The **vector** package[*]

Nick Efford[†]

`nde@dcre.leeds.ac.uk`

1994/09/16

Abstract

This package provides a set of new commands for representing vectors in various ways. The commonly-used bold and underlined notations are supported, as is the 'hat' notation for representing unit vectors. Macros are also provided to represent a row or column vector as a set of elements.

1 Introduction

LaTeX 2_ε provides the `\vec` command to represent vectors in math mode; `$\vec{a}$`, for example, produces $\vec{a}$. In the author's experience, vectors are more commonly represented either in bold face roman type or else by means of underlining. Another convenient notation is the use of the 'hat' to indicate a unit-length vector. This package defines more suitable representations for vectors and unit vectors, using different fonts (boldface roman and sans serif) and two kinds of underlining (straight and wavy). It also defines macros which represent row or column vectors as implicit or explicit sequences of elements.

2 Examples

Six new commands are defined for representing vectors with a single (possibly composite) symbol. They are shown, with sample output, in table 1. Unlike `\vec`, the new commands can be used in text mode as well as math mode.

`\uvec`
`\uuvec` By default, `\uvec` and `\uuvec` underline a symbol using a straight line. If a wavy line is prefered, then the `wavy` package option should be specified.

`\irvec`
`\icvec` Another set of commands are defined which can represent a vector as a row or column of elements. `\irvec` and `\icvec` generate 'implicit' row and column vectors, respectively. Here, only the first and last elements actually appear; `\ldots` is used to imply the existance of the rest. Both macros take one mandatory argument, a character which names the vector. By default, the first and last elements are constructed from this character and the subscripts '1' and n, respectively. An optional argument allows final subscripts other than n to be specified. The subscript for the first element cannot be altered in this way, but then it is not likely that you will want to change this often. If you do need to change it, the command
`\firstelement` `\firstelement` can be used. Table 2 shows some sample output for `\irvec`.

---

[*]This file is v1.0, last revised 1994/09/16.

[†]Address: School of Computer Studies, University of Leeds, Leeds LS2 9JT, United Kingdom

The syntax for `\icvec` is the same as that for `\irvec`. One important difference is that `\icvec` can only be used in math mode, whereas `\irvec` can be used in both math and text modes. For example, `\buvec{q} = \left(\icvec{q}\right)` produces

$$\hat{\mathbf{q}} = \left(\begin{array}{c} q_1 \\ \vdots \\ q_n \end{array} \right)$$

\rvec   The final pair of macros, `\rvec` and `\cvec`, provide *explicit* representations of
\cvec   a vector as a row or column of elements, i.e., all elements of the vector are shown[1]. Three mandatory arguments are used to specify the name of the vector, an integer subscript for the first element and an integer subscript for the final element. For instance, `\bvec{x} = \{\rvec{x}{1}{6}\}` produces

$$\underline{x} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$$

and `\bvec{y} = \left[\cvec{y}{0}{3}\right]` gives

$$\mathbf{y} = \left[\begin{array}{c} y_0 \\ y_1 \\ y_2 \\ y_3 \end{array} \right]$$

As with the implicit macros, `\rvec` may be used in both math and text modes, whereas `\cvec` may only be used in math mode.

3   The Macros

  1 ⟨∗package⟩

We require that the ifthen and calc packages are loaded:

  2 `\RequirePackage{ifthen}`
  3 `\RequirePackage{calc}`

A boolean variable is defined and set to *true* if the wavy package option has been specified:

  4 `\newboolean{@wavy}`
  5 `\DeclareOption{wavy}{\setboolean{@wavy}{true}}`
  6 `\ProcessOptions`

\bvec   `\bvec` represents a vector using a bold series roman character.

  7 `\newcommand{\bvec}[1]{\ensuremath{\mathbf{#1}}}`

\buvec   `\buvec` represents a unit vector as a bold roman character with a hat.

  8 `\newcommand{\buvec}[1]{\ensuremath{\mathbf{\hat{#1}}}}`

\svec   `\svec` represents a vector as a sans serif character. family.

  9 `\newcommand{\svec}[1]{\ensuremath{\mathsf{#1}}}`

---

[1]Clearly, these macros are suitable only for vectors with comparatively small numbers of elements!

\suvec \qquad \suvec represents a unit vector as a sans serif character with a hat.

```
10 \newcommand{\suvec}[1]{\ensuremath{\mathsf{\hat{#1}}}}
```

How we define underlined vectors depends on the value of boolean variable @wavy:

```
11 \ifthenelse{\boolean{@wavy}}{%
12   \PackageInfo{vector}{wavy underlining selected}
```

If @wavy is set then we define a macro \undertilde[2], which places a tilde symbol underneath its argument:

```
13   \newcommand{\undertilde}[1]{\mathord{\vtop{\ialign{##\crcr
14     $\hfil\displaystyle{#1}\hfil$\crcr\noalign{\kern1.5pt\nointerlineskip}
15     $\hfil\tilde{}\hfil$\crcr\noalign{\kern1.5pt}}}}}
```

\uvec \qquad We then define \uvec in terms of \undertilde:

```
16   \newcommand{\uvec}[1]{\ensuremath{\undertilde{#1}}}
```

\uuvec \qquad And similarly define \uuvec:

```
17   \newcommand{\uuvec}[1]{\ensuremath{\hat{\undertilde{#1}}}}}{%
```

If @wavy is not set, then we define \uvec and \uuvec in terms of \underline:

```
18   \newcommand{\uvec}[1]{\ensuremath{\underline{#1}}}
19   \newcommand{\uuvec}[1]{\ensuremath{\hat{\underline{#1}}}}}
```

\firstelement \qquad Now we define a variable to store the subscript used for the first element of a row or column vector, along with a command which can be used to alter that variable:

```
20 \def\first@element{1}
21 \newcommand{\firstelement}[1]{\def\first@element{#1}}
```

\irvec \qquad The \irvec command is very simple:

```
22 \newcommand{\irvec}[2][n]{\ensuremath{{#2}_{\first@element},\ldots,{#2}_{#1}}}
```

\icvec \qquad The \icvec command uses an array environment, and so can only be used in a math environment:

```
23 \newcommand{\icvec}[2][n]{%
24   \begin{array}{c}
25     {#2}_{\first@element}\\ \vdots\\ {#2}_{#1}
26   \end{array}}
```

For \rvec and \cvec, we must define a loop counter which stores the current subscript of a vector element:

```
27 \newcounter{vec@elem}
```

\rvec \qquad Now we define \rvec. We must check that the last subscript for the vector elements (#3) is greater than the first subscript (#2). If this is so, then we use \whiledo to loop over specified range of values, generating a vector element with subscript vec@elem followed by a comma on each iteration. Otherwise, we simply generate a single vector element.

```
28 \newcommand{\rvec}[3]{%
29   \ensuremath{%
30     \ifthenelse{#3 > #2}{%
```

---

[2]Note that I didn't write this macro. Unfortunately, I cannot give proper credit as I do not recall how I came by it!

```
31      \setcounter{vec@elem}{#2}
32      \whiledo{\value{vec@elem} < #3}%
33        {{#1}_{\thevec@elem}, \stepcounter{vec@elem}}%
34      {#1}_{#3}}{{#1}_{#2}}}}
```

\cvec We define `\cvec` in a similar way to `\rvec`, only here we iterate within an `array`
environment and generate a vector element and a linebreak on each iteration.

```
35 \newcommand{\cvec}[3]{%
36   \ifthenelse{#3 > #2}{%
37     \setcounter{vec@elem}{#2}
38     \begin{array}{c}
39       \whiledo{\value{vec@elem} < #3}%
40         {{#1}_{\thevec@elem} \\ \stepcounter{vec@elem}}%
41       {#1}_{#3}
42     \end{array}}{{#1}_{#2}}}
```

`\bvec{a}, \buvec{a}`	$\mathbf{a}$, $\mathbf{\hat{a}}$
`\svec{a}, \suvec{a}`	a, â
`\uvec{a}, \uuvec{a}`	$\underline{a}$, $\underline{\hat{a}}$ / $\utilde{a}$, $\hat{\utilde{a}}$

Table 1: new commands for symbolic vectors.

`\irvec{a}`	$a_1, \ldots, a_n$
`\irvec[4]{a}`	$a_1, \ldots, a_4$
`\firstelement{0}` `\irvec[9]{a}`	$a_0, \ldots, a_9$

Table 2: row vectors with implicit elements.