Typesetting 'Normaltext'*

Reinhard Wonneberger
Hamburg†

Abstract

To improve their readability, ancient texts can be displayed in a way known from mathematical formulas. This kind of display is backed by modern linguistic research, which has revealed the intrinsic structuring of texts. This revelation has produced the need for a congenial form of display. As a compromise between linguistic accuracy and practical needs, the concept of *Normaltext* has been developed and has proved helpful especially with the study of biblical and similar texts. Typesetting such texts in TEX or LATEX can save a lot of tedious work, but requires some special macros that are presented here.

1 On Readability

No one would think of typesetting long and complicated mathematical formulas in the shape of an ordinary paragraph , i.e. as a continuous stream of symbols. To display them as a separate block of text contributes to their *readability*, and vertical alignment can be used to clarify the internal structure of the formula.

The same holds true for ancient texts. Their words or, sometimes, even their single letters have as much weight for interpretation as mathematical symbols. One such text that is hard to understand, when read in paragraph mode, is the so-called *Trostwort an Baruch* in *Jeremiah 45*, which we quote here from the *The New English Bible*:[1]

> THE WORD WHICH THE PROPHET JEREMIAH SPOKE to Baruch son of Neriah when he wrote these words in a book at Jeremiah's dictation in the fourth year of Jehoiakim son of Josiah, king of Judah: These are the words of the LORD the God of Israel concerning you, Baruch: You said, 'Woe is me, for the LORD has added grief to all my trials. I have worn myself out with my labours and have had no respite.' This is what you shall say to Baruch, These are the words of the LORD: What I have built, I demolish; what I have planted, I uproot. So it will be with the whole earth. You seek great things for yourself. Leave off seeking them; for I will bring disaster upon all mankind, says the LORD, and I will let you live wherever you go, but you shall save your life and nothing more.

If we can find a sound method of applying the principles of display and vertical alignment to such texts, we can be sure of advancing their readability a great deal.

2 The Method

To achieve a method of text display that is not just left over to our intuition, we have to work out a linguistic model of segmentation. A method meeting these requirements is described in my article on *Normaltext (N)*,[2] and a text that is segmented and displayed according to this model will be called *Normaltext* to make clear that this form of display should represent the standard.

The method of *Normaltext* is based upon the linguistic model of text structuring developed by *Elisabeth Gülich* and *Wolfgang Raible*. The kernel of that theory can be summarized by the list of *Textgliederungssignale* given in Fig. 1.

To mark all the elements from the list in an existing text is a task too complicated to provide an undisputed text display at the beginning of research. From some working experience, however, it will become clear soon that there is a basic distinction be-

---

* This paper is dedicated to my teacher in the Old Testament, Prof. Dr. Klaus Koch, on the occasion of his 60^{th} birthday on October 4, 1986.

† The present article and also the macros contained in it were developed as a test case for the new ST-TEX (cf. *Klaus Guntermann: Porting TEX to the ATARI ST. TUGboat 7,3 (1986) p. 164.*) and to gain experience in the possibilities of 'job-sharing' and data transfer between the PC and the host at *DESY, Notkestraße 85, D 2000 Hamburg 54, FRG.* I should like to thank *P. K. Schilling* and *P. Stackhouse* for their help with corrections and English style. Comments should be sent to *R. W., Drachenstieg 5, D 2000 Hamburg 63* or through *Bitnet/Earn* to *B03WBG* at *DHHDESY3*.

1 New York: Oxford University Press 1971, p. 972. — The word *Lord* is capitalized when used for the name of God (JHWH).

2 *Normaltext und Normalsynopse. Neue Wege bei der Darstellung alttestamentlicher Texte. Zeitschrift für Sprachwissenschaft 3 (1984) 203–233.*

1. Metakommunikative Sätze
2. Substitution auf Metaebene
3. Episoden- und Iterationsmerkmale
4. Veränderung der Konstellation der Handlungsträger
5. Renominalisierung
6. Satzkonjunktionen und Satzadverbien

Figure 1: Prioritätenliste der Textgliederungssignale. This table is taken from *Normaltext* p. 213; more detailed information can be found there.

tween the first two and the remaining items. While the latter denote properties of one and the same text, the former describe the *embedding* of one text into another. Text embedding is normally beyond dispute and thus can provide a sound basis for our text display. On the other hand, a consensus will easily be reached on the smallest parts of a text, i.e. parts with a sentence-like character.[3]

Thus, with our method of text display, we shall not get lost in the complexity of elements, but concentrate on the two fundamental features of *basic lines* [*Basiszeilen*] and *text embedding* [*Texteinbettung*].

For an outline of that theory, we can refer to our paper on Normaltext. Suffice it here to sketch the main rules governing the display form of a *Normaltext:*

1. The text is broken into basic lines [*Basiszeilen*] according to the so-called signals of structure [*Textgliederungssignale*]. For Hebrew texts, the most common signal is the narrative form.
2. Each line is prefixed by a verseline label, showing
   (a) the chapter number in bold, if it is the beginning of a new chapter;
   (b) the verse number, if it is the start of a new verse;
   (c) the letter 'b', if it is the start of the second part of the verse after the *atnach* (applicable to Hebrew only);
   (d) one of the letters 'A B C ...' showing that it is the n-th line relative to the current verse.
3. These lines are indented *after* the label if they are from an embedded speech [*Redewiedergabe*], and this is done recursively if another speech is embedded.
4. Redactional additions can be indented *in front of* the label by a larger amount.

These rules, when applied to our example text *Jeremiah 45*, yield the display of Fig. 2. To force the material to stay together we have put it into a figure so that it can float to an appropriate place. This will help us to get an impression of the *shape* of the text which is expressed through indentation.

3 Fields of Application

The ideas of *display* and *vertical alignment* are especially useful in high level text research, and the method of *Normaltext* and *Normalsynopse* was developed for the research done by my teacher in the Old Testament, *Prof. Dr. Klaus Koch*, and his *Arbeitsstelle für Profetenforschung* in the first place. Some examples of the resulting kind of text display are given in the previously mentioned paper on *Normaltext (N)* as well as in my *Leitfaden (L)*.[4]

1. A synopsis of the Massoretic, Septuagint, and Tagum text of *Isaias 19,25b* (*L: Tafel 1*; *N: Abb. 18*).
2. A synopsis in German of *Judges 6,8f* and *1. Samuel 10,18* (*N: Abb. 14*).
3. A 'Normaltext' in German of *Isaias 7,1–9* (*N: Abb. 9*).
4. A synopsis of a reconstructed Massoretic text and its Septuagint counterpart for *1. Samuel 14,41* (*L: Tafel 3*).
5. A synopsis of the Septuagint and Theodotion Greek texts of *Daniel 14,27* (*N: Abb. 17*) and *Daniel 14,40* (*L: Tafel 4*).

[3] The standard case for Hebrew will be marked very clearly by a *Narrativ* at the very beginning. There are other clear cases, too, and they are assembled in *N: Abb. 8*. There remain only some cases that require decision (cf. the discussion in *N: p.214*).

[4] Leitfaden zur Biblia Hebraica Stuttgartensia. *Göttingen: Vandenhoeck & Ruprecht* 1984.

1	**45**	THE WORD
2	*B*	WHICH THE PROPHET JEREMIAH SPOKE
3	*C*	to Baruch son of Neriah
4	*b*	when he wrote these words in a book at Jeremiah's dictation
5	*E*	in the fourth year of Jehoiakim son of Josiah, king of Judah:
6	2	These are the words of the LORD the God of Israel *b* concerning you, Baruch:
7	3	You said,
8	*B*	'Woe is me,
9	*C*	for the LORD has added grief to all my trials.
10	*b*	I have worn myself out with my labours
11	*E*	and have had no respite.'
12	4	This is what you shall say to Baruch,
13	*B*	These are the words of the LORD:
14	*C*	What I have built, I demolish;
15	*D*	what I have planted, I uproot.
16	*b*	So it will be with the whole earth.
17	5	You seek great things for yourself.
18	*B*	Leave off seeking them;
19	*b*	for I will bring disaster upon all mankind,
20	*D*	says the LORD,
21	*E*	and I will let you live
22	*F*	wherever you go,
23	*G*	but you shall save your life and nothing more.

Figure 2: Jeremiah 45 as Normaltext.

Our method makes it easy to display original texts together with their translation, to relate scientific results to the text [*Beilisten*] and to combine several texts into a synopsis.

While making normalized synopses is admittedly tedious, the effort that has to go into a normalized text will soon pay off. This was clearly shown by our students of Old Testament exegesis using it as a standard in writing their first exegetical paper of their academic career.

The task of producing the *Normaltext*, usually with scissors and glue (the real one), brings the student into close contact with his text, the decisions to be made on its structure draw his attention, from the beginning, to certain exegetical questions,[5] and the displayed Hebrew text in the left column accompanied by its symmetrically shaped translation in the right column provides excellent access both to the text as a whole and to its parts, helping a great deal to control hypotheses found in the learned literature and to develop one's own.[6]

But the method is even worthwhile when used only to provide better access to some modern translation of the ancient text; rf. to the examples contained in *Verheißung und Versprechen*,[7] *1. Samuel 1,9–11* (p. 170), *Genesis 15,1–6* (p. 180), and *Numeri 6,22–27* (p. 201).

It is quite obvious from the examples mentioned so far that producing a normalized text by hand is a very tedious task, and so we developed a special macro for the typesetting of our *Leitfaden* to handle this kind of text. In connection with the table macros of DCF, we were even able to typeset our synopses rather automatically.[8] In this article we are going to discuss a corresponding approach for TEX and LATEX.[9]

[5] Two problem examples *1. Samuel 10,17–20* and *1. Samuel 10,24* are shown in N: *Abb.11* and *Abb.13*.

[6] It is often interesting to compare the segmentation according to our model with the conventional segmentation, rf. the example of *1. Samuel 6,21–7,3* in *Tafel 7* of the *Leitfaden*.

[7] R. W. / Hans Peter Hecht: Verheißung und Versprechen. Eine theologische und sprachanalytische Klärung. *Göttingen: Vandenhoeck & Ruprecht* 1986.

[8] The general background of the problem is presented in our article *"Verheißung und Versprechen" — A third generation approach to theological typesetting. TEX for Scientific Documentation, Proceedings of the Second European TEX Conference, Strasbourg, June 19–21, 1986. Lecture Notes in Computer Science. Heidelberg / Berlin: Springer (forthcoming)*, and we should like to refer the reader to this article both for the general background to the present paper and for the discussion of further details.

[9] Rf. Leslie Lamport: LATEX. A Document Preparation System. *Reading, Massachusetts etc. Addison-Wesley.* 1985.

4 Making a LaTeX environment

Our basic idea is to take advantage of the list making capabilities of LaTeX. Though LaTeX allows us to define a new list environment by using the `list` environment inside a `\newenvironment` command, our task is not as trivial as it might look at first sight, since we have to find a way to interfere with LaTeX's `list` environment in a controlled way.

The main task of the normal `list` environment is to give us control over line numbering, and we declare a `\newcounter{linecount}` to that end. This counter will be advanced automatically by every `\item` or replaced by an optional argument of an `\item[arg]` or left blank by an empty argument `\item[]`. To restart the counter in each new environment, we shall include in the definition a `\usecounter{linecount}`. To gain a more coherent display, we also have to reset the `\itemsep`.

4.1 Defining a new list environment

Next, we want our additional formatting to take place before the text of the item is processed, but not at the cost of rewriting the original macros. A little trick will help us here. It is built on the fact that the `\@item` macro which does the real work is finished with an `\ignorespaces` command. Mapping this control sequence to some other macro, we regain control at the end of this macro, just before the text of the the item is processed. Of course it is our first duty to remap `\ignorespaces` to its original state in the substitute macro.

In addition to the `\item` command already provided by the `list` environment, we want a similar command for headings, that do not belong to the source we are going to display, but help to denote the contents of the text or hold certain comments we want to make. Our `\head[arg]` command will also have an optional argument, which will allow us to control whether headlines are counted along with source lines or get some other marking. Unlike the normal `\item` command, it should read the text of the heading as an argument, so that we have it available for further use in running headings, in the table of contents etc., should the need arise some day.

Sometimes our linebreaking will not correspond to the inherited verse or chapter breaks. For these rare cases, we use the `\vl` (verselabel) macro, which will read the argument in the same way as `\item`, but put the verselabel in the text and link it with what follows by a tie.

At the end of our environment we will incorporate a test to make sure that the process of rushing up and down the levels of our staircases has properly come to a rest at ground level. Since that tends not to be the case in real life, we shall oblige ourselves with resetting any open levels explicit=ly with the aid of a `\reset` macro. Apart from ending the previous paragraph, this macro only has to contribute its argument so that the macros contained in it will execute by themselves.

Now we have already finished with the things that will allow us to start our new environment. But before the first item or head can be processed, we have to modify the paragraph making of the `list` environment. To understand the corresponding macro is rather complicated, but the crucial point can be easily seen looking at *Figure 5.3 'The format of a list'* from LaTeX *User's Guide & Reference Manual.* The drawing shows clearly that list items are based on a one-line-`\parshape`, the label being placed by other means. Since we want our source lines to be clearly marked as continuation lines by an indent of `\hangwidth` relative to the starting position of the text, and since that position depends on the constant width of our verse labelling and the varying width of the levels of speech and redaction, we have to use a two-line-`\parshape`. When beginning the environment, the constant parts of indentation must be calculated; these will be modified later by the varying parts.

Now we have at hand all the elements to define our new environment:

```
\newcounter{linecount}

\newenvironment{normaltext}{%begin of env.
\begin{list}{%default label:
\arabic{linecount}%
}{% begin of list declarations:
\usecounter{linecount}%
%
\def\item{\let
\@tempignorespaces= \ignorespaces
\let \ignorespaces=\makeverseline
\@ifnextchar [{\@item }{\@noitemargtrue
\@item[\@itemlabel]}}%
%
\def\head{\let
\@tempignorespaces= \ignorespaces
\let \ignorespaces= \makeversehead
\@ifnextchar [{\@item }{\@noitemargtrue
\@item[\@itemlabel]}}%
%
\let\reset=\@ureset
%
```

```
\let\vl=\makeverselabel
%
}% end of list declarations and env
% find out total width for indent
\tw@totalleftmargin= \@totalleftmargin
\tw@linewidth= \linewidth
\indentdiff= \versewidth
\advance \indentdiff by \versesep
\advance \indentdiff by \rlevel\rwidth
\advance \indentdiff by \llevel\lwidth
\advance \indentdiff by \hangwidth
\adjustparshape
\setlength{\itemsep }{\z@}
\setlength{\parsep }{\z@}
\setlength{\parskip }{\z@}
}%end of newenv begin parameter
{% test if red and lev are reset:
\ifnum\llevel= \z@
\else \adjusterror {l}\fi
\ifnum\rlevel= \z@
\else \adjusterror {r}\fi
\end{list}%
}%end of newenv end parameter
```

4.2 Extensions to the item command

We now have to specify what should be done after we wrestled control from the `\@item` command. First we shall fulfil our pledge to restore `\ignorespaces`. Then we shall advance our special `\verselinecount` and give it a chance to produce the 'A B C ...' numbering, unless it is drowned by something hidden in the `#1` argument that will be executed next. Its main purpose is to process chapter and verse count and level information, as we shall soon see.

By `\noindent` we make sure that the new item is started before we produce our own label. We then use our little trick of macro substitution another time to determine what is to be done at the end of the `\makenormlabel` macro to be executed next, because the same macro is also used for our headlines, but with a different continuation.

To make sure that no harm is done to `\ignorespaces` in case our macro is called directly, we initialize `\@tempignorespaces`.

The counterpart of the `\reset` macro can be specified much easier. To make sure that our last item cannot be affected any more by the parameters to come, we issue a `\noindent` and then simply use a LaTeX 'hack' to execute whatever will be contained in the following parameter.

```
\newcommand{\makeverseline}[1]{\let
  \ignorespaces=\@tempignorespaces
  \advance\verselinecount by \@ne
  \def\verselinelabel{$^{\em
  \@Alph{\verselinecount
  }}$}#1% execute parameters
  \let\makenormlabelend=\ignorespaces
  \noindent
  \makenormlabel}

\let \@tempignorespaces= \ignorespaces

\def\@ureset{\noindent
  \@iden} %LaTeX hack
```

The macro for headlines will insert a chapter mark if one was specified before, otherwise the field will be left blank. There is a subtlety not obvious at first glance: a chapter printed here is inherited from the previous context, while a chapter starting at this line would print in bold through the `#1` parameter. Finally, a macro to read the text of the heading will be called.

```
\newcommand{\makeversehead}[1]{\let
  \ignorespaces=\@tempignorespaces
  \def\verselinelabel{{\em \kapitel
  \/}}#1% execute parameters
  \let\makenormlabelend=\readversehead
  \noindent
  \makenormlabel}

\newcommand{\readversehead}[1]{%
  {\em #1\/}\ignorespaces}
```

The `\makenormlabel` macro will make sure that we are in horizontal mode (cf. *The TEXbook* Ex. 13.1), then produce redaction indentation, follow it with the verse line label together with a separator congenial to the label separator of the standard list environment, and finally append speech level indentation. Though in most cases both types of indentation will be just blank space, we have introduced two macros that will allow control of what goes into the indented space. As a default for testing purposes, we use `\dotfill` and `\hrulefill`. But these macros could also be used to specify names for redactional levels or vertical lines to show the depth of speech embedding. Right adjustment will be achieved using the standard `\makelabel` macro.

In some rare cases our choice of basic lines will not coincide with the cutting of verses in the source. In such a case, we will use a `\vl{\v{b}}` sequence in the text and execute it as `\makeverselabel`. First,

the parameter is read, then the label is put into the running text and connected to what follows with a tie. The tie (~) will ensure that the label is not separated from the following text by a possible linebreak.

```
\newcommand{\makenormlabel }{\leavevmode
  \hbox to \rlevel\rwidth {\rfill}\hbox
  to \versewidth{%
\makeverselinelabel {\verselinelabel}}%
\hskip \versesep
\hbox to \llevel\lwidth{\lfill}%
\makenormlabelend}

\let\makeverselinelabel= \@mklab

\newcommand{\makeverselabel }[1]{#1%
\verselinelabel ~\ignorespaces}
```

4.3 Declarations

Now we come to the boring part of the whole thing, making declarations and setting initial values. It should be noted however, which of our variables are counts, which are dimensions, and which are macros.

The shape of our display is mainly influenced by the different types of indent lengths. `\versewidth` depends mainly on the maximum width of verse or chapter numbers to be displayed, but the other parameters should be chosen according to readability considerations. A good strategy seems to start with the most important value of `\lwidth`, which indents embedded speech. A value of `1em`, representing roughly the width of two lowercase letters, seems to be sufficient. Then `\hangwidth` can be set to half that value, so that embedded text can still be distinguished from continuation lines of the governing text. `\rwidth`, the indentation for redactional parts of the text, should be made wide enough to mark the difference from text embedding very clearly, but not so wide that lines become too narrow and the reading connection to the governing text goes astray. The fields of indentation will be blank normally; for testing purposes, however, they can be filled with something, e.g. dots or rules.

```
\newcount\rlevel
 \newcount\llevel
 \newcount\verselinecount
\newdimen\versewidth
  \newdimen\versesep
  \newdimen\rwidth
  \newdimen\lwidth
  \newdimen\hangwidth
  \newdimen\indentdiff
  \newdimen\tw@totalleftmargin
  \newdimen\tw@linewidth

% set initial values
\rlevel=\z@ \llevel=\z@
\verselinecount=\z@
  \versewidth=1.2em
  \versesep=\labelsep
\lwidth=1em
  \hangwidth=0.5\lwidth
  \rwidth=4.0\lwidth
\let\lfill=\@empty
  % \def\lfill{\hrulefill}
\let\rfill=\@empty
  % \def\rfill{\hfill $\mid$}
  % \def\rfill{\dotfill}
```

4.4 Parameter macros

Now we come to the macros that will appear inside the optional arguments of `\item` and `\head`. The verse macro will make sure that our verselinecount 'A B C ...' will be replaced by the verse number or the 'b' in the starting line of the verse and its second half after the *atnach*. Note that chapters are not treated in terms of counts but of macros. This allows them to contain also letters.

The control sequences to switch the two different kinds of levels are gained by redefining those used to enclose math displays in LaTeX. Parentheses `\(...\)` are used for speech levels, square brackets `\[...\]` are used for redaction levels. All these macros can take an optional argument giving the number of level steps to be taken. This argument might also be used in a later stage of development to identify redaction levels by name.

```
\def\versehalf{b} % Text after Atnach, Hebrew
\def\versenumber{\z@}
\def\verselinelabel{\@empty}
 % \v is an accent
\renewcommand{\v}[1]{\if #1\versehalf \def
  \verselinelabel{$^{\versehalf }$}\else
  \verselinecount= \@ne \def
  \versenumber{#1}\def
  \verselinelabel{$^{\versenumber }$}\fi}

\def\kapitel{\@empty}
\newcommand{\k}[1]{\def \kapitel{#1}\def
```

```
  \verselinelabel{{\bf \kapitel}}}
\renewcommand{\(){\adjustcount {l}{\@ne}}
\renewcommand{\)}{\adjustcount {l}{\m@ne}}
\renewcommand{\[}{\def \@redlsign{+}%
\@ifnextchar [{\@redl }{\@noitemargtrue
  \@redl[\@ne]}}
\renewcommand{\]}{\def \@redlsign{-}%
\@ifnextchar [{\@redl}{\@noitemargtrue
  \@redl[\@ne]}}
\def\@redl[#1]{%
  \adjustcount{r}{\@redlsign #1}}
\renewcommand{\(}{\@switch {l}{+}}
\renewcommand{\)}{\@switch {l}{-}}
\renewcommand{\[}{\@switch {r}{+}}
\renewcommand{\]}{\@switch {r}{-}}
\newcommand{\@switch}[2]{%
  \@ifnextchar [{\@adjust{#1}{#2}}{%
    \@adjust {#1}{#2}[\@ne]}}
\def\@adjust#1#2[#3]{\adjustcount {#1}{#2#3}}

\newcommand{\adjustcount }[2]{\def
  \thiscount{\csname #1level\endcsname
  }\advance \thiscount by #2\ifnum
  \thiscount < \z@ \adjusterror {#1}\else
  \indentdiff= #2\csname #1width\endcsname
  \adjustparshape \fi}

\newcommand{\adjusterror }[1]{\message
  {**** wrong #1-level \the\thiscount in
  \normposition}%
  \csname #1level\endcsname= \z@}
\newcommand{\normposition }{list-line
  \thelinecount, Ch.\kapitel, V.\versenumber
  \@Alph{\verselinecount} on page \thepage}
```

4.5 Adjusting the parshape

Having executed all our level switching parameters, we can now specify what our paragraphs should look like. The main difference is, that our paragraphs will have a twofold shape; the first line is the same as in LaTeX lists, but the following ones are different, and we have chosen our variables accordingly.

```
\newcommand{\adjustparshape}{%
  \advance \tw@totalleftmargin by \indentdiff
  \advance \tw@linewidth by -\indentdiff
  \parshape \tw@
    \@totalleftmargin \linewidth
    \tw@totalleftmargin \tw@linewidth }
```

5 Examples

Now that we have understood the macros, we are going to explore some of their possibilities in the following examples.

5.1 Showing Redaction

If the unity of a text is disputed, we can show which parts of the text are due to redactional work. For our text *Jeremiah 45*, a corresponding hypothesis has been formulated by *Winfried Thiel*.[10] It is not our concern here to discuss the validity of his assumptions, but to show in one clear display (Fig. 3), what otherwise has to be gleaned from several pages of text. The translation is ours and done according to the principles of *distinktives Übersetzen*, which try to make visible the structure of the underlying Hebrew as far as possible.

5.2 The Greek text of Jeremiah 45

To display our text in Greek[11] (Fig. 4), we need no more than standard TeX.[12] There are some restrictions, however. First, we have to read this kind of Greek from input, since it implies redefining of category codes.[13] Second, there is no automatic hyphenation, though explicit hyphens can be specified. If columns become rather narrow, it will often look better and also save a lot of `underfull box` messages to use a `raggedright` environment.

5.3 A synopsis of the Greek text and a translation of Jeremiah 45

Our next step will be to combine the Septuagint text with its translation (Fig. 5), using the `minipage` environment.

Though this approach makes it easy to produce two columns, it does not guarantee a correct alignment of base lines, because extension lines due to automatic line breaking may be different for the two

[10] Winfried Thiel: Die deuteronomistische Redaktion von Jeremia 26–45. Mit einer Gesamtbeurteilung der deuteronomistischen Redaktion des Buches Jeremia. *WMANT 52. 1981.*

[11] In the Septuagint (LXX), it is placed at the end of chapter 51, thus bearing witness to a different text tradition.

[12] The macros to perform typesetting Greek are described in my article in note 8.

[13] Consequently, the `\fbox` macro to draw the frame around the figure has to be split into a begin and end part. In other cases it will be convenient to use the method of semi-parameters known from PLAIN footnotes and described in more detail in my article *Chapter mottos and optional semi-parameters for TeX and LaTeX. TUGboat 7,3 (1986) 177-185.*

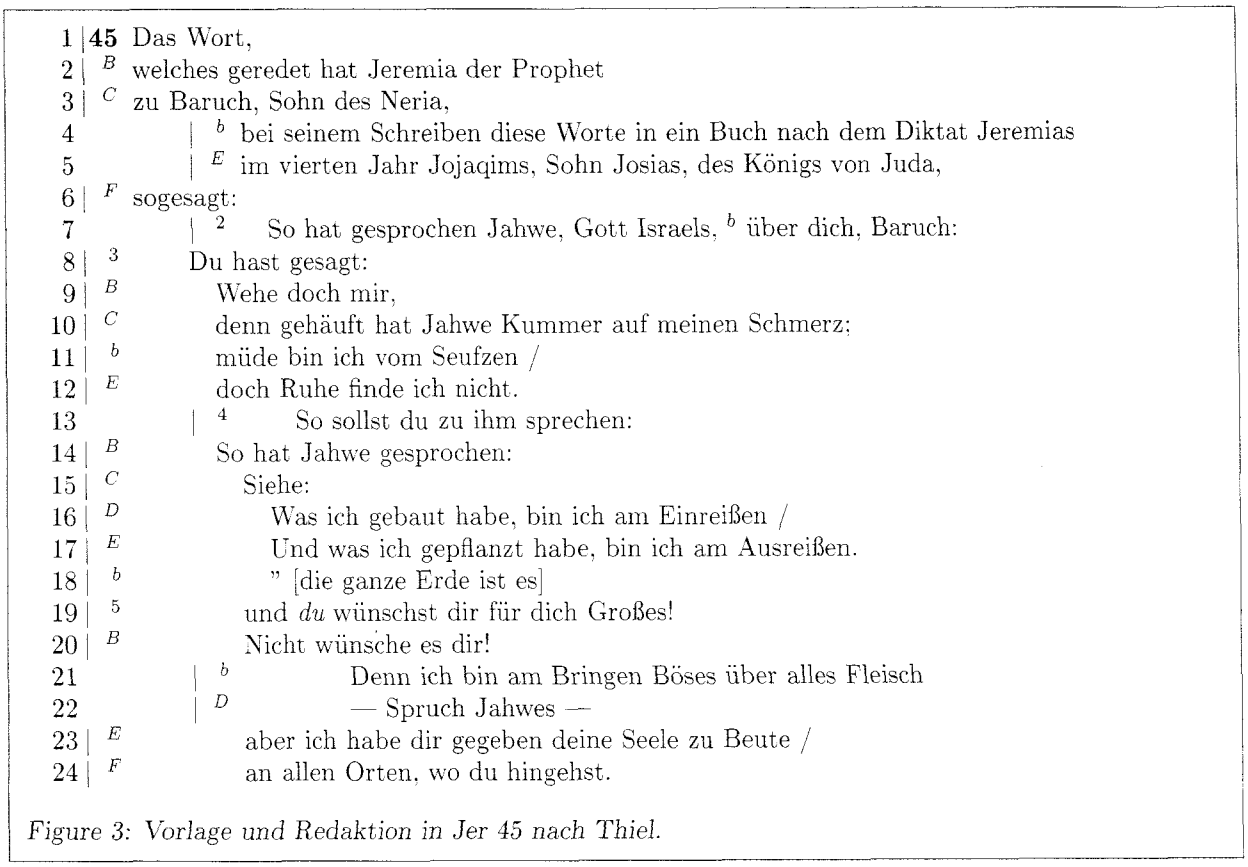

Figure 3: Vorlage und Redaktion in Jer 45 nach Thiel.

versions. The normal approach would be to regroup the synoptic texts so that the items of the different columns are presented row-wise. Though this would allow the use of `tabular`, the price would be breaking texts into pieces. On the other hand any almost automatic process of combining sequential texts will cost a lot of tricky programming and so will only be worthwhile for a larger project.

6 A verbatim of the Greek text

To show how the Greek text in Fig. 4 and Fig. 5 was produced, we give a verbatim of the corresponding input.

```
\begin{normaltext}
\item{\v{31}} {\greek \,<O lo1gov, }
\item{}  {\greek o9n e>la1lhsen Ieremiav
          o< profh1thv}
\item{}  {\greek pro3v Baroyx yi<o3n Nhrioy,}
\item{\v{b}}  {\greek o7te e4grafen toy3v
          lo1goyv toy1toyv e>n tw2j bibli1wj
          a>po3 sto1matov Ieremioy}
\item{}  {\greek e>n tw2j e>niaytw2j tw2j
          teta1rtwj tw2j Iwakim yi<w2j Iwsia
          basile1wv Ioyda}
\item{}
\item{\v{32}\()  {\greek Oy7twv ei5pen ky1riov}
 \vl{\v{b}}     {\greek e>pi3 soi1, Baroyx}
\item{\v{33}\() {\greek \,7Oti ei5pav}
\item{\() {\greek Oi4mmoi oi4mmoi,}
\item{} {\greek o7ti prose1qhken ky1riov
          ko1pon e>pi3 po1non moi,}
\item{\v{b}} {\greek e>koimh1qhn e>n
          stenagmoi2v,} /
\item{} {\greek a>na1paysin oy>x ey8ron,}
\item{\v{34}\)} {\greek ei5pon ay>tw2j}
\item{\() {\greek Oy7twv ei5pen ky1riov}
\item{\() {\greek \,>Idoy3}
\item{\() {\greek oy9v e>gw3 w>jkodo1mhsa,
          e>gw3 kaqairw2,} /
\item{}  {\greek kai3 oy9v e>gw3 e>fy1teysa,
          e>gw3 e>kti1llw;}
\item{\v{b}} %('\,' [die ganze Erde ist es])
\item{\v{35}\)}  {\greek kai3 sy3 zhtei2v
          seaytw2j mega1la?}
\item{}  {\greek mh3 zhth1shjv,}
\item{\v{b}}  {\greek o7ti i>doy3 e>gw3
          e>pa1gw kaka3 e>pi3 pa2san sa1rka,}
```

```
1  31 ῾Ο λόγος,
2  B  ὃν ἐλάλησεν Ιερεμιας ὁ προφήτης
3  C  πρὸς Βαρουχ υἱὸν Νηριου,
4  b  ὅτε ἔγραφεν τοὺς λόγους τούτους ἐν τῷ βιβλίῳ ἀπὸ στόματος Ιερεμιου
5  E  ἐν τῷ ἐνιαυτῷ τῷ τετάρτῳ τῷ Ιωακιμ υἱῷ Ιωσια βασιλέως Ιουδα
6  F
7  32   Οὕτως εἶπεν κύριος b ἐπὶ σοί, Βαρουχ
8  b b    ῾Οτι εἶπας
9  C        Οἴμμοι οἴμμοι,
10 D        ὅτι προσέθηκεν κύριος κόπον ἐπὶ πόνον μοι,
11 b        ἐκοιμήθην ἐν στεναγμοῖς, /
12 F        ἀνάπαυσιν οὐχ εὗρον,
13 34     εἶπον αὐτῷ
14 B        Οὕτως εἶπεν κύριος
15 C          ᾽Ιδοὺ
16 D            οὓς ἐγὼ ᾠκοδόμησα, ἐγὼ καθαιρῶ, /
17 E            καὶ οὓς ἐγὼ ἐφύτευσα, ἐγὼ ἐκτίλλω·
18 b
19 35           καὶ σὺ ζητεῖς σεαυτῷ μεγάλα;
20 B            μὴ ζητήσῃς,
21 b            ὅτι ἰδοὺ ἐγὼ ἐπάγω κακὰ ἐπὶ πᾶσαν σάρκα,
22 D            λέγει κύριος,
23 E            καὶ δώσω τὴν ψυχήν σου εἰς εὕρεμα /
24 F            ἐν παντὶ τόπῳ, οὗ ἐὰν βαδίσῃς ἐκεῖ.
```

Figure 4: The Greek text of Jer 45 (Jer 51,31–35 LXX).

```
1  31 ῾Ο λόγος,                                  1  45 Das Wort,
2  B  ὃν ἐλάλησεν Ιερεμιας ὁ προφήτης            2  B  welches geredet hat Jeremia der Prophet
3  C  πρὸς Βαρουχ υἱὸν Νηριου,                   3  C  zu Baruch, Sohn des Neria,
4  b  ὅτε ἔγραφεν τοὺς λόγους τούτους ἐν         4  b  bei seinem Schreiben diese Worte in ein
      τῷ βιβλίῳ ἀπὸ στόματος Ιερεμιου                Buch nach dem Diktat Jeremias
5  E  ἐν τῷ ἐνιαυτῷ τῷ τετάρτῳ τῷ               5  E  im vierten Jahr Jojaqims, Sohn Josias,
      Ιωακιμ υἱῷ Ιωσια βασιλέως Ιουδα                des Königs von Juda,
6  F                                             6  F  sogesagt:
7  32   Οὕτως εἶπεν κύριος b ἐπὶ σοί,           7  2    So hat gesprochen Jahwe, Gott
        Βαρουχ                                          Israels, b über dich, Baruch:
8  b b    ῾Οτι εἶπας                             8  3    Du hast gesagt:
9  C        Οἴμμοι οἴμμοι,                      9  B      Wehe doch mir,
10 D        ὅτι προσέθηκεν κύριος κόπον         10 C      denn gehäuft hat Jahwe Kummer
              ἐπὶ πόνον μοι,                              auf meinen Schmerz;
11 b        ἐκοιμήθην ἐν στεναγμοῖς, /          11 b      müde bin ich vom Seufzen /
12 F        ἀνάπαυσιν οὐχ εὗρον                12 E      doch Ruhe finde ich nicht.
```

Figure 5: The Greek text of Jer 45 (Jer 51,31-35 LXX).

```
\item{}  {\greek le1gei ky1riov,}
\item{}  {\greek kai3 dw1sw th3n uyxh1n soy
          ei>v ey7rema} /
\item{}  {\greek e>n panti3 to1pwj, oy8 e>a3n
          badi1shjv e>kei2.}
\reset{\)[4]}
\end{normaltext}
```

7 Conclusion

Perhaps it has become clear from our examples that displayed texts from ancient sources call for attention and handling similar to that of mathematical formulas, being the core and the aim of exegetic efforts. Just as with formulas, displaying them according to accepted rules will help to bring out their internal structure, thus contributing to their readability.

Because of the specific rules to be followed, it is not a trivial task to get the shape of such texts right. But while the formatting of mathematics is built into TEX, ours has to be done with macros.

Alas, our programming betrays a splendid inconsistency of mixing between TEX and LATEX commands. It is really bad style and should encourage others to do better. But what might be more interesting is our somewhat peculiar use of optional arguments to perform unforeseen actions on our layout, a feature that still waits to be explored to its full depth.

While other contributions like the *chapter mottos* mentioned above fall into the general area of typesetting, our problem is an example for the field of *philological* typesetting. Interest in such applications has grown recently, and though there are some severe restrictions in TEX, e.g. with respect to the reverted formatting necessary for semitic languages like Hebrew,[14] in general it seems to me to be a reliable base to join efforts in further development. That is why we decided at the Strasbourg conference, this past summer, to arrange for the formation of a TEX philology group.[15] Taking the present article as an example for philologic activities, I should like to stress two axioms on which it is based:

Axiom 1 *New developments should be published rapidly and with fully documented source.*

The slow progress of many projects in the field of the Bible and the computer, in my opinion, is due to the lack of any publishing of that kind. Now we have got the opportunity to do a better job.

Axiom 2 *Development should be based on or at least be compatible with* LATEX.

The use of LATEX is a good way to free oneself from many problems of general formatting that are already solved there in a clear and consistent way. LATEX programming philosophy also contains some useful techniques that may be borrowed for other purposes, thus speeding up macro development. Finally, LATEX seems to be the only macro package that has a good chance to emerge as *the* standard, sort of a counterpart to IBM's General Markup Language.

Axiom 3 *Development in philological typesetting should be based on firm linguistic ground.*

It is the aim of our article to give an example for the close relationship between exegetical work and linguistics on the one hand and linguistics and philological typesetting on the other.

---

[14] Rf. to note 8.

[15] Rf. *Barbara Beeton / R. W.: Towards a TEX Philology Group. TUGboat 7,3 (1986) 132-133.*