The \texttt{amsrefs} package

Michael Downes
American Mathematical Society

Version 1.12, 2002/01/03

1 Introduction

The \texttt{amsrefs} package is a \LaTeX{} package for bibliographies that provides an archival data format similar to the format of Bib\TeX{} database files, but adapted to make direct processing by \LaTeX{} easier. The package can be used either in conjunction with Bib\TeX{} or as a replacement for Bib\TeX{}.

As compared to the standard methods for putting a bibliography into a \LaTeX{} document (running Bib\TeX{} to extract selected items from a \texttt{.bib} database into a formatted \texttt{.bbl} file), the chief advantages of the \texttt{amsrefs} package for an author or for someone writing a document class are as follows:

\textbf{Preservation of structure} The internal structural information of the bibliography entries is not lost when they are imported from the database file into the \LaTeX{} document. This takes on its greatest significance when archiving documents in \LaTeX{} form or transmitting them to another user (such as a publisher).

\textbf{Deferred formatting} This means that the style of the bibliography can be readily changed without reimporting everything from the original database(s).

\textbf{More natural handling of titles} Proper nouns do not need to have braces added to prevent capitalization problems.

\textbf{Less ambiguous format for author names} When author names are given in inverted order (last name first) it is possible for \LaTeX{} to unambiguously identify the last name without any further markup, even in cases like Saunders Mac Lane (Mac Lane, Saunders) versus Stephen H. Lane (Lane, Stephen H.), or Cam Van Tran (Tran, Cam Van) versus Bert Van Keulen (Van Keulen, Bert) that require the addition of extra braces in Bib\TeX{}.

\textbf{Setup requires only \LaTeX{} knowledge} All bibliography setup can be done in \LaTeX{}; learning another programming language (such as, the one used in Bib\TeX{} bst files) is unnecessary.

\textbf{Self-printable database files} A \LaTeX{} document that contains only a bibliography in \texttt{amsrefs} forms can be used as a database for exporting entries to other documents. And because it is a \LaTeX{} document, the database
can be printed directly at any time simply by running it through \LaTeX in the usual way.

**Self-contained** In many cases it seems possible to do without Bib\TeX entirely. For example, if the entries are extracted from a single database file that is maintained in sorted order, the bibliography can be printed directly by \LaTeX on the first pass and the citations resolved on the second pass.

# 2 Using the amsrefs package

There are three main ways of using the amsrefs package:

1. Using the standard `\bibliography` command to import a `.bbl` file that was created by Bib\TeX with the amsxport bibliography style.

2. Using the `\bibselect` command. This is similar to using `\bibliography`, but with an extension of `.ltb` ("\LaTeX bibliography") for database files, rather than `.bib`, and with the extraction of the selected entries being done by \LaTeX itself on the first pass.

3. Writing bibliography items in amsrefs form directly into your \LaTeX document.

## 2.1 Using the amsrefs package with Bib\TeX

When Bib\TeX is used to generate a bibliography for use with the amsrefs package, your main document will look almost exactly the same as usual, with the `\bibliography` command used in the normal way. The only overt difference would be one line at the beginning to load the amsrefs package. The bibliography style to use would be a generic one called amsxport (see the amsxport documentation for more details).

```latex
\documentclass{article}
\usepackage{amsrefs}
...
\begin{document}
...
\bibliographystyle{amsxport}
\bibliography{bib-file-name}
```

If you look at the `.bbl` file generated by Bib\TeX, you will see that there are two main differences:

1. Instead of a thebibliography environment, the bibliography is wrapped in two environments:

   ```latex
   \begin{bibdiv}
   \begin{biblist}
   ...
   \end{biblist}
   \end{bibdiv}
   ```

2. Each entry in the bibliography is done with `\bib` instead of `\bibitem`, as discussed below in more detail.
2. USING THE \texttt{amsrefs} \texttt{PACKAGE}

2.2 \textbf{Using \texttt{\textbackslash bibselect} instead of \texttt{\textbackslash bibliography}}

One feature of the \texttt{amsrefs} package is the possibility of maintaining your bibliography data in the form of a \LaTeX\ document that can be directly printed, and using \LaTeX\ itself instead of Bib\TeX\ for importing cited items into other documents. If you write

\begin{verbatim}
\bibselect{combinatorics}
\end{verbatim}

instead of \begin{verbatim}\bibliography{combinatorics}\end{verbatim}, then \LaTeX\ will look for a file \texttt{combinatorics.ltb} (instead of \texttt{combinatorics.bib}), and search through it for a \texttt{biblist} environment. The \texttt{\textbackslash bib} entries in the list will be checked to see if they match \texttt{\textbackslash cite} commands in the calling document, and if so they will be copied into a \texttt{.bbl} file.

Because this is all being handled by \LaTeX, the contents of the bibliography can be printed on the first pass; cite numbers are resolved, consequently, on the second pass.

When \texttt{\textbackslash bibselect} is used, the \texttt{\bibliographystyle} command is irrelevant since its purpose is merely to pass information to Bib\TeX. There is one significant difference with the \texttt{\textbackslash bibselect} command: it produces only the actual contents of the bibliography, without the section title and without any bibliography environment wrapper. This is to make it possible to put them in your main document and freely customize them there, including (for example) including some introductory text at the beginning of the bibliography. In other words, you can write something like

\begin{verbatim}
\begin{bibchapter}[Annotated Bibliography]
(Some introductory text here.)
\begin{biblist}
\bibselect{combinatorics}
\end{biblist}
\end{bibchapter}
\end{verbatim}

The \texttt{\textbackslash bibselect} command always regenerates the \texttt{.bbl} file (unless the \texttt{\textbackslash nofiles} switch is used). When a time comes that you no longer want it to be regenerated, comment out the \texttt{\textbackslash bibselect} command and copy the bbl file into your \LaTeX\ file. This is particularly relevant if you put any line break or page break commands into the bibliography.

2.3 \textbf{Direct entry of bibliography items}

When the bibliography is written by hand each entry is written with \texttt{\bib} instead of \texttt{\bibitem}, and has the form

\begin{verbatim}
\bib{BW}{article}{
  author={Bertram, A.},
  author={Wentworth, R.},
  title={Gromov invariants for holomorphic maps on Riemann surfaces},
  date={1996},
  journal={J. Amer. Math. Soc.},
  volume={9},
  number={2},
}\end{verbatim}
The first and second arguments give the citation key and type; the remainder is a list of bibliographic field names and field values in more or less standard LATEX key-value syntax. The syntax for \texttt{\bib} is intentionally restricted, however, to eliminate certain variations—such as omitting the braces around the value—that are not really a good idea when the values can contain arbitrary LATEX text.

If you try leaving out an equal sign or the comma at the end of a line, you will see that the warning messages for syntax errors usually identify the type and location of the error precisely enough that making the corrections is straightforward. (For more details see the documentation of the \texttt{rkeyval} package.)

3 More about the \texttt{\bib} command

The \texttt{\bib} command is a drastically different replacement for the \texttt{\bibitem} command. Compare the example shown above as it would look if done with \texttt{\bibitem}:

\texttt{\bibitem{BW} A. Bertram and R. "Wentworth,}  
\texttt{\emph{Gromov invariants for holomorphic maps on \texttt{\{R\}iemann surfaces}},}  
\texttt{J. Amer. Math. Soc. \texttt{\textbf{9}} (1996), no. \texttt{\textbf{2}}, 529--571.}

With the \texttt{\bib} command, the form of the data in the LATEX document is fairly close to the form used in Bib\TeX\ database files. There are, however, some significant differences.

- **braces** Always use braces to enclose the value of each field (standard LATEX syntax), never double quotes (Bib\TeX\ allows both quotes and braces).
- **repeated fields** Fields can be repeated (and should be, where applicable). In this example each author name is given separately. The task of combining the names as needed for the current publication is handled by LATEX.
- **inverted names** Author and editor names should be written uniformly in Last, First order. This is the form that provides the maximum flexibility with the least ambiguity and least extra markup. When printed, the names will be automatically uninverted in whatever manner is specified by the bibliography style in use. (For example, some styles have the first author’s name inverted and the remaining names not inverted.)
- **capitalization** In titles, do not capitalize words that are not proper nouns. Capitalization will be applied automatically as needed.

**Bib\TeX\ Note:** Proper nouns do not need to be written with extra braces: you can simply write Riemann instead of \texttt{\{R\}iemann} or \texttt{\{Riemann\}}.

- **ndash** Use of \texttt{\textendash} instead of \texttt{--} for en-dashes is recommended (the \texttt{--} notation, which is tied to idiosyncrasies of the Computer Modern “\TeX\ text” font encoding, hinders document translation and the use of alternate fonts).
3. MORE ABOUT THE \bib COMMAND

**date** It is recommended to use a date field to give the year of publication; although year is also accepted, date is more general. If the date includes a month, it is recommended to use numbers for months in ISO form, e.g., 1987-12. This allows month names to be printed in full or abbreviated at will. For “Winter”, “Spring”, “Summer”, “Fall”, either use month numbers of 13, 14, 15, 16 (respectively), or just put in the text before the year:

date={Summer 1987},

3.1 Field names for the \bib command

The following field names are currently supported by the amsrefs package

<table>
<thead>
<tr>
<th>field name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>author+</td>
<td>place</td>
</tr>
<tr>
<td>editor+</td>
<td>address*</td>
</tr>
<tr>
<td>isbn+</td>
<td>date</td>
</tr>
<tr>
<td>review+</td>
<td>year*</td>
</tr>
<tr>
<td>title</td>
<td>volume</td>
</tr>
<tr>
<td>booktitle*</td>
<td>number</td>
</tr>
<tr>
<td>series</td>
<td>pages</td>
</tr>
<tr>
<td>conference</td>
<td>part</td>
</tr>
<tr>
<td>publisher</td>
<td>issn</td>
</tr>
<tr>
<td>organization</td>
<td>doi</td>
</tr>
<tr>
<td>institution</td>
<td>language</td>
</tr>
</tbody>
</table>

The + character indicates that a field is repeatable. The * character indicates an alternative field name supported for compatibility with BibTeX; the field name just above a starred name is the equivalent provided by amsrefs. For example, the date field can contain not only the year but also month and day, if relevant, in ISO 8601 form: 1987-06-21. If month and/or day are present, they can be printed in different forms depending on the current bibliography style.

3.2 Bibliography entry types

It behooves a bibliography package that has ambitions of serving for general use to support a rather wide range of entry types, e.g.,

<table>
<thead>
<tr>
<th>entry type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>book</td>
<td>supplement</td>
</tr>
<tr>
<td>collection</td>
<td>fragment</td>
</tr>
<tr>
<td>series</td>
<td>review</td>
</tr>
<tr>
<td>set</td>
<td>play</td>
</tr>
<tr>
<td>article</td>
<td>screenplay</td>
</tr>
<tr>
<td>periodical</td>
<td>personal</td>
</tr>
<tr>
<td>report</td>
<td>lecture</td>
</tr>
<tr>
<td>thesis</td>
<td>speech</td>
</tr>
<tr>
<td>manual</td>
<td>reading</td>
</tr>
<tr>
<td>reference</td>
<td>story</td>
</tr>
</tbody>
</table>

(See discussion below.)

The current release of amsrefs does not provide support out of the box for all of the above types, but only the ones that are commonly used in AMS publications. On the other hand, adding support for a new type is scarcely any
more work than modifying an existing type: in either case if you write down the
desired output style in a \BibSpec command, voilà, the type is now supported.
If it needs any unusual fields, you will need to define the extra field names as well,
but adding a new field normally requires only one line, a \DefineSimpleKey
statement following the example of the ones in amsrefs.sty.

Discussion of individual types:

book A written work by one or more authors where the authors share credit
for the work as a whole.

collection A written work by multiple authors where distinct parts are cred-
ited to particular authors. This includes conference proceedings volumes.
Collected works of a single author would also be classed as a collection,
even though everything is written by the same person.

series A sequence of books or collections with a common series name given by
the publisher. New releases are published as they become ready, rather
than following a definite regular schedule (compare to periodical).

set Like book or collection but split into multiple volumes, where the vol-
umes might be published either simultaneously or sequentially as they
are finished; as compared to series, a set has a more definite (usually pre-
established) number of volumes, and is usually the work of a single author,
or collaborating authors, rather than multiple independent authors. A set
may consist of books, articles collections, reports, etc.—e.g., a multi-part
article published over several issues of a journal can be classified as a set to
facilitate citing it as a whole—but it would be unusual for a set to contain
periodicals or series.

article A relatively short—usually less than 50 pages—written item that ap-
peared as part of a periodical or collection; or, possibly, as a World Wide
Web document, in which case the URL should be included in the citation
info as well as the date on which it was read.

periodical Like a series of collections but following a definite schedule of pub-
lication, usually two times or more per year. This includes newspapers,
magazines, and journals.

report Technical description, white paper, or the like. Similar to article but
produced by the author(s) for a particular organization, usually their em-
ployer, whose primary business is not publishing.

thesis Like report but produced by the author for an educational institution
to satisfy the requirements for a degree. Not (normally) produced by a
commercial publisher for sale.

manual A special case of book: Instructions for dealing with a particular kind
of software or manufactured product.

reference This could be considered as a general type subsuming indexes into
various kinds of literature, dictionaries, and other kinds of books where
items are looked up by name or title.

story Short story, longer story, fiction
review Like article, usually, but specifically to discuss some other citable item.

personal A personal communication in the form of mail, e-mail, phone call, face-to-face conversation, telegram, etc.; unlike most other cited items, personal communications are usually not available in libraries.

lecture This type provides for a kind of “publication” that consists primarily in the verbal presentation, or possibly in printed lecture notes given out at the time of the lecture. It is somewhat rare for these to have a formal publisher. A lecture might be published in audio or video form.

speech Political or academic address intended for persuasive rather than pedagogic purposes. A commencement address would not normally be considered a lecture.

reading A public reading of poetry or fiction. This could be subsumed under lecture in respect of the etymology.

image Visual image in some form or another. See also map.

artwork Similar to image, but could also include sculpture and other types of visual art.

map Like image but specifically for spatial and navigational information about physical locations.

audio Speech captured in the form of sounds (rather than transcription) on audio-cassette or other media. See also music, lecture, speech.

music Like audio but restricted to items that are generally held to be music.

video Moving pictures, usually also accompanied by sound.

film A slight variation of video: Moving pictures that were released “on the big screen”.

software Computer programs and electronic files whose main purpose is to contribute to the operation of a program, rather than to be read by a human being. \TeX: the Program (Knuth, 1986) is an example of a computer program published in book form; when such a form exists it could be cited either as a book or as software. Consider what will best serve the reader.

play Drama.

screenplay A variant of play.

miscellaneous None of the above, including pamphlets, restaurant menus, patents, grant proposals, court records, business plans, or what have you.

supplement Anything added to a parent published item (including supplements, corrigenda, errata, etc.), that appeared separately from the original item.

fragment Some part of one of the above types, such as a chapter from a book, an appendix from an article, a page from a World Wide Web document, etc.

The reply to a question that will surely strike some readers: “Why the absence of incollection or url types?” is that it is better to use a type that indicates the intrinsic nature of the writing, rather than the context in which
it occurs. An article that appears in a larger collection or on the World Wide Web is still an article and should be cited as such, using the place of publication or xref fields to indicate the manner of publication. Consider how silly it would be to cite all manner of material that might appear on a web page as type url: screenplay, map, video??

4 Citing a source

The standard \cite command of \LaTeX mixes single-cite and multiple-cite functionality in a single command; although most of the time this is convenient, it is something of a departure from the \LaTeX philosophy of logical markup, and gives rise to awkward complications in certain situations. You can write, for example, \cite{xx,yy} to get [13, 15] and \cite[Chapter 2]{xx} to get [13, Chapter 2] but there are serious difficulties in getting [13, Chapter 2; 15] or [13; 15, Chapter 2].

The \amsrefs package therefore provides two primary citing commands: \cite for single cites and \citelist for multiple cites. Some additional variants of those two commands are provided for convenience.

\cite{xyz} works the same as ever, but for the sake of consistency it is recommended that multiple cites be handled with \citelist (or \cites—see below), not with \cite. The square bracket notation \cite[...]{...} is also deprecated, because a superior alternative is provided:

\cite{xyz}{Chapter 2}

Not only is this ordering of the arguments more natural, the star/braces syntax prevents certain scanning problems that afflict the bracketed form—for example, if one wants to put a cite command into the optional argument of a theorem environment:

\begin{thm} \cite[Theorem 4.9]{xyz} \end{thm}

Although the square-bracket syntax continues to be supported for compatibility’s sake, users who stick to the \cite{...}{...} form will never be confronted unexpectedly with the puzzling \TeX error messages that result from scanning problems as in this example.

\citelist takes a list of \cite commands as its argument; this makes it possible for any member of the list to be a qualified cite, not just the last one. For example:

\citelist{\cite{xx} \cite{yy}{Chapter 2} \cite{zz}}

Note that the members of the list are not separated by commas. The punctuation between individual cites is supplied automatically (resist the temptation to add it by hand, since that will seldom produce the intended results).

When the cites are numerical, multiple cites are sorted and ranges of length three or more are compressed, as is done by the renowned \cite package, unless the \amsrefs option non-sorted-cites is used. However, range compression is turned off when the hyperref package is used, following its usual practice.
\cites is a variant of \citelist, provided for convenience:
\cites{aa,bb,cc}

is equivalent to \citelist{\cite{aa}\cite{bb}\cite{cc}}. This can only be used when all the citations are simple ones with no options.
\ycite (year only), \ocite (object cite), and some related commands are variants of \cite that provide the distinctions needed in an (Author, Year) citing scheme. See the discussion in Section 5 below.

5 Author-year citation schemes

In author-year citation schemes three main citing forms are required to cover the cases that in other schemes require only one form. The first form is used when the citation serves as a parenthetical annotation—i.e., it could be omitted without harming the grammatical structure of the sentence containing it. For example:

The question first arose in systems theory (Rupp and Young, 1977).

The second form is like the first but is used when the author name is already present as a natural part of the sentence, and the cite therefore ought to supply only the year:

Rupp and Young (1977) have investigated . . .

The third form is used when the citation serves as a direct object or other inomissible noun-like object within its sentence. For instance, dropping the citation from the following sentence leaves a grammatically incomplete remainder:

. . . for further details, see Rupp and Young (1977).

The logic of requiring three forms is perhaps most clearly seen if we envision replacing the author-year citations by numerical citations. The type of text replaced by [14] is different in each case:

. . . arose in systems theory [14].

Rupp and Young [14] have investigated . . .

. . . for further details, see [14].

We delegate \cite to produce the primary parenthetical form (Author, Year) and provide \ycite (“year cite”) and \ocite (“object cite”) as the other forms. Plural forms \ycites and \ocites are also provided in parallel with \cites. And \citeauthor can be used to produce the list of author names without the year.

These correspond with command names from other commonly used author-year packages as follows:

<table>
<thead>
<tr>
<th>amsrefs</th>
<th>harvard</th>
<th>natbib</th>
</tr>
</thead>
<tbody>
<tr>
<td>\cite, \cites</td>
<td>\cite</td>
<td>\citetep</td>
</tr>
<tr>
<td>\ycite, \ycites</td>
<td>\citeyear</td>
<td>\cite</td>
</tr>
<tr>
<td>\ocite, \ocites</td>
<td>\citeasnoun</td>
<td>\citet</td>
</tr>
<tr>
<td>\citeauthor</td>
<td>(none)</td>
<td>\citeauthor</td>
</tr>
<tr>
<td>\citeauthory</td>
<td>(none)</td>
<td>\citet</td>
</tr>
</tbody>
</table>
Some people like to use \citet or \citeasnoun when the author name serves as the subject of a sentence; to my mind this is a questionable blurring of the boundary between the essential text of the sentence and the bibliography pointer. Those who don’t feel a need to bother about such distinctions can write \citeauthor{xyz} if they choose as an abbreviation for \citeauthor{xyz} \ycite{xyz}

I.e., the command name is \citeauthor+\ycite except for dropping the redundant second cite.\footnote{Why not simply call it \aycite, you may ask? Well, I could hardly pass up the opportunity to get all six vowels in a single command name, could I? (If you really want a shorter command name you can always provide it yourself with \newcommand.)}

When an author-year scheme is in use, parens are normally added by \cite, \citelist and their variants unless the character immediately following the command’s argument is a closing paren. This suffices for most cases; for special cases one can use the \parenthesize command.

Starred forms \cite* and \ocite* print the full list of author names instead of an abbreviated list, when an (Author, Year) style of citation is in use. For other citation schemes they produce the same output as the unstarred forms. Some citation styles require the first cite to use the full list and subsequent ones to use the abbreviated version. With proper setup this can be done automatically, so that the starred forms of these commands should seldom be necessary in practice.

6 Section titles for bibliographies

The \amsrefs package provides three environments for producing the section or chapter title of a bibliography:

\begin{bibchapter}
...
\end{bibchapter}

If you write

\begin{bibchapter}
...
\end{bibchapter}

it is equivalent to writing

\section*{\bibname}

Similarly \bibsection corresponds to \section*{\refname}, whereas \bibdiv produces one or the other depending on context.

7 The \biblist environment

The \amsrefs package provides a \biblist environment which is used for the body of the bibliography. This makes it easier to add introductory text after the section or chapter heading but before the first bib-item. Instead of using \thebibliography, you would write, for example

\begin{quote}
...
8. LINE BREAKS IN THE BIBLIOGRAPHY

\begin{bibchapter}[Annotated Bibliography]
(Optional introductory text)
\begin{biblist}
\bib{...}{...}{ ... }
\bib{...}{...}{ ... }
\bib{...}{...}{ ... }
\end{biblist}
\end{bibchapter}

8 Line breaks in the bibliography

When you want a line break between two fields, put the \texttt{\linebreak} command at the end of the first field. Although this would seem to invite, in most cases, a quite erroneous line break preceding the punctuation that is automatically added, there is special handling behind the scenes to move the line break further along to the right place.

\texttt{title={Bayesian analysis \ldots econometrics\linebreak}},

It should be fairly obvious that any change in the bibliography style might invalidate existing linebreaks.
9 Capitalization of titles

Here is a summary of the rules given in the *Chicago Manual of Style* [3] for capitalizing titles of written works in English:

Capitalize each word, including pronouns and subordinate conjunctions, except for articles, coordinate conjunctions, and prepositions, or the word to in infinitives. Always capitalize the first and last word of the title and the first and last word of any subtitles that it may contain. Don’t capitalize the second or later word in a hyphenated compound unless it is a noun or proper adjective, or it has equal force with the first word.

10 Author names

In general BibTeX does quite a good job of dealing with names. But there are a few problem cases. Consider the following examples given in normal and inverted order.

- John Q. Smith ↔ Smith, John Q.
- John Q. Smith, Jr. ↔ Smith, John Q., Jr.
- John Q. Smith III ↔ Smith, John Q., III
- John Quincy Smith ↔ Smith, John Quincy
- Ch. B. Chase ↔ Chase, Ch. B.
- Rip van Winkle ↔ van Winkle, Rip
- Thomas W. de la Ware ↔ de la Ware, Thomas W.
- Jakob Ben Isaak ↔ Ben Isaak, Jakob
- Anna Palamara Orsi ↔ Palamara Orsi, Anna
- Yin Wuxiang ↔ Yin Wuxiang
- Wuxiang Yin ↔ Yin, Wuxiang

11 Customizing bibliography style

If you use the *amsrefs* package as is, the bibliography style you get is the kind of style customarily seen in AMS publications. The recommended way to get a different bibliography style is to write a LaTeX package which loads the *amsrefs* package with \RequirePackage and then makes the desired changes by using suitable \BibSpec commands as explained below. Thus, the general form of the custom package will be

\ProvidesPackage{xyzbib}[2002/01/03 v1.23]
\RequirePackage{amsrefs}\relax
\BibSpec{article}{
  ...
}
\BibSpec{book}{
  ...
}
The interior formatting within entries is specified by \texttt{BibSpec} commands, one for each entry type. To illustrate, let’s look at an example style spec for entries of type \texttt{article}:

\begin{verbatim}
\BibSpec{article}{% 
  +{\PrintAuthors}  \{author\}
  +{,}  \{title\}
  +{,}  \{journal\}
  +{\textbf}  \{volume\}
  +{\parenthesize}  \{date\}
  +{,}  \{pages\}
  +{,}  \{note\}
  +{\textbf}\{transition\}
  +{}  \{review\}
}\end{verbatim}

It should be pretty obvious that each line specifies the formatting for a particular field. The fundamental model here is that after reading the data for a particular \texttt{bib} command, \LaTeX{} steps through the style spec and for each field listed, prints the field with the given formatting \textit{if and only if the field has a nonempty value}. The + character at the beginning of each field spec must be followed by three arguments: the punctuation to be added if the field is nonempty; space and/or other material to be added after the punctuation; and the field name. It is permissible for the second part to end with a command that takes an argument, such as \texttt{\textbf}, in which case it will receive the field’s value as its argument. By defining a suitable command and using it here you can place material after the field contents as well as before; \texttt{\parenthesize} is an example of this.

The reason that the punctuation and the following space are specified separately is that between them there is a crucial boundary for line breaks. If you put a \texttt{\linebreak} command at the end of a field value, the break point will actually be carried onward to a suitable point after the next bit of punctuation (whose actual value may vary depending on which of the following fields is the first to turn up with a nonempty value).

The meaning of the \texttt{\parenthesize} command, supplied by \texttt{amsrefs}, should be obvious. The meaning of the \texttt{\PrintAuthors} command is a different story. But I don’t think it is all that hard to understand. If we have two or three author names which were given separately, and we need to combine them into a conventional name list using commas and the word “and”, then it would be nice if we had a command which could take a list of names and Do The Right Thing. And that is just what \texttt{\PrintAuthors} is.

The \texttt{rkeyval} package allows keys to be defined as additive: if the key occurs more than once, each successive value will be concatenated to the previous value, along with a prefix. The setup done by \texttt{amsrefs} for the \texttt{author} field is

\begin{verbatim}
\DefineAdditiveKey{bib}{author}{\name}
\end{verbatim}

This means that if two names are given, as in
THE AMSREFS PACKAGE

\author{Bertram, A.},
\author{Wentworth, R.},

then the final value of the author field seen when \LaTeX \ processes the style spec will be

\name{Bertram, A.}\name{Wentworth, R.}

The transition field in our BibSpec example is a dummy field to be used when punctuation or other material must be added at a certain point in the bibliography without regard to the emptiness or non-emptiness of the fields after it. The transition field always tests as non-empty but has no printed content. So when you use it you always get the indicated punctuation and space at the indicated point in the list of fields. If it were the last thing in this BibSpec example, it could serve just to put in the final period that is always wanted. But in AMS bibliographies, if a Mathematical Reviews reference is given, it is conventionally printed after the final period. Using the transition field as shown here ensures that the final period will be always printed, even when the review field is empty.

11.1 Miscellaneous commands provided by the amsrefs package

Most of the following commands are helper commands for use in \BibSpec statements. The others are intended for use in bibliography data.

\parenthesize This command adds parentheses around its argument. It is useful in \BibSpec statements because there is no special provision for adding material after the field value.

\bibquotes This command is much like \parenthesize but it adds quotes around its argument and it has one other important difference: there are special arrangements to print the closing quote after a following comma or similar punctuation (unless the amsrefs package is invoked with the logical-quotes option, in which case \bibquotes puts the closing quote immediately after the quoted material).

\voltext The normal definition of this command is vol.~ to supply the text that precedes a volume number.

\editiontext This command produces ed. following an edition number. See \PrintEdition for more information.

\DashPages This command is similar in spirit to \voltext but more complicated in its implementation. It takes one argument which is expected to contain one or more page numbers or a range of page numbers. The argument is printed with a prefix of “p.” if it seems to be a single page number, otherwise with a prefix of “pp.”.

\tsup, \tsub, \prime These are for text subscripts and superscripts, with \prime producing a superscript prime symbol. Unlike the standard \textsuperscript and \textsubscript functions provided by \LaTeX, these do not use math mode at all.
\nopunct This command causes following punctuation to be omitted if it is added with the internal function \@addpunct (which is used throughout the \BibSpec handling when appending a non-empty field).

\PrintAuthors This is a relatively complicated function that tests a list of author names in order to decide whether \sameauthors or \AuthorList should be called.

\AuthorList This takes a list of author names in the form
\name{Jones, Sam}\name{Smith, John}...

and prints them in standard series form with the names uninverted, e.g., “Sam Jones and John Smith”, or “Sam Jones, John Smith, and James Taylor”.

\sameauthors This is a function of one argument. If you use the default set of \BibSpecs from the amsrefs, \sameauthors is applied to the author name for a given \bib command if it matches exactly the author name of the preceding \bib command. Change the definition of \sameauthors if you don’t want to get a bysame dash.

\bysame This is a horizontal rule of length 3 em. The default definition of \sameauthors prints \bysame instead of the author names.

\PrintEditorsA This is similar to \AuthorList but adds (ed.) following the editor name (or (eds.) if applicable).

\PrintEditorsB This is like \PrintEditorsA but puts parentheses around the entire list of editor names.

\Plural, \SingularPlural These are helper functions that allow you to conditionally print singular or plural forms such as (ed.) or (eds.) depending on the number of names in the current name list. The definition of \PrintEditorsA reads, in part,
... (ed\Plural{s}. ) ...

\ReviewList This is similar to \AuthorList but is used for printing (possibly multiple) MR numbers given in the review field.

\inicap This command applies initial capitalization to its argument.

\EnglishInitialCaps This command will call \inicap if and only if the language of the current reference is English. If English is the default language, you need to specify
language={German},

(for example) for non-English references to ensure that nothing will be initial-capped.

\footnote{There is one drawback: If you don’t want to get the prime symbol for \tprime from the cmsy font, you will need to redefine \tprime in some suitable way.}
\BibField This is for more complicated programming tasks such as may be necessary for some BibSpecs. It takes one argument, a field name, and yields the contents of that field for the current \bib entry.

\IfEmptyBibField If one writes
\IfEmptyBibField{isbn}{A}{B}

then the commands in A will be executed if the isbn field is empty, otherwise the commands in B.

\PrintEdition If a bibliography entry has
\Begin{\printedition}{2}

and the bib-spec used \PrintEdition to handle this field, then the edition information will be printed as “2nd ed.”—that is, the number is converted to cardinal form and “ed.” is added (taken from \editiontext).

\CardinalNumeric This provides the conversion to cardinal number form used by \PrintEdition.

\PrintDate, \PrintYear These functions convert a date in canonical form (ISO 8601) to the form required by the current bibliography style. You can get your preferred date form by redefining these functions or by changing your \BibSpec statements to use another function of your own devising. The original definition of \PrintDate adds parens (as for the year of a journal article in normal AMS style), whereas the \PrintYear function simply prints the year without any additional material (as for a book’s year of publication in normal AMS style).

\mdash, \textendash These are short forms for \textemdash and \textendash, recommended instead of the more usual --- and -- notation. From the textcmds package.

\cite, \cites, \citelist, \ycite, \ocite, etc. See the section on Citations.

et cetera ... [mjd,2002-01-03] See the dtx files for further possibilities that I have not managed to get properly documented yet!

12 Implementation

12.1 Overview

It will be a while yet before we get to any actual code. First we need to understand what the code needs to accomplish in order to provide the user interface described above in a way that is as compatible as possible with existing \TeX mechanisms.

Normal \TeX processing of cites

First \TeX pass Various commands are written to the aux file that are mostly used by Bib\TeX.
12. IMPLEMENTATION

1. A \cite{moo} command writes one line to the aux file: \citation{moo}. This indicates to Bib\TeX that it should include ‘moo’ in the list of cited items to be searched for. The \cite command then checks to see if \b@moo is defined: If it is, its contents are printed as the cite ‘number’; if it isn’t, \LaTeX gives an ‘Undefined citation’ warning and prints question marks instead.

2. A \bibliographystyle{har} command writes one line to the aux file: \bibstyle{har}. This indicates to Bib\TeX that it should use har.bst to determine the style for sorting and formatting the bib items.

3. A \bibliography{hij,klm,...} command writes one line to the aux file: \bibdata{hij,klm,...}. This indicates to Bib\TeX that it should look in hij.bib, klm.bib, ... for bibliographic data. The \bibliography also tries to input the .bbl file, but on the first pass it won’t exist yet.

On the first pass all \cite’s normally are reported as undefined because the .bbl file has not yet been created.

Bib\TeX pass For a document named xyz.tex, the command \texttt{bibtex xyz} is used to invoke Bib\TeX. It looks in xyz.aux to find the citation info written there by \LaTeX. For each \citation line, Bib\TeX searches for a corresponding entry in the specified .bib files and formats it. The entire list is then sorted in whatever way dictated by the bibliography style, and written out to the file xyz.bbl. This normally produces entries that look something like:


Second \LaTeX pass Now the .bbl file exists and contains some \bibitem commands. At \texttt{\begin{document}}, \LaTeX reads the aux file, hoping to find some \bibcite commands, but it will not find them until the next time around. \citation, \bibstyle, and \bibdata commands in the aux file are simply ignored by \LaTeX. Then \LaTeX proceeds to typeset the body of the document.

1. Instances of \cite still print question marks.
2. The \bibliography command causes \LaTeX to input xyz.bbl and typeset its contents.
3. A \bibitem{moo} command writes one line to the aux file: \bibcite {moo}{9}, where 9 is the current item number.
4. A \bibitem[Moody]{moo} command writes one line to the aux file: \bibcite{moo}{Moody}, using the supplied label instead of a number.

Third \LaTeX pass Now the aux file contains some \bibcite commands. Once again, \LaTeX reads the aux file when it reaches \texttt{\begin{document}}.

1. A \bibcite{moo}{Moody} causes \LaTeX to define \b@moo with ‘Moody’ as the replacement text.
2. If two \bibcite commands have the same citation key, \LaTeX{} gives a warning message. This happens at \texttt{\begin{document}}, during the reading of the aux file.

3. Instances of \cite in the body of the document will print the appropriate numbers obtained from the aux file.

4. If there are any \cite commands for which the aux file did not have a \bibcite command, \LaTeX{} will give an ‘Undefined citation’ warning. This often happens if the aux file is incomplete due to a \TeX{} error on the preceding pass.

12.2 How cites are processed by amsrefs

In order to support its additional features (e.g., author-year citations and the backref option), the \texttt{amsrefs} package stores additional information for each cite in the macro \texttt{\@whatever}. Instead of simply using the defined or undefined status of this macro to trigger the standard warnings, we add some boolean flags to allow us to discriminate more finely what the current situation is.

- Each time an item is cited in the body of the document, a backref entry is added to the info of that item. The backref info is the current page and section location. Section location is a bit hard to get right without better support from the documentclass. So we provide a hook to allow it to work better when the support is there.

- When a cite occurs, if the info is undefined then a warning is issued and the info structure is created. A \texttt{\citation} command and a \texttt{\citedest} command (providing backref info) are written to the aux file. Because the backref info includes page number, it has to be a non-immediate write. An undefined info structure would normally happen only on a first pass when no .aux file exists, or when a new cite is added. i.e., when the corresponding \texttt{\citation} command is not yet present in the aux file.

- When a citation command occurs in the aux file, it initializes the info structure if necessary, setting the “bib-info-present” flag to 0.

- When a \texttt{\citedest} command occurs in the aux file, it initializes the info structure if necessary—but this shouldn’t happen: if the corresponding \texttt{\citation} command did not already get processed, then something is wrong. So normally, the \texttt{\citedest} command merely needs to add its backref info to the existing info structure.

- When a \texttt{\bibcite} command occurs in the aux file, it will normally find that \texttt{\@whatever} is already defined, if the bibliography occurs after all the \cite commands. What it must do is fill in the appropriate blank slots in the info structure set up by a previous \texttt{\citation} command.

- The aux file is actually processed two times, once at the beginning of the document and once at the end. In the latter case, \texttt{\bibcite} should give a warning if the backref-list is empty, since that means there were no \cite commands for the given key.
12. IMPLEMENTATION

- When processing the bibliography: The \bib command needs to check if it is using a key that is already used by another \bib command.

We therefore have
\b@xyz -> \citesel 00{number}{year}{backref-list}
where the first number is 1 if there has already been another citation for the same key earlier in the document (some citation styles use abbreviated forms for all instances after the first), and the second number is 1 if the same key was already used by an earlier \bib command.

Because the backref-list often includes page number information, it cannot be built on the fly as we go along; instead we have to write the information to the aux file and read it in at the beginning of the next run.

If there was no \bibcite in the aux file for a given key, then the info is
\b@xyz -> \citesel 00{}{}{backref-list}
If there was neither \citation nor \bibcite in the aux file for a given key, then the \cite command should find that \b@xyz is undefined.

If the author-year option is in effect, the “number” contains the author names instead of a number.
\b@xyz -> \citesel 00{
  \name{Smith, J. K.}\name{Jones, T. J.}
}(...){...}
Full name information is included in the data because some citation styles give full names at the first citation and abbreviated forms for subsequent instances.

12.3 Data structures

The result of scanning the key/value pairs of a \bib command is an assignment statement for \rsk@toks. (Cf. the rkeyval package.) For example, consider the entry
\bib{miller83}{article}{
  author={Miller, G.},
  title={Eine Bemerkung zur Darstellung von Polynomen "{u}ber Verb"{a}nden},
  journal={J. Math. Sent.},
  volume={10},
  year={1983},
  pages={26\ndash 30},
}
The scanned result is to assign
\global\rsk@toks{%
  \append\name\bib'author{Miller, G.}%
  \DSK@def\bib'title{Eine Bemerkung zur Darstellung von Polynomen \"{u}ber Verb"{a}nden}%
  \DSK@def\bib'journal{J. Math. Sent.}%
  \DSK@def\bib'volume{10}%
  \DSK@def\bib'year{1983}%
  \DSK@def\bib'pages{26\ndash 30}%
}
The code in the last arg of \RestrictedSetKeys then invokes \bib@exec to do something with the value of \rsk@toks.
\bib@exec{miller83}{\the\rsk@toks}{\setbib@article}{}

Now the second-best thing to do with the value, if there weren’t any memory limits to worry about, might be to store it directly into \b@miller83, reading the bibliography data at the beginning of the document. Then cite commands would have access to every field in the reference, and the bibliography could be printed wherever desired by a suitable dumping command. Since this could potentially use up a rather large chunk of main mem, depending on the size of the bibliography, and since versions of \TeX without dynamic memory allocation are still in common use, and the fatal error given by \TeX when it runs out of memory is unlikely in this case to provide much of a clue to the true source of the problem, it seems dangerous to make this behavior the default.

The first, best alternative, however, might be to have cites and bibliography filled in dynamically by the editing software during the writing process, and leave to Emacs Ref\TeX package only some final consistency checks. The Emacs Ref\TeX package is a good example of this sort of thing.

**Associating properties with field values**

The data example given above doesn’t tell the whole story, because it is in fact sometimes necessary to attach properties to a particular value of a field. The premier example of this is language. The title of a bibliography item, above all, may need to have complicated bits of information supplied about its language: not only whether it is an English or German or Spanish title—which affects hyphenation and capitalization—but for a Russian title (say) which input encoding does it use (since there are several in common use), or whether it is an ASCII transliteration (and if so, what kind), or whether it is a translation into, say, English, of the original title. Arabic and Chinese titles can be equally problematic.

It would be unnatural to plunk this information directly into the contents of the field, but it needs to be tied to the field in some way by our data structures. The simplistic approach would be to use an auxiliary field “extra-foo” for each field foo. But this tends to burn up a lot of \TeX control sequence names, since the set of field names is quasi-unbounded. So instead I have decided that if a bibliography item has extra data for any of its fields, all such extra data will be stored in another field named “aux”, with embedded tags to indicate which field each piece of the aux field is associated with. The extra bits can be extracted on demand using standard techniques, and the primary value of each field is not burdened with any attachments, so that comparisons or scanning of the field contents can remain as simple as possible.

Thus in practice there is at least one bit of auxiliary information in every bib item, and our previous example would have the title language indicated:

\DSK@def\bib'title{Eine Bemerkung zur Darstellung von Polynomen \verb"{u}ber Verb\verb{"a}nden}\
\@append\bib'title\bib'aux{\selectlanguage{german}}%
12. IMPLEMENTATION

That provides a half-way decent \TeX solution to the problem of storing field properties. The problem of how the user should specify field properties in the \bib file is another story.

12.4 Preliminaries

\langle * pkg \rangle

Standard declaration of package name and date.
\NeedsTeXFormat{LaTeX2e}[1995/12/01]

Backward handling for beta version. There should be a better way of doing this...

\@ifpackagewith{amsrefs}{beta}{{% \
\expandafter\edef\csname opt@amsrbeta.sty\endcsname
{\@ptionlist{amsrefs.sty}}% \
\def\@currname{amsrbeta}% \
\expandafter\let\csname amsrbeta.sty-h@@k\endcsname\@empty \
\def\@tempa{\input{amsrbeta.sty}\endinput}% 
}}

\@tempa
\@ifundefined{NormalCatcodes}{{% 
\RequirePackage{pcatcode}\relax\PushCatcodes{NormalCatcodes}
}}{}
\ProvidesPackage{amsrefs}[2002/01/03 v1.12]

%% WARNING WARNING WARNING: Catcode of apostrophe ’ is letter
%% throughout this file.
\catcode’=11 % letter

12.5 Some general tools

Many of these useful functions are found also in AMS documentclasses.

Some short forms to reduce code clutter.
\let\@xp=\expandafter
\let\@nx=\noexpand

Some frequently used tests for empty arguments.
\long\def\@ifempty#1{\@xifempty#1@@..\@nil}
\long\def\@xifempty#1#2@#3#4#5\@nil{\ifx#3#4\@xp\@firstoftwo\else\@xp\@secondoftwo\fi}
\long\def\@ifnotempty#1{\@ifempty{#1}{} }

A really old version of \LaTeX might still choke on \newtoks if it is written directly in the false branch of a conditional.
\@ifundefined{@emptytoks}{\csname newtoks\endcsname @emptytoks}{}

Not in the \LaTeX kernel yet.
\long\def\@ifemptythree#1#2#3{} \long\def\@onilgobble#1@#nil{} 

Test for a trailing option marked by a star. Usage:
\newcommand{\blub}[1]{\star@{\blubaux{#1}}{default}}
Arg 1 of \texttt{\textbackslash star@} is the code to be run, arg 2 is the default value of the option (could be empty). If arg 1 is \texttt{\textbackslash moo}, this test discards a star and expands to \texttt{\textbackslash moo} if a star is found, or expands to \texttt{\textbackslash moo\{#2\}} if not. As the example shows, arg 1 need not be a single token.

\begin{verbatim}
\let\star@char=* \def\star@#1#2{% \def\star@a##1{#1}\def\star@b{#1{#2}}% \futurelet\@let@token\star@test } \def\star@test{\ifx\@let@token\star@char \let\star@b\star@a\fi \star@b}
\end{verbatim}

Please note: If there is a space before the star, then the star is not treated as an option char.

Why use a star? I dunno, it’s one of the characters least likely to suffer from catcode changes I guess, since it’s already part of standard L\TeX command syntax.

Why not just put the star at the beginning in the usual way? I dunno, I guess it seemed to me that the lack of a trailing option feature was a deficiency in current \LaTeX and could be given an experimental implementation in a package like this without any adverse effect on existing documents.

\begin{verbatim}
\def\amsrefs@warning{\PackageWarning{amsrefs}}
\end{verbatim}

### 12.6 Package Options

We call the \texttt{ifoption} package to facilitate some option tests.

\begin{verbatim}
\RequirePackage{ifoption}[2000/02/15]
\end{verbatim}

Options beginning with \texttt{*} require an additional \LaTeX run in order to take proper effect. We use \texttt{*} as the marker because it is one of the few characters that are pretty much guaranteed to be free of catcode changes (because \texttt{\@ifstar} relies on it).

The standard \texttt{unsrt} biblio style prints the bibliography in citation order, not sorted by author names.

\begin{verbatim}
\DeclareExclusiveOptions{*sorted,*citation-order}
\end{verbatim}

The \texttt{alphabetic} option corresponds to the standard \texttt{alpha} biblio style with labels like \texttt{Knu66} (three letters from name plus two digits of year). Maybe should provide an alias \texttt{LlYY} for this option. Numeric is the default since it is commoner in AMS publications.

\begin{verbatim}
\DeclareExclusiveOptions{alphabetic,author-year,numeric}
\end{verbatim}

The standard \texttt{abbrv} bibliography style uses abbreviations for month names and journal names, and first names of people are abbreviated to their initials. Since the second test bibliography that I tested with had unabbreviated month names but abbreviated journal names, perhaps it is a good idea to let these choices be specified separately.

\begin{verbatim}
\DeclareBooleanOption{short-journals}
\DeclareBooleanOption{short-months}
\DeclareBooleanOption{initials}
\end{verbatim}
12. IMPLEMENTATION

In the bibliography, if a title or something is enclosed in quotes, should the closing quotes go inside the punctuation (logical position) rather than outside (traditional)? These options give you a choice.
\DeclareExclusiveOptions{traditional-quotes,logical-quotes}

A sequence of cites will be sorted and ranges of length three or greater will be compressed if these options so indicate.
\DeclareExclusiveOptions{sorted-cites,non-sorted-cites}
\DeclareExclusiveOptions{non-compressed-cites,compressed-cites}

In the bibliography, print page numbers showing where each entry was cited.
\DeclareBooleanOption{backrefs}
\def\amsrefs@verbose{01}
\DeclareOption{?}{\def\amsrefs@verbose{00}}

This option can be used by later releases as a sign that fall-back adaptations need to be done.
\DeclareOption{beta}{}
\DeclareBooleanOption{jpa}
\ExecuteOptions{numeric,*sorted,traditional-quotes,sorted-cites,compressed-cites}
\ProcessOptions\relax
\ProcessExclusiveOptions

12.7 Auxiliary packages

Now, if these other packages make use of the \texttt{pcatcode} package like they should, then we don’t need to make any fuss here about the special catcode of ‘. Just load the packages.
\RequirePackage{rkeyval}[2000/02/02]
\RequirePackage{inicap}[2000/02/02]

12.8 The \texttt{biblist} environment

The \texttt{biblist} environment can be used with a section or chapter heading.

\texttt{biblist} Arg 1 was intended for key-value options, but support for that is not done yet, instead it is just handled as additional list setup code. [mjd,2000/03/07]
\newenvironment{biblist}{1}[1][]{% 
\normalfont \footnotesize \labelsep .5em\relax 
\list{\BibLabel}{% 
\resetbiblist[00]%% 
\usecounter{bib}%% 
#1\relax 
}% 
\sloppy
Discourage page breaks within bibliography entries and disable them completely for entries that are less than four lines long.
\interlinepenalty\@m \clubpenalty\@M \widowpenalty\clubpenalty 
\frenchspacing
\ResetCapSFCodes
\let\@bibdef=normalbibdef
}\%

Change error for empty list (no items) to warning, to allow authors to leave their bibliography temporarily empty during writing:
\def\@noitemerr{\latexwarning{Empty bibliography list}}%
\endlist
}

In case this is undefined sometimes.
\def\CurrentBib{??}
\newcommand{\BibLabel}{%
[\hyper@anchorstart{cite.\CurrentBib}\relax\thebib\hyper@anchorend]%
}
\newcommand{\resetbiblist}[1]{%
\settowidth{\labelwidth}{\def\thebib{#1}\BibLabel}%
\leftmargin{\labelwidth}
\ifdim\labelwidth=\z@%
\leftmargin=1em \itemindent=-\leftmargin
\else
\advance\leftmargin\labelsep
\fi
\}

12.9 Printing a bibliography entry
These two things (\bib@start and \bib@end) used to be useful, but I haven’t entirely made up my mind to get rid of them yet. [mjd,2001-11-28]
\def\bib@start{\begingroup}

Instead of being handled by \bib@end, ending punctuation is normally handled via the ‘transition’ field qv.
\def\bib@end{\relax
@xp\PrintBackRefs@xp\CurrentBib%\par\endgroup
}

For list macros such as \name, we want them to be unexpandable except when special processing is done.
\let\name=\let\isbn=\let\review=\
\newcommand{\partialbib}[1]{%
\begingroup
\RestrictedSetKeys{\let\rsk@nocomma=\amsrefs@nocomma}{bib}%
{\bib@partial{}{\the\rsk@toks}{\endgroup}{#1}%%
And here are the predefined key names. You could always add some more if you needed them. Only worry is about compatibility if you want to share your data with other people.

\let\bib@reset\@empty % this gets modified by the following statements.

First the fields that could be repeated more than once in a single entry. Maybe publisher should be allowed to repeat also, for co-published works. But then need to worry about the address handling.

\DefineAdditiveKey{bib}{author}{\name}
\DefineAdditiveKey{bib}{editor}{\name}
\DefineAdditiveKey{bib}{isbn}{\isbn}
\DefineAdditiveKey{bib}{review}{\review}
\DefineAdditiveKey{bib}{part}{\partialbib}
\DefineSimpleKey{bib}{title}
\DefineSimpleKey{bib}{subtitle}
\DefineSimpleKey{bib}{booktitle}
\DefineSimpleKey{bib}{series}
\DefineSimpleKey{bib}{conference}
\DefineSimpleKey{bib}{publisher}
\DefineSimpleKey{bib}{organization}
\DefineSimpleKey{bib}{institution}
\DefineSimpleKey{bib}{address}
\DefineSimpleKey{bib}{place}
\DefineSimpleKey{bib}{year}
\DefineSimpleKey{bib}{date}
\DefineSimpleKey{bib}{journal}
\DefineSimpleKey{bib}{fulljournal}
\DefineSimpleKey{bib}{issn}
\DefineSimpleKey{bib}{volume}
\DefineSimpleKey{bib}{number}
\DefineSimpleKey{bib}{pages}
\DefineSimpleKey{bib}{doi}
\DefineSimpleKey{bib}{status}
\DefineSimpleKey{bib}{eprint}
\DefineSimpleKey{bib}{preprint}
\DefineSimpleKey{bib}{language}
\DefineSimpleKey{bib}{note}
\DefineSimpleKey{bib}{url}
\DefineSimpleKey{bib}{xref}
\DefineSimpleKey{bib}{label}
\DefineSimpleKey{bib}{type}
\DefineSimpleKey{bib}{edition}
\DefineSimpleKey{bib}{setup}
\DefineSimpleKey{bib}{name}

The transition key is used when we want to insert punctuation or other material at a given point in the sequence, even if all following fields happen to be empty. The key appears to have a non-empty value but its value is effectively empty.
\DefineDummyKey{bib}{transition}

This provides access for BibSpec functions to individual fields.
\newcommand{\BibField}{[1]\csname bib'\#1\endcsname}
\newcommand{\IfEmptyBibField}{\rkvIfEmpty{bib}}

Here is the sequence of macro calls when \bib runs in a normal way to print a bib item.
\bib{abk89}{proceedings.article}{...}
\bib ->@ifstar and call \BibItem

\BibItem{abk89}{proceedings.article}{...}
\BibItem #1#2 -> read key, type and call \@bibdef
#1<-abk89
#2<-proceedings.article

@bibdef \setbib@proceedings.article\space\{proceedings.article\}\{abk89\}
@bibdef #1#2#3 -> warn if #1 is undefined; call RestrictedSetKeys
#1<-\setbib@proceedings.article
#2<-proceedings.article
#3<-abk89

\RestrictedSetKeys {...}{bib}{\bib@exec ...}{...}
Read three args and then futurelet to start key scanning.
Third arg is code to call after scanning is finished.
\RestrictedSetKeys [#1]#2#3 -> ... (futurelet to start key scanning)
#1<-\amsrefs@nocomma
#2<-bib
#3<-\bib@exec\{abk89\}\{the \rsk@toks\}\{\setbib@proceedings.article\}\endgroup
\rsk@finish ->\bib@exec\{abk89\}\{the \rsk@toks\}\{
\bib@exec #1#2#3->
#1<-abk89
#2<-the \rsk@toks
#3<-\setbib@proceedings.article

Here \bib@exec = \bib@print: Start up a group, process the assignments found in \rsk@toks, run \bib@cite\{KEY\}, then run \setbib@moo

Here is where the rubber hits the road.
\newcommand{\bib}{%
\begingroup
@ifstar{\let\bib@exec\bib@store
\let@bibdef\normal@bibdef
\BibItem}{%}
\{\BibItem}%
%
}% #1 key #2 type
\newcommand{\BibItem}{[2]{%}
\def@tempa#1%
12. IMPLEMENTATION

Use a standard \LaTeX counter for numbering biblio items.

\newcounter{bib}

The standard $\textbackslash \texttt{NoCommaWarning}$ is not entirely apt here, so we make a slightly different version.

\def\amsrefs\nocomma{\amsrefs\warning{Missing comma: some text might print incorrectly}}

\% #1 \setbib@article #2 article #3 key
\def\normal@bibdef#1#2#3{\%\
  \ifx\relax#1\
    \PackageError{amsrefs}{Undefined entry type: #2}\null\fi\let#1\setbib@misc\fi\RestrictedSetKeys{\let$rsk\nocomma\amsrefs\nocomma\bib\%\
  \{\bib@exec\texttt{\{#3\}\{the$rsk\toks\}[#1]\endgroup\%}}
\let\@bibdef\normal@bibdef

This is a variation that ignores anything not having a known citation key.

\% #1 \setbib@article #2 article #3 key
\def\selective@bibdef#1#2#3{\%
  \def\selbibdef@#1\csname b@#3\endcsname{#1}{#2}{#3}\
}\def\selbibdef@#1\csname b@#3\endcsname{#1}{#2}{#3}\
  \ifx1#2\let\@tempa\copy@bibdef\fi\fi\else \def\selbibdef@#1\csname b@#3\endcsname{#1}{#2}{#3}\fi\def\copy@bibdef#1#2#3{\%
  \ifx\relax#1\
    \PackageError{amsrefs}{Undefined entry type: #2}\null\fi\let#1\setbib@misc\fi\@open@bbl@file\bbl@write{\string\bib{#3}{#2}
    \string{\iffalse}\fi}\
    \% empty arg means use current value of $rsk\set$ instead of overriding it.\RestrictedSetKeys{\let$rsk\nocomma\amsrefs\nocomma\glocal\let$rsk\set\bbl@copy\%\
  \{\bbl@write{\iffalse{\fi\string}\%}}

\edef\@tempb{\@nx\@bibdef\@xp\@nx\csname setbib@#2\endcsname{#2}{\{\@xp\strip@prefix\meaning\@tempa}\%}
}\@tempb
\let\biblio@list\@empty
12. IMPLEMENTATION

\bib@@data % execute definitions locally
\bib@resolve@xrefs % ??
\bib@field@patches
\def\@currentlabel{\@nx\citex{\bib@year}{\bib@author}\
\global\@xp\let\csname bi@#1\endcsname\bib@@data
\@xp\g@addto@macro\@xp\all@bibs\csname bi@#1\endcsname
\endgroup}

If one of the names in a name list is discovered to be an abbreviation, resolve it. I guess this requires rebuilding the list.

\def\BibAbbrevWarning#1{\amsrefs@warning{Abbreviation '#1' undefined}}
\def\xref@check@aa#1{\def\@tempa{#1}\lowercase{\def\@tempb{#1}}\
\ifx\@tempa\@tempb
\ifx\@tempa\@empty
\toks\@xp{\the\toks@ \name{}}\
\else
\text{#1}\
\fi
\else
\toks\@xp{\the\toks@ \name{#1 \unskip}}\
\fi
}

\def\BibAbbrevWarning#1{\amsrefs@warning{Abbreviation '#1' undefined}}

\def\xref@check@ab#1{\let\bib'name\@empty
\csname bi@#1\endcsname\ifx\@empty\bib'name
\@temptokena{#1}\
\else\
\@ifundefined{bi@#1}{{\BibAbbrevWarning{#1}}}{{\xref@check@ab{#1}}}\
}\fi
\def\xref@check@aa#1{\def\@tempa{#1}\lowercase{\def\@tempb{#1}}\
\ifx\@tempa\@tempb
\ifx\@tempa\@empty
\toks\@xp{\the\toks@ \name{}}\
\else
\text{#1}\
\fi
\else
\toks\@xp{\the\toks@ \name{#1}}\
\fi
}

I think the extra space and unskip helps to avoid a certain kind of error.

[mjd,2001-12-27]
\toks\@xp{\the\toks@ \name{#1 \unskip}}\
\ifx\@tempa\@tempb
\ifx\@tempa\@empty
\toks\@xp{\the\toks@ \name{}}\
\else
\toks\@xp{\the\toks@ \name{#1}}\
\fi
\fi
\def\xref@check@ab#1{\let\bib'name\@empty
\csname bi@#1\endcsname\ifx\@empty\bib'name
\@emptokena{#1}\
\else
\toks\@xp{\the\toks@ \name{#1}}\
\fi

If any title occurs in an xref-d item, assume that it is a book title. This might not always be the best assumption? Let’s see how it goes though. [mjd,2001-12-11]
12. IMPLEMENTATION

\xref@check@a\bib'author \xref@check@a\bib'editor
\xref@check@b\bib'journal \xref@check@b\bib'publisher

\def\bib@print#1#2#3{\%
#1 citation key, #2 \the\rsk@toks, #3 \setbib@moo
\bib@start
\bib@reset
\let\setbib@@#3%\%
\edef\bib@@data{#2}%%
\bib@@data % execute definitions locally
\bib@resolve@xrefs % modifies \bib@@data
\bib@field@patches
\bib'setup
\bib@title\kern\@ne sp\relax
\ifx\bib@booktitle\@empty
\else \let\bib@title\@empty\fi
\let\bib@author\@empty
\setbib@@
\bib@end
\}

\def\bib@partial#1#2#3{\%
#1 citation key, #2 \the\rsk@toks, #3 \setbib@moo
\begingroup
\bib@reset \let\bib@title\@empty \let\bib@author\@empty
\edef\bib@@data{#2}%%
\bib@@data % execute definitions locally
\bib@field@patches
\bib'setup
\bib@end
\endgroup
\}

This is just an alias for convenience.
\let\bib@exec\bib@bibcite
\AtBeginDocument{\let\bib@exec\bib@print}

When \bib@cite is called, author name and year are available in \bib@author and \bib@year.

\IfOption{author-year}{%\%
\def\cite@label{%\%
\ifx\bib@author\@empty \bib@editor \else \bib@author \fi
}\
\def\cite@label{@currentlabel}%
}%

\IfOption{alphabetic}{%\%
\def\alpha@label{%
}\
}%
For the citation key we want to check if it is already defined. But there is a slight problem. There is already one control sequence in use for each bibliography entry, to store the number or the author/year information needed by \cite. If we introduce another control sequence to check whether a particular cite is multiply defined, then we double the number of control sequences used. For a large bibliography in a book this is fairly serious. This is addressed by using a \citesel function.

\def\b@citesel#1#2#3#4#5#6#7{\if#1#4\relax \DuplicateBibKeyWarning \else \protected@edef\@tempa{% \edef@nx@citese#11{%@nx\cite@label}{%\@nx\cite@year}{#3}{\@nx\cite@authoryear}{#1}{#7}}}% \@tempa \fi}

\def\DuplicateBibKeyWarning{%
Because duplicate bibs are caught immediately, we don’t need \bibcite to run @testdef.
\AtEndDocument{\let\amsRefs@bibcite\@gobbletwo}

12.10 Some abbreviations

The commands \DefineName, \DefinePublisher, and \DefineJournal are provided to make abbreviations a little easier.

\newcommand{\DefineName}[2]{\begingroup \let\bib@exec\bib@store \BibItem{#1}{person}{name={#2},}}
\newcommand{\DefineJournal}[2]{\begingroup \let\bib@exec\bib@store \BibItem{#1}{periodical}{journal={#2},}}
\newcommand{\DefinePublisher}[3]{\begingroup \let\bib@exec\bib@store \BibItem{#1}{publisher}{% publisher={#2},% place={#3},}}

This is what the standard book.cls has for the bibliography title:
\newenvironment{thebibliography}[1]{%}
\chapter*{\bibname}
\@mkboth{\MakeUppercase\bibname}{\MakeUppercase\bibname}
\list{\@biblabel{\@arabic\c@enumiv}}%
\newcommand{\printbibliography}[1][]{\bibname{%}
\begin{bibdiv}[#1]
We need to take a little extra trouble here to pre-expand the \bibname. An extremely crude way of carrying forward the current font series (basically, bold or non-bold) for the next punctuation object. This needs work I guess. [mjd, 2001-12-11] As things currently stand, when you specify a font in the punctuation argument of a bibspec line, it interferes with the stuff in \SwapBreak that prevents redundant punctuation.

\let\bib@lastseries@empty
\def\CarryFontSeries{\global\let\bib@lastseries\@series}
\def\SameFontSeries#1{\protect\CarryFontSeries}

\let\bib@lastseries\@empty
\ifx\bib@lastseries\@empty
  \else \fontseries\bib@lastseries\selectfont
\fi
#1\endgroup
The arg has to start with a simple punctuation character in order for this to work as intended.

\def\SwapBreak#1{\%\@tempcnta\@MM\ifhmode\unskip\ifnum\lastpenalty=\z@\else \@tempcnta\lastpenalty \unpenalty\fi\fi\toks@\SameFontSeries{#1}\%\edef\@tempa{\@nx\deferredquotes\logical\ifnum\lastkern=\@ne\else \the\toks@\fi\ifnum\@tempcnta=\@MM\else \penalty\number\@tempcnta\space\fi\ifnum\lastkern=\@ne \ignorespaces\fi}\@tempa}\newcount\series@index

The \PrintSeries command prints a list of objects in series form. The essential idea is to produce something like “A, B, and C” when we are given three elements A B C, with suitable variations in the punctuation and other intervening material if the number of elements is less or more than three.

To generalize this process, we envision \PrintSeries being called as \PrintSeries{i0}{i1}{i2}{i3}{i4}{\do{A}\do{B}...}
where i0, ..., i4 are material to be interpolated and the last arg is a list of indeterminate length where each element consists of a macro and its argument.

The output, depending on the number of elements, will be

\begin{verbatim}
i0 A i4 % 1 element
i0 A i1 B i4 % 2 elements
i0 A i2 B i3 C i4 % 3 elements
i0 A i2 B i2 C i2 ... X i2 Y i3 Z i4 % 26 elements
\end{verbatim}

That is the simple explanation but in practice there are some additional complications. What if user-supplied line breaks have to be supported at the boundaries between elements? What if in addition to adding material between
elements we also want to apply some handy function to each element (e.g., \textsc{...})? Even worse, what if we want the function to be different depending on the position of the element in the list?? Indeed if this did not happen to be the case with the current application I would not have gone to the extra trouble of supportint it. But if it must be so, then the output that we need from a list \do{A}\do{B}... is
\begin{verbatim}
 f0{A} f0{A} p1 i1 f1{B} f0{A} p2 i2 f2{B} p3 i3 f3{B}
\end{verbatim}
and so on, where
\begin{itemize}
  \item $f_n$ is a macro taking one argument,
  \item $p_n$ is punctuation—material that must precede a line break if one occurs at this boundary,
  \item $i_n$ other interpolated material, as before.
\end{itemize}
To reduce the number of distinct required objects we decree that each element will get braces wrapped around it as a matter of course; then it is possible for $f_1, f_2, f_3$ to be assimilated into the tail end of $i_1, i_2, i_3$.

\begin{verbatim}
\def\PrintSeries#1#2#3#4#5#6#7#8{\%
  \begingroup
    \def\series@add@a{#2}\
    \def\series@add@b\{\SwapBreak{#3}#4\}\
    \def\series@add@c\{\SwapBreak{#5}#6\}\
    \def\series@add@d\{\SwapBreak{#7}#8\}\
    \PrintSeries@a{#1}\
  \endgroup
}
\end{verbatim}

\texttt{Plural} takes one argument and prints it if there were two or more elements in the current list. So, to get “editors” instead of “editor” after printing a list of editor names, write editor\texttt{Plural}s.

\texttt{SingularPlural} takes two arguments and prints the first if there was only one element, otherwise prints the second arg.

\begin{verbatim}
\newcommand\{\SingularPlural\}[2]{#1}
\newcommand\{\Plural\}{\SingularPlural{}\
\end{verbatim}

For \texttt{PrintSeries@a} the first arg is the iterator function present in the list which is arg 3. Args 2 and 4 are extra material to be added before and after the list that may require the use of \texttt{Plural} or \texttt{SingularPlural}.

\begin{verbatim}
\def\PrintSeries@a#1#2#3#4{\%
  \series@index\z@ \def\#1{\advance\series@index\@one \@gobble}\
  \relax
  \ifnum\series@index=\@one \let\SingularPlural@firstoftwo\relax
  \else \let\SingularPlural@secondoftwo\relax
  \fi
  \chardef\series@total=\series@index \series@index=\z@\%
  \let\#1\series@add\
  \relax
  \endgroup
\end{verbatim}
This is the inner function called by \PrintSeries that carefully distributes all the material stored previously in \series@add@... macros.

```latex
\def\series@add#1{\protect\@tempa\relax
\ifcase\series@index
\or % material before name 1
  \let\@tempa\series@add@a
\or % material before name 2
  \ifnum\series@total<\thr@@ \let\@tempa\series@add@b
    \else \let\@tempa\series@add@c
\else % material before name 3, 4, 5, ...
    \ifnum\series@index<\series@total \let\@tempa\series@add@c
      \else \let\@tempa\series@add@d
    \fi
  \fi
\fi
\@tempa{#1}\relax}
```

The function \bib@append prints the value of a field, together with associated punctuation and font changes, unless the value is empty. Arg 1 is punctuation (that may need to be swapped with a preceding line break), arg 2 gives the space to be added after the punctuation, and possibly a function to be applied to the contents of arg 3, which is a macro containing the field value. So if we have \moo and \bib'pages, from pages={21
\ndash 44}, then we want to arrange to call \moo{21\ndash 44}

We don't want to simply call \moo\bib'bar because that makes it rather difficult for \moo to look at the contents of \bib@bar.

```latex
\def\bib@append#1#2#3{\relax\ifx\@empty#3\relax
\else
  \ifx#3\relax\errmessage{#3=\relax}\fi
  \if\empty{#1}\%
    \@emptokena{\ifnum\lastkern=\@ne\ignorespaces\fi #2}\%
  \}%
\else
  \edef\@tempa{\the\@emptokena{\the\toks@}}
  \@xp\@tempa
\fi
\let\endbracket\@empty
\let\bracket@stack\@empty
\def\push@bracket#1{%}
```
The \texttt{parenthesize} function adds parens around its argument, calling \texttt{upn} to prevent italic parens from being used. But the default definition of \texttt{upn} is a no-op, meaning that this refinement lies dormant unless the \texttt{upref} package or other activation is done. (Probably better done via special fonts, anyway.)

\providecommand{\upn}[1]{#1}
\newcommand{\parenthesize}[1]{\leavevmode\push@bracket)\upn{(}#1\pop@bracket}

\let\deferredquotes\@empty
\IfOption{logical-quotes}{\def\deferredquoteslogical{\deferredquotes}}\
\let\deferredquoteslogical\relax
\newcommand{\bibquotes}[1]{\textquotedblleft#1\textquotedblright}
\providecommand{\mdash}{	extemdash}
\providecommand{\ndash}{	extendash}

For older versions of some AMS documentclasses, this patch is needed.

\providecommand{\MRhref}{\texttt{}}

The \texttt{@addpunct} function is defined by AMS documentclasses and the \texttt{amsgen} package. But if we find it undefined we had better define it.

\Ifundef{@addpunct}{\def{@addpunct}{%\relax\ifhmode\unskip\ifnum\spacefactor>\@m \else#1\fi\fi}}{}
\def{\ndash}{\textendash}
\providecommand{\ResetCapSFCodes}{%\count@='\A\def{\tempa}{\sfcodetext{1003}\sfcodetext{1002}\sfcodetext{1001}\relax}}
\providecommand{\nopunct}{\spacefactor 1004\relax}
\providecommand{\ResetCapSFCodes}{%\count@='\A\def{\tempa}{\sfcodetext{1003}\sfcodetext{1002}\sfcodetext{1001}\relax}}
The \bibspec@scan function scans one field spec from the second arg of \BibSpec. Each field spec has the form

+{punctuation}{prelim material}{field name}

We provide a certain amount of error recovery by reading an extra argument and checking if it is a plus sign as expected. If the plus sign is missing for the next field, we skip arg 4 (printing the tail of it if it is multiple tokens), give an error message, and strive to continue anyway. This might produce hilarious nonsense—but that could be, after all, not such a bad thing once in a while.

\def\bibspec@scan#1#2#3#4{\toks@\@xp{\the\toks@\bib@append{#1}{#2}}\edef\@tempa{\toks@\@xp\@nx\csname bib'#3\endcsname}\@tempa\ifx\@empty#4\@xp\@gobble % end the recursion\else \ifx +#4\else\bibspec@scan@error\fi\fi\bibspec@scan}

Accumulate spec material in \toks@, then define \setbib@moo from it.

\newcommand{\BibSpec}[2]{\toks@\@emptytoks\@ifnotempty{#2}{\@ifnextchar{+}{\@xp\bibspec@scan\@gobble}{\bibspec@scan}#2}\@empty\@empty\@empty\@empty\@xp\edef\csname setbib@#1\endcsname{\the\toks@}}

I made this a \def rather than a \let because using \let would make \BibSpecAlias statements order-sensitive in a way that seems frequently to be a stumbling block to unwary package writers. But then we should probably do at least the simplest kind of infinite loop check.

\newcommand{\BibSpecAlias}[2]{\@xp\def\@xp\@tempa\@xp{\csname setbib@#1\@xp\endcsname}\@xp\ifx\csname setbib@#2\endcsname\@tempa\PackageError{amsrefs}{Mirror alias #1->#2 not allowed (infinite loop)}\@ehc\else\@xp\def\csname setbib@#1\@xp{\csname setbib@#2\endcsname}\@xp{\csname setbib@#2\endcsname}\@tempa\fi}
12.11 Name handling

When dealing with author or editor names in a \BibSpec, it is not easy to design some way of specifying the desired style that is flexible enough to handle the multitude of different styles needed in practice. As a first level of indirection, we expect that the \BibSpec will typically start by applying a function such as \PrintAuthors to the list of names. At the point of application, the name list will have the form

\name{firstauthor}\name{secondauthor}...

Because the formatting may well depend on the number of names that are present, we first run a \NameSetup macro that looks at the number of names and prepares a list of functions to be applied to each name in the list. This method allows us to specify the internal formatting of an author name with the same kind of specifications as are used in a general \BibSpec.

\newcommand{\PrintNames}[4]{% 
  \begingroup \NameSetup{#1}{#4}% 
  #2#3\relax 
  \edef\@tempa{\seriesdefault}\edef\@tempb{\f@series}% 
  \ifx\@tempa\@tempb \else \CarryFontSeries \fi 
  \endgroup 
}

\providecommand{\bysame}{\leavevmode\hbox to3em{\hrulefill}\thinspace} 
\newcommand{\sameauthors}[1]{\bysame} 
\def\PreviousAuthors{\relax} 
\newcommand{\AuthorList}[1]{\PrintNames{author}{}{}{#1}}% 
\newcommand{\PrintAuthors}[1]{% 
  \ifx\PreviousAuthors{\bib@author} \begingroup \aftergroup\sameauthors \let\AuthorList\endgroup \else \gdef\PreviousAuthors{\bib@author} \let\AuthorList\bib@name@init \fi \AuthorList{#1} % 
}

When specifying the format of author names, we distinguish four cases: first author, co-author, middle author, and last author. By breaking down the specifications to this level, we make it easier to handle special requirements such as inverting the first author’s name but printing the remaining names in normal order.

\BibSpec{firstauthor}{ 
  +{}{given} 
  +{}{surname} 
  +{}{jr}
The Chicago Manual of Style suggests that it is slightly better not to italicize ‘et al’ and some other extremely common abbreviations inherited from Latin. (Compare ‘etc’.)

\newcommand{\etaltext}{et al.}
\newcommand{\SubEtal}[1]{\etaltext}

Some “et alii” requirements can also be handled this way, e.g., if two authors should be printed as “A and B” but three or more should be printed as “A et al.”, it could be done readily enough by changing only the specs for middleauthor and lastauthor:

\BibSpec{middleauthor}{}
\BibSpec{lastauthor}{}

For more complicated et alii rules, well, just throw your brain into gear and think of something clever. I think you will find enough points of attack to make it possible one way or another.

When we consider editor names we have to think about some further complications. First, for the case of a book where editor names are listed in place of author names, just copy the same style with a bit of added text at the end.

\newcommand{\PrintEditorsA}[1]{\PrintNames{editor}{\bfseries}{ (ed\Plural{s}.)}{#1}}
\newcommand{\PrintEditorsB}[1]{\PrintEditorsA[1]}
So what does \NameSetup do anyway (you may ask)?
\def\NameSetup{\%
\exp\NameSetup@a
\csname setbib@first#1\exp\endcsname
\csname setbib@co#1\exp\endcsname
\csname setbib@middle#1\exp\endcsname
\csname setbib@last#1\endcsname
}%
\NameSetup@a first builds a queue consisting of, e.g.,
\setbib@firstauthor \setbib@middleauthor \setbib@middleauthor ...
\setbib@lastauthor
and then checks to see if \name@queue = \setbib@firstauthor \setbib@lastauthor,
because that means there were only two authors; and then we substitute
\setbib@coauthor for the second token.
\def\NameSetup@a\#1\#2\#3\#4\#5{\%
\def\name##1{\@nx#3}\
\edef\name@queue{\@nx#1\@gobblefour#5\@nx#4\@empty\@empty}\
\def\@tempa{\#1}\
\ifx\name@queue\@tempa \let\SingularPlural\@firstoftwo \else \let\SingularPlural\@secondoftwo \fi
\def\@tempa{\#1\#4}\
\ifx\name@queue\@tempa \def\name@queue{\#1\#2}\fi
\let\name\doName
}
Implementación

\def\doName#1{% 
  \iffalse\fi \exp\get@namer\name@queue\% 
  \name@split#1,{},\@nil \setbib@@name 
}\fi

Short for get-name-handler.
\def\get@namer#1{\let\setbib@@name#1% 
  \@xp\def\@xp\name@queue\@xp{\iffalse}\fi 
}\fi

Some tests of various author name forms.

\begin{verbatim}
()->[,] [\@] [\[\]]
(Abiteboul)->[A] [biteboul] [\@]
(S. Abiteboul)->[S] [. Abiteboul] [\@]
(S. T. Abiteboul)->[S] [. T. Abiteboul] [\@]
(S.T. Abiteboul)->[S] [.T. Abiteboul] [\@]
(Steve Abiteboul)->[S] [teve Abiteboul] [\@]
(Abiteboul, S.)->[A] [biteboul] [S]
(Abiteboul, S. T.)->[A] [biteboul] [S]
(Abiteboul, S.T.)->[A] [biteboul] [S]
(P. Kanellakis)->[P] [. Kanellakis] [\@]
(K. I. Be\u{i}dar)->[K] [. I. Be\u{i}dar] [\@]
\protect \SubEtal )->[\protect ] [\SubEtal ] [\@]
\end{verbatim}

If arg3 = @ then the name is probably given in uninverted order. This may be because it is a Chinese name, and that is the correct order; but check to see if the name starts with an initial, which probably should get a warning.
\def\EmptyNameWarning{\amsrefs@warning{Empty author or editor name}}

Need to check for extra braces here.
\def\NameCheck#1#2#3\@nil{% 
  \def\@tempa{#1#2} 
  \edef\@tempa{\@nx\NameCheck@a\@xp\strip@prefix\meaning\@tempa} 
  \@tempa ??\@nil 
}\fi

This gets rid of the leading space in \on@line.
\def\MessageBreakNS{\MessageBreak\roman\numeral'\^^@}
\def\NameCheck@a#1#2#3\@nil{% 
  \if.#2\ifnum\catcode'#1=11 
  \amsrefs@warning{% 
Write ‘Lastname, First’ to ensure proper formatting\MessageBreakNS} 
  \fi
}\fi

\def\name@split#1,#2#3,#4#5,#6\@nil{% 
  \bib@name@init 
  \def\bib’surname{#1} 
  \def\bib’given{#2#3} 
  \def\bib’surname\Empty \EmptyNameWarning\fi 
}\fi
THE AMSREFS PACKAGE

\def\bib'jr{#4\#5}%
\ifx\@empty\bib'given \NameCheck \#1 ??\@nil
\else \extract@initials{\#2\#3}%
\fi

Extracting initials from the author name is an arduous chore, because of the various cases that need to be handled:
\doName{Harish, \'{E}tienne-P\'iere}
\doName{Jones, H. C.}
\doName{Jones, H.-C.}
\doName{Jones, H.C., Jr.}
\doName{Jones, Harold, IV}
\doName{'Swiderski, Jacek},
\doName{'S}widerski, Jacek},
\doName{{'S}widerski, Jacek},
\doName{Cartan, \'{E}lie}
\doName{Cartan, \'{E}lie}
\doName{Cartan, \'{E}.}
\doName{Shua Tae Yua}
\doName{Shua, Tae Yua}
\doName{Harish-Chandra}

So we do it in several steps. The result that we want to end up with is defining \bib'initials to contain, e.g.,
Jones, Harold C. \rightarrow H\InitialPunct C\InitialPunct

1. Expand accent commands in a way that replaces an accented letter with only the base letter, if it is lowercase (keep the accent command if the base letter is uppercase).
2. Convert periods to spaces.
3. Add a space before hyphens, to ensure that \InitialPunct will get added in the middle of a hyphenated compound name.
\def\xi@accent#1#2{\ifnum\uccode'#2=#2 \@nx#1{#2}\else #2\fi}
\def\xi@dot#1.{\xi@hyph#1@\space\@gobbletwo - \xi@dot}
\def\xi@hyph#1-{#1 -\xi@hyph}
\def\extract@initials#1{%
\begingroup
\def\i{i}\def\j{j}%
\def\ae{a}\let\oe\ae \let\o\ae \let\dh\ae \let\dj\ae \let\l\ae \let\ng\ae \let\ss\ae \let\th\ae
\def\"{\{x@accent"} \def\'{\{x@accent'} \def\.\{x@accent.\}%
\def\"{\{x@accent"} \def\'{\{x@accent'} \def\.\{x@accent.\}%
\def\"{\{x@accent"} \def\b{\{x@accent b}\def\c{\{x@accent c}\%\def\d{\{x@accent d}\def\h{\{x@accent H}\def\k{\{x@accent k}\%\def\r{\{x@accent r}\def\t{\{x@accent t}\def\u{\{x@accent u}%
This should probably get some extra work to handle the possibility that a name was entered in the bibliography in normal (uninverted) order.

[mjd,2000/04/07]
\def\lastName#1{\lnscan@a#1,\@nil}
\def\lnscan@a#1,#2\@nil{#1}
\def\discardName#1{}
\newcommand{\CiteNames}[1]{%
 \PrintSeries{\name}%
 \{\lastName\%
 \{} and \lastName\%
\}%
}
\def\discardName#1{}
\newcommand{{\CiteNames}[1]{%}
\PrintSeries{\name}%
\{\lastName\%
\}{ and \lastName\%
\}}

For three or more names: print ‘et al’ instead of the last name. Have to putz around with the spacefactor a bit or the comma between name and year will not be applied.
\{}\{\discardName\%
\}\{ \etaltext \}@\discardName\%
\def\PrintCiteNames#1{\leavevmode\def\@tempa{#1}\ifx\@tempa\prev@names\else\gdef\prev@names{#1}\toks\@emptytoks\CiteNames@prep#1\relax\edef\@tempa{\@nx\CiteNames{\the\toks@}}\@tempa\fi}

\def\CiteNames@prep#1{\@ifnotempty{#1}{\ifx\CitePrintUndefined#1\begingroup\def\CiteNames@prep##1{\endgroup\toks@{\name{\CitePrintUndefined{##1}}}\CiteNames@prep}\else\toks@{\the\toks@ \name{#1}}\CiteNames@prep\fi}}

\newcommand{\ReviewList}[1]{\PrintSeries{\review}{}{,}{ }{,}{ }{,}{ }{}{#1}{}{}}
\newcommand{\voltext}{vol.~}

12.12 Special formatting for other fields
Reviews are handled as a list to support the theoretical possibility of multiple reviews.
\newcommand{\ReviewList}[1]{\PrintSeries{\review}{}{,}{ }{,}{ }{,}{ }{}{#1}{}{}}
\newcommand{\voltext}{vol.~}

Scan the contents of a page value to see if it is a single page. Presence of \texttt{\ndash} or hyphen is taken to mean no. Probably should test also for spaces and commas. [mjd, 2000/01/24]
\newcommand{\DashPages}[1]{\pp\scan@\#1\\ndash \pp\scan@\#1\\ndash \pp\scan@\#3\#4\\nil}{\nil.}
12. IMPLEMENTATION

If we have eprint info and pages info and no journal name, the pages information is presumably the number of pages in the eprint.

\newcommand{\eprintpages}[1]{\% #1\IfEmptyBibField{eprint}{\{}{\IfEmptyBibField{journal}{ pp.}{}}}\%
}

The function \EnglishInitialCaps needs to verify that the language is English before applying capitalization.
\def\biblanguageEnglish{English}
\let\biblanguagedefault\biblanguageEnglish
\newcommand{\EnglishInitialCaps}[1]{\% \if\empty\bib'language \let\bib'language\biblanguagedefault\fi
\if\bib'language\biblanguageEnglish \@xp\inicap\fi
{"#1}\%
}

\def\PrintThesisType#1{\thesis@type#1?\@nil{#1}\
\def\thesis@type#1#2\@nil#3{\ifp\Ph.D. Thesis\%
\else\ifm\Master's Thesis\%
\else #3\fi\fi
}

Perhaps need to add allowbreak penalties at the parens in a DOI. Also what about prohibiting a break after the leading S?
\newcommand{\PrintDOI}[1]{\% DOI #1\%
\IfEmptyBibField{volume}{, (to appear in print)}{\%
}

When this function is called, \bib@year has been copied into \bib@date if necessary.

[This code has gotten sort of circular, it seems. Needs reassessment [mjd,2000/04/07]]
\def\bib@parsedate{\@xp\bib@parsedate@a\bib@date ---\@nil}
\let\bib@year\@empty
\let\bib@month\@empty
\let\bib@day\@empty
\def\bib@parsedate@a#1-#2-#3-#4\@nil{\def\bib@year{#1}\def\bib@month{#2}\def\bib@day{#3}\if\empty\bib@month \let\bib'date\bib@year\else \def\bib'date{#1-#2-#3}\fi
}

Print date in different forms depending on DOI and volume information.
\newcommand{\PrintDatePV}[1]{%
  \IfEmptyBibField{doi}{\let\@tempa\PrintDate}{%
    \IfEmptyBibField{volume}{\let\@tempa\PrintDatePosted}{%}
    \let\@tempa\PrintDate}}%
  \@tempa{#1}%
}

Intent is to handle variations such as 1987, August 1987, 1987-08, and 1987-08-14. If the month is present, print August or Aug. or 08 or nothing, at the behest of the bib style.

We’ve taken some special care to parse out the date info ahead of time, so this function just discards arg 1 and uses the already-parsed value.

\newcommand{\PrintDate}[1]{(\print@date)}
\def\print@date{%
  \ifx\bib@month\@empty \else\print@month@day \fi
  \bib@year}

With the Babel package, month names for a given language are typically available in a macro \month@language:
\def\month@german{
  \ifcase\month
    Januar\or Februar\or M\"arz\or April\or Mai\or Juni\or Juli\or August\or September\or Oktober\or November\or Dezember\fi}

However this is not true for English.
\def\print@month@day{%
  \ifcase 0\bib@month\ignorespaces
    \or January\or February\or March\or April\or May\or June\or July\or August\or September\or October\or November\or December\or Winter\or Spring\or Summer\or Fall\else Unknown Month\fi
  \fi
  \ifx\@empty\bib@day \else \space\number 0\bib@day,\fi
  \space}

You can use \PrintYear if you want to suppress month/day even when supplied in the data.
\newcommand{\PrintYear}[1]{\bib@year}

This one is special for AMS use.
\newcommand{\PrintDatePosted}[1]{\unskip, posted on \print@date}

\newcommand{\CardinalNumeric}[1]{%
  \number#1\relax
  \if\ifnum#1<14 \ifnum#1>thr@0 T\else F\fi\else F\fi\T% 
    th\%
  \else
    \@xp\keep@last@digit\@xp#1\number@i\relax
    \ifcase#1th\or st\or nd\or rd\else th\fi
  \fi}
\def\keep@last@digit#1#2{%
If the number assignment swept up all the contents, produce a cardinal number from \count@.
\newcommand{\print@edition}[1]{% 
\afterassignment\print@edition 
\count@ 0#1\relax@nil 
} 
\newcommand{\editiontext}{ed.} 
\def\print@edition#1#2\@nil{% 
\ifx\relax#1\relax 
\ifnum\count@>\z@ \CardinalNumeric\count@ 
\else ??th\fi 
\fi 
\editiontext 
\else \ifnum\count@>\z@\number\count@\fi #1#2\relax 
\fi 
} 
The braces here are just a simple way to terminate the number scanning. 
\newcommand{\SentenceSpace}{\relax\ifhmode{\spacefactor3000} \fi} 
\newcommand{\eprint}[1]{#1} 
The \texttt{www.arXiv.org} recommendations for citing their eprints are found at \url{http://xxx.lanl.gov/help/faq/references}, including these examples: 
arXiv:hep-th/9910001 
arXiv:math.AT/9910001 
\BibSpec{article}{% 
+{}{\PrintAuthors} {author} 
+{.}{ \textit} {title} 
+{.}{ \textit} {subtitle} 
+{.}{ } {journal} 
% +{.}{ ISSN } {ISSN} 
+{} { \textbf} {volume} 
+{} { \PrintDatePV} {date} 
+{.}{ \PrintDOI} {doi} 
+{.}{ \eprint} {eprint} 
+{} {parenthesize}{status} 
+{.}{ \eprintpages} {pages} 
+{.}{ } {note} 
+.{} {transition}
+{} \SentenceSpace \ReviewList \SentenceSpace \SentenceSpace \{review\}
+{} \SentenceSpace \{part\}
}
\BibSpec{book}{%
+{} \PrintAuthors\{author\}
+{} \PrintEditorsA\{editor\}
+{,} \textit\{title\}
+{,} \textit\{subtitle\}
+{,} \EnglishInitialCaps\{booktitle\}
+{,} \PrintEdition\{edition\}
+{,} \{series\}
+{,} vol.~\{volume\}
+{,} part~\{part\}
+{,} \{publisher\}
+{,} \{organization\}
+{,} \{place\}
+{,} \{date\}
+{,} ISBN \{isbn\}
+{,} \parentthesize\{status\}
+{,} \{note\}
+{,} \{transition\}
+{} \SentenceSpace \ReviewList \{review\}
}
\BibSpec{collection.article}{%
+{} \PrintAuthors\{author\}
+, \textit\{title\}
+, \textit\{type\}
+, \EnglishInitialCaps\{booktitle\}
+, \PrintEdition\{edition\}
+, \PrintEditorsB\{editor\}
+, Proc.~/ \{conference\}
+, \{series\}
+, vol.~\{volume\}
+, part~\{part\}
+, \{publisher\}
+, \{organization\}
+, \{place\}
+, \{date\}
+, \DashPages\{pages\}
+, ISBN \{isbn\}
+, \parentthesize\{status\}
+, \{note\}
+, \{transition\}
+{} \SentenceSpace \ReviewList \{review\}
}
\BibSpec{report}{%
+{} \PrintAuthors\{author\}
+{} \PrintEditorsA\{editor\}
+{,} \textit\{title\}
Some more entry types for implementing abbreviations.
Some special handling for “et alii” or “and others”.

\DefineName{alii}{\etaltext}
\DefineName{others}{\etaltext}

12.13 Extracting entries from an external database file

If you have a LaTeX document that contains amsrefs-style \bib entries, you can use it as a database and extract items from it for use in another document. In typical relatively simple scenarios, the extraction can be done by LaTeX itself on the first pass, so that citations in the text will be successfully resolved on the second pass (possibly even the first, depending on what kind of bibliography sorting is used).

\newread\bib@dbfile
\newcommand{\ReadBibData}[1]{%  
  \IfFileExists{#1.ltb}{%  
    \openin\bib@dbfile=@filef@und \relax  
  }{%  
    \IfFileExists{#1.ltx}{%  
      \openin\bib@dbfile=@filef@und \relax  
    }{%  
      \IfFileExists{#1.tex}{%  
        \openin\bib@dbfile=@filef@und \relax  
      }{%  
        \begingroup \NoBibDBFile{#1}\endgroup  
        \let\ReadBibPrelim\ReadBibPrelim  
      }  
    }  
  }%  
}\ReadBibPrelim

Maybe this should have a star form to mean, read everything into memory? Then it could be used at begin-document to make all the info available from the get-go. \[mjd,2001-12-13\]
\def\bblname{\jobname}
\newcommand{\bibselect}[2][\bblname]{%
12. IMPLEMENTATION

\if@filesw
  \typeout{Trying to create bbl file '1.bbl' ...
  \def\bibselect@msg{\typeout{ ... rats. Unable to create bbl file.}}%
  \let\@open@bbl@file\OpenBBLFile
  \let\@bibdef\selective@bibdef
  \ReadBibData{#2}%
  \fi
\@close@bbl@file
\let\@bibdef\normal@bibdef
\input{1.bbl}%
}
\def\NoBibDBFile#1{%
  \amsrefs@warning{No data file #1.ltb (.ltx, .tex) found}%
}

Start reading the db file one line at a time, looking for \begin{biblist}.
\def\ReadBibLoop{%
  \ifeof\bib@dbfile
    \closein\bib@dbfile
    \EmptyBibDBFile
    \@xp\@gobble
  \else
    \read\bib@dbfile to\CurLine
    \lowercase{\@xp\ReadBibLoop@a\CurLine}\@nil
  \fi
  \ReadBibLoop
}
\def\ReadBibPrelim{%
  \edef\bib@providefile{\@nx\ProvidesFile{\@filef@und}\relax}%
  \begingroup
    \let\do\@makeother \dospecials
  \ReadBibLoop
  \endgroup
  \bib@providefile
}
\edef\BibListMarker{\@backslashchar begin\string{biblist\string}}
\long\def\ReadBibLoop@a#1 #2\@nil{%
  \def\@tempa{#1}%
  \ifx\@tempa\BibListMarker
    \begingroup \def\ReadBibLoop\@nil\CurLine
  \fi
  \ReadBibLoop
}
\edef\ReadBibEntries{%
  \endgroup % turn off the mostly-verbatim catcodes
  \ReadBibEntry
}

Spaces and comments can simply be ignored, and that makes our scanning task easier.
\catcode32=9 \catcode'%=14 \relax
\ReadBibLoop
\endgroup
\bib@providefile
\edef\BibListMarker{\@backslashchar begin\string{biblist\string}}
\long\def\ReadBibLoop@a#1 #2\@nil{%
  \def\@tempa{#1}%
  \ifx\@tempa\BibListMarker
    \begingroup \def\ReadBibLoop\@nil\CurLine
  \fi
  \ReadBibLoop
}
\edef\ReadBibEntries{%
  \endgroup % turn off the mostly-verbatim catcodes
  \ReadBibEntry
}
\begingroup % pair for upcoming endgroup
\endgroup
\def\ReadBibEntry{% 
\ifeof\bib@dbfile 
\closein\bib@dbfile 
@xp@gobble 
\else 
\read\bib@dbfile to\CurLine 
% ifx\space\CurLine 
\else \ifx@empty \CurLine 
\else @xp\ReadBibLoop@e\CurLine@nil 
\fi \fi
\fi
\ReadBibEntry
\endgroup
\long\def\ReadBibLoop@e#1#2@nil{% 
% ifx\bib#1% 
% CurLine % just exec it 
\else \ifx\end#1\begingroup\let\ReadBibEntry\endgroup\fi \fi
\endgroup
\let\bbl@out=\relax
\let\bbl@write@gobble
\let\Open@bbl@file\relax
\let\Close@bbl@file\relax
\def\OpenBBLFile{% 
% if@filesw 
% Just use the next unused output stream 
% count@\count17 \advance\count@\one 
% ifnum\count@\sixt@on 
% global\chardef\bbl@out=\count@ 
% immediate\openout\bbl@out=\bblname.bbl\relax 
% global\let@close@bbl@file\CloseBBLFile 
% gdef\bbl@write\% immediate\write\bbl@out\% 
\else 
% ch@ck\count@\sixt@on\write 
\fi 
\global\let@close@bbl@file\relax
\endgroup
\def\CloseBBLFile{% 
% immediate\closeout\bbl@out\relax 
% global\let@close@bbl@file\relax 
% global\let\bbl@write@gobble 
% global\let\bbl@out\relax
\endgroup
12. IMPLEMENTATION

12.14 Cites

These variables are named to follow the precedent set by Arseneau’s cite package. \citepunct is used to separate multiple cites; \citemid is used to separate a cite number from additional information such as “Chapter 2”.

\def\citeleft{[}
\def\citeright{]}
\def\citemid{,\penalty9999 \space}
\def\citepunct{,\penalty9999 \hskip.13em plus.1em minus.05em\relax}
\def\citeform#1{}

The following definition provides some indirection that helps to deal with author-year object cites.

\def\citeleft{\citeleft}
\def\citemid\penalty9999 \space
\def\citepunct{,\penalty9999 \hskip.13em plus.1em minus.05em\relax}
\def\citeform#1{}

The information used by \cite for key moo is stored in \b@moo in the form \citesel{status1}{status2}{number}{year}{backref-info}

The first status flag is 1 if this key has already been cited earlier in the same document; 0 if not yet. This is used in some bibliography schemes to print a full list of author names for the first citation and an abbreviated author list for subsequent citations.

The second status flag is 1 if this key has already been used by a define-cite command (such as \bib); 0 otherwise. This makes it possible to issue a warning message as soon as the conflict is seen, on the first \LaTeX run, instead of on a subsequent run during the processing of the .aux file.

When an author/year citation scheme is in use, args 3 and 4 hold resp author names and year. Otherwise arg 3 simply holds a cite number.

And finally, arg 5 holds a list of backref pointers indicating the locations in the document where this entry has been cited.

\def\citesel@object#1#2#3#4#5{PrintCiteNames{#3}}
\def\citesel@year#1#2#3#4#5{#4}
\def\citesel@number#1#2#3#4#5{#3}
\let\citesel\citesel@number
\def\citesel@update#1#2#3#4#5#6{\gdef#6{1#2#3#4#5}}
\def\backref#1{\toks@{#1}\typeout{backref=[\the\toks@]}}

Here is the difference between the various optional forms of \cite:

\cite{xyz} -> \cite@\citesel{xyz}{}}
-> \cite@bc\b@xyz\citesel{}
\cite{xyz}*{blub} -> \cite@bc\b@xyz\citesel{blub}

\cite{blub}{xyz} -> \cite@bc\b@xyz\citesel{blub}

Need to handle the standard \[\ldots\] option for compatibility’s sake.

\newcommand{\InnerCite}[1]{\star@{\cite@a\citesel{#1}}{}}
\renewcommand{\cite}[2]{% 
  \if\cite@single#2,\@gobble \else\MultipleCiteKeyWarning{#2}{#1}\fi
  \@ifempty{#1}{% 
    \ObsoleteCiteOptionWarning
    \cite@o{#2}%
  }{% 
    \edef\cite@single#1,#2{\iffalse{\fi{\iffalse{\string}#2.\string}}} \%
  }%}
\edef\cite@single#1,#2{\iffalse{\fi{\iffalse{\string}#2.\string}}}

A list of simple cites. Make it robust in case used inside a figure caption.
(But then also, by the way, listoffigures should provide special handling.)

\begin{verbatim}
\edef\cite@single#1,#2{\iffalse{\fi{\iffalse{\string}#2.\string}}}
\end{verbatim}

The job of \cite@a is to convert the cite key to all catcode-12 characters
before passing it on to \cite@b.
12. IMPLEMENTATION

If it’s uninitialized, plug in an empty cite structure. \cite@bc should be executed only once for a given instance of a cite key. All further processing should go through \cite@cj.

\def\cite@bc#1#2{\def\CurrentCite{#2}\ifx#1\@@undefined \global\let#1\relax\fi \ifx#1\relax \global\let#1\empty@cite \fi \@xp\cite@nobib@test#1{}{}{}{}\@nil#1\cite@cj#1}
\def\empty@cite{\citesel 00{}{}{}{}}

If arg 4 is empty, it means there wasn’t any \bib command that defined a valid number.

\def\cite@nobib@test#1#2#3#4#5\@nil#6{\@ifempty{#4}{\G@refundefinedtrue \UndefinedCiteWarning \xdef#6{\@nx\citesel #2#3{\@nx\CitePrintUndefined{\CurrentCite}}{}{}}}{}{}}

\DeclareRobustCommand{\CitePrintUndefined}[1]{\begingroup\fontshape{n}\fontseries{mddefault} \ttfamily ?#1\endgroup}

This has to be a \let, not a \def.

\let\CPU@normal\CitePrintUndefined

\def\cite@multiple#1,#2#3\@nil{\ifx\relax#2F\else T\fi T}
\def\MultipleCiteKeyWarning#1#2{\amsrefs@warning{Use of \string\cites space is recommended instead of % \string\cite space\MessageBreak}
\def\citesel{\citesel{cite-already?}{bib-already?}{number}{year}{backrefs}}
\def\ObsoleteCiteOptionWarning{\amsrefs@warning{The form \string\cite{...}*[...] is recommended instead of \string\cite[...]{...}}\global\let\ObsoleteCiteOptionWarning\@empty}
\def\ar@hyperlink#1{\hyper@@link [cite]{}{cite.\CurrentCite}{#1}}
\def\cite@cj#1#2{\leavevmode\begingroup\cite@cb#1% write info to aux file\\citereft\ar@hyperlink{#1}{}\@ifnotempty{#2}{\citemid{#2}}\citeright\endgroup\ignorespaces % when called inside \citelist, a space might follow}
\def\extr@cite{\@xp\@gobblethree\string}
This is a copy of the standard warning from \@citex.
\def\UndefinedCiteWarning{\@latex@warning{\string\citation{\CurrentCite} on page \thepage\space undefined}}
}
Here is the original code for \nocite, need to adapt it. [mjd,2001-12-04]
\%\def\nocite#1{\@bsphack\%@for\@citeb:=#1\do{\edef\@citeb{\expandafter\@firstofone\@citeb}\% \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi\% \ifundefined{b@\@citeb}\G@refundefinedtrue\latex@warning{Citation \string\citation{\CurrentCite} undefined}}\@esphack}\%\expandafter\let\csname b@\endcsname\@empty\%\expandafter\\ctedest\%\$\\% As for the \ctedest, it would be a little more natural to add the page number as an extra argument of \citation, but Bib\TeX gives an error message if you do that (unless the second arg is put on a separate line; but I don’t care for that, for aesthetic reasons).\def\cite@cb#1{\%
Define \citesel to make \b@whatever update itself and add itself to the \all@cites list if it's not already there.

\begingroup \let\citesel\citesel@update #1#1\endgroup

When processing the aux file at begin-document, this is what \bibcite will do:

\def\amsRefs@bibcite#1\{\@xp\bibcite@a\csname b@#1\endcsname\}
\def\amsRefs#1\{\csname amsRefs@\@xp\@gobble\string#1\endcsname\}
\AtBeginDocument{\%
  \if@filesw
    \immediate\write\@mainaux{\string\providecommand\string\amsRefs{}}\%
  \fi
% #1<-@foo #2<-{number}{} or {author}{year} info
  \def\bibcite@def@#1\{\gdef#1\{\citesel\#2}\}
% Most of the time arg 1 will already be defined, by an earlier \citedest command in the aux file. Then we just need to change the number.
  \if\relax#1\gdef#1\{\citesel 00#2\}\
  \else
    \begingroup
      \@xp\bibcite@b\@xp#1#1\{#2\}
    \endgroup
  \fi
% #1 #2 #3#4 #5 #6 #7 #8
% \b@foo \rightarrow\citesel 00\{num\}{year}\{backrefs\} \{{...}\} \{{...}\}
  \def\bibcite@def@b#1#2#3#4#5#6#7#8\{#9\}{#10\}{#11\}{#12\}{#13\}{#14\}{#15\}{#16\}{#17\}{#18\}{#19\}
}

The \citedest command goes into the aux file to provide back-reference support.

\newcommand\citedest\{\@xp\cite@dest\csname b@#1\endcsname\}
\def\cite@dest#1\{\%
  \if\relax#1\gdef#1\{\citesel 00\}\
  \else
    \begingroup
      \@xp\cite@dest@b\@xp#1#1\{#2\}
    \endgroup
  \fi
% #1 #2 #3 #4 #5 #6 #7 #8
% \b@foo \rightarrow\citesel 00\{num\}{year}\{backrefs\} \{more-backref-info\}
  \def\cite@dest@b#1#2#3#4#5#6#7#8\{#9\}{#10\}{#11\}{#12\}{#13\}{#14\}{#15\}{#16\}{#17\}{#18\}{#19\}
}
By defining this as TeX’s maxint, undefined cites migrate to the end of a sorted list.
12. IMPLEMENTATION

\def\CPU@sort#1{2147483647}
\let\cite@dash\@empty

Need to reiterate the definition of \CurrentCite because I haven’t gotten the grouping complications sorted out yet. [mjd, 2001-12-20]

\def\cite@print#1#2{%
  \begingroup
  \let\CitePrintUndefined\CPU@normal
  \edef\CurrentCite{\extr@cite#1}%
  \cite@cjs#1[#2]%
  \BackCite
  \endgroup
  \restore@citepunct
}
\let\BackCite\@empty
\ifundefined{ifBR@verbose}{\let\ifBR@verbose\iffalse \let\fi\fi}{}%
\def\back@cite{%
  \ifBR@verbose
    \PackageInfo{backref}{back cite \string 'CurrentCite\string '}%
  \fi
  \Hy@backout{\CurrentCite}%
}

In an AMS-style bibliography, the backref info might follow the final period of the reference, or it might follow some Math Reviews info, without a period.

\def\print@backrefs#1{\SentenceSpace$\uparrow$\csname br@#1\endcsname}
\let\PrintBackRefs\@gobble
\def\load@backrefs{\@starttoc{brf}{}%}
\def\LoadBackRefs{}%
\IfOption{backrefs}{%\let\PrintBackRefs\print@backrefs\let\LoadBackRefs\load@backrefs\let\BackCite\back@cite\AtBeginDocument{%\PackageWarning{amsrefs}{backref option requires hyperref package}%\let\BackCite\@empty
  }{}}%

\def\cite@sorted@s{%
  \begingroup
  \let\CitePrintUndefined\CPU@sort
  \let\cite@cjs\cite@cj \let\cite@cj\cite@compress
  \begingroup
  \cite@toks\@emptytoks
When the time comes to apply compression, we have at our disposal a list of internal cite calls that looks like this:
\cite@c@cj@b@xxx{}\cite@c@cj@b@yyy{}...
12. IMPLEMENTATION

Now we bind \texttt{\cite@cj} to a suitable function and execute the list. For compression purposes, we just need to compare the previous number with the next number and keep track of the current state.

0. First number. Just print it, set the state to 1, and set last-cite-number to current-number.

1. If current-number = last-cite-number + 1, then the current number might be a middle number in a range. Save it as ”last-cite”; set last-cite-number to current-number; set the state to 2.

2a. If current-number = last-cite-number + 1, we are definitely now in the middle (or last number) of a range. Define \texttt{\cite@dash} to print a dash; save current-number as ”last-cite”; set last-cite-number to current-number; state remains 2.

2b. If current-number is NOT equal to last-cite-number + 1, then the previous number was either the second of two, or the end of a three-plus range. Print \texttt{\cite@dash} (which in the first case will just be empty), then print last-cite. Then do everything for state 0, and set the state to 1, because we might be at the beginning of a new range.

end. Print last-cite. If it happens that there was only a single cite, last-cite will be empty and nothing will happen. Otherwise this will finish off the current range or sequence of 2.

\let\last@cite\@empty
\def\cite@compress#1#2{% 
  \cite@print#1{#2}% 
  \@ifempty{#2}{% \@tempcnta 0#1\relax \let\cite@cj\cite@compress@a 
    }{% 
    \let\citepunct\citeAltPunct 
  }% 
}%
\def\cite@compress@a#1#2{% 
  \@ifempty{#2}{% 
    \advance\@tempcnta\@ne \@tempcntb 0#1\relax 
    \ifnum\@tempcnta=\@tempcntb 
      \def\cite@dash{\textendash \nopunct \cite@dash@cb} 
      \def\last@cite{\cite@print#1{#2}} 
      \let\cite@cj\cite@compress@b 
    \else 
      \cite@print#1{}\@tempcnta\@tempcntb 
  }{
THE AMSREFS PACKAGE

\fi
} \%
\citetopic{\cite@print#1{#2}}{\let\cite@cj\cite@compress
\let\citepunct\citeAltPunct
}%
}
\def\cite@compress@b#1#2{\%
@ifempty{#2}{\%
\advance@tempcnta\@ne \tempcntb 0\relax
\ifnum\tempcnta=\tempcntb
\def\last@cite{\cite@print#1{#2}}%
\else
\last@cite \let\last@cite\@empty
\cite@print#1{#2}\tempcnta\tempcntb
\let\cite@cj\cite@compress0a
\fi
}%
\cite@print#1{#2}
\let\cite@cj\cite@compress
\let\citepunct\citeAltPunct
}%
\def\citeAltPunct{; }
\IfOption{non-sorted-cites}{%
\DeclareRobustCommand{\citelist}{\citelist@unsorted}%
}{}
\IfOption{alphabetic}{%
\DeclareRobustCommand{\citelist}{\citelist@unsorted}%
}{}
\DeclareRobustCommand{\ycite}[1]{\star@{\cite@a\citesel@year{#1}}{}}%
\DeclareRobustCommand{\ycites}{\cites} not done yet
\DeclareRobustCommand{\ocite}[1]{\star@{\cite@a\citesel@object{#1}}{}}%
\DeclareRobustCommand{\ocites}{\cites} not done yet
\IfOption{author-year}{%
\let\citesel\citesel@authoryear
\def\citeleft{(}\def\citeright{)}%
\def@citeleft{\ifx\citesel\citesel@object\else\citeleft\fi}%
\def\citepunct{; }
\let\BibLabel\@empty
}%
\let\ycite\cite \let\ocite\cite
\let\ycites\cites \let\ocites\cites
}
Turn off hyper stuff if the hyperref package does not seem to be loaded.

\AtBeginDocument{%
\ifundefined{hyper@link}{%
    \let\ar@hyperlink\@firstofone
    \let\hyper@anchor@thebib\@firstofone
    \let\hyper@anchorstart\gobble \let\hyper@anchorend\relax
    \let\BackCite\@empty
}\%
}\}

\IfOption{jpa}{%
    \amsrefs@warning{The 'jpa’ option is obsolete}%
    \typeout{Trying \string\usepackage{amsjpa} instead ...}%
    \RequirePackage{amsjpa}[2000/02/02]
}\{}

Here ends the amsrefs package, unless the author-year option is in effect; then we want to use some different bibspecs.

\IfOption{author-year}{}{\endinput}
\renewcommand{\AuthorList}[1]{\PrintNames{author}{#1}}%

Invert the first author’s name.

\BibSpec{firstauthor}{
+{}{}{surname}
+{}{ }{given}
+{}{ }{jr}
}

No parens around the year.

\renewcommand{\PrintDate}[1]{\print@date}
\def\print@date{%
    \IfEmptyBibField{date}%
    {\IfEmptyBibField{year}{\BibField{status}}{\bib@year}}%
    {\bib@year}%
}

\BibSpec{article}{%
+{}{\PrintAuthors} {author}
+{}{ \PrintDate} {date}
+{}{ \textit} {title}
+{}{ } {part}
+{}{ } {journal}
% +{}{ ISSN } {ISSN}
+{} { \textbf} {volume}
}

The date form is tricky depending on presence or absence of DOI.

+{}{ \PrintDOI} {doi}
+{}{ \eprint} {eprint}
+{}{ \papertext}{status}
+{}{ \eprintpages} {pages}
+{}{ } {note}
+{}{ } {transition}
The usual \endinput to ensure that random garbage at the end of the file doesn’t get copied by docstrip.

\endinput

References


[5] Michael Piotrowski, Jens Klöcker, and Jörg Knappen, Is \LaTeX\ markup sufficient for scientific articles?