
RAPPORTSKRIVING I L^AT_EX

— Et kræsjkurs for matematikkstudenter —

Oppdatert utgave 2005
Håvard Berland

Institutt for matematiske fag — NTNU 2001-2005

Innhold

1	Innledning	1
2	Prosjektstart	3
2.1	Dataverktøy	3
2.1.1	Operativsystem	3
2.1.2	Emacs	3
2.1.3	Filbehandling	3
2.2	Gruppearbeid	4
2.2.1	Spør drift@stud.math.ntnu.no	4
2.2.2	Gi katalogen din grupperettigheter	4
2.3	Prosjektkatalog	4
3	Filbehandling og filstruktur	5
3.1	Viktige UNIX-kommandoer	5
3.2	Filstruktur for prosjektet	7
3.3	Filrettigheter for gruppearbeid	8
4	Editoren emacs	11
4.1	De mest brukte kommandoer	11
4.2	Integrering mot $\text{\LaTeX}$ — $\text{\LaTeX}$	12
4.3	$\text{\LaTeX}$	13
4.4	Tastaturtips	13
4.5	Stavekontroll	14
4.5.1	Bruk	14
4.5.2	Bytte språk	14
5	Introduksjon til $\text{\LaTeX}$	15
5.1	Noen $\text{\LaTeX}$ konsepter	15
5.2	Kompilering av $\text{\LaTeX}$ dokumenter	16
5.3	Visning av rapporten	17
5.4	Utskrift av rapporten	17
5.5	Slides fra $\text{\LaTeX}$ -foredrag	17
6	Mal for $\text{\LaTeX}$-filene	29
6.1	Oppdeling av rapporten	29
6.1.1	Fileksempel: master.tex	30
6.1.2	Fileksempel: forside.tex	32
6.1.3	Fileksempel: bakgrunn.tex	33

7	Matematikk i $\text{\LaTeX}$	35
7.1	Hvordan lære seg å skrive matematikk	35
7.2	Eksempler	35
7.3	Ligningsreferanser	36
7.4	Noen huskereglar	36
7.4.1	Husk mattemodus i løpende tekst	36
7.4.2	Derivert	37
7.4.3	Integral	37
7.4.4	Operator- og funksjonsnavn	38
7.4.5	Tekst inni ligninger	38
7.5	Definere egne kommandoer	38
8	Figurer i rapporten	41
8.1	Bitmap- vs. vektorgrafikk	41
8.2	Inkludering av figurer i $\text{\LaTeX}$	42
8.2.1	Bitmap-grafikk	42
8.2.2	Flere figurer i samme figur — subfig	43
8.3	Tegning av figurer med Xfig	44
8.3.1	$\text{\LaTeX}$ kommandoer inni xfig-figurer	44
8.4	Inkludering av figurer fra Matlab	45
8.4.1	Kildekodeeksempel	45
8.4.2	Inkludering av figurer fra Matlab med $\text{\LaTeX}$	45
8.5	Inkludering av figurer fra Gnuplot	45
8.5.1	Kildekodeeksempel	46
8.5.2	GnuPlot og $\text{\LaTeX}$	46
9	Hvordan lage referanseliste/bibliografi	47
10	Hvordan lage register/indeks	51
11	Bruk av verktøyet make	53
11.1	Eksempel på kokbar Makefile	53
12	Lage slides med Beamer	57
12.1	Eksempel	57
12.1.1	Fileksempel: presentasjon.tex	57
12.1.2	Overlays	59
12.1.3	Andre effekter	59
12.1.4	Bytte utseende på presentasjonen	59
12.2	Kompilering	59
12.3	Vise presentasjonen	60
	Bibliografi	61
A	Bruk av cvs for versjonskontroll og gruppearbeid	63
A.1	Hvorfor bruke cvs	63
A.2	Alternativ til cvs— Subversion	63
A.3	Oppsett av cvs for gruppeleder	63
A.4	Oppsett av cvs for alle gruppemedlemmer	64
A.5	Legge inn filer i prosjektet	65
A.6	Mer om commit og update	65
A.6.1	Sjekke inn en fil om gangen	65
A.6.2	Konflikter	65
A.6.3	Meldinger fra cvs under cvs update	66
A.7	Sletting av filer i prosjektet	66

A.8	Hvilke filer skal med i repository?	66
A.9	Logg og versjonsnummer for hver fil	67
A.10	Få mail hver gang noen gjør en commit	67
A.11	Videre lesning	68
B	Filtyper	69
B.1	$\LaTeX$ kildefiler	69
B.2	Filer generert av $\LaTeX$ og venner	69
B.3	Grafikkfiler	70
B.4	Andre filer	71

Kapittel 1

Innledning

Dette er et introduksjonshefte for å komme i gang med å skrive rapporter i $\LaTeX$. Det er primært rettet mot studenter som begynner i 3. klasse industriell matematikk, men kan leses av de fleste andre som har lignende skriveoppgaver.

Første utgave av dette heftet kom i 2001, med små inkrementelle oppdateringer etter som verdenen utviklet seg. I 2005 var det behov for en gjennomgang av hele heftet for å være oppdatert nok.

Kurset er laget med tanke på å skrive prosjektrapporter i matematikk (eller forsåvidt andre fag). For en tredjeklassing står valget mellom å skrive rapporten i noe som studenten vet han behersker, i Microsoft Word, eller å bruke det profesjonelle verktøyet $\LaTeX$ som “alle andre” sier bør brukes, men som ikke er like enkelt å bruke for første gang som Word er.

Begynnernivået (som den jevne student er på) i Word strekker ikke til for å skrive rapporter av tilfredstillende kvalitet. Dermed må man uansett bruke tid på å lære seg Word. Et begynnernivå i $\LaTeX$ strekker derimot lengre, og når man først har kommet i gang, så er den tekniske veien kort til et tilfredstillende resultat på rapporten. I tillegg så må man kunne veldig mye Word for å lage bra layout. For $\LaTeX$ er det omvendt, her må man kunne veldig mye $\LaTeX$ for å lage dårlig layout.

Heftet dreier seg mer om rammeverket og verktøyene som ligger rundt $\LaTeX$, enn $\LaTeX$ selv. Det finnes et utall introduksjoner til $\LaTeX$ på nett som man kan lese, og da er det bedre å holde sidetallet i dette heftet nede. Slides fra et muntlig foredrag i $\LaTeX$ er inkludert i Kapittel ??, og det gis referanser videre til hvor man lærer resten. Dette heftet går noe videre for å gi studenten en flying start ved å gi kokbare filer med mange $\TeX$ nikaliteter allerede innarbeidet, som man kan fortsette å skrive på, dermed kan man forhåpentligvis bruke minst mulig tid på å lære detaljer, og kan heller konsentrere seg om det matematiske innholdet.

Kræsjkurset har en webside hvor du finner mer informasjon, de malene som er nevnt i kapittel 6, og mange referanser til videre dokumentasjon. Se

<http://www.stud.math.ntnu.no/kurs/>

Kapittel 2

Prosjektstart

Prosjektet ditt handler om og begynner med matematikk, ikke rapportskriving. Det kan likevel lønne seg i en tidlig fase å gå gjennom deler av dette kurset og gjøre noe av det forberedende arbeidet med en gang.

2.1 Dataverktøy

2.1.1 Operativsystem

Ved Studentdatalaben til Institutt for Matematiske fag satses det på UNIX i motsetning til Windows som det satses på ved nesten alle andre datasaler. Dette kommer av at det er et mye mer effektivt system for den snart datavante matematikkstudent (du!), og derfor er dette kurset er lagt opp for bruk av UNIX. Operativsystemet på datasalene heter FreeBSD, men forskjellen på de ulike UNIX-variantene (Linux er en UNIX-variant akkurat som Linux er det) skal du ikke merke noe til i denne sammenhengen.

Det er ikke noe problem å gjøre rapportskrivingen på en annen datasal eller på egen laptop (med enten Windows eller Linux på), men du er da muligens avhengig av å sette opp en fungerende installasjon av $\text{\LaTeX}$ selv hvis det ikke allerede er gjort.

2.1.2 Emacs

Emacs er en teksteditor det anbefales å bruke hver gang du skal taste inn tekst på datamaskinen. Det finnes ingen tekst-editor som heter “latex”. Latex vet lite om å putte tekst inn i datamaskinen, det er ikke sin jobb. Emacs startes ved å skrive emacs på UNIX-kommandolinja. Kapittel 4 beskriver emacs nærmere.

2.1.3 Filbehandling

Til en viss grad må man lære seg litt filbehandling. Til det er enkle UNIX-kommandoer det enkleste og kraftigste verktøyet, men ikke det enkleste å lære. En filbehandler ala

Explorer i Windows finner du også på UNIX-maskinene (men slike verktøy blir allikevel aldri like kraftfulle som en håndfull av kommandoer). En enkel innføring er gitt i kapittel 3

2.2 Gruppearbeid

De fleste prosjekter er ikke enmannsprosjekt. Dermed må man tilpasse seg til andre studenter også, og man må til en fornuftig integrering av de forskjellige deltakernes bidrag til rapporten.

Det første en bør gjøre er å få laget en UNIX-gruppe på datamaskinene. Da kan man la gruppa eie en katalog og sette rettigheter slik at alle i gruppa kan skrive og slette og gjøre hva man vil, mens alle andre ikke engang vil kunne lese (koke) det gruppa di produserer.

2.2.1 Spør drift@stud.math.ntnu.no

om du kan få ei gruppe på systemet, og si fra om brukernavnet til alle i prosjektgruppa, og du vil i løpet av kort tid få opprettet ei slik gruppe.

2.2.2 Gi katalogen din grupperettigheter

Det trengs kun en katalog, som alle prosjektdeltakere bruker. Alle bruker samme katalog Derfor tar kun en deltaker på gruppa og går igjennom generering av den fornuftige filstrukturen som blir beskrevet i kapittel 3, men gir katalogen grupperettigheter (kommandoen `chmod`).

Etterpå lager de andre på gruppa seg linker til prosjektkatalogen med kommandoen `ln` (for å slippe å skrive en lang `cd`-kommando hver gang de vil inn der). Se neste kapittel for utførlig instruksjon angående dette.

2.3 Prosjektkatalog

Neste steg er å lage denne prosjektkatalogen og fylle den med innhold. UNIX-kommandoer for å fylle denne går igjennom i kapittel 3 og tips til filstruktur finnes i seksjon 3.2. Maler for å begynne skrivingen på finnes i kapittel 6. På dette stadiet er det også tiden for å vurdere om man vil bruke cvs for å holde styr på filene og for å forenkle samarbeidet mellom deltakerene. Cvs beskrives i appendiks A. Cvs er et utrolig nyttig verktøy, men blir “enda en ting” som må læres.

Kapittel 3

Filbehandling og filstruktur

Ditt prosjekt med rapport kommer til slutt til å bestå av mange filer. Organisering på dem er viktig for å kunne jobbe effektivt og for å kunne finne igjen ting senere. Det gis her et forslag til filstruktur som det kan være fornuftig å starte på.

Lag en katalog for hele prosjektet som du legger på på hjemmekatalogen din. La oss anta at du kaller denne for prosjekt. For å klare å komme i gang med det, må du gjennom følgende:

3.1 Viktige UNIX-kommandoer

`mkdir` Brukes for å lage en katalog i den katalogen du står i (*make directory*). Denne er den aller første kommandoen du skriver når du først skal begynne på prosjektet

```
syLOW:~:$ mkdir prosjekt
```

`cd` Brukes for å gå inn i en annen katalog (*change directory*). Trolig det første du gjør når du setter deg ned ved datamaskinen for å fortsette på prosjektet ditt:

```
syLOW:~:$ cd prosjekt
syLOW:~/prosjekt:$
```

`ls` Brukes for å skrive ut hvilke filer som finnes i katalogen du er i (*list directory contents*).

```
syLOW:~/prosjekt:$ ls
Makefile      master.tex    innledning.tex
```

Ofte har du lyst til å se litt mer informasjon om filene dine, da slenger du på opsjonen `'-l'` (*long listing format*)

```
syLOW:~/prosjekt:$ ls -l
-rwx----- 1 berland fall_s      429 Apr  2 19:36 Makefile
-rwx----- 1 berland fall_s    564288 Apr  2 19:36 master.ps
-rwx----- 1 berland fall_s     3808 Apr 19 08:34 master.tex
```

rm Brukes for å slette filer og/eller kataloger (remove files or directories).

```
syLOW:~/prosjekt:$ rm master.tex
```

og så forsvinner en av latex-kildefilene dine. Kanskje ikke så smart. Skal du slette hele kataloger, må du ta med '-r' som opsjon

```
syLOW:~/prosjekt:$ rm -r kopi_av_gfx
```

less er en enkel fil-leser. Hvis du bare skal ha en rask kikk på en tekstfil er denne raskere enn å starte opp emacs (for eksempel).

```
syLOW:~/prosjekt:$ less master.tex
```

cat er en enda enklere fil-leser enn less. Den skriver ut hele tekstfila uten å tenke på at du ikke klarer å lese teksten når den farer ut av vinduet ditt fordi den er litt lang. Men den er kjekk å bruke på korte filer

```
syLOW:~/prosjekt:$ cat Makefile
```

cp Kopierer filer (copy). Har du ei fil kalt utledning.tex og trenger en nesten lik som du vil jobbe videre med, kan du skrive

```
syLOW:~/prosjekt/rapport:$ cp utledning.tex teoremer.tex
```

så kan du jobbe videre med teoremer.tex uten at utledninger.tex blir berørt.

mv Flytter eller bytter navn på filer (move (rename) files). Har du begynt på proposisjoner.tex men finner ut at det ble bare teoremer av det, kan du bytte navn til teoremer.tex ved

```
syLOW:~/prosjekt/rapport:$ mv proposisjoner.tex teoremer.tex
```

ln Lager en symbolsk lenke til en annen fil eller katalog. Dette vil si at du kan få lagt en katalog på for eksempel hjemmekatalogen din selv om den egentlig ikke ligger der. Typisk vil du lage en symbolsk lenke til en prosjektkatalog som en annen på gruppa di har på sin egen hjemmekatalog. For å ikke måtte gå via hans hjemmekatalog hver gang (det ville vært en litt lengre cd-kommando) du skal inn i prosjektkatalogen, lager du bare en symbolsk lenke til denne katalogen.:

```
sylow:~:$ ln -s ~enannenbruker/matteprosjekt matteprosjekt
```

Her må du bytte ut “enannenbruker” med din venns brukernavn. Det siste argumentet sier hva du vil kalle linken din. Her er det egentlig overflødig siden `ln` gir deg det samme navnet hvis du ikke sier noe annet.

`grep` er en kraftig kommando som man ikke kan være rettferdig mot på tre linjer. Men den brukes til å søke i tekst. Tekniske rapporter skal av og til aldri inneholde ordet “vi” for eksempel.

```
sylow:~/prosjekt/rapport:$ grep vi *.tex
```

skriver ut de linjene som inneholder ordet du vil ha bort i alle filer som slutter på `.tex`. Bruker du neste kommando, så kan du i tillegg lære hvordan du skal sortere ut de linjene der de to bokstavene “vi” bare er en del av et større ord.

`man` er kanskje den kjekkeste kommandoen av alle. Med den får du opp en manual for en hvilken som helst kommando som beskriver (litt teknisk) hva en kommando kan gjøre (det er som regel *mye*).

```
sylow:~/prosjekt:$ man ls
```

3.2 Filstruktur for prosjektet

I tilfelle datadelen av prosjektet ditt innbefatter mer enn selve rapporten (Matlab/R-kode og annen programmering kanskje) vil du trolig ha enda en katalog inni prosjekt-katalogen som du allerede har laget. Denne kan for eksempel kalles `rapport`. For programmeringen din kan du lage en annen katalog, for eksempel `kode`.

Figurer bør du ha inni enda en katalog, inni `rapport`-katalogen, som du kan kalle `gfx`. For å lage alle disse katalogene kan altså disse kommandoene skrives:

```
sylow:~:$ mkdir prosjekt
sylow:~:$ cd prosjekt
sylow:~/prosjekt:$ mkdir rapport
sylow:~/prosjekt:$ mkdir kode
sylow:~/prosjekt:$ cd rapport
sylow:~/prosjekt/rapport:$ mkdir gfx
```

Følges dette oppsettet, er du godt rustet mot rot i filene dine. ~~ET~~^{TEX}kommer senere til å lage massevis av andre filer i disse katalogene, så det er et poeng i å ikke grise til katalogene for mye med en gang.

Kommandoen `find` kan (mellom utrolig mye annet nyttig) brukes til å liste ut kataloginnhold rekursivt, og filstrukturen skal altså nå se slik ut:

```

syLOW:~/prosjekt/rapport:$ cd ~      # gå til hjemmekatalogen min
syLOW:~:$ find prosjekt
prosjekt
prosjekt/rapport
prosjekt/rapport/gfx
prosjekt/kode

```

3.3 Filrettigheter for gruppearbeid

Dette beskriver hva du må gjøre for å gjøre prosjektkatalogen din skrivbar for de andre i gruppa di. Først må du ha fått tildelt ei gruppe du kan bruke.

La oss anta at du heter Ole Nilsen og har brukernavn olenils. Du har prosjekt i Lineære metoder og drift har derfor gitt deg ei gruppe som heter olenils-linmet. Kun en person pr. gruppe trenger å gjøre dette.

Det første du må gjøre er å la gruppa di “eie” prosjektkatalogen (som vi antar heter prosjekt og ligger rett på hjemmekatalogen din). Dette gjør du med `chgrp` kommandoen:

```

syLOW:~:$ chgrp olenils-linmet prosjekt
syLOW:~:$ ls -ld prosjekt
drwx----- 4 olenils olenils-linmet      512 Sep 18 12:28 prosjekt/

```

Nå eier gruppa katalogen, men den er fortsatt verken lesbar eller skrivbar for gruppa. Bokstavene `drwx---` angir rettighetene (den første `d` står bare for “drawer”). `r` står for read, `w` for write og `x` for execute. De tre første bokstavene er rettighetene til din egen bruker, de tre neste er for gruppa og de tre siste er for alle andre (“world” er unix-terminologien for alle andre brukere enn deg selv og de på unix-gruppa di). En strek `-` betyr at en rettighet er avskrudd. Du vil typisk sette på `rwX` for gruppa, og kanskje ta bort `r` og `x` for “world” hvis disse finnes (det avhenger av hvilke standardinnstillinger du har) hvis du ikke vil at andre i klassen din skal kunne snoke i prosjektet ditt. (execute (`x`) rettigheter på kataloger betyr at man har lov til å se på hvilke filer som er i katalogen med kommandoen `ls`).

Rettighetene endrer du med `chmod` kommandoen. For å gjøre som indikert over kan du bare skrive

```

syLOW:~:$ chmod 770 prosjekt
syLOW:~:$ ls -ld prosjekt
drwxrwx--- 4 olenils olenils-linmet      512 Sep 18 12:28 prosjekt/

```

Hver av de tre tallene representerer hvilke av de tre bitene `r`, `w` og `x` som skal være satt, og tallet er i det oktale tallsystem (det vil si at du teller 4 for `r`, 2 for `w` og 1 for `x`, så summen 7 kan bare bety at alle tre er satt). Du kan også bruke bokstavkombinasjoner for å få samme resultat uten å forstå tallene. Les man `chmod` for mer forståelse.

I tillegg har du garantert lyst til å gjøre katalogen “group-sticky”. Dette betyr at alle filer som blir laget inni denne katalogen får samme gruppe som prosjekt-katalogen din

(og slik må det være for at dette skal fungere automatisk). Group-sticky kan vi lage med `chmod`-kommandoen, men nå tar vi med et 2-tall også og resultatet blir slik:

```
syLOW:~:$ chmod 2770 prosjekt
syLOW:~:$ ls -ld prosjekt
drwxrws---  4 olenils olenils-linmet      512 Sep 18 12:28 prosjekt/
```

der `x` for gruppa er blitt overskrevet med en `s`.

Når så andre gruppemedlemmer skal legge inn filer her vil ikke disse filene få helt riktige rettigheter, fordi standard oppførsel er å ikke lage filer skrivbare for gruppa som filene tilhører. Alle gruppemedlemmene må derfor selv gjøre filene sine skrivbare for andre hvis andre skal endre på filene, og dette må gjøres med `chmod`-kommandoen. Kun fileier kan endre på rettighetene, ikke de andre på gruppa en gang. En grei kommandolinje som fikser på alle filene dine (men ikke de andre sine) er:

```
syLOW:~/prosjekt:$ chmod -R 770 *
```

der `-R` står for rekursiv og gjør det samme rekursivt for filer i underkataloger. Husk å stå inni prosjektkatalogen hvis du kjører denne kommandoen, hvis ikke må du skrive noe annet i stedet for stjerna (som betyr alle filer).

Husk at du må gjøre dette etter hver gang du har laget ei ny fil. Sjekk regelmessig utskrift fra `ls -l` for å se hvordan rettighetene ser ut for filene i prosjektkatalogen og sjekk også at de har riktig gruppe.

Kapittel 4

Editoren emacs

Emacs er trolig det kraftigste tekstediteringsprogrammet som finnes. Den har i flere tiår vært standard verktøy på UNIX-plattformen for skriving av tekstfiler, da spesielt programmering. Emacs er skrevet i sitt eget programmeringsspråk, *elisp*, en variant av lisp, og det er nettopp dette som har gjort den så kraftig. Alt du kan tenke deg kan konfigureres, selv om det ikke alltid er like lett.

For alle tenkelige programmeringsspråk, så har emacs en spesiell modus for det, inkludert $\text{\LaTeX}$. Emacs vil automatisk gå inn i $\text{\LaTeX}$ -modus (hos oss heter denne AU $\text{\LaTeX}$, seksjon 4.2) når du åpner filer som slutter på `.tex`. Modusen inneholder nyttige verktøy for å kompilere og vise $\text{\LaTeX}$ -dokument, men også fargelegging av kommandoer og korrekt indentering (innrykk).

Da emacs har såpass lang fartstid i bransjen, er ikke alle begreper i menyvalgene lik det du finner i nyere editorer. Eksempelvis så er “Find file” emacs-terminologi for “Open file”. Som regel er ikke forskjellene store nok til at det blir problematisk.

4.1 De mest brukte kommandoer

Tastekommander bruker man fordi man ikke vil bruke tid på å flytte hånda fra tastaturet og bort til musa. Når man ikke lenger husker kommandoene, så finnes de på menyen. På menyen finnes også flere andre kommandoer som ikke har tastekommandoer.

Når det står `C-x C-f` så betyr det at du skal holde nede Control-tasten og trykke ‘x’. Så holder du igjen nede Control-tasten og trykker på ‘f’. Du trenger ikke slippe Control-tasten.

Når det står `M-%` så betyr det at du skal holde nede “Meta”-tasten, som hos oss er *escape*-tasten øverst til venstre. Denne skal du trykke en gang, så slippe, og så trykke prosent-tegnet, som vil si Shift og 5-tasten.

`C-x C-f` Find file. Finner ei fil. Prompten din havner på den nederste linja i emacs-vinduet og du blir bedt om å skrive inn filnavnet på fila du vil.

`C-x C-s` Lagrer fila. Hvis du ikke har lagret før så ber den deg om et filnavn.

- C-s Søk. Søker etter en streng du gir. Trykker du C-s en gang til mens du fortsatt er i søkemodus, søker den en gang til etter det samme.
- C-g Avbryter. Hvis du har trykket noe (kanskje feiltasting) og du angreir på det, kan du alltid trykke C-g for å komme tilbake til tekstediteringen din.
- M-% Search and replace. Spør deg først om hva du vil bytte ut, skriv det inn og trykk enter, skriv så inn det du vil skrive, og trykk enter. Etter dette spør emacs deg om hvert sted den finner den gitte strengen hva du vil gjøre. '!' betyr at du vil bytte ut over alt, 'y' betyr ok på akkurat den strengen der markøren står. Du kan også trykke 'h' for hjelp (alt dette står i den aller nederste linja i emacs-vinduet).
- C-x 2 Splitt opp ditt nåværende vindu i deler, slik at du kan editere to filer samtidig.
- C-x 1 Setter deg tilbake til bare ett editingsvindu (det du har aktivt for øyeblikket).
- C-x C-c Avslutter emacs.
- C-_ Undo. Underscore taster du inn med shift og bindestrek. Denne undo-funksjonen husker langt!

4.2 Integrering mot $\LaTeX$ — AUCTEX

- C-c C-e Sett inn et eller annet $\LaTeX$ -miljø. AUCTEX kommer alltid med et forslag til miljø (environment) som du kan sette inn etter å ha trykket denne kombinasjonen, default er *itemize* (punktlist). Hvis du vil ha et annet miljø, som for eksempel *equation* trykker du de første bokstavene i miljønavnet, for eksempel equ, og så trykker du TAB og AUCTEX vil ekspandere til det du sannsynligvis mener. Trykk ENTER når du er klar, og vips så vil det bli puttet inn kjekk latex-kode i dokumentet ditt. For noen miljø ber også AUCTEX om flere argument, som blir puttet på riktig sted. Denne funksjonen anbefales det virkelig å bruke hele tiden, på alt.
- C-c C-c Kompilerer latex-fila di. Hvis du jobber med et lite dokument er denne ofte greiest, er det et stort prosjekt bruker du kanskje "make" heller (se kapittel 11) og da trenger du ikke denne.
- C-c ` Hvis latex ikke klarer kompilere kildefila di på grunn av feil, vil emacs si at du skal trykke denne kombinasjonen for å få se hva feilen var. Emacs vil også sette markøren på der latex antar feilen ligger (ikke nødvendigvis på helt korrekt linje).
- C-c C-l Vis utskriften under kompilering. Trykk denne rett etter å ha startet kompileringen med C-c C-c hvis du vil følge med på hva som skjer.

For flere kommandoer anbefales det å utforske menyen, og/eller lese manualen til AUCTEX [5]. AUCTEX har også en webside hvor du finner dokumentasjon og eksempler, <http://www.gnu.org/software/auctex/>.

4.3 Re $\text{\TeX}$

Re $\text{\TeX}$ er en annen emacs-tilleggs-pakke som gjør underverker for deg. Den gjør innsetting av kryssreferanser til andre deler av dokumentet ditt (ligninger eller seksjoner) til en lek. Vanligvis må du finne på et bra navn (en “label”) på alle deler du vil referere til senere. Re $\text{\TeX}$ både foreslår navn for deg automatisk, og gjør ting også på en slik måte at du ikke trenger huske hvilke navn du har brukt.

De viktigste tastaturkommandoene i Re $\text{\TeX}$ er

- C-c (Setter inn en “label” på det stedet du er (den prøver å foreslå noe fornuftig automagisk).
- C-c) Sett inn en kryssreferanse. Den spør om hvilken type referanse du vil ha (ligning, seksjon, figur osv.) og lar deg deretter velge fra en liste. Velg med piltaster enter-tasten. Trykk ‘q’ for å avbryte.
- C-c [Sett inn en bibliografireferanse (sitering). Den spør om en “RegExp” (regular expression = avansert søkestreng). Du taster inn en søkestreng (deler av forfatternavn, boknavn eller artikkelnavn f.eks), og den gir deg en liste over de innslagene den fant i bibliografifilene dine. Dette krever at du har lagt inn bibliografifiler som beskrevet i Kapittel 9.
- C-c & Se på en kryssreferanse. Hvis du har markøren i et par sekunder over en `\ref`- eller `\eqref`-kommando, så vil du først få et utdrag av hva det refereres til i emacs sin statuslinje nederst i vinduet. Trykker du denne tastekombinasjonen i tillegg, vil emacs-vinduet dele seg og konteksten rundt det det ble referert til blir vist.

Se også “Re $\text{\TeX}$ in a nutshell”, <http://remote.science.uva.nl/~dominik/Tools/reftex/reftex-nutshell.html>. Websiden for Re $\text{\TeX}$ er <http://remote.science.uva.nl/~dominik/Tools/reftex/>.

4.4 Tastaturtips

I $\text{\TeX}$ kildekode brukes symbolene {, [,] og } ofte. På et norsk tastatur er ikke disse enkle å få tastet inn effektivt mens man skriver touch (høyrehånden må trykke både på ALT-GR og på et av tallene 7, 8, 9 eller 0). Et triks på unix-maskiner er derfor å stille inn tastaturopsettet slik at knappen mellom venstre-ALT og venstre-CTRL (Microsoft-knappen) fungerer likt som ALT-GR. Dette kan man gjøre med følgende magiske kommandoer:

```
maskinnavn:~:$ echo "keycode 115 = Mode_switch" | xmodmap -
maskinnavn:~:$ echo "keycode 116 = Mode_switch" | xmodmap -
```

Hvis du sitter på studentsalene til matematikk, så skal allerede dette være gjort, og Bill-knappen skal allerede fungere som ALT-GR.

4.5 Stavekontroll

Det finnes også stavekontroll for emacs. Den som er installert på studentsalen heter “*ispell*”.

4.5.1 Bruk

Når du vil kjøre ispell på fila du har åpent i emacs, trykker du `'M-x ispell-buffer'`. Ispell vil da starte og kjøre på fila (fra begynnelsen av fila). Den vil stoppe på første ukjente ord, og vente på din reaksjon.

Statuslinja helt nederst i emacs-vinduet forteller deg det du kan gjøre. Du kan trykke `'?`' for å få hjelp til hvilke tastekommanduer du kan trykke. De vanligste er `'i'` (insert) for å legge ordet til din egen ordliste over tilleggsord, `'<space>'` for å ignorere ordet eller `'r'` for å bytte ut ordet med noe annet som du skriver inn selv. Dersom ispell finner andre ord i ordlista som ligner på det du har skrevet, vil den også foreslå dette, og gi hvert alternativ et tall fra 0 og oppover. Da kan du trykke direkte på tallet for det riktige ordet (om ispell fant det riktige) og det gale ordet vil bli byttet ut.

4.5.2 Bytte språk

Standardinnstillingen på studentdatasalen er norsk ordliste. Skriver du engelsk vil du bytte dette, ved å taste `'M-x ispell-change-dictionary'`, og så velger du enten `'american'` eller `'british'` (trykk space for å få se liste over tilgjengelige språk).

Kapittel 5

Introduksjon til $\text{\LaTeX}$

Dette kapitlet har med vilje blitt litt utsatt. Det er fordi det som har vært tidligere ikke har vært direkte $\text{\LaTeX}$ -relatert, men det er nå på tide med en *lynkjapp* gjennomgang. Det er absolutt meningen at man ikke skal lære annet av dette enn at man faktisk må lese mer etterhvert.

Hva du bør lese, er enten boka til Leslie Lamport [3], eller introduksjonene som du finner lenker til på websiden til dette kurset, <http://www.stud.math.ntnu.no/kurs/>.

5.1 Noen $\text{\LaTeX}$ konsepter

Latex-kildefiler består av kommandoer som begynner med en “backslash” `\`. Kommandoer har av og til et argument og da kommer det inni klammeparanteser rett etterpå. Latex-filer starter med noe som kalles “preamble” med diverse kommandoer som beskriver innstillinger som gjelder for hele rapporten (språk som brukes, sidestørrelse, hvilke tilleggspakker du vil bruke). Preamble er ferdig når kommandoen `\begin{document}` kommer. Da begynner dokumentet ditt, og det må avsluttes med `\end{document}`.

Fila `master.tex` har introdusert noen konsepter (latex-kommandoer) som ikke enda er forklart. Den beste kilden til dette er *hele* [3, kapittel 2] men det gis her en lynkjapp oversikt.

Dokumentklasse bestemmer hvilken type dokument du skriver. For en svær rapport (over 10-20 sider) kan du bruke `book`-klassen (dette kurset bruker for eksempel det). Da deles dokumentet inn i kapitler, med `\chapter` kommandoen, og hvert kapittel begynner på en høyreside. Hvis du ikke vil bruke `book` kan du bruke `report` eller `article`. For den siste klassen er `\section` det øverste nivået av oppdeling i dokumentet, se [3, side 21].

Seksjoner Dokumentet deles opp i logiske deler på flere nivå. Hva som er øverste nivå bestemmes av dokumentklassen. De tilgjengelige oppdelingskommandoene er `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph` og `\subparagraph`. Hvilke av disse er tilgjengelig avhenger av dokumentklassen. Også avhengig av

dokumentklassen, er hvordan $\text{\LaTeX}$ nummererer disse forskjellige seksjonene/avsnittene, se [3, seksjon 2.2.3].

5.2 Kompilering av $\text{\LaTeX}$ dokumenter

Det finnes to veier for å kompilere $\text{\LaTeX}$ -dokumentet ditt. Enten vil du ha postscript (ps) som resultat eller så vil du ha pdf (Adobe Inc. sitt “Portable Document Format”). Skal du skrive ut på instituttets printere, anbefales det å bruke postscript. Skal du sende dokumentet videre elektronisk til andre er pdf mer standard å bruke, især for ikke-unix-brukere.

For postscript-generering så gjøres jobben i to steg, først kompilerer man latex-dokumentet til en dvi-fil via kommandoen `latex master.tex`. Etterpå kan du gjøre dvi-fila om til postscript med kommandoen `dvips master.dvi -o`.

Skal du lage pdf, er prosessen noe enklere, du skriver bare kommandoen `pdflatex master.tex`. Resultatet er fila `master.pdf` som du kan se på som vist under.

Det er likevel noen problemer man kan støte på (lager du pdf, så bytter du ut latex med pdf latex i det under), eksempelvis:

- Referanser (både til bibliografi og ligning- og kapittelreferanser) virker ikke automatisk. Hvis du bare kjører `latex master.tex` en gang, så virker ikke disse referansene. Du må kjøre `latex master.tex` en gang til (du får beskjed om det om nødvendig).
- Feilmeldinger: Første gangs $\text{\LaTeX}$ -utprøving kommer til å føre med seg mange rare feil: $\text{\LaTeX}$ har sin egen måte å spørre om feil på:

```
! Undefined control sequence.
1.21  \textt
      {\section} det øverste nivået av oppdeling i dokumentet,
?
```

og markøren står rett etter spørsmålstegnet. “Undefined control sequence” betyr at du har skrevet inn en $\text{\LaTeX}$ -kommando som $\text{\LaTeX}$ ikke vet hva betyr. Feilen her er at kommandoen heter `\texttt` og ikke `\textt`. 1.21 betyr at feilen var på linje 21 i den fila som den holdt på med for øyeblikket. Dette filnavnet finner du ved å lese baklengs i utskriften fra kompileringen til du finner et filnavn som slutter på `.tex`. Linjetallet sier ikke alltid hvor feilen ligger, den sier når latex-kompilatoren ikke lenger skjønnte hva du ville for noe. Se ellers [3, kapittel 8].

- Hvis du skal lage pdf, så må alle vektorfigurer være tilgjengelig i pdf-format før du kompilerer. Har du figurene dine i postscript (eps) kan du lage pdf av dem med kommandoen `epstopdf`, som i eksempelet her:

```
syLOW:~/prosjekt/latex/gfx:$ epstopdf figurnavn.eps
```

5.3 Visning av rapporten

Rapporten kan du se på ved å enten se direkte på `master.dvi` fila med programmet `xdvi` (start det gjerne direkte fra emacs ved hjelp av AuCTeX, seksjon 4.2). Bruker du avansert grafikk er det ikke alltid dette blir korrekt representert på skjermen med `xdvi`, men det blir kanskje likevel korrekt på utskrift.

Lager du postscript, bruker du programmet `gv` (ghostview). Postscript-fila er den du kan sende til printeren, og her er resultatet på skjerm identisk med hva du får på utskrift.

Du kan også lage PDF (Portable Document Format) av rapporten. Dette formatet er bedre for distribusjon av rapporten til folk rundt om i verden som ikke alltid klarer lese postscript-dokumenter, og pdf er også trolig i ferd med å ta over som standarformatet man kompilerer til fra latex. Skal du lage pdf, bruker du kommandoen `pdflatex` i stedet for `latex`, da får du generert pdf-fila `master.pdf` direkte, og trenger ikke gå via enda et steg som du må for postscript. Vær obs på at du må ha alle figurene tilgjengelig i pdf-format også (eller bilder i png eller jpg), som nevnt i siste punkt ovenfor. Se også de slides om grafikk i seksjon 5.5.

5.4 Utskrift av rapporten

Utskrift skal ideelt sett ikke gjøres før man er ferdig med hele greia, for å spare NTNU for utgifter til utskrift. Skal man ha midlertidige utskrifter er det en fordel å skrive ut mange sider pr. ark. Til dette kan man bruke programmet `psnup`. Skriv `'psnup -2 master.ps > master2.ps'` for å lage en ny postscript-fil `master2.ps` som du kan skrive ut på vanlig vis med `lpr`, `'lpr -Pstefan master.ps'` for å skrive til printeren ikkho for eksempel.

Husk at hvis du vil ha pdf i stedet for postscript, så kan du *ikke* bruke `lpr`-kommandoen. Utskrift gjøres fra programmet `acroread`.

Vil du ha flere sider pr. side i pdf, anbefales det å kikke på websida <http://www.math.ntnu.no/drift/hjelp/flettepdf.imf>. Med pakken `pdfpages` kan man gjøre mye kjekt med pdf-filer, som for eksempel å lime inn slides fra foredraget som i påfølgende seksjon.

5.5 Slides fra L^AT_EX-foredrag

Her inkluderes slides fra et L^AT_EX-foredrag, og de inneholder en komprimert versjon av mye av det som nevnes ellers i dette kurset.

For enda mer litteratur, se også websida som det påfølgende foredraget hører inn under,

<http://www.pvv.ntnu.no/~berland/latex/>

En introduksjon til L^AT_EX

Håvard Berland

9. november 2006

Oversikt

- 1 Introduksjon
 - L^AT_EX-filosofi
 - Å komme igang
- 2 Grunnleggende L^AT_EX
 - Et enkelt dokument
 - Kompilering
 - L^AT_EX-syntaks — Grafikk — Referanser
- 3 Avansert L^AT_EX
 - Bibliografi
 - Matematikk
 - Hjelp!
- 4 Eksempler
 - Eksempler på tilleggspakker
 - Slutt

L^AT_EX-filosofi

Det er to måter å produsere dokumenter på

- Visuelle programmer, WYSIWYG. Eksempler er Word, FrameMaker, Quark og lignende. Krever at brukeren kan layout og typografi for bra resultat.
- Strukturorienterte programmer. Brukeren "programmerer" dokumentet ved å angi dokumentets struktur, ikke dets layout. L^AT_EX faller i denne kategorien. HTML er et annet kjent eksempel.

L^AT_EX-filosofi

I mer detalj:

- Hvert dokument *programmeres*. En kildefil med kommandoer/makroer kompiles (av programmet pdf_latex) til et presentasjonsformat (pdf). Endringene blir ikke synlige før du recompilerer.
- Du forteller ikke L^AT_EX hvordan en overskrift skal se ut, du forteller den at det er en overskrift.
- Alt dette høyner absolutt brukerskelen, og dere går derfor på dette kurset.
- Høyere brukerskel gir enorme muligheter. Det finnes i praksis ingen grenser for hva du kan få til (men det finnes grenser for mye tid du kan bruke på det).

Hvorfor L^AT_EX

For L^AT_EX:

- Profesjonell layout, *vanskelig å ikke få til god layout*.
- Enkelt å forholde seg til store dokumenter.
- Enklere å skrive formler (!)
- Kan skrive makroer for ofte brukte konstruksjoner.
- Det finnes utallige pakker for å gjøre akkurat det du vil.
- Utseende på matematikk er i en annen klasse enn alt annet.

For Microsoft Word:

- Enklere å komme i gang.
- Du ser hele tiden hvordan det ser ut.
- Integrert system.

Historie

- T_EX ble utviklet av Donald Knuth på 70-tallet siden han ikke hadde noen gode nok alternativ til å typesette sine egne bøker.



- L^AT_EX ble utgitt av Leslie Lamport i 1984, med makroer som gjør det enklere for brukeren å lage dokumenter.
- Knuth gir \$328,67 til den neste som finner en bug i T_EX.

Installering

Linux/Unix Latex-distribusjonen te_LTeX er mest utbredt og anbefales. Den er installert på alle unix-maskiner på NTNU, og følger også med alle Linux-distribusjoner.

Windows Distribusjonen MiK_TTeX anbefales (Pc_TTeX anbefales *ikke*, selv om NTNU har betalt for lisens). Hent ned gratis fra <http://www.miktex.org>.

Mac Installer TeXShop (gratis), <http://www.uoregon.edu/~koch/texshop/>
Hvis du kommer fra unix, vil du kanskje bruke te_LTeX som kan installeres med "fink" (<http://fink.sf.net>).

Editor

L^AT_EX-koden skriver man i en egen tekstfil (ASCII-format), med en egnet editor.

Emacs Finnes både for Unix, Windows og Mac, og er gratis. Installer også pakken AuC_TTeX for å integrere Emacs og L^AT_EX sammen. Installert på alle Unix-maskiner ved NTNU. Ref_TTeX er også verdt å kikke på.

TeXnicCenter For Windows, gratis og anbefales. <http://www.texniccenter.org>

TextPad For Windows, gratis for studenter og ansatte ved NTNU. Har også modul for L^AT_EX, <http://www.textpad.com>.

TeXShop For Mac.

Et enkelt dokument

```
\documentclass[a4paper, norsk, 11pt]{article}
\usepackage[T1]{fontenc}      % Nødvendig for fonter.
\usepackage[latin1]{inputenc} % Nødvendig for æså
\usepackage{babel}           % Fornorsker dokumentet
\usepackage{graphicx}        % For å kunne inkludere grafikk

\author{Håvard Berland}
\title{Et enkelt dokument}
\date{\today}

\begin{document}
\maketitle

Her er dokumentteksten.
\end{document}
```

- \documentclass angir her at dokumentklassen skal være "article". For større arbeider kan "book" brukes (den har kapitler).

Et enkelt dokument

```
\documentclass[a4paper, norsk, 11pt]{article}
\usepackage[T1]{fontenc}      % Nødvendig for fonter.
\usepackage[latin1]{inputenc} % Nødvendig for æså
\usepackage{babel}           % Fornorsker dokumentet
\usepackage{graphicx}        % For å kunne inkludere grafikk

\author{Håvard Berland}
\title{Et enkelt dokument}
\date{\today}

\begin{document}
\maketitle

Her er dokumentteksten.
\end{document}
```

- Hakeparenteser [] angir opsjoner. 11pt er skriftstørrelsen, 10 og 12 er også mulige.

Et enkelt dokument

```
\documentclass[a4paper, norsk, 11pt]{article}
\usepackage[T1]{fontenc}      % Nødvendig for fonter.
\usepackage[latin1]{inputenc} % Nødvendig for æså
\usepackage{babel}           % Fornorsker dokumentet
\usepackage{graphicx}        % For å kunne inkludere grafikk

\author{Håvard Berland}
\title{Et enkelt dokument}
\date{\today}

\begin{document}
\maketitle

Her er dokumentteksten.
\end{document}
```

- Husk at graphicx har en x til slutt!

Et enkelt dokument

```
\documentclass[a4paper, norsk, 11pt]{article}
\usepackage[T1]{fontenc}      % Nødvendig for fonter.
\usepackage[latin1]{inputenc} % Nødvendig for æså
\usepackage{babel}           % Fornorsker dokumentet
\usepackage{graphicx}        % For å kunne inkludere grafikk

\author{Håvard Berland}
\title{Et enkelt dokument}
\date{\today}

\begin{document}
\maketitle

Her er dokumentteksten.
\end{document}
```

- Det som kommer før \begin{document} er bare innstillinger vi gjør, og kalles *preamble*. Ingenting skrives ut.

Et enkelt dokument

```
\documentclass[a4paper, norsk, 11pt]{article}
\usepackage[T1]{fontenc}      % Nødvendig for fonter.
\usepackage[latin1]{inputenc} % Nødvendig for æpå
\usepackage{babel}            % Fornorsker dokumentet
\usepackage{graphicx}         % For å kunne inkludere grafikk

\author{Håvard Berland}
\title{Et enkelt dokument}
\date{\today}

\begin{document}
\maketitle

Her er dokumentteksten.
\end{document}
```

- `\begin` og `\end` definerer et *miljø*. Dette brukes mye i L^AT_EX. Miljøer er nøstede, og `document` er alltid det ytterste/øverste miljøet.

Struktur i dokumentet

```
\section{Introduksjon}

Intro bla bla...

\section{Analyse}
\subsection{Datamateriale}
\subsection{Testprosedyrer}

\section{Konklusjon}

Dette konkluderer rapporten.
```

- Sett en stjerne bak "section" for å unngå å få tall på overskriftene (da kommer den heller ikke i innholdslista).
- Sett inn `\tableofcontents` der du vil ha innholdsliste.

Eksempeldokumentet

Et enkelt dokument

Håvard Berland
16. november 2004

Innhold	
1 Introduksjon	1
2 Analyse	1
2.1 Datamateriale	1
2.2 Testprosedyrer	1
3 Konklusjon	1
1 Introduksjon	
Intro bla bla...	
2 Analyse	
2.1 Datamateriale	
2.2 Testprosedyrer	
3 Konklusjon	
Dette konkluderer rapporten.	

Store dokumenter

For "store" dokumenter (mer enn 5-10 sider) kan det være fordelaktig å skille ut dokumentet i flere deler. Da har du ei hovedfil som inkluderer de andre filene.

```
\documentclass[a4paper, norsk, 11pt]{article}
...
\begin{document}
\input{introduksjon}
\input{resultater}
\input{konklusjon}
\end{document}
```

- Du får ryddigere hovedfil, og ryddigere underfiler.
- Kall filene dine for *filnavn.tex*, men endelsen *.tex* trenger du ikke ha med i `\input`.
- Lett å ta bort deler av dokumentet for raskere kompilering ("*%*" er kommentartegnet).

Kompilering (PDF)

- Kjør kommandoen "*pdflatex hovedfil.tex*"

```
hovedfil.tex --pdflatex--> hovedfil.pdf
```

(eller trykk på en pdf-knapp i editoren din hvis du har det)

- Hver eneste lille syntaks-feil du har gjort, gjør kompileringen mislykket. Kompiler derfor ofte slik at du vet hvor du kan ha gjort feil.
- Ved feil, så gir `pdflatex`-programmet deg en feilmelding og et linjenummer feilen oppstod på.
- Manglende eller for mange krøllparenteser `{` eller `}` er ofte en kilde til feil det kan være vanskelig å finne.
- Se på resultatfila (*hovedfil.pdf*) med *acread*.

Kompilering (PostScript)

For få år siden kompilerte man alltid i to omganger, først til et filformat kalt *dvi*, og så til *postscript*.

```
hovedfil.tex --latex--> hovedfil.dvi --dvips--> hovedfil.ps
```

Dette kan gjøres med kommandoene

```
> latex hovedfil.tex
> dvips hovedfil -o
```

- På linux/unix bruker man *gv* for å se på *postscript*-filer, på Windows bruker man *gsview*
<http://www.cs.wisc.edu/~ghost/gsview/> (gratis), husk å også installere Ghostscript derfra.
- Bruk PDF med mindre du finner en god grunn til å bruke *postscript*!

Tekst i dokumentet

- Pass på de 10 spesialtegnene

\$ % & { } _ ^ \ ~

Foran de 8 første, kan du sette inn en \ foran hvis du vil ha dem med.

- Lister

```
\begin{itemize}
  \item foo
  \item bar
\end{itemize}
```

• foo

• bar

- Bytt ut itemize med enumerate hvis du vil ha tall.
- Nøsting av opplistinger går også fint.

Tekst i dokumentet forts.

- Definisjonslister

```
\begin{description}
  \item[NTNU]{Et universitet}
  \item[UiO]{Et annet universitet}
\end{description}
```

NTNU Et
universitet
UiO Et annet
universitet

- Aksenter

```
é \'{e} \'{o} \'{o}
ö \"{o} \"{o} \"{o}
```

é é ò ö
ö ö ø ö

- Tankestrek og bindestrek. Pass på forskjellen. Bindestrek skrives - og tankestrek som --. På engelsk brukes også av og til lang tankestrek, som skrives ---. Bruk kort tankestrek mellom to etternavn, "Euler-Lagrange" og i tallintervall.

Tekst i dokumentet forts.

- Tekstmarkeringer

```
\emph{Uthevet}
\textbf{Fet skrift}
\textsc{Kapit'eler}
\texttt{Skrivemaskin}
\textit{Kursiv (italics)}
\textsl{Skråstilt (slanted)}
```

*Uthevet***Fet skrift**

KAPITÉLER

Skrivemaskin

*Kursiv (italics)**Skråstilt (slanted)*

- Fotnoter¹ lager vi slik

```
Fotnoter\footnote{Dette er en fotnote} lager vi slik
```

¹Dette er en fotnote

Bytte skrifttype

- Hvis du vil bytte ut standardskrifttypen (Computer Modern Roman) med noe annet har man to trygge valg.

```
\usepackage{mathptmx}
\usepackage[scale=0.9]{helvet}
\usepackage{courier}
```

```
\usepackage{mathpazo}
\usepackage[scale=0.95]{helvet}
\usepackage{courier}
```

- Flere valg kan være tilgjengelige for akkurat din installasjon, du kan forsøke \usepackage{fourier} eller \usepackage{fourierx} for Fourier- og Utopia-fontene.
- Prøv gjerne også å bytte dokumentklasse fra article til scrartcl, en del av KOMAscript-pakken og se om du foretrekker det.

Tall med enheter

- Pakken *SIunits* er en kjekk pakke for å hjelpe deg å skrive tall med enheter korrekt (\usepackage{SIunits}).
- Fallgruber ved enheter kan være korrekt mellomrom (som heller ikke kan linjebrekkes), stående skrifttype på selve enheten og rett kapitalisering på enheten.
- Eksempler

```
\unit{10}{\kilogram}
\unit{80}{\degreecelsius}
\unit{13}{\metre \per \second}
\unit{23,4}{\metre\squared}
\unit{800}{\micro \metre}
```

10 kg

80 °C

13 m/s

23,4 m²

800 µm

Tabeller

Tabeller setter man i *tabular*-miljøet. *lcr* angir tre kolonner i tabellen, en venstrejustert kolonne, en sentrert, og en høyrejustert. Kolonner skilles med & og neste linje får man ved \\.

Tips: Vær nøye med å ha rett antall og riktig sted på & og \\.

```
\begin{tabular}{lcr}
  \emph{Navn} & & \emph{Alder} \\
  & & & \emph{Høyde} \\
  Ole & 9 & 149 cm \\
  Mari & 21 & 168 cm \\
\end{tabular}
```

Navn	Alder	Høyde
Ole	9	149 cm
Mari	21	168 cm

Grafikkformat — Konverteringsveier

Grafikk-kilde	latex+dvips	pdflatex
Innscannet tegning, skjermbilde (PrintScreen), Photoshop o.l.	* $\mapsto$ EPS*	* $\mapsto$ PNG
Digitalkamera, innscannet foto	JPG $\mapsto$ EPS*	* $\mapsto$ JPG
Excel-grafikk	WMF $\mapsto$ EPS	WMF $\mapsto$ EPS $\mapsto$ PDF
Coreldraw, AutoCAD, Xfig	* $\mapsto$ EPS	* $\mapsto$ EPS $\mapsto$ PDF

- EPS* betyr et bitmapbilde som er pakket inn i en eps-fil. Det er like fullt et bitmapbilde fortsatt.

Grafikk i dokumentet

```
\includegraphics[width=0.7\textwidth]{bildefil}
```

- Filendelsen tar vi ikke med. L^AT_EX plukker det formatet den liker best, men du må sørge for at det er tilgjengelig (png, jpg eller pdf for pdflatex, eps hvis du bruker latex+dvips).
- Bredden (eller høyden) kan du spesifisere som opsjon, enten i centimeter/millimeter, eller som en andel av tekstbredden (som her).
- Figurer vil man typisk også ha flytende:

```
\begin{figure}[htbp]
\centering
\includegraphics[height=3cm]{bildefil}
\caption{Et eksempelbilde}
\label{fig:eksempelbilde}
\end{figure}
```

Referere til tabeller og figurer

Til de flytende tabellene og figurene har vi lagt til en "label". Denne kan vi referere til:

```
Vi ser fra tabell~\ref{tab:alderhoyde} og
fra figur~\ref{fig:eksempelbilde} at ...
```

Vi ser fra tabell 2 og fra figur 3 at ...

- ~ betyr et mellomrom der L^AT_EX ikke får lov til å brette linja.
- Prefiksene tab: og fig: er ikke noe krav, bare lurt for å holde orden.
- Hvis du bruker Emacs, kan RefTeX-pakken holde styr på alle referansene dine. Hvis ikke, kan pakken `showkeys` være verdt et forsøk, `\usepackage{showkeys}` i preamble.

Referere til kapitler/seksjoner

Kapitler og seksjoner kan vi også referere til (kapitler kun for dokumentklassen *book*).

```
\chapter{Introduksjon}
\label{chap:introduksjon}
...
\section{Historie}
\label{sec:historie}
...
Historien til prosjektet (se seksjon~\ref{sec:historie})
viser at...
```

Historien til prosjektet (se seksjon 1.1) viser at...

- For at alle referanser skal være korrekte, må du ha kompilert to ganger siden sist du endret eller la til en \label.

Topp og bunn på sidene

"Header" og "footer" på sidene vil få utseende bestemt av kommandoen `\pagestyle{stil}` i preamble. Denne har tre predefinerte stiler:

`plain` (standard) Sidetall sentrert nederst.

`headings` Kapittelnavn og sidetall øverst.

`empty` Ingenting.

Stilen for nåværende side kan endres med `\thispagestyle{stil}`.

- For mer avanserte topp- og bunntekster på sidene dine, bruker du pakken `fancyhdr`.
- Hvis du skal skrive ut dobbelsidig, bruk `twoside` som opsjon til `\documentclass`. Dette vil justere margene og `headings`-stilen.

Småtips

- Bruk pakken `parskip` for å få en tom linje som avsnittskiller i ditt ferdige dokument. Dvs. legg til `\usepackage{parskip}` i preamble.
- For tvungne linjeskift, bruk `\\`, bruk med omhu!
- For to kolonner på hver side, bruk `twocolumn` som opsjon til `\documentclass`.
- L^AT_EX deler ord med bindestrek for å lage pene avsnitt og rett høyremarg. Hvis et ord deles som du ikke vil dele, sett det inn som i `\mbox{ikkedelmeg}`.
- For små figurer/bilder ved siden av teksten, se på pakkene `floatflt` eller `wrapfig`.

Definere egne kommandoer

- For ofte brukte kommandosekvenser, kan du lage dine egne kommandoer/makroer. F.eks.

```
\newcommand{\integral}{\int_{\mathbf{R}}
f(x) \, \mathrm{d}x}
```

- Kommandoer kan også ta inn argument

```
\newcommand{\integral}[1]{\ensuremath{
\int_{\mathbf{R}} \#1 (x) \, \mathrm{d}x}}
```

og vi kan etterpå skrive `\integral{g}` og få $\int_{\mathbf{R}} g(x) dx$.
`\ensuremath` er brukt for å kunne bruke kommandoen både i mattemodus og ellers.

- For å bytte ut betydningen av en eksisterende kommando, bruk `\renewcommand`.
- Ikke bruk `\newcommand` til å forkorte eksisterende kommandoer! Tenk på kodens lesbarhet.

BiB_TE_X– Litteraturliste

I en egen fil legger du inn alle bøker og artikler du vil referere til. Legg følgende til *bibliografi.bib* f.eks.

```
@Book{lamp94,
author = {Leslie Lamport},
title = {LaTeX, User's Guide and
{Reference Manual}},
publisher = {Addison-Wesley},
year = {1994},
}

@Book{goossens94,
author = {Michel Goossens and Frank Mittelbach
and Alexander Samarin},
title = {The LaTeX Companion},
publisher = {Addison-Wesley},
year = {1994},
}
```

BiB_TE_X– Litteraturliste forts.

```
@Article{knuth94,
author = {Donald E. Knuth},
title = {The Sandwich Theorem},
journal = {The Electronic Journal of Combinatorics},
year = {1994},
number = {1},
}
```

- Hvert innslag i bib-fila har en unik id, her "lamp94", "goossens94" og "knuth94".
- For å tvinge bokstaver til å være store, putt dem inn i krøllparentes.
- Andre datafelt skal/kan settes for andre typer referanser.

BiB_TE_X– Litteraturliste forts.

I dokumentet ditt refererer du til bøkene/artiklene slik

```
I henhold til \cite{goossens94} og
\cite[Kapittel 3]{lamp94} har vi ...
```

I henhold til [1] og [2, Kapittel 3] har vi ...

I tillegg må du i *hovedfil.tex* legge inn der du vil ha bibliografien

```
\bibliography{bibliografi} % hvis fila heter bibliografi.bib
\bibliographystyle{plain} % her finnes mange alternativ.
```

Korrekt kompilering er noe innfløkt, men følgende prosedyre er tilstrekkelig (men ikke alltid minimal)

```
> pdflatex hovedfil.tex
> bibtex hovedfil
> pdflatex hovedfil.tex
> pdflatex hovedfil.tex
```

Matematikk

Tre pakker bør alltid være med i preamble for å skrive matematikk:

```
\usepackage{amsmath,amssymb,amsthm}
```

- Matematikk inni løpende tekst skal omslutes av $\$$ vi setter $\$v = 2^n$ vi setter $v = 2^n$
- Det finnes egne kommandoer i mattemodus
 α β $\sum$ $\int$
`\alpha` `\beta` `\sum` `\int`
- Sub- og superskrift
 x_i x_{i^j} a^{x+y} x_i x_i^j a^{x+y}

Matematikk — ligninger

- Ligninger setter vi i et eget miljø:

```
\begin{equation}
f(x+h) = \sum_{i=0}^{\infty} \frac{h^k}{k!} f^{(k)}(x)
\label{eq:taylor}
\end{equation}
```

$$f(x+h) = \sum_{i=0}^{\infty} \frac{h^k}{k!} f^{(k)}(x) \quad (1)$$

- Bytt ut `equation` med `equation*` for å ta bort nummerering.
- Tomme linjer inni mattemiljø er ulovlig. Sett da et kommentartegn på begynnelsen av linja.
- For å referere til ligninger, bruk `\eqref{}` i stedet for `\ref{}`.

Matematikk — ligninger

For flerlinjers utregninger, bruker vi *split* inni *equation*:

```
\begin{equation}
\label{eq:tolinjers}
\begin{split}
y &= a + b \\
&\quad c + d
\end{split}
\end{equation}
```

$$y = a + b + c + d \quad (2)$$

Flere ligninger justert etter $=$ -tegnet:

```
\begin{align}
\label{eq:f-en}
f_1(x) &= a_1 x + a_2 x^2 \\
\label{eq:f-to}
f_2(x) &= b_1 x + b_2 x^2
\end{align}
```

$$f_1(x) = a_1 x + a_2 x^2 \quad (3)$$

$$f_2(x) = b_1 x + b_2 x^2 \quad (4)$$

NB: Aldri bruk *eqnarray*/*eqalign* (foreldet L^AT_EX)!

Matematikk — eksempler

```
\begin{equation*}
P(x) =
\begin{cases}
0 & \text{hvis } x \leq 0 \\
x^2 & \text{hvis } x > 0
\end{cases}
\end{equation*}
```

$$P(x) = \begin{cases} 0 & \text{hvis } x \leq 0 \\ x^2 & \text{hvis } x > 0 \end{cases}$$

```
\begin{equation*}
\frac{1}{\sqrt{2\pi}} \int_0^h \sqrt{\frac{\omega}{1+x}} e^{i\omega x} d\omega
\end{equation*}
```

$$\frac{1}{\sqrt{2\pi}} \int_0^h \sqrt{\frac{\omega}{1+x}} e^{i\omega x} d\omega$$

Matematikk — eksempler

```
\begin{equation*}
\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} x =
\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}
\end{equation*}
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} x = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

```
\begin{equation*}
\lim_{x \rightarrow 0} \frac{\sin(x)}{x}
\end{equation*}
```

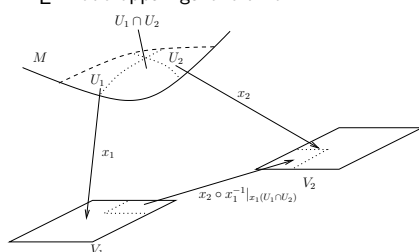
$$\lim_{x \rightarrow 0} \frac{\sin(x)}{x}$$

Hvordan få hjelp

- 1 Dette kursets hjemmeside:
<http://www.pvv.ntnu.no/~berland/latex/>
- 2 **Google!** Lim inn feilmeldinga du får fra L^AT_EX i søkefeltet, søk på kommandoer (ta med latex også som søkeord) eller søk etter dokumentasjon.
- 3 Husk også Google Groups, den søker på news.
- 4 Les dokumentasjon om L^AT_EX og tilleggspakker du finner på nett. Prøv også: > texdoc pakkenavn.
- 5 Kjøp og les boka "Leslie Lamport: L^AT_EX User's guide and reference manual".
- 6 Kjøp og les boka "Goossens, Mittelbach, Samarin: The L^AT_EX companion", andre utgave.
- 7 Spør folk du kjenner om hjelp.
- 8 Spør på news, no.it.programvare.tex.

 L^AT_EX inni figurer

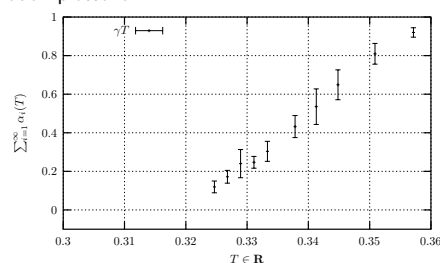
- Med tegneprogrammet *xfig* kan du tegne figurer og enkelt legge til L^AT_EX-kode oppå figurene dine.



Søk på google etter: "figtexp2eps"

 Plott med L^AT_EX-kode

- Hvis du plottet dine talldata med Gnuplot, kan du innlemme L^AT_EX-kode i plottene



Søk på google etter "gptexp2eps". For Matlab-grafikk, så kan det samme oppnås med matlab-pakken "laprint".

Skrive øvinger

- Øvinger er det ved NTNU laget en egen dokumentklasse for

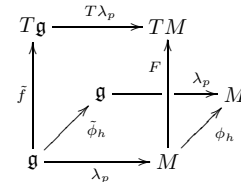


- ☐ Giv Dracula suger blod av en kropp med en sekunderate lik 0.01 av blodmengden.
 a) Hvis personen hadde 30 l blod i seg, hvor lang tid tar det før blodmengden er halvert?
 b) Skriv oppgave
☐ Flere oppgaver...

Se <http://www.math.ntnu.no/drift/hjelp/texoving.imf>

Kommutative diagram

- Kommutative diagram kan du lage med pakken *xy*



Se dokumentet "xyguide" på kursets hjemmeside.

Presentasjoner

- Presentasjoner lager du med pakken *beamer*.



Last ned fra <http://latex-beamer.sourceforge.net/>

Grafikk

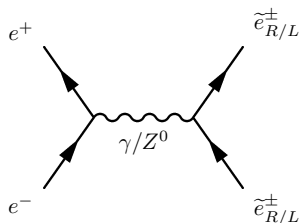
- Grafikk kan også programmeres med pakken *tikz*, spesielt nyttig sammen med *beamer*.



- <http://sourceforge.net/projects/pgf/>

Feynman-diagram

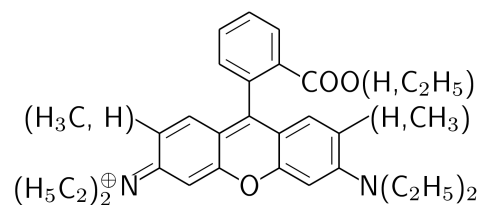
- Feynman-diagram kan lages med *feynmf*-pakken



Søk på google etter "latex feynman".

Kjemiformler

- Kjemiformler kan lages med *ochem*-pakken



Søk på google etter "latex ochem"

Musikk

- Noter i musikk kan typesettes med *MusicTeX*

D Bm
This is the manual song

A⁷ D
No-one really knows what's goin' on

D F#m
This is the manual song

A⁷ D
And now the chorus is already gone



Win - ter a - de,
schei - den tut weh. a - ber dein Schei - den - macht,

D A⁷ D
The second verse is like the first,
 A⁷ D
The music poor, the verse is worse,
 A⁷ D
I wrote this since I'd get sued,
 A⁷ D
If I used real songs. This'll do.

Søk på google etter "latex musictex"

Slutt

$$S_{LU}^T T$$

Og: *Lykke til!*

Takk til (alfabetisk) Kristian Gjøsteen, Hugo Hammer, Harald Hanche-Olsen, Per Kristian Hove, Steinar Kragset, Kjetil Midthun, Marte Thuen

Kapittel 6

Mal for L^AT_EX-filene

Dette kapitlet er det meningen at man skal koke, for så å forstå hva man har kokt, og så gjøre de endringer man finner ut at man skal.

I dette kurset anbefales det at man deler opp hele rapporten i mange små filer. En fil for hver del/kapittel anbefales. Da har man en hovedfil som man kjører `latex` eller `pdflatex` på, men som inkluderer alle de andre filene.

6.1 Oppdeling av rapporten

En fornuftig oppdeling av L^AT_EX-filene (de som slutter på `'.tex'`) som hører til rapporten kan være i følgende deler. Dette avhenger selvsagt om du har med disse delene i rapporten, og du finner selv ut hvilke modifikasjoner du vil gjøre. Under dette finnes eksempler på disse anbefalte filene.

`master.tex` Hovedfila for rapporten din. Det er denne du skal compilere med `latex`-kommandoen.

`forside.tex` En forside for rapporten din. Denne kan gjøres svært enkel, så enkel at man kanskje kan putte den inn i `master.tex`.

`innledning.tex` Innledning for prosjektet.

`bakgrunn.tex` En del om bakgrunnsteori. Filnavnene fra nå av er det ingen spesiell vits å følge slavisk, da det bør tilpasses prosjektet. Det er tanken om logisk oppdeling som teller.

`resultater.tex` Resultater/konklusjon for prosjektet.

`appendiks.tex` Hvis du har det.

Alle filene som er gjengitt under er tilgjengelig på web som <http://www.stud.math.ntnu.no/kurs/maler/<filnavn>> der `<filnavn>` byttes ut med den fila du vil ha tak i.

6.1.1 Fileksempel: master.tex

Denne fila er tilgjengelig på web som master.tex

```

1 % Først spesifiserer vi hvilken dokumentklasse vi vil ha og noen
  % globale opsjoner. Bytt ut 'article' med 'book' hvis du vil ha
  % med kapitler.
  \documentclass[a4paper, twoside, titlepage, 11pt]{article}

6 % Så sier vi fra om hvilke tilleggspakker vi trenger
  % til dokumentet vårt. De som du ikke trenger (se kommentaren)
  % kan det være en fordel å kommentere ut (sett prosenttegn foran),
  % da vil kompilering gå raskere.

11 \usepackage[norsk]{babel}           % norske navn rundt omkring
    \usepackage[T1]{fontenc}          % norsk tegnsett (æøå)
    \usepackage[latin1]{inputenc}     % norsk tegnsett
    \usepackage{geometry}             % anbefalt pakke for å styre marger.

16 \usepackage{amsmath,amsfonts,amssymb} % matematikksymboler
    \usepackage{amsthm}               % for å lage teoremer og lignende.
    \usepackage{graphicx}             % inkludering av grafikk
    \usepackage{subfig}               % hvis du vil kunne ha flere
                                      % figurer inni en figur

21 \usepackage{listings}              % Fin for inkludering av kildekode

    %\usepackage{hyperref}             % Lager hyperlinker i evt. pdf-dokument
                                      % men har noen bugs, så den er kommentert
                                      % bort her.

26
  % Indeksgenerering er kommentert ut her. Ta bort prosenttegnene
  % hvis du vil ha en indeks:
  %\usepackage{makeidx}
  %\makeindex

31
  % Selve dokumentet begynner:

  \begin{document}

36 % På forsida skal vi ikke ha noen sidenummerering:

    \pagestyle{empty}
    \pagenumbering{roman}

```

```
41 % Inkluder forsida:
    \input{forside}

    % Romerske tall på alt før selve rapporten starter er pent.
    \pagenumbering{roman}

46 % For å ikke begynne innholdslista på baksida av forsida:
    \cleardoublepage
    % (kun aktuelt når man har twoside som global opsjon)

51 % Nå vi vil ha noe i topp- og bunnteksten
    \pagestyle{headings}

    % Si til LaTeX at vi vil ha ei innholdsliste generert akkurat her:
    \tableofcontents

56 % Pass på at neste side ikke begynner på baksida av en annen side.
    \cleardoublepage

    % Arabisk (vanlige tall) sidenummerering. Starter på side 1 igjen.
61 \pagenumbering{arabic}

    % Inkluder alle de andre kildefilene:

    % NB: Vi trenger ikke ta med filendelsen .tex her. Den vet
66 % LaTeX om selv!

    \input{innledning}

    \input{bakgrunn}

71 \input{resultater}

    % Bibliografi/referanseliste skal komme før appendiks
    \bibliography{kurs}
76 \bibliographystyle{plain}

    % En latex-kommando for å si fra at kapitlene/seksjonene fra nå
    % av skal nummereres med store bokstaver:
    \appendix

81 \input{appendiks}
```

```

% Indeks for rapporten. Ta bort prosenttegn hvis du vil ha det med.
%\printindex

86 % Avslutter dokumentet vårt:
\end{document}

% Local Variables:
91 % TeX-master: "master"
% End:

```

6.1.2 Fileksempel: forside.tex

Det gis her to eksempler. Denne første er en “vanlig” latex-generert forside, tilgjengelig som forside.tex på web.

```

1 % Enkel forside som bruker latex sin \titlepage kommando:
% NB: Bruken av \and mellom navn!
3 \titlepage
\title{Lebesgueintegrering}
\author{Leonhard Euler Jr. \and Karl Fr. Gauss Jr.}
\date{2. mai 2002}
\maketitle

8 % Local Variables:
% TeX-master: "master"
% End:

```

Denne andre er en mer fancy forside, tilgjengelig fra forside2.tex

```

1 % En mer fancy forside. Ikke ta mål av deg å forstå
% alle disse kommandoene på første forsøk.

4 \newcommand{\HRule}{\rule{\linewidth}{1mm}}

\vspace*{\stretch{1}}
\noindent\HRule
\begin{center}
9 \huge
\noindent Lebesgueintegraler \[7mm]
\large
\noindent\emph{Leonhard Euler Jr.}
\end{center}
14 \noindent\HRule
\vspace*{\stretch{2}}
\begin{center}

```

```
\Large\textsc{Institutt for matematiske fag --- NTNU 2005}
\end{center}
```

19

```
% Local Variables:
% TeX-master: "master"
% End:
```

6.1.3 Fileksempel: bakgrunn.tex

Dette er den siste eksempelfila, tilgjengelig på web som bakgrunn.tex. De andre filene finner du selv av hvordan skal være. Bruk cp-kommandoen (se seksjon 3.1) for å lage kopier av filene eller lag nye ved hjelp av emacs.

```
1 % Dette eksempelet er laget for article-dokumentklassen. Hvis
  % skriver i 'book'-dokumentklassen vil du kanskje bytte ut
3 % \section med \chapter, \subsection med \section, osv...
```

```
\section{Bakgrunnsteori}
```

```
bla.bla bla
```

8

```
\subsection{Integralets opprinnelse}
```

```
bla bla bla
```

```
13 \subsection{Riemann-integralet}
```

```
bla bla bla
```

```
18 % Local Variables:
  % TeX-master: "master"
  % End:
```


Kapittel 7

Matematikk i $\text{\LaTeX}$

7.1 Hvordan lære seg å skrive matematikk

Matematikk i $\text{\LaTeX}$ skriver man med kommandoer, ikke med å peke og klikke med musa inni endeløse menyer (jf. Word). Kommandoene må man lære seg, og det er derfor en liten kneik du kommer over før det går fort å skrive din matematikk i $\text{\LaTeX}$. Men etter din første rapport på noen sider, er du over kneika og kan surfe videre gjennom all tilgjengelig $\text{\LaTeX}$ -litteratur.

Matematikk i $\text{\LaTeX}$ skriver man enten midt inni tekst eller i egne matematikk-miljøer. Da kan man bruke dollartegnet til å skru på og skru av mattemodus (hvori man skriver sin $\text{\LaTeX}$ -matematikk). Mer matematikk setter man opp i egne ligninger, med eller uten nummerering.

7.2 Eksempler

Noen matematikk-eksempler til å lære av:

$x^2 = a$	(7.1)	$\begin{array}{l} \backslash\text{begin}\{\text{equation}\} \\ \text{x}^2 = \text{a} \\ \backslash\text{end}\{\text{equation}\} \end{array}$
-----------	-------	--

$f(z) = \sum_{i=0}^{\infty} a_i z^i$	(7.2)	$\begin{array}{l} \backslash\text{begin}\{\text{equation}\} \\ \text{f(z)} = \backslash\text{sum}_{\{i=0\}}^{\backslash\text{infty}} \text{a_i z}^i \\ \backslash\text{end}\{\text{equation}\} \end{array}$
--------------------------------------	-------	---

$\phi(t) = \frac{1}{\sqrt{2\pi}} \int_0^t e^{-x^2/2} dx \quad (7.3)$	<pre>\begin{equation} \phi(t) = \frac{1}{\sqrt{2\pi}} \int_0^t e^{-x^2/2} \, \mathrm{d}x \end{equation}</pre>
--	---

For videre matematisk typesetting anbefales [2] og [4]. Ikke legg for mye vekt på det som står i [3]

7.3 Ligningsreferanser

Ligningene over fikk et nummer i høyremargen. Dette nummeret vet du ikke når du skriver dokumentet ditt, men blir satt under kompilering. Du må derfor sette navn på ligningene som du vil referere til og referere til dette navnet i kildekoden din. Navnet du setter på ligningene kommer aldri fram i det ferdige dokumentet.

For å referere til en av ligningene over gjør vi eksempelet på nytt:

$$\phi(t) = \frac{1}{\sqrt{2\pi}} \int_0^t e^{-x^2/2} dx \quad (7.4)$$

```
\begin{equation}
\phi(t) = \frac{1}{\sqrt{2\pi}} \int_0^t e^{-x^2/2} \, \mathrm{d}x
\label{eq:phi}
\end{equation}
```

Det er vanlig konvensjon å starte alle navn på ligninger med eq:, da slipper du problem hvis du kaller en tabell det samme navnet som en ligning (bruk f.eks. prefikset tbl: for tabeller).

For å referere til denne ligningen som her, “Se ligning (7.4).” kan man skrive

Se ligning \eqref{eq:phi}.

\eqref gjør at du får paranteser rundt ligningen som er vanlig for referanser til ligninger. Hvis du ikke vil ha parantesene bruker du bare \ref.

7.4 Noen huskereglar

Her presenteres regler som ikke alle vet, men som gjør rapporten din bedre. Ikke alle proffene gjør alt dette riktig engang.

7.4.1 Husk mattemodus i løpende tekst

Alle matematiske symbol skal skrives i kursiv. Du skal egentlig ikke tenke på at det skal stå i kursiv, men at du skal si i fra til LaTeX om at det du skriver faktisk er et matematisk

symbol, og ikke bare en vanlig bokstav (det som kalles logisk formatering i motsetning til visuell formatering).

Eksempel på gal formatering:

Rekursjonen foretas inn til nivå n . Da vil nodene V være avdekket.

Hvordan det egentlig skal gjøres:

Rekursjonen foretas inn til nivå n . Da vil nodene V være avdekket.

Trikset er å sette inn ett dollartegn før og ett etter 'n' og 'V', slik at det i kildekoden står `$n$`. Dette må man følge overalt i dokumentet, inkludert overskrifter og forsider.

Dette kan virke utrolig pirkete, men det blir lagt merke til av øyne som har lest mye matematikk, også dine etterhvert.

7.4.2 Derivert

Typesetting av derivert er det få som gjør riktig. Differensialet

$$\frac{dy}{dx}$$

er satt feil i ligningen ovenfor (med kildekode `\frac{dy}{dx}`). 'd'-bokstavene skal faktisk ikke settes i kursiv, men i "upright". Riktig skal være

$$\frac{dy}{dx}$$

med kildekode (for eksempel) `\frac{\mathrm{d}y}{\mathrm{d}x}`. `\mathrm` står får "Math Roman"-fonten (den vanlige mattefonten i kursiv er "Math Italic"). Det finnes nok flere måter å gjøre dette i kildekode, og siden det tar lengre tid å skrive typografisk riktig enn feil, så vil man typisk løse dette ved å definere egne kommandoer, se Seksjon 7.5 under.

Ikke alle lærebøker følger denne standarden for å sette differensialer, men det er penere å sette dem på denne måten. Springer har for eksempel satt som krav for bøkene de gir ut at differensialet d , $e = 2.718\dots$ og $i = \sqrt{-1}$ skal settes som "upright", dvs. med bruk av fonten Math Roman. Generelt er amerikanske bøker mer slepphendte med slike typografiske detaljer enn europeiske.

7.4.3 Integral

Integraler settes ofte feil på grunn av samme detalj som over, at man glemmer å sette 'd'-en i differensialet til slutt i Math Roman. Men man skal i tillegg ha et lite mellomrom før differensialet. Disse to feilene er illustrert i den følgende:

$$\int_{-\infty}^{\infty} f(x)e^{-i\omega t}dx$$

mens riktig layout er

$$\int_{-\infty}^{\infty} f(x) e^{-i\omega t} dx$$

hvor det har blitt lagt til et lite mellomrom før differensialet og d'en har fått riktig form. Legg også merke til at eksponensialfunksjonen har fått byttet formen sin til Math Roman. Fontvalg for eksponensialfunksjonen er ikke like viktig, da begge fontvalg kan argumenteres for. Det samme gjelder for imaginær enhet i.

Gal kildekode:

```
\int_{-\infty}^{\infty} f(x) e^{-i \omega t} dx
```

Riktig:

```
\int_{-\infty}^{\infty} f(x) \mathrm{e}^{-i \omega t} \, , \, \mathrm{d}x
```

7.4.4 Operator- og funksjonsnavn

Som antydnet ovenfor, skal operatorer og funksjoner som bærer sitt navn ikke nødvendigvis trykkes i kursiv. Sinus, cosinus osv. skal trykkes i “Math Roman”. De fleste kjente operatorer har fått egne $\TeX$ -kommandoer, som `\sin`, `\cos`. Har du en operator eller funksjon som $\TeX$ ikke har en kommando for allerede, må du gjøre det selv, ved f.eks. å bruke kommandoen `\operatorname{sin}` hvis ikke $\TeX$ allerede hadde hatt en egen kommando for sinus. Se [2, side 228]

7.4.5 Tekst inni ligninger

Det som skrives inni mattemodus må man huske på at tenkes på som matematiske symboler, og ikke tekst. Skriver man vann som indeks (for eksempel) til et symbol, så må man passe på at $\TeX$ ikke tror man ganger sammen v , a , n og n . Det gjør den nemlig hvis du er i mattemodus og ikke sier i fra om noe annet. Dette får følger for mellomrommene mellom bokstavene.

Galt:

$$p_{film} = Ch^2$$

mens riktig er

$$p_{\text{film}} = Ch^2$$

med gal kildekode `p_{film} = C h^2` og riktig kildekode `p_{\text{film}} = C h^2`. Man kan også få til kursiv om man ønsker det.

7.5 Definere egne kommandoer

Etter å ha funnet ut etter å ha lest det overstående at inntastinga ble mye kjedeligere, fordi man må skrive så mye mer for å få det riktig, er det godt å vite at man sitter på en UNIX-plattform hvor utvidelsesmuligheter gjennomsyrrer all programvare. Du kan

selvfølgelig lage dine egne L^AT_EX-kommandoer (du kan faktisk programmere ganske så mye hvis du bare orker, men det må du lære et annet sted).

Kommandoen du bør lære deg heter `\newcommand`. Første argument er navnet på din nye kommando, andre argument i hake-parantes er antall argumenter den tar (ikke ta med hake-parantes hvis du ikke trenger noen argumenter) og siste argument inni siste klamme-parantes er din definisjon av kommandoen der `#<tail>` betyr innholdet av argument nr. `<tail>`. Som eksempel gis to eksempler for å raskere kunne sette differensialene riktig som vist ovenfor:

```
\newcommand{\dif}{\mathrm{d}}
\newcommand{\diff}[2]{\frac{\mathrm{d}#1}{\mathrm{d}#2}}
```

Differensialet 'den deriverte av y mhp x ' kan nå raskt skrives korrekt som

```
\diff{y}{x}
```

med trykt resultat

$$\frac{dy}{dx}.$$

Se også [3, side 192] for mer om bruk av `\newcommand`.

Kapittel 8

Figurer i rapporten

Dette kapitlet vil gi noen tips til hvordan generere figurer og hvordan putte dem inn i rapporten.

8.1 Bitmap- vs. vektorgrafikk

Først må en svært viktig distinksjon mellom fundamentalt forskjellige grafikkformater forstås. Et bilde på en datamaskin må ha en representasjon internt i maskinen. Datamaskinen har da to valg, enten å vite om en pixel (et punkt du ser på skjermen) skal være på eller av (for svart-hvitt) eller hvilken farge den skal ha. Denne måten å representere bilder og figurer på kalles bitmapformat.

Alternativet er *logisk* informasjon som kan si den hvordan figuren skal tegnes, for eksempel instruksjonen: Tegn en linje med tykkelse 1 pkt mellom punktene (3,4) og (1,-5), i et spesifisert koordinatsystem. Dette kalles vektorformat.

Rapporten du skriver skal som regel til slutt skrives ut. Da er det ikke representasjonen på skjermen som er viktig, men representasjonen på *utskriften*. En figur som ser pen ut på skjermen (men som er representert i bitmapformat i maskinen) vil se hakkete ut på utskrift, med mindre oppløsningen på figuren er svært høy. En figur som derimot internt er lagret i vektorformat vil se pen ut enten den vises på skjerm eller på utskrift. Vektorgrafikk lar seg også lett skalere opp og ned uten tap av kvalitet (pass likevel på skriftstørrelsene — de blir også skalerte, og da blir det ikke nødvendigvis pent.).

Konklusjonen er altså: Bruk vektorformat til all grafikk hvis mulig. I latex-kurset i seksjon 5.5 er det gjort et forsøk på å forklare grafikk på en best mulig måte.

Bitmapformat kan du bruke på bilder (f.eks. innscannede — de er vanskelig å omgjøre til vektorformat, og det er det heller ikke noe poeng i). Angående bilder så må du også være nøye med distinksjonen mellom formatene gif/png og jpg, se seksjon B.3 for hva distinksjonen går på (kompresjon med eller uten tap av bildedetaljer).

Se også appendiks B for oversikt over flere filtyper.

8.2 Inkludering av figurer i $\text{\LaTeX}$

Under er gitt et eksempel på hvordan ei grafikkfil skal inkluderes i latex-dokumentet. Det anbefales å bruke AuCTeX for å sette inn denne latex-kodesnutten, via C-c C-e (Insert environment) og så spesifisere at man vil sette inn en 'figure'.

```
\begin{figure}[htbp]
  \begin{center}
    \includegraphics[width=.6\textwidth]{gfx/tidssymmetrisk}
    \caption{Illustrasjon av en tidssymmetrisk løsning.}
    \label{fig:tidssymmetrisk}
  \end{center}
\end{figure}
```

Hvis du ikke har brukt malen fra kapittel 6, må du huske på å ha med `\usepackage{graphicx}` for at dette skal virke.

Det inni klammetegn `[]` til `\includegraphics` er valgfrie opsjoner, her sier vi fra hvor stor figuren skal være, og vi vil ha størrelsen proporsjonal med tekstbredden. Da kan du kompilere dokumentet ditt i andre formater hvis ønskelig. Det finnes mange andre opsjoner man kan gjøre, f.eks. å rotere bildet.

Det er viktig, særlig hvis man skal kompilere både til postscript og til pdf, å ikke ta med filendelsen på grafikkfila. Latex vil selv plukke den fila den ønsker, `tidssymmetrisk.eps` hvis den kompilerer til postscript og pdf/jpg/png hvis den kompilerer til pdf (pdf støtter alle disse tre bildeformatene, og den vil lete etter alle).

Det første problemet man støter borti når det gjelder figurer i $\text{\LaTeX}$ er at de ikke alltid blir plassert der man vil. Dette fordi $\text{\LaTeX}$ vurderer nøye (utifra typografiske retningslinjer) hvorvidt en figur kan plasseres akkurat der du har satt den (i forhold til avsnittene rundt figuren i koden). Det er ikke alltid det er plass, og da plasserer $\text{\LaTeX}$ figuren på toppen av neste side, noe som ofte skjer når man har flere figurer.

Hvis man har *veldig* mange figurer kan $\text{\LaTeX}$ få problemer med å få plass til dem, og avslutter kompileringen med feilmeldingen: “*Error: Too many unprocessed floats*”. Da må man til med kommandoen `\clearpage` et par steder for å si klart i fra til $\text{\LaTeX}$ at nå skal vi begynne på ny side og at alle “floats” (figurer og tabeller) som har blitt lagt i $\text{\LaTeX}$'s figurkø skal plasseres suksessivt.

For at du skal kunne referere til figuren er det viktig at `\label`-kommandoen kommer etter `\caption`, hvis ikke vil du få feil tall når du refererer til figuren.

8.2.1 Bitmap-grafikk

For å få importert inn bitmap-grafikk i diverse forskjellige filformater inn i dokumentet, kan du enten konsultere diverse $\text{\LaTeX}$ -grafikk-dokumentasjon og finne ut hvordan man gjøre det for hver enkelt grafikkformat, eller du kan gjøre grafikken om til postscript eller pdf.

Har man mange bilder (typisk innscannede), anbefales det å bruke pdf som filformat for ferdig fil, da den takler jpg og png direkte. Bruker du postscript for ferdig fil, må du konvertere alle grafikkfiler over til eps-format.

For å konvertere dine bilder og figurer til postscript eller png, bruker du UNIX-kommandoen `convert`. Prøv

```
syLOW:~/prosjekt/rapport/gfx:$ convert bilde.gif bilde.eps
```

eller

```
syLOW:~/prosjekt/rapport/gfx:$ convert bilde.gif bilde.png
```

og du har fått en postscript- eller png-versjon av bildet ditt. Denne prosessen kan også automatiseres med `make`-verktøyet, omtalt i kapittel 11. NB: Bildet har eps-fila ikke blitt til vektorformat, det har kun blitt “embedded” i postscript som et bitmap-bilde (postscript støtter også bitmap, men ikke komprimert bitmap som gif og jpg er, så filene blir veldig store).

8.2.2 Flere figurer i samme figur — `subfig`

Figurer som er relaterte til hverandre vil man ofte samle sammen til en figur med underfigurer. Dette gjøres med latexpakken `subfig` som vist i eksempelet under. I dette eksempelet er en annen variant av skalering av figurene også vist.

```
\begin{figure}[htbp]
  \begin{center}
    \subfloat[Vanlig plott]{
      \includegraphics[width=0.4\textwidth]{gfx/plot_res_SSOR_CG}}
    \subfloat[Log-log plott]{
      \includegraphics[width=0.4\textwidth]{gfx/loglog_res_SSOR_CG}}
    \caption{Residualfeil mot iterasjoner for CG
      med SSOR som prekondisjonerer.}
    \label{fig:cgssor_residual}
  \end{center}
\end{figure}
```

NB: Husk at du må ha med `\usepackage{subfig}` og `\usepackage{graphicx}` øverst i `master.tex`, som angitt i malen i kapittel 6. Vi har heller ikke her tatt med filendelsene på eps-filene våre. Latex vet selv at den kan legge til eps eller pdf alt etter som for å bruke riktig fil.

Underfigurene blir nummerert med bokstaver og starter med *a*. Du kan referere til underfigurene ved å putte en `\label{...}` inni klammeparantesen til `\subfig` (du trenger ikke ta bort label for hele figuren), og så referere til disse som vanlig (du vil få med både tall og bokstav når du refererer til underfigur).

NB: `subfig`-pakken er en ny pakke som erstatter den tidligere brukte pakken `subfigure`. `Subfigure` brukte kommandoen `\subfigure` i stedet for `\subfloat`.

8.3 Tegning av figurer med Xfig

Xfig er et vektorbasert tegneprogram som finnes på alle UNIX-maskinene på skolen. Det er ikke “profesjonelt”, men derimot gratis, og fungerer utmerket til de aller fleste formål. Programmet startes med å skrive `xfig` på kommandolinja.

Funksjonene i Xfig bør man gjøre seg kjent med på egenhånd. Bruk Grid Mode (nede til venstre) flittig. Vær også obs på hva figuren oppe til høyre sier at de forskjellige musknappene betyr, avhengig av hvilken funksjon du har valgt. Ved lagring av bildet, så lagrer du figuren din både i `fig`-format (trykk høyreknappen på File-knappen) og så eksporterer du en kopi i `eps`-format (høyreklikk på Export-knappen). `fig`-filen du lagrer til er filen du åpner når du skal gjøre endringer på figuren, etter hver endring må du eksportere til `eps` på nytt. `eps`-fila er den du importerer inn i latex-dokumentet. Du kan ikke like enkelt endre på `eps`-figurer som på `fig`-figurer.

8.3.1 $\text{\LaTeX}$ kommandoer inni xfig-figurer

Noe av det som gjør Xfig ufattelig kjekt er at for enkle figurer så kan man generere $\text{\TeX}$ -kode for å lage figuren. ($\text{\LaTeX}$ er en forenkling av språket $\text{\TeX}$). Dette medfører at man kan ta med noenlunde kompliserte matematiske uttrykk i figuren sin, og ikke minst får man korrekt skrifttype på tekst i figuren.

For å få dette til er det et par viktige ting som må huskes på når du er inne i Xfig. Hver gang du skal skrive tekst, så må du huske å sette på et *special*-flagg. Dette kan du skru på etter å ha trykket på 'T'-knappen for å skrive inn tekst, ved å trykke på 'Text flags' knappen nede i vinduet. Her må du skru på *Special flag*. Hvis du allerede har skrevet inn teksten, kan du bruke 'Edit'-funksjonen og skru på 'special' da. Teksten du skriver inn er $\text{\LaTeX}$ -kode i tekstmodus. Skal du ha inn noen symboler, setter du dollartegn rundt dem, f.eks. kan du skrive θ_2 og resultatet i rapporten din vil bli θ_2 .

Prinsippene bak dette er å lage $\text{\TeX}$ -kode av teksten og matematikken din, og postscript-kode av tegningene dine. Så lar vi $\text{\LaTeX}$ lage et bitte lite dokument som inkluderer tegningen din, og limer $\text{\TeX}$ -koden med tekst og matematikk oppå. Dette bitte lille dokumentet kompilerer vi og lager et lite postscript-bilde av. Det finnes en kommando tilgjengelig, `figtex2eps`, som automatiserer dette for deg og er installert på studentdatalaben for matematikk og datasalene til Fysikk og matematikk.

Gitt at du har laget fila `mangfoldighet.fig` i xfig, og har lagt inn $\text{\LaTeX}$ -kode der som angitt over. For å lage et postscript-bilde av figuren skriver du

```
$ figtex2eps mangfoldighet.fig
```

og så blir fila `mangfoldighet.eps` laget. Dette postscript-bildet kan du inkludere i dokumentet som vist over. Vil du ha pdf i stedet for postscript, kan du bruke `figtex2pdf` i stedet for, som du bruker på akkurat samme måte. `figtex2pdf` går via `eps` for å få laget pdf, så `eps`-fila blir også generert og blir liggende igjen.

Programmet og mer dokumentasjon er tilgjengelig på web fra <http://www.math.ntnu.no/~berland/figtex2eps>.

Du kan automatisere kjøringen av kommandoen `figtex2eps` med Make. Se kapittel 11 for mer informasjon om verktøyet Make.

8.4 Inkludering av figurer fra Matlab

Lager man figurer i Matlab kan man manuelt gå inn i figur-menyen og trykke på print/save og skrive inn filnavn. Da dette er tungvint å gjøre hver gang man gjør en liten endring, anbefales det å scripte prosessen for å lage postscript-filene (man vil nesten aldri lage noe annet enn postscript av Matlab-figurene, pdf støtter foreløpig Matlab ikke).

8.4.1 Kildekodeeksempel

Eksempel på et matlabscript som lager en figur og skriver figuren (som postscript) til ei fil.

```
points=20; % points per dimension to plot

xy = [0:1/points:1]; % the same vector is used both for x and y.

Z = zeros(points+1, points+1);

for i=1:points+1;
    for j=1:points+1;
        Z(i,j) = 4 * xy(i) * (1-xy(i)) * (1-xy(j));
    end
end
xlabel('x')
ylabel('y')
zlabel('u(x,y,0)')
colormap(hot)
print -depsc2 'gfx/heatdistribution.eps'
```

8.4.2 Inkludering av figurer fra Matlab med $\text{\LaTeX}$

Mye penere matlab-plott får du hvis du får inn riktig $\text{\LaTeX}$ -font på tallene og teksten i plottet ditt. Dette kan du gjøre med pakken “LaPrint”, som du finner på websida <http://www.mathworks.de/matlabcentral/fileexchange/loadFile.do?objectId=4638>.

8.5 Inkludering av figurer fra Gnuplot

Gnuplot er et plotteverktøy. Ikke så avansert som matlab, men gratis og ikke så tungt å starte. For enkle plott av data er dette helt kurant å bruke. Gnuplot lar seg også scripte på en enkle måte slik at det kan automatiseres med make.

Gnuplot kan startes fra kommandolinja med kommandoen `gnuplot`. Prøv f.eks. `plot sin(x)`. Hjelp får man ved å skrive `help plot` for eksempel.

8.5.1 Kildekodeeksempel

Gnuplot blir effektivt når man i stedet for å skrive kommandoene inn etter å ha startet gnuplot, plasserer kommandoene i ei fil. Dataene finnes i dette eksempelet i fila `data/seq_timings` som har f.eks. følgende innhold:

```
4      0.0001  0
8      0.0010  0.0001
16     0.0052  0.0009
32     0.0259  0.0066
64     0.1225  0.0619
128    0.5631  0.4919
256    3.12    6.25
512    12.38   71.15
1024   54.71   596.04
```

og using 1:2 i det følgende eksempelet betyr at man skal plote tallene i kolonne 2 som funksjon av tallene i kolonne 1. Les gnuplot sin hjelp om hver av disse kommandoene for å lære hva mer man kan finne på å gjøre:

```
set term postscript eps
set output 'gfx/seq_timings.eps'
set xlabel 'Problem size (n)'
set ylabel 'Time in seconds'
set logscale x 2
set logscale y
plot 'data/seq_timings' using 1:2 title 'poisson_seq_dst' \
    with linespoints pointtype 3, \
    'data/seq_timings' using 1:3 title 'poisson_seq_mxm' \
    with linespoints pointtype 4
```

8.5.2 GnuPlot og $\text{\LaTeX}$

Med Gnuplot kan man også få laget plott med korrekt $\text{\LaTeX}$ -font akkurat som man kan med Matlab og Laprint.

Da bytter man ut `set term eps` over med `set term pslatex` og man bruker også `set output 'gfx/seq_timings.tex'`. Etterpå kan man bruke det ferdige scriptet `gptex2eps` eller `gptex2pdf` for å både kjøre Gnuplot og å kompilere etterpå med $\text{\LaTeX}$ for å få riktige fonter. Dette fungerer etter samme prinsipp som for Xfig. Mer informasjon finner man på <http://www.math.ntnu.no/~berland/gptex2eps/>

Kapittel 9

Hvordan lage referanseliste/bibliografi

Referanseliste eller bibliografi er ei liste over kilder du har referert til eller anbefalt i rapporten din. Denne lista skal komme etter siste kapitlet ditt, men før appendikset.

Det finnes ganske strenge regler for hvordan en bibliografi skal se ut, så det anbefales å få latex til å lage en for deg, og ikke prøve å lage den selv.

Man tar utgangspunkt i ei databasefil som slutter på `.bib`. Denne har sitt eget format (BIB_T_EX) som du må følge. Emacs sammen med AUC_T_EX vet om formatet på denne fila, og det anbefales å bruke menyene flittig for å få satt inn en “databaseentry” for ei bok.

Som eksempel vises her hva som skal stå i `.bib`-fila for læreboka i numerisk matematikk:

```
@Book{nummat,
  author =      {G\"unther H\"ammerlin and Karl-Heinz Hoffman},
  title =       {{N}umerical {M}athematics},
  publisher =    {Springer-Verlag},
  year =        {1991},
}
```

- Strengen `nummat` er noe du velger selv. Det er en “label” du setter på boka, som du bruker når du steder i dokumentet vil referere til denne boka.
- Legg merke til at ordet “and” er brukt for å skille forfatterne. Ikke bruk “og” hvis du skriver norsk, fordi BIB_T_EX forstår logisk sett hva “and” betyr og kan gjøre forskjellige ting med navnelista utifra det.
- Bokstaver som du vil være garantert skal være store, putter du inni klammeparanteser. På forfatternavn er BIB_T_EX stort sett enig med deg i at disse skal begynne med store bokstaver, men i `title`-feltet så kan det godt være at alle andre ord enn det første for små forbokstav, uavhengig av hva du har her. Derfor, bruk klammeparanteser når du trenger det.

- Når du setter inn en “entry” for en bok eller artikkel via menyene i emacs, vil noen av feltene prefikses med enten OPT eller ALT. Disse bokstavene skal du ikke ha med for at det skal virke, men OPT betyr at feltet er OPTional - valgfritt, du kan enten ta bort hele linja, eller bare de tre bokstavene og sette inn en verdi i feltet. ALT står for alternativ, og du skal da velge *kun en* av de som er gruppert sammen prefikset, sette inn en verdi, og ta bort prefikset ALT.

For å referere til denne boka kan du for eksempel lage følgende fiktive latex-kode:

```
En innledning til approksimasjonsteori finnes i \cite{nummat}.  
For  $L^p$  rommene har fra \cite[avsnitt 4.1.4]{nummat} at ...
```

Klammeparantesene er valgfrie og brukes når man vil spesifisere nøyaktig hvor i boka man refererer til. For eksempel avsnittsnummer, kapitler eller ligninger.

I hovedfila di, kanskje ved navn `master.tex` må du også ha med noen linjer der du skal ha inn bibliografien din:

```
\bibliography{kurs}  
\bibliographystyle{plain}
```

der “plain” er vår valgte måte å sette opp bibliografien vår på. Referansene vil her bestå av et tall inni klammeparantes. Denne er vanlig i de fleste matematiske artikler.

Hvis du ikke liker å ha tall i klammeparanteser inni teksten som referanser, men heller kan tenke deg etternavn og årstall inni en parantes for eksempel, kan følgende brukes:

Øverst i hovedfila di må du ha med linja

```
\usepackage{harvard}
```

som du putter sammen med de andre `usepackage`-kommandoene. Så må du i stedet for ‘plain’ skrive ‘agsm’ i argumentet til `bibliographystyle`. Dette er bare et eksempel på en annen stil som virker her. Det finnes et utall andre, som du kan finne eksempler på ved å lese dokumentet “Lokal guide til BibTeX” som du finner link til fra dette kursets webside, <http://www.stud.math.ntnu.no/kurs/>. Merk at ikke alle disse nødvendigvis er installert på NTNUs dataanlegg.

Kompilering av dokumentet må etterpå gjøres flere ganger. Etter å ha lagt inn flere `\cite`-kommandoer og flere bøker/artikler i `.bib`-fila må du kjøre `latex master.tex` for at latex skal finne ut hvor du vil referere til. Så må du kjøre BIBTeX på `.bib` fila, og så latex en eller to ganger til for at alt skal bli korrekt. Hver gang du har lagt til noe nytt i `bib`-fila må du kjøre BIBTeX på nytt for å få det med. Når du bare legger til flere referanser til ei allerede referert bok, trenger du bare kjøre latex en gang eller to.

```
sylow:~/prosjekt/rapport:$ latex master.tex  
...  
sylow:~/prosjekt/rapport:$ bibtex master  
...
```

```
syLOW:~/prosjekt/rapport:$ latex master.tex
...
syLOW:~/prosjekt/rapport:$ latex master.tex
```

Bibliografien din kommer ikke med i innholdsfortegnelsen uten videre. Dette kan du gjøre selv, ved å legge til linja

```
\addcontentsline{toc}{chapter}{Bibliografi}
```

i hovedfila di, rett etter `\bibliography` kommandoen din. Bytt ut `chapter` med `section` hvis seksjoner er ditt høyeste nivå.

Hvis du jobber med pdf, bytter du ut `latex` med `pdflatex` i alt ovenfor, men `bibtex` er den samme.

Kapittel 10

Hvordan lage register/indeks

Et register (eller indeks) er til hjelp for leseren. Registeret skal hjelpe lesere å raskt finne fram til hvor ting og tang står i rapporten din. Sitat fra [3, side 151]

Most books have indexes; most technical reports don't. They should. Any nonfiction work of more than twenty or so pages that is worth reading deserves an index, *Leslie Lamport*

For å lage et register til rapporten, må du kommenter ut de indikerte linjene på starten og slutten av `master.tex`, se seksjon 6.1.1.

Der du vil ha indeksert noe, legger du inn `\index{indeksord}` i teksten, og det vil etterhvert komme med i registeret bakerst i rapporten.

Etter å ha kjørt `latex` på dokumentet har du fått ei fil kalt `rapport.idx`. Du må nå kjøre `makeindex` på denne `.idx` fila for å få generert `rapport.ind` som inneholder indeksskommandoer til `latex`. Så kjører du `latex` på nytt, og denne gangen tar den med det som er blitt skrevet i `rapport.ind`, og du får med et register.

```
sylow:~/prosjekt/rapport:$ latex master.tex
...
sylow:~/prosjekt/rapport:$ makeindex master.idx
...
sylow:~/prosjekt/rapport:$ latex master.tex
...
sylow:~/prosjekt/rapport:$ xdvi master.dvi
```

Nødvendig lesning er seksjon 4.5 og A.1 i [3].

Kapittel 11

Bruk av verktøyet make

Make er et program skrevet for å holde styr på store programmeringsprosjekter der man har mange filer med kildekode. Typisk tar det lang tid å kompilere hele programmet om igjen hvis det har litt størrelse, og vanskelige kommandoer å huske i hodet. Make kan selv finne ut hvilke filer som er modifisert siden sist kompilering, og kan automatisere lange rekker av kommandoer.

I dette tilfellet vil vi ikke bruke alt det make kan. Her vil det kun brukes for å automatisere kjøring av kommandoer, slik at man slipper å huske alt, og også sparer seg for mange tastetrykk.

Make benytter seg av en kommandofil, *Makefile*, som du har en av for hvert prosjekt du har. Man kan også operere med en *makefile* for hver katalog man har, men det brukes ikke her.

11.1 Eksempel på kokbar Makefile

Under er det gitt en brukbar *Makefile* som du kan koke. Bruk emacs til å endre den første variabelen *MASTER* til det du som kaller hovedfila di. Hvis du har kokt resten av kurset, har du trolig kalt denne for rapport allerede, og da trenger du ikke gjøre noe.

Under er det tatt med en anbefalt *Makefile*. Denne kan du hente ned fra <http://www.stud.math.ntnu.no/kurs/maler/Makefile> og legge i rapportkatalogen din.

```
1 # Din hovedkildefil, endre hvis nødvendig: (endelsen .tex skal ikke være med)

MASTER = rapport

5 # Kort bruksanvisning:
# =====
#
# Hvis du ikke har bibliografi eller index i rapporten:
10 # $ make all clean
```

```

# Hvis du har bibliografi og index i rapporten:
#   $ make bibandindex all clean
# Hvis du bare har bibliografi og ikke index:
#   $ make bib all clean
15 # For å skrive to sider pr. side:
#   $ make psnup # skriv ut 'rapport.2.ps' etterpå med 'lpr'
# For å pakke ned slik at du ikke bruker så mye plass
#   $ make pack # 'make unpack' for å pakke opp etterpå.

20
# Hvis du kaller bibliografifila di med samme basisnavn som
# hoveddokumentet, trenger du ikke endre her. Hvis du kaller den noe
# annet, må du spesifisere her (som i eks. under).
# NB: Ikke ta med .bib-endelsen!
25 BIBMASTER = $(MASTER)
#BIBMASTER = bibliografi

# IKKE endre noe under her med mindre du vet hva du gjør!

30 all: $(MASTER).dvi $(MASTER).ps $(MASTER).pdf

$(MASTER).dvi: *.tex
    @latex $(MASTER).tex
    @if grep 'LaTeX Warning: Label(s) may have changed' $(MASTER).log \
35      >/dev/null ; then latex $(MASTER).tex ; else true ; fi

%.ps: %.dvi
    dvips $< -o

40 $(MASTER).pdf: *.tex
    pdflatex $(MASTER).tex

%.bib: %.tex
    latex $<

45
# Kjekkk, tvinger make til å innse at kildefila faktisk er endret hvis den ikke
# gjør det selv. Kjør f.eks. 'make force all'
force:
    touch $(MASTER).tex

50
bib: $(BIBMASTER).bib
    latex $(MASTER)
    bibtex $(MASTER)

```

```

    latex $(MASTER).tex
55    latex $(MASTER).tex
    dvips $(MASTER).dvi -o

%.idx: %.tex
    latex $<

60 index: $(MASTER).idx
    latex $(MASTER).tex
    makeindex $(MASTER)
    latex $(MASTER).tex
65    latex $(MASTER).tex

bibandindex: $(MASTER).bib $(MASTER).idx
    latex $(MASTER)
    bibtex $(MASTER)
70    makeindex $(MASTER).idx
    latex $(MASTER).tex
    latex $(MASTER).tex

tags:
75    etags *.tex

psnup: $(MASTER).ps
    psnup -2 $(MASTER).ps > $(MASTER).2.ps

80 pack:
    rm -rf *~ *.aux *.bbl *.blg *.dvi *.idx *.xyc TAGS \
        *.ilg *.ind *.log *.toc *.lof *.lot \##\# *.2.ps
    bzip2 -q -9 *.tex *.pdf *.ps */*.eps */*.ps */*.tex

85 unpack:
    bunzip2 *.bz2 */*.bz2

printsize:
    du -sh .

90 clean:
    rm -rf *~ *.aux *.bbl *.blg *.dvi *.idx *.xyc TAGS \
        *.ilg *.ind *.log *.toc *.lof *.lot \##\# *.2.ps

```

Som bruksanvisningen i fila antyder, så er det bare å skrive disse kommandoene, så blir det du ønsker gjort. Du slipper for eksempel å huske kombinasjonen av kommandoer for å generere indeks og bibliografi, (make bibandindex) og du får også en enkel måte å

rydde opp i katalogen når du er ferdig med prosjektet (`make clean`). Kikk på hvordan for eksempel `make clean` fungerer utifra fila, og pass på at du ikke lager filer med samme endelse som er gitt der, for de filene regnes som midlertidige filer som lett kan genereres på nytt, og som derfor fjernes når man vil rydde opp. Dukker det opp andre filer som du også vil ha ryddet bort når det trengs, så er det bare å legge dem til i lista over endelser.

`make pack` kan også trenge litt modifikasjon. Du vil typisk pakke andre filer i tillegg til de som er nevnt. Legg dem bare til.

Kapittel 12

Lage slides med Beamer

Beamer er en $\text{\LaTeX}$ -dokumentklasse for å lage slides for presentasjon vha. dataprojektør eller for tradisjonell overhead.

For de som kan $\text{\LaTeX}$ fra før, er det en svært liten brukerterskel for å skrive slide i beamer. Brukermanualen til beamer er svært god med mange eksempler, så her er det begrenser hvor mye som nevnes.

12.1 Eksempel

Her er det inkludert et eksempel på en enkel presentasjon med forside og en slide. Du finner igjen samme kildefila og kompilert til postscript og pdf på websida til kurset. Legg merke til at vi har spesifisert *minstil* som dokumentopsjon, og da må fila *PPRminstil.sty* vedlagt under ligge i samme katalog som denne fila.

12.1.1 Fileksempel: presentasjon.tex

```
1 \documentclass[screen]{beamer} % Bruk 'handout' i stedet for 'screen' for utskrift.
  \usepackage[latin1]{inputenc}

4 %% Utseende på slides:
  \useinnertheme{rounded} %avrundede itemize-lister
  \useoutertheme{shadow} % fotlinje og topplinjer
  \usecolortheme{whale}   % mørkeblå og svart stil, hvit bakgrunn

9 % Sett inn dine data her:
  \title{Om normalfordelte karakterer}
  \author{Karl Friedrich Gauss}
  \subtitle{Diplomoppgave levert 14. juni 1798}
  \institute{NTNU}
```

14

```

% Begynn dokumentet:
\begin{document}

19 \begin{frame}
    \maketitle
\end{frame}

\begin{frame}
24 \frametitle{Oversikt}
    \tableofcontents
\end{frame}

\section{Introduksjon}

29 \begin{frame}
    \frametitle{Introduksjon}
    Normalfordelingskurven har funksjonen
    \begin{equation*}
34 P(x) = \frac{1}{\sigma \sqrt{2 \pi}}
        \mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}}
    \end{equation*}
\end{frame}

39 \begin{frame}
    \frametitle{Momenter}

    \begin{itemize}
        \item Bombepunkt 1
44 \begin{itemize}
            \item Underbombepunkt 1
            \item Underbombepunkt 2
                \begin{itemize}
                    \item foo bar
49 \end{itemize}
                \end{itemize}
            \item Bombepunkt 2
        \end{itemize}
\end{frame}

54 \end{document}

```

12.1.2 Overlays

Beamer er spesielt kraftfullt når det gjelder å få til såkalte overlays i presentasjonen din. Det vil si at f.eks. bombepunkter på en slide kommer fram trinnvis. Mange vanlige $\LaTeX$ -kommandoer får en ekstra “overlay-syntaks” når du bruker beamer. Rett etter kommandoen kan du skrive `<overlay-spec>` der *overlay-spec* er et åpent eller lukket intervall av på hvilke overlay innholdet av kommandoen skal komme på. F.eks. vil `\item<2->` bety at dette bombepunktet skal komme fram fra overlay nr. 2 på denne slide og for alle kommende overlays på denne slide. `\includegraphics` støtter også denne syntaksen.

Hvis du i dokumentklasseopsjonen endrer fra `screen` til `handout`, så vil alle overlay-spesifikasjoner kollapse, slik at du slipper trinnvis visning på utskrift (der er overlays vanligvis bare tøys å ha).

Bruker du mye overlays som også forsvinner på kommende overlays på samme slide, vil standard-kollaps til `handout` ikke virke tilfredstillende. Da må du i tillegg i *overlay-spec* spesifisere hvordan dette skal kollapse. Se beamer-dokumentasjonen for dette.

12.1.3 Andre effekter

I beamer kan du få til mange andre effekter. Her nevnes bare `\alert` som kan brukes for å tiltrekke noe oppmerksomhet til kommandoens argument. Denne støtter selvsagt også overlay-spesifikasjoner.

12.1.4 Bytte utseende på presentasjonen

Beamer deler opp utseendet til hver slide i logiske deler, og innenfor hver del har du mange forskjellige tema-filer å velge mellom. Eksempelet over velger en “inner-theme” som avgjør hvordan itemize-lister skal se ut og lignende, det velger så et “outer-theme” som i dette eksempelet legger på noen få skyggeoverganger. Til slutt velges et fargetema som gir fargeovergangen i tittelbakgrunnen og lignende. Er man stø i $\LaTeX$ og lærer seg hvordan beamer-temaene er bygd opp, kan man skrive egne, eller gjøre modifikasjoner til temaene. Beamer-dokumentasjonen er da rett sted å starte, og også dokumentasjonen til `xcolor` (som er nødvendig for å forstå den noe omstendelige men kraftige måten beamer behandler farger på).

12.2 Kompilering

Beamer er laget med tanke på at man skal bruke `acroread` for å presentere presentasjonen til slutt. `pdflatex` er derfor programmet du skal bruke for å kompilere presentasjonen. Men `postscript` er støttet det også, og kan være greit å bruke under skriving da `gv` fungerer bedre i unix enn `acroread` gjør.

12.3 Vise presentasjonen

Hvis du ikke har tilgang til bærbar pc eller fastmontert pc, tar du utskrift på transparen-ter. Dette kan såklart også være kjekt å ha i bakhånd om alt skjærer seg, noe det av og til gjør.

Acroread anbefales for å vise dokumentet via prosjektør. Åpne dokumentet i acroread, og trykk `ctrl-l`, da går acroread over i fullskjerm-modus.

Mens du skriver på presentasjonen, holder det lenge å bare bruke postscript. En passende `gv`-kommando kan da være

```
$ gv -watch -landscape -scale -2 presentasjon.ps &
```

Bibliografi

- [1] Per Cederqvist et al. Version management with cvs, 1993. <http://www.stud.math.ntnu.no/kurs/etc/cvs.ps>.
- [2] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The Latex Companion*. Addison-Wesley, 1994.
- [3] Leslie Lamport. *LaTeX, User's Guide and Reference Manual*. Addison-Wesley, 1994.
- [4] American Mathematical Society. Ams-latex user guide, 1996. <http://www.stud.math.ntnu.no/kurs/latexdocs/amslatexdoc.ps>.
- [5] Kresten Krab Thorup and Per Abrahamsen. Auctex user guide. <http://www.stud.math.ntnu.no/kurs/etc/auctex.ps>.

Tillegg A

Bruk av cvs for versjonskontroll og gruppearbeid

A.1 Hvorfor bruke cvs

Cvs står for “concurrent version system”. Du kan ta det i bruk hvis du ønsker å holde orden på historien til alle filene dine og sørge for at flere personer kan editere samme fil hver for seg uten at det blir krøll. Ved bruk av cvs slipper du mye av styret med å lage symbolske linker for hver av de andre på gruppa og å holde for mye på med gruppe-rettigheter på filene. Det kommer garantert til å bli noe styr med folk som glemmer å sette på skrivemuligheten for andre gruppemedlemmer (seksjon 3.3) (men det er ikke dermed sagt at cvs er mye enklere enn å ikke bruke det).

Jobber du på to forskjellige maskiner (hvor du ikke deler samme hjemmekatalog), for eksempel på en bærbar og på skolens datasystemer, kan også cvs være nyttig for å klare å holde begge dine versjoner synkroniserte.

A.2 Alternativ til cvs— Subversion

Cvs er en aldrende programpakke for versjonskontroll, og en erstatning er under utvikling. Dette prosjektet heter Subversion <http://subversion.tigris.org/> og er allerede blitt bedre enn cvs. Installasjonen er derimot ikke bombesikker på NTNU ennå (vår 2005), så for å minimere antall problemer som oppstår, så er det fortsatt cvs som får hovedomtalen i dette kapitlet. Oppsett og bruk av Subversion er veldig likt som for cvs, bra dokumentasjon både for de som er nye for versjonskontroll og for de som kan cvs, finnes på websida nevnt over.

A.3 Oppsett av cvs for gruppeleder

En på gruppa (ikke nødvendigvis en leder, men en..) må sette opp en “master”-katalog for prosjektet, eller et “repository” i cvs-terminologi. Denne katalogen kan man for

eksempel kalle for cvsrepository.

```
syLOW:~:$ mkdir cvsrepository
```

Så må du la cvs få legge inn noen filer som den har lyst å ha i et repository.
Dette gjør du rett etterpå med kommandoen:

```
syLOW:~:$ cvs -d ~olenils/cvsrepository init
```

Lag også en katalog inni dette “repository” som skal være prosjektkatalogen. Du kan ha mange forskjellige prosjekter inni ditt repository.

Hvis du holder på med et prosjekt i Lineære metoder, kaller du f.eks. katalogen linmet. Lag den, gi den riktig gruppe og riktige grupperettigheter, akkurat som i seksjon 3.3.

A.4 Oppsett av cvs for alle gruppemedlemmer

Dette er noe som må gjøres av alle på gruppa, også den personen som har repository på sin hjemmekatalog.

Cvs må bli fortalt hvor repository ligger hen på serveren. Dette skal inn i en global “miljø”-variabel som skal gjelde for brukeren din. For at denne variabelen alltid skal være satt for din bruker, legges den i en oppstartsfil som kjøres hver gang du logger inn. Denne fila heter .bashrc. Du trenger å legge til noen linjer i denne fila. Vi skal under her gjøre dette uten å starte en editor som pico eller emacs, men det kunne også ha vært gjort med de. Husk å stå i roten av hjemmekatalogen din når du gjør dette (og husk for all del å skrive nøyaktig!):

```
syLOW:~:$ echo "export CVSROOT=~olenils/cvsrepository" >> .bashrc
syLOW:~:$ echo "export CVSEditor=pico" >> .bashrc
```

I den første linja må du huske å bytte ut olenils med brukernavnet til den på gruppa di som har laget et repository. Den neste linja kan du bytte til den editoren du liker best. F.eks emacs (men du kan etterhvert bli litt irritert over at emacs bruker noen flere sekunder på å starte opp enn pico).

Etter dette kan hver bruker sjekke ut sin lokale kopi av prosjektet. Cvs fungerer slik at hver bruker får sin egen kopi av alle prosjektfilene som han/hun editerer. Når brukeren synes han/hun har gjort en god jobb, “sjekker” han/hun inn endringene sine i cvs igjen. Etter dette kan andre få tak i det brukeren har gjort.

Hver bruker (også brukeren med repository) kan på dette tidspunkt skrive kommandoen

```
syLOW:~:$ cvs checkout linmet
```

og så får man en katalog som heter linmet som er brukerens egen lokale kopi av prosjektet (altså ikke noen symbolske linker).

A.5 Legge inn filer i prosjektet

Etter å ha sjekket ut prosjektet med `cvs co linmet` er det kanskje ikke så mange filer der. Disse må legges inn. Kun en person trenger å legge inn filene, men hvem som helst kan gjøre det. Hvis malen som foreslått i de andre kapitlene i dette kurset følges, legger man disse filene inn i prosjektkatalogen (*aldri* direkte i repository) og så sier man etterpå i fra til cvs om at disse filene vil man ha med i prosjektet. For filene som nevnt i kapittel 6 så skriver man kommandoene:

```
syLOW:~/linmet:$ cvs add master.tex
syLOW:~/linmet:$ cvs add forside.tex
syLOW:~/linmet:$ cvs add innledning.tex
syLOW:~/linmet:$ cvs add resultater.tex
syLOW:~/linmet:$ cvs add appendiks.tex
syLOW:~/linmet:$ cvs add Makefile
```

Cvs vil for hver fil svare at denne fila nå er klargjort for å legges inn i prosjektet (men de ligger ikke i repository ennå). Når man har skrevet `cvs add` kommandoen for alle filene man har for øyeblikket, sier man i fra til cvs om at man nå vil gjøre denne endringen permanent (det vil si at de andre også kan få tak i de endringene man har gjort - i dette tilfelle å legge til filer). Dette gjøres med kommandoen `cvs commit` som man skriver inn i prosjektkatalogen. Da vil den editoren du spesifiserte i `CVSEDITOR` startes opp og du skal skrive inn en liten logg over hva du har gjort. Når du avslutter editoren, blir filene lagt til i prosjektet (repository).

Etter dette, kan de andre gruppelemmene få tak i dine endringer ved å oppdatere sin lokale kopi med kommandoen

```
syLOW:~/linmet:$ cvs update
```

A.6 Mer om commit og update

A.6.1 Sjekke inn en fil om gangen

Du vil kanskje bare sjekke inn en fil som du har endret på, og ikke de andre. Enten fordi du ikke vil de andre endringene skal bli permanente, eller fordi du vil ha en forskjellig logg for hver fil. Da skriver du bare `cvs commit filnavn`.

A.6.2 Konflikter

Når to personer endrer på samme fil (nesten) samtidig, kan det oppstå konflikter som cvs ikke har lyst til å løse på egen hånd. Hvis man jobber i separate deler av en fil, så er ikke dette noe problem, men når to brukere har gjort forskjellige endringer i samme avsnitt, er det ikke lett for datamaskinen å avgjøre hva som skal gjøres.

I de tilfellene får den siste av de to som gjør en commit beskjed om at det finnes konflikter, og at disse må løses. Da må en inn og endre igjen på fila som skaper problem.

Inni denne fila har cvs lagt inn noe informasjon om hvilke endringer den andre brukeren har gjort. Fiks fila til slik du mener den skal være (etter å kanskje ha snakket med den andre) og gjør en ny commit.

A.6.3 Meldinger fra cvs under `cvs update`

Når en bruker oppdaterer sin lokale arbeidskopi av prosjektet med kommandoen `cvs update` kommer det forskjellige meldinger for hver fil. På de filene som du har liggende i katalogen din, men som ikke er lagt til i cvs (og som du ofte heller ikke ønsker å legge til) vil du få et spørsmålstegn foran filnavnet. Dette betyr bare at cvs ikke vet om fila. Hvis disse filene irriterer deg, kan du legge disse filnavnene inn i ei fil kalt `.cvsignore` i prosjektkatalogen.

Bokstaven U betyr at din lokale kopi av fila har blitt oppdatert fra cvs-repository.

Bokstaven A betyr at du nettopp har lagt fila til med `cvs add` men at du må gjøre en commit før endringen blir lagt til i repository.

Bokstaven M betyr at du har modifisert din kopi av fila i forhold til det som ligger i repository. Du må gjøre en commit før din endring blir permanent.

Bokstaven C betyr at en konflikt oppstod når cvs prøvde å oppdatere fila fordi en annen også har endret i den samme fila på omtrent samme sted.

A.7 Sletting av filer i prosjektet

Hvis du har filer liggende som du ikke lenger vil skal være med i prosjektet, kan du be cvs fjerne dem fra repository. Da sletter du først fila fra din lokale kopi, og så kjører du en `cvs remove filnavn` etterfulgt av en commit. De andre på prosjektet vil få fila slettet neste gang de kjører `cvs update`. Selv om fila er slettet fra prosjektkatalogen din, så kan den fortsatt letes fram fra cvsrepository igjen hvis du etterpå skulle angre.

Å bytte navn på filer er ikke enkelt i cvs. Navnebytte består egentlig i at du først bytter navn eller flytter fila i din lokale kopi av prosjektet og så ber cvs slette den originale fila, og så legger du til fila med nytt navn som ei ny fil med `cvs add`. Du får ikke overført historien (alle loggbeskjedene) fra den gamle fila til den nye.

A.8 Hvilke filer skal med i repository?

Ikke alle dine filer er det lurt å ta med i repository. Kun det som går som *kildefiler* er det lurt å ta med. Postscriptkoden som genereres av latex og dvips er det for eksempel ikke lurt, for denne skal alle kunne lage selv med en `make` kommando om du har tatt `Makefile` i bruk. Heller ikke grafikk som er generert via script og som lett kan regenereres (f.eks. via `make` om du har fått til det).

Grafikk som *ikke* kan regenereres kan være greit å ta med, men vær obs på at filer som *ikke* er vanlige tekstfiler må behandles spesielt i cvs. Med bildefiler som jpg og gif for eksempel må du si fra til cvs om at de er binærfiler og ikke vanlige tekstfiler. Hvis

ikke kan du risikere at CVS ødelegger filene. Binærfiler må du legge til med en opsjon til `cv`s `add`:

```
syLOW:~/linmet:$ cvs add -kb norwaymap.jpg
```

A.9 Logg og versjonsnummer for hver fil

CVS har en egen logg (som er alle beskjedene som er blitt skrevet under `cv`s `commit`) og et versjonsnummer for hver fil. Dette kan du ta med i filene dine slik at du har det lett tilgjengelig. Linjene med slik informasjon må du som regel kommentere ut med tanke på hvilket program som skal prosessere filene. For $\text{\LaTeX}$ filer er kommentartegnet `%`, mens det for `Makefile` er en hash `#`.

På en slik kommentarlinje i en eller annen kildefil skriver du `$Id$` der du vil ha skrevet ut versjonsnummer, tidspunkt for siste editering og hvem som har editert fila, og du skriver `$Log$` der du vil ha skrevet ut loggen. `$Id$` vil du typisk ha på toppen av fila, legg f.eks. til følgende tre linjer på toppen av alle `.tex` filene dine:

```
%  
% $Id$  
%
```

Og så kan du f.eks. på slutten av fila si fra at du vil ha loggen der (du vil kanskje heller ha loggen på toppen av fila, da legger du til følgende der heller)

```
%  
% $Log$  
%
```

Når du skriver `cv`s `commit` vil CVS oppdatere disse feltene med ny informasjon.

A.10 Få mail hver gang noen gjør en commit

CVS kan konfigureres slik at det sendes en mail hver gang noen tar en commit, mailen vil inneholde hvilke filer som ble endret og loggmeldingen den som gjorde commit skrev. Dette settes typisk opp av gruppeleder, den som har repository på sin hjemmekatalog.

Det første man må gjøre er å hente ned en spesiell modul i CVS som heter `CVSROOT`, denne inneholder noen konfigurasjonsfiler som gjelder for det repository man henter den ut i fra. Modulen lages når man skriver `cv`s `init`. For å hente ut denne modulen, gjør man akkurat som når man henter ut et hvilket som helst annet prosjekt.

```
syLOW:~:$ cvs checkout CVSROOT
```

Gå så inn i denne katalogen, og finn fila `loginfo`. Denne kan du åpne i `emacs` og endre på. For prosjektet “linmet” vil du typisk legge til en linje på slutten av fila slik:

```
linmet mail -s 'CVS commit: %s' olenils@stud.math.ntnu.no
```

Da vil alle cvs commit i linmet-prosjektet generere en mail som sendes til olenils. Hvis det skal sendes en mail til alle prosjektdeltagere anbefales det å gjøre det ved å opprette en mailadresse for gruppa, og siste linje loginfo kan se slik ut:

```
linmet mail -s 'CVS commit: %s' olenils+linmet@stud.math.ntnu.no
```

Da må man også si fra til mailsystemet på sylow om hva denne mailadressen betyr, det gjør man ved å lage en fil kalt .qmail-linmet som man legger på rota i hjemmekatalogen (skal ikke ligge i prosjektkatalogen eller noe annet sted). I denne fila legger du mailadressen til prosjektdeltagere på hver sin linje.

A.11 Videre lesning

“Man”-siden til cvs kan anbefales, den gir deg rask mer informasjon om de fleste forskjellige cvs-kommandoene. Du får tak i man-siden ved å skrive kommandoen `man cvs`. En fullstendig referanse og brukermanual er [1].

Tillegg B

Filtyper

Dette kapitlet gir en referanseoversikt over hvilke filtyper du trolig kommer til å komme borti i løpet av prosessen.

Lista under er indeksert etter hvilken filendelse filene har.

B.1 $\text{\LaTeX}$ kildefiler

`tex` Tekstfiler som du skriver med latex-kode i.

`bib` Referanseliste/bibliografi skrevet i spesielt bibtex-format. Beskrevet i kapittel 9.

B.2 Filer generert av $\text{\LaTeX}$ og venner

`dvi` Device-independent. Et format for å kunne vise filer for utskrift uavhengig av plattform. Kan vises på skjerm med `xdvi` eller skrives ut eller omgjøres til postscript med `dvips`.

`ps` Postscript. En tekstfil skrevet i programmeringsspråket postscript. Et språk for å instruere printere til hvor og hvordan de skal skrive på sidene vi vil skrive ut. Kan skrives ut med `lpr` eller vises på skjerm med `gv`.

`pdf` Portable Document Format. Et filformat fra Adobe som forhåpentligvis kan bli verdensstandard for distribuering av dokumenter elektronisk.

`aux` Auxiliary. En fil $\text{\LaTeX}$ genererer for hver `tex` fil den prosesserer. Inneholder direktiver for nummerering og innholdsliste m.m.

`log` Inneholder logg av hva $\text{\LaTeX}$ har foretatt seg på en fil under kompilering

~ Backup-fil generert av emacs. Når du lagrer en fil i emacs, flytter emacs den gamle fila til det samme filnavnet pluss endelsen `~`

- bbl Generert av `bibtex` (se kapittel 9) slik at $\TeX$ kan putte innholdet med i rapporten for å generere referanseliste ved neste gangs kompilering.
- blg Loggfil fra siste kjøring av `bibtex`.
- idx For å lage indeks/register, generert av `latex`
- ind For å lage indeks/register, generert av `makeindex` for å puttes inn i rapporten ved neste gangs `latex`-kompilering.
- ilg Loggfil fra siste gangs `makeindex`-kjøring.
- toc Table of Contents. Innholdslista.
- lof List of Figures. Hvis du har tatt med kommandoen `listoffigures` i `master.tex` blir denne generert og du får med ei figurliste i rapporten.
- lot List of Tables. Hvis du har tatt med kommandoen `listoftables` i `master.tex` blir denne generert og du får med ei tabelliste i rapporten.

B.3 Grafikkfiler

- fig Tekstfil med koordinater for hvor det går streker hen. Brukes av programmet `xfig`.
- eps Encapsulated Postscript. Variant av postscript-programmeringsspråket laget for å være småbiter av postscript-kode som man putter inn i et stort postscript-dokument. Derfor velegnet for å inkludere grafikk inn i rapporter som til slutt skal bli postscript.
- jpg Komprimerende format for bilder (bitmap). Dette er et format som brukes i stor utstrekning for bilder. Må *ikke* brukes på bilder med høy kontrast (strektegninger) eller med tekst. Dette er det mange som ikke har forstått.
- gif Tapsløs komprimering for bilder (bitmap). Standard på web ved siden av jpg. Brukes på figurer med høy kontrast (som regel burde du hatt disse i vektorformat, men høyoppløselig gif går tålig bra). Komprimeringsalgoritmen som brukes er heftet med snodige lisensbetingelser som gjør at formatet egentlig burde vært boikottet. Lisensbetingelsene rammer alle programmer som genererer gif-bilder.
- png Grafikkformat designet for å bytte ut gif som standard på web ved siden av jpg. Formatet er helt gratis og burde derfor vært mye mer brukt.
- pnm Et format som selv kaller seg “portable anymap file format”. Standard output-format for scanneprogrammet `Xsane` på datasalen.

B.4 Andre filer

- `tar` Mange filer som er blitt samlet til en fil. Filene er ikke pakket i den forstand at de tar mindre plass, de er bare samlet til en fil. Pakk opp filen med `'tar xvf filnavn.tar'`. For å lage en pakke av en hel katalog (f.eks. backup av prosjektet ditt) kan du skrive `'tar xvzf prosjektet.tar.gz katalognavn'` der z-bokstaven betyr at det også skal pakkes slik at det tar mindre plass (se under).
- `gz` En fil som har blitt pakket med programmet `'gzip filnavn'` slik at fila tar mindre plass. Pakk opp igjen med `'gunzip filnavn.gz'`.
- `tar.gz` Mange filer/kataloger som først har blitt samlet til en tar-fil, så har det blitt pakket med gzip. Pakk opp igjen med `'tar xvzf filnavn.tar.gz'`. Du kan også først pakke opp med `'gunzip filnavn.tar.gz'` og så bruke `'tar xvf filnavn.tar'` men det er en kommando mer.
- `bz2` Et annet pakkeformat som er litt bedre enn gz. For å pakke en fil bruker du `'bzip2 filnavn'` og for å pakke opp bruker du `'bunzip2 filnavn.bz2'`. Dette er brukt i eksempel-Makefile fra kapittel 11.