Maarten Gelderman

TEXniques

# Toolbox: een syllabus

**abstract**
In deze toolbox wordt een voorbeeld besproken van een redelijk omvangrijk LaTeX-document. Het nut van de verschillende packages komt aan de orde en er wordt tevens besproken hoe de packages worden gebruikt in het document zelf en welke opties daarvoor in de preamble moeten worden opgenomen.

**keywords**
LaTeX, packages, praktijkvoorbeeld

Deze toolbox is weer gewoon in het Nederlands. Eigenlijk wel zo leesbaar en het schrijft nog sneller ook. Ook in een ander opzicht wijkt deze toolbox af van eerdere toolboxen. Ik beschrijf dit keer geen verzameling van op zichzelf staande trucs, maar loop gewoon eens door een document heen dat ik zelf onderhanden heb. Ik ben zelf altijd wel benieuwd hoe de documenten van anderen eruit zien en heb ervaren dat er een hoop LaTeX-tips en -trucs zijn die je in geen handboek leest maar alleen ziet wanneer je eens een document van een ander onder ogen krijgt. Die persoon zelf heeft meestal niet eens in de gaten dat hij jou op dat moment wat bijbrengt. Hij (of zij natuurlijk, maar de TeX wereld bestaat voor een wel erg groot deel uit mannen) doet het altijd zo en kan zich niet voorstellen dat niet de hele wereld op die manier werkt.

Ten gevolge van de gekozen opzet is deze toolbox verre van 'self-contained'. Een aantal zaken stip ik alleen maar aan. Ik geef bij voorbeeld slechts aan wat *ik* in mijn document met een package doe. De door mij gebruikte functionaliteit vormt slechts een deel van de functionaliteit die het package biedt. Gelukkig is de documentatie van de packages zelf over het algemeen zeer goed. Deze kan de lezer bij voorbeeld vinden op de TeXLive CD. Onder de directory `/texmf/doc/latex` van deze CD is voor de meeste genoemde packages uitgebreide informatie in DVI- of postscript-formaat te vinden. Daarnaast is en blijft het natuurlijk mogelijk om voorbeelden te kopiëren en aan te passen zonder ze echt te snappen. Mijn ervaring is zelfs dat dit bij het gebruik van LaTeX noodzakelijk is. Hoe alle LaTeX-code aan elkaar gekoppeld is, is voor een gewone, welwillende gebruiker niet te begrijpen. Door wat code uit een package te halen en te experimenteren met kleine wijzigingen kunnen vaak echter de gewenste effecten zonder al te veel problemen gerealiseerd worden. Een voorbeeld van dit type aanpassing is de weergave van de literatuurlijst, zoals die later in deze toolbox wordt besproken.

Het document aan de hand waarvan ik dit verhaal opzet is een syllabus voor het vak Methoden en Technieken van Bedrijfswetenschappelijk Onderzoek. Het document is ongeveer 250 pagina's lang. De omvang van het hoofddocument is bijna 900 kilobyte.[1] Daarnaast worden voor het aanmaken van tabellen, figuren en de literatuurlijst nog vele andere bestanden gebruikt. Voor het bijhouden van verschillende versies van de syllabus gebruik ik RCS. De preamble van het document is meer dan 300 regels lang. Deze extreme omvang is overigens voor een groot deel het gevolg van het feit dat ik nooit de moeite heb genomen om een aantal zaken eens netjes tot een package te verwerken of tenminste via een input-commando op te vragen.

1. Met emacs valt met een document van deze omvang nog prima te werken. Wie werkt met een andere ASCII-editor geeft er wellicht de voorkeur aan om een document van deze omvang in een aantal subdocumenten op te splitsen.

## Het begin van de preamble

In tegenstelling tot wat wellicht gebruikelijk is, begint het besproken bestand niet met het commando \documentclass, maar met de volgende if-constructie:

```
\newif\ifpdf
\ifx\pdfoutput\undefined
  \pdffalse
\else
  \pdfoutput=1
  \pdftrue
\fi
```

Hier wordt vastgesteld of ik de gewone TEX-executable, of pdfTEX gebruik, zodat ik later afhankelijk van het antwoord op deze vraag bij voorbeeld de juiste opties mee kan geven bij het laden van een package of de juiste plaatjes kan opvragen (terzijde zij opgemerkt dat de syllabus in zijn huidige vorm eigenlijk alleen onder een gewone TEX compileert). De reden om direct met deze if te beginnen en hem niet pas na het inladen van documentclass en packages op te nemen is dat sommige packages \pdfoutput ook definiëren als de gewone executable wordt gebruikt.

## Documentclass en gebruikte packages

Met het volgende commando wordt de documentclass gespecificeerd en wordt een aantal packages geladen.

```
\ifpdf
  \documentclass
    [pdftex,dutch,bf,small,twoside,notitlepage,round,sort&compress]
    {syllabus}
\else
  \documentclass
    [dvips,dutch,bf,small,twoside,notitlepage,round,sort&compress]
    {syllabus}
\fi
\usepackage{amsmath,natbib,babel,tabularx,url,fancyvrb,textcomp}
\usepackage{caption2,wrisym,relsize,mflogo,lijstmac}
\usepackage{answers,makeidx}
%\usepackage[T1]{fontenc}
%\usepackage[bookmarks=true,bookmarksopen=true]{hyperref}
```

De documentclass die ik gebruik is syllabus; een eigen aanpassing van de documentclass refart die ik netjes hernoemd heb om te zorgen dat documenten die er vanuit gaan dat een ongewijzigde refart beschikbaar is gewoon blijven compileren.

Aan de documentclass geef ik een hele set opties mee. Het merendeel van de opties zou ik ook aan de packages mee kunnen geven. Ik vind het echter efficiënter om ze achter \documentclass op te nemen. Opties die bij \documentclass staan worden automatisch aan alle relevante packages doorgegeven. Dit scheelt niet alleen typewerk, maar maakt een document ook gemakkelijker onderhoudbaar: een wijziging hoeft nog maar op één plaats te worden aangebracht.

De meegegeven opties zelf zijn over het algemeen slechts voor één enkel package bestemd. Als eerste optie wordt, afhankelijk van de eerder gedefinieerde if, aan alle packages meegegeven of pdftex of dvips wordt gebruikt. Dit is bij voorbeeld van belang voor het

aanmaken van specials[2] voor afbeeldingen (iets waar het package graphicx zich mee bezig houdt) het aanmaken van specials voor hypertext-functies (iets waar hyperref zorg voor draagt). Doordat de optie bij \documentclass is meegegeven wordt deze automatisch doorgegeven aan alle relevante packages. Als ik bij voorbeeld het package geometry zou toevoegen om het pagina-formaat te wijzigen, zou deze ook automatisch de juiste optie meekrijgen.

De tweede optie (dutch) geeft aan dat we te maken hebben met een Nederlandstalig document. Met behulp van babel worden zo de juiste namen voor afbeeldingen, hoofdstukken enzovoorts geladen en in principe kan ook natbib deze informatie gebruiken om biografische verwijzingen en de literatuurlijst op correcte wijze weer te geven.

De opties bf en small worden allebei gebruikt door het package caption2. Het genoemde package dient om de weergave van bijschriften bij figuren en tabellen aan te passen. Ik geef er de voorkeur aan om de woorden 'figuur 1' vet te zetten en het hele bijschrift in een iets kleiner lettertype te zetten dan de rest van het document en dat is precies wat deze twee opties doen.

De optie twoside geeft aan dat het document dubbelzijdig geprint gaat worden (bij mijn weten heeft dit in dit document alleen invloed op de weergave van de kop- en voetteksten). Ook notitlepage zal bij de meeste lezers wel bekend zijn. Op deze manier wordt aangegeven dat TEX geen titelpagina moet maken. Deze optie is overigens wel een typisch voorbeeld van verkeerd toegepaste markup. De optie suggereert dat er geen titelpagina is. Deze is er wel degelijk, maar wordt in het document met de hand aangemaakt. Als TEX zelf nu ook nog een titelpagina toe zou voegen zouden er opeens twee zijn en dat is niet de bedoeling. Het zou natuurlijk veel netter zijn de weergave van de titelpagina in de documentclass syllabus zo aan te passen dat de automatisch aangemaakte titelpagina correct is.

De opties round en sort&compress worden allebei gebruikt door het package natbib. De combinatie van dit package en deze opties zorgt ervoor dat literatuurverwijzingen volgens het auteur-jaartal systeem worden weergegeven (dus in de vorm Jansen, 1999), dat de haakjes om literatuurverwijzingen rond zijn (standaard gebruikt natbib blokhaken) en dat verwijzingen gesorteerd en en waar mogelijk gecomprimeerd worden. Op deze manier wordt de verwijzing (Jansen, 1999c; Jansen 1999a; Jansen 1999b; Jansen 1998) automatisch vervangen door (Jansen, 1998, 1999a,b,c) hetgeen ruimte bespaart en wel zo leesbaar is.

Met een aantal \usepackage-regels worden vervolgens de packages zelf geladen. Ik ga ze weer even stuk voor stuk langs. Allereerst is er amsmath. Het LATEX-package van de American Mathematical Society. Dit package laadt een aantal commando's die het ingeven van en verwijzen naar formules gemakkelijk maken. Met name voor formules die meerdere regels beslaan is dit van belang. Ook definieert dit package een aantal commando's die het mogelijk maken formules netter te definiëren. Met standaard LATEX bestaat een formule veelal uit een set symbolen. Het AMS-package maakt het mogelijk om in plaats van naar de symbolen naar de betekenis van die symbolen te verwijzen en om gemakkelijk nieuwe functies te definiëren. De volgende formule

$$y_{ij} = \begin{cases} 0 & \text{if the expert did not mention the intended variable} \\ \frac{1}{q} & \text{if the expert mentioned } q \text{ variables, including the intended one} \\ 1 & \text{if the expert only mentioned the intended variable} \end{cases}$$

is ingegeven met de volgende standaard-LATEX-commando's

2. Special is de TEX-term voor informatie die 'letterlijk' in het DVI-bestand wordt opgenomen en door de DVI-driver gebruikt kan worden voor het aansturen van bij voorbeeld de printer. Specials zijn enerzijds de kracht van TEX (zij zorgen voor uitbreidbaarheid) en anderzijds de zwakte (compatibiliteit is door de grote vrijheid bijna niet te waarborgen).

```
\begin{displaymath}
  y_{ij}= \left\{
    \begin{array}{ll}
      0 &\mbox{if the expert did not mention the intended variable}\\
      \frac{1}{q}&\mbox{if the expert mentioned $q$ variables,
      including the intended one}\\
      1&\mbox{if the expert only mentioned the intended variable}\\
    \end{array}
  \right.
\end{displaymath}
```

Met behulp van de functionaliteit van amsmath kan de volgende, inhoudelijker, weergave worden gebruikt:

```
\begin{displaymath}
  y_{ij}= \begin{cases}
      0 &\text{if the expert did not mention the intended variable}\\
      \frac{1}{q}&\text{if the expert mentioned $q$ variables,
      including the intended one}\\
      1&\text{if the expert only mentioned the intended variable}\\
  \end{cases}
\end{displaymath}
```

Bovendien wordt enige nieuwe wiskundige functionaliteit ter beschikking gesteld (overigens, het hier niet geladen package amsfonts zorgt dat een aantal extra symbolen zoals $\mathbb{R}$, $\mathbb{C}$ via `$\mathbb{R, C}$` kan worden gebruikt). Bijkomend voordeel van het gebruik van amsmath is dat in een aantal gevallen de opmaak van formules 'verbetert' (over smaak valt natuurlijk te twisten, maar het eindresultaat komt beter overeen met mijn verwachtingen en dat beschouw ik als een voordeel).

De packages natbib en babel zijn bij de opties al kort genoemd en zorgen voor respectievelijk extra functionaliteit bij het gebruik van citaties (zo komen commando's als `\citeauthor` en `\citeyear` beschikbaar en kan in plaats van `\cite` gebruik gemaakt worden van `\citet` (voor verwijzingen in de lopende tekst) en `\citep` (voor verwijzingen tussen haakjes)) en taalgerelateerde aanpassingen.

Het package tabularx biedt een uitbreiding van de standaard `tabular`-omgeving van LaTeX. Naast de `tabular`-omgeving, bestaat er nu ook een `tabularx`-omgeving. Aan deze omgeving kan een extra optie worden meegegeven waarin staat hoe breed de tabel moet worden. LaTeX past de breedte van vooraf gespecificeerde kolommen dan automatisch aan teneinde de tabel de juiste breedte te geven. Om deze functie enigszins gemakkelijk te laten werken, worden later enige nieuwe kolomtypen gedefinieerd, die in een aparte paragraaf van deze toolbox samen met een voorbeeld besproken zullen worden.

Het volgende package luistert naar de naam url en definieert het commando `\url`. Dit commando is, het kan ook haast niet anders, uiterst geschikt voor de weergave van URL's. Het zorgt dat de gebruikte tekens goed worden weergegeven (probeer deze (onzinnige) URL maar eens met de hand in te geven: `http://www.cwis_vu.nl/~maarten\lange\url\om\afbreken\te\demonstreren&query=?`) en indien nodig op logische plaatsen worden afgebroken.

fancyvrb stelt extra functionaliteit voor `verbatim`-omgevingen beschikbaar. In de syllabus dient herhaaldelijk SPSS-syntax te worden opgenomen en dit package verschaft daarvoor de benodigde gereedschappen. Hier kom ik later nog op terug.

Dankzij textcomp heb ik de beschikking over een aantal extra tekentjes/symbolen zoals ° (`\textdegree`), £ (`\textsterling`), § (`\textsection`) en ™ (`\texttrademark`). Ook het commando `\textcircled` kan bruikbare diensten bewijzen. Bij dit alles dient overigens wel opgemerkt te worden dat niet alle symbolen even mooi zijn. De euro (€) is bij

voorbeeld ronduit lelijk en ook de Japanners komen er met hun ¥ (\textyen) wat bekaaid af. Zaken als ® (\textcircled{R}) lijken evenmin op wat ik zou verwachten[3].

Het package caption2 dient, zoals reeds is gezegd, voor het aanpassen van de weergave van de bijschriften bij figuren. Het volgende package (wrisym) is voor de meeste lezers niet van belang. Het zorgt ervoor dat ik vanuit LATEX de fonts met de wiskundige symbolen van het programma Mathematica kan gebruiken. Bovendien wordt de tekst automatisch uit times gezet. Een 'poor man's solution' is \usepackage{times,mathptm}.

Het package relsize is typisch zo'n package dat functionaliteit biedt die eigenlijk in LATEX zelf zou moeten zitten. Het zorgt ervoor dat er naast commando's als \small en \large relatief werkende commando's zoals \smaller en \larger beschikbaar komen. Deze commando's kijken wat de huidige lettergrootte is en selecteren vervolgens een iets kleiner of groter lettertype. Handig voor tekst waarvan je van tevoren nog niet weet in wat voor grootte deze gezet gaat worden.

Een voorbeeld van het gebruik van het commando \smaller is te vinden in de volgende commandodefinitie. Hier wordt, later in mijn preamble, een commando \toets gedefinieerd. Met dit commando kan ik op een willekeurige plaats aangeven dat een toets (bij voorbeeld Ctrl–C) moet worden ingedrukt.[4]

```
\newcommand{\toets}[1]{{\smaller\framebox{#1}}}
```

Ook mflogo is een relatief eenvoudig package. Het stelt het metafont- en metapost-logo (METAFONT en METAPOST (\MF{} en \MP{})) beschikbaar. De lezer zij overigens gewaarschuwd: dit zijn twee extreem 'gevaarlijke' commando's. Allereerst kennen oudere versies alleen het metafont-logo. Ten tweede gaat er relatief vaak iets mis met het vinden van deze fonts. Er zijn zowel versies in omloop waarin de fontnaam met hoofdletters wordt geschreven als versies waarin deze met kleine letters wordt geschreven. Als de gebruikte files niet allemaal dezelfde conventie hanteren kan de printer uiteindelijk het font niet vinden, met alle ellendige gevolgen van dien.

Het package lijstmac bevat wat zelf ontwikkelde macro's voor het maken van lijsten (waarover in een volgende toolbox meer). Het package answers dient voor het aanmaken van opgaven. Het biedt de mogelijkheid om antwoorden en opgaven in het brondocument bij elkaar te houden en ze toch op verschillende plaatsen (in dit geval deels achterin de syllabus en deels in een aparte docentenhandleiding) te printen. Het gebruik van dit package komt later in deze toolbox nog uitvoeriger aan bod.

Voor het aanmaken van de index bij de syllabus gebruik ik het package makeidx. De overige commando's die ik voor het aanmaken van de index gebruik worden later besproken.

De packages fontenc en hyperref worden beiden niet geladen. Zij werken alletwee niet samen met de overige packages. Via fontenc zou ik eigenlijk graag T1-encoding willen gebruiken, omdat dan ook woorden met accenten adequaat worden afgebroken. Dit leidt echter tot een conflict met het wrisym package. Het aan brengen van hyperlinks etc. voor een electronische versie via hyperref werkte altijd keurig, maar sinds ik mijn TEX heb geupgrade naar versie 5c van TEXLive is dat verleden tijd.

Het laatste te laden package wordt op de volgende manier opgevraagd.

```
\ifpdf
  \usepackage[pdftex]{graphicx}
```

3. *Produktienoot.* De euro en de cirkel voor omcirkelde letters bleken niet aanwezig in mijn versie van textcomp/Times. Deze twee, en de yen heb ik voor de MAPS gezet uit Computer Modern. Voor de cirkel kon ik gebruik maken van aer, een emulatie van het ec font met behulp van cm, dat in Type1-formaat beschikbaar is, en virtual fonts, maar de euro en de yen zijn de enige bitmapped letters uit de hele MAPS. Het marvosym pakket, beschikbaar in de teTeX distributie, heeft een aantal euro symbolen in Type1 formaat: €, € en €. De yen is aanwezig in de meeste commerciële fonts, b.v. voor Times ¥ [SK].

4. Ctrl–C werkt ook in voetnoten en gebruikt dan een navenant kleiner lettertype.

```
    \graphicspath{{pdf/}}
    \DeclareGraphicsExtensions{.pdf}
    \DeclareGraphicsRule{*}{mps}{*}{}
  \else
    \usepackage[dvips]{graphicx}
    \graphicspath{{afb/}}
    \DeclareGraphicsExtensions{.eps}
  \fi
```

Het zo ingeladen graphicx package biedt, zoals de naam al aangeeft wat grafische functionaliteit. Dit package gebruik ik onder andere om figuren in mijn document op te nemen. Daarnaast gebruik ik het om extreem grote tabellen integraal te verkleinen en eventueel te roteren, zodat zij nog op één pagina passen. Dit is met name van belang voor het weergeven van bij voorbeeld een $z$-tabel (voor de normale verdeling). Daar het graphicx-package gebruik maakt van *specials* laat ik de vraag met welke opties het geladen wordt afhangen van de vraag of ik met pdfTEX werk of met een gewone TEX (en dan uiteindelijk dvips wil gebruiken). Strikt genomen is dit door het gebruik van de globale optie overigens overbodig. Wat niet overbodig is, is het afhankelijk van de gebruikte software instellen van paden voor de vindplaats van afbeeldingen, grafische extensies en regels voor de verwerking van afbeeldingen.

## Het aanmaken van de literatuurlijst

Een andersoortig 'probleempje' dat ik oplos in de preamble is dat de literatuurlijst standaard de verkeerde titel heeft (ik geloof bibliografie, of referenties of zoiets), terwijl ik wil dat daar gewoon 'Literatuur' boven komt staan. Door het herdefiniëren van `\bibsection` wordt dit probleem opgelost. Tegelijkertijd zorg ik ervoor dat de literatuurlijst netjes in de inhoudsopgave wordt opgenomen. Dit alles gebeurt met onderstaande commando's.

```
      % Literatuurlijst hernoemen en in inhoudsopgave opnemen
\makeatletter
\renewcommand\bibsection{\def\refname{Literatuur}%
  \chapter*{\refname
    \@mkboth{\MakeUppercase{\refname}}%
    {\MakeUppercase{\refname}}%
    \addcontentsline{toc}{chapter}{\refname}}%
  }%
\makeatother
```

Wat de lezer aan deze commando's wellicht opvalt is het gebruik van `\makeatletter` en `\makeatother`. In commando-namen mag normaalgesproken het apestaartje (@, oftewel at-sign; vandaar de naam `\makeatother`) niet voorkomen. In het hier weergegeven commando heb ik echter het apestaartje wel nodig. Met het commando `\makeatletter` vertel ik TEX dat het net moet doen of het apestaartje een gewone letter is. Na de definitie van het commando maak ik deze zogenaamde 'catcode-wijziging' weer ongedaan.

Het gebruik van natbib maakt het mogelijk de layout van de literatuurlijst zonder al te veel problemen aan te passen. Ik kies een iets kleiner lettertype en verklein tevens de afstand tussen de individuele titels.

```
\def\bibfont{\small}
\def\bibsep{\smallskipamount}
```

De literatuurlijst zelf wordt aangemaakt met behulp van het volgende commando.

```
{%\citeindexfalse
```

```
\bibliography{string,data,ik,crossref,tmp,mt}%\citeindextrue
}
\bibliographystyle{my-nl}
```

Achter de procent-tekens staan twee commando's die ik nu niet gebruik. Als ik ook een auteursindex aan zou maken, zou ik de procent-tekens weghalen om te voorkomen dat de literatuurlijst zelf in de auteursindex wordt opgenomen. Het commando `\bibliography` zorgt dat de literatuurlijst (die op de gebruikelijke manier met behulp van bibTEX is aangemaakt) wordt opgenomen. De parameters van dit commando geven aan welke `bib`-files bibTEX moet lezen. De parameter van `\bibliographystyle` geeft aan welk stijlbestand voor het opmaken van de literatuurlijst moet worden gebruikt (in dit geval een zelf-gemaakte Nederlandstalige versie met ondersteuning voor het auteur-jaartal systeem).

Een kleine opmerking over de gebruikte bestanden is nog op zijn plaats. Het eerste bestand (`string.bib`) bevat een serie afkortingen die ik in de andere bestanden gebruik. Deze afkortingen (in totaal zo'n 200) hebben de volgende vorm.

```
@STRING ( AOS = "Accounting, Organizations and Society" )
```

Het bestand `data.bib` bevat het merendeel van de literatuurverwijzingen. In dit bestand worden de eerder gedefinieerde afkortingen gebruikt. Dit scheelt niet alleen typewerk, maar zorgt er ook voor dat ik de namen van tijdschriften consequent spel en gemakkelijk naar een afgekort formaat (Acc., Org. & Soc.) over kan stappen.

```
@ARTICLE{Abernethy95a,
  author =       {Abernethy, Margaret A. and Stoelwinder,
                  Johannes U.},
  title =        {The role of professional control in the
                  management of complex organizations},
  year =         1995,
  journal =      AOS,
  volume =       20,
  pages =        {1-17},
  number =       1
}
```

De bestanden `ik.bib`, `tmp.bib` en `mt.bib` bevatten eveneens gegevens. Het bestand `crossref.bib` bevat de titels van artikelenbundels en dat soort zaken. Het is voor het gebruik van bibTEX noodzakelijk dat een artikel uit een bundel eerder in de `bib`-file staat dan de bundel zelf (hetgeen tot conflicten leidt met mijn wens mijn gegevensbestanden te sorteren). Een andere mogelijkheid is het opnemen van de bundels in een file die later geladen wordt dan de file met de artikelen. Dat is de optie die ik hier kies. De artikelen staan in `data.bib` en `ik.bib`, de bundels in `crossref.bib`.

## Instellingen voor tabularx

Eén van de packages die eerder geladen is, is tabularx. Zoals gezegd biedt dit package uitbreidingen op de standaard `tabular`-omgeving van LATEX. De belangrijkste uitbreiding is dat aan een tabel een breedte mee kan worden gegeven en dat vooraf geselecteerde kolommen vervolgens automatisch breder of smaller worden gemaakt om zo de tabel de gewenste breedte te geven. Het package is zonder verdere instellingen prima te gebruiken, maar ikzelf geef er de voorkeur aan een iets 'toegankelijker' interface te definiëren. Hiertoe maak ik de volgende nieuwe kolomtypen aan:

```
          % Nieuwe kolommen voor tabularx en array
\newcolumntype{L}{>{\raggedright\arraybackslash}X}
```

```
\newcolumntype{R}{>{\raggedleft\arraybackslash}X}
\newcolumntype{C}{>{\centering\arraybackslash}X}
```

De kolomtypen L, R en C vormen nu het ‘rekbare’ equivalent van de traditionele kolomtypen l, r en c. Nu kan ik in mijn tekst bij voorbeeld de volgende commando’s gebruiken:

```
\begin{tabularx}{0.67\textwidth}{Lrr}
Enige tekst&tekst&tekst\\
Enige tekst&tekst&tekst\\
\end{tabularx}
```

Om een tabel aan te maken die 0.67 keer zo breed is als de lopende tekst. De tweede en derde kolom worden gewoon rechts uitgelijnd. De eerste kolom wordt links uitgelijnd en net zo lang opgerekt tot de tabel breed genoeg is. Indien een kolom te lang is, wordt deze over meerdere regels verdeeld:

```
\begin{tabularx}{0.67\textwidth}{L|r|r}
  Enige tekst en een boel tekst,
  en een boel tekst, en een boel tekst,
  en een boel tekst, en een boel tekst,
  en een boel tekst, en een boel tekst,
  en een boel tekst.&tekst&tekst\\
  Enige tekst&tekst&tekst\\
\end{tabularx}
```

Leidt tot de volgende tabel:

| | | |
|---|---|---|
| Enige tekst en een boel tekst, en een boel tekst, en een boel tekst, en een boel tekst, en een boel tekst, en een boel tekst, en een boel tekst. | tekst | tekst |
| Enige tekst | tekst | tekst |

Vanzelfsprekend is het mogelijk om meerdere ‘rekbare’ kolommen tegelijkertijd te gebruiken en om nog extra soorten kolommen te specificeren (bij voorbeeld om te zorgen dat de éne kolom twee keer zo snel groeit als de ander).

## Gebruik van answers

Zoals reeds eerder aangegeven gebruik ik het package answers voor het weergeven van mijn opgaven, het maken van een set uitwerkingen en hints voor studenten en het aanmaken van het antwoordboekje. Uiteindelijk ziet een opgave er in mijn document zo uit:

```
\begin{opgave}
  Hoeveel is $1+1$?
  \begin{hint}
    Gebruik je rekenmachine!
  \end{hint}
  \begin{uitwerking}
    2
  \end{uitwerking}
\end{opgave}
```

De hele opgave staat in een `opgave`-omgeving. De hint (die in de syllabus zelf terecht moet komen) staat in een hint-omgeving. De uitwerking, die alleen voor docenten is bestemd, staat in de `uitwerking`-omgeving. Van sommige opgaven komt het antwoord ook in de syllabus terecht (tussen de hints), deze staan in een `antwoord`-omgeving.

Het opnemen van de antwoorden in de syllabus gaat als volgt. Het package answer plaatst de uitwerkingen in hulpfiles. Ik gebruik er twee. Eén voor docenten en één voor studenten. Voordat ik de hulpfiles in de bijlage kan opnemen, moet ik ze eerst sluiten met \Closesolutionfile. Vervolgens laad ik de studentenfile met behulp van het commando \input.[5]

```
\Closesolutionfile{docent}
\Closesolutionfile{student}
\chapter{Antwoorden en hints}
\label{cha:antwoorden}
\input{student}
```

Voordat answer werkt zoals hierboven beschreven staat, moet er wel het één en ander gebeuren. Allereerst wil ik mijn opgaven nummeren. In de commando's komen apestaartjes voor, dus ik begin met \makeatother. Vervolgens geef ik aan dat ik een nieuwe counter wil definiëren (die luistert naar de naam 'opgavenummer') en dat de opgavenummers ieder hoofdstuk opnieuw bij 1 moeten beginnen. De laatste hieronder gepresenteerde regel zorgt dat het opgavenummer wordt voorafgegaan door het hoofdstuknummer (anders maakt de helft van mijn practicumdeelnemers weer de opgaven uit een verkeerd hoofdstuk), waarna het nummer van de opgave zelf als arabisch getal (dat is een gewoon getal en geen romeins cijfer of een letter of zo) wordt weergegeven:

```
      % Setup voor answers
\makeatletter
\newcounter{opgavenummer}[chapter]
\renewcommand{\theopgavenummer}{\thechapter.\@arabic\c@opgavenummer}
\makeatother
```

Dan moet ik de omgeving opgave nog definiëren. Aan het begin van deze omgeving wordt de opgave-teller automatisch verhoogd met één. Ik gebruik het commando \refstepcounter om te zorgen dat ik ook naar opgaven kan verwijzen. Iedere opgave begint een nieuwe paragraaf (\par) en aan het begin van de opgave wordt zonder in te springen (\noindent) in vet het woord 'opgave' en het nummer van de opgave weergegeven. Opgaves eindigen met een rechts uitgelijnd vierkant.[6] Het rechts uitlijnen wordt bereikt door een oneindig rekbare \hspace op te nemen.

```
\newenvironment{opgave}{%
  \par\refstepcounter{opgavenummer}
  \noindent\textbf{Opgave \theopgavenummer:}}
  {\nobreak\hspace*{\stretch{1}}\nobreak\EmptySquare\par}
```

Nu moet ik nog zorgen dat de uitwerkingen, hints en antwoorden in de goede hulpfile terecht komen. Uitwerkingen komen in een bestand docent.tex, de hints en uitwerkingen moeten op volgorde in een bestand student.tex worden opgenomen. Als ik de hints en de uitwerkingen apart zou willen weergeven, zou ik voor beiden een apart hulpbestand moeten gebruiken.

```
\Newassociation{uitwerking}{Uitwerking}{docent}
\Newassociation{hint}{Hint}{student}
\Newassociation{antwoord}{Antwoord}{student}
```

De bijbehorende labels moeten nog zo worden gedefinieerd dat de uitwerkigen er een beetje fatsoenlijk uitzien:

5. Het uitwerkingenboekje is een aparte TEX-file, die niets anders doet dan de docentenfile opvragen binnen een document-omgeving.
6. Dit vierkant is afkomstig uit de symbolenset van Mathematica en dus niet standaard beschikbaar.

```
\renewcommand{\Uitwerkinglabel}[1]{\textbf{Uitwerking voor opgave #1:}}
\renewcommand{\Hintlabel}[1]{\textbf{Hint voor opgave #1:}}
\renewcommand{\Antwoordlabel}[1]{\textbf{Antwoord voor opgave #1:}}
```

Door middel van een 'truc' zorg ik er vervolgens voor dat het nummer van de juiste opgave, telkens wordt meegegeven aan de hierboven gedefinieerde kopjes van de opgaves. Overigens heb ik sterk het idee dat dit efficiënter moet kunnen via één of andere optie van het gebruikte package. Voorlopig ben ik echter al heel blij dat de boel het doet.

```
\def\nummer{\theopgavenummer}
\let\Currentlabel\nummer
```

Daarna hoef ik alleen de hulpfiles nog maar te openen, zodat de gegevens daadwerkelijk naar de hulpfile weggeschreven kunnen worden.

```
\Opensolutionfile{docent}
\Opensolutionfile{student}
```

## Gebruik van fancyvrb

Het package fancyvrb gebruik ik voor het weergeven van SPSS-code. Voordat ik dit package gebruik, herdefinieer ik eerst het uitroepteken. Het uitroepteken komt in SPSS-syntax vrijwel niet voor en wil ik gebruiken als commando om cursieve tekst te selecteren voor delen van de syntax. Het eerste wat ik doe is het opslaan van de oorspronkelijke betekenis van het uitroepteken in het commando `\ExclamationPoint`. Vervolgens maak ik het uitroepteken actief. Het uitroepteken is nu geen uitroepteken meer, maar een commando. Tot slot definieer ik het commando uitroepteken en geef het standaard dezelfde betekenis als het zojuist gedefinieerde commando `\ExclamationPoint`.

```
\def\ExclamationPoint{\char'!}

\catcode'!=\active
\def!{\ExclamationPoint{}}
```

De SPSS-code moet in een omgeving genaamd SPSS terecht komen. Om deze omgeving te definiëren gebruik ik het commando `\DefineVerbatimEnvironment` zoals dat hieronder is weergegeven. Achter frame geef ik aan dat boven en onder de syntax een lijn moet komen, het lettertype moet redelijk klein zijn en ik herdefinieer het zojuist actief gemaakte uitroepteken. Het uitroepteken is nu geen uitroepteken meer, maar het commando `\em`. Verder geef ik aan dat de syntax bij elkaar moet worden gehouden op een enkele pagina (dit werkt niet optimaal, de syntax blijft inderdaad keurig bij elkaar, maar de lijn boven/onder de syntax wil nog wel eens op de verkeerde pagina terechtkomen). Tot slot specificeer ik de afstand tussen syntax en de lijnen.

```
\DefineVerbatimEnvironment{SPSS}{Verbatim}{%
  frame=lines,
  fontsize=\small,
  defineactive=\def!{\em},
  samepage=true,
  framesep=0.7\fboxsep}
```

Nu heb ik een omgeving waarin ik de volgende truc kan uithalen:

```
\begin{SPSS}
  tekst !cursieve tekst! rechte tekst !cursief!
  En alle andere vervelende letters werken nog:
  ~@#$%^&*()_+-={}[]\
\end{SPSS}
```

```
tekst cursieve tekst rechte tekst cursief
En alle andere vervelende letters werken nog:
~@#$%^&*()_+-={}[]\
```

In de praktijk ziet een stukje SPSS-syntax er dan als volgt uit (\cindex zorgt ervoor dat het commando in de index wordt opgenomen).

```
\cindex{regression}
\begin{SPSS}
  regression
    /variables = {!variabelen!|(collect)|all}
    /descriptives = {defaults|mean|stddec|corr|cov|variance|
                     all|none}
    /missing = {listwise|pairwise}
    /statistics = {defaults|r|coeff|anova|outs|all}
    /dependent = !variabelen!
    /method = enter !variabelen!
    /residuals
    /casewise
    /scatterplot = !variabele!
    /partialplot = !variabele!
    /save = !tempvar!(!newvar!).
\end{SPSS}
```

```
regression
  /variables = {variabelen|(collect)|all}
  /descriptives = {defaults|mean|stddec|corr|cov|variance|
                   all|none}
  /missing = {listwise|pairwise}
  /statistics = {defaults|r|coeff|anova|outs|all}
  /dependent = variabelen
  /method = enter variabelen
  /residuals
  /casewise
  /scatterplot = variabele
  /partialplot = variabele
  /save = tempvar(newvar).
```

## Het aanmaken van de index

Om een index aan te maken, moet ik natuurlijk eerst even zorgen dat de benodigde hulpbestanden automatisch worden gegenereerd. Dit bereik ik door het volgende commando in de preamble op te nemen.

```
\makeindex
```

Alleen bovenstaand commando zou al voldoende zijn voor het genereren van een bruikbare index (in de tekst moet natuurlijk nog wel aan worden gegeven wat er geïndexeerd dient te worden). In de praktijk heb ik echter de gewoonte nog een aantal andere zaken in de preamble op te nemen. Zo staat er ergens halverwege—het zal iedere lezer in de tussentijd duidelijk zijn dat mijn preamble één ongeordende chaos is—het volgende commando.

```
\newcommand{\cindex}[1]{\index{#1@\texttt{#1}}}
```

Betekenis en nut zijn wellicht niet direct duidelijk, maar niet moeilijk te begrijpen. Ik wil in de index onder andere SPSS-commando's opnemen. Deze moeten daar in typemachineschrift worden weergegeven. Als ik daarvoor zonder voorzorgsmaatregelen het `\index`-commando zou gebruiken, zou mijn index niet langer goed gesorteerd worden. Gelukkig voorziet het index-commando in de @-functionaliteit. De index wordt gesorteerd op hetgeen voor de apestaart staat en weergegeven als wat er achter staat. Dit brengt echter een hoop onnodig typewerk met zich mee, vandaar dat ik bovenstaand commando `\cindex` aanmaak, dat slechts één parameter (de in typemachineschrift te verschijnen index-entry, oftewel het SPSS-commando) meekrijgt en zorgt dat deze goed gesorteerd en juist weergegeven in de index verschijnt. Een voorbeeld van het gebruik van dit commando staat bij de regressie-syntax op pagina 66.

Niet alle index-entries worden in de lopende tekst aangemaakt. Om de index bruikbaar te maken zijn ook verwijzingen van groot belang. Aan het einde van mijn preamble staat een hele set commando's die er voor zorgen dat die verwijzingen in de index worden opgenomen. Zij hebben allemaal de volgende vorm.

```
\index{degree of freedom|see{vrijheidsgraad}}
```

Die zou ik eigenlijk eens in een apart hulpbestand moeten plaatsen. Dat maakt de boel weer wat leesbaarder. Nog mooier zou het zijn om een commando `\indexverwijzing` aan te maken. Dat zou op de volgende manier moeten gebeuren.

```
\newcommand{\indexverwijzing}[2]{\index{#1|see{#2}}}
```

Aan het 'einde' van de syllabus moet de index natuurlijk nog geprint worden. Dit gebeurt door het volgende commando op te nemen.

```
{\printindex}
```

Dat vereist overigens wel dat er een index-bestand aanwezig is. LaTeX maakt slechts een hulpbestand aan, dat vervolgens door een extern programma (meestal `makeindex` geheten) gesorteerd moet worden. Normaalgesproken is dit een kwestie van de namen van de commando's intypen. Bij de syllabus treedt er echter een complicatie op. Ik heb het uitroepteken actief gemaakt en standaard dient het uitroepteken voor verwijzingen van verschillende niveaus (naast een index-entry factoranalyse, is er bij voorbeeld een entry, factoranalyse; vooronderstellingen en een entry factoranalyse; interpretatie. Normaalgesproken worden de niveaus in het indexcommando als volgt ingegeven:

```
\index{factoranalyse!vooronderstellingen}
```

Met het actief gemaakte uitroepteken werkt dit niet meer. In plaats daarvan gebruik ik de punt-komma.

```
\index{factoranalyse;vooronderstellingen}
```

Dit moet echter nog wel even aan `makeindex` worden verteld, anders loopt de boel in het honderd. Om `makeindex` de niveaus juist te laten verwerken, maak ik een bestand `syllabus.ist` aan met de volgende inhoud:

```
level ';'
```

Door nu `makeindex` als volgt aan te roepen, wordt de index gegenereerd (de c dient om spaties te negeren).

```
makeindex -c -s syllabus syllabus
```

## Wat extra commando's

Tussen alle commando's die direct aan een package gerelateerd zijn, staan in mijn preamble nogal wat, eigenlijk vrij willekeurige extra commando's, die niet veel meer doen dan mijn leven vergemakkelijken. Het eerste commando dient voor het opnemen van elementen van het rooster in de studiewijzer. Iedere week worden er één of meer onderwerpen besproken die over het algemeen overeenkomen met één of meer onderwerpen uit het boek van Hair en/of de hier besproken syllabus. Het onderstaande commando definieert een nieuw commando `\rooster` met 5 parameters: het weeknummer, de datum, het hoofdstuk uit Hair, het hoofdstuk uit de syllabus en een nadere omschrijving van het onderwerp. Door dit commando aan te roepen zorg ik ervoor dat de beschrijvingen van alle weken in mijn studiewijzer er consistent uitzien—en het bespaart ook nog eens typewerk.

```
      % Collegerooster
\newcommand{\rooster}[5]{%
  \noindent\textbf{Week #1}
  \begin{description}
    \item [Datum] #2
    \item [Hair] Hoofdstuk #3
    \item [Syllabus] Hoofdstuk #4
    \item [Onderwerpen] #5
    \end{description}}
```

In de studiewijzer, gebruik ik dit commando als volgt:

```
\rooster
  {5}
  {28/2/2000}
  {2}
  {3 en 4}
  {Het eerste uur wordt het onderwerp gegevensverzameling verder
  uitgebreid.  Er wordt ingegaan op case studies en het coderen van
  gegevens. Het tweede uur wordt besteed aan verkennen en presenteren
  van gegevens, het gebruik van tabellen en figuren wordt
  overgeslagen, dit kan je zelf gemakkelijk bestuderen. Outliers,
  normaliteit en het nut van gegevenstransformaties komen wel aan bod}
```

Op een vergelijkbare wijze worden ook commando's aangemaakt voor speciale lijsten etc. In sommige gevallen zijn de commando's iets complexer dan bovenstaand voorbeeld, omdat bij voorbeeld de kantlijnen worden aangepast of er zaken worden geroteerd. Het principe is echter gelijk en ik zal dan ook niet al deze voorbeelden bespreken.

Een volgende cosmetische aanpassing is nog kleiner: ik wil dat er bij de voetnoten geen streep verschijnt. Dit effect bereik ik door het commando `\footnoterule`, waarmee de streep wordt gezet, te herdefiniëren. Door het commando `\renewcommand` zoals hieronder weergegeven wordt het bestaande `\footnoterule`-commando vervangen door `\relax`, oftewel: doe niets.

```
      % Geen streep bij voetnoten
\renewcommand\footnoterule{\relax}
```

Een actie die de meeste lezers wel bekend voor zal komen is het aanmaken van een nieuwe `theorem`-environment. Het zal geen verbazing wekken dat ik in een syllabus over onderzoeksmethoden en -technieken af en toe een hypothese nodig heb. Dit wordt hieronder gedefinieerd. In dit geval heet de environment hypo en de starttekst is Hypothese.

```
\newtheorem{hypo}{Hypothese}
```

Ik hou ervan om hoofdstukken te beginnen met een kort citaat. Dit citaat moet wat smaller zijn dan de tekst, cursief worden weergegeven en worden gevolgd door de naam van de auteur en een jaartal. Zo ongeveer als hieronder, maar dan (in het geval van de syllabus) ook nog met paginanummer.

> *Het rare van die wiskunde is dat het uiteindelijk toch altijd nog ergens op wordt toegepast.* Hugo Brandt Cortius (1997)

Het volgende commando bereikt dit.

```
      % Citaten (heeft op deze wijze wel natbib/plainnat nodig)
\newlength{\half}
\half=0.3\textwidth
\newenvironment{citaat}[2]{%
  \def\auteur{#1}\def\plaats{#2}%
  \list{}{\rightmargin\nomargin\leftmargin\half}
  \item[]\em\small}
  {\hspace*{\stretch{3}}\nolinebreak[2]\hspace*{\stretch{3}}%
    \textnormal{%
     \mbox{\citeauthor{\auteur}~(\citeyear{\auteur},~\plaats)}}%
    \endlist\medskip\ignorespaces}
```

Eerst definieer ik een nieuwe lengte, die ik de naam `\half` geef (kennelijk wilde ik in eerste instantie citaten half zo breed maken als de rest van de tekst, maar, zoals op de volgende regel te zien is, ben ik daarop teruggekomen, ze worden 0.7 keer zo breed; een typisch voorbeeld van hoe (mijn) macro's na verloop van tijd volledig onleesbaar worden). Na het definiëren van de lengte wordt een omgeving `citaat` gedefinieerd. Deze omgeving heeft twee parameters: de sleutel van het citaat in mijn literatuurdatabase (noodzakelijk voor het `\cite`-commando) en de plaats (meestal een paginanummer) waar het citaat gevonden kan worden. Zoals de meeste LaTeX-gebruikers zullen weten, bevat de definitie van een omgeving (zoals er hier één wordt aangemaakt, het citaat komt immers tussen `\begin{citaat}` en `\end{citaat}` te staan) twee parameters: de commando's die aan het begin van de omgeving moeten worden uitgevoerd en de commando's die aan het einde moeten worden uitgevoerd. Er treedt bij dit commando echter een complicatie op. De extra argumenten van een omgeving (in dit geval de sleutel voor de database en het paginanummer) zijn slechts aan het begin van de omgeving, en niet aan het einde beschikbaar. Het eerste wat ik dan ook doe is deze eerste en tweede parameter opslaan in de commando's `\auteur` en `\plaats`. Hier gebruik ik het commando `\def` voor. Ik geloof dat dit niet geheel volgens de regels van goed LaTeX-gebruik is, maar het is de gemakkelijkste mij bekende manier om te zorgen dat een commando dat nog niet bestaat wordt aangemaakt en een reeds bestaand commando rücksichtslos wordt overschreven, zodat bij een tweede aanroep van de `citaat`-omgeving de gegevens van het eerste citaat automatisch gewist worden.[7]

Na het opslaan van de parameters wordt de layout van de omgeving gespecificeerd. De gebruikelijke, maar weinig intuïtieve manier om dit in LaTeX te doen is het gebruik van een `\list`. In dit geval een erg eenvoudige. Alleen de linker- en rechtermarge hoeven maar gedefinieerd te worden. Vervolgens dient er nog een `\item`-commando gegeven te worden (anders gaat LaTeX klagen over een lijst zonder items) en daarna wordt het lettertype (cursief en klein) geselecteerd.[8]

Aan het einde van het citaat dient de naam van de auteur geplaatst te worden, samen met jaartal en paginanummer. Ik kies ervoor om deze gegevens allemaal rechtop (in romein)

7. Wellicht was het gebruik van `\bgroup` en `\egroup` eleganter geweest.
8. In mijn geval is `\nomargin` al gedefinieerd in de gebruikte documentclass. Dit is niet standaard het geval.

te zetten en ze op één regel bij elkaar te houden. Indien het citaat op de laatste regel voldoende ruimte overlaat voor de 'aftiteling' wordt deze rechts uitgelijnd achter de laatste regel van het citaat geplaatst. Is er te weinig ruimte, dan komt de hele regel op een nieuwe regel. Dit is, samen met het toevoegen van wat wit aan het einde van het totale citaat, ongeveer alles wat de laatste regels van het commando doen, maar deze excercitie vraagt nogal wat commando's.

De regel:

```
{\hspace*{\stretch{3}}\nolinebreak[2]\hspace*{\stretch{3}}%
```

zorgt ervoor dat indien er genoeg ruimte is, de volgende 'aftiteling' direct achter de laatste regel van het citaat komt. Indien dit niet het geval is, komt de 'aftiteling' op een aparte regel, zoals hieronder.

> *Validity refers to the capacity of a test to tell us what we already know.*
> George Kelly

De truc werkt als volgt. In bovenstaande regel staan twee oneindig rekbare stukken wit. Deze zorgen er met zijn tweeën danwel voor dat de aftiteling die op dezelfde regel blijft staan rechts wordt uitgevuld, danwel dat de laatste regel van het citaat netjes wordt opgevuld met wit en op de volgende regel de aftiteling links met wit wordt opgevuld (zodat de aftiteling rechts uitgevuld verschijnt). Het commando `\nolinebreak` zorgt ervoor dat tussen beide stukken oneindig rekbaar wit bij voorkeur geen nieuwe regel wordt begonnen, zodat de aftiteling als het even mogelijk is direct achter het citaat terecht komt.

Op de volgende en laatste regel van de commandodefinitie wordt het standaard lettertype weer gekozen, zorgt de `\mbox` ervoor dat zijn inhoud op één enkele regel blijft staan (ook als die regel te kort is), halen `\citeauthor` en `\citeyear` respectievelijk auteur en publicatiejaar uit de literatuurlijst en wordt vervolgens de lijst afgesloten. De gebruikte `\cite`-commando's vereisen overigens wel dat natbib wordt gebruikt in combinatie met een adequate bibliography-style.

```
\textnormal{%
 \mbox{\citeauthor{\auteur}~(\citeyear{\auteur},~\plaats)}}%
\endlist\medskip\ignorespaces}
```

Na deze citaat-exercitie volgt een ander commando uit de categorie 'restpost' dat hieronder wordt besproken. Op een gegeven moment heb ik er in de syllabus de behoefte een spatie weer te geven als ␣. In lopende tekst zou dit geen probleem zijn, maar in dit geval vormt de spatie het argument voor een ander commando. Het gebruik van `\verb*| |` is derhalve niet mogelijk. Om dit probleem op te lossen maak ik een box genaamd spatie aan en in deze box plaats ik de ␣. De box met de naam spatie kan ik vervolgens zonder problemen als argument van een ander commando gebruiken. Overigens is dit een typisch voorbeeld van een niet-elegante, maar wel werkende oplossing.

```
\newbox\spatie% Spatie bevat een verbatim spatie
\setbox\spatie=\hbox{\verb*| |}
```

Als parameter van het commando `\hyphenation` volgt een aantal woorden dat anders verkeerd of onvoldoende wordt afgebroken (in werkelijkheid is de lijst langer).

```
\hyphenation{res-pon-se com-man-do ma-na-ge-ment-acti-vi-tei-ten}
```

Omdat ik geen zin heb mijn tekst na te kijken op overfull boxes zet ik `\emergencystretch` op een redelijk grote waarde:

```
\setlength{\emergencystretch}{1em}
```

Een andere actie is het herdefiniëren van de afstand die met `\,` wordt overgeslagen in tekstmode (ik vind deze standaard te groot voor times, bij computer modern voldoet de standaardwaarde beter). De afstand in math-mode laat ik ongewijzigd, omdat formules in times er al enigszins krap uitzien.

```
\def\thinspace{\kern .15em } % was \def\thinspace{\kern .16667em }
\DeclareRobustCommand{\,}{%
   \relax\ifmmode\mskip\thinmuskip\else\thinspace\fi
}
```

In sommige gevallen is het handig een pagina één of meer regels langer of korter te kunnen maken, om zo net een pagina minder te gebruiken of weduwen en wezen (alleenstaande eerste en laatste regels van een alinea) te voorkomen. Met dit doel definieer ik een commando `\erbij`, dat wanneer het op een pagina wordt gebruikt, de lengte van die pagina met het als parameter opgegeven aantal regels verlengt:

```
\newcommand{\erbij}[1]{\enlargethispage{#1\baselineskip}}
```

Af en toe wil ik in de tekst een breuk (3/8 of 123/597) kunnen gebruiken. Het commando `\breuk` zorgt dat deze er enigszins beschaafd uitzien. De opgegeven verplaatsingen zijn sterk lettertype afhankelijk. De hier weergegeven waarden werken redelijk voor een times (de slash zou eventueel nog iets groter kunnen worden gezet).

```
\newcommand{\breuk}[2]{\leavevmode\kern.1em
  \raise.7ex\hbox{\footnotesize #1}\kern-.1em
  /\kern-.05em\lower.25ex\hbox{\footnotesize #2}}
```

Het volgende commando eet zijn argumenten op en doet verder helemaal niets. Dat is op zich ook nog een behoorlijke kunst, spaties om het commando heen moeten namelijk ook genegeerd worden, dat doen de twee hacks.

```
\makeatletter
\newcommand{\opm}[1]{\@bsphack\@esphack}
\makeatother
```

Wat het nut van het hierboven beschreven commando is, is misschien niet iedereen duidelijk. Mijn doel is dat ik in mijn document opmerkingen op kan nemen, die standaard niet worden uitgeprint. Als ik een papieren versie wil waarop mijn opmerkingen wel staan, dan herdefinieer ik `\opm` als volgt:

```
\newcommand{\opm}[1]{\{@bsphack\par\flushleft\small\em #1}\@esphack}
```

## Oproep

Hierboven heb ik laten zien hoe een relatief groot document er bij mij na verloop van tijd uit gaat zien. Ik heb het syllabusvoorbeeld gebruikt om te tonen welke packages ik handig vind, welke trucs ik gebruik en wat een chaos een document na verloop van tijd wordt. Dit is geen voorbeeld van hoe iets hoort of moet, maar een voorbeeld van hoe iets kan. Het lijkt me interessant om in een volgende MAPS eens aandacht te besteden aan de trucs van anderen. Vele lezers zullen zelf een vergelijkbaar document onder handen hebben dat ook een aparte MAPS-bijdrage verdient. Anderen werken wellicht met minder omvangrijke documenten, maar zullen zich toch ook een aantal trucs eigen hebben gemaakt en een set van favoriete packages hebben. Dit soort zaken en voorbeelden van oplossingen die handiger zijn dan wat ik hier presenteer kan eenieder aan mij of de redactie mailen, zodat we daar in een volgende MAPS één of meer artikelen aan kunnen wijden.