kan være at finde den del af koden, som får pilene til at gå ned mod jorden og aktivere koden. Elever, der har fundet det rigtige sted i koden og tjekket, at pilene rent faktisk nu går begge veje, bliver opfordret til at snakke med deres kammerater om, hvad de har gjort.

Eleverne arbejder også med at tilføje 'molekyler' til atmosfæren og vurdere, om det er en rimelig del af strålingen, der nu reflekteres. De bliver bedt om at overveje, om modellen skal forbedres yderligere, og hvordan de i givet fald vil gøre det. Lærerens opfordring lyder, 'Prøv en lille ændring ad gangen, så kan du altid fortryde og ændre det tilbage'.

Endelig arbejder eleverne med at forbedre den forsimplede model for albedo (Albedo_model.nlogo), så den fagligt set bliver mere tilfredsstillende. Derefter kan de vende tilbage til den samlede model. De klasser, der vil arbejde videre med at skrive kode i NetLogo, introduceres til de Tutorials, der ligger i *NetLogo User Manual*, og som eleverne selv kan arbejde med (se tekstboks s. 49).

Efter afviklingen af forløbet konkluderede lærerne:

- Afprøvningen af simuleringen supplerede litteratur og eksperimenter.
- Den legende del af afprøvningen virkede motiverende.
- Elevernes diskussioner af simuleringens og modellens muligheder og begrænsninger brugtes til at arbejde mod NV's faglige mål om modellering.
- Små trinvise ændringer af koden med omgående afprøvninger får kodning til at virke.
- Ideer til – og gennemførelse af – ændringer i koden med henblik på at forbedre simuleringen sætter tanker i gang om kritiske vinkler til modellen, især hvis de diskuteres.

Udsagn fra lærere, der har gennemført undervisningsforløb i CT og modellering med computermodeller:

"Jeg kunne arbejde med et [fagligt] komplekst fænomen, som ellers ville kræve koblede differential-ligninger. Jeg kunne IKKE have lært dem det samme med en lærebog og PHET-simuleringer [fra nettet]."
(Mandlig fysiklærer, 2019)

"Det faglige udbytte behøver ikke at blive mindre, fordi man arbejder med CT. De to kan bringes til at understøtte hinanden. Aldrig har jeg set så mange elever, som faktisk har forstået, hvad strålingsbalancen betyder. Nogle vil måske hævde, at nye forløb altid vækker begejstring, fordi lærerens begejstring smitter af. Men første gang, jeg kørte forløbet, var jeg lidt af en skeptiker selv. Jeg har oplevet, at NetLogo faktisk danner en god ramme for indlæring af faglige og CT-kompetencer samtidigt."
(Kvindelig fysiklærer, 2020)

11. LÆRINGSMÅL I FORBINDELSE MED CT-I-FAG

I forbindelse med de kursusforløb, der er afholdt om CMC-tilgangen (eller som p.t. afholdes), er der oprindeligt formuleret en række mål for, hvad de deltagende lærere bør styre efter med hensyn til elevernes læring, når de deltager i CT-aktiviteter. I overensstemmelse med, at de tre områder – fagligheden, modellering og kodeforståelse – integreres i CMC-undervisning, skelnes der ikke mellem de mål, der er knyttet til hvert af de tre forskellige områder. Ligesom undervisningen er målene også integreret. Den konkrete integration af målene betyder dog ikke, at det ikke kan være af værdi for læreren at reflektere over, hvilken læring der – inden for rammerne af CMC-tænkningen – er knyttet til hvert af de tre områder: Det faglige, det modelleringsmæssige og det kodemæssige.

De faglige mål afhænger dels af det fag og det faglige fænomen, som læreren har valgt at inddrage, dels af hvad læreren i øvrigt ønsker af opnå af faglig læring. Principielt adskiller opstilling af de faglige mål i forbindelse med CMC sig ikke fra de faglige mål, en lærer i øvrigt arbejder med. Men læreren bør nok være opmærksom på, at andre lærere, der har arbejdet med CMC-tilgangen, ofte fortæller, at nogle elever lærer det faglige stof på en anden måde end i anden undervisning (Se desuden citat fra kvindelig fysiklærer, 2020, i boksen overfor).

Overordnet gælder, at det, at eleverne lærer at bruge computere til at udforske, forstå og forklare faglige problemstillinger, er et CT-mål. Ligeledes er det et CT-mål, at eleverne indser, at computere er redskaber til at løse problemer med, og at både data, kode og modeller i den sammenhæng er redskaber, som man kan lære at mestre. Hermed er både mål knyttet til kodning og til modellering specifikke CT-mål.

I forbindelse med de igangværende CT-aktiviteter arbejdes der løbende på at udvikle læringsmålene og specielt på at beskrive modelleringskompetencerne tydeligere for både lærere og elever.

Nedenstående seks mål er formuleret på grundlag af en lang række afprøvninger af CMC-forløb med tilhørende tilbagemeldinger fra de lærere, der har udviklet forløbene eller har afprøvet forløb, der er udviklet af andre. På den baggrund anser vi målene for realistiske i den forstand, at målene kan indfries i forbindelse med konkret undervisning. Selvfølgelig opfyldes ikke alle mål i lige høj grad i alle forløb. Lærere, klasser og undervisningsforløbene er forskellige. Men som en kort opsummering af, hvad der sker af læring i en bred vifte af forløb, er målene realistiske og bliver opfyldt.

Elever, der arbejder med CT-aktiviteter gennem CMC-tilgangen og computermodeller, bør blive i stand til at:

- downloade og installere relevant software
- benytte en computermodel som bruger (i Interface) og derigennem få en faglig indsigt i det modellerede fænomen
- lave simple ændringer i koden (som f.eks. at ændre form og farve på agenter).
- redegøre for, hvordan udvalgte dele af koden styrer modellens opførsel
- lave og begrunde fagligt relevante ændringer i koden, som ændrer på modellens opførsel (eksempelvis ved at ændre en talværdi eller ændre en underrutine).
- reflektere over modellens muligheder og begrænsninger i forhold til det modellerede fænomen.

I forbindelse med pilotprojektet, der gav os 'Proof of concept' for CMC-tilgangen, svarede 210 elever på et præ- og posttest-spørgeskema, der undersøgte, i hvor høj grad projektets læringsmål blev opfyldt. Denne undersøgelse viste, at overraskende store procentdele af eleverne svarede bekræftende på følgende udsagn:

- Jeg kan bruge og forstå modellen (90%)

- Jeg kan forstå sammenhængen mellem modellen og det faglige fænomen (74%)
- Jeg kan forstå sammenhæng mellem koden og interfacet (78%)
- Jeg kan lave meningsfulde ændringer i koden (79%).

(Musaeus & Musaeus 2019, s. 916)

Som supplement til ovenstående kan lærere opstille mere detaljerede mål, der specifikt retter sig mod eleverne modelforståelse eller deres arbejde med koden.

Overvejelser om mål knyttet til elevernes arbejde med koden kunne være:

- Hvor langt bør eleverne gå i fasen 'modify'. Hvordan? Hvorfor?
- Skal eleverne bevæge sig ind i fasen 'create'? Hvor går i det givne undervisningsforløb grænsen mellem modify og create?
- I hvor høj grad skal eleverne generalisere en algoritme fra at virke et bestemt sted i koden til at bruges mere generelt?
- I hvor høj grad skal eleverne kunne generalisere (evt. genbruge) essentielle egenskaber ved en samling af agenter?
- Hvor omfattende/komplekse/åbne problemer skal eleverne løse ved at arbejde med koden?
- I hvor høj grad - og på hvilken måde - kan elevernes arbejde med koden tilrettelægges, så det virker motiverende på flest mulig elever?

Overvejelser om mål knyttet specifikt til arbejdet med at modellere:

- Forstår eleverne, at regler for agenternes opførsel afbilder de årsags-virkningsforhold, der repræsenterer forståelsen af det faglige fænomen?
- Forstår eleverne, at modellering af et fænomen altid indebærer, at modelbyggeren træffer valg, der forenkler modellen i forhold til fænomenet?
- Kan eleverne overføre forståelse af årsagssammenhænge fra modellen til det virkelige fænomen og omvendt?
- Kan eleverne forstå sammenhængen mellem modellens mikroniveau og dens makroniveau?
- Kan eleverne dekomponere en model i 'undermodeller' eller delelementer?
- Kan eleverne analysere og identificere forskellige niveauer i en kompleks model?
- Kan eleverne overføre deres analyse af og forståelse af en given model til andre modeller, eventuelt i andre fag.

Udsagn fra lærer, som har afprøvet CMC-tilgangen:

'Det var helt tydeligt, at NetLogo-simuleringen hjalp eleverne med forståelsen af det faglige'.
(Kvindelig fysiklærer, 2019)

12. HVORDAN KOMMER JEG I GANG MED NETLOGO?

Der er forskellige måder at komme i gang med NetLogo. Vores erfaring er, at det er vigtigt at tilpasse denne proces til målgruppen, da lærere og elever kan have meget forskellige forudsætninger for at programmere og tænke computationelt. Vi har valgt at præsentere nogle tilgange, som er ordnet efter stigende programmeringskompetencer hos målgruppen.

Tilgangene kan være:

- Brug af modeller, som er udarbejdet af andre danske gymnasielærere, samt tilhørende beskrevne elevaktiviteter (se https://library.ct-denmark.org)
- Brug af eksempler fra NetLogo-biblioteket, som er inkluderet i NetLogo-programmet. Der findes en række gode modeller her, men desværre er de ofte meget komplicerede i koden. De er derfor mest velegnede til selvstudium og inspiration for lærere. De kræver, at man har opnået en vis fortrolighed med NetLogo. Brug af biblioteket forudsætter, at man har installeret NetLogo (se tekstboksen)
- Brug af de introducerende tutorials i NetLogo. Disse kan være udmærkede for personer med en vis erfaring i programmering, men kan være vanskelige for begyndere.
  Se https://ccl.northwestern.edu/netlogo/docs
- Producér egne modeller. Det kan f.eks. være aktuelt efter de introducerende tutorials, som er nævnt i punktet ovenfor
- Se/gennemfør eventuelt et helt videokursus om, hvordan man udvælger relevante fænomener at modellere, og hvordan man laver konkrete modeller i NetLogo.
  Se https://ccl.northwestern.edu/netlogo/docs.

NetLogo kan installeres fra https://ccl.northwestern.edu/netlogo. Når du åbner NetLogo, kan du under File og Models Library finde et bibliotek af modeller til forskellige fag. Brug af NetLogo er gratis.

Færdige undervisningsforløb udviklet af danske gymnasielærere kan findes på https://library.ct-denmark.org.

Introducerende tutorials på engelsk til NetLogo findes på https://ccl.northwestern.edu/netlogo/docs

To helt korte introduktioner på dansk til, hvordan man bruger NetLogo:

- https://www.youtube.com/watch?v=TMvxq6U_91w
- https://www.youtube.com/watch?v=eq5E3zXfLnk

12.1 GOD KODE I NETLOGO

I det følgende beskrives nogle retningslinjer for 'god kode', der tilgodeser behovene i CMC-tilgangen. Disse retningslinjer er på ingen måde fyldestgørende, men har status af 'work-in-progress'. De er baseret på erfaringer fra afprøvninger med elever og lærere både i Danmark og USA, samt på afprøvninger i forbindelse med et ph.d.-projekt, der bl.a. prøver at afdække danske gymnasieelevers kompetencer inden for det at læse, forstå og skrive kode i NetLogo. De er samlet af udviklere og forskere fra CCTD i nedenstående punkter.

God kode:
I CMC-tilgangen til undervisning i CT er der fokus på hhv. det faglige fænomen (Content), modellering af fænomenet (Modeling) og koden bag computermodellen (Code). Derfor er overvejelser om, hvordan man designer god kode til computermodellerne, så elever og lærere kan læse, forstå og ændre dele af koden, meget vigtige.

I det nedenstående er opsummeret nogle af de betragtninger og erfaringer, der er fælles for alle udviklere, forskere, kursusundervisere og kursister, der har været aktive i at designe kode til de bagvedliggende projekter. Nogle af punkterne lægger sig op ad de principper, som professionelle programmører følger for 'god programmeringsstil'. Andre skiller sig ud pga. didaktiske hensyn. Det er her vigtigt at forstå, at en kode, der er 'god' for en professionel programmør, ikke nødvendigvis er let at gå til for elever. Det er nogle helt andre hensyn, en trænet programmør, der laver programmer, der skal læses og vedligeholdes af andre professionelle programmører, skal tage, end en underviser, der vil have sine elever til at arbejde med kode – og måske endda kun et lille udsnit eller et særligt aspekt af koden. Målet er derfor ikke, at koden fremstår som en professionel programmør ville skrive den, men at den er skrevet og struktureret, så den understøtter elevernes forståelse af koden og faglige arbejde.

Generelt kan erfaringerne deles i fire kategorier: Syntaks, struktur, kommentering og overskuelighed/læsbarhed.

Syntaks:
- ☺ Brug selvforklarende navne til agenter, variabler og procedurer. F.eks. *enzymer, startantal-får, spigræs.*
- ☺ Brug bindestreger til navne. F.eks. *spis-græs.*
- ☺ Brug verber i procedurenavne (så det er nemt at forestille sig, hvad de gør). F.eks. *bevæg-molekyler, beregn-temperatur.*
- ☺ Brug altid den lange form af en kommando. F.eks. *forward* i stedet for *fd* og *clear-all* i stedet for *ca.*

Struktur:
- ☺ Undgå globale variable (hvis muligt). Kan f.eks. gøres ved at anvende *let* i stedet for *set* eller ved at bruge *turtles-own.*
- ☺ Hav altid procedurerne *setup* og *go* øverst i koden.
- ☺ Sørg for, at din setup-procedure og din go-procedure kan læses og forstås i sig selv uden et forudgående kendskab til modellen. Pak de enkelte ting ned i *underprocedurer.*

```
to go
  bevæg-molekyler
  kemisk-reaktion
  beregn-temperatur
end
```

- ☺ Gør én ting per underprocedure. Hvis en procedure er godt navngivet, giver det sig selv, hvad den ene ting er.
- ☺ Overvej de programstrukturer, der bruges. Det kan f.eks. nogle gange være lettere for eleverne at forstå to *if*-sætninger end en *ifelse*-sætning (men pas på: det er ikke altid, det kan gøres).
- ☺ Overvej at lave både en *go-once* og en *go-forever* knap (giver eleven en forståelse af løkken).
- ☺ Bottom-up: Gør agenterne tydelige en efter en (med deres egenskaber og dernæst deres opførsel).
- ☺ Lav agentbaserede modeller.

Kommentering:

☺ Lav kommentarer til hver procedure. Men husk: God kode behøver ikke lange kommentarer. Graden af kommentering afhænger af, hvad eleverne skal fokusere på og arbejde med.
☺ Kommentarer til koden laves ved at anvende dobbelt semikolon foran teksten. F.eks *;; Her skabes en ulv, der gives en størrelse, farve og form.*
☺ Lav evt. en introduktion til hele programmet, som giver mening for eleverne. Bør laves under 'Info'-fanen.
☺ Lav evt. kommentarer i linjer, som er særligt svære at forstå eller kræver særlig opmærksomhed fra eleven (f.eks fordi de er centrale for computermodellens opførsel, eller fordi eleverne skal arbejde med netop den linje).
☺ En velstruktureret og didaktiseret kode behøver relativt få kommentarer (men de kan ikke undværes).

Overskuelighed/læsbarhed:

☺ Brug tomme linjer til at opdele koden i 'afsnit'.
☺ Brug indrykninger (indentering) til at fremhæve kodens struktur. Det gør det nemmere at læse koden (kan klares ved at markere alt og trykke på *Tab*). F.eks.:

```
ask får [
  if any? ulve in-radius 1 [
    set color red
  ]
]
```

☺ Brug ekstra parenteser dér, hvor det letter læsbarheden. F.eks s*etxy (top + 5) (random 10)* i stedet for *setxy top + 5 random 10.*
☺ Overvej, om du skriver navne, kommentarer og interface på dansk eller engelsk. Men vær konsistent.
☺ Udfør kun én ting per linje. Det er lettere at forstå, hvis der kun er ét udsagn per linje. F.eks.:

```
create-turtles 5 [
  set size 5
  set shape "square"
  set color red
  setxy 10 5
]
```

er bedre end:

```
create-turtles 5 [set size 5 set shape "square" set color red setxy 10 5]
```

- ☺ Ryd op i koden (forskøn og forsimpl), når du er færdig med en model og planlægger undervisningsaktiviteterne, så koden passer til dine aktiviteter (*refaktorering*). Det gælder syntaks, struktur, kommentarer, overskuelighed og læsbarhed.
- ☺ Gør koden så simpel som muligt (blandt andet gennem fagligt enkle modeller).
- ☺ Overvej om kode skal gentages i forskellige underprocedurer, hvis det fremmer læsbarheden og forståelsen/sammenhængen.
- ☺ Fokuser på de dele af koden, som eleverne skal arbejde med. De dele skal måske 'foldes helt ud' i koden.

13. REFERENCER

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal*, 55(7), 832-835.

Baas, N. A., & Emmeche, C. (1997). On emergence and explanation. *Intellectica*, 25(2), 67-83. Lokaliseret 20/5-2020: https://www.nbi.dk/~emmeche/coPubl/97d.NABCE/ExplEmer.html

Buch, J., Etches, A., Nielsen, K., Musaeus, L. H., & Hjorth, A. (2019). Computational Thinking i Matematik og Naturvidenskab (CTiMNAT). Lokaliseret 14/5, 2020: https://cctd.au.dk/fileadmin/www.cct.au.dk/cctd.au.dk/PDFs/CTIMNAT-1-evaluering-fin3.pdf

Bundsgaard, J., Bindslev, S., Caeli, E. N., Pettersson, M., & Rusmann, A. (2019). Danske elevers teknologiforståelse: Resultater fra ICILS-undersøgelsen 2018.

Caspersen, M. E., & Nowack, P. (2013, January). Computational thinking and practice: A generic approach to computing in Danish high schools. In *Proceedings of the Fifteenth Australasian Computing Education Conference-Volume 136* (pp. 137-143).

Caspersen, M. E., Iversen, O. S., Nielsen, M., Hjorth, H. A., & Musaeus, L. H. (2018). Computational Thinking – hvorfor, hvad og hvordan?: Efter opdrag fra Villum Fondens bestyrelse.

Caspersen, M. E. (2017). Computational Thinking, kapitel 4.15 i. G*ymnasiepædagogik – En grund-bog*.

CCTD (2018a). Computational Thinking in High School subjects. *Center for Computational Thinking & Design*, Aarhus University. Lokaliseret 6/8, 2020: http://cctd.au.dk/projects/computational-thinking-in-high-school-subjects/

CCTD (2018b). CT i Gymnasiefag – Afsluttende rapport til Region Midtjylland /Regional Udvikling. *Center for Computational Thinking & Design*, Aarhus Universitet. Lokaliseret 16/6, 2020: http://cctd.au.dk/fileadmin/user_upload/CT_i_Gymnasiefag-afslutningsrapport-ur.pdf

Danske Gymnasier (2019) Digitaliseringsstrategi: De almene gymnasiale uddannelser i den digitale tidsalder. *Danske Gymnasier*. Lokaliseret 16/5-2020: https://issuu.com/rektorforeningen/ docs/digitaliseringsstrategi_1.0?fr=sM2I1YjI3NDQ2MQ

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, 60(6), 33-39.

DiSessa, A. A. (2018). Computational literacy and "the big picture" concerning computers in mathematics education. *Mathematical thinking and learning*, 20(1), 3-31.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, 42(1), 38-43.

Guy-Gaytán, C., Gouvea, J. S., Griesemer, C., & Passmore, C. (2019). Tensions Between Learning Models and Engaging in Modeling. *Science & Education*, 28(8), 843-864.

Guzdial, M. (2019). A new definition of Computational Thinking: It's the Friction that we want to Minimize unless it's generative. *Computing Education Research Blog*. https://computinged.wordpress.com/2019/04/29/what-is-computational-thinking-its-the-friction-that-we-want-to-minimize/

Guzdial, M., Kay, A., Norris, C., & Soloway, E. (2019). Computational thinking should just be good thinking. *Communications of the ACM*, 62(11), 28-30.

Hansen, J. Ø. (2020). Computational thinking i fysikfaget – henfaldsloven, *LMFK-bladet*, 1, 23-31.

Hansen, J.Ø., Peulicke, F., Jensen, A. & Skadhauge, S. (2020). Computational thinking i gymnasiefag, *LMFK-bladet*, 1, 11-18.

Honey, M., Pearson, G., & Schweingruber, H. A. (Eds.). (2014). *STEM integration in K-12 education: Status, prospects, and an agenda for research* (Vol. 500). Washington, DC: National Academies Press.

Iversen, O. S., Dindler, C., & Smith, R. C. (2019). *En designtilgang til teknologiforståelse*. Dafolo.

Jacobson, M. J., Kim, B., Pathak, S., & Zhang, B. (2015). To guide or not to guide: issues in the sequencing of pedagogical structure in computational model-based learning. I*nteractive Learning Environments,* 23(6), 715-730.

Jensen, A. (2020). Computational thinking i matematik – differentialligninger, *LMFK-bladet*, 1, 32-34.

Jespersen, J. (2018). Professor: Finansministeriets regnemodel dumper i økonomi. *Altinget*.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, 2(1), 32-37.

Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2020). Computational Thinking from a Disciplinary Perspective: Integrating Computational Thinking in K-12 Science, Technology, Engineering, and Mathematics Education. *Journal of Science Education and Technology*, 29(1), 1-8.

Musaeus, L. H. & Musaeus, P. (2019, February). Computational Thinking in the Danish High School: Learning Coding, Modeling, and Content Knowledge with NetLogo. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* , 913-919.

National Research Council (2014). STEM Integration in K-12 Education: Status, Prospects, and an Agenda for Research. Washington, DC: The National Academies Press. https://doi.org/10.17226/18612.

Nersessian, N. J. (2002). The cognitive basis of model-based reasoning in science. *The cognitive basis of science*, 133-153.

Nowack, P., & Caspersen, M. E. (2014, November). Model-based thinking and practice: A top-down approach to computational thinking. In *Proceedings of the 14th Koli Calling International Conference on Computing Education Research* (pp. 147-151).

Papert, S. (1980). Mindstorms: children, computers, and powerful ideas, Basic Books, New York.

Rapp, D. N. (2005). Mental models: Theoretical issues for visualizations in science education. In *Visualization in science education* (pp. 43-60). Springer, Dordrecht.

Resnick, M., & Rosenbaum, E. (2013). Designing for tinkerability. *Design, make, play: Growing the next generation of STEM innovators*, 163-181.

Skadhauge, S. (2020). Computational thinking i naturvidenskabelig grundforløb, *LMFK-bladet*, 1, 19-22.

Sun, Y., Newstetter, W., & Nersessian, N. J. (2006). Promoting model-based reasoning in problem-based learning. *Trabajo presentado en la reunión anual de la Cognitive Science Society*, Vancouver, Canadá.

Tedre, M., & Denning, P. J. (2016, November). The long quest for computational thinking. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research (*pp. 120-129).

Tisue, S., & Wilensky, U. (2004, May). Netlogo: A simple environment for modeling complexity. In *International conference on complex systems* (Vol. 21, pp. 16-21).

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127-147.

Wilensky, U. (1999). Center for connected learning and computer-based modeling. In *NetLogo*. Northwestern University.

Wilensky, U., & Rand, W. (2015). *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.

COMPUTATIONAL THINKING
I GYMNASIEFAG

Her gives et bud på, hvordan **Computational Thinking** (CT) kan inddrages i gymnasiets eksisterende faglige undervisning, så eleverne lærer faglige begreber, mens de arbejder med CT, og de lærer CT, mens de arbejder med faget. CT som et selvstændigt fag, som det f.eks. optræder i it-fagene i gymnasiets fagrække, behandles ikke.

I implementeringen af undervisning, der udvikler elevernes CT-kompetencer, lægger vi stor vægt på anvendelse af computermodeller og modellering. Dels fordi det giver anledning til at inddrage nye metoder og en ny it-faglighed i undervisningen, dels fordi det medvirker til at gøre fagene mere autentiske i en tid, hvor forskning i alle faglige discipliner inddrager computere og computermodeller. Vi foreslår således ikke bare en ny it-faglighed, vi udvider også fagenes eksisterende fagligheder.

CENTER FOR COMPUTATIONAL THINKING & DESIGN
AARHUS UNIVERSITET
DASG Danske Science Gymnasier
VILLUM FONDEN
EGAA GYMNASIUM
midt regionmidtjylland

ISBN 978-87-94078-00-9