The apxproof package

Pierre Senellart
pierre@senellart.com
http://github.com/PierreSenellart/apxproof

2025/07/29 v1.2.5-dev

Abstract

This package makes it easier to write articles where proofs and other material are deferred to the appendix. The appendix material is written in the LaTeX code along with the main text which it naturally complements, and it is automatically deferred. The package can automatically send proofs to the appendix, can repeat in the appendix the theorem environments stated in the main text, can section the appendix automatically based on the sectioning of the main text, and supports a separate bibliography for the appendix material.

1 Usage

The apxproof package is intended to simplify the writing of articles where some of the content needs to be deferred to an appendix. This is in particular useful for the submission of scientific articles to conferences or journals that limit the number of pages in the main text but allow an extra appendix, where proofs of theorems and other material can be added.

1.1 Basics

To use apxproof, first load it in the header of your document:

```
\usepackage{apxproof}
```

On its own, this does not do anything and should not change the appearance of your document. To add an appendix with some material from your document, use the `toappendix` environment:

`toappendix` (*env.*)

```
\begin{toappendix}
  ...
\end{toappendix}
```

The content will appear at the end of your document, in an automatically generated section that refers to the current section in the main text.

Example 1. *Throughout this documentation, all examples produce content deferred to the appendix, at the very end of this document.*

```
\begin{toappendix}
```

```
This content is in the appendix.
\end{toappendix}
```

When the content to put in appendix is an entire section, make sure that `\section` is the very first command that appears within the `toappendix` environment. It will disable the automatic production of a section heading.

1.2 Repeated Theorems and Proofs

In some scientific papers that include proofs, it is common to defer proofs to the appendix. This can easily be achieved using the `appendixproof` environment:

`appendixproof` (*env.*)

```
\begin{appendixproof}
  ...
\end{appendixproof}
```

This behaves like the `toappendix` environment, except that a proof environment is generated.

Example 2. *We now send a proof to the appendix:*

```
\begin{appendixproof}
This proof is in the appendix.
\end{appendixproof}
```

When deferring proofs to the appendix, an annoying problem is that the statement of the theorem remains in the main text; it is hard to read a proof that is far away from the statement it proves. `apxproof` solves this issue by allowing statements of theorems to be *repeated*: once in the main text, and once in the appendix before the proof of the statement. To use this feature, you can define a new *repeated theorem* environment using the `\newtheoremrep` command:

`\newtheoremrep`

```
\newtheoremrep{⟨name⟩}[⟨counter⟩]{⟨title⟩}[⟨countersec⟩]
```

Usage is exactly the same as that of AMS LaTeX's `\newtheorem` macro:

- ⟨*name*⟩ (e.g., `theorem`) is the name of an environment that is created for this kind of theorem;
- ⟨*counter*⟩ (e.g., `definition`) is an optional counter describing from which kind of environment the numbering of these environments should be inherited;
- ⟨*title*⟩ (e.g., `Theorem`) is the title that will be used to display this theorem environment;
- ⟨*countersec*⟩ (e.g., `section`) is an optional counter of a sectioning command indicating that counters for this theorem should be prefixed by this counter (and reset at each occurrence of the sectioning command).

⟨*counter*⟩ and ⟨*countersec*⟩ should not be used together. What differs from `\newtheorem` is that, when the following is written:

```
\newtheoremrep{foobar}{Foobar}
```

then *two* environments are defined: the `foobar` environment, which behaves as if `\newtheorem` had been used, and the `foobarrep` environment, which results in the statement of this environment being repeated in the appendix.

`proof` (*env.*)

One interesting feature of apxproof is that in most situations, there is no need to use the `appendixproof` environment. Indeed, the `proof` environment is redefined by apxproof to automatically put the proof either in the main text (if it follows a regular theorem) or in the appendix (if it follows a repeated theorem).

Example 3. *Assume we have first defined a repeated theorem environment* `foobar` *as above. We can now use this theorem environment, first for a regular theorem in the main text, then for a theorem repeated in the main text and in the appendix:*

```
\begin{foobar}
This foobar is a regular one, in the main text.
\end{foobar}
\begin{proof}
This is the proof of the regular foobar.
\end{proof}
```

We obtain:

Foobar 1. *This foobar is a regular one, in the main text.*

Proof. This is the proof of the regular foobar. □

Now, if we use a repeated theorem:

```
\begin{foobarrep}
This foobar is repeated in the appendix.
\end{foobarrep}
\begin{proof}
This is the proof of the repeated foobar.
\end{proof}
```

We now obtain:

Foobar 2. *This foobar is repeated in the appendix.*

Note that, since `hyperref` *is loaded, there are hyperlinks created between the statements of the theorems in the main text and in the appendix.*

When the proof is deferred to the appendix, it is common practice to add a proof sketch in the main text. apxproof defines a simple `proofsketch` environment for this purpose:

`proofsketch` (*env.*)

```
\begin{proofsketch}
  ...
\end{proofsketch}
```

`inlineproof` (*env.*)

The proof sketch is typeset similarly to a proof, but is always in the main text. Similarly, an `inlineproof` environment is provided so as to be able to have both a proof in the appendix (using the regular `proof` environment, or alternatively the `appendixproof` environment) and a different proof in the main text (using the `inlineproof` environment).

Example 4. *Here are simple examples of proof sketches and inline proofs:*

```
\begin{proofsketch}
This is a proof sketch.
\end{proofsketch}
```

Proof sketch. This is a proof sketch. □

```
\begin{inlineproof}
This is an inline proof.
\end{inlineproof}
```

Proof. This is an inline proof. □

1.3 Bibliography

By default, apxproof automatically adds a bibliography in the appendix with only the references cited in the appendix material. This allows for a clean separation of references used solely in the main text, and those used in the appendix.

Example 5. *Assume we have citations both in the main text and in the appendix.*

```
This is a citation in the main text~\cite{lamport86}.
\begin{toappendix}
This is a citation in the appendix~\cite{proofsAreHard}.
\end{toappendix}
```

This is a citation in the main text [1].

The bibliography in the appendix can use a different style and heading than the bibliography in the main text (and, by default, it does). See Section 1.5 for how to configure the appearance of that bibliography.

option `bibliography`

In order to use a single appendix for the main text and the bibliography, one can specify the value `common` to the `bibliography` option when loading the package. (By default this option is set to `separate`.)

1.4 Mode

option `appendix`

An optional ⟨*mode*⟩ can be specified when loading the package:

```
\usepackage[appendix=⟨mode⟩]{apxproof}
```

⟨*mode*⟩ can take one of the following three values:

`append` This is the default. Appendix material gathered by apxproof is appended to the main text.

`inline` In this mode, apxproof simply inlines the content along with the main text.

`strip` This mode functions similarly to `append` except that the appendix is not appended at the end of the document. All appendix material is therefore removed.

1.5 Customization

`apxproof` provides a few macros that can be redefined (using `\renewcommand`) to customize the appearance of the appendix:

`\mainbodyrepeatedtheorem` `\mainbodyrepeatedtheorem` is a macro that is executed at the beginning of the body of every repeated theorem. This can be used to notify the reader that the theorem is repeated in appendix in some way, e.g., with a margin note.

`\appendixsectionformat` `\appendixsectionformat{`⟨***number***⟩`}{`⟨***title***⟩`}` is a macro that indicates how to format the section titles in the Appendix, given the number and title of the section in the main text. By default, they appear as "Proofs for Section ⟨*number*⟩ (⟨*title*⟩)".

`\appendixrefname` `\appendixrefname` contains the heading that is displayed before the bibliography. By default, this is "References for the Appendix". (Note that this command is also defined and used by the `memoir` document class.)

`\appendixbibliographystyle` `\appendixbibliographystyle` contains the `.bst` bibliography style that is used in the bibliography in appendix. By default, this is `alpha`.

`\appendixbibliographyprelim` `\appendixbibliographyprelim` contains arbitrary code that is executed just before the production of the bibliography in appendix, which can be used to configure the way it is displayed.

`\appendixprelim` `\appendixprelim` contains arbitrary code that is executed just before the production of the appendix, which can be used to configure the way it is displayed. By default, this command contains `\clearpage\onecolumn` (the appendix is typeset on a new page in single-column mode) but redefining this option allows changing this behavior.

option `repeqn` Another customization capability concerns *numbered equations* that are present within repeated theorems. An optional `repeqn` option can be specified when loading the package, which controls whether equation numbers should be as in the main text (by setting this option to `same`, the default) or independently numbered (by setting this option to `independent`). In the latter case, whenever a referenceable counter is set with `\label{`⟨*counter*⟩`}`, `\ref{`⟨*counter*⟩`}` references the counter in the main text, while `\ref{`⟨*counter*⟩`-apx}` references the counter in the appendix (except in `inline` mode, where both have the same effect).

option `forwardlinking` Another customization option concerns hyperlinking. Usually, when `hyperref` is loaded, `foobarrep` environments in the main text have their number link to their repetition in the appendix. To suppress this behavior and have `foobarrep` environments treated as if `hyperref` were not loaded, one can specify the value `no` to the `forwardlinking` option when loading the package. (By default this option is set to `yes`.)

1.6 Advanced Features

We now describe a few advanced macros and environments, the usage of which is limited to special cases:

`nestedproof` (*env.*) `nestedproof` is an environment that can be used within a `proof` environment deferred in the appendix; this is required because, for technical reasons, no `proof` environment can be nested within a deferred `proof` environment.

\noproofinappendix `\noproofinappendix` can be used inside repeated theorems that are not followed by a `proof` or `appendixproof` environment; the point is to ensure that a further `proof` environment cannot be mistakenly understood as a proof of the repeated theorem. It should not be needed in most situations as apxproof tries figuring out when a proof follows a repeated theorem automatically, but may occasionally be needed in complex scenarios.

\nosectionappendix `\nosectionappendix` is to be used inside a section that *does* contain appendix material, but for which a section in the appendix should not be created. This should be rarely needed. When this command is present, appendix material is appended to the end of the previously created section.

2 Supported Document Classes

Because apxproof modifies sectioning commands, bibliographies, and proofs, it may not work straight away with arbitrary document classes. It has currently been tested with and is supported for the following document classes:

- LaTeX standard document classes (e.g., `article.cls`)
- KOMA-Script (e.g., `scrartcl.cls`, `scrbook.cls`)
- `memoir.cls`
- ACM SIG Proceedings (e.g., `sig-alternate.cls`, `acmart.cls`)
- Springer's Lecture Notes in Computer Science (e.g., `llncs.cls`)
- Schloß Dagstuhl's Leibniz International Proceedings in Informatics (e.g., `lipics.cls`, `lipcs-v2016.cls`)

Other classes may work out of the box. Adding support for specific classes is possible and can be requested from the author of this package.

3 Known Issues and Limitations

We report here some issues we are currently aware of:

- When using hyperref, the appendix in the bibliography is not hyperlinked. This is to avoid possible issues with multiply defined bibliography entries.
- `appendixproof`, `proof`, `toappendix` environments cannot be nested. This is a limitation of the fancyvrb package that apxproof relies on. Note the existence of the `nestedproof` environment for nested proofs.
- apxproof poorly interacts with SyncTeX: identifying which source line has produced which box does not work for appendix content managed by apxproof or repeated theorems. No obvious fix is known, though this issue will be investigated in the long term.
- Unless the `bibliography` option is set to `common`, the bibunits package is used to generate a second bibliography. This means any package, such as biblatex, that is incompatible with bibunits will not be compatible with apxproof unless `bibliography` is set to `common`.

Issues not listed here should be reported to the author.

4 License

5 Contact

- `https://github.com/PierreSenellart/apxproof`
- Pierre Senellart <pierre@senellart.com>

Bug reports and feature requests should preferably be submitted through the *Issues* feature of GitHub.

6 Acknowledgments

Thanks to Antoine Amarilli for feedback and proofreading. Thanks to K. D. Bauer for the implementation of the forward-linking mechanism, and for various bugfixes. Thanks to Leonid Kostrykin for an initial implementation of the `forwardlinking` option.

7 Implementation

We now describe the entire code of the package, in a literate programming fashion. Throughout the package, we use the `axp@` prefix to identify local macros and environment names, which are not meant to be used by the final user.

7.1 Dependencies

We first load a few package dependencies:

- environ to easily define the repeated theorem environments.

```
\RequirePackage{environ}
```

- etoolbox to define simple toggles.

```
\RequirePackage{etoolbox}
```

- fancyvrb for the bulk of the work of exporting appendix material in an auxiliary file.

```
\RequirePackage{fancyvrb}
```

- ifthen for easier comparison of character strings.

```
\RequirePackage{ifthen}
```

- kvoptions to manage options passed to the package.

```
\RequirePackage{kvoptions}
```

- catchfile to be able to check the content of files \input within appendix content.

```
\RequirePackage{catchfile}
```

- amsthm for its \newteorem macro. Some document classes (e.g., lipics) preload amsthm: this is fine, \RequirePackage{amsthm} will simply have no effect. On the other hand, some other document classes (e.g., llncs or sig-alternate) define a proof environment that conflicts with amsthm, so we have to undefine this environment before loading amsthm. In that case, we reestablish the existing proof environments, in case they had been customized (e.g., sig-alternate)

```
\@ifpackageloaded{amsthm}{
  }{
    \let\apx@oldamsthmproof\proof
    \let\apx@oldamsthmendproof\endproof
    \let\proof\undefined
    \let\endproof\undefined
  }
\RequirePackage{amsthm}
\ifdefined\apx@oldamsthmproof
  \let\proof\apx@oldamsthmproof
  \let\endproof\apx@oldamsthmendproof
\fi
```

7.2 Option Processing

Many names throughout the package use an arobase (@) to avoid name conflict with user-defined names. To simplify the compilation of the documentation, we simply make it a regular character in all the rest.

```
\makeatletter
```

We setup the processing of options using keyval facilities.

```
\SetupKeyvalOptions{
  family=axp,
  prefix=axp@
}
```

We declare the following options:

- appendix, with a default value of append (other possible values: strip, inline);
- bibliography, with a default value of separate (other possible value: common);
- repeqn, with a default value of same (other possible value: independent).

\axp@appendix

```
\DeclareStringOption[append]{appendix}
```

\axp@bibliography

```
\DeclareStringOption[separate]{bibliography}
```

\axp@repeqn

```
\DeclareStringOption[same]{repeqn}
```

\axp@forwardlinking

```
\DeclareStringOption[yes]{forwardlinking}
```

```
\ProcessLocalKeyvalOptions*
```

We check that the value of the options are valid, and add a message to the compilation log.

```
\ifthenelse{\equal{\axp@appendix}{append}}{
  \message{apxproof: Appendix material appended to the document}
}{\ifthenelse{\equal{\axp@appendix}{strip}}{
  \message{apxproof: Appendix material stripped}
}{\ifthenelse{\equal{\axp@appendix}{inline}}{
  \message{apxproof: Appendix material inlined within the document}
}{
  \errmessage{Error: unsupported option appendix=\axp@appendix\ for
  package apxproof}
}}}
\ifthenelse{\equal{\axp@bibliography}{separate}}{
```

The external bibunits package is used to add a second bibliography for the appendix material.

```
  \RequirePackage{bibunits}
  \message{apxproof: Separate bibliography for appendix material}
}{\ifthenelse{\equal{\axp@bibliography}{common}}{
  \message{apxproof: Common bibliography for appendix and main text}
}{
  \errmessage{Error: unsupported option bibliography=\axp@bibliography\ for
  package apxproof}
}}
\ifthenelse{\equal{\axp@repeqn}{same}}{
  \message{apxproof: Repeated equations keep the same numbering}
}{\ifthenelse{\equal{\axp@repeqn}{independent}}{
  \message{apxproof: Repeated equations are independently numbered}
}{
  \errmessage{Error: unsupported option repeqn=\axp@repeqn\ for
  package apxproof}
}}
```

\axp@forward@suppress

```
\newbool{axp@forward@suppress}
\ifthenelse{\equal{\axp@forwardlinking}{yes}}{
}{\ifthenelse{\equal{\axp@forwardlinking}{no}}{
  \message{apxproof: Disable forward linking}
  \global\booltrue{axp@forward@suppress}%
}{
  \errmessage{Error: unsupported option forwardlinking=\axp@repeqn\ for
  package apxproof}
}}
```

7.3 Macros Common to All Compilation Modes

xp@newtheoremrep@definetheorem

Common to all compilation modes, we define `\axp@newtheoremrep@definetheorem`. When called with first argument `foobar`, we first undefine the existing `foobar` environment (and its counter) if it has already been defined (e.g., by the document class), then invoke `\axp@newtheorem` for the regular version of the theorem `foobar`, saving and restoring any existing theorem counter unless the `\newtheoremdep` redefines the base counter.

```
\def\axp@newtheoremrep@definetheorem#1#2#3#4{%
  \expandafter\let\csname #1\endcsname\undefined
  \expandafter\let\csname the#1\endcsname\undefined
  \ifcsname c@#1\endcsname
    \expandafter\expandafter\expandafter\let\expandafter\expandafter
      \csname c@axp@#1\endcsname\csname c@#1\endcsname
    \expandafter\let\csname c@#1\endcsname\undefined
  \fi
  \axp@newtheorem{#1}{#2}{#3}{#4}%
  \ifcsname c@axp@#1\endcsname
    \ifx\relax#2\relax
      \expandafter\expandafter\expandafter\let\expandafter\expandafter
        \csname c@#1\endcsname\csname c@axp@#1\endcsname
    \else
    \fi
  \fi
}
```

\axp@newtheorem
\@axp@newtheorem
\@@axp@newtheorem

We introduce an intermediate `\axp@newtheorem` command to define a new theorem, differently depending on whether there is a section counter or not. This will be useful, in particular to allow changing this definition depending on the document class. This command uses two intermediary commands, `\@axp@newtheorem` and `\@@axp@newtheorem`, for the non-starred and starred versions.

```
\def\axp@newtheorem{\@ifstar\@@axp@newtheorem\@axp@newtheorem}
\def\@axp@newtheorem#1#2#3#4{%
  \ifx\relax#4\relax
    \newtheorem{#1}[#2]{#3}%
  \else
    \newtheorem{#1}{#3}[#4]%
  \fi
}
\def\@@axp@newtheorem#1#2{%
  \newtheorem*{#1}{#2}%
}
```

\newtheoremrep
\axp@newtheoremreptmp

We define the high-level `\newtheoremrep` to have the same syntax as amsthm's `\newtheorem`. For this purpose, we need a little trick to deal with the second and fourth optional arguments, which is what `\@oparg` and `\axp@newtheoremreptmp` are used for. `\axp@newtheoremrep` is defined differently depending on the compilation mode.

```
\newcommand\newtheoremrep[1]{%
  \@oparg{\axp@newtheoremreptmp{#1}}[]%
}
\def\axp@newtheoremreptmp#1[#2]#3{%
  \@oparg{\axp@newtheoremrep{#1}[#2]{#3}}[]%
}
```

`proofsketch` (*env.*)

Simple `proofsketch` environment.

```
  \newenvironment{proofsketch}{\begin{axp@oldproof}[Proof sketch]}{\end{axp@oldproof}}
```

`\mainbodyrepeatedtheorem`
`\appendixrefname`
`\appendixbibliographystyle`
`\appendixbibliographyprelim`
`\appendixprelim`
`\appendixsectionformat`

We provide sensible defaults for these user-customizable macros. Even though they are not all useful in all modes, we define them for all modes so that a `\renewcommand` works in all cases.

```
\newcommand{\mainbodyrepeatedtheorem}{}
\providecommand{\appendixrefname}{References for the Appendix}
\newcommand{\appendixbibliographystyle}{alpha}
\newcommand{\appendixbibliographyprelim}{}
\newcommand{\appendixprelim}{\clearpage\onecolumn}
\newcommand{\appendixsectionformat}[2]{Proofs for Section~#1\ (#2)}
```

`axp@oldproof` (*env.*)

We save the definition of the existing `proof` environment.

```
  \let\axp@oldproof\proof
  \let\endaxp@oldproof\endproof
```

We define a utility macro that will be used to properly set the `\label` command (and its `amsmath` counterpart, `\label@in@display`) for equations within repeated theorems, depending on the compilation mode.

`\axp@redefinelabels`

```
  \newcommand{\axp@redefinelabels}{%
    \providecommand\label@in@display{}%
    \ifthenelse{\equal{\axp@appendix}{inline}}{%
      \let\axp@oldlabel\label
      \let\axp@oldlabel@in@display\label@in@display
      \renewcommand\label[1]{%
        \axp@oldlabel{##1}%
        \axp@oldlabel{##1-apx}%
      }%
      \renewcommand\label@in@display[1]{%
        \axp@oldlabel@in@display{##1}%
        \axp@oldlabel{##1-apx}%
      }%
    }{%
      \let\axp@oldlabel\label
      \let\axp@oldlabel@in@display\label@in@display
      \renewcommand\label[1]{\axp@oldlabel{##1-apx}}%
      \renewcommand\label@in@display[1]{\axp@oldlabel@in@display{##1-apx}}%
    }%
  }
```

7.3.1 Class-Specific Behavior

Finally, some class-specific behavior common to all compilation modes.

llncs and other Springer document classes

```
  \ifdefined\spnewtheorem
```

\@axp@newtheorem
\@@axp@newtheorem

It is necessary to use `\spnewtheorem` instead of `\newtheorem` in Springer document classes to obtain standard formatting.

```
\def\@axp@newtheorem#1#2#3#4{%
  \ifx\relax#4\relax
    \ifx\relax#2\relax
      \spnewtheorem{#1}{#3}{\bfseries}{\itshape}%
    \else
      \spnewtheorem{#1}[#2]{#3}{\bfseries}{\itshape}%
    \fi
  \else
    \spnewtheorem{#1}{#3}[#4]{\bfseries}{\itshape}%
  \fi
}
\def\@@axp@newtheorem#1#2{%
  \spnewtheorem*{#1}{#2}{\upshape\bfseries}{\itshape}%
}
```

proofsketch

We redefine the `proofsketch` environment, which is used differently in the base class.

```
\renewenvironment{proofsketch}{\begin{axp@oldproof}[sketch]}{\end{axp@oldproof}}
```

We have to redefine the macro `\@thmcountersep` for proper sectioned counters.

```
\def\@thmcountersep{.}
  \fi
```

acmart Some versions of acmart define a `\ACM@origsection` macro and verify that `\section` has the same content. We need to update `\ACM@origsection` accordingly.

```
\ifdefined\ACM@origsection
  \let\ACM@origsection\section
\fi
```

7.4 Inline Compilation Mode

```
\ifthenelse{\equal{\axp@appendix}{inline}}{
```

\axp@newtheoremrep

In inline mode, `\axp@newtheoremrep` uses `\axp@newtheoremrep@definetheorem` to define the regular theorem environment and creates a repeated theorem environment that behaves exactly as the regular theorem environment, while calling `\axp@redefinelabels` to make sure that `-axp` variants of equation counters are defined.

```
  \def\axp@newtheoremrep#1[#2]#3[#4]{%
    \axp@newtheoremrep@definetheorem{#1}{#2}{#3}{#4}%
    \NewEnviron{#1rep}[1][]{%
      \ifx\relax##1\relax
        \begin{#1}\axp@redefinelabels\BODY\end{#1}%
      \else
        \begin{#1}[##1]\axp@redefinelabels\BODY\end{#1}%
      \fi
    }
  }
```

inlineproof (*env.*)
nestedproof (*env.*)
appendixproof (*env.*)

In inline mode, these environments behave like the regular proof environment.

```
\let\inlineproof\proof
\let\endinlineproof\endproof
\let\nestedproof\proof
\let\endnestedproof\endproof
\let\appendixproof\proof
\let\endappendixproof\endproof
```

toappendix (*env.*)
\noproofinappendix
\nosectionappendix

In inline mode, this environment and these macros are no-ops.

```
\newenvironment{toappendix}{}{}
\let\noproofinappendix\relax
\let\nosectionappendix\relax
```

```
}
```

7.5 Append or Strip Compilation Modes

```
{
```

We now deal with the case where apxproof really does something useful: either append the appendix material to the document, or strip it entirely.

7.5.1 Auxiliary File for the Appendix

\axp@proofsfile

We open a new auxiliary file, with extension .axp, where the appendix material will be dumped.

```
\AtBeginDocument{
  \newwrite\axp@proofsfile
  \immediate\openout\axp@proofsfile=\jobname.axp
}
```

proof (*env.*)
\section

At the beginning of this file, we make @ a regular character (since it will be used in several places for internal names) and reestablish the original definition of the proof environment and the \section macro.

```
\AtBeginDocument{
  \immediate\write\axp@proofsfile{%
    \noexpand\makeatletter
    \noexpand\let\noexpand\proof\noexpand\axp@oldproof
    \noexpand\let\noexpand\endproof\noexpand\endaxp@oldproof
    \noexpand\let\noexpand\claimproof\noexpand\axp@oldclaimproof
    \noexpand\let\noexpand\endclaimproof\noexpand\endaxp@oldclaimproof
    \noexpand\let\noexpand\section\noexpand\axp@oldsection
  }
}
```

\axp@unactivateeightbit

We need an auxiliary macro to disable active characters that have the high bit set when writing to the .axp file. See https://tex.stackexchange.com/a/145361/166858

```
\def\axp@unactivateeightbit{%
  \count@=128%
  \loop
    \catcode\count@=12%
    \ifnum\count@<255%
    \advance\count@\@ne
  \repeat}
```

axp@VerbatimOut (*env.*)
\FVB@axp@VerbatimOut
\FVE@axp@VerbatimOut

Using the functionalities of the fancyvrb package, we define a custom verbatim environment axp@VerbatimOut that writes every line to the \axp@proofsfile. We also use the previous macro to disable active characters with the eighth bit set, and we make sure the catcode of @ is reset for every verbatim environment, in case it is used by the user (e.g., as in the xypic package). Finally, as an additional precaution, we reset \FV@CatCodesHook that is for example set by the commandchars or commentchar option of \fvset.

```
\DefineVerbatimEnvironment{axp@VerbatimOut}{axp@VerbatimOut}{}
\def\FVB@axp@VerbatimOut{%
  \@bsphack
  \begingroup
    \axp@unactivateeightbit
    \FV@DefineWhiteSpace
    \def\FV@Space{\space}%
    \FV@DefineTabOut
    \def\FV@ProcessLine{\immediate\write\axp@proofsfile}%
    \let\FV@FontScanPrep\relax
    \let\@noligs\relax
    \def\FV@CatCodesHook{}%
    \FV@Scan}
\def\FVE@axp@VerbatimOut{%
    \immediate\write\axp@proofsfile{\noexpand\makeatletter}%
    \endgroup\@esphack}
```

toappendix (*env.*)

The entire content of this environment is put in appendix, starting a new appendix section beforehand if needed.

```
\newenvironment{toappendix}
  {\axp@writesection\axp@VerbatimOut}
  {\endaxp@VerbatimOut}
```

7.5.2 Definition of New Theorems

axp@seenreptheorem

Used to indicate whether a repeated theorem was just typeset, without its proof.

```
\newtoggle{axp@seenreptheorem}
```

axp@rpcounter

Sequentially incremented for every repeated theorem, used to create labels.

```
\newcounter{axp@rpcounter}
```

axp@equation
axp@equationx

Used to save the value of the equation counter, when repeqn is set to same.

```
\newcounter{axp@equation}
\newcounter{axp@equationx}
```

axp@newtheoremrep

With first argument foobar, we use \axp@newtheoremrep@definetheorem to define the regular version of the theorem foobar. We then patch \begin{foobar} so as not to expect a proof in the appendix and define an internal theorem axp@foobarrp that will be used in the appendix to restate the existing theorem.

```
\def\axp@newtheoremrep#1[#2]#3[#4]{%
  \axp@newtheoremrep@definetheorem{#1}{#2}{#3}{#4}%
  \expandafter\pretocmd\csname #1\endcsname{\noproofinappendix}{}{}%
  \axp@newtheorem*{axp@#1rp}{#3}%
  \axp@forward@setup{#1}{#2}{#3}{#4}%
```

We then define a foobarrep environment that increments the axp@rpcounter and typeset the regular foobar theorem with a label derived from the counter, along with a possible custom command to identify repeated theorems. We distinguish the case when the theorem argument has a note and when it does not. We save the equation counter before typesetting the theorem environment, to reset it to the same value in the repeated environment when repeqn is set to same.

```
  \NewEnviron{#1rep}[1][]{%
    \ifthenelse{\equal{\axp@repeqn}{same}}{%
      \setcounter{axp@equation}{\value{equation}}%
    }{}%
    \addtocounter{axp@rpcounter}{1}%
    \ifx\relax##1\relax
      \axp@with@forward{#1}{\begin{#1}}\label{axp@r\roman{axp@rpcounter}}%
    \else
      \axp@with@forward{#1}{\begin{#1}[{##1}]}\label{axp@r\roman{axp@rpcounter}}%
    \fi
    \mainbodyrepeatedtheorem
    \BODY\end{#1}%
```

We set the axp@seenreptheorem toggle to indicate that we are looking for the proof of the theorem, then store in a macro the content of the theorem's body.

```
    \global\toggletrue{axp@seenreptheorem}%
    \global\expandafter\let\csname rplet\roman{axp@rpcounter}%
                           \endcsname
    \BODY
```

Possibly after starting a new appendix section if needed, we typeset a repeated version of the theorem using the axp@foobarrp environment and a reference to the previously defined label. We use \axp@redefinelabels in this environment to avoid multiply defined labels. We have to deal in a careful way with theorem notes: we want to use a theorem note to display the number of the repeated theorem, but theorem notes are usually typeset in a much different way (different font, parentheses) than theorem headings. In the case of the Springer document classes, we use the \theopargself macro to disable parentheses. For other document classes, we need to manually patch the \thmhead command at the right time. We also specially cover the case of the ACM document class where \@acmplainnotefont is used instead of \thm@notefont.

```
    \axp@writesection%
    \immediate\write\axp@proofsfile{\noexpand\makeatletter}%
    \ifthenelse{\equal{\axp@repeqn}{same}}{%
      \immediate\write\axp@proofsfile{%
        \noexpand\setcounter{axp@equationx}{\value{equation}}%
        \noexpand\setcounter{equation}{\theaxp@equation}%
      }%
    }{}%
    \ifbool{axp@forward@suppress}{%
      \global\def\axp@refstar{\ref*}
    }{%
      \global\def\axp@refstar{\ref}
    }
    \immediate\write\axp@proofsfile{{%
      \ifdefined\theopargself
        \noexpand\theopargself
      \else
        \noexpand\pretocmd{\noexpand\@begintheorem}{%
          \noexpand\patchcmd{\noexpand\thmhead}{\noexpand\@acmplainnotefont}{}{}{}%
          \noexpand\patchcmd{\noexpand\thmhead}{\noexpand\the\noexpand\thm@notefont}{}{}{}%
          \noexpand\patchcmd{\noexpand\thmhead}{(}{}{}{}%
          \noexpand\patchcmd{\noexpand\thmhead}{)}{}{}{}%
        }{}{}
      \fi
      \noexpand\begin{axp@#1rp}
        [%
          \noexpand\axp@refstar{axp@r\roman{axp@rpcounter}}%
          \@ifnotempty{##1}{%
            \ifdefined\theoargself
            \else
              \ifdefined\@acmplainnotefont
                \noexpand\@acmplainnotefont
              \else
                \noexpand\ifdefined\noexpand\thm@notefont
                  \noexpand\the\noexpand\thm@notefont
                \noexpand\fi
              \fi
            \fi
            {} (\unexpanded{{##1}})%
          }%
        ]%
        \noexpand\axp@forward@target{axp@fw@r\roman{axp@rpcounter}}{}%
        \noexpand\axp@redefinelabels
        \expandafter\noexpand\csname rplet\roman{axp@rpcounter}%
                             \endcsname
      \noexpand\end{axp@#1rp}
    }}%
    \ifthenelse{\equal{\axp@repeqn}{same}}{%
      \immediate\write\axp@proofsfile{%
        \noexpand\setcounter{equation}{\value{axp@equationx}}%
      }%
    }{}%
  }%
}
```

7.5.3 Forward-Linking Mechanism

When hyperref is loaded, foobarrep environments in the main text have their number link to their repetition in the appendix.

\axp@with@forward In order to make the number of the foobarrep theorem a link to its repeated version, we temporarily redefine the \thefoobar command, or, if we inherited the counter from a bazbar environment, the \thebazbar command. This seems to be the only robust way to make the number a \hyperlink, without adding extensive dependence on internals of amsthm, the builtin \newtheorem and possibly document-class specific definitions.

In order to allow users to redefine \thefoobar without breaking this feature, we redefine \thefoobar only for the duration of the \begin{foobar} form, resetting it to the old value as soon as possible.

Redefining `\thefoobar` has the side effect of changing `\newlabel` entries in the `.aux` file, so we need to to be able to disable addition of the hyperlink, which is why we use an intermediate `\axp@forward@link{`⟨*target*⟩`}{`⟨*text*⟩`}` macro, We also redefine `\theHfoobar` which is used by `hyperref` but not defined if `hyperref` was loaded after `\newtheoremrep` was used and `\protect` it to output it verbatim into the `.aux` file.

These hyperlinks are of course disabled in the `strip` compilation mode.

```
 \newcommand{\axp@with@forward}[2]{%
   \ifthenelse{\equal{\axp@appendix}{strip}}{#2}{
     \global\booltrue{axp@forward}%
     \ifcsundef{axp@old@the\csname axp@cn@#1\endcsname}{%
       \csletcs{axp@old@the\csname axp@cn@#1\endcsname}{the\csname axp@cn@#1\endcsname}%
       \csletcs{theH\csname axp@cn@#1\endcsname}{the\csname axp@cn@#1\endcsname}%
       \csdef{the\csname axp@cn@#1\endcsname}{%
         \protect\axp@forward@link{axp@fw@r\roman{axp@rpcounter}}%
           {\csname axp@old@the\csname axp@cn@#1\endcsname\endcsname}%
       }%
     }{}%
     #2%
     \ifcsdef{axp@old@the\csname axp@cn@#1\endcsname}{%
       \csletcs{the\csname axp@cn@#1\endcsname}{axp@old@the\csname axp@cn@#1\endcsname}%
     }{}%
     \global\boolfalse{axp@forward}
   }}%
```

`\axp@forward@link`
`axp@forward`

Dummy macro, for handling the unwanted change of the `\newlabel` entry in the `.aux` file caused by changing the definition of `\thefoobar`. We also use `\texorpdfstring` to detect whether the reference appears within a PDF bookmark, in which case we skip the test altogether.

```
\newbool{axp@forward}
\newcommand{\axp@forward@link}[2]{%
  \ifdefined\texorpdfstring\else\newcommand\texorpdfstring[2]{#2}\fi
  \texorpdfstring{%
    \ifboolexpr{ bool {axp@forward} and not bool {axp@forward@suppress} }{%
      \ifcsdef{hyperlink}{%
        \hyperlink{#1}{#2}%
      }{%
        #2%
      }%
    }{%
      #2%
    }%
  }{%
    #2%
  }%
}%
```

`\axp@forward@target`

Provides the needed `\hypertarget`. Intended to be written to the `.axp` file.

```
\newcommand{\axp@forward@target}[2]{%
  \ifcsname hypertarget\endcsname
    \hypertarget{#1}{#2}%
  \else
    #2%
    \fi
  }
```

\axp@forward@setup In order to support counter inheritance with the first optional argument of `\newtheoremrep`, we need access to the name of the counter. For compliance with the behavior of `\@axp@newtheorem`, the first optional argument (`#2`) is ignored if the second optional argument (`#4`) is given. We also check whether `aliascnt` was used to define this counter; if so, we use the original name as the counter name to handle the way such aliases work.

```
\newcommand{\axp@forward@setup}[4]{%
  \csedef{axp@cn@#1}{\ifblank{#4}{\ifblank{#2}{#1}{\ifcsdef{AC@cnt@#1}{#1}{#2}}}{#1}}%
}
```

7.5.4 Proof Environments

\noproofinappendix Utility macro that toggles `axp@seenreptheorem` to false.

```
\newcommand\noproofinappendix{%
  \global\togglefalse{axp@seenreptheorem}%
}
```

appendixproof (*env.*) We dump the content of this in appendix, within an original `proof` environment, possibly after creating a new appendix section. We support optional arguments in proofs, but we need to be careful not to gobble any newline character, that would make it impossible to process the first line with fancyvrb.

```
\def\appendixproof{\catcode`\^^M=\active\@ifnextchar[{\catcode`\^^M=5\@@appendixproof[axp@c
\def\@appendixproof[#1]%
  {%
    \axp@writesection
    \immediate\write\axp@proofsfile{%
      \noexpand\makeatletter\noexpand\begin{#1}\noexpand\makeatother%
    }%
    \axp@VerbatimOut
  }
\def\@@appendixproof[#1][#2]%
  {%
    \axp@writesection
    \immediate\write\axp@proofsfile{%
      \noexpand\makeatletter\noexpand\begin{#1}[\unexpanded{#2}]\noexpand\makeatother%
    }%
    \axp@VerbatimOut
  }
\def\endappendixproof
  {%
    \endaxp@VerbatimOut
    \immediate\write\axp@proofsfile{%
      \noexpand\end{axp@oldproof}%
    }%
    \noproofinappendix
  }
```

proof (*env.*) This environment either puts the proof in appendix, if we are after a repeated theorem without its proof, or inlines it otherwise. We support optional arguments in proofs, but we need to be careful not to gobble any newline character, that would make it impossible to process the first line with fancyvrb.

```
\def\proof{\catcode`\^^M=\active\ltx@ifnextchar@nospace[{\catcode`\^^M=5\axp@@proof}{\catco
\def\axp@proof
  {%
    \iftoggle{axp@seenreptheorem}{%
      \appendixproof
    }{%
      \axp@oldproof
    }%
  }
\def\axp@@proof[#1]%
  {%
    \iftoggle{axp@seenreptheorem}{%
      \appendixproof[#1]
    }{%
      \axp@oldproof[#1]
    }%
  }
\def\endproof
  {%
    \iftoggle{axp@seenreptheorem}{%
      \endappendixproof
    }{%
      \endaxp@oldproof
    }%
  }
```

claimproof (*env.*) If the claimproof environment exists (lipics document class or user-defined), we also redefine it, in the same way.

```
\AtBeginDocument{
  \ifdefined\claimproof
  \let\axp@oldclaimproof\claimproof
  \let\endaxp@oldclaimproof\endclaimproof
  \def\claimproof{\catcode`\^^M=\active\ltx@ifnextchar@nospace[{\catcode`\^^M=5\axp@@claimp
  \def\appendixclaimproof{\catcode`\^^M=\active\@ifnextchar[{\catcode`\^^M=5\@@appendixproc
  \def\axp@claimproof
    {%
      \iftoggle{axp@seenreptheorem}{%
        \appendixclaimproof
      }{%
        \axp@oldclaimproof
      }%
    }
  \def\axp@@claimproof[#1]%
    {%
      \iftoggle{axp@seenreptheorem}{%
        \appendixclaimproof[#1]%
      }{%
        \axp@oldclaimproof[#1]%
      }%
    }
  \def\endclaimproof
    {%
```

```
      \iftoggle{axp@seenreptheorem}{%
        \endappendixclaimproof
      }{%
        \endaxp@oldclaimproof
      }%
    }
  \def\endappendixclaimproof
  {%
    \endaxp@VerbatimOut
    \immediate\write\axp@proofsfile{%
    \noexpand\end{axp@oldclaimproof}%
    }%
    \noproofinappendix
  }
  \fi
}
```

`inlineproof` (*env.*)
`nestedproof` (*env.*)

These two environments are synonyms for the original `proof` environment.

```
\let\inlineproof\axp@oldproof
\let\endinlineproof\endaxp@oldproof
\let\nestedproof\axp@oldproof
\let\endnestedproof\endaxp@oldproof
```

7.5.5 Section Management

`axp@seccounter`

Sequentially incremented for every section, used to create labels.

```
\newcounter{axp@seccounter}
```

`\axp@sectitle`

Saves the title of the last encountered section.

```
\def\axp@sectitle{}
```

`\axp@section`
`\axp@@sectiontestinput`
`\axp@@sectiontestsection`

This command `\axp@section` behaves similarly to `\axp@oldsection`, except that it first tests whether a `\section` follows, and if so, does not produce anything. This is useful to avoid producing empty sections in the appendix. Using the catchfile package, we also check whether a `\section` is within an `\input` that immediately follows.

```
\def\axp@section#1{%
  \@ifnextchar\input
    {\axp@@sectiontestinput{#1}}%
    {\axp@@sectiontestsection{#1}}%
}
\def\axp@@sectiontestinput#1\input#2{%
  \CatchFileDef{\axp@tmp}{#2}{}%
  \def\axp@tmptmp{\axp@@sectiontestsection{#1}}%
  \expandafter\axp@tmptmp\axp@tmp%
}
\def\axp@@sectiontestsection#1{\@ifnextchar\section{\makeatother}{\axp@oldsection{#1}\makea
```

`\axp@oldsection`
`\section`
`\@section`
`\@@section`

We redefine the `\section` command to create a label based on `axp@seccounter` and to store its title in `\axp@sectitle`. In order to support starred and unstarred versions, as well as the optional short-title argument, the intermediate macros `\@section` and `\@@section` are needed.

```
\let\axp@oldsection\section
```

```
\def\section{\@ifstar\@section\@@section}
\newcommand{\@section}[2][\relax]{\axp@@@section{*}{#1}{#2}}%
\newcommand{\@@section}[2][\relax]{\axp@@@section{}{#1}{#2}}%
\newcommand{\axp@@@section}[3]{%
  \global\def\axp@sectitle{#3}%
  \ifx\relax#2\relax
    \axp@oldsection#1{#3}%
  \else
    \axp@oldsection#1[{#2}]{#3}%
  \fi
  \addtocounter{axp@seccounter}{1}%
  \label{axp@s\roman{axp@seccounter}}%
}
```

\nosectionappendix We remove the current section title, to indicate no section should be created in the appendix.

```
\newcommand{\nosectionappendix}{
  \global\def\axp@sectitle{}%
}
```

\axp@writesection If \axp@sectitle is not empty, we create a new section in the appendix, referring to the main text section.

Here, we wrap \ref{axp@si} into \axp@protectref@i, in order to protect the label name from wrongly being converted to uppercase, e.g., in fancyhdr with \pagestyle{fancy}.

This macro is defined both in the .aux file (in order to ensure availability when typesetting the \tableofcontents), and immediately before typesetting the appendix section (to ensure availability in the \section command).

```
\newcommand\axp@writesection{%
  \ifx\axp@sectitle\@empty
  \else
    \edef\axp@tmp{%
      \noexpand\global\noexpand\def
      \expandonce{\csname axp@protectref@\roman{axp@seccounter}\endcsname}{%
        \noexpand\ref{axp@s\roman{axp@seccounter}}%
      }%
    }%
    \immediate\write\@auxout{\expandonce\axp@tmp}
    \immediate\write\axp@proofsfile{%
      \expandonce\axp@tmp^^J%
      \noexpand\axp@section{%
        \noexpand\appendixsectionformat{%
          \protect
          \expandonce{\csname axp@protectref@\roman{axp@seccounter}\endcsname}%
        }{\expandonce\axp@sectitle}%
      }%
    }%
    \nosectionappendix
  \fi
}
```

Finally, in a somewhat ad hoc manner, we disable the whole section management for \tableofcontents, which may be typeset using a section heading, but for which automatic section management does not make sense.

\axp@oldtableofcontents
\tableofcontents

```
\let\axp@oldtableofcontents\tableofcontents
\def\tableofcontents{{\let\section\axp@oldsection\axp@oldtableofcontents}}
```

7.5.6 Append Compilation Mode

```
\ifthenelse{\equal{\axp@appendix}{append}}{
```

\axp@oldbibliography
\bibliography

Unless the `bibliography` option is set to `common`, we need to set the appendix bibliography source to be the same as that of the main text, thanks to bibunits's `\defaultbibliography` macro.

```
  \ifthenelse{\equal{\axp@bibliography}{separate}}{
    \let\axp@oldbibliography\bibliography
    \renewcommand\bibliography[1]{%
      \defaultbibliography{#1}%
      \axp@oldbibliography{#1}%
    }
  }{}
```

After the end of the main text, we add the appendix (after the command `\appendixprelim` is issued) within a `bibunit` environment so as to typeset a separate bibliography for the appendix (unless the `bibliography` option is set to `common`). This is done using `\pretocmd` on `\@enddocumenthook` instead of `\AtEndDocument` because we want the code to be run before any code in the `\@enddocumenthook` that has been set in the document class, as in the amsart document class. There is an extra test to ensure an empty bibliography environment is not produced. The name of the bibliography is changed to `\appendixrefname`; in most document classes, it is called `\refname` but it is occasionally (`scrartcl`, `scrreprt`) called `\bibname`. An ad-hoc test is added to fix a conflict with the natbib package which redefines `bibcite` at the end of the document.

```
  \pretocmd{\@enddocumenthook}{%
    \ifdefined\NAT@testdef
      \renewcommand\bibcite[2]{%
        \global\@namedef{b@#1\@extra@binfo}{#2}%
      }
    \fi
    \appendixprelim
    \appendix
    \ifthenelse{\equal{\axp@bibliography}{separate}}{
    \begin{bibunit}[\appendixbibliographystyle]
    }{}
      \immediate\closeout\axp@proofsfile
      \input{\jobname.axp}
    \ifthenelse{\equal{\axp@bibliography}{separate}}{
      \ifdefined\refname
        \renewcommand{\refname}{\appendixrefname}
      \else\ifdefined\bibname
        \renewcommand{\bibname}{\appendixrefname}
      \fi\fi
      \let\axp@oldthebibliography\thebibliography
      \let\endaxp@oldthebibliography\endthebibliography
      \renewenvironment{thebibliography}[1]{%
        \def\axp@tmp{#1}%
        \ifx\axp@tmp\empty
          \gdef\axp@noappendixbibliography1\relax
        \else
          \begin{axp@oldthebibliography}{#1}%
        \fi
      }{%
        \ifdefined\axp@noappendixbibliography\relax\else\end{axp@oldthebibliography}%
      \fi}
      \appendixbibliographyprelim
      \putbib
    \end{bibunit}
    \ifdefined\NAT@testdef
      \let\bibcite\NAT@testdef
    \fi
   }{}
  }{}{}
}{}
```

7.5.7 Class-Specific Behavior

We conclude with some class-specific behavior.

ACM Document Classes (old versions, till 2017)

```
\ifdefined\@acmtitlebox
```

We first redefine the `proofsketch` environment, which is used differently in the base class.

```
\renewenvironment{proofsketch}{\begin{axp@oldproof}[sketch]}{\end{axp@oldproof}}
```

We adjust the styling of theorems for the needs of apxproof.

```
\newtheoremstyle{mystyle}
  {6pt}
  {6pt}
  {\itshape}
  {10pt}
  {\scshape}
  {.}
  {.5em}
  {}
\theoremstyle{mystyle}
```

`\thebibliography`
`\refname`
`\appendixrefname`

The section title of the bibliography is in uppercase in these document classes. In addition, the `\thebibliography` macro hard-codes twice the section title, so we un-hardcode it so that it can be modified in the appendix.

```
  \patchcmd{\thebibliography}{References}{\protect\refname}{}{}
  \patchcmd{\thebibliography}{References}{\protect\refname}{}{}
  \newcommand{\refname}{REFERENCES}
  \renewcommand{\appendixrefname}{REFERENCES FOR THE APPENDIX}
```

```
\fi
```

lipcs

```
  \ifdefined\lipics@opterrshort
```

\appendixbibliographyprelim The default bibliography in the lipics document class formatting is not compatible with the alpha bibliography style. We fix this here.

```
    \renewcommand{\appendixbibliographyprelim}{%
      \global\let\@oldbiblabel\@biblabel
      \def\@biblabel{\hspace*{-2em}\small\@oldbiblabel}%
    }
```

```
  \fi
}
```

Change History

v1.0.0

General: Initial released version . . 1

v1.0.1

General: Prevent empty bibliography environment; fix typos 1

v1.0.2

`axp@newtheoremrep`: Fix display of repeated theorem counter in some document classes 15

Fix missing space between repeated theorem counter and theorem note 15

v1.0.3

General: Note on entire sections in appendix 2

`\appendixbibliographyprelim`: Support for lipics-v2016 24

`proofsketch`: Ignore spaces after beginning of Proof sketch . . . 11

v1.0.4

General: More faithful theorem style for ACM templates 23

More robust coherent styling of proof sketches 23

Re-establish custom proof environments 8

Show options commented on in margin and index 1

`\appendixprelim`: Configurable appendix style 11

`\axp@bibliography`: `bibliography` option 9

v1.0.5

General: Ability to specify a sectioning counter in newtheoremrep 2

Fix compilation of proofsketch environment in inline mode . . . 3

v1.0.6

General: Better support of Springer document classes 11

Deal with document classes where the bibliography is called `\bibname` 22

Support of new ACM document class (`acmart.cls`) 23

`\axp@newtheorem`: Introduce intermediary command for theorem macro 10

`\axp@writesection`: Fix extraneous space after section number in appendix titles . . . 21

`axp@newtheoremrep`: Better handling of note-free theorems in document classes that treat theorems differently when they have an empty note 15

Fix incorrect use of `\noexpand` in optional argument of macro environment 15

v1.1.0

General: Added forward-link mechanism (K. D. Bauer) . . . 16

`\appendixsectionformat`: Fix missing space in default `\appendixsectionformat` . . . 11

`\axp@proofsfile`: Initialization deferred to `\AtBeginDocument` for compatibility with `\dumped` precompiled preambles (K. D. Bauer) 13

\axp@redefinelabels: Fix \label not being disabled in amsmath environments, where \label@in@display is used instead (K. D. Bauer) 11

\axp@repeqn: repeqn option 9

\axp@unactivateeightbit: Fix compilation of non-ASCII characters with \usepackage[utf8]{inputenc} 13

\axp@writesection: Make \axp@tmp wrapper more robust. Resolves issues from use of section title in fancyhdr, and in \tableofcontents (K. D. Bauer). 21

\FVE@axp@VerbatimOut: Make apxproof compatible with independent use of fancyvrb . . 14

\section: Fix handling of fragile macros within section headings. See #22. 20

Rewrote definition of \section to enable optional argument. See #23. (K. D. Bauer) 20

\tableofcontents: Disable section management for table of contents 22

v1.2.0

General: Do not load bibunits if bibliography is set to common . 9

\axp@newtheoremrep: Fix formatting of theorems without notes in some document classes in inline mode 12

\axp@newtheoremrep@definetheorem: Restore predefined theorem counters 10

\axp@redefinelabels: Fix extra spacing erroneously introduced within the \axp@redefinelabels macro . 11

\mainbodyrepeatedtheorem: Configurable repeated theorem command 11

proofsketch: Fix proof sketches in inline compilation mode for Springer document classes . . . 12

axp@newtheoremrep: Fix display of theorem notes 15

v1.2.1

General: Ad hoc fix for natbib package conflict 22

\appendixrefname: Fix compatibility with memoir document class 11

\FVE@axp@VerbatimOut: Fix compatibility with xypic package 14

Fix compatibility with other uses of fancyvrb that set \FV@CatCodesHook 14

v1.2.2

General: Compatibility with AMS document classes: do not use \AtEndDocument 22

\axp@section: Detect a section within an included file to avoid produced useless sections 20

axp@newtheoremrep: Fix handling of optional arguments of repeated theorems containing optional arguments 15

v1.2.3

General: More robust redefinition of thebibliography environment, for compatibility with tocbibind 22

\axp@forwardlinking: forwardlinking option 9

v1.2.4

appendixproof: Support optional arguments in proofs 18

claimproof: Support for claimproof environment from lipics 19

v1.2.5

claimproof: Support for locally defined claimproof environment 19

axp@forward: Remove forward linking command from PDF bookmarks 17

v1.2.5-dev

General: Fix compilation issues in recent versions of acmart 12

\axp@forward@setup: Support for aliascnt counters 18

\axp@newtheoremrep@definetheorem: Undefine counter printing command to avoid warning . . 10

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

References

[1] Leslie Lamport. *LATEX: A Document Preparation System.* Addison–Wesley Pub. Co., Reading, MA, 1986.

A Proofs for Section 1 (Usage)

Proof. □

Foobar 2. *This foobar is repeated in the appendix.*

Proof. □