Fakultet for naturvitenskap og teknologi
Institutt for fysikk

MASTEROPPGAVE FOR

STUD. TECHN. ARNE STORMO

Oppgaven gitt: 20.01.2009
Besvarelsen levert: 16.06.2009

FAGOMRÅDE: NUMERISK FYSIKK OG UNDREVISNING

Norsk tittel: *"Integrering av numeriske beregninger i grunnleggende fysikkurs"*
Engelsk tittel: *"Integration of numerical calculations in basic physics courses"*

Hovedoppgaven er utført ved Institutt for fysikk, NTNU, under veiledning av

Alex Hansen, Jon Andreas Støvneng og Ingve Simonsen.

Forord

Da sto man på dørstokken til å måtte forlate den trygge og ubekymrede tilværelsen som student på vei ut i det virkelige liv, med de forpliktelser og krav som det fører med seg. Takket være mange tilfeldigheter, gode valg, hardt arbeid og støttende medmennesker går jeg i det minste ut i livet som sivilingeniør. Det har vært fem fantastiske år i Trondheim, og jeg har aldri angret et eneste sekund på at jeg kom hit.

Jeg har mange å takke, og de jeg måtte glemme, får bare tilgi meg det. Jeg vil begynne med å takke familien min, for min gode oppvekst, og for at de oppdrog meg til å bli en nysgjerrig liten guttunge. Jeg vil gjerne takke de lærerne som har hjulpet meg opp gjennom barne-, ungdom- og videregående skole. Jeg vil takke alle vennene jeg har fått i løpet av studietiden, og da spesielt de på Fysikk og matematikk. Jeg håper de ikke har gått helt lei meg enda, og at jeg får se mest mulig til dem senere i livet. Jeg vil gjerne takke NTNUI for å ha holdt med sånn noen lunde i form de siste fem årene, og Samfundet for det motsatte. Takk til mine veiledere gjennom både prosjektoppgaven og masteroppgaven.

Til sist vil jeg takke alle de gode foreleserne på NTNU for alt de har lært meg, og de mindre gode foreleserne for at de lærte meg å finne ut av ting på egenhånd.

Studenter i den gamle stad, ta vare på byens ry!
-Arne Stormo 16. juni 2009.

Sammendrag

Masteroppgaven *Integrering av numeriske beregninger i grunnleggende fysikkurs* tar sitt utgangspunkt i Steinbach-rapporten og egne erfaringer med hensyn på manglende integrering av IKT i sivilingeniørutdannelsen ved NTNU. Det er undersøkt i hvilken grad IKT er integrert i fysikkutdannelsen på studielinjen Teknisk fysikk på sivilingeniørstudiet Fysikk og matematikk ved NTNU. Det er også gjort en studie av "Computers in Science Education"-reformen ved UiO. Dette i den hensikt å kunne gi gode anbefalinger til tiltak ved Institutt for fysikk ved NTNU, angåene bruk av IKT i undervisningen. Dette dokumentet er ment til å bidra til debatten om økt bruk av IKT ved Institutt for fysikk ved NTNU.

Abstract

The Master-thesis *Integration of numerical calculations in basic physics courses* has its outspring in the Steinbach-evaluation and personal experience with lack of integration of ICT in the engineering education at NTNU. It has been examined how ICT is integrated in the physics education in the Mayor Applied Physics in the Applied Physics and Mathematics, Master of Science program at NTNU. It is also done a survey of the "Computers in Science Education"-reform at UiO. This is done with the intention to give sound advice to the Department of Physics at NTNU, regarding its use of ICT in the education. This document is intended to contribute to the debate regarding the use of ICT at the Department of Physics at NTNU.

Innhold

Introduksjon

Etter ni semestre på sivilingeniørstudiet Fysikk og matematikk har en tilegnet seg en rekke egenskaper. Et studium som er så bredt, matematisk tungt og som har et slikt høyt tempo gjør at en utvikler evnen til å lære fort og jobbe strukturert. De studentene som fullfører, sitter igjen med en høy kompetanse på generell fysikk, en skarp analytisk tankegang og en god evne til faglig tilpasning. Etter en sommerjobb i SINTEF Energi i avdelingen for forbrenning i 2007 oppdaget jeg likevel at det var en egenskap jeg ikke hadde tilegnet meg tilfredsstillende. Jeg var særdeles blank når det kom til programmering og maskinelle beregninger.

Dette førte til at jeg tok faget TFY4235 Numerisk fysikk våren 2008 og forsøkte å legge et fokus på numerikk og programmering i prosjektoppgaven høsten 2008. Jeg mente likevel at jeg ikke fikk brukt tilstrekkelig tid på å drille programmeringen inn i fingrene. Jeg innså at om jeg på et tidligere stadium hadde innsett hvilket kraftig verktøy datamaskinen er for å gjøre fysiske beregninger, hadde jeg kunnet bruke den i små porsjoner under hele undervisningsløpet, og dermed fått den mengdetreningen jeg hadde behov for. Etter samtaler med mine nåværende veiledere ble vi enige om at mye av grunnen til studentenes manglende kompetanse innen programmering måtte skyldes manglende eksponering opp i gjennom studiene, siden vi ikke mangler formell opplæring. Det undervises to IKT-fag på linjen for Fysikk og matematikk, TDT4105 IT Grunnkurs og TDT4102 Prosedyre- og objektorientert programmering. Disse fagene er nyinnført i studieprogrammet og har erstattet TDT4115 IT Grunnkurs og TDT4100 Objektorientert programmering.

Mangler i IKT-ferdigheter er ikke et problem isolert til studentene ved Institutt for fysikk. I perioden 2007–2008 ble det gjennomført en ekstern evaluering av sivilingeniørutdannelsen ved NTNU. Resultatene av denne evalueringen er å finne i hva populært blir kalt Steinbach-rapporten[1]. I konklusjonen fra rapporten kan det siteres at *"... bruken av IKT i undervisning og læring (eksperimenter, simuleringer, eksempler, øvinger...) burde bli utviklet og integrert*[1]*."*

I sammenheng med den eksterne evalueringen ble det foretatt en intern evaluering ved alle studieprogrammene der det utdannes sivilingeniører ved NTNU. Den interne evalueringen for Fysikk og matematikk avdekket også en manglende bruk av IKT. Det kom fram blant annet at *"fysikkstudentene ønsker en sterkere matematisk bakgrunn enn andre studieprogrammer."*, og at de ønsket en økt *"... bruk av informatikk og datamaskiner i fysikkemnene*[2]*."* Evalueringskomitéen finner det blant mulighetene for å styrke studie-

programmet at en kan og burde *"revitalisere computer-basert klasseromsdemonstrasjoner i grunnleggende kurs"*, og *"styrke numerisk fysikk i programmet*[2]*."* Det er også ifølge komitéen et viktig læringsmål at studentene skal ha *"kunnskap og erfaring i numerisk fysikk og matematikk."*

Denne masteroppgaven skal derfor i lys av de nevnte bekymringer gi et mer detaljert bilde av hvordan IKT blir undervist og benyttet på siv.ing.-linjen Fysikk og matematikk, med fokus på studieretningen Teknisk fysikk, og undersøke tiltak for å rette opp de aktuelle problemene.

Under et besøk ved NTNU den 13. februar 2009 holdt professor Morten Hjorth-Jensen fra Universitetet i Oslo (UiO) foredraget "Datamaskiner i realfagsopplæringen, en ny måte å undervise realfag på?"[3]. Der ble det lagt ut om hvilken posisjon programmering og numeriske metoder hadde fått på Det matematisk-naturvitenskapelige fakultet (MN-fakultetet) ved UiO. Etter dette foredraget ble oppgaven utvidet til å sammenligne fysikkutdannelsen ved NTNU og UiO, se hvilke prosesser som hadde ført til hevingen av IKT-integreringen ved MN-fakultetet, og undersøke hvilke erfaringer en kan benytte seg av for å gjennomføre en lignende prosess ved NTNU.

Fra sentralt hold foregår nå lignende prosesser. Med bakgrunn i Steinbach-rapporten[1] og notat fra UiO, "Computers in Science Education: A new way to teach science?"[4], har Forvaltningsutvalget for sivilingeniørutdannelsen (FUS) nedsatt en arbeidsgruppe for integrert bruk av IKT i sivilingeniørutdannelsen. Det forventes at gruppen skal være ferdig med sitt arbeid innen utgangen av august 2009.

I kapittel 1 vil vi først ta for oss undervisning og bruk av IKT-verktøy på Fysikk og matematikk. Deretter vil det bli presentert tre undersøkelser gjort blant vitenskapelig ansatte, nyutdannede og masterstudenter ved Teknisk fysikk. Disse undersøkelsene er med på å kartlegge kompetanse, behov og holdninger i forhold til IKT på studieretningen Teknisk fysikk. Til sist vil vi diskutere mulighetene for et tverrfaglig samarbeid for å heve kompetansen på IKT blant studentene på Teknisk fysikk.

I kapittel 2 vil vi se på motivene for den omfattende CSE-reformen ved MN-fakultetet ved UiO. Deretter vil vi se på hvordan denne i praksis har blitt implementert, og hvilke kostnader og gevinster dette har gitt, både finansielt, menneskelig og pedagogisk. Til sist vil vi gå igjennom en rekke anbefalinger fra de involverte i CSE-reformen ved UiO.

I kapittel 3 blir det gjennomgått noen eksempler på forskjellige type IKT-fokuserte fysikkoppgaver. Deretter vil vi se på hvordan forskjellige IKT-verktøy passer til forskjellige typer oppgaver. Til sist vil vi redegjøre for noen forsøk med IKT-oppgaver gjort i forbindelse med denne masteroppgaven.

Til sist, i kapittel 4, vil vi først foreta en oppsummering av de momentene som er blitt drøftet i de foregående kapitlene, før det kommer en anbefaling til konkrete tiltak som kan iverksettes vet Institutt for fysikk.

Kapittel 1

Dagens situasjon ved Institutt for fysikk

Studentene som velger studieretningen Teknisk fysikk ender ofte opp med å programmere i forbindelse med prosjekt- og masteroppgave, og ofte i jobb etter endt utdanning. Ifølge Steinbach-rapporten[1] er ikke slike ferdigheter de mest prioriterte ved NTNU. For å kunne gjøre fornuftige tiltak, må det først kartlegges hvilke muligheter en har, og hvilke mangler som finnes. Derfor tar vi her først for oss dagens utdanningsløp ved Teknisk fysikk, og i hvor stor grad IKT allerede er integrert i undervisningen. Deretter skal vi gjøre rede for undersøkelser gjort blant studenter, ferdigutdannede og ansatte ved Institutt for fysikk. Dessuten skal vi se hvilke muligheter som finnes for tverrfaglig samarbeid for å styrke studentenes ferdigheter i IKT.

1.1 Undervisningen ved Institutt for fysikk

Emnevalget for siv.ing.programmet Fysikk og matematikk, og dermed for Teknisk fysikk, har forandret seg noe i de siste årene. JSP i TDT4115 IT Grunnkurs er blitt byttet ut med MATLAB i TDT4105 IT Grunnkurs. Samtidig har faget TDT4100 Objektorientert programmering blitt erstattet med TDT4102 Prosedyre- og objektorientert programmering. Dette innebærer et skifte av programmeringsspråk fra Java til C/C++. Dette har forandret forutsetningene for bruk av numerikk og programmering.

Samtidig er det begynt med enkelte forsøk på å innføre bruk av maskinell beregning i de tidlige fysikkemnene. Det er da interessant å vite hvilke tiltak som er nødvendige, og ikke minst hvilke problemer som allerede er tatt hånd om. Derfor tar vi her for oss de obligatoriske fagene for Tekntisk fysikk. Først kommer en oppsummering av hva studentene lærer i løpet av de to IKT-emnene TDT4105 IT Grunnkurs og TDT4102 Prosedyre- og objektorientert programmering, som holdes henholdsvis i 1. og 4. semester. Deretter kommer en oversikt over hvor IKT-ferdighetene blir tatt i bruk senere.

1.1.1 TDT4105 IT Grunnkurs

Læringsmål: *Studentene skal få en generell innsikt i informasjonsteknologi og utvikle kunnskaper, ferdigheter og holdninger til bruk av informasjonsteknologiske metoder i en ingeniørs arbeidssituasjon. Studentene skal lære seg grunnleggende prosedyreorientert programmering, HTML og databaser.*[5]

En gjennomgang av forelesningsnotater og øvingsoppgaver viser hva studentene skal sitte igjen med av kunnskap. Siden det er fagets nytteverdi som beregningsverktøy som her er interessant, skiller vi mellom det som er programmeringskunnskap, og det som er mer generell informasjon og informatikk.

For å ta det siste først, så inneholder kurset en del praktisk informasjon angående bruk av NTNUs it-tjenester. Dette innebærer bruk av webmail (NTNUs eposttjeneste), UNIX-serverne og hjemmeområdet, VPN, remote desktop, progdist og hvordan legge ut en hjemmeside.

Det foreleses i forskjellen på maskinvare og programvare, og hvordan informasjonsteknologi påvirker samfunnet vi lever i før og nå. Det blir gitt en gjennomgang av det norske lovverket, som regulerer bruk av informasjon og informasjonsteknologi. Her blir studentene bedt om å reflektere over egne holdninger til nevnte lovverk, eksempelvis hva de synes om ulovlig fildeling, opphavsrett og personvern.

Studentene får opplæring i hvordan en designer en hjemmeside ved bruk av HTML og cascading style sheet (CSS). Her kommer for første gang konseptet om tekstbasert formatering av dokumenter. I sammenheng med dette blir det gjennomgått hvordan nettverk, herunder internett, er bygd opp, både av maskin- og programvare. Det gjennomgåes hvordan datamaskiner sender, mottar og tyder informasjon, og hva tjenestekvalitet innebærer. Det undervises også i datarepresentasjon, informasjonsentropi, systemutvikling og ikke minst databaser.

Størstedelen av kurset omhandler likevel programmering i MATLAB. Hovedtema som undervises er listet i Tabell 1.1. I øvingsopplegget blir det gitt oppgaver i informatikk, som sortering og registrering. Flere av øvingene er dog lenket opp mot øvingene i TMA4100 Matematikk 1. Dette inkluderer løsning av ligninger ved hjelp av Newtons metode, enkel numerisk integrasjon og beregning av rekker og følger.

Gjennomgangen av TDT4105 IT Grunnkurs viser at studentene allerede etter første semester skal være i stand til å gjøre de enkleste maskinelle beregninger. Desuten skal de være i stand til å visualisere data ved å lage grafer. Numerikken i øvingene går ikke ut over pensum i TMA4100 Matematikk 1.

1.1.2 TDT4102 Prosedyre- og objektorientert programmering

Læringsmål: *Studentene skal få ferdigheter i programmering, med hovedvekt på prosedyreorientert programmering, men også med forståelse for sentrale prinsipper innen objektorientert programmering, og kjennskap til hvilke problemtyper disse to ulike programmeringsparadigmene passer for. Studentene skal få trening i bruk av relevante programmeringsmetoder og verktøy, og kjennskap til anvendelsesområder, begrensninger og underliggende*

- Tilordning og lesing av variable.
- Representasjon av vektorer og matriser.
- Kall av innebygde funksjoner og operatorer.
- Logiske verdier og spørringer.
- Plotting av funksjoner.
- Input/output, både til fil og kommandolinje.
- Formatering av input/output.
- Skrive og kjøre skript.
- Pseudokode/flytdiagram.
- Nøstede funksjoner.
- Skrive egne funksjoner.
- Pekere.
- Synbarhet for variable.
- Feilsøking i kode.
- Lineære ligningsysstemer.
- Strengbehandling og lagring.

Tabell 1.1: Hovedtema i MATLAB-undervisningen i TDT4105 IT Grunnkurs[5]

teori.[6]

Der TDT4105 IT Grunnkurs tar for seg et bredt spekter av IKT-relaterte tema, er TDT4102 Prosedyre- og objektorientert programmering et rent programmeringskurs. Det fokuserer på C/C++ og deler, som navnet tilsier, mellom programmeringsparadigmene prosedyre- og objektorientert. Tabell 1.2 lister opp tema som kommer utover det som undervises i TDT4105 IT Grunnkurs.

- Programmeringsspråk og datamaskiner.
- Problemløsnings- og programmeringsmetodikk.
- Variable, datatyper og datastrukturer.
- Kontrollstrukturer.
- Prosedyrer, funksjoner, parameteroverføring.
- Filer og filbehandling, innlesing / utskrift.
- Rekursjon.
- Minneallokering. Pekere og dynamiske variable, lenkede lister, binære trær.
- Objekter og klasser, arv og innkapsling, metodekall, overstyring.
- Funksjons- og klassebiblioteker.
- Unntak og feilbehandling. Innebygde og selvdefinerte.
- Iteratorer.

Tabell 1.2: Hovedmomenter i TDT4102 Prosedyre- og objektorientert programmering[6]

Editoren som brukes er Visual Studio. Gjennom øvingsopplegget får studentene øving i hvordan en programmerer i større prosjekt ved at det stilles krav til eksakt grensesnitt

på funksjoner og klasser. Studentene blir tvunget til å lete opp informasjon og dokumentasjon av bibliotek selv, riktignok med noen hint, slik at de blir mer selvhjulpne utover i faget.

Flere av øvingene er bygd opp slik at de første oppgavene tar for seg det spesifikke temaet for øvingen. Deretter bygger en opp et lite program rundt det aktuelle temaet. Slik får studentene trening i å bygge opp en komplett, om enn kort, programvare.

Det gjennomføres ingen øvingsoppgaver som direkte kan relateres til numerikk eller på noen annen måte ha direkte applikasjoner i fysikk. Faget dreier seg utelukkende om programmeringstekniske tema og applikasjoner innen informatikk. Til gjengjeld får studentene en grundig gjennomgang i disse temaene, og skal etter endt semester være kompetente programmerere.

1.1.3 Bruk av IKT i obligatoriske fysikkfag

Som nevnt i de to foregående delkapitlene, går studentene på Fysikk og matematikk gjennom to IKT-emner, TDT4105 IT Grunnkurs og TDT4102 Prosedyre- og objektorientert programmering. Disse emnene skal sette studentene i stand til å bruke MATLAB som regneverktøy og skrive egne programmer i C/C++. For å klargjøre i hvilket omfang studentene får bruk for disse kunnskapene i de obligatoriske fysikk- og matematikkemnene, har forelesninger, lab- og øvingsoppgaver blitt gjennomgått. Bruken av ferdighetene i IKT blir redegjort for under.

Det blir i denne utredningen skilt mellom tre forskjellige grunnoppgaver. Vi deler inn i databehandling, visualisering og beregning. De emene som gjør bruk av programering for å løse slike oppgaver vil bli skjematisk listet opp under. I tillegg brukes noe spesialisert programvare. Disse blir nevnt separat.

Databehandling innebærer at studentene får eller måler data fra reelle eller tenkte eksperimenter. Fra disse data skal de så finne samenhenger eller parametre ved hjelp av f.eks. regresjonsanalyse. Et eksempel på en oppgave i databehandling er gitt i delkapittel 3.1.3. Fag som inneholder bruk av disse oppgavene på Teknisk fysikk er oppgitt i Tabell 1.4.

Visualiseringsoppgaver innebærer at studentene skal lage en grafisk fremstilling av data eller funksjoner. Dette er i sin enkleste form å plotte grafer. En eksempeloppgave er gitt i delkapittel 3.1.2. Fag som inneholder bruk av disse oppgavene ved Teknisk fysikk er oppgitt i Tabell 1.5.

Beregningsoppgaver er hva vi ofte mener når vi snakker om numerikk i fysikkoppgavene. Dette er å løse ligninger eller ligningssett ved hjelp av numeriske algoritmer. En eksempeloppgave er gitt i delkapittel 3.1.1. Fag som inneholder bruk av disse oppgavene ved Teknisk fysikk er oppgitt i Tabell 1.6.

De fag der IKT-oppgaver er nevnt nærmest i forbifarten, er ikke med i oversikten.

Det finnes allerede noe integrert bruk av IKT i emnene som er obligatoriske for Teknisk fysikk (1.-4. årskurs). Dette går hovedsaklig ut på plotting av data og regresjonsanalyse i lab-undervisningen. Vi registrerer et skifte fra Excel til Matlab. Utover lab-oppgavene gjøres det et par oppgaver i TFY4160 Bølgefysikk og TFY4155 Elektromagnetisme i plott av funksjonsuttrykk. Disse oppgavene er innført i løpet av de to

Emnekode	Emnenavn
TMA4100	Matematikk 1
TFY4145	Mekanisk fysikk
TDT4105	Informasjonsteknologi grunnkurs
EXPH0001	Filosofi og vitenskapsteori
TMA4105	Matematikk 2
TMA4115	Matematikk 3
TFY4155	Elektromagnetisme
TMT4110	Kjemi
TMA4120	Matematikk 4K
TFY4160	Bølgefysikk
TEP4105	Fluidmekanikk
TIØ4256	Teknologiledelse 1
TMA4245	Statistikk
TFY4165	Termisk fysikk
TFY4215	Kjemisk fysikk og kvantemekanikk
TDT4102	Prosedyre- og objektorientert programmering
TFY4185	Måleteknikk
TFY4230	Statistisk fysikk
TFY4240	Elektromagnetisk teori
TFY4250	Atom- og molekylfysikk
TFY4190	Instrumentering
TFY4195	Optikk
TFY4220	Faste stoffers fysikk
TFY4205	Kvantemekanikk
TFY4225	Kjerne- og strålingsfysikk
EiT	Eksperter i Team. Tverrfaglig prosjektarbeid.

Tabell 1.3: Emnekoder for obligatoriske fag ved Teknisk fysikk, NTNU.

foregående semestrene. I TFY4215 Kjemisk fysikk og kvantemekanikk gjøres det nå en beregningsoppgave i Matlab med vedlagt kildekode. Dette er nytt i 2009. I TMA4105 Matematikk 2 brukes Maple i to gruppeoppgaver. I TFY4215 Kjemisk fysikk og kvantemekanikk brukes SPARTAN til Hartree-Fock-beregninger av mangepartikkel-system (molekyler). I TFY4185 Måleteknikk brukes PSpice til elektronisk kretsdesign. I TFY4190 Instrumentering brukes LabVIEW for vekselvirkning mellom datamaskinen og egenprodusert måleinstrument. Ellers brukes egne programmer for digitalt måleutstyr i flere av lab-oppgavene.

1.2 Undersøkelse blant vitenskapelig ansatte

I forbindelse med oppgaven ble det utført en undersøkelse blant de vitenskapelig ansatte ved Institutt for fysikk for å kartlegge kompetansenivået og viljen til å bygge et koordi-

Semester	Fagkode			
1	TMA4100	***TFY4145***	TDT4105	EXPH0001
2	TMA4105	TMA4115	***TFY4155***	TMT4110
3	TMA4120	TFY4160	TEP4105	TIØ4256
4	TMA4245	***TFY4165***	TFY4215	TDT4102
5	TFY4185	TFY4230	TFY4240	TFY4250
6	TFY4190	TFY4195	***TFY4220***	Valgbart
7	TFY4205	TFY4225	Valgbart	Valgbart
8	EiT	Valgbart	Valgbart	Valgbart

Tabell 1.4: Emner som har oppgaver i databehandling i fet skrift. Fag i kursiv har disse utelukkende i lab-oppgaver.

Semester	Fagkode			
1	TMA4100	***TFY4145***	**TDT4105**	EXPH0001
2	TMA4105	TMA4115	**TFY4155**	TMT4110
3	TMA4120	**TFY4160**	TEP4105	TIØ4256
4	TMA4245	TFY4165	**TFY4215**	TDT4102
5	TFY4185	TFY4230	TFY4240	TFY4250
6	TFY4190	TFY4195	***TFY4220***	Valgbart
7	TFY4205	TFY4225	Valgbart	Valgbart
8	EiT	Valgbart	Valgbart	Valgbart

Tabell 1.5: Emner som har oppgaver i visualisering i fet skrift. Fag i kursiv har disse utelukkende i lab-oppgaver.

Semester	Fagkode			
1	TMA4100	TFY4145	**TDT4105**	EXPH0001
2	TMA4105	TMA4115	TFY4155	TMT4110
3	TMA4120	TFY4160	TEP4105	TIØ4256
4	TMA4245	TFY4165	**TFY4215**	TDT4102
5	TFY4185	TFY4230	TFY4240	TFY4250
6	TFY4190	TFY4195	TFY4220	Valgbart
7	TFY4205	TFY4225	Valgbart	Valgbart
8	EiT	Valgbart	Valgbart	Valgbart

Tabell 1.6: Emner som har oppgaver i beregning i fet skrift.

nert undervisningsopplegg for numerikk i fysikkfagene. Denne undersøkelsen ble utført i samarbeid med Kåre Olaussen, nestleder ved Institutt for fysikk.

Undersøkelsen ble gjennomført ved at de vitenskapelig ansatte ble bedt om å svare på følgende spørsmål via epost.

1. Hvilke emner kan du huske å ha undervist de siste 10 år?

2. Har noen av disse emnene involvert noen form for beregninger på datamaskin (inkludert f.eks. generering av plott til prosjektrapport e.l.)?

3. Anser du det for ønskelig/ikke ønskelig at emner du underviser involverer bruk av beregningsprogrammer? Hvilke hindringer ser du eventuelt for slik bruk?

4. Kan du tenke deg å ta med oppgaver/opplæring i numerikk/beregningsverktøy i emner du underviser? Spesielt hvis du ikke trenger å utvikle opplegget selv?

5. Synes du at studentene har tilstrekkelig kompetanse i bruk av numerikk/programmering/beregningsprogramvare? Gi eventuelt eksempler på god/dårlig kompetanse du har erfart.

6. Hvilke beregningsprogrammer kan du (angi kompetansenivå)?

7. Hvilke programmeringsspråk kan du (angi kompetansenivå)?

24 personer gav grundige svar på undersøkelsen.

Flere av dem som svarte, fortalte at de allerede brukte en eller annen form for beregningsverktøy eller numerisk beregning i fagene de hadde undervist. Dette gjelder spesielt fagene i høyere årskurs. På lavere årskurs blir det mest bruk av MATLAB eller Excel for databehandling og plott i lab-oppgavene. Det er gjort noen forsøk på å legge til numeriske oppgaver i øvingsopplegget for enkelte av fagene i de to første årene, i TFY4160 Bølgefysikk, TFY4155 Elektromagnetisme og TFY4215 Kjemisk fysikk og kvantemekanikk. Det var et klart ønske fra faglærerne på høyere årskurs som brukte beregningsverktøy i undervisningen, at studentene allerede var komfortable med slike verktøy når de begynte i disse kursene. Dermed kunne en større del av tiden gå til fysikk, og mindre på teknisk implementasjon av kode. Mange av de spurte nevnte at studentene trengte kunnskaper i numerikk og databehandling under prosjekt- og masteroppgave og burde vært mye bedre forberedt, da mange av studentene aldri hadde tatt i et beregningsverktøy før. Derfor måtte de bruke en god del tid på å sette seg inn i det.

Det var gjennomgående at de vitenskapelig ansatte ønsket å heve kompetansenivået til studentene i bruk av programmering og numerikk. 3 av 4 svarte at de ønsket et opplegg for bruk av numerikkberegninger i fagene velkommen. Svært mange av dem ønsket å ha innflytelse på opplegget for sitt eget fag. Flere kommenterte at dette måtte være et sentralisert program fra instituttet sin side, à la hvordan lab-øvingene er lagt opp, både for at en skulle kunne gi en bredest mulig opplæring, og for at det ikke skulle stå og falle på den enkelte faglærers vilje og/eller kompetanse hvordan opplæringen i de enkelte kursene ble. Et fåtall (2 personer) ønsket å overlate hele prosessen til en eventuell komité. Det var et uttrykt ønske om at studentene ikke kun fikk kompetanse på beregninger og numerikk, men også en grundig opplæring i maskinell databehandling og analyse.

Den største bekymringen fra de ansatte på studentenes vegne, var ikke nødvendigvis at studentmassen som helhet hadde for lav kompetanse, men at forskjellene var så store. Der noen studenter uten videre skrev avansert kode i C++, hadde andre problemer med å produsere en lesbar graf i MATLAB.

Verktøy	Brukere
MATLAB	15
FORTRAN	14
Maple	10
C/C++	10
Mathematica	7
Pascal	4
Excel	4

Tabell 1.7: Programmeringsspråk og beregningsverktøy som beherskes blant vitenskapelig ansatte ved Institutt for fysikk.

Noen av faglærerne uttrykte en viss skepsis mot å ta programmering/beregningsverktøy inn som en del av fagene sine, da de så for seg at dette kunne gå ut over "den egentlige fysikken". Desuten følte de dette var noe som ikke bare tok opp tid fra selve forelesningene, men også tok lang tid å forberede, da dette var nytt stoff. Derfor var det flere som gav uttrykk for at om dette skulle bli obligatorisk, så måtte det settes inn større ressurser for utvikling og forberedelse. En annen grunn til bekymring var at studentene var for dårlig opplært i bruk av dataverktøy. Derfor ville de bruke uforholdsmessig lang tid på slike øvinger.

MATLAB ser ut til å være de ansattes "weapon of choice" når det kommer til numeriske beregninger. 15 av de 24 som svarte fortalte at de hadde gode, eller i det minste grunnleggende ferdigheter i MATLAB. Det var omtrent like mange som svarte at de kunne FORTRAN og C/C++, men der var det flere som mente de var blitt for rustne. Andre verktøy som ble relativt ofte nevnt var Mathematica og Maple. En skjematisk oversikt over de maskinelle verktøy som beherskes av de vitenskapelig ansatte vises i Tabell 1.7. Verktøy med færre enn tre brukere er ikke listet.

Kort oppsummert kan det sies at de vitenskapelig ansatte mener at studentenes kompetansenivå er svært varierende, og at de har et sterkt ønske om at studentene får en grundig opplæring i bruk av beregningsverktøy tidlig i studiene. Dermed er de forberedt på de oppgavene de måtte møte når de tar valbare emner i høyere årskurs. Denne opplæringen burde ta for seg både beregninger og databehandling. Det burde ligge et koordinert og gjennomtenkt opplegg bak denne undervisningen, helst med tid, sted og lærerressurser spesielt satt av til dette. Hvis utviklingen av et slikt undervisningsopplegg skal gjøres kun av den faste staben på instituttet, må det hvile tungt på MATLAB som verktøy, da MATLAB er det programmet som dominerer når det kommer til hva de ansatte har kjennskap og kompetanse på. Det er svært viktig for faglærerne at bruk av beregningsverktøy forsterker fysikkforståelsen, og ikke kommer i stedet for den.

1.3 Undersøkelse blant nyutdannede fra Teknisk fysikk

Det ble foretatt en undersøkelse blant studentene som ble uteksaminert fra Teknisk fysikk sommeren 2008, for å kartlegge hvilke behov nyutdannede sivilingeniører fra Teknisk

fysikk har når de møter arbeidslivet. Undersøkelsen ble foretatt ved å ringe rundt til de aktuelle personene. Av 48 på lista, svarte 36 på henvendelsen.

De oppringte svarte på følgende spørsmål:

1. Har du fått deg jobb? Hvis ja, hvor/hvilket firma? Hvis nei, har du søkt jobb?
2. Hvis ja på 1: Hva består arbeidsoppgavene dine i?
3. Hvis ja på 1: Bruker du programmering eller beregningsverktøy i jobben din?
4. Mener du at du fikk en god nok opplæring i bruk av programmering/beregningsverktøy i undervisningen på Fysikk og matematikk?
5. Hvis du har søkt/fått jobb: Var dine kunnskaper i bruk av programmering/beregningsverktøy et tema under jobbsøking/ansettelsesprosessen?
6. Har du noen kommentarer/innspill til undervisningen i programmering/beregningsverktøy på Fysikk og matematikk?

Av de 36 som svarte på undersøkelsen hadde 35 fått seg jobb. Av disse var det 26 som sa de brukte programmering eller sin programmeringskompetanse direkte i jobbsammenheng. Dette tilsvarer rundt tre fjerdedeler av studentmassen som går ut fra Teknisk fysikk. 23 av de spurte oppgav at de ble konfrontert med sine programmeringskunnskaper under jobbsøkingsprosessen. Flere oppgav det som en forutsetning for at de fikk jobben, og noen få påstod at de hadde gått glipp av jobbtilbud på grunn av manglende kompetanse i programmering. En grafisk framstilling av hvilke arbeidsoppgaver studentene fra kull 2003 endte opp med er fremstilt i Tabell 1.8

Arbeidsoppgaver	#
Beregninger/databehandling	26
Administrasjon	3
Rent ekperimentelle arbeidsoppgaver	3
Undervisning	2
Under opplæring	1

Tabell 1.8: Fordeling av arbeidsoppgaver blant nyutdannede fra kull 2003 (av 35).

På spørsmål om de mente at de hadde fått en god nok opplæring i bruk av programmering/beregningsverktøy, kom det flere frustrerte kommentarer om hvor lite relevant datafagene hadde vært i forhold til hva som trengtes i fysikkfagene, da spesielt i prosjekt/masteroppgave. Det var et gjennomgående tema at det studentene måtte ha tilegnet seg av kompetanse på numerisk løsning av ligninger og programmering var gjort på egenhånd. De spurte var derfor svært fornøyde da det ble påpekt at JSP nå var byttet ut med MATLAB i IT Grunnkurs (byttet fra TDT4115 til TDT4105). Noen påpekte likevel at de ønsket at det skulle vært enda mer fokus på programmeringsbiten og mye mindre på web. 27 av de som svarte nevnte at det burde komme mer programmering inn i utdanningen, og av disse var 13 direkte misfornøyd med den opplæringen de fikk innenfor emnet.

De oppringte var overraskende engasjerte og positive da de ble bedt om å komme med innspill. Med noen få unntak var alle de spurte tilhengere av en økt bruk av programmering i fysikkfagene. Kun en var direkte i mot dette. Ni av de spurte trakk fram TFY4235 Numerisk fysikk som faget hvor de lærte å programmere. Flere av disse mente en burde vurdere å gjøre faget obligatorisk, og gjerne tidligere i studiet, da de følte det var det mest anvendelige faget de hadde hatt under studiene.

Det ble også nevnt at MATLAB eller lignende skulle vært brukt mer i databehandlingen i lab-oppgavene, da dette var mer realistisk i forhold til hva studentene kan forvente av oppgaver i arbeidslivet. Det kunne gjerne være en egen del av lab-tiden hvor de programmerte skriptene de skulle bruke i selve lab-oppgaven.

Flere mente at det var viktig å lære MATLAB tidlig, og bruke det jevnlig i undervisningen, blant annet for at studentene skulle få se fysikk i praksis, og for at de skulle se nytten av programmering som verktøy. Det kom flere forslag til hvordan dette skulle gjøres, deriblant obligatoriske øvingsoppgaver à la det eksisterende øvingsopplegget. Andre mente at større prosjektoppgaver gav studentene tid til å gjøre en grundigere og mer innsiktsfull jobb enn hva som kunne bli gjort i en ordinær øving, fordi programmering er en disiplin som krever en del mengdetrening for å sitte i fingrene. Noen mente at skulle slike prosjekter være vellykkede, måtte de ha en god nok grunnopplæring, slik at studentene kunne kastes ut på upløyd mark og løse oppgavene på egenhånd.

En stemme uttrykte sin skepsis mot en økt satsing på bruk av numerikk i fysikkfagene. Skepsisen lå i frykten for at dette ville skyve bort den grunnleggende fysikkforståelsen og dermed redusere studentene fra fysikere til kalkulatorer. Andre bekymringer som kom fram blant de spurte, var at det kunne bli for stort fokus på de spesifikke verktøyene/språkene som måtte bli brukt. Flere påpekte at skulle det satses på bruk av beregningsverktøy i fysikkfagene, måtte det være ganske grunnleggende kunnskaper som stod på dagsorden, og ikke bruk av ferdig utviklede applikasjoner.

Kort oppsummert kan en si at et stort flertall av studentene som blir uteksaminert fra Teknisk fysikk ender opp med arbeidsoppgaver direkte relatert til programmering og numeriske beregninger. De mener det skulle ha vært mer relevant opplæring under studiet før prosjekt- og masteroppgave. De er svært positive til en plan for å heve det generelle kompetansenivået innenfor numerikk og programmering blant studentene på Teknisk fysikk, såfremt dette ikke undergraver den fysiske forståelsen studentene oppnår i kursene. De to løsningene som de spurte i hovedsak kom med, var enten et obligatorisk numerikk/programmeringsprosjekt i fysikkfagene eller et eget numerisk fysikkfag obligatorisk og tidlig i studiene.

1.4 Undersøkelse blant avgangselever på Teknisk fysikk

For å få et detaljert bilde av hva studenter som har gjennomgått studieløpet på Teknisk fysikk mener om studeløpet, gjennomførte jeg en kombinert spørreundersøkelse og intervjuer blant avgangsstudentene. En kunne da spørre litt mer rundt de oppstilte spørsmålene. Av de 33 som ble spurt om å bidra var det 20 som deltok, enten på intervju eller ved å svare på de oppstilte spørsmålene via epost. 5 personer svarte elektronisk.

De oppstilte spørsmålene var:

1. Føler du deg komfortabel med bruk av programmering?
2. Hvilke programmeringsspråk behersker du?
3. Føler du deg komfortabel med bruk av beregningsprogrammer?
4. Hvilke beregningsprogrammer behersker du?
5. Har du hatt fag der bruk av programmering/beregningsprogrammer har vært nyttige? Hvilke?
6. Har du hatt bruk for programmering/beregningsprogrammer under prosjekt/master?
7. Så du nytten av IT-GK og Objektorientert programmering da du tok fagene?
8. Påvirket dette hvor mye du fikk ut av fagene?
9. Ser du nytten av IT-GK og Objektorientert programmering nå?
10. Har du jobberfaring hvor programmering/beregningsprogrammer var relevant?
11. Hva er din oppfatning av opplæringen i programmering/beregningsprogrammer i studieprogrammet?
12. Hva har vært mest positivt med fysikkundervisningen i studieprogrammet?
13. Hva har vært mest negativt med fysikkundervisningen i studieprogrammet?
14. Hvilke forslag har du til forbedringer?
15. Hva kan ofres for eventuelt å gi plass til numerikk/programmering?
16. Burde en ha en satsing på bruk av numerikk på Teknisk fysikk?

På de første spørsmålene svarte samtlige at de kunne bruke et eller annet programmeringsspråk eller et beregningsverktøy. Dette sa de med varierende grad av selvsikkerhet. Av disse var det først og fremst MATLAB og C/C++ som dominerte blant studentene. Det var henholdsvis 18 og 16 studenter som mente å beherske disse programmeingsspråkene. I Tabell 1.9 er en oversikt over hvor mange studenter som mente å beherske forskjellige programeringsspråk. De språk eller programmer mindre enn tre personer har nevnt er utelatt.

Alle hadde hatt fag der de var nødt til å bruke en eller annen form for dataverktøy, da noe er med i de obligatoriske fagene. Foruten dette, var det noen som hadde tatt fag utenfor Institutt for fysikk. Dette var for eksempel TTT4120 Digital signalbehandling, TTK4530 Reguleringsteknikk eller TDT4120 Algoritmer og datastrukturer, der de hadde fått bruk for noe programmering. Noen hadde brukt beregningsverktøy i TFY4275

Verktøy	Brukere
MATLAB	18
C/C++	16
Java	10
Maple	9
FORTRAN	9
Bash	3
Php	3

Tabell 1.9: Programmeringsspråk og beregningsverktøy behersket blant avgangstudenter (av 20).

Klassisk transportteori, FY3402 Subatomær fysikk, TFY4292 Kvanteoptikk og TFY4305 Ikkelineær dynamikk. Det var TFY4235 Numerisk fysikk som ble hyppigst nevnt. 14 av de spurte svarte dette faget på spørsmål 5. Flere påstod at det var i dette faget de lærte å programmere, og at det burde holdes tidligere i studieprogrammet, og at det med fordel kunne vært gjort obligatorisk. Dermed kunne man også åpne for muligheten for å holde enda mer avanserte kurs i numerisk fysikk i 4. eller 5. klasse.

På spørsmål om studentene hadde prosjekt- eller masteroppgaver der bruk av maskinell beregning var nødvendig, svarte hele 17 av 20 ja. De fleste hadde hatt behov for slike beregninger på begge oppgavene.

17 personer hadde i løpet av studietiden hatt en deltidsjobb eller en sommerjobb som innebar at de måtte kunne en eller annen form for maskinell beregning. Dette inkluderer ikke program der alt er i sorte bokser og metodene er skjult for brukeren.

På spørsmål om hva studentene syntes om standarden på opplæringen i programmering var det mye negativ respons. Den varierte i styrke fra "mangelfull" og "lite relevant" til "ikkeeksisterende" og "lært alt selv". Det var nesten ingen som var helt fornøyd med opplæringen i IKT. Det må påpekes at dette er studenter som hadde IKT-fagene før det ble skiftet til MATLAB og C/C++. Flere understreket at dette var utelukkende positivt, og noen mente også at en slik omlegging var det som skulle til for å rette opp en tidligere mangelfull utdanning. Mye av misnøyen lå på den manglende bruken av programmering som verktøy de første årene, da IT-fagene dermed blir liggende for seg selv, og studentene ser ikke helt hva de skal med dem. Derfor mente flere at den bruken av f.eks. MATLAB i enkelte av fagene i de aller høyeste årskursene burde blitt benyttet mye tidligere. Enkelte påpekte at det var merkelig at både programmeringsspråk og kanskje spesielt MATLAB ganske enkelt var forutsatt kjent i enkelte høyere årskurs. Dette samsvarte ikke med den manglende eksponeringen i de første årene.

Det var ikke bare elendighet å spore. Misnøyen med utdannelsen begrenset seg i de fleste tilfeller til den noe mangelfulle opplæringen innen programmering og mulighetene for å kunne bruke det som et verktøy innen fysikk, og etter manges mening en alt for stor andel med ikketekniske emner. På spørsmål om hva som kunne vært annerledes innen utdanningen, var det nettopp det siste som opptok de fleste spurte. 10 av de spurte mente det var for mange slike fag. Dette gjenspeilte seg ved at 13 stykker mente man enkelt

kunne ha ofret faget TIØ4256 Teknologiledelse eller et av de andre ikketekniske emnene for en grundigere gjennomgang av numerikk og programmering innen fysikk.

På spørsmål om hva studentene mente fungerte best, og hva de mente de satt igjen med som positive egenskaper etter 5 år på studiet var det en bred enighet om at de aller fleste fagene som ble tilbudt holdt en meget høy faglig standard, og at det høye tempoet gjorde en godt rustet til å sette seg raskt inn i nye fagfelt og nye problemstillinger. Det ble satt stor pris på de gode analytiske egenskapene og den faglige bredden som studentene mente å ha tilegnet seg. Dette gjenspeiler seg i hvilke holdninger de viste til en satsing på større fokus på numeriske beregninger i fysikkfagene. Studentene var på ingen måte ukritiske til å innføre en integrert bruk av numerikk i fysikkfagene som i UiO. Studentene mente de fleste fagene i utgangspunktet allerede var svært gode, og at en stor forandring kunne komme til å forårsake større skade enn gagn. Det var riktignok mindretallet som satte seg helt i mot en slik innføring, men det måtte i såfall gjøres med stor omhu og planlegges nøye. Det måtte på ingen måte ende opp som et kjapt krav fra ledelsen at nå måtte alle faglærerne innføre numeriske beregninger som en del av fagene, hvorpå faglærerne dermed ukritisk la til oppgaver i øvingsopplegget for å tekkes ledelsen. Dette var en problemstilling mange av de spurte var redd for kunne bli virkelighet hvis man iverksatte en slik reform uten å tenke seg om.

Dette endte i et tredelt svar på spørsmål 16. 9 svarte ja, vi trenger en økt satsing på numeriske beregninger i fysikkfagene. 7 svarte ja, vi kan godt satse på økt numerisk kompetanse, men det må på ingen måte føre til dårligere fysikkfag. Det var 4 som mente det ikke var noe behov for en slik satsing.

Et siste ankepunkt mot en storsatsing på numerisk fysikk var at et slik fokus i fagene som tilbys fra Institutt for fysikk ville kunne gjøre det vanskelig for studenter fra andre linjer som måtte ønske å ta disse fagene.

Oppsummert kan det sies at studentene som nå holder på å avslutte sin utdannelse på Teknisk fysikk i hovedsak er svært fornøyd med fysikkutdannelsen sin. Manglene innen programmering og maskinelle beregninger mente de aller fleste måtte rettes opp ved å ta av den tiden som nå ble brukt til altfor mange ikketekniske fag. Hvis det skulle gjøres plass til dette i fysikkfagene, måtte det planlegges nøye, slik at det ikke forringet den delen av utdanningen studentene var aller mest fornøyd med.

1.5 Muligheter for tverrfaglig samarbeid

Siden numerikk og programmering ikke er tradisjonelle fysikkfag, er det interessant å undersøke hvilke muligheter som finnes for tverrfaglig samarbeid. Dette ble gjort ved å kontakte personale ved de instituttene som tar seg av IKT og matematikkundervisningen på NTNU.

1.5.1 Idéer fra Institutt for datateknikk og informasjonsvitenskap

Etter samtale med Jørn Amundsen, koordidnator for IT Grunnkurs, har jeg følgende inntrykk av IDI-instituttets tanker om videre utvikling av TDT4105/TDT4110/IT1102

IT Grunnkurs.

Det kjøres to paralleller med IT Grunnkurs i høstsemesteret. En kjører MATLAB (TDT4105), og den andre kjører JSP (TDT4110). Siv.ing.linjen Fysikk og matematikk, som Teknisk fysikk er en forgrening av, følger nå MATLAB parallellen.

Ifølge Amundsen inneholder TDT4105 IT Grunnkurs mange forskjellige emner innenfor informasjonsteknologi. Dette gjør det vanskelig å undervise alle grundig nok. Derfor kunne en vurdere å skjære bort HTML, CSS fra kurset for å gjøre større plass for programmering. Databasedelen vil falle bort allerede fra høsten 09. Faget kan ikke utelukkende gjøres om til et rent programmeringsfag, da kurset er ex.fac.-emne for studenter på datateknikk, og derfor må inneholde tilstrekkelig teoretisk kunnskap om datamaskiner, deres oppbygging og kommunikasjonsteknologi.

Det planlegges for tiden en revidering av IT Grunnkurs. Dette vil begynne med endringer i parallellen for datateknikk. Når opplegget er kvalitetssikret, vil det ferdig utprøvde opplegget bli presentert for de andre linjene. Mulige forandringer inkluderer:

- Avslutte undervisningen i HTML.
- Avslutte undervisningen i CSS.
- Styrke programmeringsundervisningen.
- På sikt kjøre alle parallellene med ett språk.

Det er et ønske fra de ansvarlige for TDT4105 IT Grunnkurs med en integrering med TMA4100 Matematikk 1 og ex.fac.-emnene ved de forskjellige siv.ing.-linjene, for at studentene lettere skal forstå nytteverdien av programmering. Det er gjort forsøk på å legge øvingsoppgavene i TDT4105 IT Grunnkurs opp mot pensum i TMA4100 Matematikk 1 høsten 2008. Dette gav betinget suksess, da det ifølge Amundsen manglet en helhetlig plan, og oppfølging fra Institutt for matematikk. Det manglet tid og ressurser, og begge derer er nødvendige for å kunne begynne med numeriske problemstillinger i TMA4100 Matematikk 1. For å oppnå den fulle integreringen som er ønsket, mener Amundsen det er nødvendig med føringer og midler fra sentralt hold ved NTNU. I skrivende stund avventes konklusjonen fra Forvaltningsutvalget for Sivilingeniørutdanningen (FUS), der temaet er oppe til diskusjon.

1.5.2 Idéer fra Institutt for matematiske fag

Etter samtale med Brynjulf Owren, programrådsleder for studieprogrammet for Fysikk og matematikk og tidligere leder for studieretningsutvalget for Industriell matematikk, har jeg følgende inntrykk av hvilke muligheter en har til videre utvikling av TMA4100 Matematikk 1.

Ved Institutt for matematiske fag ser de for seg muligheten for en samkjøring med TDT4105 IT Grunnkurs om en felles kobling til de forskjellige linjefagene på sivilingeniørutdanningen. Dette skulle kunne gjøres ved at en identifiserer et sett med matematiske og numeriske verktøy som brukes i de forskjellige fagene, finne de som er mest generelle,

og gi studentene opplæring i matematisk teori og praktisk implementasjon av disse numeriske metodene. Deretter burde det gjøres tydelig hvilke kunnskaper studentene skal tilegne seg i disse fagene, slik at faglærere og koordinatorer på de forskjellige instituttene rundt om på Gløshaugen har et klart bilde på hvordan de kan ta i bruk programmering av numeriske metoder for å løse problemer spesifikke for linjefagene.

Det er viktig at det fortsatt skal være grunnleggende matematikkundervisning i TMA4100 Matematikk 1. Utvidelsen av numerikkdelen må ikke ende opp som en ren applikasjonsopplæring. Studentene må få grunnleggende forståelse av hvordan numerikk fungerer.

Tema i TMA4100 Matematikk 1 som kunne legges om mot et mer programmeringsvennlig fokus inkluderer:

- Maskinell beregning av rekker.
- Numerisk integrasjon.
- Numerisk derivasjon.
- Løsninger av differensialligninger.
- Newtons metode.
- Rekkeutvikling
- Feilestimering.

Flere av disse temaene blir allerede undervist, men beregningene gjøres for hånd og ved hjelp av kalkulator. Kompleksiteten i disse oppgavene er derfor begrenset.

Det som kreves av Institutt for fysikk, er en klar dialog angående behovene for matematiske verktøy ved de grunnleggende fysikkfagene. Dermed blir det enklere å gi studentene tilstrekkelige ferdigheter i både numerikk og programmering, slik at de i fysikkfagene kun trenger oppgaveteksten for å løse problemet.

Et konkret eksempel på hva Institutt for matematiske fag ser for seg å kunne gi studentene, er den teoretiske bakgrunnen for differensialligninger og den numeriske Runge-Kutta-metoden i TMA4100 Matematikk 1. Deretter får de opplæring i å implementere denne algoritmen i TDT4105 IT Grunnkurs. Dermed er alt en trenger for å løse det reelle pendelproblemet i TFY4145 Mekanisk fysikk å forstå fysikken bak problemet, og hvordan en setter opp differensialligningen

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0$$

for å finne pendelbanen.

Kapittel 2

CSE-reformen ved UiO

Ved universitetet i Oslo har det i en lengre periode vært arbeidet målrettet for å oppnå en tett integrering mellom numerikk, programmering og de klassiske realfagene. Dette har bakgrunn i at det på UiO tradisjonelt har vært et sterkt beregningsmiljø, og et terkt miljø for beregningstung fysikk. Dette, kombinert med at de faglig ansatte har vært svært opptatt av undervisningen, førte i utgangspunktet til at det ble påbegynt individuelle initiativ ved ulike institutter ved Det matematisk-naturvitenskapelige fakultet (MB-fakultetet) ved UiO. Over en tiårsperiode har dette utviklet seg til et koordinert og bevisst undervisningsløp. Denne reformen er blitt kalt 'Computers in Science Education', eller CSE.

Dette kapitlet skal oppsummere hvilke tiltak som har vært iverksatt ved UiO. Det skal framlegges hvilke motiver som lå til grunn for innføringen av CSE-reformen. Vi skal se på hvordan reformen har vært gjennomført rent praktisk, og hvilken betydning den har fått for faglig innhold, vitenskapelig ansatte og ikke minst studentene ved MN-fakultetet. Det kommer til å bli fokusert på studieretningen Fysik på bachelor-linjene Fysikk, astronomi og meteorologi. Dette blir gjort for å ha et best mulig sammenligningsgrunnlag med Teknisk fysikk ved NTNU. Til sist kommer en oppsummering av anbefalinger til Institutt for fysikk ved NTNU fra de involverte i CSE-reformen. Store deler av dette kapitlet baserer seg på intervjuer med Morten Hjort-Jensen, Knut Mørken, Anders Malthe-Sørenssen og Simen Sørby. Ellers har informasjon blitt hentet inn fra fagsider på UiOs nettsider[7] og korrespondanse med andre ansatte ved UiO.

2.1 Motivasjon for CSE-reformen

Store forandringer i undervisningen er både risikabelt, resurs- og tidkrevende. Derfor kreves det klare motiver for i det hele tatt å begynne en slik prosess. Etter samtaler med flere av de involverte i omveltningene ved UiO har det blitt kartlagt hvilke argumenter og motiver de involverte måtte ha for å legge inn den nødvendige innsatsen.

I likhet med faglærerne på Institutt for fysikk ved NTNU (kapittel 1.2), så var mange av initiativtagerne for CSE-reformen ved UiO bekymret over at studentene var for dårlig rustet i numeriske metoder og programmeringsevner når de begynte på master- og

doktorgradsstudiet. Konsekvensen av dette var at kandidatene brukte uforholdsmessig lang tid på å sette seg inn i grunnleggende programmering før de kunne begynne med de oppgavene som skulle ligge til grunn for oppgaven.

Moderne forskning har i større og større grad tatt i bruk numeriske simuleringer og maskinell beregning. Med tanke på hvor mange av mastergrad-studentene på realfag som ender opp i forskningsarbeid, er det viktig at de har kjennskap til de verktøyene som blir brukt. Veduniversitetene ønsker en å legge seg på en linje med forskningsbasert undervisning. Ved UiO er en konkret manifestering av dette at studentene bruker numeriske simuleringer for å illustrere fysiske fenomener.

I tillegg til det akademiske behovet for beregningskompetanse, er det i økende grad behov i næringslivet for kandidater med en sterk kompetanse på programmering og beregningsverktøy, kombinert med en solid realfaglig bakgrunn. Denne kombinasjonen lar seg vanskelig oppdrive uten også å ha en kombinert utdanning.

Foruten behovet for et kompetanseløft så også flere faglærere at numeriske beregninger nå kunne brukes som et verktøy til bedre faglig forståelse. Beregningskraft og utstyr er blitt billigere og mer tilgjengelig enn hva det var 10 – 20 år tilbake. Dette gir nye muligheter. En ligning som beskriver et system er ikke alltid like illustrerende for hvordan et system utvikler seg, eller ser ut i virkeligheten. Det er heller ikke alltid at fysikken bak er tydelig for studentene. Ved å visuelt illustrere hvordan et system utvikler seg over tid og rom, kan en få studentene til å gjøre fysiske betraktninger direkte.

I den tradisjonelle realfagundervisningen har fysiske fenomener blitt beskrevet matematisk og ligninger for systemer blitt utledet fra disse. For å få en følelse av oppførselen til disse systemene må en løse disse ligningene. Analytiske løsninger av slike ligninger er ikke alltid mulig å oppdrive. Derfor må en ofte ty til svært idealiserte forenklinger. I andre tilfeller må man ignorere enkelte fysiske fenomener fullstendig for å oppnå en analytisk løsning av ligningen. Ved å ta i bruk numeriske løsere lar en studentene bryne seg på mer generelle, og ofte mer realistiske problemstillinger innen realfag. Ved å introdusere maskinell databehandling, numerikk og programmering i studiene kan faglærerne nå la studentene arbeide med reelle data og reprodusere aktuelle forskningsresultater.

Det er de ansvarlige for CSE-reformen ved UiO sin klare oppfatning at numeriske algoritmer er en naturlig måte å tenke matematisk modellering. Dermed er dette er et nyttig verktøy som kan benyttes til å beskrive naturfaglige fenomener. Samtidig fører en slik måte å jobbe med problemer på til en algoritmisk og strukturert tankegang som er viktig i naturfag.

2.2 Implementasjon av CSE-reformen

Prosessen med å få en koordinert og grundig implementasjon av CSE har vært lang og ressurskrevende. De første spredte forsøkene med bruk av numeriske verktøy i andre realfag tok form mot slutten av nittitallet. Dette ble gjort på individuelt initiativ fra faglærere som så behovet for økt numerikk- og programmeringskompetanse blant studentene. Da det manglet en overordnet plan for denne delen av utdannelsen, lå det heller ikke til rette for at de forskjellige fagmiljøene kunne bygge på hverandres kompetanse. Dette førte til

en svært isolert og ineffektiv innsats. Dette ble det gjort noe med i 2000, etter ønske fra informatikk og fysikk om en differensiert matematikkundervisning. De kontaktet de ansvarlige for matematiske fag og fikk gehør for sine behov. Dermed ble det lagt opp tre parallelle løp med matematikk; en klassisk linje, en fysikklinje og en beregningstung linje. Etter hvert kom fysikerne fram til at den beregningstunge mattelinjen var mer optimal enn fysikkmatten. Dermed ble det redusert til to mattefag. Det ene ble fulgt av studentene fra bachelor-linjene Fysikk, astronomi og meteorologi (FAM) og Matematikk, informatikk og teknologi (MIT). Dette tvang fram en større samhandling om hva faget skulle inneholde. Studentene på bachelor-linjen Fysikk, astronomi og meteorologi følger nå MAT1100 Kalkulus, som er koordinert med og utfyller MAT-INF1100 Modellering og beregninger.

Kvalitetsreformen i 2003 sørget for at bachelor-programmene ved universitetene ble opprettet. For MN-fakultetet ved UiO betydde dette en større mulighet for samkjøring av undervisningen i ulike fagfelt. To år tidligere ble ordningen med Senter for Fremragende Forskning (SFF) opprettet. SFF er iverksatt av Norges forskningsråd. Ifølge Forskningsrådet er intensjonen at "Ordningen skal stimulere norske forskningsmiljøer til å etablere sentre viet langsiktig, grunnleggende forskning på høyt internasjonalt nivå, og har som mål å heve kvaliteten på norsk forskning."[8] SFF gis til miljøer som viser en høy vitenskapelig kompetanse. Dette medfører en administrativ samling og finansiering av disse miljøene, i utgangspunktet for en tiårsperiode[9]. For videreutviklingen av satsingen på CSE innen fysikkfagene ble opprettelsen av "Centre of Mathematics for Applications", CMA eller "Senter for matematikk for anvendelser" den mest betydningsfulle. CMA ble opprettet i juni 2002 som en av tre, nå åtte, sentre for fremragende forskning ved UiO[10]. Dette samlet kompetanse og gav en felles arena og et naturlig ankerpunkt for pådriverne for en styrket undervisning i maskinell beregning. NTNU har tre slike senter for fremragende forskning, men ingen innen matematikk eller fysikk[11]. I 2004 besluttet styrene for bachelorprogrammene Fysikk, astronomi og meteorologi og Matematikk, informatikk og teknologi at de ønsket å satse på et slik løft, og med bevilgninger fra Fleksibel læring ved UiO[12] ble prosjektet "Computers in Science Education" opprettet ved CMA.

For å konsolidere CSE-reformen ble det i 2005 vedtatt at den skulle inn i strategisk plan for MN-fakultetet 2005–2009[13]. Under punktet utdanning står det: "Fakultetet vil integrere sentrale, moderne hjelpemidler, instrumentering og teknikker for å utvide og modernisere realfagsutdannelsen. *"Numeriske beregninger og modellering har en viktig plass her."* Dette gav hjemmel for ytterligere finansiering, og et incentiv for faglærere til å utvikle fagene de underviste. En direkte måte å øke incentivene var å bruke penger bevilget prosjektet som ekstra midler til de faglærerne som gjorde en renovering av sine fag. Dette forutsatte at de tok i bruk de numeriske verktøyene som var blitt innført. Disse midlene har til nå blitt delt ut i to omganger. Den første i 2007 omfattet 9 fag, inkludert to nyoppstartede kurs, og beløp seg tilsammen til 420.000 NOK[14]. Den andre i 2008 omfattet 8 fag, og beløp seg til 350.000 NOK[15]. Det er planlagt å fortsette å gi ut slike midler i løpet av 2009.

CSE-reformen startet for fullt i 2007, da faget INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser ble holdt først gang. Dette er et kurs i programmering

Emnekode	Kursnavn	Bevilgninger i NOK
MAT1110	Kalkulus og lineær algebra	46.000
INF1100	Grunnkurs i programmering for naturvitenskapelige anvendelser	146.000
MAT-INF1100	Modellering og beregninger	18.000
MAT1120	Lineær algebra	93.000
FYS-MEK1100	Mekanikk	48.000
MEK1100	Feltteori og vektoranalyse	X
MEK1500	Faststoffmekanikk	X
FYS2130	Svingninger og bølger	46.000
FYS-MENA3110	Kvantenanofysikk	48.000
AST1100	Innføring i astrofysikk	40.000
GEF1000	Klimasystemet	*
GEF2200	Atmosfærefysikk	*
GEF2210	Regionale og globale luftforurensninger	*
GEF2500	Geofysisk fluidmekanikk	*
GEO1040	Grunnkurs i programmering for geofaglige problemstillinger	*

Tabell 2.1: Tabell over fag revidert av CSE-reformen i UiO[16]. 70.000 NOK er fordelt på de to fagene merket X. 215.000 NOK er fordelt på 5 fag merket *. Totale bevilgninger 770.000 NOK.

for naturvitere, med fokus på implementasjon av numerisk beregning. Dette erstattet det mer informatikkrettede programmeringskurset INF1000 Grunnkurs i objektorientert programmering for de beregningstunge bachelor-linjene.

En komplett liste over de fag som er revidert med midler fra CSE-reformen, og hvor store summer som er tildelt er gitt i Tabell 2.1

2.3 Integrering av dataverktøy i fysikkundervisningen

For å vurdere CSE-reformens suksess, ser vi hvordan den har påvirket studenter og faglærere på studieretningen Fysikk. Studieretningen er bygd opp av emner som vist i Tabell 2.3.

Det tunge fokuset på numerikk og programmering manifesterer seg tydelig i det første semesteret. Fagene er samkjørt til å gi studentene de nødvendige ferdighetene for å kunne følge et undervisningsløp preget av CSE. Å bruke så stor del av undervisningen på å bygge opp en kompetanse innen numerisk programmering blant studentene i starten av studieløpet må følges opp om det skal kunne rettferdiggjøres. Det har heller ikke vært gratis å bygge opp og tilpasse fagene. Derfor har det vært satt av betydelige ressurser hvert år siden 2005. Fordelingen over år er vist i Tabell 2.3.

Vi skal ta en nøyere titt på hvordan fagene i første semester er bygd opp og koordinert,

Emnekode	Emnenavn
FYS2160	Termodynamikk og statistisk fysikk
EXPHIL03	Examen philosophicum
FYS1210	Elektronikk med prosjektoppgaver
FYS2150	Eksperimentalfysikk
FYS2130	Svingninger og bølger
FYS2140	Kvantefysikk
AST1100	Innføring i astrofysikk
GEF1000	Klimasystemet
FYS1120	Elektromagnetisme
MAT1120	Lineær algebra
FYS-MEK1110	Mekanikk
MEK1100	Feltteori og vektoranalyse
MAT1110	Kalkulus og lineær algebra
INF1100	Grunnkurs i programmering for naturvitenskapelige anvendelser
MAT-INF1100	Modellering og beregninger
MAT1100	Kalkulus

Tabell 2.2: Emnekoder for obligatoriske fag ved Fysikk, UiO.

Semester	Emnekode		
1	INF1100	MAT-INF1100	MAT1100
2	FYS-MEK1110	MEK1100	MAT1110
3	AST1100 / GEF1000	FYS1120	MAT1120
4	FYS1210 / FYS2150	FYS2130	FYS2140
5	FYS2160	EXPHIL03	Valgfritt
6	Valgfritt	Valgfritt	Valgfritt

Tabell 2.3: Oppbygging og gjennomføring av studieretningen Fysikk ved UiO[7]

År	Tildelte midler	Finansieringskilde
2005	400.000 NOK	UiO sentralt (Fleksibel læring)
2006	750.000 NOK	MN-fakultetet
2007	550.000 NOK	MN-fakultetet
2007	1.000.000 NOK	Kunnskapsdepartementet
2008	500.000 NOK	MN-fakultetet
2009	500.000 NOK	MN-fakultetet

Tabell 2.4: Tildeling av midler til CSE-reformen [12]

og hvilken verktøykasse studentene skal sitte igjen med etter endt semester. Deretter skal vi se på hvordan den numeriske verktøykassen blir brukt i den videre utdanningen.

2.3.1 Bygging av numerisk verktøykasse

Det første semesteret på bachelor-linjene FAM og MIT på MN i UiO er en koordinert innsats for å gi studentene en solid matematisk verktøyboks før de begir seg ut på resten av realfagsstudiet. Dette gjøres ved å holde tre kurs på ti studiepoeng hver. Dette er fagene MAT1100 Kalkulus, MAT-INF1100 Modellering og beregninger og INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser. Disse fagene har fordelt hvilke temaer som skal foreleses, samtidig som de er koordinert for å få en best mulig synergieffekt mellom kursene.

MAT1100 Kalkulus er et rent kalkulus-kurs. Her blir det tatt for seg grunnleggende kalkulus som rekker, grenser, komplekse tall, derivasjon og integrasjon.

MAT-INF1100 Modelering og beregninger tar for seg differensialligninger og hvordan informasjon blir representert i diskrete systemer, som f.eks. en datamaskin. Det fokuseres på overgangen fra kalkulus til diskrete systemer. Det gis også en grundig innføring i de numeriske feil som oppstår som følge av diskretisering. Det gjennomgås både matematiske avrundingsfeil i metodene og de feil som oppstår på grunn av tallenes representasjon i en datamaskin. Dette er kunnskap som er nødvendig for å kunne gjøre sikre beregninger maskinelt.

INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser er et rent programmeringskurs. I likhet med TDT4105 IT Grunnkurs ved NTNU tar det for seg programmering i et script-språk. I INF1100 programmeres det i Python [17]. Det fokuseres på det programmeringstekniske og språkforståelse. I motsetning til TDT4105 IT Grunnkurs, innehar ikke INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser HTML og CSS-mark up eller databaser.

Den numeriske verktøyboksen studentene skal sitte igjen med etter endt semester er

- Produksjon av grafer av høy kvalitet.
- Monte Carlo-simuleringer.
- Modellering ved hjelp av rekker og følger.
- Interpolasjon.
- Løsning av ordinære ligninger.
- Derivasjon med feilestimat.
- Integrasjon med feilestimat.
- Løsning av ordinære differensialligninger med feilestimat.

Dette er verktøy som undervises med tanke på bruk i senere kurs.

Fagene er koordinert slik at INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser ligger litt bak MAT-INF1100 Modelering og beregninger, og MAT-INF1100 Modelering og beregninger ligger litt bak MAT1100 Kalkulus i den delen av pensum som spenner alle tre fagene. Dermed lærer studentene først matematikken, så de numeriske metodene og algoritmene, før de til sist lærer å implementere dem på en best mulig programmeringsteknisk måte. Denne måten å legge opp fagene faller pent inn i hverandre, da studentene trenger tiden fra semesterstart til numerisk implementasjon kommer på bordet til å lære seg grunnleggende programmering.

For å illustrere denne måten å legge opp undervisningen bruker vi her eksempler på oppgaver i de tre fagene. I MAT1100 Kalkulus lærer studentene teorien bak derivasjon og integrasjon. Senere, i MAT-INF1100 Modelering og beregninger lærer de differensialligninger og diskretiseringen av disse. Til sist, i INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser skal studentene skrive et program som løser de diskretiserte ligningene.

I obligatorisk oppgave 2 a) og c) i MAT1100 Kalkulus skal studentene finne den deriverte og integrerte til en bestemt funksjon, i dette tilfellet

$$f(x) = \arctan x - \frac{x}{1+x}, \quad x \neq 1$$

Etter denne basisøvelsen skal studentene i obligatorisk oppgave 3 a) i MAT-INF1100 Modelering og beregninger løse en 1. ordens differensialligning med initialbetingelser, i dette tilfellet

$$\frac{dx}{dt} - x^2 = 1, \quad x(0) = 1$$

analytisk. I oppgave b) skal ligningen løses numerisk og plottes på intervallet [0,0.6] ved hjelp av 6 steg med Eulers metode. Videre skal studentene løse den samme oppgaven med mer og mer sofistikerte numeriske løsere av differensialligningen. Dette er med på å gjøre studentene i stand til å gjøre de siste oppgavene i INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser . I oppgave 11.20 i boka "Introduction to Computer Programming"[18] skal studentene implementere en klasse for 2. ordens Runge-Kutta-løser for ordinære differensialligninger (ODE) i et klasseheirarki for ODE-løsere.

Denne gjennomgangen av første semesteret på studieretning Fysikk for bachelorprogrammet FAM ved UiO har i hovedsak basert seg på informasjon hentet fra hjemmesidene til fagene MAT1100 Kalkulus[19], MAT-INF1100 Modelering og beregninger[20] og INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser[21], deriblant pensumlitteratur og øvinger.

2.3.2 Bruk av numerisk verktøykasse

Det har etter en gjennomgang av kursmateriell vist seg at alle obligatoriske fag på studieretningen Fysikk gjør bruk av dataverktøy i større eller mindre grad. De grunnleggende fysikk- og matematikefagene som går i de to første studieårene har alle øvinger som må gjøres i enten MATLAB eller Python. Python virker å ha blitt faset inn i de siste par årene, og dukket opp i øvingsopplegget første gang våren 2008 i FYS-MEK1110 Mekanikk

etter at programmeringsfaget INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser ble innført høsten 2007. For å illustrere hvor stort fokuset på numeriske beregninger har blitt i fysikkfagene kan det nevnes at begge de obligatoriske innleveringene i MEK1100 Feltteori og vektoranalyse og MAT1110 Kalkulus og lineær algebra krever numerisk programmering, og 9 av 12 obligatoriske oppgaver i FYS-MEK1110 Mekanikk følger samme mønster. Dette stiller i kontrast til de tidligste fysikkfagene ved NTNU, der maksimalt to deloppgaver i løpet av et semester krever bruk av IKT.

Et annet meget viktig moment er at IKT-relaterte spørsmål har begynt å dukke opp på eksamen. Dette fører til at studentene legger mye større vekt på å lære seg denne ferdigheten enn om de kunne ha fått full uttelling uten. Ved siste ordinære eksamen, høst eller vår 2008, ble det gitt IKT-oppgaver i fagene FYS-MEK1110 Mekanikk, MAT1120 Lineær algebra og til dels i MAT1110 Kalkulus og lineær algebra, der det begrenser seg til å gjenkjenne MATLAB-kode. For å gi et illustrerende eksempel, ligger eksamensoppgaven i FYS-MEK1110 Mekanikk vedlagt i tillegg A.1.

For å få en mer nyansert oversikt over hvordan CSE-reformen har påvirket fysikkfagene utover mengden øvinger og eksamensoppgaver har faglærer i et av de første, og kanskje det mest reviderte fysikkfaget FYS-MEK1110 Mekanikk, Anders Malthe-Sørenssen blitt intervjuet. Under intervjuet ble det diskutert hvordan de nye elementene påvirket forelesningene, øvingsoppgavene og eksamen. Det var interessant å få vite hvorvidt CSE-reformen har ført til en vanskeligere situasjon som foreleser, og ikke minst hva dette har hatt å si for studentene. Ifølge Malthe-Sørenssen har ikke CSE-reformen ført til merkbare endringer i pensum for mekanikkfaget. Fysikken som blir forelest er den samme som før. Den største endringen på forelesningene merkes ved at i stedet for å løse alle problemene analytisk på tavla, programmerer han nå også numeriske løsere og viser prosessen med projektor i timen. Denne "live" programmeringen mener Malthe-Sørenssen er en trøstende affære for studentene, da også han gjør feil og kan plages litt med kjøringen før den blir riktig. Dermed blir studentene vant til at knoting med kode er en vanlig og nødvendig del av det å programmere. Det blir ikke gitt noen stor innføring i verken programmering eller numerikk i forelesningene. Det forutsettes at studentene kan det de har lært semesteret før. En konsekvens av at problemene ikke lenger kun løses analytisk er at en ikke lengre er nødt til å bruke spesialtilfeller der en analytisk løsning er mulig. Dermed kan det undervises i langt mer kompliserte problemer. Eksempelvis dukker det opp problemer med kast i tornado og elastisk pendelbevegelse. Valget av slike problemer flytter fokuset over til mer generelle løsningsmetoder framfor spesialiserte matematiske triks.

Ifølge Malthe-Sørenssen kunne faget ha blitt forelest på samme måte selv om studentene hadde hatt betydelig mer begrensede ferdigheter i programmering og numerikk. Det hadde riktignok kommet til å kreve mer tid og innsats fra studentenes side, spesielt i øvingene. Faglærer derimot burde i det minste ha et minimum av forståelse for programmeringsspråket og numerikken som blir brukt. Alternativet måtte være et særdeles velutviklet og ferdiglaget undervisningsopplegg.

Det er hovedsaklig i øvingsopplegget en ser den store forandringen etter CSE-reformen. I FYS-MEK1110 Mekanikk har de lagt seg på en linje der øvingsoppgavene krever omtrentlig like stor innsats og tid fra studentenes side som før CSE-reformen. Oppfølgingen

av studentene derimot har økt betraktelig, og det er nå flere øvingsassistenter per student, og flere timers oppfølging per student enn det var tidligere. Slike tiltak koster penger, og det må være nok tilgjengelige datasaler.

En betenkelighet som dukket opp under intervjuet var at overgangen til generelle ligningsløsere hadde kostet litt når det gjaldt noe av studentenes fysikkforståelse. Spesielt ble dekoblingen av x- og y-koordinater i kast nevnt. Dette er et problem som burde adresseres, og om mulig omgås. Desuten mister de svakeste studentene noe av mestringsfølelsen underveis, i og med at programmering er en ny og derfor vanskeligere metode for å løse ligninger, og at et program ikke kjører før det er fullstendig fritt for syntaksfeil. Dette kan føre til en noe høyere fare for at de aller svakeste studentene slutter. Derimot mente Malthe-Sørenssen at de studentene som kom seg igjennom øvingene, hadde fått med seg mer fysikk, og dermed stilte sterkere til eksamen. Totalt sett mente han at det samme antallet studenter fullførte faget som før CSE-reformen, ved at det var et litt større frafall, men også en litt mindre andel av studentene som gikk opp til eksamen som strøk.

Øvingsopplegget i FYS-MEK1110 Mekanikk, og mange av de andre fagene revidert under CSE-reformen, er laget av studenter ansatt av fakultetet. Dette kommer av at det er lite faglitteratur og undervisningsmateriell på numerisk fysikk kommersielt tilgjengelig. Dette er et problem som det gjøres forsøk på å bøte på, og det lages nå mer slikt undervisningsmateriell enn tidligere.

2.4 Studentenes tilpasning til CSE-reformen

En vesentlig indikator på CSE-reformens suksess er hvordan studentene reagerer på den nye måten å tilnærme seg fysikk på. Dette undersøkes nå av masterstudent Simen Sørby ved UiO. Han har fulgt utvalgte studenter i kullet som begynte på bachelor-linjen FAM høsten 2008. Målet for oppgaven har vært å se på om numerisk tilnærming er en god måte å lære fysikk på, og om CSE-reformen har styrket studentenes forståelse for fysikkfagene.

Sørby har tatt for seg kursstrukturer og arbeidsmetoder i fag der CSE har blitt implementert. Gjennom spørreundersøkelser, dybdeintervjuer og observasjon av oppgaveløsning har Sørby forsøkt å finne ut hvordan studentene har opplevd det å bruke datamaskiner såpass ekstensivt, og tildels avansert, i undervisningen. Resultatene er ikke endelige, da Sørby i skrivende stund ikke har levert sin masteroppgave. De resultater og konklusjoner som blir presentert videre er derfor midlertidige, og kan ha endret seg til da. De ferdige resultatane fra undersøkelsen kan skaffes fra UiO etter mars 2010.

De studentene Sørby har fulgt i første året av studiene har virket svært fornøyde, alt sett under ett. De erfaringene som ble gjort, var at studentene som begynner på universitetet, ikke har all verdens forventninger eller formeninger om hva de skal igjennom. Derfor føler ikke studentene seg snytt for fysikkfagene i første semester, der de ikke har annet enn matematikk og programmering. De kan føle det er litt rart, og kanskje lure på hvorfor det er slik, men det er ikke observert en negativ holdning til hvordan starten av studiet er lagt opp. Studentene stiller seg riktignok spørsmålet "hvor er fysikken?" Dette løser seg i løpet av 2. semester når de tar i bruk sine kunnskaper i numerikk i

fysikkfagene.

Studentene som er med i undersøkelsen er ifølge Sørby overraskende selvstendige i bruken av blant annet Python når de gjør de numeriske oppgavene. De aller fleste studentene som fulgte de obligatoriske oppgavene i FYS-MEK1110 Mekanikk skrev sine program fra bunnen av, selv om de har tilgang på tidligere utviklet kode. Dette fører til at disse studentene også får en bedre kontroll på hva som faktisk skjer når de prøver å løse den aktuelle problemstillingen numerisk. Ifølge Sørby er det nettopp de studentene som prøver å lime sammen eller modifisere kode fra INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser, som har de største problemene med å løse oppgaven. En sterk positiv tilbakemelding fra studentene er hvordan det er lettere å motivere programmering når en anvender det på fysikkberegninger. En samlet vurdering har vært at studentene har blitt sterke på programmering.

Innføringen av et slikt tydelig fokus på programmering har likevel innført en ny utfordring for studentene. Når de kommer til et problem de ikke klarer å løse umiddelbart, har studentene nå flere alternativer når de skal begynne å feilsøke. Det kan tyde på at de fleste studentene, ofte noe forhastet, legger skylden på programkoden.Derfor begynner de å lete etter feil der, selv om problemet vel så ofte ligger i studentenes manglende forståelse av den matematiske modellen, eller deres manglende evne til å forstå resultatet. Dette fører fort til at mye tid går med til å saumfare feilfri kode når de egentlig burde studert den matematiske modellen.

Tross de problemene som er observert, mener Sørby at studien hans kan tyde på at studentene i sum har tjent på forandringene i undervisningen som CSE-reformen har forårsaket.

Undersøkelsen om studentenes tilpasning til numerisk fysikk er gjort blant studenter som fortsatt følger studiet andre semester. Det er ikke tatt noen form for høyde for såkalte "drop-outs". Dette forsterker faglærernes oppfatning av at innføringen av CSE har gjort de sterke studentene sterkere, men at de svake studentene sliter desto mer enn det de gjorde før reformen.

2.5 Anbefalinger fra UiO

Under intervjuer og møter angående CSE-reformen i Oslo kom det fram en rekke råd og anbefalinger basert på førstehåndserfaring for hvordan en større undervisningsreform kan gjennomføres. Først blir det presentert råd for å få aksept og muligheter for å gjennomføre en slik reform på et organisatorisk plan. Deretter kommer en del idéer om hvordan en CSE-reform burde gjennomføres rent teknisk og faglig.

I utgangspunktet var de av faglærerne som ønsket seg et sterkere fokus på maskinelle beregninger, alene med sine tanker og idéer. Dette førte som tidligere nevnt til en isolert innsats som lå i fare for å dø hen på grunn av manglende resultater. Dette endret seg drastisk da de kreftene som ønsket en forandring, kom i kontakt med hverandre, både innad og på tvers av fagmiljøer. En stor institusjon som et universitet er ofte tungrodd, og alene er det vanskelig å få skapt varige endringer. Derfor anbefalte de ansatte ved UiO på det sterkeste å starte hele prosessen med å finne de alliansepartnerne som måtte

kunne være med på å forme en CSE-reform ved NTNU.

CSE-reformens ildsjeler hadde en todelt strategi for å få med seg resten av fakultetet. For det første var det avgjørende at de fikk med seg ledelsen på at numeriske beregninger i realfagundervisningen burde bli et satsingsfelt. I og med at UiOs organisasjon er noe annerledes bygd opp enn på NTNU, var en godkjennelse og støtte fra fakultetsledelsen tilstrekkelig for et slikt omfattende arbeid. Avhengig av ambisjonsnivå og størrelse for eventuelle lignende tiltak ved NTNU, må Forvaltningsutvalget for Sivilingeniørutdannelsen (FUS) kobles inn.

For endelig å forankre CSE-reformen og dens støtte fra ledelsen, ble disse intensjonene som tidligere nevnt nedfelt i MN-fakultetets strategiske plan. Dette gav hjemmel til både finansiering og faglig styring, noe som er helt nødvendig om hele undervisningsløpet skal bli formet etter CSE-visjonen. Rådet var å gjøre dette ved NTNU også, slik at en hadde dokumenter å vise til når en møtte motvilje.

Den andre og minst like viktige delen av strategien for aksept for CSE-reformen gikk ut på å skape interesse, entusiasme og innsatsvilje blant de berørte faglærerne. Dette er viktig, for selv om en kan tvinge noen til å gjøre en jobb, kan en ikke tvinge noen til å gjøre en *god* jobb. De gode resultatene i Oslo stammer ifølge de ansvarlige for CSE-reformen selv i stor grad fra det faktum at hele prosessen har foregått ved konsensus fra de berørte. Den beste måten å skape en slik entusiasme, så langt personalet i Oslo har erfart, er personlig kontakt framfor formelle møter. Her er det viktig å få fram nettopp hvorfor hvert enkelt fagfelt kan tjene på en CSE-reform. Eventuelt må en tydeliggjøre hvor stor rolle de kan ha for reformens suksess. Det er viktig å ikke detaljstyre de delaktige for mye, selv om en er klar på hva en ønsker å oppnå.. Dette hindrer dem å utvikle en eierskapsfølelse for prosjektet. Dermed risikerer en å drepe det gryende engasjementet.

For å kunne gjøre en god jobb er det ikke alltid nok med interesse og engasjement. En må ha tilgang på de ressurser som kreves. Ved UiO har de hatt svært gode erfaringer med å premiere de faglærerne som gjør en innsats for utviklingen av CSE-programmet. Dette har vært i form av de monetære bevilgningene som vist i Tabell 2.1.

Pådriverne for CSE-reformen er overbevist om at det er denne kombinasjonen av støtte fra ledelsen og viljen hos de vitenskapelig ansatte til å bidra, som har ført til at de både har fått de nødvendige midlene og fått gjennomført sine visjoner. Dermed har reformen til nå har vært en suksess. Skal en få til en CSE-reform ved NTNU, må begge momentene ligge på plass.

Det har vært gjort forsøk på å holde fag som underviser i programmering og numerikk, av fysikere. Den erfaringen som er høstet er enkel: La ekspertene på matematikk lære studentene numerikk og la ekspertene på informatikk lære studentene å programmere. Dette fører til en bedre undervisning i disse emnene. Dermed kan faglærere i fysikk konsentrere seg om å lære bort det de kan best, nemlig fysikk. På spørsmål om hva de mente fungerte best; en langsgående, stegvis innføring i numerikk og programmering, eller en rask og bestemt opplæring i form av egne fag, mente personalet bestemt at det var det siste alternativet som var det beste.

Når CSE skal innføres i slik stor skala som det er gjort i Oslo, kan det føre til mange kokker og mye søl. Hvis undervisningen i de forskjellige fagene legges opp hver for seg,

risikerer en at enkelte tema blir repetert til det kjedsommelige. Andre tema kan mangle helt. Dermed skapes hull i den kompetansen en ønsker at studentene skal tilegne seg. Det kan også være vanskelig å sikre progresjon oppover i årskursene. Derfor er det viktig å tenke nøye igjennom hva en ønsker studentene skal lære. Dette må settes inn i en overordnet plan som burde omfatte hele studieløpet.

Det er viktig at det er tydelig for studentene at det er meningen de skal lære seg de numeriske verktøyene. En fare er at studentene tar lett på de numeriske oppgavene og ser på dem som en underordnet del av fysikkfagene. Derfor har de i UiO obligatoriske numerikkoppgaver i kursene. For å endelig understreke at numeriske løsninger er pensum, har UiO som nevnt begynt å innføre numeriske algoritmer og programmeringsspørsmål i eksamensoppgavene. Arbeidet fram til eksamen er en oppsummering av det stoffet som er viktig. Hvis numerikk aldri dukker opp på eksamen, blir det dermed ansett som uvesentlig av studentene. Det er viktig å gi et slikt tydelig signal om at numerikk ikke er stoff studentene kan omgå. Det er også lettere for studentene å huske tilbake ved senere årskurs de emnene som har vært tydelig definert i pensum. Dermed kan en lete seg direkte tilbake til en spesifikk forelesning, spesifikk øving eller et spesifikt kapittel i læreboka.

De vitenskapelig ansatte i Oslo mente det var viktig å få den numeriske verktøykassen tidlig inn i studiene. Dermed kan den brukes jevnlig gjennom studieløpet. Dette fører til en lengre drilling, og sterkere studenter. Desuten gir en tidlig start muligheten til å tilegne seg en høy spisskompetanse mot slutten av studiet. Her er det viktig at eksponeringen begynner på et lavt vanskelighetsnivå, og at studentene får stoffet repetert gjennom senere bruk.

For å kunne holde de reviderte fysikkursene er det viktig at faglærerne også er oppdatert og kompetente. MN-fakultetet ved UiO tilbyr sine professorer lynkurs i flere programmeringsspråk og beregningsverktøy. Dette var en ordning de mente burde eksistere også ved NTNU.

Som en siste advarsel og råd ble det klart påpekt at en eventuell innføring av CSE i fysikkfagene måtte ha klare pedagogiske mål. Numerikk og maskinelle beregninger må hjelpe fram den fysiske forståelsen. Den må på ingen måte komme i konkurranse med fysikken om studentenes oppmerksomhet. Når de organisatoriske problemene først er løst, så er det dette som er det vanskeligste i den praktiske implementasjonen av CSE-reformen.

Kapittel 3

Oppgaveløsning med IKT

En CSE-reform må nødvendigvis bringe med seg oppgaveløsning med IKT-verktøy. Derfor skal vi i dette kapitlet ta en nærmere titt på hvilke typer oppgaver og hvilke verktøy som kan være aktuelle. Deretter skal vi se på hvordan studenter har reagert på forskjellige oppgaver.

3.1 Eksempeloppgaver

Vi har operert med tre hovedtyper av oppgaver. Dette er en kunstig oppdeling, og mange oppgaver flyter gjerne i grensene mellom disse. Denne oppdelingen er likevel gjort for å ha en konkret framstilling av hvordan IKT-relaterte oppgaver kan se ut. Eksemplene under er tatt fra undervisningen ved UiO.

3.1.1 Numerisk beregning

Et eksempel fra den første gruppen med oppgaver, hvor studentene skal skrive numeriske løsere på egenhånd, er i faget FYS-MEK1110 Mekanikk som tas 2. semester. I obligatorisk oppgave 4 blir studentene konfrontert med en elastisk pendel. Her skal studentene i første omgang identifisere de kreftene som påvirker kulen som henger i enden av den elastiske tråden ved null utslag. Deretter skal studentene sette opp et matematisk uttrykk for de samlede kreftene ved et hvilket som helst utslag og den tilhørende akselerasjonen av kula. Så skal studentene forklare hvorvidt vinkelen er tilstrekkelig for å beskrive posisjonen til en elastisk pendel og hvilken posisjon kula har ved null vinkelutslag.

Når så disse innledende fysikkoppgavene er fullført, begynner programmeringen og numerikken. Først skal studentene skrive et program som beregner posisjonen til en pendel de første ti sekundene etter at den blir sluppet fra en vinkel på 30° og avstand L fra origo. Deretter skal studentene kjøre programmet, plotte og betrakte resultatene. Til sist skal studentene plotte strekkraften og vinkelen over tid, og strekkraften som funksjon av tiden. Deretter kommer en del frivillige ekstraoppgaver som i all hovedsak dreier seg om forskjellige modifikasjoner av simuleringen. Hele oppgaveteksten står gjengitt i tillegg A.2

3.1.2 Visualisering

For å illustrere oppgavene der studentene skal visualisere fysiske fenomener, vises her et eksempel fra faget MEK1100 Feltteori og vektoranalyse, som holdes i 2. semester. I obligatorisk oppgave 1 bes studentene å ta i bruk MATLAB for å visualisere hastighetsfelt og strømlinjer. Hastighetsfeltet er i dette tilfellet definert ved

$$u = \cos(x)\sin(y), \quad v = -\sin(x)\cos(y)$$

der u og v er hastigheten i henholdsvis x- og y-retning.Dette gir strømlinjefunksjonen

$$\psi = \cos(x)\cos(y)$$

Studentene får utlevert kildekoden for strømlinjefunksjonen, og skal i deloppgave 3 a) bruke denne for å plotte strømlinjene i et 2D-plott med forskjellige antall diskretiseringspunkter. I 3 b) skal de skrive en egen funksjon for det oppgitte hastighetsfeltet og bruke denne for å produsere et vektorplott. I oppgave 3c) skal studentene skrive en funksjon for numerisk beregning av hastighetsfeltet ved hjelp av den oppgitte strømlinjefunksjonen, den innebygde MATLAB-funksjonen *gradient* og relasjonen

$$\vec{u} = \nabla\psi$$

Dette er egentlig en numerisk oppgave, men studentene bruker en "black box" funksjon, og skriver ikke løseren selv. Til sist, i oppgave 3 d), skal det plottes den numeriske feilen som blir gjort i utregningen av hastighetsfeltet ved ulike antall diskretiseringspunkter. Dette gjøres ved å plotte $\vec{u} - \vec{u}_{num}$.

Hele oppgaveteksten står gjengitt i tillegg A.3.

3.1.3 Databehandling

Som et eksempel på oppgaver i databehandling har vi oppgavene i FYS2150 Eksperimentalfysikk, som holdes i 4. semester. Her brukes MATLAB aktivt for å behandle måledata fra laboratorieforsøk. Et konkret eksempel er laboratorieoppgaven "Tid og frekvens". Målet med denne oppgaven er å forstå normalfordeling, middelverdi og standardavvik. Dertil hører det med å vurdere om måledata er normalfordelte.

Utgangspunktet for denne laboratorieøvelsen er at studentene skal skrive ferdig et ufullstendig MATLAB-skript. Studentene skal generere normalfordelte datasett med kjent middelverdi og standardavvik. Deretter blir studentene bedt om å skrive kommandoer som estimerer disse middelverdiene og standardavvikene. Disse blir brukt for å definere et konfidensintervall. Studentene blir oppfordret til å finne egnede innebygde funksjoner i MATLAB på grunn av den store forskjellen i kjøretid. Samtidig blir de bedt om å forklare hva som foregår i de ferdigskrevne funksjonene ved hjelp av *help*-funksjonen eller MATLABs dokumentasjon på nettet. Videre skal studentene visualisere både kumulativ og vanlig fordeling. Avslutningsvis skal det avgjøres hvorvidt datasettet virkelig er normalfordelt.

Etter at skriptet er ferdigskrevet skal studentene øves i forskjellige metoder for å beregne tid ved hjelp av periodiske hendelser. Disse hendelsene har forskjellige kilder til feil, og dermed varierer oppløsningen og nøyaktigheten som tidsmåler. Først skal en pendel brukes til å kalibrere et timeglass. Perioden til pendelen blir beregnet ut fra snorlengde og gravitasjonsakselerasjonen. Deretter skal svingetiden bestemmes med gjentatte målinger med stoppeklokke. Her skal det bestemmes hvorvidt målingene er normalfordelte, og så hvor stor presisjon målingene har ved å se på standardavviket. Dette skal legge grunnlaget for en vurdering av nøyaktigheten til timeglasset. Til sist skal den samme prosedyren gjentas med en fotodiode med en gitt samplingsrate som tidsmåler i stedet for stoppeklokken.

Denne øvelsen skal danne grunnlaget for en rapport som skal leveres og vurderes med karakter. (Studentene kan velge hvilken av labøvingene som skal rapporteres.) Denne typen oppgaver skal sette studentene i stand til å foreta maskinell analyse og presentasjon av eksperimentelle data.

Hele oppgaveteksten står gjengitt i tillegg A.4.

3.2 Verktøy for oppgaveløsning

I teorien kan en løse alle typer oppgaver for hånd, med papir og blyant som eneste hjelpemiddel. Det er likevel noe upraktisk å sitte med et problem som krever flere hundre millioner iterasjoner for å løse. Tiden det vil ta å gjøre slike beregninger for hånd kan være et helt liv. Sannsynligheten for at noe er blitt feil underveis er også overhengende. Det kan derfor være greit å ta noen hjelpemidler i bruk for å kutte ned tiden og øke nøyaktigheten.

Før i tiden brukte en regnestav for å gjøre enkle beregninger. Dette instrumentet sørget for at det gikk en smule raskere enn om en hadde gjort alle utregningene for hånd. Da den elektroniske lommekalkulatoren ble allmemanseie på slutten av 70-tallet, ble tiden hver beregning tok mer enn halvert. Selv om disse verktøyene har senket beregningstiden, så tar det fortsatt betraktelig lengre tid å gjøre de nevnte millioner av kalkulasjoner enn det en har til rådighet.

Moderne datamaskiner derimot kan gjøre disse hundretalls av millioner utregninger i løpet av brøkdelen av et sekund. Dermed åpner datamaskiner opp for praktisk bruk av numeriske metoder. Det vi skal fokusere på i dette delkapitlet er hvilke verktøy vi har tilgjengelig for å få maskinene til å gjøre disse utregningene for oss. Vi opererer her med tre båser for verktøyene. Dette er en overforenkling, men det hjelper med å se i grove trekk hvilke verktøy som passer til hva.

De tre typene av verktøy vi skal gå igjennom er programmeringsspråk, biblotekbaserte skriptspråk og spesialiserte beregningsprogrammer. Det vil under bli gitt en beskrivelse og eksempler på de forskjellige verktøyene. Deretter vil det bli utredet hvilke fordeler og ulemper som er forbundet med dem, og hvilke type oppgaver som verktøyene passer til.

3.2.1 Programmeringsspråk

Programmering går i bunn og grunn ut på å be datamaskinen om å gjøre akkurat det du ber den om å gjøre. Datamaskinen forstår bare det språket den er bygd opp til å forstå. Dette er et sett med svært enkle kommandoer i stil med "flytt ladning z fra x til y". Det krever dessuten at disse kommandoene er skrevet i maskinkode. For å kommunisere enklere med maskinene er det laget programmeringsspråk. Dette er formelle språk som er forståelig for mennesker, og som ved hjelp av kompilatorer kan oversettes til maskinkode. Dette letter kommunikasjonen med datamaskinen betraktelig. Eksempler på slike programmeringsspråk er FORTRAN, C/C++, Java og Python.

Det finnes i hovedsak to måter å kjøre programmer skrevet i slike språk. Den første er å skrive hele programmet, for så å oversette det hele med en kompilator, slik at det hele blir lagt i en kjørbar fil av maskinkode. Et eksempel på en slik type språk er C++. Den andre er å kjøre et program som tar i mot linje for linje med kode, oversetter og kjører direkte etterhvert som den leser kode. Sistnevnte type språk blir ofte omtalt som scriptspråk. Et eksempel på en slik type språk er Python. Siden sistnevnte er tregere enn språk som kompileres, blir disse som regel brukt til å lime sammen ferdigkompilerte bibliotek, og vi skal videre snakke om språk som kompileres.

Den største fordelen med språk som FORTRAN og C/C++ er at programmer skrevet i disse kjører raskt. Dette gjør dem til velegnede verktøy til å kjøre numeriske algoritmer, spesielt der mange iterasjoner er påkrevd. Det er imidlertid noe rigid, og ikke minst komplisert å skrive ned alle kommandoene, så det kreves en viss kunnskap, nøyaktighet og ikke minst tålmodighet for å bruke dette verktøyet.

Denne type verktøy passer derfor best til tunge beregningsoppgaver der algoritmene skal kodes av studentene selv, og problemer som ikke kan løses ved hjelp av allerede velskrevne bibliotek. Faget TDT4102 Prosedyre- og objektorientert programmering tar for seg denne type verktøy.

3.2.2 Bibliotekbaserte verktøy

Som nevnt i delkapitlet over, er scriptspråk relativt trege, spesielt når det kommer til iterasjoner. Dette er fordi koden må oversettes til maskinkode for hver kjøring, linje for linje. Styrken til mange av disse språkene er at de rimelig ukomplisert kan importere kompilerte bibliotek, og kjøre kommandoer derfra med en uforminsket hastighet. Dette gjør det enkelt å gjøre beregninger og oppgaver som allerede har veldefinerte løsere. Å produsere en lesbar graf er for eksempel rimelig enkelt i MATLAB.

Fordelen med å bruke slike pakker, er at en ikke trenger et like høyt kompetansenivå i programmering. Det tar også kortere tid å kalle en Runge-Kutta-rutine enn det tar å skrive den selv. En kan da sette opp problemet slik at studentene ikke trenger full kjennskap til hvordan de numeriske metodene fungerer. Det samme kan også betraktes som en ulempe, da studentene kan komme til å bruke rutinene ukritisk, og dermed få gale eller ingen svar. Desuten koster en del slike biblioteksløsninger, som f.eks. MATLAB, en god slump med penger. Andre igjen, kan være et lappeteppe av bibliotek, og det kreves en del kunnskap og innsats for å skaffe alle en trenger for å løse et problem.

Slike bibliotekløsninger passer best til oppgaver i databehandling og beregning som har kjente løsere, som for eksempel ordinære differensialligninger, og som ikke krever manuell implementasjon av algoritmen. Denne typen verktøy er også ideell for presentasjon av data og andre typer plott. Faget TDT4105 IT Grunnkurs tar for seg denne type verktøy.

3.2.3 Spesialiserte beregningsprogrammer

Det er laget en del ferdigutviklede beregningsverktøy. Eksempler på disse er PSpice som brukes i TFY4185 Måleteknikk, SPARTAN som blir benyttet i TFY4215 Kjemisk fysikk og kvantemekanikk, og Comsol, et verktøy for løsning av partielle differensialligninger benyttet av flere av de vitenskapelig ansatte på Institutt for fysikk.

Disse programvarene er i større eller mindre grad spesialiserte til et bestemt formål. PSpice er for eksempel beregnet på simuleringer og design av elektriske kretser. Comsol sine applikasjoner spenner bredere, men er konsentrert rundt løsning av differensialligninger med grensebetingelser. Fordelen med denne spesialiseringen er at uhyre kompliserte beregninger kan gjøres uten altfor inngående kunnskap og innsats. Dermed kan intrikate elektriske kretser designes og kompliserte fysiske fenomener illustreres. "Black box"-problematikken er enda større med disse verktøyene enn biblotek-vektøyene beskrevet i delkapitlet over, så de kan ikke brukes i oppgaver der forståelse av de numeriske algoritmene er poenget. Ethvert verktøy må brukes til sitt formål. Denne typen verktøy er også oftere lukket programvare som koster penger.

Denne type verktøy passer best til illustrasjon av komplisert fysikk, som f.eks. i TFY4215 Kjemisk fysikk og kvantemekanikk, og design, som f.eks i TFY4185 Måleteknikk.

3.3 Forsøk med numeriske oppgaver

Det ble tidlig i masteroppgaven gjort tre forsøk med IKT-oppgaver i fysikkemner. Resultater og erfaringer fra disse er beskrevet i de følgende underkapitlene.

3.3.1 Øvingsoppgaver i TFY4155 Elektromagnetisme

Det første forsøket med en implementering av numerikk under masteroppgaven var en enkel plotteoppgave under en øving i TFY4155 Elektromagnetisme. Den baserte seg på en elektrisk dipoloppgave fra øvingen uken før, der studentene beregnet det faktiske elektriske feltet fra en reell dipol, og etterpå feltet fra en ideell dipol. Oppgaven gikk ut på å plotte potensialet fra den reelle og ideelle dipolen, for så å plotte den relative forskjellen mellom dem, for å visualisere når den ideelle dipolen er en fornuftig tilnærming til det reelle problemet.

Oppgaven og løsningsforslaget var enkle å lage. Det tok ca. fem timer. Det ble anslått at studentene skulle bruke rundt halvparten av øvingstiden på denne oppgaven. Undertegnede og faglærer mente at dette burde være oppgaver som var enkle nok til at de fleste skulle klare å gjennomføre dem, sett i lys av hva de skal lære i TDT4105 IT Grunnkurs og i labøvingene for TFY4145 Mekanisk fysikk.

Det vi oppdaget under veiledningstimen for øvingene, var at ingen stilte et eneste spørsmål angående MATLAB-delen av øvingen. Dette gjenspeilte seg i innleveringene der bare to av femten innleveringer inneholdt en besvarelse av MATLAB-oppgaven. Dette tyder på at denne framgangsmåten splitter studentene i to grupper: De som er usikre i programmering, og derfor hopper over oppgaven, og de som føler seg komfortable med programmering, og dermed klarer oppgavene uten hjelp. Oppgaven finnes i sin helhet i tillegg B.1

3.3.2 Laboppgaver i TFY4155 Elektromagnetisme

Laboratorieoppgavene i TFY4155 Elektromagnetisme har lenge inneholdt oppgaven "Kraft på strømførende ledning" der studentene skal måle den magnetiske kraften på en strømførende ledning som funksjon av strømmen, og dermed finne fysiske parametre ved hjelp av lineær regresjon. I oppgaven før dette har studentene tatt for seg et fiktivt datasett og øvd seg på dette i Excel sine innebygde funksjoner. Teorien for lineær regresjon blir oppgitt, men ikke brukt i denne oppgaven.

Derfor bestemte vi oss for å gi studentene en utvidet oppgave. Nå skiftet vi ut de innebygde funksjonene i Excel med MATLAB, og i stedet for å be dem om å bruke ferdiglagede funksjoner, måtte de nå skrive et program for å utføre en lineær regresjon ved bruk av "minste kvadraters metode". Studentene fikk, i tillegg til algoritmen til regresjonsmetoden, en pseudokode, slik at de hadde en viss anelse om hvordan en skulle skrive programmet. Dette var ett nivå høyere enn en lignende laboppgave i TFY4145 Mekanisk fysikk, der hele kildekoden var gitt i labheftet. Den ferdige koden skulle testes på et forhåndsgitt datasett. Når studentene hadde sjekket at koden gav forventede resultater, kunne de anvende den på det reelle eksperimentet i neste oppgave.

De fleste studentene greide å skrive regresjonsprogrammet selv med litt eller ingen hjelp utover labteksten. Noen behøvde en del mer hjelp, spesielt med MATLAB-tilknyttede problemer som å definere en funksjon o.l. Oppgaven i sin helhet er utgitt i labheftet for TFY4155[22]

3.3.3 Prosjektoppgave i FY0001 Brukerkurs i fysikk

I FY0001 Brukerkurs i fysikk, arrangerte faglærer Tor Nordam en kort prosjektoppgave i numerisk fysikk. Dette ble gjennomført på frivillig basis, men med den gulroten at godkjent prosjektrapport telte som to godkjente øvinger. Det var 7 studenter av 42 som deltok på prosjektet. Dette var studenter som ikke hadde hatt opplæring i bruk av programmering. Derfor ble det også skrevet et lite hefte på 7 sider med rask innføring i programmeringskonseptet. Heftet er å finne på fagets hjemmeside[23].

Oppgaven som ble gitt var todelt, og studentene kunne velge en. Den første oppgaven var å simulere et planetsystem. Den andre oppgaven var å simulere et skrått kast med luftmotstand. Det var satt av tid til veiledning tre ganger to timer over to uker. Det var litt vanskelig å dimensjonere oppgavestørrelsen når studentene ikke hadde hatt TDT4105 IT Grunnkurs, men vi antok at med seks timers veiledning skulle alle klare å komme seg i mål. Dette var en typisk beregningsoppgave.

Begge oppgavetekstene begynner med å definere kreftene som påvirker systemet på vektorform. Deretter utledes hvordan dette bestemmer akselerasjonen. Ut ifra dette vises algoritmen som brukes for å finne posisjon etter en tid, Δt. Dermed skal studentene ha nok informasjon til å kunne kjøre en enkel simulering av systemet. Det ble først holdt en times forelesning i de tekniske kunnskapene som var nødvendige for å kunne fullføre oppgaven. Når de i tillegg fikk utlevert heftet vi skrev, skulle de kunne bryte løs på egenhånd.

Studentene prøvde i utgangspunktet å gjøre det meste selv, men bad om hjelp når de hadde stått fast en stund. Det var en balansegang å veilede dem. Gav en dem for vage hint, satt de fortsatt som spørsmålstegn og kom seg ikke videre. Hvis en derimot gav dem for mye hjelp og forklarte nøyaktig hva de skulle skrive, så fikk de riktige resultater, men de forstod ikke hva som hadde skjedd, og kunne ikke analysere resultatet på noen god måte.

Alle de 7 studentene som leverte, hadde produsert gode resultater og forstått dem. Alle fikk derfor godkjent oppgaven. Prosjektoppgaven ligger i sin helhet i tillegg B.2.

Kapittel 4

Oppsummering og anbefalinger

I dette kapitlet skal vi først oppsummere de viktigst momentene fra de undersøkelser og drøftinger som er gjort. Deretter kommer en anbefaling av tiltak basert på det som har kommet fram i løpet av denne masteroppgaven.

4.1 Oppsummering

Dette delkapitlet er en gjennomgang av de resultater som har kommet fram etter undersøkelsene gjort ved NTNU og UiO. Først skal vi oppsummere hva som har skjedd i forbindelse med CSE-reformen ved UiO. Deretter skal vi kort gå igjennom hvordan situasjonen ved Institutt for fysikk ved NTNU er per dags dato.

Bakgrunn for oppgaven

Sivilingeniørutdannelsen ble i 2008 gått nøye gjennom i sømmene. Dette resulterte i Steinbach-rapporten som tydelig konkluderer med at siv.ing.-linjene på NTNU har mye å hente når det kommer til bruk av IKT i undervisningen. Denne masteroppgaven er skrevet for å kartlegge dagens bruk av IKT på Teknisk fysikk ved NTNU. Hensikten med dette har vært å finne løsninger som kan bøte på de mangler i IKT opplæringen, som påpekes både i Steinbach-rapporten og studielinjens egenevaluering.

Motiv for CSE-reformen ved UiO

I moderne forskning er det et stort behov for kompetanse på numerikk og maskinelle simuleringer. Det var en oppfatning blant faglærerne ved UiO at studentene var for svake i nevnte tema når de startet på master- og/eller doktorgradsstudiet. Dette dannet motivasjonen blant faglærerne på MN-fakultetet ved UiO for å skape et løft for maskinelle beregninger i undervisningen. Det var også et økende større påtrykk fra næringslivet, som etterspurte disse ferdighetene. Det at personalet i Oslo bestemt mener at en innføring av numerikk i blant annet fysikkfagene har ført til en pedagogisk gevinst, har bare forsterket motivasjonen for CSE-reformen etterhvert som den har tatt form.

Implementasjon av CSE-reformen ved UiO

Hele prosessen har sin forankring i et beregningstungt fagmiljø ved UiO. Dette, kombinert med vitenskapelig ansatte som har sterk interesse av utdanningstilbudet, førte til personlige initiativ for å få innført numeriske øvinger i realfagskurs. Disse elementene fant hverandre og begynte en koordinert innsats for å få gjennomført en reform for IKT-opplæringen. Ved å skaffe støtte fra ledelsen og skape engasjement blant faglærere fikk de startet CSE-reformen. Dette ble offisielt en del av MN-fakultetets satsingsområder da det kom med i strategisk plan for 2005 – 2009. Dette gav hjemmel for å skaffe midler til renovering av fagene ved de berørte studieprogrammene. Det viktigste var kanskje utviklingen og opprettelsen av et eget numerikkfag, MAT-INF1100 Modellering og beregninger.

For bachelor-programmet Fysikk, astronomi og meteorologi betydde dette at første semester ble helt og fullt dedikert til den nye reformen. Studentene skulle nå igjennom fagene MAT-INF1100 Modellering og beregninger, MAT1100 Kalkulus og INF1100 Grunnkurs i programmering for naturvitenskapelige anvendelser. Disse fagene er koordinert for å kunne gi en best mulig synergieffekt. Etter 1. semester er fagene lagt om slik at studentene må nyttiggjøre seg de matematiske og informatikkmessige ferdighetene de skal ha tilegnet seg i 1. semester. Alle de obligatoriske fagene på studielinjen Fysikk gjør i større eller mindre grad bruk av IKT.

Disse verktøyene gir muligheter for helt andre oppgaver enn før. Nå trenger ikke faglærerne finne spesialtilpassede problemer i den hensikt at løsningen må kunne regnes ut analytisk. Det gir også en større mulighet for å trekke dagsaktuell forskning inn i undervisningen. Majoriteten av studentene har reagert positivt på innføringen av numeriske beregninger i fysikkfagene. Det nye abstraksjonsnivået har likevel ført til større problemer for de svakeste studentene. Dette har ført til en større andel av studentene som avslutter studiene før tiden. Samtidig står en større andel av studentene som møter til eksamen.

Anbefalinger fra UiO

De anbefalinger som ildsjelene for CSE-reformen har kommet med, innebærer å samle de kreftene som er for en utvikling av en CSE-reform ved NTNU, dernest å sørge for en solid forankring hos ledelsen, samt å forsøke å skape en entusiasme blant de faglærerne som vil bli berørt av reformen. Det kom klare anbefalinger om å starte integreringen av IKT tidlig i studiet, slik at en kunne bygge kompetanse blant studentene over flere år. Dette mente de burde gjøres etter modell av 1. semester for Fysikk ved UiO. Det vil si at de ferdighetene som skal brukes senere, burde komme i egne fag, i motsetning til å prøve å bake det inn i fysikkundervisningen opp igjennom studieløpet. Det er viktig å motivere programmering og numerikk ved å vise at det er nyttige verktøy for løsning av fysiske problemer. Til sist men ikke minst må det nevnes at personalet i Oslo var helt klare på nødvendigheten av en overordnet plan for CSE-reformen. Uten en slik plan risikerte en at hele opplegget endte opp som et lappeteppe der noen tema gjentok seg, og andre manglet.

Opplæring og bruk av IKT for studenter på Teknisk fysikk, NTNU

Den formelle IKT-undervisningen for Teknisk fysikk består av to emner, TDT4105 IT Grunnkurs og TDT4102 Prosedyre- og objektorientert programmering. Disse fagene går henholdsvis i 1. og 4. semester. I TDT4105 lærer studentene det bibliotekbaserte skriptspråket MATLAB og grunnleggende programmering. Dette kommer i tillegg til generell IKT-teori. I øvingsopplegget er det nå innført en del oppgaver som har tilknytning til TMA4100 Matematikk 1. Dette innebærer at studentene blir eksponert for datamaskinen som et verktøy for beregninger. I TDT4102 Prosedyre- og objektorientert programmering får studentene en grundigere opplæring i programmering. Faget baserer seg på programmeringsspråkene C/C++. Faget er et rent programmeringskurs og inneholder ingen form for beregninger eller simuleringer direkte relevant for fysikkoppgaver. Imidlertid får studentene en god opplæring i programmeringstekniske tema.

Etter en gjennomgang av undervisningsløpet for studieretningen Teknisk Fysikk på Fysikk og matematikk kan en konkludere med at bruk av IKT i hovedsak skjer i prosjekt- og masteroppgave. Det brukes noe spesialisert programvare for illustrasjon og kompliserte beregninger og design. Dette skjer i TFY4215 Kjemisk fysikk og kvantemekanikk (SPARTAN brukes til Hartree-Fock-beregninger av mangepartikkel-system) og i TFY4185 Måleteknikk (PSpice brukes til elektronisk kretsdesign). I laboratorieoppgavene kreves det noe plott og enkel databehandling. Vi kan se et skifte fra bruk av Excel til MATLAB. Det brukes også en del spesialdesignet programvare som drivere for digitalt måleutstyr. På toppen av dette kan vi se en økt bruk av MATLAB i øvingsoppgavene i de tidlige fysikkoppgavene det siste året. Dette innebærer plotteoppgaver i TFY4160 Bølgefysikk og TFY4155 Elektromagnetisme, og en beregningsoppgave i TFY4215 Kjemisk fysikk og kvantemekanikk. Dette er individuelle initiativ som burde bli oppmuntret og tatt med i en helhetlig plan for IKT-undervisningen ved sivilingeniørlinjen Fysikk og matematikk. Noen av de valgbare emnene på høyere årskurs nyttegjør i større grad IKT-verktøy i undervisningen.

Undersøkelse blant vitenskapelig ansatte ved Institutt for fysikk

De vitenskapelig ansatte på Institutt for fysikk var bekymret over det svært varierende nivå av programmering- og numerikkferdigheter blant studentene som begynner på prosjekt- og masteroppgave. Selv hadde de også noe varierende ferdigheter i programmering. Det kunne med fordel ha blitt innført et tilbud om kursing i de viktigste IKT-verktøyene, skulle det komme til en bred innføring av en CSE-reform. MATLAB var det dominerende verktøyet i bruk blant de vitenskapelig ansatte. De var hovedsaklig positive til en oppgradering av studentenes IKT-ferdigheter, men med forbehold om at det ikke måtte gå på bekostning av fysikkundervisningen.

Undersøkelse blant nyutdannede fra Teknisk fysikk

Av de som gikk ut fra Teknisk Fysikk sommeren 2008, hadde 97% fått jobb, og av disse hadde 74% en jobb direkte koblet til numeriske beregninger av et eller annet slag. Dette

vil enten si bruk av spesialisert programvare eller egenkomponert kode. De fleste av disse var relativt misfornøyd med IKT-opplæringen de hadde fått på NTNU. Det de hadde lært seg, hadde de lært på egenhånd. De stilte seg sterkt positive til en satsing på IKT i fysikkfagene. Dessuten mente de aller fleste at byttet fra TDT4115 IT Grunnkurs med JSP, til TDT4105 IT Grunnkurs med MATLAB, og byttet fra faget TDT4100 Objektorientert programmering, som brukte Java, til det C/C++ fokuserte TDT4102 Prosedyre- og objektorientert programmering var udelt positivt. Enkelte mente at dette var nok til å tette de hullene i IKT-undervisningen som fantes på Fysikk og matematikk.

Undersøkelse blant masterstudenter på Teknisk fysikk

Intervjuene med dagens masterstudenter på Teknisk fysikk avdekket at så å si alle kunne programmere. Dette var riktignok på et svært varierende nivå, med alt fra "Jeg kan åpne MATLAB og plotte en graf" til de mer kompetente programmererne. Undersøkelsen avdekket at de verktøyene som var mest utbredt blant masterstudentene, var MATLAB og C/C++, noe som kan virke litt merkelig, ettersom den formelle opplæringen i programmering for dagens masterstudenter har bestått av Java og JSP. Java var ikke i nærheten så populært som C/C++, og JSP var det ingen som brukte. Meningene om opplæringen i IKT ved Fysikk og matematikk varierte fra "dårlig" til "ikkeeksisterende", da spesielt siden de hadde fått opplæring i språk og tema de mente var helt irrelevant for deres videre utdanning. På spørsmålet om det burde bli innført en CSE-reform på Institutt for fysikk ble studentene overraskende skeptiske sett i lys av deres holdning til hvilken standard IKT-undervisningen hadde holdt. Dette begrunnet de med at de var svært fornøyd med selve fysikkundervisningen de hadde fått. En endring kunne altfor lett føre til en forringelse. Skulle det satses på en CSE-reform, måtte den gode fysikkundervisningen bli tatt godt vare på. Det studentene var mest misfornøyd med, var det de mente var en overflod av ikketekniske emner. Skulle det innføres egne fag i tilknytning til en CSE-reform, var det disse som måtte ofres.

Muligheter for tverrfaglig samarbeid

Etter å ha snakket med personal både på Institutt for datateknikk og informasjonsvitenskap og Institutt for matematiske fag kan det konstateres at mulighetene for et tverrfaglig samarbeid for en CSE-reform i høyeste grad er tilstede. I TMA4100 Matematikk 1 kan en for eksempel innføre en mer programmeringsvennlig vinkling på forskjellige tema og muligens innføre nye. Alternativer som ble diskutert, var numerisk derivasjon og integrasjon, numeriske løsere av differensialligninger og feilestimering med mer. De ansvarlige for TDT4105 IT Grunnkurs ønsker seg et koordinert løp mellom TDT4105 IT Grunnkurs, TMA4100 Matematikk 1 og linjefagene for de forskjellige siv.ing.-programmene. Dette er noe Institutt for fysikk burde ta tak i for å være med på å kunne påvirke hvilken retning IT Grunnkurs tar i framtiden.

Hensiktsmessige oppgaver og verktøy

Hvilke verktøy som undervises i de forskjellige semestrene, påvirker hvilken type oppgaver det er fornuftig å innføre på gitte tidspunkt. Før TDT4102 Prosedyre- og objektorientert programmering i 4. semester er det lite hensiktsmessig å gi studentene oppgaver i tunge numeriske algoritmer, eller oppgaver i programutvikling. Skulle slike numeriske løsere være påkrevd i en gitt oppgave, burde en gjøre bruk av de ferdigskrevne funksjonene og modulene allerede implementert i MATLAB, eller annet egnet verktøy. Dette setter dog ikke en stopper for enkle numeriske beregninger. Ferdigutviklet programvare gir ikke studentene den samme følelsen med de numeriske metodene, så disse kan ikke brukes i den hensikt å styrke deres evner i programmering og numerikk. De har likevel sin plass i undervisningen, da de er uovertrufne når det kommer til beregning, design og illustrasjon av komplekse fysiske systemer.

4.2 Anbefaling til instituttet

En titt i matematikkpensumet for sivilingeniørlinjen Fysikk og matematikk avslører at det er lite numerikkopplæring i løpet av de to første årene. De numeriske metodene som kommer opp, blir i liten grad utnyttet til sitt fulle potensial, da det kalkuleres for hånd, og ikke maskinelt. I IKT-fagene, spesielt i TDT4102 Prosedyre- og objektorientert programmering er det smått med direkte relevante oppgaver for fysikere og matematikere. Dette forverres ved manglende eksponering for IKT i de aktuelle fysikkfagene. Det er heller ingen plan over de ulike initiativene for styrking av IKT i fysikkfagene. Hullene i utdanningsløpet, relatert til IKT, som må tettes er:

- Svake ferdigheter i numerikk.
- Lite eksponering for programmering utenfor IKT-emnene.
- Manglende motivering av programmering før slutten av studiet.
- Planmessighet i IKT-bruken.

Manglene i numerikkopplæring, og dertil maskinell beregning, kan lettest bøtes på ved et godt samarbeid med Institutt for datateknikk og informasjonsvitenskap og Institutt for matematiske fag. Dette burde være høyst mulig, da det allerede er interesserte elementer ved de respektive instituttene. Med tanke på jevnlig eksponering for numeriske beregninger, må det handles internt på Institutt for fysikk.

Bakgrunn for vedtak ved Institutt for fysikk

Det første en må ta stilling til, er om et kompetanseløft for IKT ved Institutt for fysikk er ønskelig. Denne vurderingen kan gjøres blant annet på grunnlag av denne rapporten. Supplerende data på hvordan studenter tilpasser seg en slik reform kommer i masteroppgaven til Simen Sørby ved UiO i mars 2010. Ønsker en en enda grundigere analyse

av den pedagogiske forskjellen mellom klassisk fysikkundervisning og undervisning utarbeidet i samsvar med CSE-reformen, kan det anbefales å gjøre en studie av tilsvarende emner ved UiO og NTNU. De mest aktuelle emnene å sammenligne er TFY4145 Mekanisk fysikk ved NTNU, som har forelesninger og øvinger fullstendig fritt for IKT, og FYS-MEK1100 Mekanikk ved UiO, som har det mest omfattende IKT-øvingsopplegget av de obligatoriske fysikkemnene ved UiO.

Den videre utredningen av anbefalte tiltak forutsetter at instituttet vedtar å innlede en oppgradering av IKT-ferdighetene blant studentene.

Innledende og allmene tiltak

I første omgang må rammeverket på plass. Det burde opprettes en gruppe, som har ansvaret for å utvikle en plan over hvilke kunnskaper og ferdigheter det er ønskelig at studentene tilegner seg i løpet av studieløpet. Når det er gjort klart hvilke behov som skal tilfredsstilles, må det startes et samarbeid med Institutt for datateknikk og informasjonsvitenskap og Institutt for matematiske fag. Slik kan en i fellesskap sørge for at en har de kunnskapene som studentene trenger for å kunne bruke IKT som et effektivt verktøy. De nevnte, berørte instituttene oppfordrer til slikt samarbeid.

Gitt at en får de verktøyene som trengs i IKT- og Matematikkfagene, vil en implementasjon i fysikk-emnene måtte fokusere på å terpe og vedlikeholde disse ferdighetene, slik at de ikke glemmes før studentene når 9. og 10. semester, og åpne for de pedagogiske mulighetene numeriske beregninger i fysikken gir.

Vi skal videre presentere noen alternativer til praktisk implementsjon av en CSE-reform ved Institutt for fysikk. De to første kan implementeres uavhengig av hverandre. Ønskes en veldig omfattende CSE-refom, kan disse implementeres sammen. Det anbefales likevel å ta for seg en av gangen, i et tempo som gjør det mulig å vurdere resultatene etterhvert, slik at reformen blir best mulig. Det siste tiltaket som blir diskutert burde bero seg på at minst ett av de to første tiltakene blir gjennomført i en viss skala.

Revisjon av fysikkemner etter modell fra UiO

En mulighet for heving av IKT kompetansen, er å kjøre en CSE-reform direkte etter mønster fra UiO. Dette vil si at en sørger for å legge om forelesningene og øvingsopplegget i noen, eller alle de obligatoriske fysikkfagene. Dette krever et godt samarbeid med, og mye drahjelp fra, Institutt for datateknikk og informasjonsvitenskap og Institutt for matematiske fag, da en slik omfattende innføring vil ta stor plass, og fysikkfagene ikke primært kan brukes til numerikk- og programmeringsopplæring. Skal en slik modell innføres er det vesentlig at studentene blir kompetente i bruk av numeriske verktøy *før* de skal bruke dem i fysikkfaget. Det burde opprettes en egen gruppe med ansvaret for det faglige og pedagogiske innholdet. Dette må gjøres for å sikre at alle punktene over ønsket kompetanse blir oppfylt, og for å forhindre at omfanget og kvaliteten blir for avhengig av den enkelte faglærers kompetanse og vilje. For å få en best mulig undervisning etter en slik omlegging burde også den faglige kompetansen blant faglærerne oppgraderes i de programmeringsspråkene som brukes.

En CSE-reform etter modell fra UiO koster, og skal en ha håp om å gjennomføre den med suksess, må den finansieres. Slik kan en gjøre en grundig opprusting av emnene og skaffe nok studentassistenter og infrastruktur. Hvis en velger en begrenset eller stegvis innføring av en slik modell, bør en prioritere de tidlige fagene, da TDT4105 IT Grunnkurs går i 1. semester, og sørge for at minst ett emne i semesteret blir berørt, for slik å sørge for kontinuitet. En kan dessuten øke kompleksiteten i programmeringsoppgaver etter 4. semester og TDT4102 Prosedyre- og objektorientert programmering. En stor fordel med å velge denne modellen er at den allerede er utviklet i Oslo, og en kan dermed dra nytten av å studere tilsvarende emner når en skal utvikle øvinger og forelesninger. Dette medfører at det muligens ikke kreves at like store ressurser blir satt av til reformen som i Oslo. Hvordan CSE-reformen har blitt gjennomført i store trekk ved UiO er dokumentert i kapittel 2.

Revisjon av laboratorieøvingene

En annen mulighet er å reformere laboratorieøvingene. Dette har den fordelen at de er dekoblet forelesningene. Laboratorieøvingene holdes av en egen gruppe og er dermed heller ikke avhengig av velvilje og kompetanse blant samtlige faglærere. Denne modellen må tildels utvikles fra bunnen av, da laboratorieøvingene på Fysikk ved UiO ikke er like omfattende utarbeidet for IKT. Det er forskjell på gode øvingsoppgaver og laboratorieoppgaver.

En kan se for seg å revidere noen av laboratorieoppgavene i TFY4145 Mekanisk fysikk i 1. semester, TFY4155 Elektromagnetisme i 2. semester, TFY4160 Bølgefysikk i 3. semester og TFY4165 Termisk fysikk i 4. semester. Dermed har en dekt hele løpet fram til studentene skal ha TDT4102 Prosedyre- og objektorientert programmering, slik at de får befestet sine MATLAB-ferdigheter, og det blir tydelig at programmering er et verktøy for løsning av fysikkproblemer. Her har en muligheten for å emulere reelle forskningssituasjoner, og koble simuleringer med eksperiment. Dette kan gjøres ved å først holde teoriseminar, deretter gjøre numeriske beregninger og/eller simuleringer, for til sist kjøre et eksperiment og sammenligne med numeriske data. Dette oppfyller behovet for en forskningsbasert undervisning. En har allerede en del oppgaver i databehandling, og en burde beholde og utvikle disse oppgavene.

Et eksempel på en slik oppgave kan være å gjøre en utvidet versjon av obligatorisk oppgave 4 i FYS-MEK1100 Mekanikk ved UiO. Ved først å gjennomgå den teoretiske bakgrunnen, Hookes lov og Newtons 2. lov, gis studentene et grunnlag for å beregne pendelbanen numerisk. Avslutningsvis kan det gjøres et praktisk eksperiment med en kule i strikk. Dette kan danne grunnlaget for en rapport som inneholder beskrivelse av teori, simulering, eksperiment og en sammenligning mellom numeriske og eksperimentelle data.

Etter 4. semester kan og burde en ha med oppgaver i C/C++, gjerne oppgaver som krever de programmeringstekniske ferdighetene. Slik får en befestet ferdighetene som læres i TDT4102 Prosedyre- og objektorientert programmering. Dette kan gjøres ved å begynne med enda mer omfattende oppgaver i laboratorieoppgavene fra 5. semester.

Tverrfaglig prosjektarbeide

Da studentene på Fysikk og matematikk tok TDT4100 Objektorientert programmering, måtte de gjennomføre et spillprosjekt. Det vil si at de i grupper laget et kjørbart spill fra bunnen av. Dermed fikk de erfare det å være med på å bygge opp en fulstendig programvare. Dette spillprosjektet finnes ikke i TDT4102 Prosedyre- og objektorientert programmering.

Å innføre prosjektoppgave i ett av fysikkfagene i 5. semester kan gi studentene den samme erfaringen. Oppgaven kan være omfattende og kreve at studentene skal lage en liten, men komplett programvare for beregning av et gitt eller valgbart fysisk system. Hvis man er ambisiøs, kan en gjøre et tverrfaglig prosjekt sammen med studentene fra Industriell matematikk. Dette er også en forberedende øvelse i reelle forskningssituasjoner, der personer fra forskjellige miljøer jobber sammen i tverrfaglige prosjekt. Dette krever ett godt samarbeid med Institutt for matematiske fag, og nøye planlegging for å legge prosjektoppgaven på riktig vanskelighetsnivå.

For å kunne gjennomføre et slikt prosjekt, anbefales det at studentene har blitt eksponert for IKT-oppgaver jevnlig i de fire første semestrene.

Bibliografi

[1] Eksternevaluering: Evaluation of the engineering education at NTNU. http://www.ntnu.no/omntnu/evaluering/siving_2008. Professor Jorg Steinbach et al.

[2] Internevaluering: Applied Physics and Mathematics. http://www.ntnu.no/omntnu/evaluering/siving_2008.

[3] Morten Hjort-Jensen. Fredagskollokvium, NTNU, 13.02.09. Datamaskiner i realfagsopplæringen, en ny måte å undervise realfag på?

[4] Morten Hjort-Jensen, Knut Mørken, Annik Myhre, and Hanne Sølna. Computers in science education: A new way to teach science? http://www.uhr.no/documents/cse_nov2008.pdf.

[5] http://www.ntnu.no/studier/emner/TDT4105. Webside for faget TDT4105 ved NTNU.

[6] http://www.ntnu.no/studier/emner/TDT4102. Webside for faget TDT4102 ved NTNU.

[7] http://www.uio.no/studier/program/fam/fysikk/. UiO sin webside om studieprogrammet Fysikk.

[8] http://www.forskningsradet.no/sff/. Forskningsrådet sine websider om SFF.

[9] http://www.forskningsradet.no/sff/. Krav og retningslinjer,Sentre for Fremragende Forskning.

[10] http://www.uio.no/forskning/satsing/sff.html. UiO sine websider om SFF.

[11] http://www.ntnu.no/forskning/sff. NTNU sine websider om SFF.

[12] Epost fra Jan Eskil Aldal, budsjettansvarlig MN-fakultetet, UiO.

[13] http://www.matnat.uio.no/internt/fakultetet/plandokumenter/Strategisk_plan.pdf. Strategisk plan 2005 - 2009, Universitetet i Oslo, Det matematisk-naturvitenskapelige fakultet.

[14] Oversikt over søknader til cse mars 07. Epost fra Hanne Sølna, studiekoordinator ved Det matematisk-naturvitenskapelige fakultet, UiO.

[15] Tildeling for undervisningsomlegging 2008. Epost fra Hanne Sølna, studiekoordinator ved Det matematisk-naturvitenskapelige fakultet, UiO.

[16] Epost fra Hanne Sølna, studiekoordinator ved Det matematisk-naturvitenskapelige fakultet, UiO.

[17] `http://www.python.org/`. Python prosjektets webside.

[18] Hans Petter Langtangen. *Introduction to Computer Programming.* `http://www.uio.no/studier/emner/matnat/math/MAT-INF1100/h08/kompendiet/komp.html`, 2009.

[19] `http://www.uio.no/studier/emner/matnat/math/MAT1100/`. Webside for faget MAT1100 ved UiO.

[20] `http://www.uio.no/studier/emner/matnat/math/MAT-INF1100/`. Webside for faget MAT-INF1100 ved UiO.

[21] `http://www.uio.no/studier/emner/matnat/ifi/INF1100/`. Webside for faget INF1100 ved UiO.

[22] *Laboratorium i emne TFY4155 Elekromagnetisme.* Institutt for fysikk, NTNU, 2009.

[23] `http://web.phys.ntnu.no/~nordam/fy0001/`. Webside for faget FY0001 ved NTNU.

Tillegg A

Oppgaver fra UiO

A.1 Eksamen FYS-MEK1110 Mekanikk vår 2008

UNIVERSITETET I OSLO

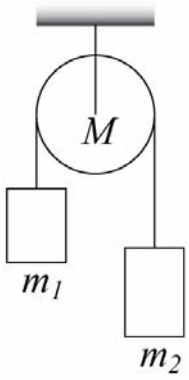
Det matematisk-naturvitenskapelige fakultet

Eksamen i: FYS-MEK1110
Eksamensdag: Mandag 9. juni 2008
Tid for eksamen: Kl. 0900-1200
Oppgavesettet er på 5 sider inkludert formelarket.
Tillatte hjelpemidler: Øgrim og Lian: Størrelser og enheter i fysikk og teknikk eller
Angell, Lian, Øgrim: Fysiske størrelser og enheter: Navn og symboler
Rottmann: Matematisk formelsamling
Elektronisk kalkulator av godkjent type.

Kontrollér at oppgavesettet er komplett før du begynner å besvare spørsmålene.

Ved sensur vil alle deloppgaver bli tillagt like stor vekt med mindre annet er oppgitt i oppgaven. Vi forbeholder oss retten til justeringer.

Oppgave 1



a) Atwoods fallmaskin består av en talje med masse M som henger i en snor fra taket. I en masseløs snor om taljen henger to masser $m_1 > m_2 > M$.
Tegn et frilegemediagram for taljen og navngi kreftene. Ranger absoluttverdien av kreftene på snorene.

b) Du slipper en ball i gulvet fra en høyde h. I støtet med gulvet reduseres hastigheten til ballen med en faktor r slik at hastigheten v_1 etter støtet er relatert til hastigheten v_0 før støtet ved: $v_1 = -rv_0$. Hvor høyt spretter ballen etter støtet? Se bort fra luftmotstand.

c) Du kaster en ball horisontalt fra en høyde h slik at den treffer gulvet med hastigheten $\vec{v}_0 = v_{0,x}\hat{i} + v_{0,y}\hat{j}$. I støtet med gulvet reduseres hastigheten til ballen med en faktor r slik at hastigheten $\vec{v}_1$ etter støtet er $\vec{v}_1 = rv_{0,x}\hat{i} - rv_{0,y}\hat{j}$. Hvor høyt spretter ballen etter støtet? Se bort fra luftmotstand.

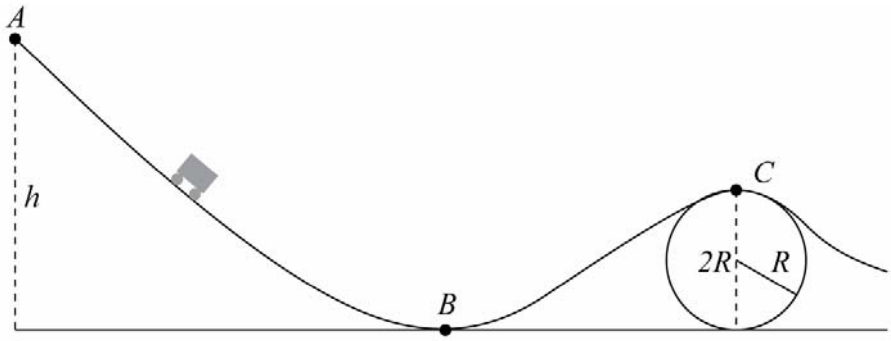
d) En planet befinner seg i posisjonen $\vec{r}$ i et koordinatsystem hvor solen er i origo. For hvilke hastigheter $\vec{v}$ vil planetens bevegelse være en sirkel?

e) Kari kjører i et romskip som holder hastigheten $0.5c$ i forhold til Ole. Kari måler romskipet til å være 10m langt i fartsretningen. Forklar hvordan Ole og Kari måler lengden av romskipet. Hvilken lengde finner Ole at romskipet har?

Oppgave 2

I denne oppgaven skal vi studere en vogn med massen m som sklir langs en bane. Du kan se bort fra friksjon og luftmotstand. Tyngdens akselerasjon er g.

Vognen starter i ro i punktet A i høyden h over punktet B, som illustrert i figuren. Vognen sklir ned til punktet B i bunnen av kurven og deretter opp mot punktet C på bakketoppen. I punktet C er krumningsradiusen R. Punktet C ligger en høyde $2R$ over punktet B.

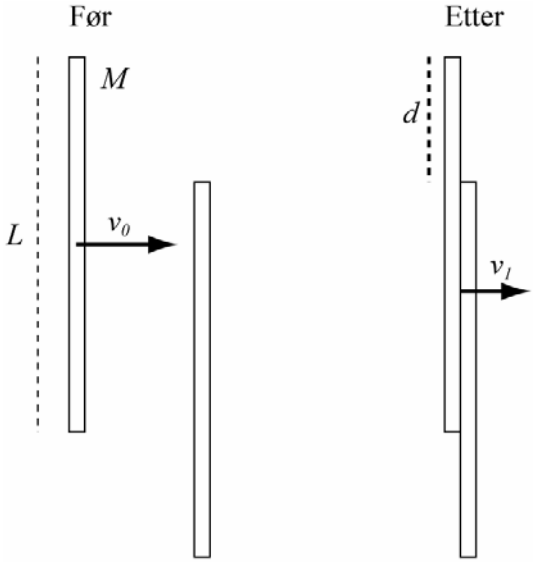


a) Finn hastigheten til vognen i punktet C. Hvor stor må h være for at vognen skal nå punktet C?

b) Hva er betingelsen for at vognen skal beholde bakkekontakten i punktet C? For hvilke høyder h vil vognen miste bakkekontakten i punktet C?

Oppgave 3

Vi skal i denne oppgaven se på en kollisjon mellom to lange, tynne staver som blir hengende sammen. Dette kan for eksempel være en modell for hvordan to lange, lineære molekyler kolliderer. De to stavene er identiske. Hver stav har masse M og lengde L. For hver stav er treghetsmomentet om dens massesenter $I_0 = ML^2/12$. Stavene glir på en horisontal, friksjonsfri flate som illustrert i figuren.



Før kollisjonen er stavene parallelle. En stav ligger i ro, mens den andre staven har hastigheten v_0. Etter kollisjonen blir de hengende sammen som en stav, som illustrert i figuren. Startposisjonen karakteriseres ved forskyvningen d som illustrert i figuren. Du kan se bort fra bredden og høyden av staven.

a) Vis at treghetsmomentet om massesenteret for det samlede legemet er:

$$I = \frac{M}{2}\left(d^2 + \frac{L^2}{3}\right)$$

Anta først at $d = 0$.

b) Finn hastigheten v_1 til massesenteret til det samlede legemet etter støtet.

La oss nå se på det generelle tilfellet, hvor $0 \le d \le L$.

c) Finn hastigheten v_1 til massesenteret til det samlede legemet etter støtet.

d) Finn vinkelhastigheten ω_1 til det samlede legemet om massesenteret etter støtet.

e) Hva er tapet i energi i støtet? For hvilken d blir tapet minst? Kommenter resultatet.

f) Beskriv bevegelsen etter støtet.

Oppgave 4

I denne oppgaven skal vi studere en person som hopper strikk. Strikken oppfører seg som en ideell fjær med fjærstivhet k når den blir strukket, men den har ingen styrke når den blir dyttet sammen. Strikkens likevektslengde er d. Det er også en viss demping i strikken som vi modellerer som en kraft som er avhengig av hastigheten strikken deformeres med. Når strikken er strukket til en lengde x, og strikken strekkes med den momentane hastigheten v, er kraften fra strikken gitt som:

$$F(x,v) = \begin{cases} -k(x-d) - c_v v & x > d \\ 0 & x \le d \end{cases},$$

Her er c_v en konstant som beskriver dempingen i tauet, og k er fjærstivheten.

Vi legger nullpunktet for høyden der strikken er festet og regner positiv retning nedover. En person med masse m fester strikken om livet og hopper fra punktet hvor strikken er festet. Han starter med null hastighet. Du kan se bort fra luftmotstanden og du kan anta at strikken er masseløs. Bevegelsen er kun vertikal. Tyngens akselerasjon er g.

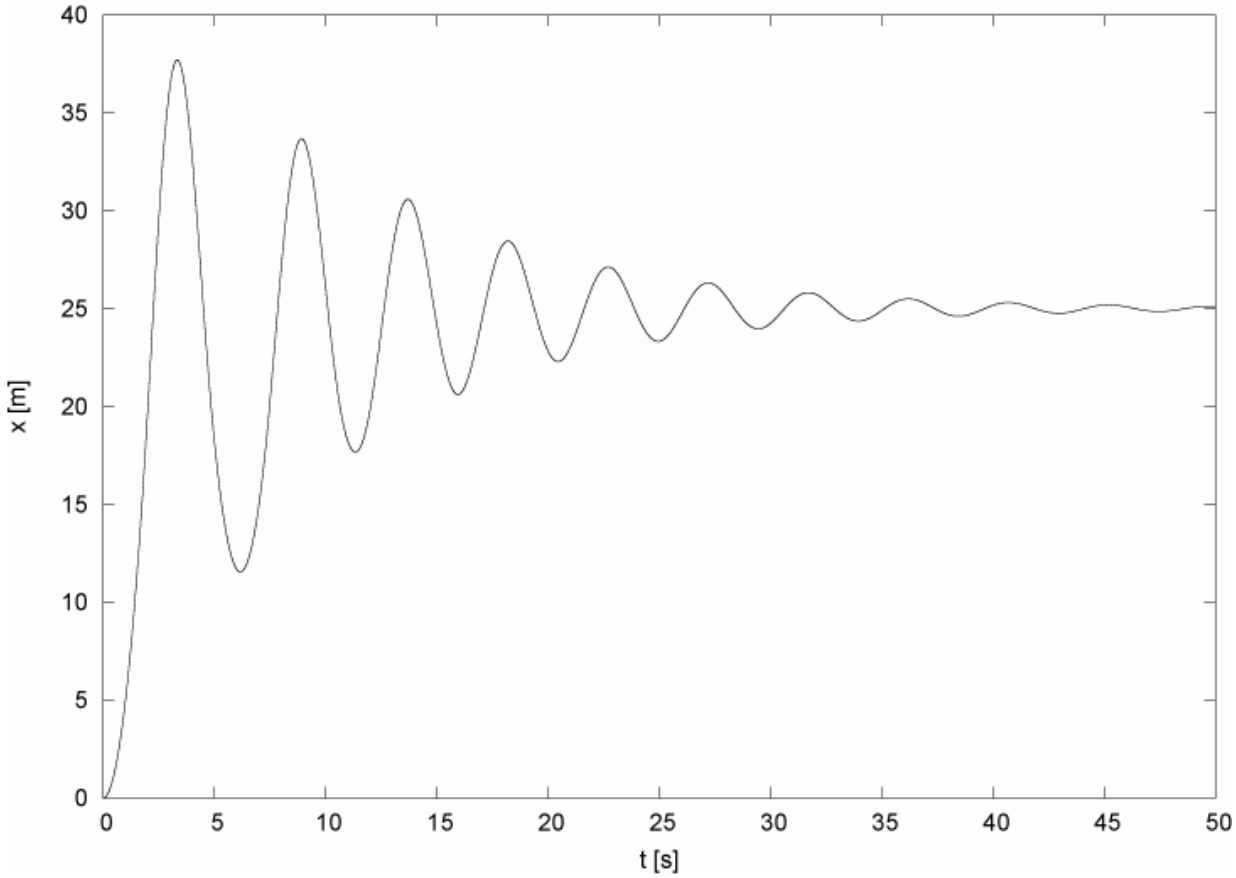
a) Tegn et frilegemediagram for personen når strikken er stram. Navngi alle kreftene.

b) I hvilken høyde blir personen hengende når bevegelsen har stanset?

c) Skisser en numerisk algoritme som finner posisjonen og hastigheten til personen ved tiden $t + \Delta t$ gitt posisjonen og hastigheten ved tiden t, hvor Δt er et lite tidsintervall. Du kan gjøre dette i Python, matlab eller pseudo-kode. Med pseudo-kode mener vi her at du skisserer algoritmestegene på et vis som gjør metoden tydelig.

d) Figur 1 nedenfor viser resultatet av en simulering av et strikkhopp med denne modellen for en strikk med lengden $d = 20\,\text{m}$ og en person med masse $m = 80\,\text{kg}$. Forklar resultatet. Gi et estimat for fjærkonstanten k brukt i simuleringen vist i figur 1.

e) Er systemet konservativt gjennom hele bevegelsen, i deler av bevegelsen, eller ikke i det hele tatt? Begrunn svaret.



Figur 1: Resultat av simulering av strikkhopp.

***

Dette er siste ark i oppgavesettet. Lykke til med oppgavene!

Formelark Fys-mek1100 våren 2008

$\sum \vec{F} = m\vec{a} = \frac{d\vec{p}}{dt}$, hvor $\vec{p} = m\vec{v} = m\frac{d\vec{r}}{dt}$, og $\vec{a} = \frac{d\vec{v}}{dt} = \frac{d^2\vec{r}}{dt^2}$.

Konstant $\vec{a}$: $\vec{v} = \vec{v}_0 + \vec{a}t$, $\vec{r} = \vec{r}_0 + \vec{v}_0 t + \frac{1}{2}\vec{a}t^2$, $v^2 - v_0^2 = 2\vec{a} \cdot (\vec{r} - \vec{r}_0)$.

Konstant α: $\omega = \omega_0 + \alpha t$, $\theta = \theta_0 + \omega_0 t + \frac{1}{2}\alpha t^2$, $\omega^2 - \omega_0^2 = 2\alpha \cdot (\theta - \theta_0)$.

Baneakselerasjon: $\vec{a} = \frac{dv}{dt}\hat{u}_T + \frac{v^2}{\rho}\hat{u}_N$.

Rotasjon: $\vec{v} = \vec{\omega} \times \vec{r}$, $\vec{a} = \vec{\alpha} \times \vec{r} + \vec{\omega} \times (\vec{\omega} \times \vec{r})$.

Galilei-trans.: $\vec{r} = \vec{R} + \vec{r}'$, $\vec{v} = \vec{V} + \vec{v}'$.

Fjærkraft: $F(x) = -k\,(x - x_0)$. Luftmotstand: $\vec{F}_v = -k\vec{v}$ eller $\vec{F}_v = -Dv\vec{v}$.

Friksjon: $|F_s| \leq \mu_s N$ eller $|F_d| = \mu_d N$.

Arbeid: $W_{AB} = \int_A^B \vec{F} \cdot d\vec{r} = K_B - K_A$, Kinetisk energi: $K = \frac{1}{2}mv^2$.

Potensiell energi: $U(\vec{r})$. Tyngdekraft: $U = mgy$. Fjærkraft: $U = \frac{1}{2}k(x - x_0)^2$.
Konservativ kraft: $\vec{F} = -\nabla U(\vec{r})$.

Impuls: $\vec{J} = \int_{t_0}^{t_1} \vec{F} dt = \Delta\vec{p} = \vec{p}(t_1) - \vec{p}(t_0)$.

Rakett-likningen: $\vec{F}^{\text{ext}} + \vec{v}_{\text{rel}}\frac{dm}{dt} = m\vec{a}$.

Massesenter: $\vec{R} = \frac{1}{M}\sum_i m_i\vec{r}_i = \frac{1}{M}\int_M \vec{r} dm$, $M = \sum_i m_i = \int_M dm$.

Kraftmoment: $\vec{\tau} = \vec{r} \times \vec{F}$. Spinn: $\vec{L} = \vec{r} \times \vec{p}$.

Spinnsats: $\vec{\tau} = \frac{d\vec{L}}{dt}$. Stive legemer: $L_z = I_z\omega_z$, $\tau_z = I_z\alpha_z$.

Kinetisk energi: $K = \frac{1}{2}I\omega^2$, $I = \sum_i m_i\rho_i^2 = \int_M \rho^2 dm$.

Parallellakseteoremet: $I = I_{\text{cm}} + Md^2$.

Rullebetingelse: $V = \omega R$.

Fiktive krefter: $m\vec{a}' = \sum \vec{F}^{\text{ext}} - m\vec{A} - m\frac{d\vec{\omega}}{dt} \times \vec{r}' - 2m\vec{\omega} \times \vec{v}' - m\vec{\omega} \times (\vec{\omega} \times \vec{r}')$.

Gravitasjon: $\vec{F}(\vec{r}) = -G\frac{m_1 m_2}{r^2}\hat{u}_r$, $U(r) = -G\frac{m_1 m_2}{r}$.

Spenning og tøyning: $\sigma_{xx} = \frac{F_x}{A_x} = E\frac{\Delta x}{x} = E\epsilon_{xx}$, $\frac{\Delta y}{y} = -\nu\frac{\Delta x}{x}$.

Lorentz-trans.: $x' = \gamma\,(x - ut)$, $y' = y$, $z' = z$, $t' = \gamma\left(t - \frac{u}{c^2}x\right)$, $\gamma = \frac{1}{\sqrt{1 - \frac{u^2}{c^2}}}$.

Relativistisk: $m = \gamma m_0$, $\vec{p} = m\vec{v}$, $E = mc^2$.

A.2 Obligatorisk oppgave 4 i FYS-MEK1110 Mekanikk

Fys-Mek1110 – 2009 – Oblig 4

Project 6: Ball in a spring

In this project you will study an advanced model of a pendulum. The pendulum consists of a ball in a rope moving in a vertical plane. The ball has mass m. You can neglect air resistance. We describe the position of the ball by the position vector, $\vec{r}$. You may describe the force from the rope using a spring model with a spring constant k and equilibrium length L.

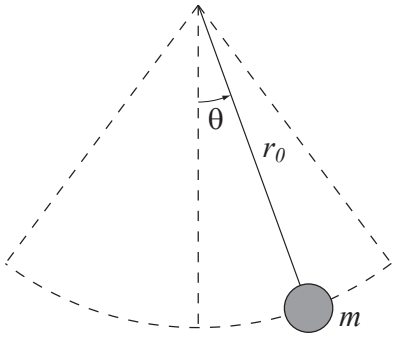


Figure 0.2: Illustration of the pendulum consiting of a ball of mass m attached to rope of length r_0.

(a) Identify the forces and draw a free-body diagram of the ball when $\theta = 0$.

(b) Show that the net external force acting on the ball can be written as:

$$\sum_j \vec{F} = -mg\hat{j} - k\,(r - L)\,\frac{\vec{r}}{r}\,, \tag{13}$$

where $r = |\vec{r}|$ is the length of the (stretched) rope, and the origin of the coordinate system is chosen to be the attachment point of the rope.

(c) Find an expression for the acceleration of the ball.

In this project, we will not prescribe the motion of the ball, but instead use Newton's second law to determine the motion of the ball. In this model, we can measure the tension in the rope, as well as the motion of the ball, and analyze these to learn about the motion.

(d) It is customary to describe the position of the pendulum by the angle θ with the vertical alone. Does this give a sufficient description of the position of the ball in this case?

(e) If the ball is at $\theta = 0$ with no velocity, what is the position of the ball?

We will now study a specific pendulum, consisting of a ball with a mass of 0.1kg, and a rope of length $L = 1$m with a spring constant $k = 200$N/m, which corresponds to a rather elastic rope. We want to study the motion of the ball by integrating the equations of motion numerically.

(f) Write a program that finds the motion of the ball, if it starts at the angle $\theta = 30°$ at a distance L from the origin, starting with zero velocity. The program should plot the position of the ball in the xy-plane for the first 10s of the motion.

(g) Use the program to find the behavior for the given initial conditions. Describe and interpret the motion.

(h) Rerun the program with $k = 2000.0$. Describe the motion in this case. How can you use this method to model a pendulum in a stiff rope? What do you think would be the limitation of this approach?

(i) Find and plot the rope tension, T, – the magnitude of the spring force – and the angle, θ, as functions of time. You can find the angle θ from $\theta = \arctan(-r_x/r_y)$ in this case, since the angle is the angle with the vertical. How do you interpret the results? Plot T as a function of θ? How do you interpret this?

In order to interpret this data, you need to make an average of the various values of the rope tension, T, as a function of θ. We have included a very simple function for this type of binned averaging. The function `binavg` takes as arguments an array such as `t` and an array such as `f`, both of the same length, and generates the average values for `f` for each value of `t` binned into bins with sizes `dt`:

```
def binavg(t,f,dt):
    nnt=t/dt
    nt = nnt.astype('int');
    nmin = min(nt);
    nmax = max(nt);
    N = nmax - nmin + 1;
    tavg = zeros(N,float);
    ntavg = zeros(N,float);
    favg = zeros(N,float);
    nfavg = zeros(N,float);
    for i in range(len(t)):
        n = nt[i] - nmin;
        favg[n] = favg[n] + f[i];
        nfavg[n] = nfavg[n] + 1;
        tavg[n] = tavg[n] + t[i];
        ntavg[n] = ntavg[n] + 1;
    tavg = tavg/ntavg;
    favg = favg/nfavg;
    return tavg,favg
```

(j) (Optional) Use this function to find T as a function of θ, and compare with the component of gravity in the direction θ. Comment on the results.

(k) Modify your program to find the acceleration of the ball as a function of θ, and decompose it into two components: The radial component, a_r, in the direction of the spring, and the tangential component, a_θ, in the direction of motion, normal to the direction of the spring. Plot the radial and tangential acceleration as functions of θ, and discuss the results. Illustrate by a drawing how you made the decomposition of the acceleration vector.

(l) (Optional) Modify your program to measure the period of the pendulum – the time from the pendulum is at a given θ, until it returns to the same angle with the same direction of the tangential velocity.

(m) (Optional) Determine how the period depends on the initial angle, θ, by plotting the period, T, as a function of θ for $theta = 5°, 10°, 15°, 20°, 25°$, and $30°$.

(n) (Optional) Determine how the period depends on the mass of the ball.

(o) (Optional) Determine how the period depends on the length of the rope.

(p) (Optional) Rewrite your program to ensure that the rope tension is zero if the spring is compressed, because the rope cannot sustain compression. Use this program to determine the motion when $\vec{v}_0 = 6.0\hat{i}$m/s when the ball hanging straight down with no elongation. What happens?

End of Oblig 4

A.3 Obligatorisk oppgave 1 i MEK1100 Feltteori og vektoranalyse

Oblig 1 MEK1100, vår 2009

Krav til innlevering og godkjenning

På hvert av punktene 1,2a-2e kan du oppnå 8 poeng, mens 3a-3d kan gi 13 poeng hver. Poengene tildeles på liknende vis som på en eksamen. I alt kan du oppnå 100 poeng og du må ha minimum 70 poeng for å få godkjent. Dersom dette ikke er innfridd ved innlevering, men besvarelsen vurderes som et seriøst forsøk, kan det gis anledning til ny innlevering. For nærmere informasjon om regler se
http://www.math.uio.no/academics/obligregler.shtml
Tidsfrister er gitt på kurssidene til MEK1100.

Du kan bruke Latex eller levere en håndskrevet besvarelse. I det første tilfellet skal oppgave 1 og 2b gjøres for hånd og enten stiftes sammen med latexsidene eller scannes/fotograferes og inkluderes som grafikk i latexkoden. Dersom du leverer håndskrevet besvarelse må du ta med printinger av alle skript og figurer i oppgave 3. Det skal skrives filnavn på skriptlistinger og oppgavenummer, med bokstav, på figurer.

Det er tillatt å samarbeide i grupper med den obligatoriske oppgaven. Likevel må alle levere en individuell besvarelse. På denne besvarelsen skal navnene på de man har samarbeidet med angis.

1 Et vektorfelt

I denne oppgaven brukes ikke Matlab.
Skisser en del piler for vektorfeltet

$$\vec{F}(x,y) = x\vec{\imath} - y\vec{\jmath}$$

2 Et todimensjonalt strømfelt

I denne oppgaven brukes ikke Matlab.
Et hastighetsfelt i xy-planet er gitt ved $\vec{v} = u\vec{\imath} + v\vec{\jmath}$ der

$$u = \cos(x)\sin(y), \quad v = -\sin(x)\cos(y). \tag{1}$$

(a) Finn divergens og virvling av hastighetsfeltet.

(b) Tegn opp strømvektorer langs x- og y aksen.

(c) Finn sirkulasjonen om randa til kvadratet definert ved $-\frac{1}{2}\pi \leq x \leq \frac{1}{2}\pi$ og $-\frac{1}{2}\pi \leq y \leq \frac{1}{2}\pi$.

(d) Forklar hvorfor det eksisterer en strømfunksjon for feltet gitt i likning (1). Vis at strømfunksjonen kan skrives

$$\psi = \cos(x)\cos(y). \tag{2}$$

(e) Bruk Taylorutvikling av andre orden til å finne tilnærmede strømlinjer nær origo.

3 Strømlinjer og hatighetsfelt i Matlab

I denne og neste oppgave skal du skrive noen funskjoner og skript i Matlab. Disse og de plott som produseres skal leveres inn som en del av besvarelsen. Det er viktig at de filene du lager har de navn som oppgis. Alle figurer skal ha tittel og tekst på akser.

Sammen med oppgaveteksten, på hjemmesiden, finner du en fil **streamfun.m** med en funksjon som beregner strømfunksjonen (2) i området $-\frac{1}{2}\pi \leq x \leq \frac{1}{2}\pi$, $-\frac{1}{2}\pi \leq y \leq \frac{1}{2}\pi$. Kildekoden er

```
% Regner ut et grid og en strømfunksjon
function[X,Y,psi] = streamfun(n)

%setter standardverdi for n til 20
if nargin < 1;
    n=20;
end

%linspace brukes istedet for  -0.5*pi:step:0.5*pi for å unngå
%tøys pga. avrunding
x=linspace(-0.5*pi,0.5*pi,n);
%resultatet er en vektor med n elementer, fra -pi/2 til pi/2
[X,Y] = meshgrid(x,x);
psi=cos(X).*cos(Y);
```

Et kall på funksjonen kan se ut som

$$\gg \textbf{[x,y,psi]=streamfun(n);}$$

der **x,y,psi** er arrayer for hhv. x-verdier, y-verdier og ψ. Inngangsparameteren **n** er antall punkter som brukes i hver retning. Avstanden mellom punktene, i begge retninger, blir da $\Delta x = \Delta y = \frac{\pi}{n-1}$.

(a) Bruk funksjonen **streamfun** i et skript, **strlin.m**, som plotter konturlinjer for ψ når

(i) $n = 5$

(ii) $n = 30$

Framstill tilfellene (i) og (ii) i hvert sitt diagram. Sammenhold plottene med oppgave 2(e) og diskuter de valgte verdier for n.

(b) Skriv en funksjon (filnavn **velfield.m**) som beregner hastigheter utfra likning (1) ved kallet

$$\gg \textbf{[x,y,u,v]=velfield(n);}$$

Bruk denne i et skript, **vec.m**, som tegner et vektorplott av hastighetsfeltet. Legg vekt på å velge et passende antall punkter for lesbarheten av plottet.

(c) Lag en funksjon (**[x,y,u,v]=numfield(n);**) som beregner hastigheter ved hjelp av kall på funksjonen **streamfun** og Matlab funksjonen **gradient** som beregner $\nabla\psi$ numerisk.

(d) Funksjonen **gradient** gir bare en tilnærmelse til gradienten til ψ slik denne er definert i likning (2). Hvor god denne tilnærmelsen er vil avhenge av oppløsningen, dvs. antall punkter vi bruker i x- og y-retning (n). En måte å kvantifisere feilen på er å finne maksimalt avvik for alle beregnede verdier i området. Du gjør da følgende:
Vi betegner hastighetene fra **velfield** og **numfield** med hhv. $\vec{v}$ og $\vec{v}_n$ og søker et feilmål definert ved $E = \max|\vec{v} - \vec{v}_n|$, der vi tar maksimum over alle punktene i matrisene. Strengt tatt burde vi dividert E på en typisk verdi for $|\vec{v}|$ for å finne en relativ feil. Siden $|\vec{v}|$ er mellom 0 og 1 slipper vi dette.
Bruk **velfield** og **numfield** i en funksjon som beregner E (**[avvik]=feil(n);**), og benytt denne i et skript (**maxfeiliste.m**) som lister, eller plotter, feilen sammen med Δx for $n = 5, 8, 10, 15, 20, 25, 30$.

Matlab-hint

- En array med n jevnt fordelte punkter i intervallet $[a, b]$ lages best med **x=linspace(a,b,n);**.
- Feks. en pdf utgave av gjeldende figur i Matlab lages med
  print('-dpdf',filnavn);
  Ønsker du eps bytter du ut '-dpdf' med '-depsc'.
- I figurtitler kan det være gunstig å kombinere tekster og verdien av variable som brukes i Matlabkoden. Er n et heltall kan man feks. benytte
  tit=sprintf('%s%d','Antall punkter =',n);title(tit);
- De som ønsker å skrive hele, eller deler, av obligen i latex oppmuntres til det. Det er da nyttig å inkludere alle **.m** filer i latex koden. Dette gjørest enkelt ved verbatim-kommandoen
  \verbatiminput{streamfun.m}
  som er brukt ovenfor. En må huske å inkludere verbatim i starten på latex fila
  \usepackage{verbatim}

A.4 Laboratorieoppgave "Tid og frekvens" i FYS2150 Eksperimentalfysikk

Tid og frekvens

Dag Kristian Dysthe and Anja Røyne*
Fysisk institutt, UiO
(Dated: February 26, 2009)

Målet i denne oppgaven er å få et bevisst forhold til hvordan man måler tid. Vi vil gå fra helt grunnleggende til mer avanserte metoder. Et viktig poeng er også hvordan man beregner "usikkerhet" og å finne ut av hvordan man skal minimere denne. Sist men ikke minst skal dere trene på å lage en rapport og å føre en labjournal.

I. BAKGRUNN

Måling av tid baserer seg på periodiske hendelser. Årssyklusen og soluret er de eldste og best kjente tidsmålerne. Hvor nøyaktig man kan måle tid kommer an på hvor regulære periodene er (f.eks. jordrotasjonen) og hvor fint man kan dele opp perioden i underenheter (f.eks. antall streker på sirkelen i et solur). Det vi kaller en klokke i dag er et instrument med noe som svinger (f.eks. pendelen i et pendelur, en elektronisk svingekrets, eller lysbølgen fra en veldefinert kvantemekanisk overgang) og noe som teller antall svingninger. Dagens beste tidsstandard har en nøyaktighet på under ett sekund på 30 millioner år. Det er den mest nøyaktige standarden for noen fysisk enhet vi kan måle på. Se f.eks. http://tf.nist.gov/general/museum/847history.htm.

Vi skal i denne øvelsen bare bruke "helt vanlige" oscillatorer. Vi skal se på hva som ligger i begrepene "nøyaktighet" og "presisjon" og hvordan man som fysiker forholder seg til at alt man måler har en "usikkerhet". Kapittel 1, 2 og 3, samt avsnitt 7.4 og 7.5 i Squires er pensum og nyttig for å løse denne øvingen.

II. FORHÅNDSOPPGAVE

A. Formål

Å gi forståelse av hva en normalfordeling, en middelverdi og et standardavvik er. Å trene på bruk av Matlab. Å lage en første liten rapport. Å gi en introduksjon til innleveringsformen som vil brukes i dette kurset.

B. Innleveringsform og frist

Denne oppgaven skal leveres i Fronter før klokken 12:00 dagen før labøvingen skal gjøres. Innleveringen skal inneholde skriftlige svar på spørsmålene, Matlab-koden som dere lager og evt. grafikk som illustrerer svarene. Det er mulig å skrive og tegne for hånd for så å scanne arkene til en pdf-fil. Hele innleveringen skal samles i **en** enkelt pdf-fil som leveres i Fronter. Som et eksempel på hvordan man kan lage et dokument til innlevering ligger tex-filen til dette dokumentet på kurssidene. Denne pdf-filen er generert fra filene `tid_frekvens.tex`, `prelab_oving1.m` og `gaussian.png` med kommandolinjen

```
pdflatex tid_frekvens
```

på en av instituttets Linux-maskiner. Hvis dere heller vil bruke Word, OpenOffice eller noe annet så må dere finne utav det selv.[1] Men alle innleverte dokumenter skal være i pdf-format, intet annet format godtas.

C. Oppgaver

Oppgavene er gitt i m-filen i appendiks A. Fyll inn din Matlab-kode i din egen kopi av denne m-filen og kjør den i Matlab. Følgende spørsmål skal besvares i et tekstdokument (ikke i m-filen):

- Hvilken beregningsmetode bruker Matlab i sin funksjon "std"?
- Oppgi et intervall der forventningsverdien (mu) skal være med 95% sikkerhet
- Normaliser fordelingen f og plott den sammen med normalfordelingen med ditt estimerte middelverdi og standardavvik. Inkluder plottet i besvarelsen.
- I hvilken grad vil du si at fordelingen f er normalfordelt? Bruk histogrammet, kumulativ fordeling og evt. normalfordelingsplott for å diskutere det.

I appendiks B har vi inkludert plottet fra en realisering av øvingen. Figur 1 viser histogram av tilfeldige tall med gaussfordeling med 1000 "målinger". Hvordan ser fordelingene ut med bare 100 eller 10?

III. LABORATORIEØVING

A. Innleveringsform og frist

I løpet av denne øvingen skal dere notere alle observasjoner, ideer, måleresultater og beregninger o.l. i en labjournal. Grafiske plott fra målinger og beregninger i Matlab kan skrives ut på skriveren, klippes til og limes inn i labjournalen etterhvert som dere arbeider. Kl.

*Electronic address: d.k.dysthe@fys.uio.no

13:00 skal arbeidet med eksperimentene avsluttes. Tiden fra 13:00 til 13:30 kan brukes til å gjøre ferdig labjournalen samt å scanne sidene i labjournalen, sette dem sammen til ett pdf-dokument og levere det dokumentet i Fronter. Det vil ikke bli gitt karakter på denne journalen, men den vil for noen danne grunnlag for en rapport som skal leveres i uke 8. Den rapporten vil bli gitt karakter på.

B. Oppgaver

1. oppgave: Timeglass og pendel

Opp til fire grupper deler ett timeglass. Hver gruppe har en pendel.

- Mål tiden på timeglasset i antall pendelsvingninger.
- Hva vil dere angi som nøyaktigheten i målingene deres?
- Mål avstanden fra opphengspunktet til massesenteret til pendelen.
- Anta at tyngdens akselleråsjon er 9.81 m s^{-2} og beregn svingetiden for pendelen i sekund.
- Hva er de viktigste kildene til usikkerhet?
- Hvilke er tilfeldige og hvilke er systematiske feil?
- Sett tall på (estimert eller ved gjentatte målinger) alle feilkildene.
- Dere har nå "kalibrert" timeglasset. Hvilken presisjon og nøyaktighet har dette timeglasset som tidsmåler?

2. oppgave: Pendel og stoppeklokke

- Mål tiden på pendelen med en stoppeklokke. Bruk mellomtidsfunksjonen til å få fortløpende, gjentatte målinger
- Er mellomtidene normalfordelt?
- Hva er nøyaktighet, presisjon og hovedfeilkilder i målingen av svingetiden til pendelen?
- Basert på den målte svingetiden til pendelen, hva er nå presisjonen til timeglasset som tidsmåler?
- Mål timeglasset noen ganger med stoppeklokken mens dere klargjør til oppgave III B 3. Hva vil dere nå si er nøyaktigheten til timeglasset som tidsmåler? Kommenter presisjonen dere anga i forrige spørsmål.

3. oppgave: Pendel, fotodiode og 20MHz klokke

Nå skal dere bruke en fotodiode til å måle når pendelen passerer et visst punkt i banen sin. Fotodioden består av en lysdiode som sender ut IR-lys og en lysfølsom diode som gir ut 5 Volt når den mottar reflektert lys fra lysdioden og 0 Volt når den ikke mottar lys. Fotodioden krever 15 Volt drivspenning.

Last ned "svingeperiode.m" til lab-PC-en, start Matlab og åpne "svingeperiode.m" i editoren. Akvisisjonsboksen, NI USB-6211, er styrt av PC-en via en USB-port. Den har en intern svingekrets (20 MHz) som holder takten på når den foretar seg noe. Boksen kan gjøre en rekke ting, men i dag skal dere bare bruke en analoginngang, det vil si to kontaktpunkter der boksen måler spenningen og omformer det til et digitalt tall som er proporsjonalt med spenningen. Den kan måle spenninger opp til 250 000 ganger i sekundet. Det uttrykkes som at samplingraten er maksimalt 250 kHz. Siden taktholderen er på 20 MHz skal variasjonen i perioden mellom to samplinger være mindre enn $5 \cdot 10^{-8}$ s.

Bruk loddet som reflektor eller klistre en bit aluminiumstape på loddet som reflektor. Koble et multimeter på utgangen fra fotodioden for å teste hvor det er best å plassere fotodioden i forhold til reflektoren på pendelen. Koble så måleledningene fra akvisisjonsboksen (ledningene skal være koblet til AI0 (inngang 15) og AI GND (inngang 28)) til utgangen på fotodioden. Nå kan dere sette igang pendelen og kjøre scriptet "svingeperiode.m" for å måle svingetiden til pendelen.

- Hvordan er svingetiden nå sammenlignet med målingen med stoppeklokke?
- Betyr det noe
  - hvilken samplingsfrekvens dere bruker?
  - hvor i pendelbanen fotodioden plasseres?
  - hvor stort utslaget av pendelen er?
- Er mellomtidene normalfordelt?
- Hva er nøyaktighet, presisjon og hovedfeilkilder i målingen av svingetiden til pendelen?
- Basert på den målte svingetiden til pendelen, hva er nå presisjonen til timeglasset som tidsmåler?

4. Ekstraoppgave

Her må to grupper samarbeide. Lag et nytt Matlab-script der dere sender ut en tidsvarierende spenning (AO = analogoutput('nidaq','Dev1');, addchannel(AO,1); osv., bruk hjelpsidene i Matlab). Velg selv tidsvariasjon og frekvens. Finn en høyohmig motstand i skapet og koble både output og input fra begge gruppene sin akvisisjonsboks over motstanden. Hvilken frekvens og hvor mange svingeperioder må dere måle over for å

bestemme forskjellen i klokkefrekvensen til de to akvisisjonsboksene? Hvor stor er forskjellen?

APPENDIX A: M-FIL TIL PRELAB

Denne filen ligger på kurssidene på Fronter.

```
%Denne m-filen er ikke komplett og lar seg ikke kjøre slik som den er nå
%Hjelpesidene til Matlab på "normal distribution" er nyttige
%
%En advarsel: bruk av for-løkker og if-strukturer går veldig tregt i
%Matlab. Lær deg å bruke lineæralgebra (vektor- og matrise-operasjoner)
%og Matlab sine innebygde rutiner i stedet.
%
%Generer et normalfordelt datasett
N=1000; %Varier dette tallet og se på resultatet
mu=?; %velg et tall for middelverdi av fordelingen
sigma=?; %velg et tall for standardavvik av fordelingen
f=normrnd(mu,sigma,N,1);

%Finn estimat for mu (middelverdi), sigma (standardavvik)
%Bruk funksjonen "sum" i Matlab og bruk (3.14) og (3.30) i Squires.
%Hvilken beregningsmetode bruker Matlab i sin funksjon "std"?
muhat=sum(f)/N; %=mean(f);
sqmean=? %fortsett å fylle inn egen kode her

%Hva er standardavviket til middelverdien, sigma_m?
%Angi intervallet [muhat-sigmahat_m muhat+sigmahat_m]
%Er mu innenfor dette intervallet?
sigmahat_m=?
[muhat-sigmahat_m muhat+sigmahat_m];
s=?
%Oppgi et intervall der forventningsverdien (mu) skal være med 95% sikkerhet
[? ?]

%Hvor mange ganger måtte du generere et nytt datasett før mu var
%utenfor intervallet ovenfor (kalt 95%-konfidensintervallet)?

%Bruk funksjonen "hist" til å visualisere datasettet som en fordeling
[n,xout]=hist(f,100);
%Normaliser fordelingen og plott den sammen med normalfordelingen
%med ditt estimerte middelverdi og standardavvik
%Kommenter hva kodelinjene nedenfor gjør.
area=sum(n(2:end).*diff(xout));
x=-40:40;
hold off
plot(xout,n/area,'ok')
axis([-40 40 0 1.05*max(n/area)])
hold on
[n,xout]=hist(f,10);
area=sum(n(2:end).*diff(xout));
plot(xout,n/area,'sk','MarkerFaceColor','k')

plot(x,normpdf(x,muhat,s),'k-')
line([muhat muhat],[0 1],'Color','k','LineStyle',':')
line([muhat-s muhat-s],[0 1],'Color','k','LineStyle',':')
line([muhat+s muhat+s],[0 1],'Color','k','LineStyle',':')
xlabel('x','FontSize',12)
```

```
ylabel('f(x)','FontSize',12)
legend('Gaussian random, 100 bins','Gaussian random 10 bins',...
'Gaussian theoretical','Location','NorthWest')
hold off
%Klikk på File>Save As på figuren og velg jpg, png eller pdf som format

%Kumulativ fordeling F(X>M)
%Her slipper man å velge boksstørrelse som i histogram
%Kommenter koden for å foklare hva som gjøres og hva plottene betyr.
%Gjør plottet klart til bruk i rapporten og inkluder det der.
hold off
F=sort(f);
M=(1:N)/N;
subplot(2,1,1),plot(F,M,'k')
hold on
Erf=(1+erf((F-muhat)/(sqrt(2)*s)))/2;
subplot(2,1,1),plot(F,Erf,'k:');
hold off
Err=Erf-M';
subplot(2,1,2),plot(Err,'k.')


%Bruk Matlab-funksjonen "normplot" for å sjekke videre om datasettet er
%normalfordelt. Sammenlign med normalfordelingsgraf for et datasett
%generert med "rand". Kommenter det du får ut. Se forklaring av
%normalfordelingsgraf i Matlab help og:
%http://www.itl.nist.gov/div898/handbook/eda/section3/normprpl.htm
normplot(f)
normplot(rand(10))
```

APPENDIX B: INKLUDERING AV GRAFIKK I LATEX

Figuren vi fikk ut i vår realisering av oppgaven så ut slik som vist i figur 1. Om figurer i rapporter: En figur med figurtekst skal være selvforklarende. Alle linjer, symboler, tekst og tall skal være lett lesbare i den størrelsen figuren gjengis. Man ser tydelig av figur 1 at Matlab sine standard fonter er litt pinglete, men fortsatt lesbare når figuren krympes til å passe inn i dette rapportformatet med to kolonner. Så langt mulig bør forskjellige datasett i en figur kunns skilles fra hverandre selv om figuren skrives ut i svart hvitt. Det vil si at man bør bruke linjetyper og symboler for å skille datasettene istedet for farger.

APPENDIX C: UTSTYRSLISTE

- Timeglass
- Pendel (lodd, tråd, festeanordning)
- Stoppeklokke
- Fotodiode (svart liten boks)
- Måleledninger
- Meterstokk
- Spenningsforsyning
- Akvisisjonsboks NI USB-6211
- PC

[1] OpenOffice Writer har sin egen pdf-generator som ser ut til å fungere bra. De gratis windows-programmene "primo pdf" og "pdf creator" kan installeres som printere som lager en pdf-fil av hva som helst.

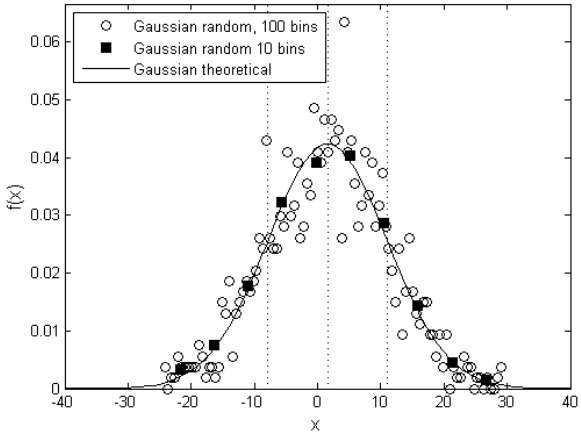


FIG. 1: Fordelingsfunksjon fra gaussisk tilfeldig tallgenerator med 1000 realiseringer og tilsvarende teoretisk gaussfordeling. De stiplede linjene angir forventningsverdien til fordelingen og et 68% konfidensintervall. Antall bokser i histogrammet påvirker spredningen ved visualisering av fordelingen.

Tillegg B

Oppgaver fra NTNU

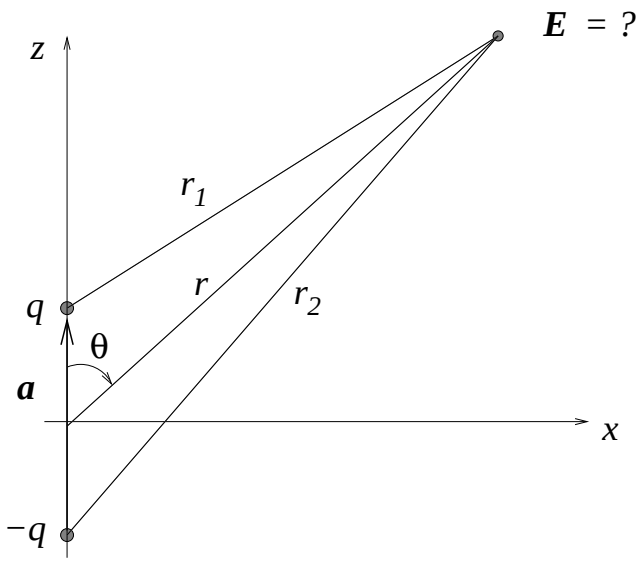
B.1 Øvingsoppgave i TFY4155 Elektromagnetisme

Institutt for fysikk, NTNU
TFY4155/FY1003: Elektromagnetisme
Vår 2009

Øving 4

Veiledning: Mandag 9. februar og fredag 13. (evt 6.) februar
Innleveringsfrist: Fredag 13. februar kl 12.00

Oppgave 1



I forrige øving betraktet vi en elektrisk dipol, bestående av to punktladninger $\pm q$ lokalisert på z-aksen i $z = \pm a/2$. Vi regnet ut det eksakte potensialet $V_e(x, z)$ og fant

$$V_e(x,z) = \frac{q}{4\pi\varepsilon_0}\left(\frac{1}{\sqrt{x^2 + (z - a/2)^2}} - \frac{1}{\sqrt{x^2 + (z + a/2)^2}}\right)$$

Deretter viste vi at potensialet i stor avstand fra dipolen ($r \gg a$) blir tilnærmet lik (indeks a for "approximately")

$$V_a(r,\theta) = \frac{qa\cos\theta}{4\pi\varepsilon_0 r^2}$$

Her er r avstanden fra origo, dvs dipolens midtpunkt, og θ er vinkelen mellom z-aksen og $\boldsymbol{r}$. (Dipolmomentet er $p = qa$.)

a) I denne første deloppgaven skal vi visualisere dipolpotensialet og sammenligne det tilnærmede uttrykket V_a med det eksakte uttrykket V_e. Dette kan vi gjøre ved å skrive et program i Octave (eller MatLab) som regner ut differansen, eller kanskje like gjerne det prosentvise avviket $\Delta = 100 \cdot |(V_e - V_a)/V_e|$ mellom det eksakte og det tilnærmede uttrykket gitt ovenfor, og som plotter $V_e(x, z)$, $V_a(x, z)$ og "feilen" $\Delta(x, z)$ i tre forskjellige figurer.

Noen tips og kommentarer:

- Konverter først $V_a(r, \theta)$ til $V_a(x, z)$.
- Vurder om du skal plotte potensialene i SI-enhet (V) som funksjon av x og z i en passende enhet, *eller* om du skal skrive om uttrykkene for V_e og V_a og plotte dimensjonsløse potensialer som funksjon av dimensjonsløse koordinater.
- Finn et fornuftig område i (x, z)-planet for plottene dine.
- Det kan være lurt å begrense også "funksjonsaksen" i plottene dine, da potensialet blåser opp i nærheten av ladningene.
- Noen kommandoer og funksjoner som du kan få bruk for: linspace, meshgrid, mesh, axis, caxis, figure, xlabel, ylabel, zlabel. Let opp dokumentasjon for hjelp til å bruke disse. (Google for eksempel gnu octave manual (for Octave) eller mathworks matlab manual (for MatLab).)
- Det viktigste poenget med denne oppgaven er å få litt trening i å bruke programmering i tilknytning til det å jobbe med fysikk. Du vil få flere anledninger senere.

b) Tilbake til papir og blyant! Ta utgangspunkt i det tilnærmede uttrykket $V_a(r, \theta)$ og bestem det elektriske feltet $\boldsymbol{E}(r, \theta) = E_r\hat{r} + E_\theta\hat{\theta}$ i stor avstand fra dipolen.
Det oppgis at gradientoperatoren i kulekoordinater er

$$\nabla = \hat{r}\frac{\partial}{\partial r} + \hat{\theta}\frac{1}{r}\frac{\partial}{\partial \theta} + \hat{\phi}\frac{1}{r\sin\theta}\frac{\partial}{\partial \phi}$$

[Fasit: $E_r = p\cos\theta/2\pi\varepsilon_0 r^3$, $E_\theta = p\sin\theta/4\pi\varepsilon_0 r^3$.]
Kontroller om disse uttrykkene er fornuftige for $\theta = 0$ og for $\theta = \pi/2$. Hva med $r = 0$?

c) På grunn av rotasjonssymmetrien omkring z-aksen kan vi f.eks. anta at vi befinner oss i xz-planet. Bestem det elektriske feltet $\boldsymbol{E}(x, z) = E_x\hat{x} + E_z\hat{z}$ uttrykt i kartesiske koordinater for $r \gg a$. Tips: Ta utgangspunkt i uttrykkene for E_r og E_θ i punkt *b*). Tegn gjerne opp en figur og finn på den måten sammenhengen mellom koordinatene (x, z) og (r, θ), og feltkomponentene E_x, E_z og E_r, E_θ.
[Fasit: $E_x = 3pxz/4\pi\varepsilon_0(x^2 + z^2)^{5/2}$, $E_z = p(2z^2 - x^2)/4\pi\varepsilon_0(x^2 + z^2)^{5/2}$.]

d) En *ideell elektrisk dipol* tilsvarer (formelt) at vi lar avstanden a mellom q og $-q$ gå mot null, uten at dipolmomentet $p = qa$ forsvinner. Det innebærer at vi må la $q \to \infty$ og $a \to 0$ samtidig, og på en slik måte at produktet $p = qa$ blir endelig (dvs verken null eller uendelig). I *praksis* tilsvarer dette at vi befinner oss i stor avstand r fra dipolen, slik vi nettopp har gjort i denne oppgaven.
Det elektriske feltet fra en slik ideell elektrisk dipol kan skrives på såkalt *koordinatfri form*:

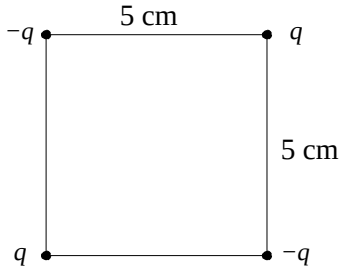
$$\boldsymbol{E}_{\text{dipol}}(\boldsymbol{r}) = \frac{1}{4\pi\varepsilon_0 r^3}\left[3\left(\boldsymbol{p}\cdot\hat{r}\right)\hat{r} - \boldsymbol{p}\right]$$

Vis at dette uttrykket er i samsvar med $\boldsymbol{E}(r,\theta)$ og $\boldsymbol{E}(x,z)$ som du regnet ut i hhv *b*) og *c*).

Oppgave 2 (fra tidligere midtsemesterprøver)

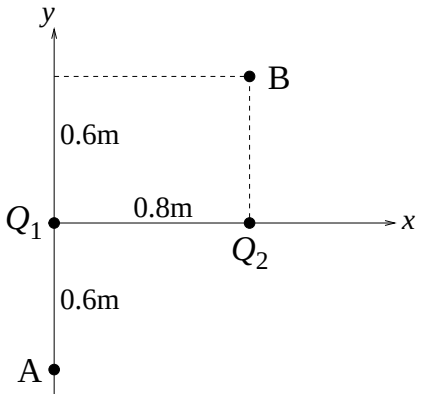
a) Fire punktladninger, to positive og to negative ($q = 9\ \mu$C), er plassert i hjørnene på et kvadrat med sidekanter 5 cm, som vist i figuren. Hva er systemets potensielle energi?

A 19 J
B Null
C -7 J
D -38 J



b) To punktladninger $Q_1 = 69$ nC og $Q_2 = -98$ nC er plassert i xy-planet, som vist i figuren. Et elektron flyttes fra punkt A til punkt B. Hvor stor endring gir denne forflytningen i systemets potensielle energi? ("Systemet" = de to punktladningene og elektronet.) ($1\text{ eV} = 1.6 \cdot 10^{-19}$ J)

A -1 keV
B -1 eV
C 1 eV
D 1 keV

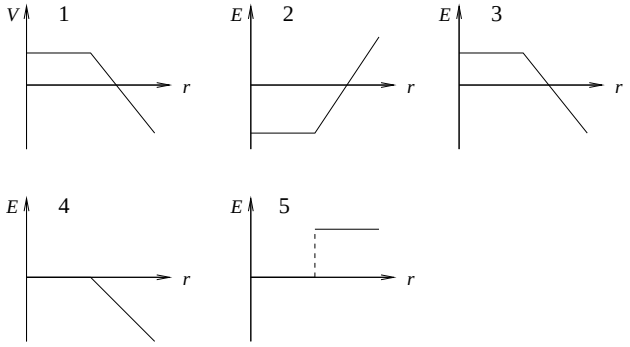


c) Hvor stor er radien til en (kuleformet) ekvipotensialflate på 50 V med en punktladning 10 nC i sentrum? Null potensial velges uendelig langt unna.

A 1.3 m
B 1.8 m
C 3.2 m
D 5.0 m

d) Hvis potensialet V som funksjon av avstanden r fra en ladningsfordeling er som vist i graf nr 1, hvilken graf viser da det elektriske feltet E som funksjon av avstanden r?

A 2
B 3
C 4
D 5



e) Potensialet i et område er

$$V(x, y, z) = (2\ \text{V/m})x + (3\ \text{V/m})y + (4\ \text{V/m})z$$

Da er x-komponenten av det elektriske feltet i dette området

A –2 V/m
B –3 V/m
C –4 V/m
D –9 V/m

B.2 Prosjektoppgave i FY0001 Brukerkurs i fysikk

Prosjekt i FY0001

1 Introduksjon

Numeriske beregninger blir en stadig mer utbredt måte å studere fysiske systemer på. Ettersom datamaskinene blir stadig kraftigere kan man på en helt vanlig pc gjøre simuleringer som for ikke så veldig mange år siden var forbeholdt forskere med tilgang til superdatamaskiner.

2 Oppgavene

Som ble nevnt på forelesningen 18. februrar har jeg to forslag til problemer man kan studere. Om noen ønsker å se på et annet problem er det greit, men det bør ikke være for vanskelig.

2.1 Banebevegelse i gravitasjonsfelt

Gravitasjonskraften som påvirker en ting som går i bane rundt jorden er avhengig av avstanden, som vi ser av gravitasjonsloven:

$$F_g(r) = G\frac{m_1 m_2}{r^2}.$$

Siden $F = ma$ finner vi at akselerasjonen også er avhengig av avstanden. Det er med andre ord ikke helt trivielt å regne på en ting som går i noe annet enn en sirkelbane rundt jorden. Det vi imidlertid lett kan finne er akselerasjonen i et hvilket som helst punkt. Dette kan vi bruke til å finne farten og posisjonen på et senere tidspunkt ved å ta små tidssteg Δt:

Ved tiden t_0:

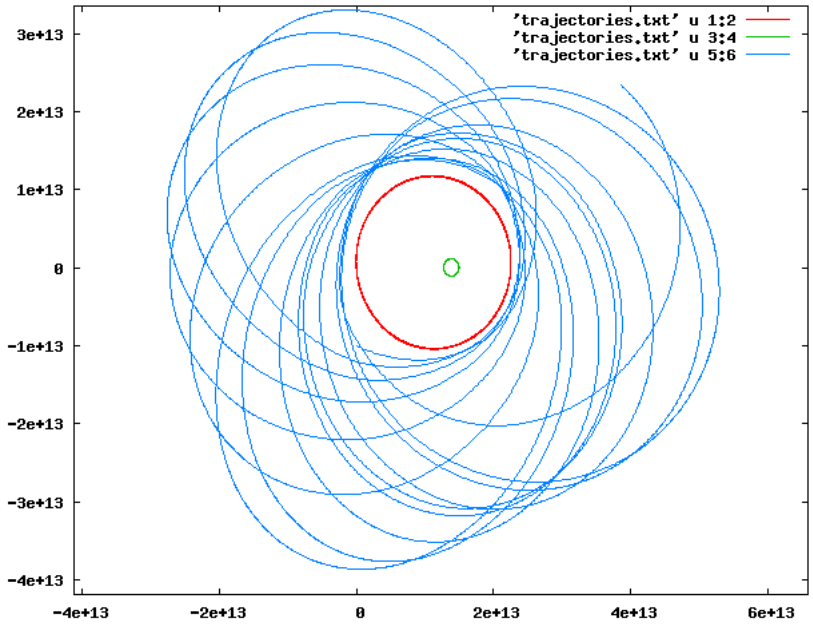
$$\begin{aligned} x &= x_0 \\ y &= y_0 \\ \vec{v} &= \vec{v}_0 \\ \vec{a} &= G\frac{M}{r^2}\hat{r} \end{aligned}$$

Ved tiden $t' = t_0 + \Delta t$:

$$x' = x_0 + v_{0x}\Delta t$$
$$y' = y_0 + v_{0y}\Delta t$$
$$\vec{v}' = \vec{v}_0 + \vec{a}\Delta t$$
$$\vec{a}' = G\frac{M}{r'^2}\hat{r}'$$

På denne måten kan vi finne banen til en ting som beveger seg i gravitasjonsfeltet til jorden. Nøyaktigheten er avhengig av størrelsen på Δt. Denne fremgangsmåten kalles gjerne Eulers metode.

Det enkleste når man ser på et slikt problem er å anta at man har en planet med stor masse, og en ting med liten masse som går i bane, der man antar at planeten står i ro hele tiden. Problemet kan også ganske lett utvides til å inkludere flere legemer, som alle påvirker hverandre med gravitasjonskrefter.



Figur 1: Et dobbelstjernesystem (rød og grønn strek), og en planet som går i bane rundt (blå strek).

2.2 Skrått kast med luftmotstand

For å kunne regne på dette tilfellet må vi anta en modell for luftmotstand. En rimelig antagelse er å si at luftmotstanden er proporsjonal med kvadratet av farten, og alltid har motsatt retning av farten, altså:

$$F_r = -\hat{v}Kv^2,$$

der K er en empirisk bestemt konstant. Vi får dermed det samme tilfellet som i forrige problem, nemlig at akselerasjonen er avhengig av posisjonen. Oppgaven kan løses på samme måte:

Ved tiden t_0:

$$\begin{aligned} x &= x_0 \\ y &= y_0 \\ \vec{v} &= \vec{v}_0 \\ \vec{a} &= -\hat{v}\frac{Kv^2}{m} \end{aligned}$$

Ved tiden $t' = t_0 + \Delta t$:

$$\begin{aligned} x' &= x_0 + v_{0x}\Delta t \\ y' &= y_0 + v_{0y}\Delta t \\ \vec{v}' &= \vec{v}_0 + \vec{a}\Delta t \\ \vec{a}' &= -\hat{v}\frac{Kv^2}{m} \end{aligned}$$

3 Fremgangsmåte

For begge problemene beskrevet her er det samme fremgangsmåte som gjelder. Man vet farten og posisjonen, og man kan regne ut kraften, og dermed akselerasjonen. Så er det bare å bruke dette til å regne ut en ny fart og en ny posisjon en tid Δt senere, og gjenta dette veldig mange ganger.

Å programmere noe slikt er relativt enkelt. Skriver koden som skal til for å regne ut verdiene for neste steg, og så putter man denne koden inn i en såkalt «løkke» som kjøres ønsket antall ganger. I tillegg må man på et eller annet vis plotte resultatene.

4 Aktuelle verktøy

Som nevnt tidligere er mulige verktøy MatLab, Octave eller Python. Jeg anbefaler å bruke Octave, fordi det er gratis, det ligner på MatLab (som er veldig utbredt, men ikke gratis) og det har innebygde funksjoner for plotting, som gjør det enkelt å komme i gang. Om noen foretrekker å bruke Python, MatLab eller noe helt annet, er det også greit, men veiledningen blir innrettet mot Octave.

5 Veiledning

I øvingstimen torsdag 26. februar har vi reservert et datarom (R92, i etasje U1 i Realfagbygget), og der vil det bli gitt veiledning. Litt avhengig av hvor langt vi kommer vil det bli satt opp en veiledningstime til på et senere tidspunkt.

6 Innlevering

Jeg ser for meg at man leverer inn en kort rapport der man inkluderer plottene man har kommet frem til. Man kan jobbe sammen to og to, og leverer en felles rapport. Omtrent en side tekst, pluss figurer, burde holde. Rapporten bør ha en kort beskrivelse av problemet og fremgangsmåten, og noen tanker om resultatet. Innlevert rapport teller som to godkjente øvinger.

Tillegg C

Emnekoder ved NTNU og UiO

Emnekode	Emnenavn
FYS2160	Termodynamikk og statistisk fysikk
EXPHIL03	Examen philosophicum
FYS1210	Elektronikk med prosjektoppgaver
FYS2150	Eksperimentalfysikk
FYS2130	Svingninger og bølger
FYS2140	Kvantefysikk
AST1100	Innføring i astrofysikk
GEF1000	Klimasystemet
FYS1120	Elektromagnetisme
MAT1120	Lineær algebra
FYS-MEK1110	Mekanikk
MEK1100	Feltteori og vektoranalyse
MAT1110	Kalkulus og lineær algebra
INF1100	Grunnkurs i programmering for naturvitenskapelige anvendelser
MAT-INF1100	Modellering og beregninger
MAT1100	Kalkulus

Tabell C.1: Emnekoder for obligatoriske fag ved Fysikk, UiO.

Emnekode	Emnenavn
TMA4100	Matematikk 1
TFY4145	Mekanisk fysikk
TDT4105	Informasjonsteknologi grunnkurs
EXPH0001	Filosofi og vitenskapsteori
TMA4105	Matematikk 2
TMA4115	Matematikk 3
TFY4155	Elektromagnetisme
TMT4110	Kjemi
TMA4120	Matematikk 4K
TFY4160	Bølgefysikk
TEP4105	Fluidmekanikk
TIØ4256	Teknologiledelse 1
TMA4245	Statistikk
TFY4165	Termisk fysikk
TFY4215	Kjemisk fysikk og kvantemekanikk
TDT4102	Prosedyre- og objektorientert programmering
TFY4185	Måleteknikk
TFY4230	Statistisk fysikk
TFY4240	Elektromagnetisk teori
TFY4250	Atom- og molekylfysikk
TFY4190	Instrumentering
TFY4195	Optikk
TFY4220	Faste stoffers fysikk
TFY4205	Kvantemekanikk
TFY4225	Kjerne- og strålingsfysikk
EiT	Eksperter i Team. Tverrfaglig prosjektarbeid.

Tabell C.2: Emnekoder for obligatoriske fag ved Teknisk fysikk, NTNU.