Pedagogisk mappe Lex Nederbragt

onsdag, 03. mar, 2021

Table of Contents

1. Introduksjon

En pedagogisk mappe er ment til å dokumentere erfaringer og utvikling som underviser, og er relevant for ansettelse og merittering ved universiteter.

Denne pedagogiske mappen har en struktur som stort sett følger SoTL (Scholarship of Teaching and Learning) prinsippene.

Spesifisering av hvordan innholdet i denne mappen henger sammen med SoTL kriteriene

Fokus på studentenes læring

- Behandles i undervisningsfilosofien (Teaching Statement)
  - Kognitiv last teori og samkoding
  - Formativ vurdering og Peer Instruction
  - Samstemt undervisning
- Behandles i et eget kapittel
  - Motivasjon gjennom valg av relevante biologiske problemstillinger
  - Systematisk bruk av studentevalueringer

En klar utvikling over tid

- Behandles i undervisningsfilosofien (Teaching Statement)
  - Implementering av samkoding i BIOS1100
  - Samstemt undervisning i BIOS1100
- Behandles i et eget kapittel

En forskende tilnærming

- Behandles i et eget kapittel
  - Bruk av resultater fra forsning om læring i undervisningen
  - Forskning på egen undervisning

En kollegial holdning og praksis

- Behandles i et eget kapittel
  - Undervisnings Verksted ved Institutt for Biovitenskap
  - Undervisning i et felleskap med gruppelærere og studenter
  - Studenter som bidragsyttere i det faglige innholdet

Det digitale

- Bruk av læringsfremmede digitale midler er å finne i nesten alle kapitler
  - Pionér i bruk av interaktive brukergrensesnitt for programmeringsopplæring (Jupyter), som resulterte i en sky-basert løsning for flere kurs på UiO (JupyterHub)
  - Pågående prosjekt "Integrasjon av Jupyter og Canvas for digital vurdering"
  - Bruk av GitHub og MS Teams for interaksjon og deling av materialet med gruppelærere
  - Bruk av Student Respons Systemet Mentimeter for formativ vurdering og innhenting av besvarelser i etterkant

2. Hvordan jeg som underviser har utviklet meg

- ha med felelsdelen 2019 og kollegaveiledningen
- ha med CSE@IBV koordinator

Begynnelsen

Student og PhD student

Min første erfaring som underviser var da jeg var gruppelærer for et kurs i zoologi. Det ble en veldig fin opplevelse: mens studentene så et tversnitt av en kattenyre hver, så jeg mange flere når studenter ba meg om hjlep med deres. Jeg følte at jeg lærte en god del mer av å undervise faget, da da jeg tok faget selv.

Som PhD student fikk jeg mulighet til å bli med som underviser på det europeiske Erasmus kurset "Marine Cell Biology," som fant sted på et marin biologsk laboratorium i Frankrike (Observatoire Oceanologique, Banyuls-sur-mer, France). Forkergruppen min var en av de som sto for organisering av kurset, og mine veiledere ba meg å holde et par foredrag og bidra med labøvelser. Denne erfaringen fortalte meg at jeg faktisk likte å undervise, å stå foran en gruppe studenter og formidle min kunnskap til dem.

2005-2009

Jeg mener at disse erfaringene har bidratt da jeg fikk stilling som postdokter ved Biologisk Institutt (nå Institutt for Biovitenskap) ved Universitete i Oslo. Den jog jobbet for ba meg allered etter kort tid å bidra med noen forelesninger i kurset BIO2120 Evolusjonsbiologi om evolusjon og utviklingsbiologi (fagfeldet mitt som doktorgradstudent), og evolusjon av gener og genomer. Jeg ble også bedt å steppe inn en gang for å organisere en av labbene i BIO1000, det stor introduksjonskurset for nye studenter i biologi. Det var en ny erfaring å undervise unge studenter, noe jeg synes var spennende. Men, jeg var fortsatt opptatt av den klassiske undervisningsmåten med 45 minutter lange forelesnignen og ingen studentaktiviserende øvelser.

Som forsker byttet jeg imidlertid fagfelt i retning bioinformatikk og genomikk. Datasettene jeg jobbet med begynte å bli så store at de ikke lenger kunne bearbeides med Excel. Jeg måtte lære meg bruk a kommandolinje-baserte verktøy og supercomputere. I tillegg lærte jeg meg et programmeringsspråk (Perl). Alt dette trengte jeg for å delta i genomrevolusjonen som startet i 2006/2007, der nye instrumenter gjorde det mulig for mindre forskningsgrupper å sette sammen, og undersøke, genomer til diverse organismer.

Jeg merket fort at jeg trivdes med å videreformidle min nye kunnskap til mine kollegaer, som hadde samme behov. Jeg ga korte kurs for kolleger i bruk av de verktøyene jeg brukte mest, og i progammering.

INF-BIO5121/9121 (2012-2016)

Fra 2009 ble jeg med i Norwegian Sequencing Centre (NSC), en nasjonal forskningsinfrastruktur for DNA sekvensering. NSC bestemt seg i 2011 for å starte med kursing av sine 'kunder,' og satt opp et kurs med en generell introduksjon til sekvensering, og to paralelle praktiske kurs i bruk av bioinformatiske verktøy. Sammen med en kollegaforsker fra England organiserte vi en av disse praktiske kursene: en to-dagers workshop om teknikker for å sette sammen genomer, sokalt 'genom assembly.' Vi tok bevisst en veldig praktisk tilnærming, der deltakere, PhD studenter og erfarne forskere, fikk mye hands-on erfaring med ekte data og bruk av verktøyene. Dette var en

fantastisk erfaring: her følte jeg at deltakere virkelig lærte noe, og at de når de kom 'hjem' med en gang kunne bruke det de hadde lært. Dette var starten av en ny periode der jeg ble mer og mer overbevist om at denne hands-on læringen fungerer veldig bra, og er en god mulighet til å koble undervisningen veldig tett på pågående forskning.

Rundt denne tiden oppdaget jeg Software Carpentry, en internasjonal organisasjon av frvilige som holdt workshops om bruk av verktøy for forskere som bruker mye beregninger og programmering i sin forskning ('research computing'). Målgruppen var forskere som meg, som senere i karrieren sin oppdaget et behov for å kunne bruke disse verktøy, uten å ha fått lært dette i sin opplæring. Jeg så at tilnærming til Software Carpentry var grundig forankret i pedagogisk forskning, og at målet deres var å undervise kunnskap opparbeidet gjennom årene om 'best practices,' og forhindre å finne på hjulet på nytt. Jeg merket at jeg ønsket å delta på en slik workshop og tok kontakt. Resultatet var at deres direktør Greg Wilson kom til Oslo og holdt den første Software Carpentry workshoppen i Norden. Ikke bare lærte jeg mye om hvordan jeg kunne bli mer effektiv i min forskning, jeg bled - gledelig - rekrutert til å bli instruktør for Software Carpentry. Etter et (online) instruktør treningsprogram med stor fokus på pedagogikk, ble jeg og en kollega sertifisert får å kunne holde Software Carpentry workshops selv. Erfaringer med Software Carpentry endret mitt syn på undervisning totalt. Plutselig ble jeg bevisst på viktigheten av motivasjon for læring, det å begrense kognitiv last, formativ vurdering osv. Jeg så at jeg måtte tenke annerledes om undervisning, og forankre min tilnærming i forskning og kunnskap på feltet.

INF-BIO5121/9121

På denne tiden var det flere grupper på UiO som organiserte korte kurs eller workshops rundt analyse av DNA sekvenseringsdata. Alle sammen ble de bedt av det som på denne tiden var Computational Life Science initiaivet, en satsning fra fakultetet for å fremme bioinformatikk forksning og undervisning, å laget et poengivende kurs på master og PhD nivå basert på disse kursene. Vi ble enige å slå oss sammen til et 5 poengs intensivkurs der studentene fikk en generell introduksjon til DNA sekvensering og data analyse, etterfuglt av moduler om anvendte analyser tilsvarende våre opprinnelige workshops. I 2013 ble jeg tilbudt en 20% 1. amanuensis II stilling ved Institutt for Informatikk for blannet annet å være kursansvarlig for dette kurset. Kurset har blitt en suksess og tiltrakk seg mange studenter, flest biologer, men alltid noen informatikere. Jeg tok selv min allerede utbiklede assembly modul inn i kurset. Fra og med 2014 endret vi kurset til en 10 studiepeongskurs, siden vi opplevde at mange studenter droppet eksamen fordi de bare fikk 5 studiepoeng ut av det.

Dette var min første erfaring som kursansvarlig (emneansvarlig), som var noe nytt for meg. Kurset var er samarbeid mellom undervisere, med meg som leder, og jeg lærte mye om å drive fram et slikt prosjekt. Vi brukte studentevalueringer akitvt med et egetutviklet nettskjema som studentene fylte ut etter siste kursøkt. Dette førte til at jeg besluttet etterhvert å bytte ut noen av undervisere, som fikk veldig dårlig tilbakemelding fra studentene. Jeg synes det var en vanskelig, men helt nødvendig beslutning å ta, og det fikk det ønskede effekten med bedre læringsutbytte for studentene for den aktuelle modulen. Erfaringen som emneansvarlig var positiv, og førte til en bedre forståelse for hele kursets løp, fra forberedelse, gjennføring, eksamen og evaluering. Jeg fikk også bedre innsikkt i studentene perspektiv på å ta unversitetskurs.

Våren 2014 tok jeg fellesdelen av Universitetspedagogisk Basiskompetanse. Jeg skrev et utviklingsarbeid om hvordan å øke studentdeltakelse i undervisningen i INF-BIOS5121/9121 kurset. Dette fikk jeg gode tilbakemeldinger på. Siden jeg var ansatt i en 20% 1. amanuensis II stilling trengte jeg ikke å gjenomføre hele fellesdelen og jeg valgte da bort kollegaveiledningen.

Jeg fortsatte å involvere meg i The Carpentries. Vi fikk til et samarbeid med realfagsbiblioteket ved UiO. dette førte til at vi kunne tilby flere workshops. I 2016 ble jeg instruktør trener for the Carpentries. Dette innbærer at jeg kan lære opp nye instruktører. Til det har The Carpentries utviklet et instruktør trenings opplærings materialet som er forankret i noen elementære pedagogiske forskning (pedagogisk psykologi) rundt hvordan vi lærer. Sentrale begreper er motivasjon, kognitiv last, novise-ekspert, mentale modeller, tilbakemelding og summativ versus formativ vurdering.

BIOS1100 (2017-)

I 2017 ble det introdusert nye bachelorprogrammer i alle studieprogrammer ved det matematisk-naturvitenskapelige fakultet ved Universitetet i Oslo. Alle programmene inkludert fra da av et beregningsaspekt fra første semester. På Institutt for Biovitenskap ble dette implementert ved et nytt introduksjonskurs BIOS1100 - Innføring i beregningsmodeller for biovitenskap, som er obligatorisk for alle biovitenskapelige studenter. Videre skulle andre kurs, både obligatoriske og fordypningskursene, impelementere beregningsperspektivet som en del av undervisningen.

Jeg var ikke involvert å prosessen som førte til disse endringene. Men jeg var, og er, sterkt overbevist om alle biologer bør skaffe seg beregningskompetanse. Arbeidet med The Carpentries viser at mange ferdigutdannede biologer mengler denne kompetansen. Jeg er selv et godt eksempel på det.

Da jeg i 2016 fikk tilbudet å ta emneansvar for BIOS1100 og utvikle kurset fra første semester det ble undervist, takket jeg gledelig ja. Dette var en fantastisk mulighet til å utvikle et unikt kurs som skulle forberede studentene til å være framtidens biologer.

Det viste seg at dette ansvaret var mye større enn jeg trodde, men også har vært veldig givende. BIOS1100, og jeg som underviser, har gått gjennom en utrolig utvikling de første fire gangene det ble undervist. Nedenfor beskriver jeg først selve kurset, og så utviklingen av det i kort trekk.

Om BIOS1100

BIOS1100 - Innføring i beregningsmodeller for biovitenskap, underviser studenter i enkel (matematisk) modellering, implementering av disse modellene i programmeringsspråket Python mens det hele tiden fokuseres på problemer som er relevante for studenter i biovitenskap. Fokuset på biologi tar sikte på å sikre at studentene ser relevans av det som blir undervist, noe som er viktig for studentens motivasjon og læring. Biologiske problemer som for esksempel poplasjonsvekst og -dynamikk, arv, DNA-analyse og sykdomsepidemier brukes til å gradvis innføre mer kompleks programmering og modellering.

Kurset er basert på en ny lærebok skrevet av fire tidligere doktorgradsstudenter i nevrovitenskap fra Biologisk Institutt som alle har bakgrunn innen fysikk, samt meg som emneansvarlig.

Kurset består av:

- To timers forelesnigner
- Fire timers gruppeundervisning, obligatorisk fra og med 2018
- Fra og med 2019 to timers ikke obligatorisk samkodingstimer
- Flere obligatoriske innleveringer gjennom semesteret
- Fire timers avsluttende digital eksamen

Kurset og kursboken har blitt urviklet fra grunnen av med oppstart i 2017. Siden dette er et unikt kurs var det lite materialet utover kursboken som kunne brukes til å bygge kurset på. Kurset har gjennomgått en stor utvikling fra den første gang det ble undervist i 2017. Herunder beskriver jeg noen av aspektene der det har vært størst endring, motiverer endringene og beskriver effekten på studentenes læring.

Undervisningsform

Da jeg planla undervisningen for første gang kurset skulle bli undervist i 2017 valgte jeg å formidle til studentene at jeg forventet at de brukte pensumboken og øvelsene i gruppetimene for å tilegne seg programmeringskunnskappen. For å støtte opp under deres læring la jeg inn mye formativ vurdering (kombinert med Peer instruction) med hjelp av flervalgsoppgaver i forelesninger og gruppetimene. Bakgrunnen for dette er beskrevet i undervisningsfilosofien under avsnittet formative_assessment.

Jeg gikk også gjennom noen utvalgte oppgaver i forelesningene. Det viste seg 2/3 del inni kurset at mange studenter ikke kom på forelesningene lenger, og at deltakelse på gruppetimene, som på det tidspunktet ikke var obligatorisk, også var lavt. I tillegg fikk studieseksjonen gjennom samtaler med studenter inntrykk av at mange hadde falt av tidlig og ikke klarte å komme tilbake igjen.

Jeg bestemte meg da å ta grep og bytte ut den planlagte undervisningen for de tre siste ukene med et tilbudt til studentene som trengte det med økter med en telknikk som heter samkoding, Samkoding, eller 'Participatory Live Coding' er beskrevet utfyllende i undervisningsfilosofien under asvnittet samkoding. I korte trekk er dette en teknikk der en undeviser skriver kode og programmer sammen med studentene. Teknikken, som er mye brukt av The Carpentries, er ment å senkte terskelen for å komme i gang med programmeringen. Jeg tenkte derfor at den kunne egne seg for BIOS1100. Men, jeg var redd for at det ville være vanskelig å skalere opp til det forventede antallet av 200 studenter. Utover samkoding fikk studenter en del 'mengdetrening' øvelser der jeg repeterte all Python stoff som hadde blitt undervist. med fokus på forståelse av programmeringen

60 til 70 studenter følget denne ekstraundervisningen. Tilbakemeldinger fra studenter på dette var overveldende positivt, og en god del fortalte meg og studieseksjonen etterpå at de nå følte at de hadde forstått det vesentlige med programmeringen. Interessant nok meldte noen studenter skuffelse over at de siste kapitlene ikke ble undervist og ba om tilgang til dem.

Suksessen med samkoding som undervisningsform førte til at jeg bestemte meg å bruke samkoding for å undervise elementære kunnskap i Python gjennom hele kurset. Jeg mente, og mener fortsatt, at dette ikke lar seg gjøre med veldig store grupper studenter. Et av målene med samkodingen er at alle studenter følger, og at de skal kunne få hjelp når de sitter fast slik at de ikke faller av underveis. I en stor gruppe kan det føre til at en student holder opp undervisningen for hele gruppen. Derfor bestemte jeg meg at samkodingen i BIOS1100 måtte dette skje i gruppetimene. For å muliggjøre dette satset jeg på at dette skulle undervises av gruppelærere.

Andre gang kurset gikk i 2018 ble dette implementert. Samkoding ble brukt som hovedundervisningsform i gruppetimene. Til det hadde jeg skrevet en 'oppskrift' for hva som skulle undervises i de ukentlige samkoding sesjoner som gruppelærere kunne bruke. Jeg gjennomførte en egen opplæring for samkoding med 7 av gruppelærere (alle PhD studenter) modellert etter opplæringen The Carpentries gjør i sin instruktør trening.

Erfaringer underveis og besvarelser av sluttevalueringen viste at veldig mange studenter mente samkodingen i gruppetimene hadde stor nytteverdi: 94% mente dette var 'bra' eller 'meget bra.' Gruppelærere uttrykte også at samkoding som undervisningsform fungerte bra. Som en

gruppelærer som selv hadde tatt kurset i 2017 uttrykte det 'grunnmuren i programmering satt mye bedre' og at studenter følte de fikk mye ut av gruppetimene. Dessverre tok opplegget for mye av tiden i gruppetimene og det ble for lite tid til overs for selvstending jobbing med med (mer komplekse) oppgaver. Studenter fikk heller ikke utviklet effektive problemløsningsstrategier, de lette heller etter liknende oppgaver for å kopiere tidligere løsninger.

Erfaringen fra de to første semestrene med BIOS1100 viste altså at det i 2017 var for lite fokus på grunnopplæringen i programmering, mens det i 2018 har vært for mye fokus på det, på bekostning av jobbing med selvstendig problemløsning av større oppgaver.

Fra og med 2019 ble derfor det som ble undervist med hjelp av samkoding delt i to:

- Samkodingstimene: et frivillig tilbudt der nytt Python stoff ble undervist
- Gruppetimene: for å vise hvordan man løser oppgaver ('worked examples') og om nødvendig for å forklare ukens stoff på nytt eller på en annen måte

I tillegg ble det i planlegging av gruppetimene lagt opp til å ikke ha mer enn to timer organisert arbeid, slik at det ble to timer til overs for selvstendig jobbing med stoffet og oppgaver, med gruppelærer til stedet.

Disse grepene var vellykket. Det har blitt betydelig mer tid i gruppetimene tils selvstendig oppgavejobbing mens det samtidig kunne tilbys samkoding om elementær Python stoff. En utfordring er å få de studentene som trenger det faktisk til å gå på samkodingsgruppene.

Programmering

Boken som ble (og blir) skrevet for kurset har som mål å undervise Python *i kontektst av* en biologien Det vil si at den biologske problemstilling blir gitt først og deretter blir det forklart nytt Python stoff som trengs for å kunne løse prolemet. Dette fører til en jevn spredning av stoffet over semesteret. Da BIOS1100 ble undervist for første gang i 2017 viste det seg at den opprinnelige versjonen av kursboken som vi brukte hadde et problem. To måter å gjøre omtrent samme ting i Python ble undervist i to påfølegende kaptiler, og dermed i to påfølgende uker i kurset. Dette hanldet om det å samle en rekke verdier i en liste-struktur, enten med hjelp av vanlige lister i Python, eller med såkalte 'Numpy arrays.' Her ble studentene veldig forvirret, de to strukturere og operasjoner de skulle utføre med dem lignet veldig på hverandre, og det var ikke tydelig når og hvorfor de skulle bruke den ene heller enn den andre. Med andre ord, å undervise disse to lignende ting så kort oppå hverandre overbelastst studentenes kognitiv last. En annen ting jeg oplevde studentene strev med var vanskeligheten av konseptet 'løkker,' noe som er vesentlig i programmeringsspråk, som for eksempel Python. Løkker lar deg repetere noe flere ganger, eller gjøre noe med alle elementer i en liste av ting. Også her er det to forskjellige måter å gjøre dette på (som heldigvis ble undervist på vidt forskjellige tidspunkter i kurset). Jeg ble overbevist om at vi underviste disse formene i feil rekkefølge. Den såkalte 'while løkke,' som opprinnelig ble undervist sist, er mer transparant og gjør det mer klart for studentene hva som skjer i hver steg, enn den såkalte 'for løkke' som kurset startet med.

Jeg brukte dermed en del tid før kurset skulle starte i 2018 for å skrive om kursboken: bytte om rekkefølge av løkketype ('while' før 'for') og utsette bruk av den andre liste-strukturen (Numpy arrays) til siste del av boken. Effekten var at jeg opplevde framgangen i Python som mye smidigere. Det ble ikke rapporert om så mye forvirring som før.

Matematikk undervisning

Studieprogrammet som ble erstattet med biovitenskapsprogrammer i 2017 hadde et oligatorisk 10 studiepoeng mattekurs i første semester. Jeg har fra starten av ønsket at BIOS1100 skulle undervise en del av denne matematikken. I 2017 ble det lite tid til fokus på matematikkundervisning, i tillegg til at jeg selv ikke har tistrekkelig kompetanse på dette området. Som en løsning foreslo jeg at en professor fra Matematisk Institutt ved fakultetet skulle undervise matematikk i BIOS1100. Forslaget fikk gjenomslag og Professor Arne Sletsjøe har siden 2018 undervist matematikk i tre uker i kurset. Eksamen fikk også en egen matematikk oppgave.

Matematikkundervisningen utviklet seg fra å bruke en pensumbok, BioCalculus, til et eget mattekompendiet til BIOS1100 skrevet av Arne Sletsjøe. Vi testet flere varianter av når i kurset matematikken undervises, og har landet på at en todeling, med undervisning i uke 4 og 5, og uke 12, fungerer best. Vi oppdaget at en del gruppelærere slet med å kunne hjelpe studentene med matteoppgavene, noe som også studentene meldte tilbake om. Derfore ble ny 1. amanuensis på institutt, Øystein Langangen, fra 2020 involvert i opplæring av gruppelærere for å hjelpe studenene bedre. Evalueringen viste dette fungerte veldig bra.

En utfordring som jeg ikke har funnet en god nok løsning for er at matematikken oppleves av flere studenter som et eget fag som ikke henger sammen med resten av kurset.

Obligatoriske innleveringer og eksamensform

Utviklingen av de Obligatoriske innleveringer er beskrevet i undervisningsfilosofien under avsnittet obliger. Hvordan eksamensform har endret seg er også beskrevet der, under avsnittet eksamen.

3. Min undervisningsfilosofi (Teaching Statement)

Om denne undervisningsfilosofien

Denne teksten er skrevet til å leses som et selfstendig dokument, uavhengig av det øvrige innholdet i denne pedagogiske mappen. Dermed er det nok noe overlapp mellom denne teksten og de øvrige kapitlene. I tillegg er teksten skrevet på engelsk siden jeg ønsker å dele den med internasjonale kollegaer.

Introduction

My approach to developing my teaching is based on the observation that education is its own science, and a conviction that we should take the results of that science seriously when we develop courses. Just as prior research informs us when we develop our own research, so should educational research inform us when we develop our own teaching.

Educational research is a vast field and there is a lot to learn. But, one can not use all there is to learn, one has to limit oneself and focus on a few areas at the time. For me, these areas are i) Cognitive Load Theory, ii) Formative Assessment and Peer instruction, and iii) Constructive Alignment. In the following, I will introduce what I have learned from these areas and describe how I have implemented this in my teaching. I will focus on a new course that I have developed in the period since 2017, BIOS1100 – Introduction to computational models for Biosciences. When developing, and improving the course, I increasingly tried to incorporate what I have learned, and am learning, from educational science. This will be described in the following three sections.

Cognitive load theory and the need to manage cognitive load

Theoretical background

Knowing some of the research of how humans, and especially university students, learn can (and should) inform us on how to organise, plan and execute our teaching. The study of mental processes, including learning, is called Cognitive Psychology. Part of this field is concerned with cognitive load theory.

Learning theory starts from acknowledging that humans have two memory systems: *working memory* and *long-term memory*. Working memory is where new information is processed, and coupled to pre-existing information present in long-term memory. It is said that learning happens if this new information is transferred to long-term memory. While long-term memory can contain vast amounts of information, working memory is considered small, and has a capacity of about "seven plus or minus two" pieces of information [1]. Cognitive load theory is "a set of learning principles that deals with the optimal usage of the working memory" [2]. This theory, as defined in a recent review on the subject [3], "aims to explain how the information processing load induced by learning tasks can affect students' ability to process new information and to construct knowledge in long-term memory." The theory argues that the limited capacity of working memory severely restricts how much new information can be processed at any one time. When too much is asked from this working memory, there is a risk of overloading it, hampering learning. Overloading working memory inhibits the effective transfer of new knowledge to long-term memory, which is required for learning. It is argued that instructional methods need to take these limits into account.

What does this mean for BIOS1100?

Learning how to program is an important part of the teaching in BIOS1100. Learning to program is generally considered difficult [4] [5] [6]. Reducing cognitive load for students then becomes an important goal. I have always felt students can not learn programming from looking at a slide presentation of programming concepts, and then asking them to start programming themselves.I have experienced this approach myself at some time and it did not work for me, nor did it seem a useful approach. One reason for this is that this approach would result in a large cognitive load: students would be required to retrieve the factual knowledge presented during lecture and apply it to solve complex problems without any guidance on how to approach the problem. There is thus a need for alternative ways of teaching the fundamental building blocks of a programming language. One that is more based on Guided Instruction [7].

My main technique for reducing cognitive load when learning programming is called Participatory Live Coding.

Managing cognitive load in teaching programming: Participatory live coding

Participatory live coding is a guided instructional technique "in which a teacher or instructor writes and narrates code out loud as they teach and invites learners to join them by writing and executing the same code" [8]. The instructor reads what is being typed out loud, explaining the different elements and principles. Teacher and students all execute the commands or program, leading to an immediate evaluation of the results - hence the term 'participatory.' Crucially, the session contains regular, often short, exercises, where students are asked to solve a small relevant problem on their own or in pairs or small groups.

I am convinced that participatory live coding can help reduce this cognitive load for students learning programming. This approach works better than lecturing about programming, or relying on students reading a textbook or compendium. What is taught is immediately applied and the execution of the program being written provides immediate feedback. This helps student couple programming code with its result. Students' questions arising during the session can immediately be addressed. During this form of guided instruction, students are shown not only what to program and how each element works, but also how to program, i.e. how to go from a problem formulation to a working solution (the thinking process). It also slows the teacher down relative to using slides to show the concepts and code, giving students more time to actively engage with the material before moving on to the next concept. Interrupting the live coding with exercises enables immediate practice using the material.

The main drawback of using this technique is that it takes time to develop appropriate material and to teach it. Additionally, it does not scale too well as a single student with a problem that keeps them from following along and that takes some time to fix may hold up the entire group. Finally, students may think that they learn enough by simply following along. They should be made aware that to properly learn how to program they should practice, for example by doing exercises.

Participatory live coding in BIOS1100

The first edition of BIOS1100 (2017) relied on the students using the textbook for learning the programming concepts needed each week. Programming was not taught during lectures for reasons described above. Oppgaves were handed out during group work where students could practice applying programming to simple biological problems. Two-thirds into the semester it became clear that a significant group of students did not master the Python programming, and were in danger of failing the course. I then decided to not introduce any new Python material, but instead offer some

extra teaching using Participatory Live Coding. I had learned this technique from being an instructor for Software Carpentry. Software Carpentry, now part of the global non-profit called The Carpentries. The Carpentries, "teaches researchers the computing skills they need to get more done in less time and with less pain" and is mostly aimed at researchers at the PhD and postdoc level. Participatory live coding is the main method of teaching in the two-day workshops, and it is part of the training and assessment for becoming a certified instructor [9]. In BIOS1100, I thus offered sessions re-teaching the Python material with Participatory Live Coding, with the effect that many students reported that they now finally understood it.

I had previously considered using Live Coding for BIOS1100, but felt it would not scale to such a large group. The experience in 2017, and the student's feedback, convinced me that I had to find a way to adapt the technique of Participatory Live Coding to a course with 200 students, starting from the 2018 edition of BIOS1100. I decided that it should be taught in small groups, not with the entire group of students, and that I could not teach it all by myself. My solution was to develop a completely new set of teaching materials to teach Python programming in BIOS1100 using Participatory Live Coding, and train Teaching Assistants to be able to teach using this technique. One of the things taught during Instructor Training workshops for The Carpentries is Participatory Live Coding. I am a certified Instructor Trainer for The Carpentries, meaning I already had taught Participatory Live Coding to incoming instructors. I reused the material developed by The Carpentries [9] to train enough Teaching Assistants so that they could teach Python using PLC during the group sessions.

The results were that students in 2018 had a much greater confidence in Python programming. To start learning programming using Live Coding helped them overcome the initial barrier (sometimes fear) of programming, and led to a feeling of mastery early on. Students really loved the Live Coding ('samkoding' in Norwegian), some reported they learned the most there. A drawback of this approach was that much, sometimes all of the 4 hour group sessions were used doing Live Coding with the students. Too little time was left for students working on their own with exercises. A master student who studied the BIOS1100 students for his project that year concluded that students lacked good problem solving skills because of this [10].

In other words, while in 2017, we did not help students enough with learning programming, in 2018 we helped them too much and did not challenge them enough to apply what they learned.

From 2019 onwards, the Participatory Live Coding was moved to voluntary sessions, two hours each week. Students new to Python were strongly encouraged to participate. In group sessions, organised activities led by Teaching Assistants were limited to the first two hours, which left two hours for students working on their own solving problems. During the first two hours, some Live Coding was done to further explain concepts, or for so-called worked examples. Worked examples "provide a full problem solution that learners must carefully study" [3], and are another technique for reducing cognitive load.

Participatory Live Coding was what made introducing Python programming to new Bioscience students possible. I believe this technique can be used in many settings where students who traditionally do not have to learn programming are introduced to it.

Formative Assessment

Theoretical background

It would be a mistake to assume that students have learned the thing you just presented to them. Formative assessment is concerned with informing both the teacher and the learner about how

much they understand about a topic, and discover any misunderstandings. This allows for addressing misunderstandings promptly, which helps learners to progress through the material. Formative assessment is not graded, although sometimes teachers make participation count towards being able to pass a course. Note that graded assignments are what is called part of a course's summative assessment. Ideally, formative assessment can be done quickly and in an easy way for students and teachers to participate and see the results.

There are many forms for formative assessment, but a very common one is multiple Choice Questions. A well-designed Choice Question poses a problem with one or more correct, and multiple incorrect answers, so-called distractors. Ideally, distractors should not be too obviously wrong, but rather be indicative of possible misconceptions.

Formative assessment combined with Peer Instruction

Originally created by Eric Mazur at Harvard [11], Peer Instruction is an evidence-based method where students are actively discussing the material amongst themselves based on prompts provided by the teacher. By asking students to reflect on course material together in their own words, a student that just understood the material is able to explain it in a way that matches better a student that almost understands the material, than the way a teacher would explain it. Thus, *Peers* are *Instructing* themselves.

Often, Peer Instruction is combined with formative assessment through Multiple Choice Questions. There are different approaches on how to do this, but commonly, a Multiple Choice Question is posed and students consider the different answers for themselves. They then vote for the answer they think is the correct one. The tally of votes is shown, without revealing the correct answer. If there is a spread of answers among the correct one and one or more of the distractors, students are asked to discuss the question in pairs or small groups. They then vote again (individually, not as a group). More than often, the results show many more students converging on the correct answer. If needed or desired, the teacher can go over the different answers and explain how they are right or wrong.

Peer instruction has also been studied in the context of programming, with positive results on student learning [12].

What does this mean for BIOS1100?

Also in BIOS1100 there is a need to investigate student learning and check for misunderstandings. This can be partly achieved by Teaching Assistants helping students during group work, and by studying the obligatory assignments students hand in during the course. But, also the technique of Peer Instruction through Multiple Choice Questions is an ideal addition to this.

Teaching Assistants are a vital resource for student learning. In BIOS1100, they have the most direct contact with students (during the 4 hours group sessions and the non-compulsory live coding sessions). This means that they have a lot of experience that could inform me as a teacher about student progress and misunderstandings. The challenge is then how to ensure this information reaches me as a teacher, in other words, how to implement formative assessment through Teaching Assistants.

Practical implementation

I use the following Formative Assessment techniques in BIOS1100:

Multiple Choice Questions and peer Instruction

I have written a set of around hundred Multiple Choice Questions for BIOS1100. Some of these are based on common misconceptions I found in the scientific literature. Often, when I observe students displaying a misconception, or are told about one by the Teaching Assistants, I use that as inspiration for writing new ones.

I have used Multiple Choice Questions, through the online Student Response System Mentimeter, during lectures and group work. I always combine this with Peer Instruction, using the approach described above. I regularly observe a mix between questions that are 'too simple' (a large majority of students get it right at the first try) and questions that reveal a misunderstanding in a significant proportion of students, that then gets largely resolved in the group discussion. Students and Teaching Assistants really like the "Menti's," as they are fondly called. It is an easy form of Active Learning that helps create a dynamic group session or lecture. A drawback is that executing Multiple Choice Questions take time. I usually use no more than 4 questions, and those easily take up half an hour or more. Ideally, I would be able to always see the tally of votes for all questions to be able to filter out the ones that are too easy for next time, but is it challenging to collect that data from all Teaching Assistants.

It is fairly straightforward to adopt Multiple Choice Questions to an online teaching setting, provided the platform used allows students to work in small groups (for example, in so-called Breakout Rooms that tools like Zoom offer). In my experience, this works best if the students in these groups know each other from before, otherwise it is much more challenging to get them to discuss the question.

Obligatory assignments

In BIOS1100, students hand in 5 obligatory assignments spread evenly throughout the course. These are meant to ensure students work with the material throughout the course. The assignments are deliberately modeled after exam problems, so as to help prepare students for the exam. These "Oblig's" are graded pass/fail by the Teaching Assistants, and students have to pass 80% of them to be able to take the exam. When students fail on their first attempt for an assignment, they get two additional chances.

In the first edition of BIOS1100, not only were there 11 assignments, I intended these to be only used for summative assessment. I had attended a presentation where it was argued that one should not try to combine formative with summative assessment as assignments can not effectively serve these two purposes. So I instructed the Teaching Assistants to grade them without leaving any comments. It quickly became clear that it made much more sense to give students feedback on their assignment, regardless of whether they had passed or not. Having the option to hand in some of the work students doing, and receiving constructive comments on it, is very useful for student learning. Using the obligatory assignments for this is really a very good way for *all* students to receive such feedback, and one of the few ways to offer this to all students throughout the course. I thus concluded that I deliberately want to use the obligatory assignments for both formative and summative assessment. From then on, I asked Teaching Assistants to give feedback to all delivered assignments.

Ideally, I as a teacher would also study the deliveries to distill common misunderstandings, in other words, use them as a proper formative assessment tool for myself. Unfortunately, there has as of yet not been enough time during the course to do this.

Constructive Alignment

Theoretical background

Constructive Alignment is concerned with aligning the learning activities with the intended learning outcomes. Following Biggs [13], we start with determining:

1) What are the desired learning outcomes, these are the objectives
2) How to measure whether desired learning has been achieved, assessment
3) What (teaching and learning) activities can we use that engage students in a way that leads to learning

Biggs calls this Constructive Alignment, which is an aligned system of instruction where "the objectives define what we should be teaching; how we should be teaching it; and how we could know how well students have learned it" [14].

What does this mean for BIOS1100?

Following Biggs' approach, we should start by determining our objectives.

The learning outcomes of BIOS1100 are described on the course homepage (English). But since these are deliberately kept short, there is a need to expand on them.

The next step would be to determine the summative assessments, in other words, make exam questions and any obligatory assignments before the course starts. In practice, this is not often done.

Finally, we need to design appropriate activities that help new students learn the mix of biology, mathematical modelling and programming that BIOS1100 aims to teach.

Practical implementation

Jupyter Notebooks and JupyterHub

To align how we teach programming In BIOS1100, we use Jupyter Notebooks for everything. Jupyter Notebooks are a form of 'computational' notebooks combining text, media, programming code and the ability to execute that code and include the results of running it in the same notebook. Teaching materials, including exercises, are delivered as Jupyter Notebooks. During Live Coding sessions and group work,students as well as teachers and Teaching Assistants , do all their programming in them. Obligatory assignments are also handed out, and handed in and graded, as notebooks.

When students, assistants and teacher all are seeing and commenting code in the same environment, discussing problems and helping each other, this reduces the extra cognitive load of switching programming environments.

We use a cloud-based server, called JupyterHub, to provide students with this programming environment. An additional benefit of this programming environment is that it saves the students from installing software on their own laptop: as long as they have a working internet connection they can log in (using university credentials) to the server and start working.

Exam

In the first two years of BIOS1100, however, there was one component that did not use these notebooks. During the exam the students had to use a different environment which did not allow them to test and run their code and programs. The four hour exam was done in a UiO's digital exam environment Inspera, which did not have a technical solution for running programming code. This was initially a deliberate choice. In dialogues with my colleagues who teach beginner programming courses at the Department of Informatics, I became convinced that not being able to run code during the exam is of benefit for the students. Informatics students until recently even handed in their programs during the exam on paper. It prevents them from getting stuck with a relatively minor error (in the syntax, for example), and having to spend a lot of time fixing it - which can be hard. I thus decided to follow the same strategy, the motivation for which I also explicitly explained to the students.

In dialogues with students it became increasingly clear to me that not being able to test and run programs they write during the exam caused a lot of stress for the students. They had not experienced this way of working during the entire semester. I also learned more and more about the benefits of Constructive Alignment, and concluded that the exam introduced a major mis-alingment in the course. I thus decided that the situation needed to change.

Thus, in the fall of 2019, with help from the university's IT department and the faculty's Inspera team, it became possible, for the first time for any digital exam at the university, for students to submit exam assignments in the form of Jupyter Notebooks. This means that the students could now also run and test their code before handing it in. This led to a much increased Constructive Alignment between teaching and examination methods.

A large drawback to this approach is that students have to work in two different systems during the exam, the Inspera system and the JupyterHub browser. There is a risk of uploading the wrong notebook or the wrong version of it. Experience so far has shown students that are able to manage this satisfactorily. In 2020, the exam was changed to a 4-hour digital home exam, but otherwise organised as before. Incidentally, this aligned the exam situation even more than the 2019 exam, as it allowed students to use all available resources, as they are used to when working with (obligatory) assignments.

Conclusion

After several years of continued development, I am very satisfied with BIOS1100. i have found the right ‘form’ for this course, with lectures, Participatory Live Coding sessions and group sessions. The material written for the course, the set of problem exercises and Multiple Choice Questions is of sufficient quality and amount.

Along the way, educational theory has informed me for the many choices a teacher has to make. It has made me choose methods to lower cognitive load, for example by successfully scaling up Participatory Live Coding. It has shown me the importance of formative assessment, how there are different ways to be informed about student progress and how collecting this information can guide course adjustment immediately, or in between course editions. Finally, it has led to a much better alignment of the way students are exposed to programming, and work with it, with the summative assessment, by making the exam situation as similar as possible to the rest of the activities in the course.

Students also report more satisfaction with the course now than in the beginning. In the student evaluation of 2019, for the first time the words feeling of mastery (‘mestringsfølelse’) appeared in the open questions.

I plan to continue to explore new areas in educational science to make adjustments in my teaching. I am convinced that using it as a basis for trying out things is the most fruitful way forward.

4. Fokus på studentenes læring

My av det som faller under 'Fokus på studentenes læring' er beskrevet i min undervisningsfilosofi (kapittel 'Teaching Statement'):

- Bruk av samkoding som undervsiningsform for å redusere kognitiv last når studenter lærer programmering
- Bruk av formativ vudering kombinert med Peer Instruction for å sjekke forståelse og rette opp misforståelser
- Bruk av tilbakemeldinger på obligatoriske innleveringer for å kunne hjelpe studenter med å forbedre seg faglig
- Fokus på samstemt undervisning i utforming av obligatoriske øvelser og eksamensoppgaver, samt hvordan eksamen gjennomføres

Jeg begrenser med i dette kapittelet dermed til

XX

andre aspekter.

Valg av problemstillinger og motivasjon

Utfordringen med BIOS1100 er å motivere studentene å lære programmering og modellering i biologi.

Et vesentlig poeng med hvordan BIOS1100 er organisert er at, så mye som mulig, materialet som brukes er biologisk relevant. Dette har med studentene motivasjon å gjøre: de fleste valgte biovitenskspogrammet ikke fordi de er interessert i å studere programmering og modellering, men fordi de vil studere biologi. Som et eksempel mener jeg, når eksponensiell vekst undervises, at vi bør unngå å regne på renter, men heller studere eksponensiell bakteriell vekst.

Koblinger med BIOS1110.

Evaluering og systematisk utvikling av undervisningen

- felles studentrepresentanter med BIOS1110
- studentevalueringer
- evalueringer med gruppelærere
- 'tilbakemeldinger' fra lektorstudenter som tar undervisningsrettet master
- utskifte undervisere inf-bio5121 basert på tilbakemeldinger fra studenter

5. Bruk av forskning i undervisningen

Bruk av resultater fra forsning om læring i undervisningen

Som nevnt i kapittelet om min undervisningsfilosoi anses det å lære å programmere anses generelt som vanskelig [4] [5] [6]. For et motargument (og et utmerket sammendrag av forskningen som støtter denne posisjonen), se imidlertid [15]. Å lære et programmeringsspråk er litt som å lære et nytt skriftspråk: man må ikke bare lære seg syntaksen, men også hvordan man kombinerer ord for å gi meningsfulle setninger ('grammatikken'). Når man programmerer fører ofte relativt små feil til feilmeldinger, som attpåtil ikke er skrevet for nybegynnere (Denny et al., 2014). Kode som kjører, men som ikke gir forventet eller korrekt resultat, dvs. har 'bugs,' er vanskelig å rette - feilsøking er en ferdighet i seg selv (McCauley et al., 2008). Å be en student om å 'skrive et dataprogram som gjør X' fra bunnen av, krever mange ferdigheter samtidig: å vite hvilke kodingskonstruksjoner og datastrukturer som skal brukes, å kunne bruke syntaksen for dataprogrammeringsspråk, å kunne designe og teste programmet, og sørger for at løsningen faktisk løser problemet som blir adressert (dvs. 'X'), for et problem som kan kreve kunnskap fra et bestemt domene (f.eks. biologi når det gjelder BIOS1100-studentene mine). Det er trygt å si at det er lett å komme til en situasjon hvor mengden informasjon som må behandles samtidig er for mye for en nybegynner å takle. Det å lære å programmere fører med andre ord raskt til det som i kognitiv psykologi kalles høy kognitiv last.

Teorien om Kognitiv Last, og hvordan den brukes i utviklingen av undervisningen min, er beskrevet i undervisningsfilosoi under avsnitter {kognitiv_last}:

Forskning på egen undervisning

Det nye bachelor studieprogrammet Biovitenskap, som startet høsten 2017, er unik i at det integrerer programmering i hele studieløpet. Første semester møter studentene BIOS1100 som gir en innføring i å lage og eksperimentere med enkle modeller av biologiske systemer.

Det er per dags dato lite forskningsbasert kunnskap om hvordan biologistudenter stiller seg til programmering i biologi og enda mindre hvilke læringsstrategier de bruker for å løse biologiske problemstillinger med programmering. Vi er derfor i en unik posisjon til å fremskaffe slik kunnskapp ved å 'forske på' studenter i BIOS1100.

Som et første steg i å samle kunnskap på dett området har kollega Tone Gregers ved IBV tatt initiativ til å tilby lektorstudenter som tar en undervisningsrettet master til å skrive sin oppgave om studenter som tar BIOS1100.

Jeg har vært medveileder for alle fire studenter. Veiledningsteamet har vært tverrflaglig, med utdanningsforskere fra fysisk institutt (Maria Vetleseter Bøe og Ellen Henriksen) en underviser fra skolelaboratoriet ved Institutt for Biologi (Tone Gregers) og meg som emneansvarlig i BIOS1100.

Formålet med disse prosjektetene har vært å undersøke studenters holdninger og motivasjon for programmering og modellering i biologi, samt finne ut hvordan studentene velger å løse ulike biologiske problemstillinger ved hjelp av programmering. Gjennom prosjektet har vi ønsket å bidra til bedre læring og motivasjon i beregningsorientert biologi.

Funnene i masterprosjekter har jeg brukt aktivt for å forbedre kurset, både faglig og hvordan emnet undervises. I tillegg har dette arbeidet blitt presentert på MNT konferansen, en nasjonal konferanse om erfaringsdeling av en forskningsbasert og vitenskapelige tilnærming til undervisning og læring etter SoTL (Scholarship of Teaching and Learning) prinsippene, i 2019 og 2021. Det har også det

blitt skrevet en vitenskapelig artikkel om hvordan lektorstudenter som tar en undervisningsrettet master ved instituttet bidrar til undervisningskvaliteten [16], og om resultatene av to av deres prosjekter [17].

Nedenfor gjøres kort rede for prosjektene, funnene, og hvordan disse har blitt intergrert i videreutvikling av BIOS1100.

June Edvarda Eliassen 2020

- Tittel: Biologistudenters motivasjon for beregningsorientert biologi etter innføring av krav om full fordypning i realfaglig matematikk
- Problemstilling: Hvordan var motivasjonen hos studentene for programmering og modellering i biologi etter innføring av krav om full fordypning i realfaglig matematikk?
- Metode: spørreskjema som ble delt ut første og siste forelesning i 2019
- Teoretisk rammeverk brukt i studiet: Eccels' forventningsverdi teori om prestasjonsorienterte valg; Hidi og Renningers fire-fasemodell for interesseutvikling; Banduras mestringsforventningsteori
- Lenke til oppgaven: http://urn.nb.no/URN:NBN:no-82918

Kort sammendrag:

- Resultatene indikerer at matematikk R2 ser ut til a henge sammen med storre mestringsforventning og storre interesse for BIOS1100
- Det ser ut som om mange av studentene valgte bort full fordypning i biologi til fordel for R2
- Funnene støtter antagelsen om at nytteverdi er den sterkeste motiverende faktoren for emne
- Nytteverdien må ses i lys av at emne er obligatorisk for studieprogramet, og at studenter eller har lite interesse for programmering og modellering i biologi

Anbefaling for BIOS1100:

- Å spille på relevansen for å skape større interesse for programmering og modellering i biologi

I 2020 ble blant annet dette implemenert da vi tok i bruk data fra den pågående korona pandemien for å modellere sykdomsutbrudd.

Sofie Rudberg 2020

- Tittel: Relevansen av kompetansen fra matematikk R2 i beregningsorientert biologi
- Problemstilling: Hvordan gjør kompetansen fra R2 seg gjeldende i problemløsningsstrategier hos studenter i arbeid med beregningsorientert biologi?
- Metode: observasjoner av studentene i situasjoner med oppgaveløsning, og intervju, i 2019
- Teoretisk rammeverk brukt i studiet: Computational thinking, mathematical thinking
- Lenke til oppgaven: http://urn.nb.no/URN:NBN:no-82936

Kort sammendrag:

- Studentene er stort sett positive til at det har blitt R2-krav på utdanningen, og anser de strategiske kunnskapene fra matematikken som relevante
- Studentene benytter seg av problemløsningsstrategier kjent fra matematikk og computational thinking

- Studentene har derimot utfordringer med å løse mer sammensatte oppgaver, og benytter seg ikke av problemløsningsstrategiene kjent fra matematikk og computational thinking når de arbeider med mer tradisjonell biologi

Anbefalinger for BIOS1100:

- Kompetansen fra matematikk R2 er relevant for studentene på biovitenskap, men undervisningen bør legge opp til mer trening innen ulike problemløsningsstrategier

I 2020 ble det brukt en del tid på å gi studentene en forståelse av at det er nyttig å ha en strategi for å løse problemer i faget. Dette er fortsatt et forbedringspunkt i kurset.

Marthe Mjøen Berg 2019

- Tittel: Studenter sin interesse og mestringsforventning for programmering og modellering i biologi
- Problemstilling: Hva kjennetegner studenter sin interesse og mestringsforventning for programmering og modellering i biologi?
- Metode: spørreskjema som ble delt ut første og siste forelesning i 2018
- Teoretisk rammeverk brukt i studiet: Eccels' forventningsverdi teori om prestasjonsorienterte valg; Hidi og Renningers fire-fasemodell for interesseutvikling; Banduras mestringsforventningsteori
- Lenke til oppgaven: http://urn.nb.no/URN:NBN:no-73633

Kort sammendrag:

- Studentene har høyere mestringsforventning og interesseverdiene for studieprogrammet sitt, enn for BIOS1100
- Studentene får gjennom semestret noe høyre mestringsforventning til BIOS1100, mens interessen for emnet går litt ned

Anbefalinger for BIOS1100:

- bevisstgjøre studenten av relevansen for programmering utenfor emnet BIOS1100
- tilby ekstra faglig støtte til studentene som har lav mestringsforventning

Flere grep som ble gjort fra 2019 prøver å koble faget sammen med de andre fag studentene tar. Endringer i undervisningsform bidrar til mer faglig støtte.

Lars Erik Håland 2019

- Tittel: Studenters arbeid med programmering i biovitenskapelige problemstillinger. En kvalitativ studie av biologistudenters arbeid med Python
- Problemstilling: Hvordan arbeider studenter når de programmerer i introduksjonskurs i biologi?
- Metode: observasjoner av studentene i situasjoner med oppgaveløsning, og intervju, i 2018
- Teoretisk rammeverk brukt i studiet: Computational Thinking
- Lenke til oppgaven: http://urn.nb.no/URN:NBN:no-73632

Kort sammendrag:

- Studentene opplever de samme typer utfordringer som man ser i andre introduksjonskurs i programmering: det er utfordrende å skrive store programmer og finne gode problemløsningsstrategier
- Det biologifaglige virker å ha positivt effekt på deres evne til å løse programmeringsoppgaver, samtidig som at det gjør oppgaven vanskeligere, fordi det krever at de både må bruke biologi- og programmeringskunnskaper
- Bruk av biovitenskapelige oppgaver fører også til at programmeringen blir mer interessant for de fleste studentene.

Fra og med 2019 ble det å undervise problemløsningsstrategier mer og mer et foksu i BIOS1100.

6. Kollegialt samarbeid om utvikling av undervisning

Underverk

I 2019 tok kollega Tone Gregers ved IBV initiativ til UndervisningVerksted ved IBV, også kalt Underverk. UnderVerk er "en workshopserie der alle som er involvert i undervisning ved Institutt for Biovitenskap skal få mulighet til å arbeide med alle aspekter av undervisning i et kollegialt felleskap." (fra siden om UnderVerk, krever login med UiO brukernavn og passord, se også denne siden).

Sammen med kollega Hans-Petter Hersleth har jeg vært medorganisator for UnderVerk under Tone Gregers ledelse. Med støtte fra MatNat fakultetets studiekvalitetsmidler har vi organisert flere workshops. Noen av temaene har vært

- Hvordan å skrive LæringsUtbytte Beskrivelser ('LUB'er)
- Læringsmål og baklengsdesign
- Praktisk implementasjon av pedagogisk teori i undervisningsplanlegging
- Studentaktive læringsformer
- Bruk av digitale verktøy
- Vurderingsformer

Under korona pandemien vår 2020 organiserte Underverk et ukentlig treff i Zoom for erfaringsutveksling.

Mitt bidrag har vært å hjelpe med planlegging og gjennomføring, samt bidra konkret til noen av workshoppen:

- Kognitiv last teori (oktober 2019)
- Bruk av Zoom for digital undervisning (mars 2020)
- Digital hjemmeksamen i BIOS1100 (november 2020)

UnderVerk oppleves som veldig nyttig av undervisere ved instituttet. Det er varierende oppmøte, men deltakere synes altid dat har vært verdt å delta. Jeg opplever UnderVerk som et fint tilbud for å dele erfaringer.

Undervisning i et felleskap med gruppelærere og studenter

Gruppelærere

Jeg ser på gruppelærere som undervisere. Det er de som har ofte har flest kontaktimer med studentene gjennom semesteret. Derfor synes jeg det er flere viktige aspekter med å jobbe med gruppelærere:

- Jeg skal hjelpe de til å bli gode undervisere
- De skal oppleve at de jobber i et kollegialt felleskap
- Jeg skal aktivt involvere dem i å forbedre undervisningsopplegget

For å oppnå disse målene har jeg to tiltak: pedagogisk opplæring, og bruk av et erfaringsdokument underveis i kurset.

Pedagogisk Gruppelæreropplæring

Det første året BIOS1100 skulle undervises hadde jeg en sesjon med alle gruppelærere for å forberede dem på deres rolle, før sommerferien, det vil si en del uker før kurset skulle starte. eg opplevde at det det mye informasjon til dem lang tid før det skulle brukes, og at utbytte av dette møte derfor var lavt.

De siste årene har jeg brukt de ukentlige møter med gruppelærere for å styrke grppelæreres pedagogiske kompetanse. Hver uke tar jeg opp et tema som er pedagogisk relevant, eller inviterer en annen person til å presentere et slikt tema. I disse sesjonene bruker vi aktive lærings teknikker som tenk-par-del for å fremme diskusjon blant gruppelærere samt vise nytte av slike undervisningsteknikker.

Temaene vi vanligvis diskuterer:

- Introduksjon til rollen som gruppelærer
- Lærinsgmål og baklengs undervisningsdesign
- Motivasjon (student *og* underviser)
- Kognitiv last - hvorfor er det viktig å ta hensyn til det
- Noviser og eksperter
- Mentale modeller

Noen av disse teamaene er inspirert av The Carpentries instruktør trening, som jeg underviser selv.

Etter at jeg innførte denne måten å lære opp gruppelærere i noen pedagogiske begreper mm har jeg intrykk av at nivået på deres undervisning, og måten de samarbeider med studentene på, har blitt forbedret. Det er som om de tar jobben mer seriøst når de forstår at det er en vitenskap bak det å lære og undervise. Jeg tror det også gir dem intrykk av at jeg tar deres rolle på størst alvor.

Dette tiltaket har noe overlapp med det såkalte 'LA programmet' prosjektet ved fysisk institutt på fakultetet. Under dette konseptet organiseres det aktiviteter for å hjelpe gruppelærerne til å bli bedre undervisere gjennom pedagogisk opplæring og ukentlige utviklingsmøter. Jeg håper at LA programmet kan komme til Institutt for Biovitenskap i framtiden og skal da tilby å slå dette sammen med mine aktiviteter på området, slik at det gagner flere gruppelærere på instituttet.

Erfaringsdokument

Det er flere gruppetimer per uke i BIOS1100, og det er nyttig at de som har undervist en gruppe tidlig i uken deler sine erfaringer med de som skal undervise senere. Helt fra begynnelsen av kurset har jeg derfor brukt et delt dokument (en Google Doc) hvor jeg ba gruppelærere om å notere sine erfaringer umiddelbart etter undervisningen. De andre gruppelærere kan da lese dette såkalte erfaringsdokumentet og bruke det for å hjelpe dem med å forberede seg på undervisningen. Jeg bruker dokumentet for å umiddelbart rette opp feil i undervisningsmaterialet, eller foreslå løsninger når ting ikke fungerer som jeg hadde tenkt dem. Dette har fungert veldig bra, og mye nyttig tilbakemelding, eller inspirerende forslag, har blitt skrevet ned av gruppelærere gjennom årene.

Å bruke et slikt dokument kan ansees for en form for formativ vurdering. Det viser også gruppelærere at deres tilbakemeldinger blir tatt på alvor, og at deres innspill til kurset er veldig velkommen. Det gir dem eierskap i undervisningen, noe de også gir meg som tilbakemelding under kursevalueringene mot slutten av semesteret.

Når jeg forbereder meg til neste års undervisning i kurset, bruker jeg dette dokumentet, og et privat 'erfaringsdokument' som jeg skriver selv under semesteret, som et viktig hjelpemiddel for å

forbedre undervisningsmaterialet ytterligere og justere undervisningsøkter der det er nødvendig. Disse dokumentene viste seg å være viktig for å forbedre kurset.

Studenter som bidragsyttere i det faglige innholdet

CCSE, Centre for Computing in Science Education, har i alle år, som et ledd i fakultetets studiekvalitetsmidler, tilbudt stillinger som sommerstudenter. Formålet med prosjektene til disse 'CSE' sommerstudenter har vært å utvikle undervisningsmaterial til kurs der beregninger inkorporeres. Jeg har søkt, og fått tildelt, totalt 12 slike stillinger. Disse studentene har bidratt med mer enn 100 oppgaver, og en stor del av disse brukes i undervisningen. Det har vært en stor glede å kunne jobbe med disse studentene. Disse vet ofte selv veldig godt hvordan det er å lære programering som biolog, og er veldig kreativ når det kommer til å finne gode biologiske problemstillinger studentene kan jobbe med.

7. Dokumentasjon

Universitetspedagogisk kompetanse

- Jeg tok *Felles innføringsdel* for Universitetspedagogisk Basiskompetanse i 2019
  - Kursbevis: **vedlegg** `Kursbevis_universitetspedagogisk_basiskompetanse.pdf`
  - Utviklingsnotat: **vedlegg** `Utviklingsnotat_Fellesdelen_2019.pdf`
- moduler tatt i tillegg til Fellesdelen:
  - Forskningsveiledning (2014)
    - Utviklingsarbeid: **vedlegg** `Utviklingsarbeid_Forskningsveiledning_2014.pdf`
  - Forelesning og undervisning - det dramaturgiske aspekt (2015)
  - Pedagogisk mappe (2020)
- Jeg tok *Felles innføringsdel* i 2014 som 20% 1. amanuensis II
  - Utviklingsarbeid: **vedlegg** `Utviklingsarbeid_Fellesdelen_2014.pdf`

En klar utvikling over tid

Emneansvar

- BIOS1100 (2017-)
  - Kurstittel: **Innføring i beregningsmodeller for biovitenskap**
  - Emnesidene: https://www.uio.no/studier/emner/matnat/ibv/BIOS1100/
  - 2017
    - Semestersidene
    - Fil: `Sluttrapport BIOS1100 Høsten 2017.pdf`
  - 2018
    - Canvas sidene
    - Fil: `Sluttrapport BIOS1100 Høsten 2018.pdf`
  - 2019
    - Canvas sidene
    - Fil: `Sluttrapport BIOS1100 Høsten 2019.pdf`
  - 2020
    - Canvas sidene
    - Fil: `Sluttrapport BIOS1100 Høsten 2020.pdf`
  - Pensumbok "Introduction to Analysis and Modeling in Biology with Python," siste versjon (august 2020)
  - Jeg har skrevet noen blogposter om BIOS1100: http://lexnederbragt.com/bios1100
- INF-BIO5121/9121 (2012-2016)
  - Kurstittel: **High Throughput Sequencing technologies and bioinformatics analysis**
  - Emnesidene: https://www.uio.no/studier/emner/matnat/ifi/INF-BIO5120/index-eng.html
  - Semestersidene for høsten 2016: http://inf-biox121.readthedocs.org/en/2016

Annen universitetsundervisning

- MBV-INF4410/9410 Bioinformatics for Molecular Biology 2013-2016
  - Forelesning "The bioinformatics of sequencing and assembling genomes"
  - Forelesning "What does it mean to do bioinformatics?"
- BIO9905MERG1 - Bioinformatics for Metagenomic Analyses and Environmental Sequencing (2011)
  - Forelesning "Next Generation Sequencing techniques and data relevant for metagenomics analyses"
  - Forelesning "Assembly of metagenomes"
- BIO2120 Evolusjonsbiologi 2006-2007
  - Forelesning "Evolution and Development"
  - Forelesning "Evolution of Genes and Genomes"
  - Oppgaver for gruppearbeid

Workshops

- Next-Gen Sequence Analysis Workshop 'week 3' (intermediate and advanced skills) (invitert), Michigan State University 2015
  - Websidene
- University of California Davis Assembly Masterclass (invitert) 2013
  - Websidene
- Norwegian Sequencing Centre course: High Through-put Sequencing: technology basics, applications and bioinformatic analysis 2011
  - Websidene
  - Jeg gjennomførte en Workshop om Genome Assembly
- *De novo* genome assembly (invitert), Univ. of Gothenburg, 2011
- Erasmus ICP course Marine Cell Biology (Observatoire Oceanologique, Banyuls-sur-mer, France) 2000
  - Forelesning "Fundamental aspects of development"
  - Forelesning "Cell cycle changes during development"

Software og Data Carpentry

- Software Carpentry
- Data Carpentry
- fra og med 2018 ble disse sammen til The Carpentries
- jeg er en sertifisert instruktør og instruktør trener
  - **Vedlegg** `SoftwareCarpentry_Instructor_Certificate.pdf`
  - **Vedlegg** `DataCarpentry_Instructor_Certificate.pdf`
  - **Vedlegg** `Carpentries_Instructor_Trainer_Certificate.pdf`
- Jeg har bidratt til undervisningsmaterial til The Carpentries
  - The Carpentries Instructor Training [9]
  - Software Carpentry: The Unix Shell [18]
  - Software Carpentry: Programming with Python [19]
  - Software Carpentry: Version Control with Git [20]
  - Software Carpentry: Automation and Make [21]
  - Data Carpentry Wrangling Genomics Lesson [22]

- Jeg er også medlem av The Carpentries Executive Council, som kan sees på som organisajsonens styret (2018-2021)
- Sammen med Karin Lagesen og Realfagsbiblioteket etablerte vi i 2014 Carpentry@UiO
- UiO er siden 2017 medlemsorganisasjon av The Carpentries
- Software Carpentry workshops jeg har bidratt til
  - Universitetet i Oslo: 2012, 2013, 2015 - 2019
  - Netherlands eScience Centre: 2017
  - Universitet i Bergen: 2014
  - Science for Life Laboratory, Stockholm, Sverige: 2014
  - Online (ved UiO): 2020, 2021
- Instruktør trening workshops jeg har undervist for
  - Universitetet i Oslo: 2016, 2018
  - Online for the Carpentries: 2016-2019

En forskende tilnærming

- Vitenskapelige publikasjoner
  - **A. Nederbragt, R. M. Harris, A. P. Hill and G. Wilson** (2020). "Ten quick tips for teaching with participatory live coding," *PLOS Computational Biology*, 16(9),pp. e1008090, doi: 10.1371/journal.pcbi.1008090
- Konferansebidrag
  - MNT konferansen mars 2019: **Gregers, T.F., and Nederbragt, Lex** (2019). Lektorstudenter utvikler unik kompetanse og bidrar til økt kvalitet på begynneremner gjennom en undervisningsrettet master. *Nordic Journal of STEM education* 3, 23–27, https://doi.org/10.5324/njsteme.v3i1.2992. Tilgjengelig som fil: `Gregers and Nederbragt, Lex - 2019 - Lektorstudenter utvikler unik kompetanse og bidrar.pdf`
  - MNT konferansen mars 2021: **J.E.Eliassen, M.V.Bøe, L.Nederbragt, M.M.Berg, og T.F.Gregers** (2021). Motivasjon for beregningsorientert biologi og sammenhengen med matematikk R2 fra videregående opplæring. *Nordic Journal of STEM education*, *in press*. Tilgjengelig som fil: `Eliassen_etal_2021_Motivasjon for beregningsorientert biologi.pdf`
- Forskning på egen undervisning
  - Masterstudent June Edvarda Eliassen (2020)
    - tittel: Biologistudenters motivasjon for beregningsorientert biologi etter innføring av krav om full fordypning i realfaglig matematikk
    - lenke til oppgaven: http://urn.nb.no/URN:NBN:no-82918
    - fil: `Masteroppgave_Eliassen.pdf`
  - Masterstudent Sofie Rudberg (2020)
    - tittel: Relevansen av kompetansen fra matematikk R2 i beregningsorientert biologi
    - lenke til oppgaven: http://urn.nb.no/URN:NBN:no-82936
    - fil: `Masteroppgave_Rudberg.pdf`
  - Masterstudent Marthe Mjøen Berg (2019)
    - tittel: Studentar si interesse og meistringsforventning for programmering og modellering i biologi
    - lenke til oppgaven: http://urn.nb.no/URN:NBN:no-73633

        - fil: `Masteroppgave_Berg.pdf`
    - Masterstudent Lars Erik Håland (2019)
        - tittel: Studenters arbeid med programmering i biovitenskapelige problemstillinger. En kvalitativ studie av biologistudenters arbeid med Python
        - lenke til oppgaven: http://urn.nb.no/URN:NBN:no-73632
        - fil: `Masteroppgave_Håland.pdf`

En kollegial holdning og praksis

- UnderVerk
    - Oversikt over workshops (krever login med UiO brukernavn og passord)
    - Se også denne side om UnderVerk
- Gruppelærere
    - Dokumentasjonssider for gruppelærere (krever login med UiO brukernavn og passord)
- CCSE
    - Jeg er en del av ledergruppen
    - Bidrag til CCSE sine årsrapporter

Fokus på studentenes læring

Det digitale

- Prosjekt "Intergrasjon av Jupyter og Canvas for digital vurdering"
    - Prosjektsiden (krever login med UiO brukernavn og passord)
    - finansert av LINK - Senter for Læring og Utdanning ved UiO
        - Utlysning av prosjektmidler for digital vurdering
- JupyterHub ved UiO
    - https://jupyterhub.uio.no
    - Dokumentasjon for studenter
    - Dokumentasjon for undervisere
    - Canvas side om JupyterHub for studenter i BIOS1100

8. Litteraturliste

1. Miller GA. The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review. 1956;63: 81–97. doi:10.1037/h0043158

2. Caspersen ME, Bennedsen J. Instructional Design of a Programming Course: A Learning Theoretic Approach. Proceedings of the Third International Workshop on Computing Education Research. New York, NY, USA: ACM; 2007. pp. 111–122. doi:10.1145/1288580.1288595

3. Sweller J, van Merriënboer JJG, Paas F. Cognitive Architecture and Instructional Design: 20 Years Later. Educational Psychology Review. 2019. doi:10.1007/s10648-019-09465-5

4. Jenkins T. On the difficulty of learning to program. Proceedings for the 3rd Annual conference of the LTSN Centre for Information and Computer Sciences. Loughborough University; 2002. pp. 53–58.

5. Robins A, Rountree J, Rountree N. Learning and Teaching Programming: A Review and Discussion. Computer Science Education. 2003;13: 137–172. doi:10.1076/csed.13.2.137.14200

6. Guzdial M. Learner-Centered Design of Computing Education: Research on Computing for Everyone. Morgan & Claypool; 2015.

7. Fisher D, Frey N. Better Learning Through Structured Teaching: A Framework for the Gradual Release of Responsibility. ASCD; 2013.

8. Nederbragt A, Harris RM, Hill AP, Wilson G. Ten quick tips for teaching with participatory live coding. PLOS Computational Biology. 2020;16: e1008090. doi:10.1371/journal.pcbi.1008090

9. Becker EA, Koch C, Word K, Harris RM, Sane M, Nederbragt L, et al. Carpentries/instructor-training: The Carpentries Instructor Training June 2019. Zenodo; 2019. doi:10.5281/zenodo.3258398

10. Håland LER. Programmering I Biovitenskapelige Problemstillinger. University of Oslo; 2019.

11. Mazur E. Peer instruction: A user's manual. Upper Saddle River, N.J.: Prentice Hall; 1997.

12. Crouch CH, Mazur E. Peer Instruction: Ten years of experience and results. American Journal of Physics. 2001;69: 970–977. doi:10.1119/1.1374249

13. Biggs J. What the student does: Teaching for enhanced learning. Higher Education Research & Development. 2012;31: 39–55. doi:10.1080/07294360.2012.642839

14. Biggs JB, Tang CS. Teaching for quality learning at university: What the student does. Philadelphia, Pa.]; Maidenhead, Berkshire, England; New York: McGraw-Hill/Society for Research into Higher Education ; Open University Press; 2011.

15. Luxton-Reilly A. Learning to Program is Easy. Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education. New York, NY, USA: ACM; 2016. pp. 284–289. doi:10.1145/2899415.2899432

16. Gregers TF, Nederbragt L. Lektorstudenter Utvikler Unik Kompetanse Og Bidrar Til øKt Kvalitet På Begynneremner Gjennom En Undervisningsrettet Master. Nordic Journal of STEM education. 2019;3: 23–27. doi:10.5324/njsteme.v3i1.2992

17. Eliassen JE, Bøe MV, Nederbragt L, Gregers TF. Motivasjon for beregningsorientert biologi og sammenhengen med matematikk R2 fra videregående opplæring. Nordic Journal of STEM Education. 2021;5. doi:10.5324/njsteme.v5i1.3917

18. Mensa IA, Alexander H, Allen J, Alsheikh-Hussain A, Attali D, Baird D, et al. Software Carpentry: The Unix Shell. 2017. doi:10.5281/zenodo.278226

19. Achterberg H, Adams J, Adelman J, Allen J, Aranda J, Bae S, et al. Software Carpentry: Programming with Python. 2017. doi:10.5281/zenodo.278222

20. Ahmadia A, Allen J, Appling A, Aubin S, Bachant P, Baird D, et al. Software Carpentry: Version Control with Git. 2017. doi:10.5281/zenodo.278219

21. Allen J, Bachant P, Banaszkiewicz P, Bekolay T, Blischak J, Boissonneault M, et al. Software Carpentry: Automation and Make. 2017. doi:10.5281/zenodo.278220

22. Wilson G, Becker E, McKay S, Michonneau F, Williams JJ, Mayes AC, et al. Data Carpentry Wrangling Genomics Lesson. 2017. doi:10.5281/zenodo.1064254