The TEXshade package[*]

Typesetting
nucleotide and peptide alignments

Eric Beitz[†]

v1.18; 2008/04/15

Abstract

Setting alignments of nucleotides and peptides for publication or presentation purposes is usually a time consuming two-step process. First, a scientific software is used for the calculation of the alignment. This is done in a few minutes. Then, in order to highlight special sequence relationships and to label positions and regions of interest a second software with high quality output capability is needed. Manipulating sequence alignments with standard word processing or graphics programs takes its time—often several hours—and simple layout changes such as re-breaking lines, say from 50 to 40 residues per line, elongate the working time considerably.

TEXshade is an alignment shading software written in TEX/LATEX which can process multiple sequence alignments in the MSF, ALN and FASTA file format. It provides in addition to common shading algorithms special shading modes featuring functional aspects, e. g. charge or hydropathy, and a plenitude of commands for handling shading colors, text styles, labels, legends and even allows the user to define completely new shading modes. TEXshade combines highest flexibility and the habitual TEX output quality—with reasonable time expenditure.

Contents

1 Package Overview

After `texshade.ins` is run through TEX the following files should appear in the directory:

`texshade.sty`	the style file with all TEXshade commands
`texshade.def`	an example parameter file with the standard parameter settings
`AQPDNA.MSF`	an example nucleotide alignment (MSF-format)
`AQPpro.MSF`	an example protein alignment (MSF-format)
`AQP_TC.asc`	an example T-Coffee shading file (`score_ascii`-format)
`AQP2spec.ALN`	a further protein alignment (minimal ALN-file)
`AQP1.phd`	secondary structure information (PHD-format)
`AQP1.top`	topology data extracted from `AQP1.phd`
`AQP1_HMM.sgl`	topology information (single line, HMMTOP-format)
`AQP1_HMM.ext`	topology information (extended, HMMTOP-format)
`standard.cod`	standard genetic code definitions
`ciliate.cod`	ciliate macronuclear genetic code

The alignment file examples as well as the topology data file are needed for TEXing this documentation and can serve as illustrations for the MSF and ALN file format.

The following subsections give an overview on the capabilities of the TEXshade package. All commands are described in detail later on.

1.1 Version History

v1.18 2008/04/15

Correction: several bug fixed concerning featurename display, sequence ordering and numbering.

Introduction: (a) T-Coffee shading information can be loaded and put on the alignment.[1] The conservation data can also be displayed in the consensus as well as feature color scales and bar plots. (b) Two more feature lines were added on the top and at the bottom (`ttttop`, `tttop`, `bbbottom`, `bbbbottom`). (c) The `startnumber` and `setends` commands have been fused; either command can set both, a new start number as well as end definitions of the sequence section to be displayed.

---

[1]Suggestion by Florian Mertes.

v1.17 2007/06/19

Introduction:[2] (a) A second threshold percentage was introduced in order to label two levels of conservation in 'identical' and 'similar' mode. This is achieved by setting an optional parameter in \threshold or in \allmatchspecial, or by using a number as an optional parameter in \shadingmode. (b) The feature lines can be additionally labeled with a name left or right of the feature. This is handled using \showfeaturename, \showfeaturestylename, \hidefeaturename, \hidefeaturestylename, \hidefeaturenames, \hidefeaturestylenames. The color of such names can be changed with \featurenamecolor, \featurestylenamecolor, \featurenamescolor, \featurestylenamescolor, Font styles in feature names can be set as usual, e.g. \setsize{featurenames}{large} or \featurestylenamesrm.

v1.17 2007/06/19

Correction: T<small>E</small>Xshade crashed when calculating conservation using sequences with untypical residue characters, such as "X". Fixed. The reference sequence in diverse mode can now be shaded with \conservedresidues and, if active, \allmatchresidues.[3]
Introduction:[4] (a) A command was introduced, i.e. \exportconsensus which produces a pymol script file for coloring a 3D model according to T<small>E</small>Xshades conservation calculation. (b) With namerulerpos labels of the ruler can be exchanged by a string. (c) Various parts of the alignment can now be hidden by \hideblock.
New home: T<small>E</small>Xshade, T<small>E</small>Xtopo, and B<small>IO</small>T<small>E</small>X have a new home: www.pharmazie.uni-kiel.de/chem/Prof_Beitz/biotex.html.

v1.15 2006/06/27

Correction: Sequence and subfamily logos can now be plotted with pdflatex; pstricks is not needed anymore.

v1.14 2006/05/11

Introduction: In order to better recognize relevant positions in a subfamily logo [14], a bit-value can now be set by \relevance above which

---

[2] Asked for by Marat Kazanov.
[3] For this and suggesting namerulerpos credit to Marco Pasi.
[4] Both extensions were suggested by Phillip Hahn.

a deviation is considered relevant. Such positions can be labeled with a symbol by `\showrelevance` and hidden by `\hiderelevance`.

v1.13 2006/02/23

Corrections: Helix symbols in feature lines were not drawn correctly if the standard Computer Modern Font was changed to another one, e.g. Palatino.[5] Fixed. Unintended gaps occurred due to numbers at the end of lines in Clustal W alignment files. Fixed. The limitations in the number of sequences per alignment have finally been overcome by a more restrictive use of counter variables.
Introductions: (a) The numbering can now be displayed—in addition to left or right—on both sides of the alignment with the optional parameter `{leftright}` in the `\shownumbering` command (p.52). (b) TeXshade tries to guess the sequence type, i. e. protein or nucleotide, if not defined by the user. (c) Plotting of sequence logos has been implemented (p.49). Logos can be shown in addition to or together with the consensus, or alone without any alignment sequences. (d) The ruler numbering can be rotated in order to make labeling of every position possible. (e) A new way to visualize subfamily characteristics has been implemented, i.e. subfamily logos (p.51) [14].

v1.12 2005/09/20

Corrections: When regional labeling with `\shaderegion`, `\emphregion`, `\tintregion`, or `\frameblock` was combined with `\setends` incorrect output was produced lacking the labeling.[6] Other minor fixes.
Introductions: An additional optional parameter for setting consensus colors was implemented in the `\showconsensus` command (p.47). This even allows one to use color scales illustrating sequence conservation in the consensus line.

v1.11 2005/04/13

Corrections: Bounding boxes with `\frameblock` had a wrong height when `\separationline`s were used. Other minor fixes.
Introductions: (a) An additional parameter for setting individual bar and arrow thicknesses in feature lines has been introduced. (b) Additional parameters for setting the frame color and thickness of boxes in feature lines have been implemented. (c) Three more color scales have

---

[5]Thanks to Markus Heller
[6]Discovered by Chris Page.

been defined: `RedBlue`, `RedGreen`, and `HotCold`. (d) Plotting of amino acid features (`hydrophobicity`, `molweight`, `charge`) as bar graphs or color scales. (e) Plotting of protein sequence `conservation` as bar graph or color scale[7]. (f) Color scales can be used for shading the consensus sequence according to protein sequence conservation. (g) Separate command for stretching color scales `\colorscalestretch`.

v1.10 2005/03/29

Corrections: Plotting of color scales and bar graphs has been sped up by more than a factor of 10.[8]
Introductions: (a) More colors have been introduced, i.e. even lighter versions of the existing PostScript colors 'LightLight' plus color name and 'LightLightLight' plus color name. (b) Sequence stretches and blocks can be tinted for labeling purposes `\tintreqion`, `\tintblock` and `\tintdefault`. (c) A new feature label style `{restriction}` has been introduced. (d) Java-typical 'NaN' values are now allowed in data files for bar graphs and color scales.

v1.9 2005/02/08

Corrections: TEXshade version 1.8 introduced an incompatibility with TEXtopo. This problem was identified by Meike Schmedt and has been fixed.
Introductions: (a) A short version of the figure caption can now be defined for display in the list of figures[9] `\shortcaption{⟨text⟩}`. (b) A colored frame can be drawn around a sequence block for labeling purposes with the command `\frameblock`.[10] (c) A new look for feature arrows has been implemented with scalable line thickness and a new end style 'ball'. (d) HMMTOP topology predictions can now be included for plotting feature lines with information on the location of the transmembrane domains.[11]

v1.8 2004/08/26

Corrections: Only minor bugs were fixed.
Introductions: (a) More colors have been designed, i.e. 'light' versions of the existing PostScript colors. (b) Three color ramps in 5% steps

---

[7]Ahmad Mirza asked for (e) and (f), great suggestion!
[8]This and (d) I owe again to Christoph Gille.
[9] Meike, here you go . . .
[10]Alan Robinson, this is for you.
[11]Implemented after a request by Steffen Moeller.

have been introduced: i) Blue-Red, ii) Green-Red and iii) Cold-Hot. (c) Two new feature label styles `bar` and `color` have been introduced which allow one to display number values as bar graphs or color scales along the alignment[12].

v1.7 2004/01/05

Corrections: Several bugs were fixed. In gaps the wrong character was plotted in 'donotshade' mode. Gaps were colored incorrectly when a single sequence was set as consensus. Another 'donotshade' problem was solved which led to a halt of the LaTeX run[13]. Due to several requests, the gap and match labels in `diverse` mode were switched ('-' in gaps; '.' at matching positions) in order to follow convention. *Introduction:* T$_{\!E}$Xshade speaks spanish (`\spanishlanguage`). Necessary translations were contributed by Mikel Egaña Aranguren. A new feature label style `helix` has been introduced.

v1.6 2002/03/26

Corrections: The unnecessary restriction to the DVIPS driver for `color.sty` has been removed[14]. Any color.sty compatible driver option can be given with the `\usepackage{texshade}` call and is then passed to the `color` package. The '`\namecolor`' and '`\numbercolor`' commands do now support sequence lists.[15] *Introductions:* (a) The FASTA file format is supported by T$_{\!E}$Xshade as alignment inputs. (b) Two commands set the space between sequence blocks either to be flexible (as so far) '`\flexblockspace`' or the be fixed '`\fixblockspace`'. (c) One can now refer to sequences by their name in addition to the number in the input file. (d) Using '`\firstcolumnDSSP`' and '`\secondcolumnDSSP`' one can choose which of the first to columns should refer to the sequence numbering (the second column remains default setting)[16].

v1.5a 2001/03/08

Corrections: 'X's in the alignment file caused a run-time error. Fixed. *Introductions:* (a) The vertical space between feature lines can be controlled by four new commands: `\ttopspace`, `\topspace`,

---

[12]Inspired by Christoph Gille's `STRAP`

[13]Thanks to Jeferson J. Arenzon and Naomi Siew

[14]As suggested by Eckhart Guthöhrlein.

[15]Thanks to Denys Bashtovyy.

[16]c and d were suggested by Christoph Gille.

`\bottomspace` and `\bbottomspace`[17]. (b) It is now easily possible to add a caption to the alignment with the `\showcaption` command. (c) TeXshade stores the sequence lengths in the `.aux` file in order to have correct breaks of the gaps after the sequences.

v1.4&4a 2000/9/12 & 2000/10/3

Introductions: (a) The alignment legend can now be moved by the command '`\movelegend`'. (b) In commands with parameters that contain series of sequence numbers, e. g. `\orderseqs`, a dash can be used, e. g. `{1-3,6-4,7}` instead of `{1,2,3,6,5,4,7}`.

v1.3a&b 2000/7/28 & 2000/7/30

Introductions: (a) It is now possible to force TeXshade to display gap symbols before and after the actual sequence by the commands '`\showleadinggaps`' and '`\hideleadinggaps`' (4.5.1). (b) The sequence names input routine is now more tolerant concerning special characters.

v1.3 2000/3/3

Corrections: Line scrambling occured when features where set in the `ttop` row without a feature in the `top` row. Fixed. The incompatible command '`\language`' with the `babel` package has been replaced by '`\germanlanguage`' and '`\englishlanguage`'[18].
Introductions: (a) Now, translations of sequence stretches are possible. Either nucleotide or amino acid sources can be translated. This is done by the new `{translate}` option for the feature command. (b) The codons are defined by the new command '`\codon`'. Complete codon sets can be loaded by '`\geneticcode`'. (c) Further, the size and style of the nucleotide triplets of backtranslations can be set by '`\backtranslabel`' and '`\backtranstext`'. (d) Two more feature counter styles were introduced: '`\Romancount`' and '`\romancount`'. (e) TeXshade is now compatible with TeXtopo, a new TeX software for drawing and shading topology plots of membrane proteins.

v1.2a 1999/6/24 (not released)

Minor corrections: '`\namecolor`' and '`\numbercolor`' are now really correctly reordered. Brackets (and) are now allowed in sequence names. The option `{case}` in '`\funcshadingstyle`' works now.

---

[17]Suggested by Ulrike Folkers.
[18]Thanks to Eckhart Guthöhrlein.

v1.2 1999/6/12

Corrections: (a) Functional group definitions of more than seven groups produced an error when displaying group number eight. These residues where skipped in the alignment. Fixed.

Introductions: (a) Protein secondary structure files in the DSSP, STRIDE and PHD format can be included and displayed automatically within the alignment by '`\includeDSSP`' (and similar commands for STRIDE, PHDsec and PHDtopo, 4.6.5). (b) Which types of secondary structures are to be included or skipped in the alignment is chosen by '`\showonDSSP`' and '`\hideonDSSP`' (and respective commands for STRIDE, PHDsec and PHDtopo). (c) The appearance of the labels is defined by '`\appearance`'. (d) Internal counters for repeatedly occuring structure types can be activated by '`\numcount`', '`\alphacount`' and '`\Alphacount`'. All commands are described in 4.6.5.

v1.1 1999/5/26

Corrections: (a) The activation of '`emphregion`' lead to an emphasized following alignment. This has been corrected. (b) '`\namecolor`' and '`\numbercolor`' were not reordered with the command '`orderseqs`'. Fixed. (c) Sequence gaps at the beginning or the end of a sequence, i.e. before the first and after the last residue where labeled with the gap symbol. Now these positions are left blank.

Introductions: (a) In order to treat the preceeding and sequence following gaps correctly, TeXshade needs to know the length of the sequences. Therefore, the command '`\seqlength`' was introduced (4.5). (b) With '`\gapcolors`' (also 4.5) the color selection for gap symbols is independent from non conserved residues. (c) The divisions of the ruler where so far fixed to 10. Now, this value is changeable by '`\rulersteps`' (again 4.5). (d) '`\hideresidues`' and '`\showresidues`' turn off or on the residue names, i.e. one can choose between a display of shaded boxes only or with letters in the boxes (4.5.2). (e) The changes (c) through (d) were necessary for the introduction of '`\fingerprint`'. This command allows one to display the complete sequence in one line for an easy survey of the alignment (4.5.4).

v1.0 1999/5/12

First release.

1.2 LaTeX basics

1.2.1 Typesetting documents using LaTeX

In order to use any of the macros provided by the BioTeX-project (TeXshade/TeXtopo) efficiently a basic understanding of the TeX typesetting system and its usage is required. Several books are available on this topic, but a rather quick and easy introduction is the *Not so short introduction to LaTeX*. This document is available from all Comprehensive TeX Archive Network (CTAN) servers, e.g. from `ftp://ftp.dante.de/pub/tex/documentation/lshort/`, in many different languages and formats besides LaTeX, such as PostScript and on-line viewable PDF. I also put a link from the BioTeX (TeXshade/TeXtopo) homepage to the document collection (`http://pharmazie.uni-kiel.de/chem/Prof_Beitz/BioTeX`).

1.2.2 Memory shortness when using TeXshade

If you are using TeXshade to align several large sequences (about 1000 residues/sequence), LaTeX will probably stop compiling and quit with one of the following messages:
`!\ TeX capacity exceeded, sorry [main memory size=384000]`
or
`!\ TeX capacity exceeded, sorry [stack size=300].`
TeX allocates space for different kinds of internal variables. Setting alignments needs lots of memory, usually more than for typesetting plain text. Thus, the parameter settings of a standard TeX installation might not be sufficient for certain projects. This manifests in TeX error messages about insufficient memory and the setting process is interrupted. There is no reason to be concerned. The parameters can be set by hand. Unfortunately, each TeX system hides its default parameter file in a different place in the system.
In the following, an excerpt from the FAQ-list to TeXshade is added. This explains how to increase the settings in OzTeX for the Macintosh, MikTeX for Windows and teTeX for *NIX TeX distributions. Please contribute to this list!

1. **OzTeX 4.0 for the Macintosh:**

   Find the file 'OzTeX:TeX:Configs:Default'. This file contains all memory settings. Look for the section '% TeX parameters' and increase the values that TeX complains about during the run. You will have to restart OzTeX before the changes are active.

For older versions of OzTEX the configuration file has the same name but the path is somewhat different.

2. **teTEX for *NIX:** (contributed by Joerg Daehn)

   Find the file: '/usr/share/texmf/web2c/texmf.cnf' or use

   `locate texmf.cnf` at the command prompt to find it.

   Login as super user. Backup 'texmf.cnf' in case you destroy something and then open the 'texmf.cnf' file in your favorite text editor and use its search function to locate `main_memory`. This variable is set to 384000. Change this to some higher value, i.e. 4000000 (works fine for me!). The total amount of memory should not exceed 8000000, so check the other values in that section.

   Next, you want to change the stack size. Search for `stack_size`. This will be set to 300. I changed it to 4000 and it works fine.

   There might be complains by TEX about further specific parameters such as `stack_size`. You find all those in the same file.

   After this you have to run 'texconfig init'.

   Logout as root.

   After this all should be set for large alignments. Happy TEXing!

   The information on how to achieve this was derived from a mail in the teTEX mail archive. The original question was posted by Pascal Francq and answered by Rolf Nieprasch.

3. **MiKTEX for Windows:**

   The MiKTEX documentation describes very detailed how the memory settings can be changed. In brief, you must locate the configuration file 'miktex/config/miktex.ini'. In the [MiKTeX] section of this file you find all the parameters you need, e.g. `mem_min`, `mem_max`, `buf_size`, `stack_size` etc.

   It appears, that the standard settings of MiKTEX are bigger than that of other TEX installations, so it may not always be necessary to increase the values.

1.3 System requirements

TEXshade requires LATEX 2$_\varepsilon$ with `color.sty` and `graphics.sty` for shading. For arrows in the feature line (p.60) the AMS Math style is needed. David Carlisle's `color.sty` is part of the Standard LATEX 'Graphics Bundle' [1]. This and the other packages can be downloaded from any TEX archive, e.g. `ftp.dante.de`; usually they are included in a comprehensive TEX installation.

The `color` style allows one to use several [⟨*options*⟩], e.g. `dvips`, `pdftex` or `dviwin`. These provide the commands which different devices/programs need to display colored output. It is advisable to make yourself familiar with the `color.sty` manual. You should define a default driver in the file `color.cfg`. Since there is no direct call of `color.sty` by the user, the option can be stated when TEXshade is loaded, see next subsection. If no option is stated the `DVIPS` driver will be loaded.

With the `[dvips]` option the output DVI-file can be converted to POSTSCRIPT using the `DVIPS` program and can later be viewed or printed with the public domain GHOSTVIEW program which is available for almost all computer platforms. Further, more and more standard TEX viewers are to a certain extent POSTSCRIPT compatible.

1.4 The `texshade` environment

The commands provided by the TEXshade package are enabled by the following command in the document header section:

> `\usepackage[`⟨*option*⟩`]{texshade}`

Make sure that the file 'texshade.sty' is present in a directory searched by TEX (see the installation notes in the file 'texshade.txt'). The ⟨*option*⟩ given here is passed to `color.sty` which handles the color commands for a particular output device, see previous subsection and the `color.sty` manual.

The TEXshade package provides only one single new environment: `texshade`. This environment has one mandatory and one optional argument, both of them designating file names which must be present in a directory searched by TEX. The required file ⟨*alignmentfile*⟩ contains the aligned nucleotide or peptide sequences (see section 2). This file is needed, because TEXshade does no alignment by itself, it has to

take a preprocessed alignment as input. The optional file is a parameter file (section 3) with definitions for the customized calculation of the consensus, special sequence features or labels etc. In this parameter file all TEXshade commands which are allowed in the texshade environment can be used and are fully functional. Within the environment further TEXshade commands can be given to replace or complete settings from the parameter file.

Thus, setting an alignment with TEXshade is as simple as this:

> \begin{texshade}[⟨*parameterfile*⟩] {⟨*alignmentfile*⟩}
>
> *further* TEXshade *commands, if needed*
>
> \end{texshade}

1.5 Shading modes predefined in this package

1.5.1 Identity mode

This basic type of shading is provided by almost any alignment program. All identical residues at a position are shaded if the number of matching residues is higher than a given threshold (default is 50%).

```
AQP1.PRO   TLGLLLSCQISILRAVMYIIAQCVGAIVASAIL   112
AQP2.PRO   TVACLVGCHVSFLRAAFYVAAQLLGAVAGAAIL   104
AQP3.PRO   TFAMCFLAREPWIKLPIYTLAQTLGAFLGAGIV   112
AQP4.PRO   TVAMVCTRKISIAKSVFYITAQCLGAIIGAGIL   133
AQP5.PRO   TLALLIGNQISLLRAVFYVAAQLVGAIAGAGIL   105
```

Code:

```
\begin{texshade}{AQPpro.MSF}
   \setends{1}{80..112}
   \hideconsensus
\end{texshade}
```

If you like, positions where conservation is very high (here $\geq 80\%$) can be shaded in a special color and the consensus can be shown with or without shading according to the degree of conservation:

14

```
AQP1.PRO   TLGLLLSCQISILRAVMYIIAQCVGAIVASAIL    112
AQP2.PRO   TVACLVGCHVSFLRAAFYVAAQLLGAVAGAAIL    104
AQP3.PRO   TFAMCFLAREPWIKLPIYTLAQTLGAFLGAGIV    112
AQP4.PRO   TVAMVCTRKISIAKSVFYITAQCLGAIIGAGIL    133
AQP5.PRO   TLALLIGNQISLLRAVFYVAAQLVGAIAGAGIL    105
consensus   T.A.l....iS.lravfY..AQ.lGAi.GAgIL
```

X	non conserved
X	$\geq 50\%$ conserved
X	$\geq 80\%$ conserved

Code:

```
\begin{texshade}{AQPpro.MSF}
    \threshold[80]{50}
    \setends{1}{80..112}
    \showconsensus[ColdHot]{bottom}
    \defconsensus{.}{lower}{upper}
    \showlegend
\end{texshade}
```

1.5.2 Similarity mode

In many cases it is expedient—mostly when comparing protein sequences—to shade also residues which are not identical but similar to the consensus sequence. Consider a position where three out of five residues are basic arginines and two more residues are also basic but lysines. In similarity mode TEXshade shades similar residues in a different color to distinguish them from the consensus residue. Even when none of the residues alone reaches the threshold but a group of similar residues does these are shaded in the 'similarity' color. This case is given for instance when at a position in a five sequence alignment two aliphatic valines and two also aliphatic isoleucins are present and the threshold is set to 50%. Neither residue exceeds this percentage but as a group of similars they do.

In grayscale printouts some colors of the following alignment may appear undistinguishable. Don't worry if you usually use grayscale—all colors/grays can be selected freely (see 5).

first case (see text)

AQP1.PRO	TLGLLLSCQISILRAVMYIIAQCVGAIVASAIL	112
AQP2.PRO	TVACLVGCHVSFLRAAFYVAAQLLGAVAGAAIL	104
AQP3.PRO	TFAMCFLAREPWIKLPIYTLAQTLGAFLGAGIV	112
AQP4.PRO	TVAMVCTRKISIAKSVFYITAQCLGAIIGAGIL	133
AQP5.PRO	TLALLIGNQISLLRAVFYVAAQLVGAIAGAGIL	105

second case (see text)
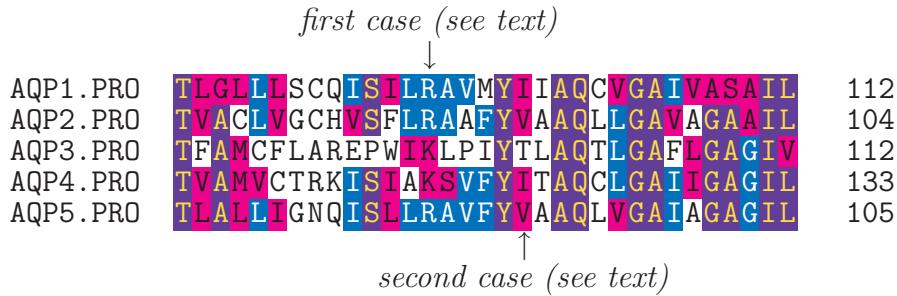
Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode{similar}
   \threshold[80]{50}
   \setends{1}{80..112}
   \hideconsensus
   \feature{top}{1}{93..93}{fill:$\downarrow$}{first case (see text)}
   \feature{bottom}{1}{98..98}{fill:$\uparrow$}{second case (see text)}
\end{texshade}
```

Probably you know this kind of shading from the public domain program BoxShade by KAY HOFMANN or from the Macintosh version MacBoxShade by MICHAEL D. BARRON. TEXshade provides the same functionality—and goes truly beyond—for the TEX community.

1.5.3 T-Coffee shading

TEXshade's capabilities of calculating alignment shadings are limited. T-Coffee (www.tcoffee.org) is a sophisticated alignment/shading software. You can apply shading from T-Coffee in TEXshade by loading the shading information file (score_ascii) generated by T-Coffee.

feat-cons		
AQP1.PRO	GSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL	63
AQP2.PRO	GSALQWA...SS....PPSVLQIAVAFGLGIGIL	55
AQP3.PRO	GSVAQVVLSRGTH....GGFLTINLAFGFAVTLA	63
AQP4.PRO	GSTINWG...GSENPLPVDMVLISLCFGLSIATM	84
AQP5.PRO	GSALKWP...SA....LPTILQISIAFGLAIGTL	56
consensus		

16

Code:

```
\begin{texshade}{AQPpro.MSF}
    \shadingmode[AQP_TC.asc]{T-Coffee}
    \setends{1}{30..63}
    \hideconsensus
    \feature{top}{1}{30..63}{color:conservation[T-Coffee]}{}
    \showfeaturestylename{top}{feat-cons}
    \showconsensus{bottom}
\end{texshade}
```

1.5.4   Diversity mode

Contrary to the above described modes this shading style displays sequence differences. Thus, it is most suitable for comparing very similar sequences, e. g. species variants of a protein.

One sequence is used as consensus. Matching residues in other sequences are blanked out, mismatches are shown in lowercase.

<div align="center">

AQP2 species variants

</div>

```
                       80            90          100
Bos taurus         SFLRAVFYVAAQLLGAVAGAALLHEITPPAIRG
Canis familiaris   .....a.......................hv..
Dugong dugong      .....t..l...............i.......d...
Equus caballus     .....a.......................d..r
Elephas maximus    .....t..l.................l...d...
```

Code:

```
\begin{texshade}{AQP2spec.ALN}
    \seqtype{P}
    \shadingmode{diverse}
    \setends{1}{77..109}
    \featureslarge
    \feature{top}{1}{77..109}{}{AQP2 species variants}
    \namesrm\namessl
    \hidenumbering\showruler{top}{1}
    \shownames{left}
    \nameseq{1}{Bos taurus}
    \nameseq{2}{Canis familiaris}
    \nameseq{3}{Dugong dugong}
    \nameseq{4}{Equus caballus}
    \nameseq{5}{Elephas maximus}
    \frameblock{1}{82..82,106..106}{Red[1pt]}
\end{texshade}\label{frame}
```

1.5.5   Functionality modes

Displaying functional peptide similarities is one of T_EXshade's strong capabilities. Six functional shading modes are predefined; further user specific modes can easily be created. The examples may not look very impressive when printed in grayscale. Enjoy them on your screen or use color printouts. As mentioned before, all colors can be changed to others or to grays without restrictions (see chapter 5).

charge: residues which are charged at physiological pH (7.4) are shaded if their number at a position is higher than the threshold

```
AQP1.PRO  GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL   170
AQP2.PRO  AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL   162
AQP3.PRO  GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF   186
AQP4.PRO  GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL   191
AQP5.PRO  AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL   163
```

X   acidic (−)
X   basic (+)

Code:
```

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[charge]{functional}
   \setends{1}{138..170}
   \feature{top}{3}{153..165}{bar[-50,50]:-50,-45,%
      -40,-30,-20,-10,0,10,20,30,40,45,50}{}
   \feature{top}{3}{167..186}{color:5,10,15,20,25,30,35,%
      40,45,50,55,60,65,70,75,80,85,90,95,100[ColdHot]}{}
   \showlegend
\end{texshade}
```

hydropathy: discrimination between acidic and basic, polar uncharged and hydrophobic nonpolar residues

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[hydropathy]{functional}
   \feature{top}{1}{158..163}{brace}{tinted}
   \tintblock{1}{158..163}
   \setends{1}{138..170}
   \showlegend
\end{texshade}
```

structure: displays the potential localization within the tertiary structure of the protein

19
```

transmembrane domain 4   *trans. dom. 5*

```
                                  G M   M M G
                                  A G   G G G A
                                  Y N   N N N Y
                  α-helix
AQP1.PRO   GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL   170
AQP2.PRO   AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL   162
AQP3.PRO   GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF   186
AQP4.PRO   GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL   191
AQP5.PRO   AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL   163
```
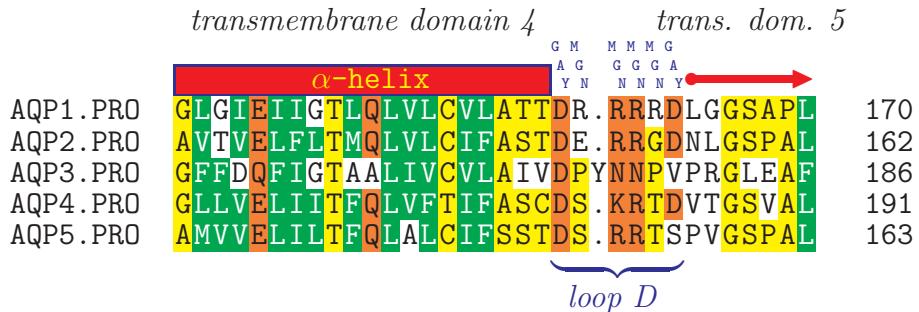
loop D

X external
X ambivalent
X internal

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[structure]{functional}
   \setends{1}{138..170}
   \feature{top}{1}{138..157}{box[Blue,Red][0.5pt]: %
       $\alpha$-helix[Yellow]}{transmembrane domain 4}
   \feature{top}{1}{158..163}{translate[Blue]}{}
   \backtranslabel{oblique}
   \feature{bottom}{1}{158..163}{brace[Blue]}{loop D [Blue]}
   \feature{top}{1}{164..170}{o->[Red]}{trans. dom. 5}
   \showlegend
\end{texshade}
```

chemical: residues are shaded due to chemical properties of their functional
groups

```
AQP1.PRO   GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL   170
AQP2.PRO   AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL   162
AQP3.PRO   GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF   186
AQP4.PRO   GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL   191
AQP5.PRO   AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL   163
```

X	acidic (−)
X	aliphatic
X	aliphatic (small)
X	amide
X	aromatic
X	basic (+)
X	hydroxyl
X	imino
X	sulfur

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[chemical]{functional}
   \setends{1}{138..170}
   \showlegend
\end{texshade}
```

With \shadeallresidues the threshold is ignored and all residues are shaded due to their group assignment. This is *not* identical to a threshold of 0% where only the majority group would be shaded. See the difference:

```
AQP1.PRO  GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL  170
AQP2.PRO  AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL  162
AQP3.PRO  GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF  186
AQP4.PRO  GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL  191
AQP5.PRO  AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL  163
```

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[chemical]{functional}
   \setends{1}{138..170}
   \shadeallresidues
\end{texshade}
```

rasmol: similar to [chemical] but with shading following the rasmol color scheme

```
                                                        site A      site B
                  .138
                  .139
                  .140
                  .141
                  .142
                  .143
                  .144
                  .145
                  .146
                  .147
                  .148
                  .149
                  .151
                  .152
                  .153
                  .154
                  .156
                  .157
                  .158
                  .159
                  .160
                  .161
                  .162
                  .163
                  .164
                  .165
                  .166
                  .167
                  .168
                  .169
                  .170
AQP1.PRO   GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL   170
AQP2.PRO   AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL   162
AQP3.PRO   GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF   186
AQP4.PRO   GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL   191
AQP5.PRO   AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL   163
```

```
X    Asp, Glu
X    Arg, Lys, His
X    Phe, Tyr, Trp
X    Ala, Gly
X    Cys, Met
X    Ser, Thr
X    Asn, Gln
X    Leu, Val, Ile
X    Pro
```

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[rasmol]{functional}
   \setends{1}{138..170}
   \showruler{bottom}{1}
   \rulersteps{1}
   \namerulerpos{150}{site A[Red]}
   \namerulerpos{155}{site B[Green]}
   \shadeallresidues
   \showlegend
\end{texshade}
```

standard area: this shading displays the differences in the surface area of the
different amino acid's sidechains

```
AQP1.PRO   GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL   170
AQP2.PRO   AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL   162
AQP3.PRO   GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF   186
AQP4.PRO   GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL   191
AQP5.PRO   AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL   163
```

```
X   88.1 (G); Standard sidechain area (Å²)
X  118.2 (A); 129.8 (S)
X  146.1 (C); 146.8 (P)
X  152.5 (T); 158.7 (D); 164.5 (V); 165.5 (N)
X  181.0 (I); 186.2 (E)
X  193.1 (L); 193.2 (Q); 202.5 (H); 203.3 (M)
X  222.8 (F); 225.8 (K)
X  238.8 (Y)
X  256.0 (R); 266.2 (W)
```

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[standard area]{functional}
   \setends{1}{138..170}
   \showlegend
   \shadeallresidues
\end{texshade}
```

accessible area: here, the surface area which can be accessed by solvent molecules is used as a basis for shading; low accessibility means hydrophobic (i. e. strongly buried residues), whereas highly accessible sidechains are hydrophilic (compare to **hydropathy** and **structure**)

```
                      membr.            loop   membr.
              ooooooooooooooooooooo  ▬▬▬▬  oooooo
AQP1.PRO  GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL  170
AQP2.PRO  AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL  162
AQP3.PRO  GFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF  186
AQP4.PRO  GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL  191
AQP5.PRO  AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL  163
```

23
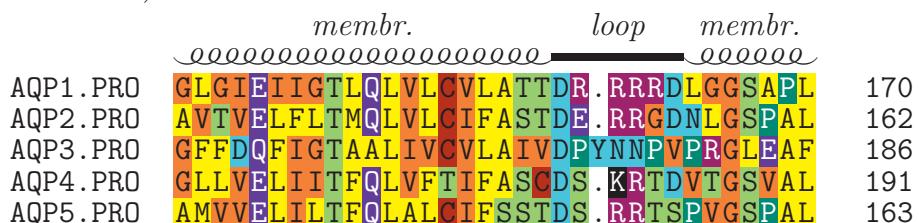
X	13.9 (C); Accessible sidechain area (Å^2)
X	23.0 (I); 23.5 (V); 25.2 (G)
X	28.7 (F); 29.0 (L); 30.5 (M); 31.5 (A)
X	41.7 (W); 44.2 (S); 46.0 (T); 46.7 (H)
X	53.7 (P)
X	59.1 (Y); 60.9 (D); 62.2 (N)
X	72.3 (E); 74.0 (Q)
X	93.8 (R)
X	110.3 (K)

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[accessible area]{functional}
   \setends{1}{138..170}
   \showlegend
   \feature{top}{1}{138..157,164..170}{helix}{membr.}
   \feature{top}{1}{158..163}{---}{loop}
   \featurerule{1mm}
   \shadeallresidues
\end{texshade}
```

1.6 Bar graphs and color scales

Amino acid properties, such as hydrophobicity, molecular weight, or charge can be shown as bar graphs or color scales along the alignment. Further, the degree of protein sequence conservation can be indicated. As an example, in the following aquaporin alignment plots of residue conservation (bars, top), are shown as well as properties of the AQP1 sequence: charge (scale, top), molecular weight are shown (scale, bottom), and hydrophobicity (bars, bottom).
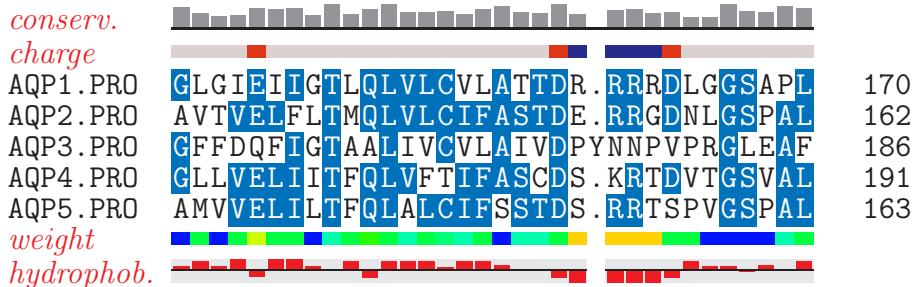
Code:

```
\begin{texshade}{AQPpro.MSF}
   \setends{1}{138..170}
   \feature{ttop}{1}{138..170}{bar:conservation}{}
   \showfeaturestylename{ttop}{conserv.}
   \ttopspace{-\baselineskip}
   \feature{top}{1}{138..170}{color:charge}{}
   \showfeaturestylename{top}{charge}
   \feature{bottom}{1}{138..170}{color:molweight[ColdHot]}{}
   \showfeaturestylename{bottom}{weight}
   \bbottomspace{-\baselineskip}
   \feature{bbottom}{1}{138..170}{bar:hydrophobicity[Red,Gray10]}{}
   \showfeaturestylename{bbottom}{hydrophob.}
   \featurestylenamescolor{Red}
   \featurestylenamesrm
   \featurestylenamesit
   \hideconsensus
\end{texshade}
```

1.7 Secondary structures

Predicted protein secondary structures in the DSSP, STRIDE PHD or HMMTOP file format can be included and displayed in the alignment. As an example, the following few commands show an aquaporin alignment with the PHD topology data for aquaporin type 1 (top sequence).

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[allmatchspecial]{similar}
   \includePHDtopo{1}{AQP1.phd}
\end{texshade}
```

Abbr.: *int.* – internal; *ext.* – external; *TM* – transmembrane domain

```
AQP1.PRO    MAS.........................EIKKKLFW    11
AQP2.PRO    MW...........................ELRSIAFS    10
AQP3.PRO    M.........NRCG.....EMLHIRYR......LL     15
AQP4.PRO    MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFW    35
AQP5.PRO    MK..........................KEVCSLAFF   11
consensus   !                           *** * ***
```

int. A

ext. B

```
                     TM1
AQP1.PRO    RAVVAEFLAMTLFVFISIGSALGFNYPLERNQTLV    46
AQP2.PRO    RAVLAEFLATLLFVFFGLGSALQWA...SS....P    38
AQP3.PRO    RQALAECLGTLILVMFGCGSVAQVVLSRGTH....    46
AQP4.PRO    KAVTAEFLAMLIFVLLSVGSTINWG...GSENPLP    67
AQP5.PRO    KAVFAEFLATLIFVFFGLGSALKWP...SA....L    39
consensus   ****!!*!*****!****!!****         *
```

int. A

ext. B

```
                     TM2
AQP1.PRO    QDNVKVSLAFGLSIATLAQSVGHISGAHSNPAVTL     81
AQP2.PRO    PSVLQIAVFGLGIGILVQALGHVSGAHINPAVTV     73
AQP3.PRO    GGFLTINLAFGFAVTLAILVAGQVSGAHLNPAVTF     81
AQP4.PRO    VDMVLISLCFGLSIATMVQCFGHISGGHINPAVTV    102
AQP5.PRO    PTILQISIAFGLAIGTLAQALGPVSGGHINPAITL    74
consensus   ** ****!!********* *!**!!*!*!!!*!*
```

int. C

26

```
                                                              ext. D
                                   TM3
AQP1.PRO   GLLLSCQISILRAVMYIIAQCVGAIVASAILSGI.                        115
AQP2.PRO   ACLVGCHVSFLRAAFYVAAQLLGAVAGAAILHEI.                        107
AQP3.PRO   AMCFLAREPWIKLPIYTLAQTLGAFLGAGIVFGLY                        116
AQP4.PRO   AMVCTRKISIAKSVFYITAQCLGAIIGAGILYLV.                        136
AQP5.PRO   ALLIGNQISLLRAVFYVAAQLVGAIAGAGILYWL.                        108
consensus  ****   ********!*  !!  *!!*****!**  *

                                        ext. D

                                                                TM4
AQP1.PRO   .........TSSLLENSLGRNDLARGVNSGQ....                        137
AQP2.PRO   .........TPVEIRGDLAVNALHNNATAGQ....                        129
AQP3.PRO   YDAIWAFAGNELVVSGPNGTAGIFATYPSGHLDMV                        151
AQP4.PRO   .........TPPSVVGGLGVTTVHGNLTAGH....                        158
AQP5.PRO   .........APLNARGNLAVNALNNNTTPGK....                        130
consensus           ***  *  * ******   *  * !

                           TM4
AQP1.PRO   .GLGIEIIGTLQLVLCVLATTDR.RRRDLGGSAPL                        170
AQP2.PRO   .AVTVELFLTMQLVLCIFASTDE.RRGDNLGSPAL                        162
AQP3.PRO   NGFFDQFIGTAALIVCVLAIVDPYNNPVPRGLEAF                        186
AQP4.PRO   .GLLVELIITFQLVFTIFASCDS.KRTDVTGSVAL                        191
AQP5.PRO   .AMVVELILTFQLALCIFSSTDS.RRTSPVGSPAL                        163
consensus   **  *****!  *!*********!   **  *  !*  **

                                                           int. E

                                                     ext. F
                           TM5
AQP1.PRO   AIGLSVALGHLLAIDYTGCGINPARSFGSAVLTR.                        204
AQP2.PRO   SIGFSVTLGHLLGIYFTGCSMNPARSLAPAVVTG.                        196
AQP3.PRO   TVGLVVLVIGTSMGFNSGYAVNPARDFGPRLFTAL                        221
AQP4.PRO   AIGFSVAIGHLFAINYTGASMNPARSFGPAVIMG.                        225
AQP5.PRO   SIGLSVTLGHLVGIYFTGCSMNPARSFGPAVVMN.                        197
consensus  **!**!  **********!***!!!!*********

                ext. F
                                             TM6
AQP1.PRO   .NFS.N......HWIFWVGPFIGSALAVL..IYDF                        229
AQP2.PRO   .KFD.D......HWVFWIGPLVGAIIGSL..LYNY                        221
AQP3.PRO   AGWGSEVFTTGQNW..WVPIVSPLLGSIGGVFVY                        254
AQP4.PRO   .NWE.N......HWIYWVGPIIGAVLAGA..LYEY                        250
AQP5.PRO   .RFSPS......HWVFWVGPIVGAMLAAI..LYFY                        223
consensus     *        *!**!**!******* *   ** *
```

```
AQP1.PRO   ILAPRSSDFTDRMK.............VWTS.....       247
AQP2.PRO   LLFPSAKSLQERL..AVLKG.LEPDTDWEEREVRR        253
AQP3.PRO   QL...............................          256
AQP4.PRO   V.FCPDVELKRRLKEAFSKAAQQTKGSYMEVEDNR        284
AQP5.PRO   LLFPSSLSLHDRV..AVVKGTYEPEEDWEDHREER        256
consensus  ****     * *** *  **      * *     *
```

int. G

```
AQP1.PRO   GQVEEYDLDAD.......DINSRVEMKPK......        269
AQP2.PRO   RQ..SVELHSPQSLPRG.................        268
AQP3.PRO   ..MIGCHLEQPPPSTEAENV.KLAHMKHKE.....       283
AQP4.PRO   SQVETEDLILKPGVVHVIDIDRGDEKKGKDSSGEV       319
AQP5.PRO   KK..TIELTAH.......................        265
consensus  ** * *!              *        * *
```

int. G

```
AQP1.PRO   ....   269
AQP2.PRO   .SKA   271
AQP3.PRO   ..QI   285
AQP4.PRO   LSSV   323
AQP5.PRO   ....   265
consensus
```

1.8   Sequence fingerprints

To gain a quick overview of sequence similarities or properties the
\fingerprint command has been implemented. It can depict the
complete sequence in one single line. The residues are presented as
colored vertical lines. The implementation of this kind of output was
inspired by a publication by KAI-UWE FRÖHLICH [6].
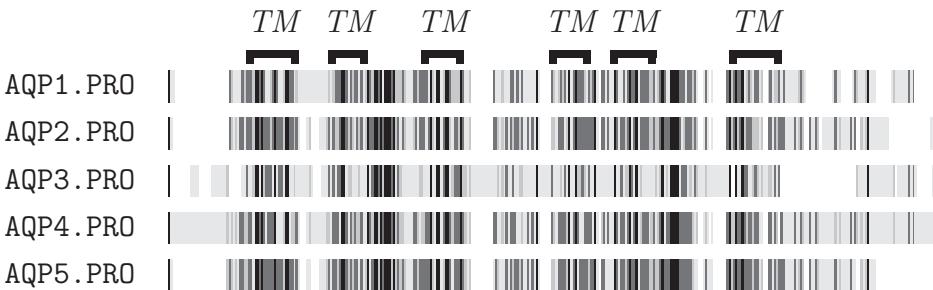
| non conserved
| similar
| ≥ 50% conserved
| all match

Code:

```
\begin{texshade}{AQPpro.MSF}
    \shadingmode[allmatchspecial]{similar}
    \shadingcolors{grays}
    \fingerprint{360}
    \showlegend
    \feature{top}{1}{13..36,51..68,94..112,138..156,%
                    165..185,211..232}{,-,}{TM}
\end{texshade}
```

The higher the similarity the darker the vertical lines. In this overview it becomes obvious that the transmembrane regions of the aquaporin isoforms are most conserved.

A fingerprint of charge distribution on different aquaporins is shown. below. Sequence gaps can be left blank (example above) or drawn as lines between the sequence blocks.
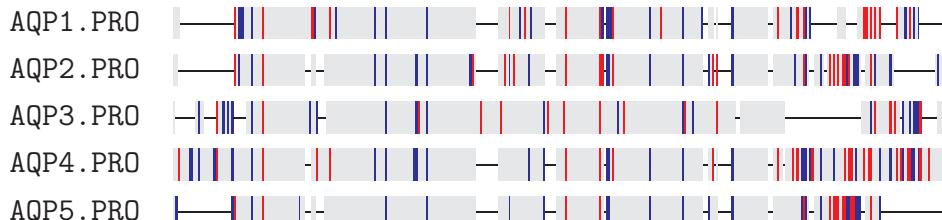
| acidic (−)
| basic (+)

Code:

```
\begin{texshade}{AQPpro.MSF}
   \shadingmode[charge]{functional}
   \shadeallresidues
   \fingerprint{360}
   \gapchar{rule}
   \showlegend
\end{texshade}
```

1.9   Sequence logos

Sequence logos represent the information content of the aligned sequences at a position in bit (max. 2 bit for DNA, i. e. $\log_2 4$, and 4.322 bit for proteins, i. e. $\log_2 20$) and the relative frequency of a base or amino acid at this position [7]. Thus, more information is contained in logos than in a standard consensus sequence. The example below shows a DNA sequence alignment with the logo on the top.

It must be remarked that a logo from only five sequences does not produce meaningful results - it rather illustrates the technique.

```
logo

AQP1nuc.SEQ   CCTGGGCATTGAGATCATTGGCACCCTGCA   443
AQP2nuc.SEQ   TGTGACTGTAGAGCTCTTCCTGACCATGCA   419
AQP3nuc.SEQ   .CCAATGGCACAGCTGGTATC..TTTGCCA   424
AQP4nuc.SEQ   GCTCCTGGTGGAGCTAATAATCACTTTCCA   506
AQP5nuc.SEQ   CATGGTGGTGGAGTTAATCTTGACTTTCCA   422
consensus      *** **** *!!*! *!   * *****!!
```

Code:

```
\begin{texshade}{AQPDNA.MSF}
   \setends{1}{414..443}
   \showsequencelogo{top}
\end{texshade}
```

Next, only the logo of a protein alignment is displayed plus the degree of sequence conservation as a color scale in the consensus line. Note, that the full functionality of the feature lines remains.

30
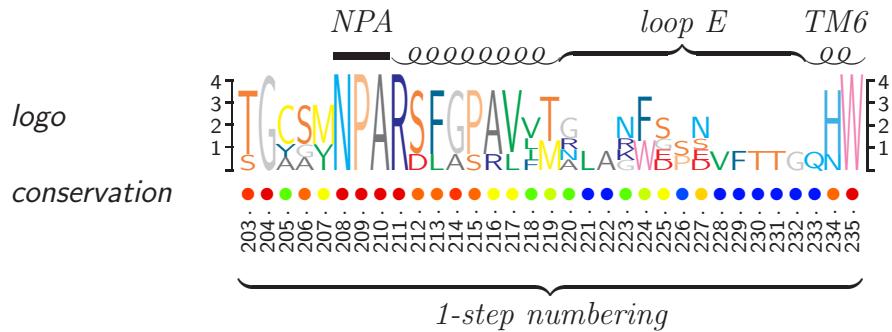
Code:

```
\begin{texshade}{AQPpro.MSF}
    \setends{AQP3.PRO}{203..235}
    \showsequencelogo{top} \showlogoscale{leftright}
    \hideseqs
    \residuesperline*{33}
    \defconsensus{{$\bullet$}}{{$\bullet$}}{{$\bullet$}}
    \showconsensus[ColdHot]{bottom}
    \nameconsensus{conservation} \namessf\namessl
    \showruler{bottom}{AQP3.PRO} \rulersteps{1}
    \feature{top}{AQP3.PRO}{208..210}{---}{NPA}
    \feature{top}{AQP3.PRO}{211..219}{helix}{}
    \feature{top}{AQP3.PRO}{220..232}{brace}{loop E}
    \feature{top}{AQP3.PRO}{233..235}{helix}{TM6}
    \feature{bottom}{AQP3.PRO}{203..235}{brace}{1-step numbering}
\end{texshade}
```

The same logo is shown below but with frequence correction turned on (\dofrequencycorrection), see p.. This takes into account the difference between the amino acid distribution in the alignment and the equal distribution of 5% for each residue.
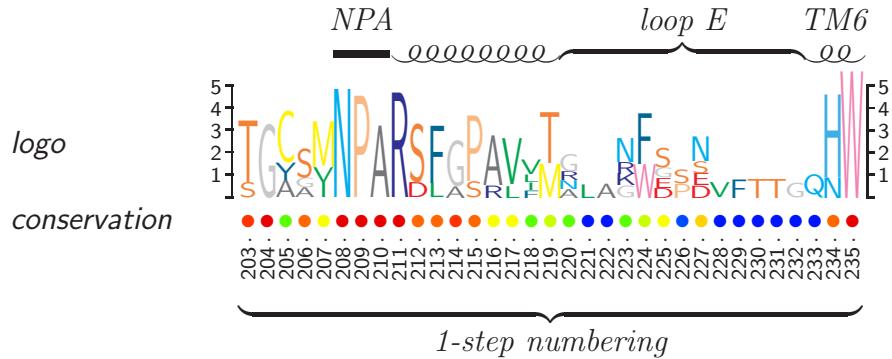
1.10 Subfamily logos

The following output is derived from the calculation of a subfamily logo [14]. Such logos display relevant deviations of a subfamily compared to the remaining set of sequences. Here, typical residues of AQP3 are shown (upright) which deviate from the remaining four aquaporins of this alignment (upside-down). The output can be directly compared to the sequence logo above, which displays the same section of the alignment. Note, that five sequences are far too few to obtain meaningful results with this method. This is just to illustrate the approach.
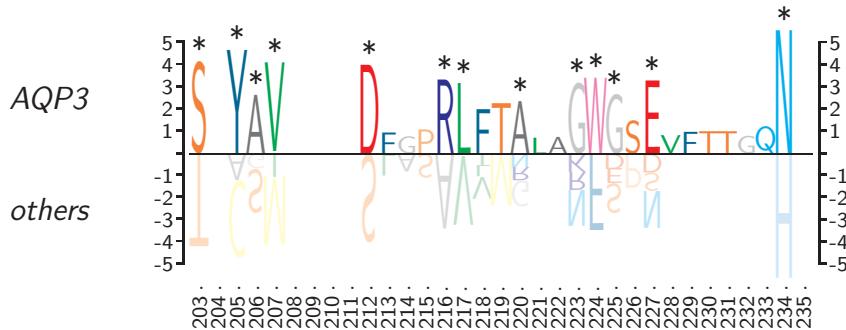
Code:

```
\begin{texshade}{AQPpro.MSF}
   \setends{AQP3.PRO}{203..235}
   \residuesperline*{33}
   \setsubfamily{3}
   \showsubfamilylogo{top} \showlogoscale{leftright}
   \namesubfamilylogo[others]{AQP3}
   \namessf \namessl
   \showruler{bottom}{AQP3.PRO} \rulersteps{1}
   \hideseqs
   \hideconsensus
   \dofrequencycorrection
\end{texshade}
```

1.11   Customization of the alignment output

Extensive possibilities are given to the user to customize the final output of an alignment. Thus, all parameters defining the appearance of letters can be changed individually for sequence residues, names and numbering or the describing feature texts. Additional manual shading can be applied to any region or block of residues. Sequences are easily re-ordered, separated, hidden or blanked out without recalculation of the entire alignment; sections of the alignment can also be shown. Numbering and rulers can be displayed and set to any value. A powerful tool is the \feature command which allows one to label stretches of residues with bars, arrows, braces or any fill character and describing text. Legends are set automatically if desired, but user commands are also provided to build individual legends.

2   Format of alignment input files

TEXshade can handle two common alignment input formats, i. e.
the MSF format (multiple sequence format) and the ALN format
(alignment format). The MSF format is used by `PILEUP` of the Unix
GCG sequence analysis package[19]. Files in the ALN format are pro-
duced by `CLUSTAL` which is available for free for Unix, DOS and Mac-
intosh. Further, upon request, the FASTA format is supported since
version 1.6. In addition to the mentioned software many alignment
programs have export filters for the MSF, ALN or FASTA format,
e. g. `MACAW` produces ALN files. If you are not sure whether your fa-
vorite sequence aligner produces one of the required formats compare
its output to the following examples. TEXshade determines the for-
mat from the internal file structure, thus extensions like MSF, ALN
or FASTA are not required. If you can choose the alignment format
MSF is recommended, because this format gives information about
the sequence type, i. e. peptide or nucleotide sequences, and length
(for the correct setting of gaps at the sequence end).

2.1   The MSF file format

Files of this type are divided into a header section and the multiple
sequence alignment. The header may contain the following compo-
nents:

File Type: (optional) The first header line reads for nucleic acids alignments
`!!NA_MULTIPLE_ALIGNMENT 1.0` and for amino acid sequences
`!!AA_MULTIPLE_ALIGNMENT 1.0` (all uppercase).

Description: (optional) Informative text describing what is in the file.

Dividing line: (required!) Must include the following attributes:

    `MSF`: Displays the number of bases or residues in the multiple
sequence alignment.

    `Type`: Displays the sequence type, 'P' for a peptide and 'N' for a
nucleotide alignment.

    `Checksum`: Displays an integer value that characterizes the contents of
the file.

---

[19]For a description see `http://gene.md.huji.ac.il/Computer/GCG9doc`

.. The two periods act as a divider between the descriptive information and the following sequence information.

Name/Weight: (required!) Must include the name of each sequence included in the alignment, as well as its length, checksum and weight.

Two slashes (//): (required!) This separating line divides the name/weight information from the sequence alignment

The alignment section consists of sequence blocks divided by an empty line. Each sequence line starts out with the sequence name. An example file is shown here:

```
AQP.MSF  MSF: 87  Type: P  May 1st, 1998 Check: 2586 ..
Name: AQP1.PRO  Len: 66  Check:  1367    Weight: 1.00
Name: AQP2.PRO  Len: 58  Check:  2176    Weight: 1.00
Name: AQP3.PRO  Len: 83  Check:  1893    Weight: 1.00
Name: AQP4.PRO  Len: 63  Check:  3737    Weight: 1.00
Name: AQP5.PRO  Len: 59  Check:  3413    Weight: 1.00
//
         1                                           45
AQP1.PRO  MAS.......................EIKKKLFWRAVVAEFLAM
AQP2.PRO  MW........................ELRSIAFSRAVLAEFLAT
AQP3.PRO  M.........NRCG.....EMLHIRYR......LLRQALAECLGT
AQP4.PRO  MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFWKAVTAEFLAM
AQP5.PRO  MK........................KEVCSLAFFKAVFAEFLAT


         45                                          87
AQP1.PRO  TLFVFISIGSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL
AQP2.PRO  LLFVFFGLGSALQWA...SS....PPSVLQIAVAFGLGIGIL
AQP3.PRO  LILVMFGCGSVAQVVLSRGTHGGF....LTINLAFGFAVTLA
AQP4.PRO  LIFVLLSVGSTINWG...GSENPLPVDMVLISLCFGLSIATM
AQP5.PRO  LIFVFFGLGSALKWP...SA....LPTILQISIAFGLAIGTL
```

TeXshade extracts only the information from the file it really needs. So, do not mind all the checksums listed in the file—TeXshade does not either. The same is true for `Weight`. Required are the string `MSF:` for the identification of the file format and `Type:` for the determination of the sequence type (both in the dividing line), further all `Name:` definitions and finally `//`. The MSF format allows one to comment out sequences. This is done by putting an exclamation point directly infront of the respective `Name`. These sequences are neither

displayed nor used for the calculation of the consensus. This works for TEXshade, too. To comment out sequences without changing the input file use the TEXshade command `\killseq{`⟨*seqref*⟩`}` (4.5.2).

```
   AQP.MSF  MSF: 87  Type: P  May 1st, 1998 Check: 2586 ..
   Name: AQP1.PRO  Len: 66  Check:  1367    Weight: 1.00
   !Name: AQP2.PRO  Len: 58  Check:  2176    Weight: 1.00
   !Name: AQP3.PRO  Len: 83  Check:  1893    Weight: 1.00
   Name: AQP4.PRO  Len: 63  Check:  3737    Weight: 1.00
   Name: AQP5.PRO  Len: 59  Check:  3413    Weight: 1.00
   //
           1                                            45
   AQP1.PRO  MAS.......................EIKKKLFWRAVVAEFLAM
   AQP2.PRO  MW........................ELRSIAFSRAVLAEFLAT
   AQP3.PRO  M.........NRCG.....EMLHIRYR......LLRQALAECLGT
   AQP4.PRO  MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFWKAVTAEFLAM
   AQP5.PRO  MK........................KEVCSLAFFKAVFAEFLAT


           45                                           87
   AQP1.PRO  TLFVFISIGSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL
   AQP2.PRO  LLFVFFGLGSALQWA...SS....PPSVLQIAVAFGLGIGIL
   AQP3.PRO  LILVMFGCGSVAQVVLSRGTHGGF....LTINLAFGFAVTLA
   AQP4.PRO  LIFVLLSVGSTINWG...GSENPLPVDMVLISLCFGLSIATM
   AQP5.PRO  LIFVFFGLGSALKWP...SA....LPTILQISIAFGLAIGTL
```

The sequence lengths given after `Len:` are not used by TEXshade. Due to the fact that most alignment programms calculate the sequence length by summing up residues and additionally gaps which is not really correct. In order to have the sequence break right after the last residue without printing further gap symbols TEXshade counts the number of residues by itself. You can also use the command `\seqlength` in the TEXshade environment to set the values manually if you do not trust a machine.

2.2 The ALN file format

ALN files are quite similar to the above described MSF files. They simply lack a defined header section. Nevertheless, describing text is allowed before the alignment part. TEXshade determines the number of sequences and their names from the last sequence block—so, no further text lines are allowed after this block! Due to a lacking

declaration in the file the sequence type has to be set in the `texshade` environment by `\seqtype{⟨type⟩}` with 'P' for peptide and 'N' for nucleotide sequences; for the example below: `\seqtype{P}`. If no `\seqtype` command is used TEXshade assumes a nucleotide sequence.

```
                profalign  May 1st, 1998, 16:58


   of AQPpro.MSF{}


   Muliple alignment parameter:


   Gap Penalty (fixed):          10.00
   Gap Penalty (varying):        .05
   Gap separation penalty range: 8
   Percent. identity for delay:  0%
   List of hydrophilic residue:  GPSNDQEKRH
   Protein Weight Matrix:        blosom


                    10        20        30        40
                     .         .         .         .
   AQP1.PRO  MAS.......................EIKKKLFWRAVVAEFLAM
   AQP2.PRO  MW.......................ELRSIAFSRAVLAEFLAT
   AQP3.PRO  M.........NRCG.....EMLHIRYR......LLRQALAECLGT
   AQP4.PRO  MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFWKAVTAEFLAM
   AQP5.PRO  MK.......................KEVCSLAFFKAVFAEFLAT
             *                                  .  ** *.


   AQP1.PRO  TLFVFISIGSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL
   AQP2.PRO  LLFVFFGLGSALQWA...SS....PPSVLQIAVAFGLGIGIL
   AQP3.PRO  LILVMFGCGSVAQVVLSRGTHGGF....LTINLAFGFAVTLA
   AQP4.PRO  LIFVLLSVGSTINWG...GSENPLPVDMVLISLCFGLSIATM
   AQP5.PRO  LIFVFFGLGSALKWP...SA....LPTILQISIAFGLAIGTL
             .. *   .**                     .  **  .
```

The minimal contents of an ALN file are shown below; this is fully sufficient. Many sequence alignment programs can produce such an output. Have a look at `seqpup` by DON GILBERT if you need a comprehensive conversion program[20].

---

[20]Sorry, `seqpup` is much more!

```
AQP1.PRO   MAS.........................EIKKKLFWRAVVAEFLAM
AQP2.PRO   MW..........................ELRSIAFSRAVLAEFLAT
AQP3.PRO   M.........NRCG.....EMLHIRYR......LLRQALAECLGT
AQP4.PRO   MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFWKAVTAEFLAM
AQP5.PRO   MK..........................KEVCSLAFFKAVFAEFLAT


AQP1.PRO   TLFVFISIGSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL
AQP2.PRO   LLFVFFGLGSALQWA...SS....PPSVLQIAVAFGLGIGIL
AQP3.PRO   LILVMFGCGSVAQVVLSRGTHGGF....LTINLAFGFAVTLA
AQP4.PRO   LIFVLLSVGSTINWG...GSENPLPVDMVLISLCFGLSIATM
AQP5.PRO   LIFVFFGLGSALKWP...SA....LPTILQISIAFGLAIGTL
```

2.3   The FASTA file format

In FASTA files each sequence is led by a single description line starting with a '>'. TEXshade uses the first word delimited by the leading '>' and a space as the sequence name. If no descriptive text is present TEXshade generates a sequence name consisting of 'seq' plus a consecutive number. The lines following the description line contain the sequence.

```
>AQP1.PRO
MAS.........................EIKKKLFWRAVVAEFLAM
TLFVFISIGSALGFNYPLERNQTLVQDNVKVSLAFGLSIATL

>AQP2.PRO
MW..........................ELRSIAFSRAVLAEFLAT
LLFVFFGLGSALQWA...SS....PPSVLQIAVAFGLGIGIL

>AQP3.PRO
M.........NRCG.....EMLHIRYR......LLRQALAECLGT
LILVMFGCGSVAQVVLSRGTHGGF....LTINLAFGFAVTLA

>AQP4.PRO
MSDGAAARRWGKCGPPCSRESIMVAFKGVWTQAFWKAVTAEFLAM
LIFVLLSVGSTINWG...GSENPLPVDMVLISLCFGLSIATM

>AQP5.PRO
MK..........................KEVCSLAFFKAVFAEFLAT
LIFVFFGLGSALKWP...SA....LPTILQISIAFGLAIGTL
```

3   Use of a TEXshade parameter file

Using predefined parameter files for repeatedly occuring situations can save a lot of typing and makes the output throughout the publication or presentation more consistent. Further, such files are an easy way to exchange self-defined shading modes or new color schemes (i. e. for a satisfying grayscale output) with other users. If you have created a parameter file, which you think is of interest for others, please submit it to me[21] as an e-mail attachment together with a short description. I will take care of those files and post them—with a reference to the author—together with the next TEXshade distribution to make them available for all interested users.

No special file format is required for parameter files. TEXshade simply calls the file using the \input command right after resetting all parameters to default. An example parameter file is present containing the standard parameters of TEXshade called texshade.def. This file can be changed freely and can be used as a template for the creation of personal parameter files.

Five steps are executed by TEXshade when processing the texshade environment:

\begin{texshade}[⟨*parameterfile*⟩]{⟨*alignmentfile*⟩}

1. Analysis of the ⟨*alignmentfile*⟩; determination of the number of sequences and sequence names

2. Setting parameters to default

3. Setting parameters to the definitions of the ⟨*parameterfile*⟩, if existent

4. Execution of further TEXshade commands within the evironment, if existent

\end{texshade}

5. Loading and setting the alignment on a line by line basis

---

[21]ebeitz@pharmazie.uni-kiel.de

4 `texshade` user commands

The TEXshade package must be loaded by the `\usepackage` command in the document header section.

   `\usepackage[`⟨*option*⟩`]{texshade}`

Then, the `texshade` environment is ready to use as described in 1.4. See also section 3 for a description of the optional parameter file. All other commands provided by TEXshade (except `\molweight`, `\charge` [4.10] and `\shadebox` [4.7]) must be used within the `texshade` environment.

4.1 Using predefined shading modes

If no `\shadingmode` command is given in the `texshade` environment the default shading mode (*identical*, see 1.5.1) is active. For the selection of one of the other predefined shading modes the following command is provided.

   `\shadingmode[`⟨*option*⟩`]{`⟨*mode*⟩`}`

You can choose from four shading modes and declare one option which depends on the selected mode.

1. `\shadingmode[`⟨*allmatchspecial/number*⟩`]{identical}`

   There is not much to explain here (see 1.5.1). Use the option `allmatchspecial` to shade positions with a special color where all residues are identical. Or use a percentage number (0–100) as an option to set an additional threshold for highly conserved residues, e. g. `\shadingmode[90]{identical}`. `\allmatchspecial` can also be used as a command with or without an optional parameter for setting the high conservation threshold. As both, option or command `allmatchspecial` is only active in the *identical* and *similar* shading modes.

   One can choose from five predefined shading color schemes with the command `\shadingcolors{`⟨*scheme*⟩`}`. The sets are named 'blues' (used in the example, 1.5.1), 'reds', 'greens', 'grays' and 'black'. Default is `\shadingcolors{blues}`. Further, the colors for the non matching, the conserved and all matching (or highly conserved) residues can be set individually plus the letter case (lower or upper) or any character can be chosen:

\nomatchresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}

\conservedresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}

\allmatchresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}

For how to handle colors for the foreground ⟨*res.col.*⟩ and the background ⟨*shad.col.*⟩ see section 5. The third parameter ⟨*case*⟩ tells TEXshade to print the corresponding residue as a lowercase or an uppercase letter or even to print any other character. Finally, the ⟨*style*⟩ parameter tells TEXshade which shape to use for the letters. Use one of the following styles for ⟨*style*⟩.

⟨*style*⟩	*effect*
bf	bold face series
md	normal series
up	upright shape (normal shape)
it	italics shape
sl	slanted shape
rm	modern roman family
sf	sans serif family
tt	typewriter family

In order to change only some of the parameters it is sufficient to declare these and use empty braces for the others. Examples:

  \conservedresidues{White}{Blue}{upper}{bf}: the conserved residues are printed as bold face white uppercase letters on blue.

  \nomatchresidues{}{}{{$\bullet$}}{}: instead of the non matching residues a '•' is printed. The colors and style are not changed. Note the double curly braces which make TEXshade interpret this complex symbol description as one single character.

2. \shadingmode[⟨*allmatchspecial/number*⟩]{similar}

   See 1.5.2 for an example output and an explanation of the shading. In addition to the described commands for changing shading colors this shading mode provides the command \similarresidues. Use it in analogy to the commands above.

How does T_EXshade know which residues are considered to be similar? These definitions are set by two command couples, i.e. `\pepsims`,`\pepgroups` for peptides and `\DNAsims`,`\DNAgroups` for nucleotides. With `\pepsims` and `\DNAsims` residues are defined which are similar to the consensus residue. Examples:

`\pepsims{S}{TA}` If a serine is the consensus residue then all threonins and alanines at this position are shaded in the color for similars. This definition does *not* imply that threonine and alanine are similar to each other! This becomes obvious when you inspect the next definition:

`\pepsims{T}{S}` Serine but not alanine is declared to be similar to threonine.

What happens if there is no consensus residue? How does T_EXshade decide if a group of similars is greater than the threshold? For this groups are pre-defined:

`\pepgroups{FYW,ILVM,RK,DE,GA,ST,NQ}` This command allows one to set up to nine groups of similars, separated by commas. Each residue can belong to only one group. If one residue is assigned to several groups only the last assignment is carried out.

`\DNAgroups{GAR,CTY}` This command is used in analogy to the amino acid groups. Here, two ambiguity codes ('R' for purine base and 'Y' for pyrimidine base) are assigned in addition.

Residues which do not appear in any of the four commands are considered not to belong to a group. The default settings for similars are listed below:

```
\pepgroups{FYW,ILVM,RK,DE,GA,ST,NQ}

\pepsims{F}{YW}  % Y and W are similar to F
\pepsims{Y}{WF}  % W and F are similar to Y
\pepsims{W}{YF}  % Y and F are similar to W

\pepsims{I}{LVM} % L, V and M are similar to I
\pepsims{L}{VMI} % V, M and I are similar to L
\pepsims{V}{MIL} % M, I and L are similar to V

\pepsims{R}{KH}  % K and H are similar to R
```

```
    \pepsims{K}{HR}  % H and R are similar to K
    \pepsims{H}{RK}  % R and K are similar to H

    \pepsims{A}{GS}  % G and S are similar to A
    \pepsims{G}{A}   % A (but not S) is similar to G

    \pepsims{S}{TA}  % T and A are similar to S
    \pepsims{T}{S}   % S (but not A) is similar to T

    \pepsims{D}{EN}  % E and N (but not Q) are similar to D
    \pepsims{E}{DQ}  % D and Q (but not N) are similar to E
    \pepsims{N}{QD}  % Q and D (but not E) are similar to N
    \pepsims{Q}{NE}  % N and E (but not D) are similar to Q

    \DNAgroups{GAR,CTY}

    \DNAsims{A}{GR}  % G and R are similar to A
    \DNAsims{G}{AR}  % A and R are similar to G
    \DNAsims{R}{AG}  % A and G are similar to R

    \DNAsims{C}{TY}  % T and Y are similar to C
    \DNAsims{T}{CY}  % C and Y are similar to T
    \DNAsims{Y}{CT}  % C and T are similar to Y
```

3. \shadingmode[⟨*filename*⟩]{T-Coffee}

   Enter a ⟨*filename*⟩ to load the shading information from a
   T-Coffee score_ascii file (www.tcoffee.org); see example
   in 1.5.3. Make sure that the alignment file specified in the
   \texshade command and this shading file correspond to each
   other.

   If you do not enter a ⟨*filename*⟩ here, a separate command
   \includeTCoffee{⟨*filename*⟩} must be used.

   T-Coffee shading can also be used in the consensus p. 47 and in
   the feature lines, in particular color scales and bar plots p. 64,
   for the display of shading information.

4. \shadingmode[⟨*seqref*⟩]{diverse}

   1.5.4 depicts an example alignment. Choose the number or the
   name of the sequence ⟨*seqref*⟩ which will be treated as the con-
   sensus and to which the other sequences are compared. If no

⟨*seqref*⟩ is declared the first sequence is set as consensus (⟨*seqref*⟩ = 1).

Standard definitions for `diverse` mode are:

```
\nomatchresidues{Black}{White}{lower}{up}
\similarresidues{Black}{White}{lower}{up}
\conservedresidues{Black}{White}{{.}}{up}
\allmatchresidues{Black}{White}{{.}}{up}
\gapchar{-}
```

After calling `\shadingmode{diverse}` these commands can be used to redefine the `diverse` mode settings (mind the double curly braces around the dot-symbol!).

5. `\shadingmode[`⟨*type*⟩`]{functional}` There are seven different functional shading modes available for peptide sequences; nucleotide sequences can not be shaded due to functional aspects. Five of TeXshade's functional modes correspond to the four 'alphabets' employed by KARLIN and GHANDOUR for peptide alignments [2] or by the rasmol software. Additional 'alphabets' to the standard 20-letter array of amino acids can highlight peptide similarities which were otherwise not visible. For the 'alphabet' definitions see below:

   - ⟨*type*⟩ = `charge`   Acidic (D, E) and basic (H, K, R).
   - ⟨*type*⟩ = `hydropathy`   Acidic and basic (as above), polar uncharged (C, G, N, Q, S, T, Y) and hydrophobic nonpolar (A, F, I, L, M, P, V, W), see also KYTE and DOOLITTLE [3].
   - ⟨*type*⟩ = `structure`   External (D, E, H, K, N, Q, R), internal (F, I, L, M, V) and ambivalent (A, C, G, P, S, T, W, Y).
   - ⟨*type*⟩ = `chemical`   Acidic (D, E), aliphatic (I, L, V), aliphatic (small) (A, G), amide (N, Q), aromatic (F, W, Y), basic (H, K, R), hydroxyl (S, T), imino (P) and sulfur (C, M).
   - ⟨*type*⟩ = `rasmol`   (D, E), (K, R, H), (F, Y, W), (A, G), (C, M), (S, T), (N, Q), (I, L, V), (P).

44

The two modes described below highlight sidechain sizes and hydrophobicity, respectively, according to Rose *et al.* [4,5]. Standard area stands for the surface area of the residue in $\mathring{A}^2$, i.e. it is a measure for the size of a residue's sidechain. The accessible area value (also in $\mathring{A}^2$) gives information about the size of the surface area which is accessible by solvent molecules within the folded protein. A very small area means that the residue is strongly buried and is thus very hydrophobic. Hydrophilic residues in turn possess large accessible areas due to their prefered location at the protein surface. Therefore, this kind of shading provides another method, in addition to `hydropathy` and `structure`, for the visualization of structural protein properties.

- $\langle type \rangle =$ `standard area`   for the area values see legend of the alignment in 1.5.5

- $\langle type \rangle =$ `accessible area`   for values see 1.5.5

If no $\langle type \rangle$ or an unknown $\langle type \rangle$ is designated as option all functional groups and shading colors are cleared. This is also achieved by the command `\clearfuncgroups`. With all groups cleared one can start to build new shading modes from scratch. How to do this is explained in the next section.

In order to exchange the colors but to keep the group definitions and descriptions the command `\funcshadingstyle` can be employed. Usage:

`\funcshadingstyle{`$\langle residue \rangle$`}{`$\langle res.col. \rangle$`}{`$\langle shad.col. \rangle$`}`

`{`$\langle case \rangle$`}{`$\langle style \rangle$`}`

$\langle residue \rangle$ is one representative of the whole amino acid group. The colors which are declared by the next four parameters are used for all residues in this group. $\langle case \rangle$ and $\langle style \rangle$ are as described for example in `\nomatchresidues`.

With `\shadeallresidues` the threshold is ignored and all residues are shaded due to their group assignment.

4.2 Creating new functional shading modes

The grouping of amino acids due to other properties can make sense as suggested by KARLIN and GHANDOUR [2], e.g. physical properties (molecular weight, shape), kinetic properties (reaction velocity, Michaelis-Menton constant), or structure (α-helices, β-sheets, turns). New amino acid groups are defined with the \funcgroup command. This command needs six parameters:

\funcgroup{⟨*descr*⟩}{⟨*residues*⟩}{⟨*res.col.*⟩}{⟨*shad.col.*⟩}
{⟨*case*⟩}{⟨*style*⟩}

⟨*descr*⟩ contains descriptive text which is displayed in the legend. The second parameter ⟨*residues*⟩ holds the amino acids to be grouped. The colors for the foreground and background are set with the following two parameters, the case and style is declared by the last parameters. The example below defines a funcional group named 'acidic (−)' containing the amino acids aspartic and glutamic acid with white letters on a red background:

\funcgroup{acidic ($-$)}{DE}{White}{Red}{upper}{up}

For the usage of colors see section 5. Up to nine individual groups can be defined. New groups are simply added to the already existing groups, i.e. if an extension of the group definitions of an existing shading mode is desired there is no need to clear these groups und re-define them again. Just add the new groups with the \funcgroup command. To create completely new modes use the command \shadingmode{functional} without an option *before* setting the new groups. The new definitions are active only in the functional shading mode—so be sure to have it switched on before setting the new groups. Remember, \shadingmode{functional} without an optional parameter clears all groups defined before, see above. The following example shows the definitions needed to produce an output which is identical to the functional mode 'charge':

\begin{texshade}{⟨*alignmentfile*⟩}

  \shadingmode{functional}
  \funcgroup{acidic ($-$)}{DE}{White}{Red}{upper}{up}
  \funcgroup{basic ($+$)}{HKR}{White}{Blue}{upper}{up}

\end{texshade}

4.3 Appearance of the consensus line

An important parameter for the calculation of the consensus is the threshold percentage. Default setting is 50%, i.e. to become the consensus residue more than half of the residues at this position must be identical or similar, depending on the shading mode. Any percentage between 0 and 100 is allowed and can be set with \threshold{⟨percentage⟩}, e.g. \threshold{50}.

Additionally, an optional parameter can be set, e.g. \threshold[90]{50}, to label residues that are highly conserved in a special color (see example on page 14).

Another possibility is to set one sequence of the alignment as consensus and compare the other sequences to this one. Therefore, the command \constosingleseq{⟨seqref⟩} is provided. The ⟨seqref⟩ selects the sequence to be used as consensus (numbering according to the appearance in the alignment file; top sequence is number 1, or use the sequence name). Nevertheless, the threshold percentage is also taken into account, i.e. with a threshold of 50% half of the sequences must be identical or similar compared to the specified consensus sequence in order to be shaded. With \constoallseqs the consensus is calculated considering all sequences (the case described in the paragraph above).

Consensus lines are displayed either on the top or at the bottom of the alignment by calling

\showconsensus[⟨color/scale⟩[,⟨color/scale⟩]]{⟨position⟩}

with ⟨scale⟩ Gray, BlueRed, RedBlue, GreenRed, RedGreen, ColdHot (recommended), HotCold, or T-Coffee 1.5.3, and ⟨position⟩ top or bottom.

The first color defines the foreground, i.e. the letters, the second color—if specified—defines the background. If a color scale is named the consensus will be shaded according to the level of sequence conservation. For an example see page 14. You can find more information on color scales on page 64. These scales can be exported as a Pymol [8] script by \exportconsensus[⟨filename⟩]{⟨seqref⟩}. If no ⟨filename⟩ is specified export.txt will be used. The generated file can be opened in Pymol in order to shade a 3D model of the sequence ⟨seqref⟩.

To hide the consensus use \hideconsensus. The consensus line is named 'consensus' in english texts, 'consenso' in spanish or 'Konsensus' if the german.sty is used. With \nameconsensus{⟨name⟩} any name can be set.

You can tell TeXshade which symbols or letters to use in the consensus line for different matching qualities by

$\mathtt{\backslash defconsensus\{}\langle symbol1 \rangle \mathtt{\}\{}\langle symbol2 \rangle \mathtt{\}\{}\langle symbol3 \rangle \mathtt{\}}.$

The following parameters are allowed for symobols 1–3:

1. $\langle symbol1 \rangle$ = no match symbol (if below threshold)

   - any character or letter
   - `{}` (empty braces) for blank space

2. $\langle symbol2 \rangle$ = conserved symbol (if threshold is exceeded)

   - `upper` (prints the consensus residue in uppercase)
   - `lower` (prints the consensus residue in lowercase)
   - any character or letter
   - `{}` (empty braces) for blank space

3. $\langle symbol3 \rangle$ = highly conserved symbol (if

   $\mathtt{\backslash allmatchspecial}$ is active)

   - see $\langle symbol2 \rangle$

Example: `\defconsensus{{}}{*}{upper}` does not show non matching residues in the consensus line, marks conserved residues with '`*`', and displays the uppercase letter of the consensus residue at positions with high conservation.

Finally, the colors of the above defined symbols are adjustable by the command:

$\mathtt{\backslash consensuscolors\{}\langle res.col.1 \rangle \mathtt{\}\{}\langle shad.col.1 \rangle \mathtt{\}}$
$\mathtt{\{}\langle res.col.2 \rangle \mathtt{\}\{}\langle shad.col.2 \rangle \mathtt{\}}$
$\mathtt{\{}\langle res.col.3 \rangle \mathtt{\}\{}\langle shad.col.3 \rangle \mathtt{\}}$

The color definitions are in the same order as in the `\defconsensus` command:

1. $\langle res.col.1 \rangle$ = no match residue color (if below threshold)

   $\langle shad.col.1 \rangle$ = no match background color

48

2. $\langle res.col.2\rangle$ = conserved residue color (if threshold is exceeded)

$\langle shad.col.2\rangle$ = conserved background color

3. $\langle res.col.3\rangle$ = highly conserved residue color (if

`\allmatchspecial` is active)

$\langle shad.col.3\rangle$ = highly conserved background color

For colors which are not to be changed empty braces can be used. Example:

`\consensuscolors{}{}{Blue}{White}{Red}{Green}`

Non matching symbol colors are not changed, conserved residues are displayed blue on white and highly conserved residues appear as red symbols on a green background in the consensus line.

4.4 Display of logos

4.4.1 Sequence logos

In a sequence logo [7], the information content $I(P_i)$ of each alignment position i is defined as

$$I(P_i) = \log_2 |\Sigma| + \sum P_{ij} \cdot \log_2 P_{ij}$$

with $|\Sigma|$ being the cardinality of the used alphabet, i.e. 4 for DNA and 20 for protein sequences, and P_{ij} being the frequency of residue j at this position. Each position is displayed as a stack of residue symbols whose heights represent their proportion of the information content (example on p.30).

The display of sequence logos can be either on the top or at the bottom of a nucleotide or protein alignment. Logos will be shown after the command: `\showsequencelogo[`$\langle colorset\rangle$`]{`$\langle top/bottom\rangle$`}`. If no optional $\langle colorset\rangle$ is selected the residues will be shaded as follows:

- Nucleotide sequences

  G : Black

  A : Green

  T,U : Red

  C : Blue

- Protein sequences (similar to rasmol)

D,E	:	Red
C,M	:	Yellow
K,R	:	Blue
S,T	:	Orange
F,Y	:	MidnightBlue
N,Q	:	Cyan
G	:	LightGray
L,V,I	:	Green
A	:	DarkGray
W	:	CarnationPink
H	:	CornflowerBlue
P	:	Apricot
B,Z	:	LightMagenta

Optional color sets correspond to the functional shading modes `chemical`, `rasmol`, `hydropathy`, `structure`, `standard area`, `accessible area` (see p.44). The `\showsequencelogo` command can be reversed by `\hidesequencelogo`.

Logo colors can be turned to 'Black' with the command `\clearlogocolors[⟨color⟩}` with the optional parameter not set. The optional parameter can be used to set all residue colors to ⟨color⟩, e.g. `\clearlogocolors[Blue]`. User specific logo color sets are defined by using `\logocolor{⟨residues⟩}{⟨color⟩}`, e.g. `\logocolor{DE}{Red}` `\logocolor{CM}{Yellow}` etc.

It is common practice for protein sequence logos to correct amino acid frequencies to the background frequency in the alignment, which usually differs from the equal distribution of 5% for each residue. Frequency correction can be turned on by `\dofrequencycorrection` and off by `\undofrequencycorrection`.

The vertical extent of the logo can be changed by `\logostretch{⟨factor⟩}`, e.g. `\logostretch{1.5}`. The width of the logo characters is dependent on the character width set for the alignment, see `\charstretch` on p.56.

Finally, the bit-scale can be turned off and on using `\hidelogoscale` and `\showlogoscale[⟨color⟩]{⟨position⟩}`, respectively, with

⟨*position*⟩ `left`, `right`, or `leftright` and an optional ⟨*color*⟩. A name for the sequence logo can be set, which is displayed next to the scale by `\namesequencelogo{`⟨*name*⟩`}`.

4.4.2 Subfamily logos

Subfamily logos provide a novel tool to visualize subfamily-specific sequence deviations at alignment positions with a high information content in an intuitive way [14].

This is achieved by subtracting from the frequency of a residue within a pre-defined subset of sequences, i. e. a subfamily, the frequency of this residue in the remaining set of sequences. The difference is then weighted by the information content, see above section on sequence logos. An example is shown on p.32.

Subtraction of frequencies produces values from -1 to 1. Positive values correspond to residues which are characteristic for the subfamily (shown upright in the output), negative values to those that are typical for the remaining sequences (shown upside-down). Positions with an equal distribution of the residue result in a zero value.

Subfamily logos are displayed analogous to sequence logos by the command `\showsubfamilylogo[`⟨*colorset*⟩`]{`⟨*top/bottom*⟩`}` and hidden by `\hidesubfamilylogo`. To calculate a subfamily logo, it is further required to define a subfamily within the alignment by `\setsubfamily{`⟨*seqrefs*⟩`}`, e.g. `\setsubfamily{1-10,20,AQP3}`.

For coloring residues, display/stretching of the scales, and frequency correction the same commands as for sequence logos apply with two exceptions. First, subfamily logos contain negative values, which can be displayed `\shownegatives[`⟨*weak, medium, strong*⟩`]` or hidden `\hidenegatives`. Without the optional parameter negative residues will be tinted by 50%, i.e. `medium`. This greatly improves readability. Second, a name for the subfamily logo is set by `\namesubfamilylogo[`⟨*neg.name*⟩`]{`⟨*pos.name*⟩`}` with a required name for the positive part of the logo and an optional name for the negative part.

In order to better recognize relevant positions in the subfamily logo, a bit-value can be set above which the deviation is considered relevant by the command `\relevance{`⟨*bit-value*⟩`}`. If this command is not given 2.321 bit is assumed for proteins, i. e. $\log_2 5$, and 1 bit for DNA, i. e. $\log_2 2$. Such positions will be labeled by `\showrelevance[`⟨*color*⟩`]{`⟨*symbol*⟩`}`, e. g. `\showrelevance[Blue]{$\nabla$}`. The symbol will be hidden with

`\hiderelevance`.

4.5 Appearance of the sequence lines

4.5.1 Names, numbers and gaps

Many parameters that influence the appearance of the actual sequence lines can be changed for customization. Thus, the sequence names can be shown colored via ⟨*color*⟩ either left or right by

`\shownames[`⟨*color*⟩`]{`⟨*position*⟩`}`

with ⟨*position*⟩ set to `left` or `right`. The numbering can be displayed either left or right and even on both sides by

`\shownumbering[`⟨*color*⟩`]{`⟨*position*⟩`}`

with ⟨*position*⟩ `left`, `right` or `leftright`. Both, names and numbering can be displayed on the same side. The colors can also be set with `\namescolor{`⟨*color*⟩`}` and `\numberingcolor{`⟨*color*⟩`}`, respectively. TEXshade uses the sequence names from the alignment input file. This can cause some problems during the TEX-run when special characters are present in those names! TEXshade does not accept the following characters in sequence names: `\ { } @` spaces and the tilde. Those have to be replaced in the input file. The characters `#` and `%` can only be used with a leading backslash, e. g. `\#`. This must also be changed in the input file. All other special characters should be displayed properly.

Sequence names that are accepted by TEXshade can further be changed in the `texshade` environment:

`\nameseq{`⟨*seqref*⟩`}{`⟨*name*⟩`}`

⟨*seqref*⟩ selects the sequence whose name is to be changed. The basis for the ⟨*seqref*⟩ is the appearance in the alignment input file with the top sequence = 1, or the old name. In order to change the colors only of some sequence names or numbers the commands `\namecolor{`⟨*seq1*⟩`, ... ,`⟨*seq n*⟩`}{`⟨*color*⟩`}` and `\numbercolor{`⟨*seq1*⟩`, ... ,`⟨*seq n*⟩`}{`⟨*color*⟩`}` are provided.

In order to hide all names or the numbering use the command `\hidenames` or `\hidenumbering`. If only the names or numbers of some sequences should be hidden apply `\hidename{`⟨*seq1*⟩`, ... ,`⟨*seq n*⟩`}` or `\hidenumber{`⟨*seq1*⟩`, ... ,`⟨*seq n*⟩`}`, respectively.

In some situations, e.g. when only sections of sequences are displayed, one may not want to have the residue numbering start out with number 1. The command \startnumber[⟨*start..stop*⟩]{⟨*seqref*⟩}{⟨*startnumber*⟩} allows one to set the starting number of any sequence to any value incl. negative values but except '0' which is not usually used in sequence numbering (the transition from negative to positive values is like this: ... −2, −1, 1, 2 ...). If, however, the use of the number '0' is wanted as sometimes in sequence logos this can be turned on by \allowzero and off with \disallowzero. The optional parameter can be used to truncate the sequence display to a certain section (see also \setends below).

TEXshade needs to know the correct length of the sequences to be able to break them right after the last residue. If MSF files are used as an input the length is already given but the calculation is usually wrong because the gaps are also counted. Thus, TEXshade counts the number of residues during each run by itself and stores the values in the .aux file. That means that it needs two runs to get the numbers right. Again, this is only important if the gap symbol after the sequence end should be suppressed, see below (\hideleadinggaps).

If you know the correct length of the sequences you can use the command

\seqlength{⟨*seqref*⟩}{⟨*length*⟩}

in order to set the values by hand and have the gaps break properly already in the first TEX run.

Example: \seqlength{1}{346} means that sequence no. 1 is 346 residues long.

TEXshade can display a section of the complete alignment without the need to edit the alignment input file or even to re-calculate the entire alignment. This allows one to use one single alignment of the full length proteins or open reading frames for multiple visualizations of different sections in a document as done in this manual. Thus, the file AQPpro.MSF contains the full-length multiple protein alignment of five aquaporins but only sections are displayed as examples in 1.5.1 through 1.5.5. The definition of a section is done by

\setends[⟨*startnumber*⟩]{⟨*seqref*⟩}{⟨*start..stop*⟩}.

Again, ⟨*seqref*⟩ is the sequence number based on the appearance in the alignment file, or the name; further, in order to use the consensus as a measure for the sequence section the string 'consensus' as

⟨*seqref*⟩ is accepted. The specified sequence is truncated at positions ⟨*start*⟩ and ⟨*stop*⟩. All other sequences are cut accordingly. If the number of the first residue in the sequence is set to a new value with the `\startnumber` command (s. a.) this is taken into account. The ⟨*startnumber*⟩ can be set as an optional parameter directly in the `\setends` command as well.

Some examples:

a) `\setends{1}{21..100}`

b) `\startnumber{1}{101} \setends{1}{121..200}`

Both commands select the same sequence section from the alignment but numbering for sequence 1 starts at position 21 in the first example and at position 121 in the latter.

c) `\setends[101]{1}{121..200}` equals example b.

d) `\startnumber[121..200]{1}{101}` also equals example b.

e) `\setends{consensus}{21..100}`

This may describe a very different section of the multiple sequence alignment because the consensus counts every position including gaps. Another possibility to label sequence positions is to switch on a ruler on the top or at the bottom of the sequence block using `\showruler[`⟨*color*⟩`]{`⟨*position*⟩`}{`⟨*seqref*⟩`}`. The residue ruler of one sequence ⟨*seqref*⟩ or the consensus (declare '`consensus`' as ⟨*seqref*⟩) can be displayed at ⟨*position*⟩ `top` or `bottom`. The ruler is hidden with `\hideruler`. The steps between two numbers are set by `\rulersteps{`⟨*number*⟩`}`. If the steps are set to be very close (< 4) or when every position is numbered, the numbering is automatically rotated by 90°. Using `\rotateruler` and `\unrotateruler` this can be done and undone manually. In order to change the ruler color use the optional parameter or the command `\rulercolor{`⟨*color*⟩`}`. Also, the label and its color at individual ruler positions can be changed by the user to a string using `\namerulerpos{`⟨*number*⟩`}{`⟨*text*⟩`[`⟨*color*⟩`]}` (see example on p.).

Further, the symbol which is displayed in sequence gaps is freely selectable with `\gapchar{`⟨*symbol*⟩`}`. ⟨*symbol*⟩ can be any character or symbol. If math symbols are to be used math mode must be activated by `$` characters, i. e. `\gapchar{{$\triangle$}}`. Note the double curly braces in the last command. Everytime a 'complex' character is used, i. e. a character definition consisting of more

than one letter, it must be braced in order to be interpreted as one character. One exception is `\gapchar{rule}`; with this parameter lines are drawn in the sequence gaps with a certain thickness defined by `\gaprule{`⟨*thickness*⟩`}`, e.g. `\gaprule{1.5pt}`. The colors of the gaps and gap symbols are set by `\gapcolors{`⟨*symbol color*⟩`}{`⟨*background color*⟩`}`.

There are some discussions whether or not to display gap symbols before and after the actual sequence. Since v1.3a one can control the appearance of those gap symbols by the commands `\showleadinggaps` and `\hideleadinggaps`. By default, leading gaps are indicated by symbols despite my personal thinking that it could suggest that there are some not displayed residues upstream resp. downstream of the gap.

4.5.2 Hiding, killing, separating and ordering

If one or more sequences from the alignment input file should be used for the calculation of the consensus but it is desired not to display these sequences in the final output use the command `\hideseq{`⟨*seq1*⟩`,`⟨*seq2*⟩`,...,`⟨*seq n*⟩`}`. For consecutive sequence numbers a dash can be used, e.g. `\hideseq{1-3}` instead of `\hideseq{1,2,3}`. Decending series are also permitted, e.g. `\hideseq{3-1}`. This command allows one for example to hide the sequence which has been defined as the consensus sequence with `\constosingleseq`. When all sequences should be hidden, e.g. to show a sequence logo alone, one can simply say `\hideseqs`. This command is reversed by `\showseqs`.

In order to completely exclude sequences the command `\killseq{`⟨*seq1*⟩`,`⟨*seq2*⟩`,...,`⟨*seq n*⟩`}` is provided. Again, for number series the dash can be used (s. a.). The designated sequences are neither displayed nor considered for the calculation of the consensus. This is another possibility to comment out sequences in addition to the use of an exclamation point infront of the `Name:` definition in an MSF-file (see figure on page).

The command `\donotshade{`⟨*seq1*⟩`,`⟨*seq2*⟩`...,`⟨*seq n*⟩`}` makes one or more sequences (remember the dash, s. a.) appear unshaded in black letters on white background. This does not influence any other sequences or the consensus calculation.

If a very graphical output of the sequences is desired, the residue symbols or letters can be blanked out by `\hideresidues`. Now, only the shaded boxes are printed. In combination with `\gapchar{rule}`

one obtains alignments in a style à la Mondrian. The residues reappear with `\showresidues`.

If an alignment contains members of several subgroups of a protein or a gene family it may be rather helpful to visualize the group divisions by a separation line. Therefore, the command `\separationline{⟨seqref⟩}` is applicable. This command inserts vertical space after the sequence which is refered to by ⟨seqref⟩. How much space is inserted is defined by one of the following commands: `\smallsep`, `\medsep` (default) or `\bigsep`. These lengths correspond to the known `\small-`, `\med-` and `\bigskip` commands. With `\vsepspace{⟨length⟩}` any length with any TeX unit can be assigned, e. g. `\vsepspace{2mm}`.

The sequence order given by the alignment input file is easily reorganized by `\orderseqs{⟨seq1⟩,⟨seq2⟩,...,⟨seq n⟩}` without the need for editing the alignment input file (which would be a big copy'n'paste job). Make sure that all sequences are assigned in this command. If there are more sequences present than numbers or names in the command an error message will occur. Here also, the dash can be used for sequence number series. Example: `\orderseqs{1-3,6-4,7}` is equivalent to `\orderseqs{1,2,3,6,5,4,7}`. Reordering of sequences only changes the output; all commands using the parameter ⟨seqref⟩ are not influenced, because ⟨seqref⟩ always corresponds to the appearance in the alignment file. Thus, to completely reverse the order of a five sequence alignment simply type `\orderseqs{5-1}`.

4.5.3 Residues per line and further settings

By default TeXshade puts the highest possible by five divisible number of residues in one line depending on the `\textwidth`. With `\residuesperline{⟨number⟩}` a new value can be set. If this value exceeds the highest possible number of residues per line it is ignored; lower values are accepted of course. But also in the latter case the number of residues printed per line is rounded such to be divisible by five. To force TeXshade to set lines with exactly the desired number of residues use the asterisk-extended command `\residuesperline*{⟨number⟩}`. Expect multiple *overfull hbox* errors after this command, because in this mode TeXshade does not check the length of the lines any more.

TeXshade calculates the dimensions of a shaded box from the width and height of the uppercase letter 'M' and the depth of the lowercase 'g'. Depending on the font used for the sequence

residues the box dimensions might not be fully satisfactory. With `\charstretch{`⟨*factor*⟩`}` and `\linestretch{`⟨*factor*⟩`}` the width and height/depth, respectively, of the boxes can be multiplied individually by a ⟨*factor*⟩ to stretch (> 1) or shrink (< 1) the dimensions.

The reserved space for the sequence numbering is set by the command `\numberingwidth{`⟨*n digits*⟩`}`. Here, the default setting is four-digit numbering, i. e. -999 through 9999. If this range is to be changed assign the desired number as parameter ⟨*n digits*⟩, e. g. `\numberingwidth{111111}` reserves space for 6 digit numbering.

The vertical space between the sequence blocks can be controlled by the commands `\smallblockskip`, `\medblockskip` (default setting), `\bigblockskip` or `\noblockskip`. Further, the command `\vblockspace{`⟨*length*⟩`}` allows one to set a defined space length using any TeX unit, e. g. `\vblockspace{0.4in}`.

Two more commands set the space between the sequence blocks to be flexible (`\flexblockspace`) (default) or fixed (`\fixblockspace`). Flexible means, that only the vertical white space between the blocks is kept to the settings by e. g. `\medblockskip`. This results in flexible space between the actual blocks depending on the presence of feature lines. When switching to fixed space the distance of the blocks is kept constant by using more white space between blocks without feature lines. Thus, a difference between flexible and fixed space will only be noticeable when features are used.

The position of the output can be aligned left, right or centered on the page by `\alignment{`⟨*position*⟩`}` with the ⟨*position*⟩ parameter `left`, `center` or `right`.

4.5.4 Fingerprinting

An easy way to gain an overview on complete alignments is provided by displaying a so called alignment 'fingerprint'. In this style the whole sequence can be shown in one line. Due to the lacking space the residue names are hidden and the shaded boxes are reduced to thin vertical colored lines. The command `\fingerprint{`⟨*res. per line*⟩`}` takes one argument stating the desired number of residues per line, e. g. `\fingerprint{1000}`. All TeXshade commands are compatible with `\fingerprint`, i. e. all shading modes are applicable for displaying overviews on similarity or every functional aspect. Also, all kinds of labeling—as described in the following—work with this command.

4.6 Individual shading and labeling of sequence stretches

Computer calculated alignment shading is informative—but even more information can be visualized by manual labeling of positions and regions of interest with different colors, text styles or graphical marks and descriptive text. All this is provided by easy to handle T_EXshade commands.

4.6.1 Manual shading of regions and blocks

Besides the shading calculated by T_EXshade any region can be shaded manually with a color specified by the user. This is very useful to highlight secondary protein modification sites such as phosphorylation or glycosylation sites or longer motifs for example protein/protein interaction sites. This is done by the use of the following command:

\shaderegion{⟨*seqref*⟩}{⟨*start1*⟩..⟨*stop1*⟩,⟨*start2*⟩..⟨*stop2*⟩,
        ...,⟨*start n*⟩..⟨*stop n*⟩}{⟨*res.col.*⟩}{⟨*shad.col.*⟩}

Example: in order to shade residue number 13 and the region 20–30 of sequence number 1 in red letters on green ground type the following command:

\shaderegion{1}{13..13,20..30}{Red}{Green}

If the consensus is to be shaded use consensus as ⟨*seqref*⟩.
In analogy to \shaderegion which is restricted to one single sequence \shadeblock shades the corresponding region in all other sequences as well except the consensus. If also the consensus is to be shaded define the region using consensus as ⟨*seqref*⟩.

\shadeblock{⟨*seqref*⟩}{⟨*start1*⟩..⟨*stop1*⟩,⟨*start2*⟩..⟨*stop2*⟩,
        ...,⟨*start n*⟩..⟨*stop n*⟩}{⟨*res.col.*⟩}{⟨*shad.col.*⟩}

4.6.2 Emphasizing and tinting regions and blocks

If it is prefered to keep the calculated shading colors but distinct regions or blocks are yet to be emphasized one can use the following commands to change the font style of such regions:

\emphregion{⟨*seqref*⟩}{⟨*start1*⟩..⟨*stop1*⟩,⟨*start2*⟩..⟨*stop2*⟩,
                ...,⟨*start n*⟩..⟨*stop n*⟩}

and

```
\emphblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
                            ...,⟨start n⟩..⟨stop n⟩}
```

Which style TEXshade uses for emphasizing regions is defined by
`\emphdefault{⟨style⟩}`. Default setting is the *italics* font shape (set
by `\emphdefault{it}`). In order to change this setting choose one of
the styles `bf, md, up, it, sl, rm, sf, tt`.
Example: `\emphdefault{bf}`

Further, it is possible to tint the region or block in question by using
the commands (for example see hydropathy-figure on page 19):

```
\tintregion{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
                            ...,⟨start n⟩..⟨stop n⟩}
```

and

```
\tintblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
                            ...,⟨start n⟩..⟨stop n⟩}
```

The level of tinting in the region in question can be set by
`\tintdefault{⟨level⟩}` with `weak`, `normal`, and `strong` as possible
⟨level⟩s.
Another option is to draw a bounding box around the sequence block
in question (for an example see diversity mode-figure on page 17) with
the command:[22]

```
\frameblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
                    ...,⟨start n⟩..⟨stop n⟩}{⟨color⟩[⟨length⟩]}
```

With the optional parameter the default line thickness of the frame can
be changed, example: `\frameblock{1}{10..20,50..70}{Red[2pt]}`

4.6.3 Hiding blocks of the alignment (experimental!)

*Be aware that this feature is still highly experimental. There are many
instances in which this command fails. Please be patient until every-
thing is fixed.*
Complete blocks of the alignment incl. all sequences, consensus and la-
bels can be hidden, e.g. to eliminate uninteresting alignment stretches
or to condense the output, by:

```
\hideblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
        ...,⟨start n⟩..⟨stop n⟩}{⟨label⟩[⟨color⟩]}{⟨text⟩[⟨color⟩]}
```

---

[22]Thanks to Alan Robinson for inspiration.

Here, $\langle seqref \rangle$ denotes the sequence by its number of name which is used to define the alignment positions $\langle start1 \rangle..\langle stop1 \rangle$ etc. to be hidden. A symbol can be displayed as a $\langle label \rangle$ of the alignment deletion, i.e. `circ`, `triangle`, `square`, or `none`. A color can be assigned to the label by the optional parameter $\langle color \rangle$. Further, an explanatory text can be displayed above the symbol $\langle text \rangle$ with an optional $\langle color \rangle$. For how to change the appearance of the font see p.74.

Examples:
```
\hideblock{1}{20..80}{circ}{condensed![Red]}
\hideblock{1}{20..80}{triangle[Red]}{hidden block}
\hideblock{AQP1}{20..80}{none[Blue]}{$\Delta$20-80)}
\hideblock{consensus}{20..80,100..150,200..220}{none}{}
```

It may be advisable to show a ruler (probably single-stepped, see p.54) to further indicate the alignment deletion. Make sure that there are no feature definitions (also frames, emphs, shades, tints) starting or ending within the hidden block. This may cause messy sequence labels.

4.6.4 Graphical labeling of sequence features

The `\feature` command is designed to fulfill most needs for the graphical labeling of sequence stretches and the setting of descriptive text. It needs five parameters:

`\feature{`$\langle position \rangle$`}{`$\langle seqref \rangle$`}{`$\langle start1 \rangle$`..`$\langle stop1 \rangle$`,` $\langle start2 \rangle$`..`$\langle stop2 \rangle$`,...,`$\langle start\ n \rangle$`..`$\langle stop\ n \rangle$`}{`$\langle labelstyle \rangle$`}{`$\langle text \rangle$`}`

In the following paragraphs all possible parameter settings of this rather complex but mighty command are discussed in detail. The parameter $\langle position \rangle$ tells TEXshade where to display the feature label, i. e. on the top of the alignment (`top`), or at the bottom (`bottom`). Further, there can be three more feature lines ontop of the top feature line (`ttop`, `tttop`, and `ttttop`) or below the bottom feature line (`bbottom`, `bbbottom`, `bbbbottom`). Thus, up to eight features overlapping in eight different lines may be displayed. Depending on the content of the feature lines the gaps between them might be not satisfactory. Therefore, eight separate commands can be employed to change the space below `ttttop`, `tttop`, `ttop`, or `top` (`\topspace{`$\langle length \rangle$`}` etc.), and above `bottom`, `bbottom`, `bbbottom`, or `bbbbottom` (`\bottomspace{`$\langle length \rangle$`}` etc.). Use positive values to further separate the lines, e. g. `\ttopspace{3mm}` or negative values to reduce the space, e. g. `\bottomspace{-0.1in}`.

The argument $\langle seqref \rangle$ and the third parameter containing the definitions of the specified regions are identical to the ones described before in several commands, e.g. `\ruler` (4.5.1) or `\shaderegion` (4.6.1). New is the fourth parameter for the definition of the label style. There are many possibilities like braces, helices, boxes, arrows, bars, any fill character, bar graphs, color scales or even translations of the specified regions.

Braces:
In order to display an over- or underbrace as a label use the parameter `{brace}`. Depending on the $\langle position \rangle$ (`ttttop`, `tttop`, `ttop`, `top`, `bottom`, `bbottom`, `bbbottom`, or `bbbbottom`) the respective brace is displayed. The standard color of braces is black. It can be changed by an optional parameter directly after the definition of the symbol, e.g. `{brace[Red]}`.

Protein α-Helices:
The parameter `{helix}` will plot a symbolized α-helix as a label. The standard color of the helix spiral is black. It can be changed by an optional parameter directly after the definition of the symbol, e.g. `{helix[Red]}`.

Filling a stretch with a symbol:
A region can be filled with any character for labeling purposes using the parameter `{fill:`$\langle symbol \rangle$`}`. The $\langle symbol \rangle$ is freely selectable; the usage is like in `\gapchar` (4.5.1). Do not use spaces before or after the expression $\langle symbol \rangle$; this will shift the symbols to the respective direction. The standard color of the fill symbol is black. It can be changed by an optional parameter directly after the definition of the symbol, e.g. `{fill:$\bullet$[Red]}`.

The `\feature` command does not like special characters in text mode, e.g. `\dag`. One has to use the math version of those symbols between `$`-signs. The following quite common text symbols have also a math equivalent[23]:

---

[23]Thanks to Darrell Conklin for reporting this problem

symbol	command	description
†	$\dagger$	dagger
‡	$\ddagger$	double dagger
¶	$\mathparagraph$	paragraph mark
§	$\mathsection$	section mark
$	$\mathdollar$	dollar
{	$\lbrace$	left brace
}	$\rbrace$	right brace

Labeling restriction or protease cutting sites:

If a label is needed that points between two residues, e.g. for showing restriction sites, simply use the feature style {restriction[⟨*color*⟩]}. This will show a filled triangle with the tip right between the residues to be labeled, e.g. \feature{top}{1}{25..26}{restriction[Blue]}{EcoR I}.

Boxes:

Boxed text is printed using the parameter {box:⟨*text*⟩}. By default black letters in a white framed box are displayed. In order to change these colors optional parameters can be included in the argument:

{box[⟨*framecolor,boxcolor*⟩][⟨*length*⟩]:⟨*text*⟩[⟨*textcolor*⟩]}.

If the box frame and fill colors are the same it is sufficient to use only this one color as an argument in the command. The optional parameter ⟨*length*⟩ defines the thickness of the box frame. If this parameter is not set in the command the value from the \featurerule{⟨*length*⟩} command (see below) is used.

Examples:

```
{box[Blue]:$\alpha$~helix[Yellow]}
{box[Blue,Red]:$\alpha$~helix[Yellow]}
{box[Blue,Red][2pt]:$\alpha$~helix[Yellow]}
```

Horizontal bars and arrows:

For displaying bars and arrows a simple selection scheme consisting of three consecutive characters is used as the ⟨*labelstyle*⟩ parameter. Each bar or arrow is defined by its left end, the middle part, and the right end. The following table gives some examples for the construction of arrows and bars.

	middle	
left end ↓ right end		
---		plain bar
===		double bar
-->		right arrow
'->		right arrow with up hook
<-|		left *maps to* arrow
<-o		left arrow with ball at right end
<=>		double arrow, two heads
,-,		plain bar with down hooks
|=|		double bar with vertical ends

All combinations of the left-end-characters (-=<',|o), the middle-characters (-=), and the right-end-characters (-=>',|o) are allowed and produce the desired arrow or bar. The color is changed as described above. The thickness can be generally set by the separate command `\featurerule{⟨length⟩}` with any TEX measure as ⟨length⟩, e.g. `\featurerule{3pt}`. This value is then used for all arrows, bars, and boxes (see above) throughout the alignment. If an individual thickness for a particular arrow should be set one can add an optional parameter to the ⟨labelstyle⟩ parameter, e.g. `{o->[Red][1mm]}`. Similar to the boxes described above, a text can be put on the arrow or bar, e.g. `{<->[Red][1mm]:$\beta$-sheet[Blue]}`.

In TEXshade versions before v1.9, the original LATEX-arrows were used. These have now been replaced by more modern looking arrows with scalable line thickness. If the classical look is requested, use v instead of < or > in the arrow definition, e.g. `{--v}`, to get them back. The new arrow style makes use of of the AMS math symbol font (amssymb.sty). Thus, in order to display the arrow heads correctly make sure that this style is present on your system (usually it is in a common LATEX installation).

Sequence translations:
With the option `{translate}`, sequence stretches can be translated from nucleotide to peptide sequences as well as backtranslations from peptide to nucleotide sequences are possible. Default setting for the translations is the standard genetic code. Of course, the codons can be re-defined by the user. The command `\codon{⟨amino acid⟩}{⟨triplet1, ..., triplet n⟩}` has been implemented for this issue. The usage is simple. Replace ⟨amino acid⟩ by the single letter code of the amino acid to be defined and add a list of triplets for this residue. Example definition for the amino acid *alanine*:

```
\codon{A}{GCA,GCG,GCC,GCT,GCU,GCN}
```

Note the last triplet in the list. It contains an ambiguity code N which stands for *any* nucleotide. This triplet has been added at the last position because the last triplet is used for the generation of the backtranslated nucleotide sequence from a peptide. Two files are included in the TEXshade distribution as examples (`standard.cod, ciliate.cod`). If you want to define a new genetic code store your commands in a file like the examples. Such files with the suffix `.cod` can be loaded in the TEXshade environment by `\geneticcode{⟨filename⟩}`, e. g. `\geneticcode{ciliate}`. Do not designate the suffix `.cod` in ⟨*filename*⟩. Please note, when inspecting the example files, that only the exchanges compared to the standard code need to be defined in a new genetic code file.

When DNA sequences are translated to protein the resulting amino acids are aligned to the second nucleotide of each triplet. It is more difficult to produce a satisfactory display of backtranslated nucleotide sequences due to the lack of space. You need thrice as much space than the original peptide sequence, because single letter amino acid code is translated to a triplet code. Therefore, the user can choose from five display styles for backtranslations depending on personal preferences:

`\backtranslabel[⟨size⟩]{⟨style⟩}`, with

$$
\begin{aligned}
\{⟨style⟩\} &= \{\texttt{horizontal}\} \\
&= \{\texttt{alternating}\} \\
&= \{\texttt{zigzag}\} \\
&= \{\texttt{oblique}\} \\
&= \{\texttt{vertical}\}
\end{aligned}
$$

⟨*size*⟩ can be any TEX size from `tiny` up to `Huge`, but `tiny` is recommended (and default setting). Translations can be colored as all other labels, see above.

Bar graphs and color scales:
Sequence related numeral data, such as hydropathy or solvent accessibility data etc., can be shown in a feature line as bar graphs or color scales. The data are (a) pre-defined or calculated by TEXshade due to amino acid properties or conservation, (b) are provided in a separate file or (c) may be entered by hand in the `\feature` command.

(a) Currently, three different properties can be plotted, i.e. `hydrophobicity`, `molweight`, and `charge`. Further, the level of sequence conservation at the given protein sequence stretch can be shown (`conservation`).

(b) The format of a data file is simple: every value must appear in a separate line. Numbers and the Java-typical 'NaN' for 'Not a Number' are permitted. Comments are allowed, because TeXshade ignores all lines starting with a letter except 'NaN' lines (avoid '`-`' as the first character of a comment line as this is interpreted as a negative number). Make sure that there are as many values as positions defined as the sequence stretch in the feature command. TeXshade will read this file and determine the minimal and maximal values. These data are then normalized for plotting. Due to TeX's limited calculation capabilities no values above 10737 are allowed and the difference between minimum and maximum must not exceed this very number. Values below 0.001 may be susceptible to major rounding errors. Thus, try to provide your data already normalized to moderate scales, e.g. $0.0 - 1.0$ or $-100 - 100$.

(c) Data which is directly entered in the `\feature` command must be normalized to integer values with a maximal difference of 100 between the highest and lowest value, e.g. $-50 - 50$ or $0 - 100$.

For (b) and (c), the range to be plotted can be set by hand as an optional parameter in the `\feature` command. This can be necessary when the data file contains values between e.g. -0.44 and 0.87. Without help TeXshade will assume -0.44 as minimum and 0.87 as maximum. But if the actual range to be plotted should be $-1.0 - 1.0$ this needs to be set manually, see examples below. Be aware of the fact, that if you define a scale by hand, which is more narrow than the values of the input, this will stretch the bars accordingly. It is NOT recommended to use this method for stretching bars vertically. Instead another command has been introduced. The plotted bars can be stretched by a factor if the appearance is not as desired: `\bargraphstretch{`⟨*factor*⟩`}`. Here, the factor is multiplied with the bar length, e.g `\bargraphstretch{2}` will double the bar height, `\bargraphstretch{0.5}` will make them half as high. Similarly, color scales can be stretched vertically with `\colorscalestretch{`⟨*factor*⟩`}`.

The default color of bar graphs is gray and can be changed by an optional parameter at the end of the `label` definition. Further, an optional background color can be chosen for the bars. Doing so will visualize the maximal bar extension.

Default for color scales is a 5% gray scale from very light gray to black
(Gray). More colorful scales have been implemented, i.e. BlueRed,
RedBlue, GreenRed, RedGreen, ColdHot and HotCold, the latter two
being particularly useful for ranges from negative to positive values.
Further, a scale called T-Coffee is available if T-Coffee shading in-
formation has been imported as the \shadindmode 4.1.

The general format of this feature label definition for bar graphs is:

{bar[⟨*min*⟩,⟨*max*⟩]:⟨*properties/file/data*⟩[⟨*color(,bgcolor)*⟩]}

and for color scales:

{color[⟨*min*⟩,⟨*max*⟩]:⟨*properties/file/data*⟩[⟨*scale*⟩]}

Some examples:

```
{bar:conservation}
{bar:conservation[T-Coffee]}
{bar:hydrophobicity}
{bar:charge[Red]}
{bar:molweight[Red,Gray10]}
{bar:10,20,30,40,50[Red]}
{bar[-20,40]:-10,0,10,20,30[Red,Gray10]}
{bar:data.txt}
{bar[-10,10]:data.txt[Red,Gray10]}
{color:conservation[BlueRed]}
{color:conservation[T-Coffee]}
{color:hydrophobicity[GreenRed]}
{color:charge}
{color:molweight}
{color[-10,10]:data.txt[ColdHot]}
{color[-0.1,0.1]:otherdata.txt[ColdHot]}
```

See also the example output in section 1.6 on page 24.

No graphical label, only text:
If no graphical label is wanted the fourth parameter of \feature can
be empty braces.

Finally, the fifth parameter of the \feature command contains the descriptive text for the labeled region. Type whatever you want incl. symbols and math chars. The text field can also contain sequence translations. In this case just set ⟨*text*⟩ = {translate}. There is a command for setting the size and style of backtranslated sequences in the feature ⟨*text*⟩ which corresponds to the one described above:

\backtranstext[⟨*size*⟩]{⟨*style*⟩}

Again, the color can be set by an optional parameter appended to the text. For how to change the font size of text or symbols in the feature style line (featurestyles) or the in descriptive text line (features) see section 4.9.1, page 73.

Another set of commands can be used to set a name for a feature line, which is printed together with the sequence names at the left or right side of the alignment, i. e.

\showfeaturename{⟨*ttttop...bbbbottom*⟩}{⟨*name*⟩},

\showfeaturestylename{⟨*ttttop...bbbbottom*⟩}{⟨*name*⟩}

\hidefeaturename{⟨*ttttop...bbbbottom*⟩}

\hidefeaturestylename{⟨*ttttop...bbbbottom*⟩}

\hidefeaturenames, and \hidefeaturestylenames.

Using \showfeaturename will print the name in the same line as the descriptive text of the feature whereas \showfeaturestylename will put the name in the same line as the feature symbols.

The color of such names can be generally changed with

\featurenamescolor{⟨*color*⟩} and

\featurestylenamescolor{⟨*color*⟩}

or individually with

\featurenamecolor{⟨*ttttop...bbbbottom*⟩}{⟨*color*⟩} and

\featurestylenamecolor{⟨*ttttop...bbbbottom*⟩}{⟨*color*⟩}

See section 5 for how to select colors in TEXshade.

Font styles can be set as usual, e. g. \setsize{featurenames}{large} or \featurestylenamesrm etc. (see section 4.9.1).

Examples for the appearance of features are given in the overview section (1), see:

similarity mode (1.5.2): fill-character; here, only one position is labeled. It is also possible to label a longer stretch, then, the character is printed several times to fill the specified region.

```
\feature{top}{1}{93..93}{fill:$\downarrow$}{first...}
\feature{bottom}{1}{98..98}{fill:$\uparrow$}{second...}
```

T-Coffee mode (1.5.3): `T-Coffee` color scale

```
\feature{top}{1}{30..63}{color:conservation[T-Coffee]}{}
\showfeaturestylename{bottom}{cons}
```

diversity mode (1.5.4): frames, text only

```
\feature{top}{1}{77..109}{}{AQP2 species variants}
\frameblock{1}{82..82,106..106}{Red[1pt]}
```

functional mode (1.5.5): bar graph, color scale, tinting, box, arrow, translation, brace, helix

```
\feature{top}{3}{153..165}
        {bar[-50,50]:-50,-45,-40,...,40,45,50}{}

\feature{top}{3}{167..186}
        {color:5,10,15,...,90,95,100[ColdHot]}{}

\feature{top}{1}{158..163}{brace}{tinted}
\tintblock{1}{158..163}

\feature{top}{1}{138..157}
        {box[Blue,Red][0.5pt]:$\alpha$~helix[Yellow]}
        {transmembrane domain 4}
\feature{top}{1}{164..170}{o->[Red]}{trans. dom. 5}
\feature{top}{1}{158..163}{translate[Blue]}{}
\backtranslabel{oblique}
\feature{bottom}{1}{158..163}
        {brace[Blue]}{loop D[Blue]}

\feature{top}{1}{138..157,164..170}{helix}{membr.}
\feature{top}{1}{158..163}{---}{loop}
\featurerule{1mm}
```

bar graphs and color scales (1.6): sequence conservation, charge, molecular weight, hydrophobicity

```
\feature{ttop}{1}{138..170}{bar:conservation}{}
\feature{top}{1}{138..170}{color:charge}{}
\feature{bottom}{1}{138..170}
        {color:molweight[ColdHot]}{}
\feature{bbottom}{1}{138..170}
        {bar:hydrophobicity[Red,Gray10]}{}
```

68

4.6.5 Including secondary structure information

The DSSP [9], STRIDE [10], PHD [11] and HMMTOP [12] algorithms produce secondary protein structure predictions. PHD files contain both, secondary structure information and topology data. This information can be displayed in an alignment by one of the commands:

\includeDSSP	sec. structure calculated by DSSP
\includeSTRIDE	sec. structure calculated by STRIDE
\includePHDsec	sec. structure calculated by PHD
\includePHDtopo	topology data calculated by PHD
\includeHMMTOP	topology data calculated by HMMTOP

The syntax is \includeDSSP{⟨*seqref*⟩}{⟨*filename*⟩}, with seqref indicating the number or name of the sequence for which the secondary structure data is calculated and filename designating the corresponding structure file to be included.

Several types of secondary structures are predicted by these programs; in order to designate them in TEXshade use the names from the right column:

secondary structure	designation
DSSP and STRIDE	
4-helix (α-helix)	alpha
isolated β-bridge	bridge
extended strand (β-strand)	beta
3-helix (3_{10}-helix)	3-10
5-helix (π-helix)	pi
H-bonded turn	turn
PHDsec	
helix	alpha
sheet	beta
PHDtopo and HMMTOP	
internal region	internal
external region	external
transmembrane domain	TM

By default all three types of helices and the strands are displayed whereas turns and bridges are skipped. If it is desired to shown them

69

as well, call for example `\shownonDSSP{bridge,turn}`. In analogy to this example all structure features can be activated in DSSP, STRIDE, PHDsec, PHDtopo and HMMTOP. In order to hide certain structure types use for example `\hideonDSSP{3-10,pi}`.

The DSSP format has two columns of sequence numberings. The first column is consecutive, whereas the second column contains the actual sequence numbering. This can be different from the first column when sequence parts are missing in the DSSP file. One can choose which column will be read by TEXshade by '`\firstcolumnDSSP`' and '`\secondcolumnDSSP`'. The second column is still default.

The HMMTOP algorithm can present its results as plain text or as HTML—plain text needs to be selected here. Further, the output can be formatted in a single line or in an extended form (see the HMMTOP documentation). Both can be read and interpreted by TEXshade. Importantly, HMMTOP files can contain topology predictions of multiple sequences. TEXshade tries to find the correct data based on the respective sequence name. If the sequence name is not found in the file, the first topology data is used. Using an optional parameter (number of the prediction in the file or name) one can define which data from the file is to be used:

`\includeHMMTOP{`⟨*seqref in texshade*⟩`[`⟨*seqref in file*⟩`]}{`⟨*filename*⟩`}`

PHD predictions: when starting the PHD software do not restrict the prediction to secondary structure or topology alone. This leads to changes in the PHD output file which are not correctly interpretable by TEXshade due to ambiguities. There is no way around it—thus, run the full prediction.

Now, some information on how TEXshade extracts and displays secondary structure features. In short, it is a two step process. First, TEXshade analyzes the secondary structure file and extracts all necessary data. This data is converted into a format which is readable and processable by TEXshade using the `feature` command (see 4.6.4). This command allows one to label sequence stretches graphically. For a detailed explanation see the indicated reference. A list of feature commands is saved in a file with the ending '`.sec`' for DSSP, STRIDE and PHDsec or '`.top`' for PHDtopo. Then, in a second step, this file is loaded again and executed. When TEXshade encouters this file a second time, i.e. in a second TEX run, it uses the already existing file for the output. The great advantage of this method is its flexibility. Due to the simple reason that the feature file can be edited in the meantime. Thus, the user has the ability to change

the computer generated file according to his personal needs. On the other hand, one can force TEXshade to write a new file every time by the optional argument `[make new]` in the include command, e. g. `\includePHDsec[make new]{1}{AQP.phd}`.

Finally, the appearance of the feature labels can be assigned by the command

`\appearance{`⟨*filetype*⟩`}{`⟨*type*⟩`}{`⟨*position*⟩`}{`⟨*labelstyle*⟩`}{`⟨*text*⟩`}`.

Here, ⟨*filetype*⟩ stands for one of the following secondary structure file types: `DSSP`, `STRIDE`, `PHDsec`, `PHDtopo` or `HMMTOP` and ⟨*type*⟩ designates the secondary structure type as shown in the right column of the table above. The other arguments ⟨*position*⟩, ⟨*labelstyle*⟩ and ⟨*text*⟩ are almost as described in 4.6.4.   One further possibility is to include internal counters for each secondary structure type. Just add one of the following commands to the text in the feature description.

counter	*display*
`\numcount`	1, 2, 3 …
`\alphacount`	a, b, c …
`\Alphacount`	A, B, C …
`\romancount`	i, ii, iii …
`\Romancount`	I, II, III …

Examples:
```
  \appearance{DSSP}{alpha}{ttop}
           {-->}{$\alpha$-helix~\Alphacount}
  \appearance{PHDtopo}{TM}{bottom}
           {box[Blue]:TM\numcount[Yellow]}{}
```

4.7   Displaying and building legends

For each predefined shading mode TEXshade can print an appropriate legend to explain the used shading colors. The commands `\showlegend` and `\hidelegend` display or clear the legend at the end of the alignment. The legend is displayed by default beneath the first residue of the last alignment line. The location can be changed by `\movelegend{`⟨*x-offset*⟩`}{`⟨*y-offset*⟩`}`. Both parameters require a TEX length, e. g. `\movelegend{5cm}{-2cm}` moves the legend 5 cm to the right and 2 cm up.

The language for the descriptions is english by default; if the `\german.sty` package is active legend texts are in german. So far, german, spanish and english are implemented.

71

With the commands \germanlanguage, \spanishlanguage and \englishlanguage switching between the languages is made possible. For the addition of other languages contact me. Finally, the color of the describing legend texts can be set with the command \legendcolor{⟨*color*⟩}.

User defined legends are easily built with the following command \shadebox{⟨*color*⟩}. Use this command outside the T_EXshade environment, e. g. in the text or in the caption. As ⟨*color*⟩ any color can be designated (see section 5) or one of the following parameters:

- nomatch = the color used for nonmatching residues

- similar = the color used for similar residues

- conserved = the color used for conserved residues

- allmatch = the color used for highly conserved residues (if \allmatchspecial is active)

The command simply prints a shaded box in the specified color then a describing text can be appended. Examples:

```
\shadebox{nomatch}---nonmatching residues
\shadebox{similar}: similar residues
\shadebox{conserved}~conserved residues
\shadebox{Yellow}\quad PKA phosphorylation sites
```

4.8   Adding captions to the alignment

Since T_EXshade v1.5 captions can be added to the alignment. So far, captions were difficult to use when the alignment was bigger than one page and therefore did not fit into a figure environment. The T_EXshade captions behave exactly as normal figure captions. They adopt their style, use the figure counter number and appear in the list of figures as any other figure.

The usage is slightly different from normal captions but intuitive:

```
\showcaption[⟨position⟩]{⟨text⟩}
```

The optional ⟨*position*⟩ tells T_EXshade to put the caption on top or at the bottom of the alignment. If nothing is stated here the caption will appear at the bottom. The parameter ⟨*text*⟩ just holds the caption text as in the normal \caption. The command can be used at any

position within the `texshade` environment. A simple example would be:

```
\showcaption{A beautiful \TeXshade{} alignment.}
```

In order to show a short version of the caption in the "List of Figures" the `\shortcaption{⟨short caption text⟩}` command can be used.

4.9   Font handling

4.9.1   Changing font styles

The font styles for the numbering, the sequence names, the sequence residues, the descriptive feature texts and the legends can be changed by several commands.

```
\setfamily{⟨text⟩}{⟨family⟩}
\setseries{⟨text⟩}{⟨series⟩}
\setshape{⟨text⟩}{⟨shape⟩}
\setsize{⟨text⟩}{⟨size⟩}
```

The first parameter selects the text whose style is to be changed. Possible first parameters are `numbering`, `names`, `residues`, `features`, `featurestyles`, `hideblock`, and `legend`.

The style is set by the second parameter:

command	⟨2. parameter⟩	
\setfamily	rm	modern roman font family
	sf	sans serif font family
	tt	typewriter font family
\setseries	bf	bold face series
	md	normal series
\setshape	it	italics shape
	sl	slanted shape
	sc	small capitals shape
	up	upright shape
\setsize	tiny	the known TeX sizes
	scriptsize	
	footnotesize	
	small	
	normalsize	
	large	
	Large	
	LARGE	
	huge	
	Huge	

Example: `\setfamily{features}{it} \setseries{features}{bf}`

With the command

   $\setfont\{\langle text\rangle\}\{\langle family\rangle\}\{\langle series\rangle\}\{\langle shape\rangle\}\{\langle size\rangle\}$

all four font attributes of one ⟨text⟩ can be changed simultaneously. The order of the parameters is as indicated.

Example: `\setfont{features}{rm}{it}{bf}{normalsize}`

Further, short commands are provided to change single font attributes quickly. The following commands set attributes of feature texts.

\featuresrm	\featurestiny
\featuressf	\featuresscriptsize
\featurestt	\featuresfootnotesize
\featuresbf	\featuressmall
\featuresmd	\featuresnormalsize
\featuresit	\featureslarge
\featuressl	\featuresLarge
\featuressc	\featuresLARGE
\featuresup	\featureshuge
	\featuresHuge

74

Corresponding sets are provided for the numbering (\numberingrm etc.), featurestyles (featurestylesrm etc.), names (\namesrm etc.), featurenames (\featurenamesrm etc.), featurestylenames (\featurestylenames etc.), residues (\residuesrm etc.), hideblock labels (hideblockrm etc.), and legend texts (legendrm etc.).

4.9.2 Using PostScript fonts

As already mentioned TEXshade makes intensive use of POSTSCRIPT for shading. Now, that POSTSCRIPT output is active anyway, including POSTSCRIPT fonts is very easy. Just declare in the document header

\usepackage{⟨PS-font⟩}.

The typewriter font of TEX is always a topic of discussions. By including the package \usepackage{courier} TEX's typewriter font is replaced by the widely accepted COURIER. Have a look into the directory ..texinputs:latex:psnfss; there, some styles are located which exchange the common TEX fonts by POSTSCRIPT fonts, e.g. avant.sty, bookman.sty, chancery.sty, courier.sty, helvet.sty or utopia.sty. Depending on the style used the \rmdefault-, \sfdefault-, and \ttdefault fonts are substituted partly or completely. Thus, courier.sty for instance exchanges only the typewriter font, whereas bookman.sty sets BOOKMAN as \rmdefault, AVANTGARDE as \sfdefault and COURIER as \ttdefault.
For further information see TOMAS ROKICKI's dvips manual [13].

4.10 Goodies—molweight and charge

During the process of sequence setting TEXshade sums up the molecular weight and charge of the aligned proteins. This data can be accessed by the following two commands.

\molweight{⟨seqref⟩}{⟨Da/kDa⟩}
\charge{⟨seqref⟩}{⟨i/o/N/C⟩}

The first parameter ⟨seqref⟩ selects the sequence. The second parameter in the \molweight command allows one to switch the units between Dalton (Da) and kilo-Dalton (kDa). The \charge command needs the second parameter for the correct consideration of the charged protein termini. Thus, 'i' refers to internal sequences, 'o' to the overall charge, 'N' to N-terminal sequence parts, and 'C' to the C-terminal end of a protein.

75

Example: Charge: \charge{1}{o}; Weight: \molweight{1}{Da}

5 The PostScript color selection scheme

POSTSCRIPT provides 64 standard colors. All these colors are predefined in the `color.sty`. Each color has a pictorial name such as `Bittersweet` and a distinct composition, e. g. 0% cyan + 75% magenta + 100% yellow + 24% black—the so-called CMYK scheme. TEXshade enhances this color scheme by gray scales in 5% steps. The following colors and grays can be used in TEXshade by simply declaring the name of the color in the respective command, e. g. `\consensuscolors`:

name	CMYK	name	CMYK
GreenYellow	0.15,0,0.69,0	Yellow	0,0,1,0
Goldenrod	0,0.10,0.84,0	Dandelion	0,0.29,0.84,0
Apricot	0,0.32,0.52,0	Peach	0,0.50,0.70,0
Melon	0,0.46,0.50,0	YellowOrange	0,0.42,1,0
Orange	0,0.61,0.87,0	BurntOrange	0,0.51,1,0
Bittersweet	0,0.75,1,0.24	RedOrange	0,0.77,0.87,0
Mahagony	0,0.85,0.87,0.35	Maroon	0,0.87,0.68,0.32
BrickRed	0,0.89,0.94,0.28	Red	0,1,1,0
OrangeRed	0,1,0.50,0	RubineRed	0,1,0.13,0
WildStrawberry	0,0.96,0.39,0	Salmon	0,0.53,0.38,0
CarnationPink	0,0.63,0,0	Magenta	0,1,0,0
VioletRed	0,0.81,0,0	Rhodamine	0,0.82,0,0
Mulberry	0.34,0.90,0,0.02	RedViolet	0.07,0.90,0,0.34
Fuchsia	0.47,0.91,0,0.08	Lavender	0,0.48,0,0
Thistle	0.12,0.59,0,0	Orchid	0.32,0.64,0,0
DarkOrchid	0.40,0.80,0.20,0	Purple	0.45,0.86,0,0
Plum	0.50,1,0,0	Violet	0.79,0.88,0,0
RoyalPurple	0.75,0.90,0,0	BlueViolet	0.86,0.91,0,0.04
Periwinkle	0.57,0.55,0,0	CadetBlue	0.62,0.57,0.23,0
CornflowerBlue	0.65,0.13,0,0	MidnightBlue	0.98,0.13,0,0.43
NavyBlue	0.94,0.54,0,0	RoyalBlue	1,0.50,0,0
Blue	1,1,0,0	Cerulean	0.94,0.11,0,0
Cyan	1,0,0,0	ProcessBlue	0.96,0,0,0
SkyBlue	0.62,0,0.12,0	Turquoise	0.85,0,0.20,0
TealBlue	0.86,0,0.34,0.02	Aquamarine	0.82,0,0.30,0
BlueGreen	0.85,0,0.33,0	Emerald	1,0,0.50,0
JungleGreen	0.99,0,0.52,0	SeaGreen	0.69,0,0.50,0
Green	1,0,1,0	ForestGreen	0.91,0,0.88,0.12
PineGreen	0.92,0,0.59,0.25	LimeGreen	0.50,0,1,0
YellowGreen	0.44,0,0.74,0	SpringGreen	0.26,0,0.76,0
OliveGreen	0.64,0,0.95,0.40	RawSienna	0,0.72,1,0.45
Sepia	0,0.83,1,0.70	Brown	0,0.81,1,0.60
Tan	0.14,0.42,0.56,0		
White (Gray0)	0,0,0,0	Black (Gray100)	0,0,0,1
Gray5	0,0,0,0.05	Gray10	0,0,0,0.10
Gray15	0,0,0,0.15	Gray20	0,0,0,0.20

name	RGB	name	RGB
Gray25	0,0,0,0.25	Gray30	0,0,0,0.30
LightGray	0,0,0,0.33	Gray35	0,0,0,0.35
Gray40	0,0,0,0.40	Gray45	0,0,0,0.45
Gray50	0,0,0,0.50	Gray	0,0,0,0.50
Gray55	0,0,0,0.55	Gray60	0,0,0,0.60
Gray65	0,0,0,0.65	DarkGray	0,0,0,0.66
Gray70	0,0,0,0.70	Gray75	0,0,0,0.75
Gray80	0,0,0,0.80	Gray85	0,0,0,0.85
Gray90	0,0,0,0.90	Gray95	0,0,0,0.95
LightGreenYellow	0.08,0,0.35,0	LightYellow	0,0,0.50,0
LightGoldenrod	0,0.05,0.42,0	LightDandelion	0,0.15,0.42,0
LightApricot	0,0.16,0.26,0	LightPeach	0,0.25,0.35,0
LightMelon	0,0.23,0.25,0	LightYellowOrange	0,0.21,0.50,0
LightOrange	0,0.31,0.44,0	LightBurntOrange	0,0.26,0.50,0
LightBittersweet	0,0.38,0.50,0.12	LightRedOrange	0,0.39,0.44,0
LightMahagony	0,0.43,0.44,0.18	LightMaroon	0,0.44,0.34,0.16
LightBrickRed	0,0.45,0.47,0.14	LightRed	0,0.50,0.50,0
LightOrangeRed	0,0.50,0.25,0	LightRubineRed	0,0.50,0.07,0
LightWildStrawberry	0,0.48,0.20,0	LightSalmon	0,0.27,0.19,0
LightCarnationPink	0,0.32,0,0	LightMagenta	0,0.50,0,0
LightVioletRed	0,0.40,0,0	LightRhodamine	0,0.41,0,0
LightMulberry	0.17,0.45,0,0.01	LightRedViolet	0.04,0.45,0,0.17
LightFuchsia	0.24,0.46,0,0.04	LightLavender	0,0.24,0,0
LightThistle	0.06,0.30,0,0	LightOrchid	0.16,0.32,0,0
LightDarkOrchid	0.20,0.40,0.10,0	LightPurple	0.23,0.43,0,0
LightPlum	0.25,0.50,0,0	LightViolet	0.40,0.44,0,0
LightRoyalPurple	0.38,0.45,0,0	LightBlueViolet	0.43,0.46,0,0.02
LightPeriwinkle	0.29,0.28,0,0	LightCadetBlue	0.31,0.29,0.12,0
LightCornflowerBlue	0.33,0.07,0,0	LightMidnightBlue	0.49,0.07,0,0.22
LightNavyBlue	0.47,0.27,0,0	LightRoyalBlue	0.50,0.25,0,0
LightBlue	0.50,0.50,0,0	LightCerulean	0.47,0.06,0,0
LightCyan	0.50,0,0,0	LightProcessBlue	0.48,0,0,0
LightSkyBlue	0.31,0,0.06,0	LightTurquoise	0.43,0,0.10,0
LightTealBlue	0.43,0,0.17,0.01	LightAquamarine	0.41,0,0.15,0
LightBlueGreen	0.43,0,0.17,0	LightEmerald	0.50,0,0.25,0
LightJungleGreen	0.50,0,0.26,0	LightSeaGreen	0.35,0,0.25,0
LightGreen	0.50,0,0.50,0	LightForestGreen	0.46,0,0.44,0.06
LightPineGreen	0.46,0,0.30,0.13	LightLimeGreen	0.25,0,0.50,0
LightYellowGreen	0.22,0,0.37,0	LightSpringGreen	0.13,0,0.38,0
LightOliveGreen	0.32,0,0.48,0.20	LightRawSienna	0,0.36,0.50,0.23
LightSepia	0,0.44,0.50,0.35	LightBrown	0,0.41,0.50,0.30
LightTan	0.07,0.21,0.28,0		

LightLight- and LightLightLight-versions were derived by dividing all values from Light-color definitions by 2 and 4, respectively.

name	RGB	name	RGB
BlueRed5	0.15,0.17,0.55	BlueRed10	0.20,0.23,0.57
BlueRed15	0.24,0.29,0.60	BlueRed20	0.33,0.35,0.64
BlueRed25	0.43,0.43,0.68	BlueRed30	0.52,0.52,0.73

BlueRed35	0.60,0.60,0.78	BlueRed40	0.70,0.70,0.84
BlueRed45	0.80,0.80,0.85	BlueRed50	0.86,0.82,0.82
BlueRed55	0.87,0.73,0.73	BlueRed60	0.89,0.64,0.64
BlueRed65	0.90,0.55,0.55	BlueRed70	0.91,0.47,0.46
BlueRed75	0.91,0.39,0.37	BlueRed80	0.90,0.33,0.28
BlueRed85	0.89,0.25,0.20	BlueRed90	0.88,0.23,0.14
BlueRed95	0.87,0.21,0.09	BlueRed100	0.87,0.16,0.04
GreenRed5	0,1,0	GreenRed10	0.05,0.95,0
GreenRed15	0.10,0.90,0	GreenRed20	0.15,0.85,0
GreenRed25	0.20,0.80,0	GreenRed30	0.25,0.75,0
GreenRed35	0.30,0.70,0	GreenRed40	0.35,0.65,0
GreenRed45	0.40,0.60,0	GreenRed50	0.45,0.55,0
GreenRed55	0.50,0.50,0	GreenRed60	0.55,0.45,0
GreenRed65	0.60,0.40,0	GreenRed70	0.65,0.35,0
GreenRed75	0.70,0.30,0	GreenRed80	0.75,0.25,0
GreenRed85	0.80,0.20,0	GreenRed90	0.85,0.15,0
GreenRed95	0.90,0.10,0	GreenRed100	0.95,0.05,0
ColdHot5	0,0.08,1	ColdHot10	0,0.29,1
ColdHot15	0,0.49,1	ColdHot20	0,0.70,1
ColdHot25	0,0.90,1	ColdHot30	0,1,0.87
ColdHot35	0,1,0.68	ColdHot40	0,1,0.46
ColdHot45	0,1,0.25	ColdHot50	0,1,0.04
ColdHot55	0.16,1,0	ColdHot60	0.35,1,0
ColdHot65	0.56,1,0	ColdHot70	0.79,1,0
ColdHot75	0.98,1,0	ColdHot80	1,0.82,0
ColdHot85	1,0.60,0	ColdHot90	1,0.40,0
ColdHot95	1,0.20,0	ColdHot100	0.91,0,0

and reverse definitions: `RedBlue`, `RedGreen`, `HotCold`.

Type the color names with the upper case letters exactly as described above. For the definition of new colors use one of the `color.sty` commands:

$\verb|\definecolor{|\langle name\rangle\verb|}{cmyk}{|\langle C,M,Y,K\rangle\verb|}|$

$\verb|\definecolor{|\langle name\rangle\verb|}{rgb}{|\langle R,G,B\rangle\verb|}|$

The $\langle name\rangle$ can be chosen freely, the values for the color composition must be in the range 0–1, i.e. 0–100% of the respective component ('C' – cyan, 'M' – magenta, 'Y' – yellow, 'K' – black; or 'R' – red, 'G' – green, 'Blue' – blue) separated by commas.

Examples:

`\definecolor{Salmon}{cmyk}{0,0.53,0.38,0}`

`\definecolor{ColdHot15}{rgb}{0,0.49,1}`

6 Listing of the `texshade` default settings

6.1 Standard definitions

The file `texshade.def` mirrors all commands which are carried out at the beginning of the `texshade` environment. Short comments are also included, thus, it is refered to this file for further information.

6.2 Colors used in the different shading modes

Color scheme *blues*:

res.color	shad.color	residues
Black	White	no match
Black	Magenta	similar
White	RoyalBlue	identical
Goldenrod	RoyalPurple	all match

Color scheme *greens*:

res.color	shad.color	residues
Black	White	no match
Black	GreenYellow	similar
White	PineGreen	identical
YellowOrange	OliveGreen	all match

Color scheme *reds*:

res.color	shad.color	residues
Black	White	no match
Black	YellowOrange	similar
White	BrickRed	identical
YellowGreen	Mahagony	all match

Color scheme *grays*:

res.color	shad.color	residues
Black	White	no match
Black	LightGray	similar
White	DarkGray	identical
White	Black	all match

Color scheme *black*:

res.color	shad.color	residues
Black	White	no match
Black	White	similar
White	Black	identical
White	Black	all match

Functional mode *charge*:

res.color	shad.color	residues
Black	White	no match
White	Red	acidic
White	Blue	basic

Functional mode *hydropathy*:

res.color	shad.color	residues
Black	White	no match
White	Red	acidic
White	Blue	basic
Black	Yellow	polar uncharged
White	Green	hydrophobic nonpolar

Functional mode *chemical*:

res.color	shad.color	residues
Black	White	no match
White	Red	acidic
White	Black	aliphatic
White	Gray	aliphatic (small)
White	Green	amide
White	Brown	aromatic
White	Blue	basic
Black	Magenta	hydroxyl
Black	Orange	imino
Black	Yellow	sulfur

Functional mode *rasmol*:

res.color	shad.color	residues
Black	White	no match
Red	White	Asp, Glu
Blue	White	Arg, Lys, His
MidnightBlue	White	Phe, Tyr, Trp
Gray	White	Ala, Gly
Yellow	White	Cys, Met
Orange	White	Ser, Thr
Cyan	White	Asn, Gln
Gree	White	Leu, Val, Ile
Apricot	White	Pro

Functional mode *structure*:

res.color	shad.color	residues
Black	White	no match
Black	Orange	external
Black	Yellow	ambivalent
White	Green	internal

Functional mode *standard area*:

res.color	shad.color	residues
Black	White	no match
Black	BrickRed	G
Black	Orange	A, S
Black	Yellow	C, P
Black	YellowGreen	T, D, V, N
White	PineGreen	I, E
Black	SkyBlue	L, Q, H, M
White	RoyalPurple	F, K
White	RedViolet	Y
White	Black	R, W

Functional mode *accessible area*:

res.color	shad.color	residues
Black	White	no match
Black	BrickRed	C
Black	Orange	I, V, G
Black	Yellow	F, L, M, A
Black	YellowGreen	W, S, T, H
White	PineGreen	P
Black	SkyBlue	Y, D, N
White	RoyalPurple	E, Q
White	RedViolet	R
White	Black	K

7 Quick Reference

The TEXshade logo

```
\TeXshade
```

The TEXshade environment (13 ff.)

```
\begin{texshade}[⟨parameterfile⟩] {⟨alignmentfile⟩}
    further TEXshade commands, if needed
\end{texshade}
```

Predefined shading modes

$\backslash$seqtype{⟨*type*⟩}   (P – peptide, N – nucleotide)   [37]

$\backslash$shadingmode[⟨*option*⟩]{⟨*mode*⟩}   [40]

⟨*mode*⟩	⟨*option*⟩	
identical	allmatchspecial/number	
similar	allmatchspecial/number	
T-Coffee	⟨*filename*⟩	
diverse	⟨*seqref*⟩	
functional	⟨*type*⟩	charge
		hydropathy
		structure
		chemical
		rasmol
		standard area
		accessible area

$\backslash$allmatchspecial[⟨*percentage*⟩]   [40]
$\backslash$shadeallresidues   [45]

Shading colors (40 ff.)

$\backslash$shadingcolors{⟨*scheme*⟩} (blues, reds, greens, grays, black)
$\backslash$nomatchresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}
$\backslash$similarresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}
$\backslash$conservedresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}
$\backslash$allmatchresidues{⟨*res.col.*⟩}{⟨*shad.col.*⟩}{⟨*case*⟩}{⟨*style*⟩}

```
\funcshadingstyle{⟨residue⟩}{⟨res.col.⟩}{⟨shad.color⟩}
                                 {⟨case⟩}{⟨style⟩} [45]
```

Residue grouping

```
\pepsims{⟨residue⟩}{⟨similars⟩}                                    [41]
\pepgroups{⟨group1⟩,⟨group2⟩,  ... ,  ⟨groupn⟩}                     [41]
\DNAsims{⟨residue⟩}{⟨similars⟩}                                    [41]
\DNAgroups{⟨group1⟩,⟨group2⟩,  ... ,  ⟨groupn⟩}                     [41]
```

Definition of new functional shading modes

```
\clearfuncgroups                                                  [45]
\funcgroup{⟨descr⟩}{⟨residues⟩}{⟨res.col.⟩}{⟨shad.col.⟩}
               {⟨case⟩}{⟨style⟩}                                   [46]
```

Appearance of the consensus line

```
\threshold[⟨percentage⟩]{⟨percentage⟩}                            [47]
\constosingleseq{⟨seqref⟩}                                        [47]
\showconsensus[⟨color/scale⟩[,⟨color/scale⟩]]{⟨top/bot.⟩}         [47]
\exportconsensus[⟨filename⟩]{⟨seqref⟩}                             [47]
\hideconsensus                                                    [47]
\nameconsensus{⟨name⟩}                                            [47]
\defconsensus{⟨symbol1⟩}{⟨symbol2⟩}{⟨symbol3⟩}                     [47]

\consensuscolors{⟨res.col.1⟩}{⟨shad.col.1⟩}
              {⟨res.col.2⟩}{⟨shad.col.2⟩}
              {⟨res.col.3⟩}{⟨shad.col.3⟩}                          [48]
```

Sequence logos

```
\showsequencelogo[⟨colorset⟩]{⟨top/bottom⟩}                        [49]
\hidesequencelogo                                                 [49]
\clearlogocolors[⟨color⟩]                                         [50]
\logocolor{⟨residues⟩}{⟨color⟩}                                   [50]
\showlogoscale[⟨color⟩]{⟨left/right/leftright⟩}                   [50]
\hidelogoscale                                                    [50]
\logostretch{⟨factor⟩}                                            [50]
\namesequencelogo{⟨name⟩}                                         [51]
\dofrequencycorrection                                            [50]
```

`\undofrequencycorrection`

Subfamily logos

Appearance of the sequence lines

`\gapchar{⟨symbol⟩}`   (incl. rule)   [54]
`\gapcolors{⟨symbol color⟩}{⟨background color⟩}`   [54]
`\showleadinggaps`   [55]
`\hideleadinggaps`   [55]
`\fingerprint{⟨res. per line⟩}`   [57]

Hiding, killing, separating and ordering

`\hideseq{⟨seq1⟩,⟨seq2⟩,...,⟨seq n⟩}`   [55]
`\hideseqs`   [55]
`\showseqs`   [55]
`\killseq{⟨seq1⟩,⟨seq2⟩,...,⟨seq n⟩}`   [55]
`\donotshade{⟨seq1⟩,⟨seq2⟩,...,⟨seq n⟩}`   [55]
`\separationline{⟨seqref⟩}`   [56]
`\smallsep`   [56]
`\medsep`   [56]
`\bigsep`   [56]
`\vsepspace{⟨length⟩}`   [56]
`\orderseqs{⟨seq1⟩,⟨seq2⟩,...,⟨seq n⟩}`   [56]

Residues per line and further settings

`\residuesperline{⟨number⟩}`   [56]
`\residuesperline*{⟨number⟩}`   [56]
`\charstretch{⟨factor⟩}`   [56]
`\linestretch{⟨factor⟩}`   [56]
`\numberingwidth{⟨n digits⟩}`   [57]
`\smallblockskip`   [57]
`\medblockskip`   [57]
`\bigblockskip`   [57]
`\noblockskip`   [57]
`\vblockspace{⟨length⟩}`   [57]
`\flexblockspace`   [57]
`\fixblockspace`   [57]
`\alignment{⟨left/center/right⟩}`   [57]

Individual shading and labeling of sequence stretches

`\shaderegion{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,`
`          ...,⟨start n⟩..⟨stop n⟩}{⟨res.col.⟩}{⟨shad.col.⟩}` [58]

\shadeblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}{⟨res.col.⟩}{⟨shad.col.⟩}
\emphregion{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}
\emphblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}
\emphdefault{⟨style⟩}
\tintregion{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}
\tintblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}
\tintdefault{⟨effect⟩}   weak, normal, strong
\frameblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}{⟨color⟩[⟨length⟩]}
\hideblock{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,⟨start2⟩..⟨stop2⟩,
    ...,⟨start n⟩..⟨stop n⟩}{⟨label⟩[⟨color⟩]}{⟨text⟩[⟨color⟩]}
\feature{⟨position⟩}{⟨seqref⟩}{⟨start1⟩..⟨stop1⟩,
⟨start2⟩..⟨stop2⟩,...,⟨start n⟩..⟨stop n⟩}{⟨labelstyle⟩}{⟨text⟩}

    {⟨labelstyle⟩} = {brace[⟨color⟩]}
                 = {fill:⟨symbol⟩[⟨textcolor⟩]}
                 = {restriction[⟨color⟩]}
                 = {helix[⟨helixcolor⟩]}
                 = {box[⟨framecolor,boxcolor⟩][⟨length⟩]:
                              ⟨text⟩[⟨textcolor⟩]}
                 = arrows and bars (-=<',|o)(-=)(-=>',|o)
                 = {translate[⟨color⟩]}
                 = {bar[⟨min⟩,⟨max⟩]:
                     ⟨properties/file/data⟩[⟨color(,bgcolor)⟩]}
                 = {color[⟨min⟩,⟨max⟩]:
                     ⟨properties/file/data⟩[⟨scale⟩]}
                     ⟨properties⟩: hydrophobicity, charge,
                            molweight, conservation

\ttopspace{⟨length⟩}
\topspace{⟨length⟩}
\bottomspace{⟨length⟩}
\bbottomspace{⟨length⟩}
\featurerule{⟨length⟩}
\bargraphstretch{⟨factor⟩}

`\colorscalestretch{`⟨*factor*⟩`}`
`\codon{`⟨*amino acid*⟩`}{`⟨*triplet1,..., triplet n*⟩`}`
`\geneticcode{`⟨*filename*⟩`}`
`\backtranslabel[`⟨*size*⟩`]{`⟨*style*⟩`}`
`\backtranstext[`⟨*size*⟩`]{`⟨*style*⟩`}`

```
{⟨style⟩} = {horizontal}
         = {alternating}
         = {zigzag}
         = {oblique}
         = {vertical}
```

`\showfeaturename{`⟨*ttttop...bbbbottom*⟩`}{`⟨*name*⟩`}`
`\showfeaturestylename{`⟨*ttttop...bbbbottom*⟩`}{`⟨*name*⟩`}`
`\hidefeaturename{`⟨*ttttop...bbbbottom*⟩`}`
`\hidefeaturestylename{`⟨*ttttop...bbbbottom*⟩`}`
`\hidefeaturenames`
`\hidefeaturestylenames`
`\featurenamescolor{`⟨*color*⟩`}`
`\featurestylenamescolor{`⟨*color*⟩`}`
`\featurenamecolor{`⟨*ttttop...bbbbottom*⟩`}{`⟨*color*⟩`}`
`\featurestylenamecolor{`⟨*ttttop...bbbbottom*⟩`}{`⟨*color*⟩`}`

Including secondary structure information

`\includeDSSP[make new]{`⟨*seqref*⟩`}{`⟨*filename*⟩`}`
`\includeSTRIDE[make new]{`⟨*seqref*⟩`}{`⟨*filename*⟩`}`
`\includePHDsec[make new]{`⟨*seqref*⟩`}{`⟨*filename*⟩`}`
`\includePHDtopo[make new]{`⟨*seqref*⟩`}{`⟨*filename*⟩`}`
`\includeHMMTOP[make new]{`⟨*seqref*⟩`[`⟨*seqref*⟩`]}{`⟨*filename*⟩`}`
`\showonDSSP{`⟨*structures*⟩`}`
`\showonSTRIDE{`⟨*structures*⟩`}`
`\showonPHDsec{`⟨*structures*⟩`}`
`\showonPHDtopo{`⟨*structures*⟩`}`
`\showonHMMTOP{`⟨*structures*⟩`}`
`\hideonDSSP{`⟨*structures*⟩`}`
`\hideonSTRIDE{`⟨*structures*⟩`}`
`\hideonPHDsec{`⟨*structures*⟩`}`
`\hideonPHDtopo{`⟨*structures*⟩`}`
`\hideonHMMTOP{`⟨*structures*⟩`}`
`\appearance{`⟨*type*⟩`}{`⟨*position*⟩`}{`⟨*labelstyle*⟩`}{`⟨*text*⟩`}`

Displaying and building legends

Adding captions to the alignment

Font handling

Corresponding sets are provided for the numbering (`\numberingrm` etc.), featurestyles (`featurestylesrm` etc.), names (`\namesrm` etc.), featurenames (`\featurenamesrm` etc.), featurestylenames (`\featurestylenames` etc.), residues (`\residuesrm` etc.), hideblock labels (`hideblockrm` etc.), and legend texts (`legendrm` etc.).

Goodies—molweight and charge

`\molweight{`⟨*seqref*⟩`}{`⟨*Da/kDa*⟩`}`   [75]
`\charge{`⟨*seqref*⟩`}{`⟨*i/o/N/C*⟩`}`   [75]

8   References

[1] CARLISLE, D. The Standard LaTeX 'Graphics Bundle', `color.sty`.

[2] KARLIN, S.; GHANDOUR, G. (1985) Multiple-alphabet amino acid sequence comparisons of the immunoglobulin κ-chain constant domain. *Proc. Natl. Acad. Sci. USA*: **82**, 8597–8601.

[3] KYTE, J.; DOOLITTLE, R. F. (1982) A simple method for displaying the hydropathic character of a protein. *J. Mol. Biol.*: **157**, 105–132.

[4] ROSE, G. D.; GESELOWITZ, A. R.; LESSER, G. J.; LEE, R. H.; ZEHFUS, M. H. (1985) Hydrophobicity of amino acid residues in globular proteins. *Science*: **229**, 835–838.

[5] LESSER, G. J.; ROSE, G. D. (1990) Hydrophobicity of amino acid subgroups in proteins. *Proteins: structure, function and genetics*: **8**, 6–13.

[6] FRÖHLICH, K.-U. (1994) Sequence similarity presenter: a tool for the graphic display of similarities of long sequences for use in presentations. *Comput. Applic. Biosci.*: **10**, 179–183.

[7] SCHNEIDER, T.D.; STEPHENS, R.M. (1990) Sequence logos: a new way to display consensus *Nucleic Acid Res.*: **18**, 6097–6100.

[8] DeLano Scientific LLC 'www.pymol.org'

[9] KABSCH, W.; SANDER, C. (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*: **22**, 2577–2637.

[10] FRISHMAN, D.; ARGOS, P. (1995) Knowledge-based protein secondary structure assignment. *Proteins: structure, function and genetics*: **23**, 566–579.

[11] ROST, B.; SANDER, C. (1994) Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins: structure, function and genetics*: **19**, 55–72.

[12] TUSNADY, G.E.; SIMON, I. (2001) The HMMTOP transmembrane topology prediction server. *Bioinformatics*: **17**, 849-850.

[13] ROKICKI, T. DVIPS: A TeX driver.

[14] BEITZ, E. (2006) Subfamily logos: visualization of sequence deviations at alignment positions with high information content. *BMC Bioinformatics*: **7**:313.