

## Vildere L<sup>A</sup>T<sub>E</sub>X!

Tips, tricks, gode råd og noget om store projekter

Steffen Videbæk Petersen latex@spet.dk

Mat/Fys StudenterRåd – MFSR

DK-TUG

16. februar 2011

1 / 63

## Del I

### HverdagsL<sup>A</sup>T<sub>E</sub>X

2 / 63

## Første halvdel: HverdagsL<sup>A</sup>T<sub>E</sub>X

### 1 Valg af editor

- Hvad en god editor kan gøre for dig
- Mine forslag til dig
- Valg af PDF-fremviser

### 2 Hvad du måske allerede ved

- L<sup>A</sup>T<sub>E</sub>X på den „rigtige“ måde
- Dokumentklasser og uundværlige pakker

### 3 Nye makroer og environments

- Hvorfor bruge tid på det?
- Egne makroer: Sådan

3 / 63

## Hvad betyder valget?

### En rigtig god editor vil

- give færre dumme fejl, fx manglende }
- give færre anslag på tastaturet
- give bedre overblik over din kode
- give færre referencefejl
- gøre det enklere at finde dine fejl

### Konklusion

En god editor vil spare dig tid og frustrationer. Den vil gøre det nemmere og mere behageligt at skrive dokumenter i L<sup>A</sup>T<sub>E</sub>X.

4 / 63

## Den gode editor *skal* kunne

- Syntaks-fremhævning
- Syntax completion
- Liste over labels
- Understøtte dokumenter delt i flere filer
- Compile og trække fejlene ud af loggen

5 / 63

## Derudover *bør* den kunne

- Outline dit dokument
- Understøtte PDF-synkronisering (SyncTeX)
- Tilbyde stavekontrol
- Have skabeloner for indsættelse af figurer
- Hjælpe med at lave tabeller
- Lave genveje til ofte brugte makroer og environments

6 / 63

## Men fremfor alt

**Du skal kunne bruge den effektivt!**

### Råd 1: Find den rette ditor

Brug den tid, det tager at finde den rigtige editor. Brug også tid på at lære den at kende og tid på at indstille den efter dine behov.

Tiden du bruger her, sparer du tifoldigt!

7 / 63

## For alle: T<sub>E</sub>Xmaker

### Fordele

- Kan det, den skal
- Kan det meste af det, den bør
- Overskueligt interface
- Enkel opsætning
- Virker på både Windows, Mac og Linux

### Ulemper

- Begrænsede skabeloner
- Kender ikke fx beamer- eller memoir-makroer
- Dansk stavekontrol er langt fra perfekt

8 / 63

## For de modigste: emacs eller vim

### Fordele

- Kan det hele – *alt* – også beamer- og memoir-makroer
- Uovertrufne udvidelsesmuligheder
- Uovertrufne konfigurationsmuligheder
- Kan bruges til meget andet end  $\LaTeX$
- Virker på både Windows, Mac og Linux

### Ulemper

- Stejl indlæringskurve
- Besværlig opsætning, specielt af SyncTeX og stavekontrol

9 / 63

## Glem ikke at vælge PDF-fremviser

### Undgå Adobe Reader

- Langsom til at vise filerne
- Låser din PDF-fil
- Besværliggør opsætning af editor
- Kan ikke sættes op til SyncTeX

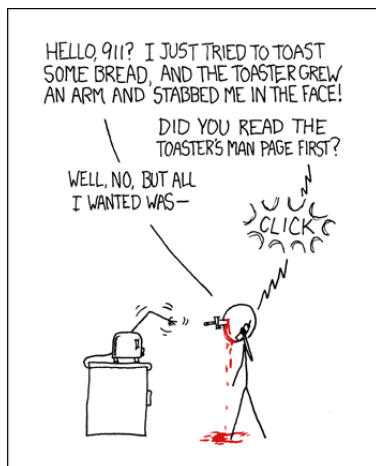
### Råd 2: Brug fx Sumatra PDF

- Låser ikke din PDF-fil
- Overvåger din PDF-fil og opdaterer visningen

$\TeX$ maker indeholder sin egen fremviser, og den er ret god

10 / 63

## $\LaTeX$ er ikke en brødrister, men ...



- $\LaTeX$  er et kraftfuldt værktøj. Modsat power tools i den virkelige verden risikerer du dog ikke at miste lemmer, hvis du bruger det forkert.
  - Der er god og grundig dokumentation for det meste, specielt for de fleste hverdagspakker.
- ← XKCD 293: RTFM.

11 / 63

## Undgå at blive stukket i ansigtet med en kniv

### Råd 3: Lær $\LaTeX$ at kende

10 minutter brugt på at læse dokumentation er oftest en investering og sjældent spildt. Minutterne kan spare dig for timers arbejde og/eller frustrationer senere hen.

### Råd 4: Kend din dokumentation

Nogle manualer, fx den til memoir-klassen, er meget omfattende. Hvis du mangler en overspringshandling, så bladr igennem den. Det er en gode måde at lære noget på.

### Råd 5: Dokumenter din egen kode

Når du laver en ny makro, ændrer marginer, tilføjer egne sidehoveder og -fødder, så dokumenter koden. Hvis ikke det virker, som det skal, kan du eller en  $\TeX$ pert se, hvad idéen var. Hvis andre får din kode kan de se, hvordan det skal bruges.

12 / 63

## Målet helliger midlet

### Der er mere end én måde

Der findes meget ofte mere end én måde at gøre tingene på, og nogle vil være mere „rigtige“ end andre. Du kan måske, i fremtiden, drage fordel af at gøre det „rigtigt“ allerede nu, men er tiden til det?

### Råd 6: Hacks er ok!

Hvis du er tilfreds med resultatet, så ignorer T<sub>E</sub>Xperter, der prædiker den „rigtige“ løsning (også mig!).

### Råd 7: ... oftest

Den „rigtige“ løsning kan spare dig tid senere, så måske du skulle spørge T<sub>E</sub>Xperter igen, når du har tid?

13 / 63

## Der er en nemmere måde

### Hvis du tænker

- Det kan da ikke passe, at jeg skal gøre det *hver gang*?
- Hvorfor skal jeg copy/paste så meget kode?
- Det her kode har gigant-SARS! Aaaaarg!

### Råd 8: Spørg en T<sub>E</sub>Xpert til råds

Der kan meget ofte findes en nemmere måde. Find ud af, hvad det er du gentager igen og igen og igen eller hvad du synes er rigtig grim kode. På nettet eller på uni findes en T<sub>E</sub>Xpert, der har gjort det før, eller som er villig til at hjælpe dig med at finde den nemmere måde.

14 / 63

## Dokumentklasse til print

### Råd 9: Brug **memoir** til print-dokumenter

Når først du har lært at bruge **memoir** er der ikke behov for at bruge andre dokumentklasser.

### Hvad **memoir** kan

- Mange indbyggede pakker, ofte med udvidet funktionalitet
- Mange ting er enklere, fx ændring af sidehoveder og -fodder
- Ændring af virkemåde er enklere, fx nummerering af fodnoter
- Omfattende dokumentation
- Let at få yderligere hjælp

15 / 63

## Dokumentklasse til slides

### Råd 10: Brug **beamer** til slides

Disse slides er lavet i **beamer**. Koden kan findes på nettet og viser, hvor simpelt det er.

### Slides er et foredrag for sig

At lave slides i L<sup>A</sup>T<sub>E</sub>X er ikke svært; med **beamer**-klassen er det ret enkelt. Er der interesse for et foredrag herom?

16 / 63

Pakken **microtype**

Vi giver en opsummering af arbejdet i det forløbne år og fortæller om vores planer for næste år, hvor vi selvfølgelig også modtager dine gode ideer. Har du fx ideer til, hvordan det sociale eller fysiske studiemiljø kan forbedres eller hvordan undervisningen skal udvikles, så har du nu muligheden for at få en finger med i spillet. Der bliver lyttet til MFSR, og vi har repræsentanter i mange råd og nævn, så hvis du har gode ideer, er der god mulighed for, igennem MFSR, at få gjort noget ved dem og få forbedret dit eget studie.

Uden **microtype**Råd 11: Brug **microtype**

Pakken sørger for at mikrojustere afstanden mellem bogstaver og kan placere tegnsætning (delvist) i marginen. Det kan spare dig for orddeliger. Er særdeles anvendelig i flerspaltede dokumenter.

Virker kun i final-mode, så i din preamble skal du skrive `\usepackage[final]{microtype}`.

Vi giver en opsummering af arbejdet i det forløbne år og fortæller om vores planer for næste år, hvor vi selvfølgelig også modtager dine gode ideer. Har du fx ideer til, hvordan det sociale eller fysiske studiemiljø kan forbedres eller hvordan undervisningen skal udvikles, så har du nu muligheden for at få en finger med i spillet. Der bliver lyttet til MFSR, og vi har repræsentanter i mange råd og nævn, så hvis du har gode ideer, er der god mulighed for, igennem MFSR, at få gjort noget ved dem og få forbedret dit eget studie.

Med **microtype**

17 / 63

Pakken **graphicx**Råd 12: Brug pakken **graphicx** og læs dens dokumentation

Pakken tillader dig at indsætte billedfiler i dine dokumenter. Der er dog mange konfigurationsmuligheder, som de færreste kender til.

## Global konfiguration (i preamble)

- Led efter billedfiler i specifikke mapper:  
`\graphicspath{{figurer/}{spektre/}}`
- Sæt bredde for alle inkluderede billeder:  
`\setkeys{Gin}{width=0.8\textwidth}`

18 / 63

## ams-pakkerne

Råd 13: Inkluder altid **ams**-pakkerne

Pakkerne **amsmath**, **amssymb** og **amsfonts** giver dig mulighed for at skrive matematiske udtryk – noget de færreste kan være foruden.

## Dokumentation vs. daleif

Dokumentationen for **ams**-pakkerne er omfattende, men *ikke* brugervenlig. Kig istedet daleifs bog igennem for eksempler på crazy matematik. Hvis du vil gøre det er det med garanti gjort før.

19 / 63

Pakken **mathtools**

$$V = \sum_{1 \leq i \leq j \leq n} v_{ij}$$

```
\[
V = \sum_{
  1 \leq i \leq j \leq n
} v_{ij}
\]
```

$$V = \sum_{1 \leq i \leq j \leq n} v_{ij}$$

```
\[
V = \sum_{\mathclap{
  1 \leq i \leq j \leq n
}} v_{ij}
\]
```

Råd 14: Brug **mathtools**

**mathtools** tilføjer nogle muligheder, som ikke findes i **ams**-pakkerne. I mange tilfælde er lige det man mangler og mange af eksemplerne på avanceret matematik i daleifs bog bruger pakken. Manualen er ganske god og fyldt med eksempler.

20 / 63

Pakken **siunitx**Råd 15: Brug **siunitx**

Alle, der bruger tal og/eller enheder vil drage fordel **siunitx**, der gør det nemmere at opskrive og enheder på en konsistent måde. Derudover tilføjer den nogle uundværlige værktøjer for datatabeller.

## Eksempel

- $\$g=\SI{9.82}{\kilo\gram\per\second\squared}\$$   
 $g = 9.82 \text{ kg s}^{-2}$
- $\SI{3,23}{\kg.J^{-1}}.\angstrom$   
 $3.23 \text{ kg J}^{-1} \text{ \AA}$
- $\SI[per-mode=fraction]{1e-34}{\joule\per\mole\per\kelvin}$   
 $1 \times 10^{-34} \frac{\text{J}}{\text{molK}}$
- $\num[separate-uncertainty]{1.34(5)e4}$   
 $(1.34 \pm 0.05) \times 10^4$

21 / 63

## Målet med egne makroer

Målet med egne makroer og environments er

- at spare tid. Færre tastaturtryk giver et mindre tidsforbrug: `\pcc` er hurtigere end `photocatalyst coating`
- at give mere overskuelig kode. `\inv` er nemmere at læse (og taste) end `^-1`.
- at givet et konsistent udseende. Egen notation i egne makroer giver mulighed for at ændre notation for hele dokumentet ved at ændre én linje i preamblen.

22 / 63

## Hvad koster det mig

Du sparer tid, men først efter du har investeret den i

- at tænke over, hvor du kan spare tiden
- at lære værktøjerne at kende
- kodning og debugging
- dokumentation

## Råd 16: Brug egne makroer

Det koster indledningsvis noget tid, men den får du igen i form af færre tastaturtryk og mere overskuelig kode.

23 / 63

## Midlerne

Der er mange muligheder for at skrive egne makroer:

**Simple** `\newcommand` med eller uden argumenter

**Betingelser** Pakkerne **ifthen** og **xifthen**

**Avancerede** Pakken **etoolbox**

## Der er håb

Syntaksen for disse makroer og pakker er ganske ligetil. Det kræver ikke lang tid og utallige forsøg at lave de makroer, som de fleste skal bruge.

24 / 63

Makroen `\newcommand`Syntaksen for `\newcommand`

Uden argumenter er `\newcommand` en meget enkel makro:

```
\newcommand\nymakro{<handling>}
```

Eksempel (`\pcc`)

```
\newcommand\pcc{photocatalyst coating\space}
```

Råd 17: Brug pakken `xspace`

Makroen `\xspace` fra pakken `xspace` indsætter automatisk et mellemrum, hvis det skal være der. Dvs. hvis det næste er et ord men ikke hvis det er tegnsætning.

25 / 63

## Overskuelighed og nye makroer

Eksempler på, at makroer kan gøre koden mere læsbar og overskuelig:

- `\newcommand{\isomorphic}{\cong}`
- `\newcommand{\DistAs}{\sim}`
- `\newcommand{\ObsFromDist}{\mathrel{\sim\sim}}`

## Råd 18: Læsbarhed vinder over antal anslag

Læsbar kode er enkel at debugge. Det er derfor en væsentlig fordel, at koden er læsbar. `\isomorph` er længere end `\cong`, men øger læsbarheden og er derfor en god makro.

26 / 63

`\newcommand` med argumenterSyntaksen for `\newcommand`

Med argumenter er det ikke meget sværere:

```
\newcommand\nymakro[<antal args>]{<handling>}
```

I handlingen refereres til argumenter med `#n`.

## Eksempel (Partielt afledte)

Et  $\partial$  laves med `\partial`, så partielt afledte er trælse at skrive, fx kræves `\frac{\partial f}{\partial x}` for at få  $\frac{\partial f}{\partial x}$ . Med `\newcommand\pdiff[2]{\frac{\partial #1}{\partial #2}}` kan det nu laves med `\pdiff{f}{x}`.

27 / 63

## Eksempler på konsistent notation

Konsistent notation, med mulighed for global ændring, er fx indre produkt, norm og absolutværdi:

```
\newcommand{\inner}[2]{\langle #1,#2\rangle}
%\newcommand{\inner}[2]{(#1; #2)}
\newcommand{\norm}[1]{\lVert #1\rVert}
%\newcommand{\norm}[1]{\lvert #1\rvert}
\newcommand{\abs}[1]{\lvert #1\rvert}
```

## Råd 19: Brug dine makroer konsekvent

Hele idéen bag at bruge makroer til notation forsvinder, hvis ikke man bruger dem hver gang. Hvis du laver en makro for fx indre produkt skal du også bruge den hver gang du opskriver et indre produkt.

28 / 63

## \newcommand med valgfri argumenter

### Syntaks

```
\newcommand\nymakro[⟨args⟩][⟨default⟩]{⟨handling⟩}
```

Der kan kun angives ét valgfrit argument, og det er det første.

### Eksempel (Valgfri argumenter)

Ved at ændre definitionen af \pdiff til:

```
\newcommand\pdiff[3][\partial]{\frac{#1 #2}{#1 #3}}
```

Som før virker `\pdiff{f}{x}` men nu vil `\pdiff[d]{f}{x}` give  $\frac{df}{dx}$ .

### Råd 20: Valgfri argumenter giver fleksibilitet

Man kan opnå større fleksibilitet med valgfri argumenter. Pakken `etoolbox` giver modsat `\newcommand` mulighed for mere end ét valgfrit argument.

## Makroen \newenvironment

### Syntaks

Meget tæt opad `\newcommand`, men tager en start- og en sluthandling:

```
\newenvironment{⟨env⟩}{⟨start⟩}{⟨slut⟩}
```

```
\newenvironment{⟨env⟩}[⟨args⟩]{⟨start⟩}{⟨slut⟩}
```

```
\newenvironment{⟨env⟩}[⟨args⟩][⟨default⟩]{⟨start⟩}{⟨slut⟩}
```

Pas på: Du kan ikke brug argumenterne i sluthandlingen. Løsningen er at gemme dem fra starthandlingen og så bruge det gemte i sluthandlingen.

## Eget citerings-environment

Mål: At lave et environment, hvor citeringer indsættes. Det skal skille sig ud fra resten af teksten og skrive hvor man citerer fra:

```
% Environmentet myquote, der tager et argument: Hvem der
% citeres fra. Dette sættes i fed paa en centeret linje
% over selve citatet, der indrykkes fra begge marigner og
% goeres slanted (dvs. skraa, men ikke kursiv).
```

```
\newenvironment{myquote}[1]
{\centerline{\bfseries Citat fra #1}\begin{quote}\slshape}
{\end{quote}}
```

### Råd 21: Environments er sære

I og omkring environments opfører afstande sig sært, lidt ukendt. Lad være med at tænke over det i starten. Hvis du synes det ser grimt ud, så snak med en T<sub>E</sub>Xpert om det.

## (Næsten) kun fantasien sætter grænser

Med list, snedighed og brug af forskellige pakker kan man næsten alt:

- Lave rammer omkring stort set alt, også andet end bare lige linjer og måske du kun vil have i top og bund?
- Ændre baggrunds- og tekstfarver
- Behandle samme argument som både tekst og makro
- Automatisere fx tabelgeneration
- ...

### Råd 22: Drøm og spørg til råds

Der er intet i vejen for at ønske noget bestemt. I de fleste tilfælde kan det også lade sig gøre. T<sub>E</sub>Xperterne hjælper gerne men prædiker måske lidt (undskyld).

## Del II

## Store projekter

33 / 63

## Anden halvdel: Store projekter

- 4 Dokumenter i flere filer
  - Opdelingen
  - Inkludering af filer
  - Selektiv inkludering
- 5 BibT<sub>E</sub>X
  - Hvad det er
  - Inklusion i dokument
- 6 Mit forslag til en skriveproces
  - Form vs. indhold
  - Gode værktøjer
  - Undervejs
  - Til sidst

34 / 63

## En fil, to filer, mange filer

L<sup>A</sup>T<sub>E</sub>X kan inkludere indholdet af andre filer. Det kan bruges til at skabe

- mere overskuelighed
- enklere debugging
- kortere compile time

Nogle pakker tillader, at man henter input fra andre filer, fx **listings**.

35 / 63

## Hvornår skal man opdele

Hvornår skal man dele noget mellem flere filer?

- Preamblen i sin egen fil
- Hvert kapitel i sin fil
- Uoverskuelige tabeller/figurer i egne filer

## Din master-fil

Mange editorer bruger udtrykket master-fil. Det er den fil, der kalder `\include` for de andre filer. Det er en god idé, at din preamble er din masterfil og alt indhold i andre filer.

## Råd 23: Mådehold

Man kan lave filer for hvert afsnit, eller hver subsection eller hver tabel etc. På et tidspunkt bliver det mindre overskueligt, så overvej hvor du deler.

36 / 63

Makroen `\input`

Makroen `\input` virker som copy/paste. Den er identisk med at kopiere indholdet fra den `input`'ede fil ind, hvor der står `\input` i koden.

## Syntaks

```
\input{<filnavn>}
Filnavnet behøver ikke slutte på .tex.
```

Råd 24: `\input` hvornår?

Brug `\input` til fx tabeller, figurer eller andet, der skal skjules i overskuelighedens navn.

Makroen `\include`

Makroen `\include` indsætter indholdet af den fil, den kalder.

## Syntaks

```
\include{<filnavn>}
Filnavnet behøver ikke slutte på .tex.
```

`\include` starter på en ny side.

Råd 25: `\include` hvornår?

Brug `include` når du alligevel skal starte på en ny side. Skriver du i **memoir** vil det sige: en fil per `\chapter`.

## Inkludering af kode

Pakken **listings** giver mulighed for at inkludere hele kodefiler, eller dele deraf.

```
\lstinputlisting[language=C++,firstline=7,lastline=18]
{isPrime.cpp}
```

```
1 bool isPrime (int n) {
2   // A primitive trial division
3   if (n <= 1)
4     return false;
5   if (n % 2 == 0)
6     return (n == 2);
7   for (int d = 3; d <= sqrt(n); d++) {
8     if (n % d == 0)
9       return false;
10  }
11  return true;
12 }
```

## Indsættelse af data i tabeller I

En meget effektiv brug af `\input` er til at slippe for at copy/paste. Hvis man fx vil inkludere en datafil kunne man:

```
\begin{table}[b]
 \centering
 \begin{tabular}{r r r}
 \toprule
 Vinkel & Kanal 1 & Kanal 2 \\
 \midrule
 \input{data.dat}
 \bottomrule
 \end{tabular}
 \caption{Data fra program}
 \label{tab:data}
 \end{table}
```

Dette virker kun, hvis dataene i `data.dat` er i L<sup>A</sup>T<sub>E</sub>X's tabelformat.

## Indsættelse af data i tabeller II

Hvis dine data er i en CSV fil og derfor skal skrives om skal der bruge hjælp:

### Råd 26: Brug pakken `datatool`

Pakken `datatool` kan hente data fra eksterne filer og formatere dem anderledes. Den kan også hente specifikke søjler eller rækker.

### Pakken `siunitx`

Hvis man kombinerer `datatool` med `siunitx` kan man lave vilde ting. Fx kan man hente data fra en ekstern fil, der ikke er i  $\LaTeX$ -format og endda omskrive  $1.03e3$  til  $1.03 \times 10^3$ .

41 / 63

## Spar noget compile tid væk

Makroen `\includeonly` benyttes i stedet for at kommentere de enkelte `\include`-statement ud. Den

- Inkluderer kun de nævnte filer
- Benyttes i preamblen
- Beholder referencerne til de filer, der ikke compiles

```
\includeonly{intro,eksperiment}
...
\begin{document}
\include{intro}
\include{eksperiment}
\include{resultater}
\include{litteratur}
...
```

42 / 63

## Effektiv brug af `\includeonly`

`\includeonly` er smart, men virker kun for filer inkluderet med `\include`. Filer inkluderet med `\input` indsættes altid.

### Råd 27: Brug af `\includeonly`

Brug `\includeonly` til minimere den tid, du bruger på at compile. Ved kun at inkludere de filer, de laver ændringer i, bliver processen lidt mere flydende. Idet `\includeonly` husker labels fra de ikke-inkluderede filer minimerer man også referencefejlene på tværs af kapitler.

43 / 63

## Hvad er BibTeX

BibTeX er

- Et program, der lever uafhængigt af  $\LaTeX$
- En database over det litteratur, du har læst
- Indeholder alt information om alle værkerne

BibTeX kan

- Give en konsistent visning af referencer til det, du citerer
- Styles, så den viser fx [1] eller [Jackson, 1998]
- Opsættes til at vise alle værker eller kun dem, du citerer

44 / 63

## Indgange til databasen

- Man kan ikke generere en bibliografi uden at have noget at citere. Man skal derfor opbygge en fil med værkerne. Filen skal have endelsen `.bib`, fx `spet.bib`
- Man kan skrive informationerne i hånden, men man kan også finde informationerne på nettet. De fleste naturvidenskabelige artikeldatabaser giver mulighed for at downloade en `.bib`-fil tilhørende hver artikel

### Råd 28: Byg din `.bib`-fil fra dag ét

At skulle lave citeringer som noget af det sidste er besværligt. Hvor var det lige man læste hvad henne? Derfor er det en god idé at opbygge sin `.bib`-fil fra starten og citere løbende.

## Eksempel på en `.bib`-fil

Et uddrag fra Rasmus Villemoes' `thesis.bib`.

```
@article {MRS79573,
  AUTHOR = {Hatcher, A. and Thurston, W.},
  TITLE = {A presentation for the mapping class group of a closed
orientable surface},
  JOURNAL = {Topology},
  FJOURNAL = {Topology. An International Journal of Mathematics},
  VOLUME = {19},
  YEAR = {1980},
  NUMBER = {3},
  PAGES = {221--237},
  ISSN = {0040-9383},
  CODEN = {TPLGAF},
  MRCLASS = {57N05},
  MRNUMBER = {MRS79573 (81k:57008)},
  MRREVIEWER = {Frank Quinn},
}
@phdthesis {ARS2006,
  AUTHOR = {Skovborg, Anders Reiter},
  TITLE = {The Moduli Space of Flat Connections on a Surface -- {Poisson} Structures and Quantization},
  YEAR = 2006,
  SCHOOL = {University of Aarhus},
  NOTE = {\url{http://www.imf.au.dk/publs?id=623}},
}
```

Her er `MR579573` og `ARS2006` citeringsnøgler.

## Hvordan bruges BibTeX

- Citationer laves i teksten med `\cite{<nøgle>}`, fx

```
\cite{MR579573}
\cite{MR579573,ARS2006}
```

- I ens dokument skriver man

```
\bibliographystyle {alpha}
\bibliography{../minfil}
```

der hvor bibliografien skal indsættes.

- For at generere bibliografien skal man køre kommandoen

```
bibtex master
```

- Næste gang du compiler `master.tex` indsættes bibliografien for de elementer, du har citeret.

## Nævnelse af ikke-citerede værker

Som standard inkluderer BibTeX udelukkende de værker, der nævnes med en `\cite`. Det er der dog råd for:

### Råd 29: Ikke-citerede værker

Man kan bruge `\nocite{<nøgle>}` til at få inkluderet netop ét værk. Bruger man istedet `\nocite{*}` får man alle værkerne i ens `.bib`-fil nævnt i bibliografien.

## Skriv!

### Råd 30: Skriv!

Der skal nok komme tidspunkter, hvor du sidder med en skriveblokering. Skriv, når du kan. Så kan du bruge blokeringsperioder til at nørde  $\LaTeX$  og lege med layoutet.

### Råd 31: Udkommenter det, der ikke virker

Ved at udkommentere kan du bruge tid på at skrive fremfor på at løse  $\LaTeX$ niske problemer. Problemerne kan du altid vende tilbage til og spørge en  $\TeX$ pert om senere.

49 / 63

## Skabelon eller ej?

Der er fordele ved at bruge en god skabelon

- Den sparer dig for meget tid i forhold til at sætte en preamble op fra bunden
- Man kan skele til de løsninger, der er brug i den, og bruge dem som de er eller som inspiration

Men der er også ulemper

- En udokumenteret skabelon kan spilde din tid
- En inkomplet eller uddateret skabelon kan være en hæmsko for hele skriveprocessen

50 / 63

## Når du vælger en skabelon

### Råd 32: Dokumentation

Kig efter dokumentation, særligt i preamble. Ingen dokumentation er som regel et tegn på, at den kan blive tung at danse med.

### Råd 33: Overskuelighed

Hvis preamble virker fuldstændig uoverskuelig er du nok også på vej ud i noget snavs. Hvis ikke der er dokumentation må du snakke med forfatteren og høre, hvad han/hun har lavet.

### Råd 34: Minimalisér skabelonen

Din første opgave bør være at pille alt, du ikke skal bruge, ud af skabelonen. Når du har gjort det skal du have den til at virke. Lav et eksempel der kan compile.

51 / 63

## Den gode skabelon

Den gode skabelon har nogle helt bestemte egenskaber

- Den er skrevet i **memoir**
- Den er veldokumenteret, både i preamble og dokument
- Preamble er velstruktureret
- Der er kontaktoplysninger til forfatteren, så man kan få hjælp hvis kommentarerne ikke er nok

### Råd 35: Brug Rasmus Villemoes' skabelon

Rasmus Villemoes er  $\TeX$ pert. Han brugte sin ph.d.-afhandling på at skrive en god og veldokumenteret skabelon. Den kan være overkill, men er som minimum et godt sted at hente inspiration.

<http://rasmusvillemoes.dk>, og download  $\LaTeX$ -koden fra hans afhandling.

52 / 63

Pakken **fixme**

**fixme** giver mulighed for

- at lave noter i margin
- at lave en liste over noter
- at  $\LaTeX$  giver en compile-fejl, hvis der er noter tilbage

**Råd 36: Brug **fixme****

Brug pakken **fixme** til at holde styr på de ting, der skal undersøges nærmere eller gøres noget ved senere. Det giver dig mulighed for at skrive nu og tage dig af fejlene senere.

53 / 63

Pakken **showkeys**

**showkeys** giver mulighed for

- at vise labels i din PDF-fil
- at få overblik over, hvor man refererer forkert
- at holde styr på labels, hvis ens editor ikke gør det

**Råd 37: Brug **showkeys****

Brug **showkeys** undervejs i skrive- og retteprocessen til at holde styr på labels og referencer. Den er et uvurderligt værktøj når man skal fejlfinde referencer.

54 / 63

Pakke **lipsum**

Hvis man eksperimenterer med layout er pakken **lipsum** fantastisk. Det kan indsætte noget tekst, der kan bruges som placeholder.

**Eksempel (`\lipsum[4]`)**

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Uden argument indsætter den ca. halvanden side. Med argument indsætter den et eller flere afsnit, fx [1-3] eller [1,5].

55 / 63

Snedigere referencer med pakken **varioref**

Pakken **varioref** giver mulighed for snedigere referencer. Hvor `\pageref{<label>}` skriver et sidetal for denne label, vil `\vref{<label>}` skrive fx:

- På foregående side
- På næste side
- Herunder

**Råd 38: Brug **varioref****

Hvis du har mange referencer og specielt hvis de er lokale, er det en fordel at bruge **varioref**. Dokumentet bliver noget nemmere at læse.

56 / 63

## Kodehygiejne

- Selv hvis du er den eneste, der skal læse din kode, så skal du kunne finde rundt
- `\begin` og `\end` på linjer for sig
- Mellumrum i matematik ignoreres, hvilket kan gøre koden mere læsbar. Fx:

```
$(a_1, b_1 + na_1, a_2, b_2, \ldots, a_g, b_g)$
$(a_1, b_1+na_1, a_2, b_2, \ldots, a_g, b_g)$
```

- Antallet af tomme linjer mellem afsnit er irrelevant, så lav gerne et par tomme linjer inden `\section` og lignende.

57 / 63

## Labels defineret to gange

Dette er en klassisk fejl og god grund til at bruge `\includeonly`

```
LaTeX Warning: Label 'eq:einstein' multiply defined.
...
LaTeX Warning: There were multiply-defined labels.
```

### Råd 39: Ret dobbeltlabels med det samme

Ret den med det samme, ellers kan du ikke huske hvilken ligning du egentlig ville referere til. For hver `\eqref{eq:einstein}` skal du finde ud af hvilken du egentlig mente.

58 / 63

## Labels, der ikke er definerede

En ligeså klassisk fejl er:

```
LaTeX Warning: Reference 'eq:32' on page 1 undefined on input line 11.
...
LaTeX Warning: There were undefined references.
```

### Råd 40: Ret ikke-definerede labels med det samme

Hvis fejlen ikke forsvinder ved næste compile, så har du henvist til en ligning der ikke findes. Ret mens du kan huske hvad du ville referere til.

59 / 63

## Fonte og marginer

Når indeholdet er færdigt skal følgende afgøres:

- Skrifttype.  $\LaTeX$ s standardfont, Computer Modern, er ikke eneste valg. Se fx [www.tug.dk/FontCatalogue](http://www.tug.dk/FontCatalogue).
- Skriftstørrelse. Typisk 10–12pt, men andre fonte kan se bedre ud i andre størrelser.
- Papirstørrelse. For de fleste A4, men der kan være krav alt efter hvad du skriver. Spørg din trykker/forlægger.
- Marginer. Brede marginer giver korte linjer, og det er med vilje. Lange linjer er sværere at læse end korte.

60 / 63

## Sidens layout

Indhold, fonte og marginer er ok. Næste skridt er layoutet:

- Kapitelforsider
- Environments, fx sætningskonstruktioner
- Udseendet af hjemmelavede makroer
- Tabeller, figurer og andre flydende elementer

### Råd 41: Layout og fremtoning

Sidens layout betyder rigtigt meget for hele projektets fremtoning. Hvad der er rigtigt for forfatteren af din skabelon er ikke nødvendigvis rigtigt for dig. Tag en snak med en T<sub>E</sub>Xpert, gerne et stykke tid inden deadline.

61 / 63

## Fejl og advarsler

Indhold og al layout er på plads. Nu skal det sikres at:

- Dokumentet skal kunne oversættes uden fejl. daleif har en liste over almindeligt forekommende fejlmeddelelser og hvad de betyder på dansk.
- Rent faktisk bør fejl rettes så snart de opstår; så har man lettere ved at finde den skyldige kodebid. Jo mere kode man fylder på, jo sværere er det at afkode L<sub>A</sub>T<sub>E</sub>Xs til tider kryptiske meddelelser.
- Advarsler skal man tage alvorligt. Nogle kan ignoreres, men det bør være et aktivt valg for hver enkelt.

### Råd 42: Sæt tid af til at rette fejl og advarsler

Specielt for store dokumenter kan det være lidt af en opgave, at rette fejl og advarsler. Hvis man bruger en ringe skabelon vil det også kræve ekstra tid. Sørg for at have sat tid af inden din deadline.

62 / 63

## Almindelige advarsler

De mest almindelige advarsler er (under/over)full (h/v)box:

### Underfull hbox

L<sub>A</sub>T<sub>E</sub>X strækker en linje mere end den bryder sig om. Er der et ord, den ikke kan dele eller har du måske brugt \\ til at dele en linje med (fy, skam dig!).

### Overfull hbox

Den mest almindelige advarsel af alle. L<sub>A</sub>T<sub>E</sub>X brokker sig over, at noget tekst stikker ud i marginen. Du hjælpe den med at foretage orddeling, omskrive teksten eller acceptere overskridelsen.

### Råd 43: Badboxes er ikke ligegyldige

Badboxes vidner om, at noget er galt. Det er ikke altid man helt kan lure, hvad det er L<sub>A</sub>T<sub>E</sub>X brokker sig over, men det skal være et valg at ignorere den enkelte badbox

63 / 63