nimsticks v2.0.1

Peter Rowlett

August 14, 2022

1 Background

`nimsticks` is a package for LaTeX that draws sticks for representating games of multi-pile Nim.

Nim objects could be anything, of course, but conventionally sticks or stones are used. There are various types of dot in LaTeX that might look like stones, but somehow a line of dots didn't seem satisfactory. There are various ways to draw a line (e.g. just typing IIIII), including some tally markers (e.g. in hhcount). My problem with these (call me picky) is that they are all identical lines, and a 'heap' of them just looks very organised. Really, I want a set of lines that looks like someone just threw them into heaps (though probably without crossings for the avoidance of ambiguity).

The way this works is it draws a thick vertical line in TikZ with a little wobble added so each one doesn't look extremely well-lined-up with its neighbour, achieved by adding or subtracting a small random number to the top and bottom coordinate. There are various built-in options to customise the size and colour of the sticks, and flexibility to draw heaps of different objects.

N.B. The precise look of example output in this document is affected by random wobble in the sticks.

2 Basic usage

2.1 Draw a game of multi-pile Nim

For example, the input `\nimgame{5,3,4}` will produce output like this:

This is designed to look like a 3-pile Nim game with 5 sticks in the first pile (or heap), 3 in the second and 4 in the third.

`\nimgame` will happily work with one heap, so for example the input `\nimgame{7}` will produce output like this:

The command `\nimgame` presents the Nim game as its own paragraph with `\centering`. An optional flag `inline` can be used to produce the Nim game inline

without the `\centering`. For example, the command `\nimgame[inline]{5,3,4}` will produce outline like this: 

2.2 Draw a single Nim stick

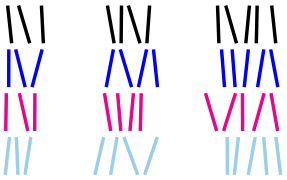
It is likely the user will want to use `\nimgame` and not `\drawnimstick` directly, but the input `\drawnimstick` will produce output like this: 

3 Customisation

3.1 Colour

A command `\setnimstickcolour{colour}` is used to set the colour of the sticks, by default black. The input `colour` can be any named colour defined in the document.

```
\nimgame{3,4,5}

\setnimstickcolour{blue}\nimgame{3,4,5}

\setnimstickcolour{magenta}\nimgame{3,4,5}

\definecolor{lightcornflowerblue}{rgb}{0.6, 0.81, 0.93} % from https://latexcolor.com/
\setnimstickcolour{lightcornflowerblue}\nimgame{3,4,5}
```

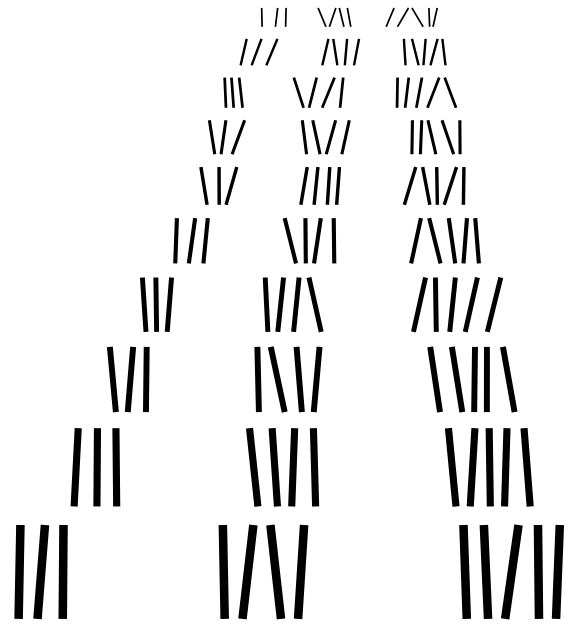


4 Size

4.1 Basic size configuration

`\nimgame` and `\drawnimstick` respond to the current text size, as demonstrated below. This is because their size is defined in ex, with the default height of a stick 3.3ex (roughly twice the size of a capital letter in the current font).

tiny:
scriptsize:
footnotesize:
small:
normalsize:
large:
Large:
LARGE:
huge:
Huge:



4.2 Advanced size configuration

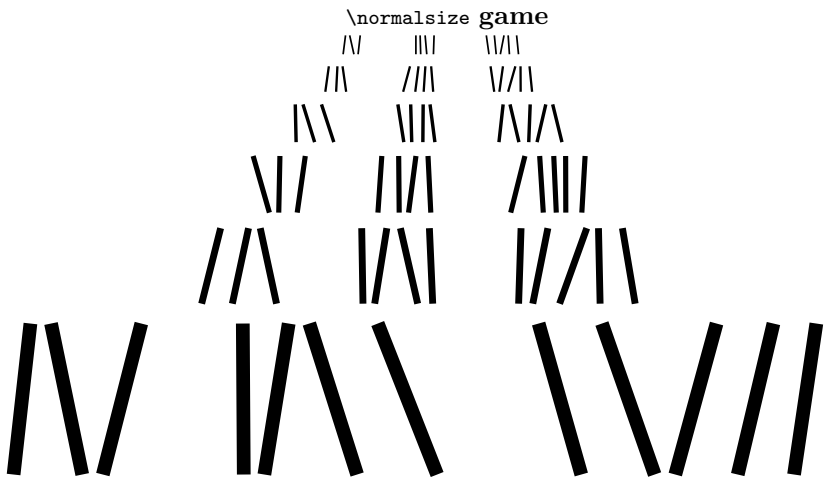
The size can be further customised by a command `\setnimscale{num}`. This controls the height of the sticks, their thickness, the gaps between sticks, the gaps between heaps and the size of the wobble.

Usage is like `\setnimscale{2}\nimgame{3,4,5}`

For example:

Nim scale `\normalsize` **game**

0.5:
0.675:
1:
1.5:
2:
4:



5 Advanced use

5.1 Changing the look of the Nim sticks/stones/etc.

There is a command `\onenimstick` that includes only the TikZ code for drawing a single Nim stick. This can be redefined to draw different objects as Nim sticks using in principle arbitrary TikZ code.

`\onenimstick` is called from within a `tikzpicture` environment.

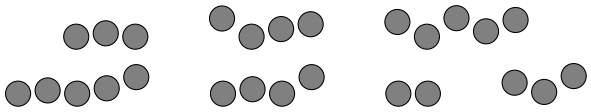
You have three randomisation parameters available to you:

- `\topx` and `\botx`: small random numbers which are in the range -0.1 to 0.1 (for default scale `\normalsize` heaps).
- `\lift`: small random number which is in the range 0 to 0.2 (for default scale `\normalsize` heaps).

5.1.1 Examples

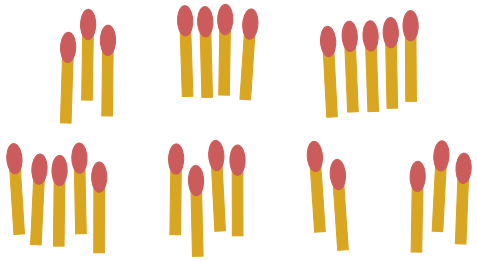
1. Here we draw each stick as a little circle (to look like stones), with a little vertical `\lift` to make them sit irregularly.

```
\renewcommand{\onenimstick}{%
    \node[draw,black,fill=gray,circle] at (0,\lift) {};
}
\nimgame{3,4,5}

\nimgame{5,4,2,3}
```



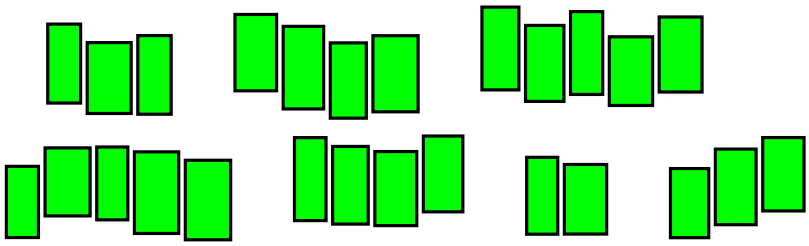
2. Here we draw each stick as a little matchstick, with a little vertical `\lift` to make them sit irregularly.

```
% Colours from https://latexcolor.com/
\definecolor{goldenrod}{rgb}{0.85, 0.65, 0.13}
\definecolor{indianred}{rgb}{0.8, 0.36, 0.36}
\renewcommand{\onenimstick}{%
    \draw[goldenrod,fill=goldenrod,rotate=\topx*40] (0,\lift) rectangle (0.14,1+\lift);
    \draw[indianred,fill=indianred,rotate=\topx*40] (0.07,1+\lift) ellipse (0.1 and 0.2);
}
\nimgame{3,4,5}

\nimgame{5,4,2,3}
```



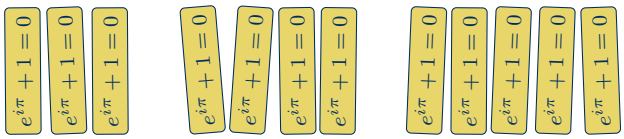
3. To demonstrate arbitrary drawing, here each item in the Nim heap is a little green rectangle, with size randomised using `\topx` and `\botx`, with a little vertical `\lift` to make them sit irregularly.

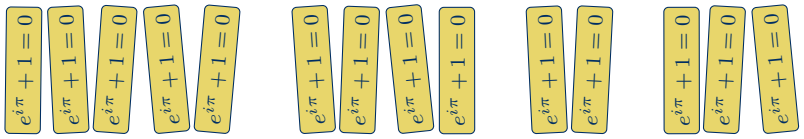
```
\renewcommand{\onenimstick}{%
    \draw[very thick,draw,fill=green] (0,\lift) rectangle (\topx+0.5,\botx+1+\lift);
}
\nimgame{3,4,5}

\nimgame{5,4,2,3}
```



4. And, just to get silly:

```
% Colours from https://latexcolor.com/
\definecolor{coolblack}{rgb}{0.0, 0.18, 0.39}
\definecolor{arylideyellow}{rgb}{0.91, 0.84, 0.42}
\renewcommand{\onenimstick}{%
    \node[draw,coolblack,fill=arylideyellow,rotate=90+\topx*60,anchor=north,rounded
    corners=0.5ex] at (\topx,\botx) {\(e^{i\pi} + 1 = 0\)};
}
\nimgame{3,4,5}\bigskip
```





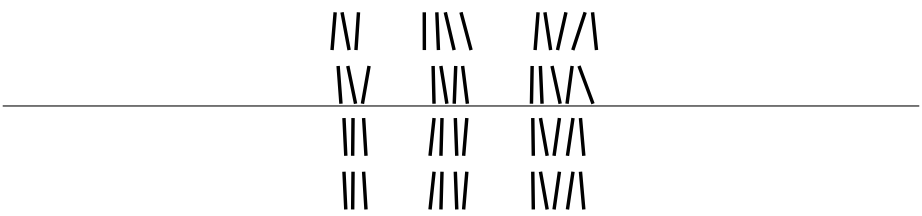
5.2 Creating the same randomisation twice

The randomisation is done with `lcg` and you can set the seed using `\reinitrand` from that package. The seed is supposed to be a number from 1 to $2^{31}-1$, otherwise the package will use a combination of the time, page number and line number to make the seed. So if you reset the randomisation to the same seed before running `\drawnimstick` or `\nimgame` you'll get the same look to the sticks.

5.2.1 Examples

1. Here we generate the same Nim pile twice with exact reproduction of the randomisation. The first pair of Nim games (without `\reinitrand`) appear different; the second pair of Nim games appear identical because `\reinitrand[first=-100,last=100,seed=1]` is used to reset the randomisation to the same point before each is drawn. Different seeds can be used if there are multiple sets of Nim games that require consistent looks.

```
\nimgame{3,4,5}

\nimgame{3,4,5}

\hrule

\reinitrand[first=-100,last=100,seed=1]
\nimgame{3,4,5}

\reinitrand[first=-100,last=100,seed=1]
\nimgame{3,4,5}
```



2. Here we use nimsticks within Beamer with a reveal slide using `\only`. On the first slide a Nim game is produced and when the slide is progressed to reveal "Text" the Nim sticks all move. On the second slide the Nim sticks do not move when "Text" is revealed.

```
\documentclass{beamer}
```

```
\usepackage{nimsticks}
\begin{document}
\begin{frame}
    \nimgame{2,3,4} \only<2>{Text}
\end{frame}
\begin{frame}
    \reinitrand[first=-100,last=100,seed=123]\nimgame{2,3,4} \only<2>{Text}
\end{frame}
\end{document}
```

3. Here two pairs of Nim games are reproduced. Games 1 and 3 are identical using `seed=1`, and games 2 and 4 are identical using `seed=2`.

   ```
   Game 1: \reinitrand[first=-100,last=100,seed=1]\nimgame{5,8,1,4}

   Game 2: \reinitrand[first=-100,last=100,seed=2]\nimgame{2,3}

   Game 3: \reinitrand[first=-100,last=100,seed=1]\nimgame{5,8,1,4}

   Game 4: \reinitrand[first=-100,last=100,seed=2]\nimgame{2,3}
   ```

   Game 1:

   Game 2:

   Game 3:

   Game 4:

6 Warnings

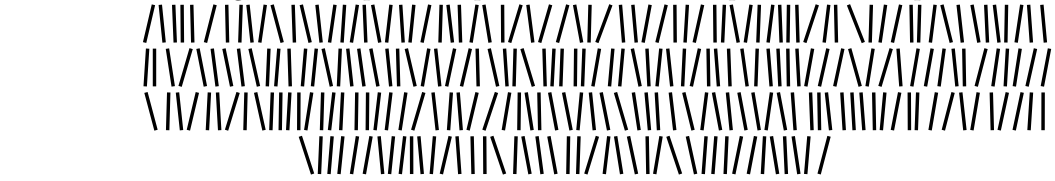
There is no limit in the code to the number of piles or the number in a pile, but this code doesn't do anything to cope when line breaks start happening, and presumably there is a computational limit.

In principle, if you add lots of piles it will just wrap onto multiple lines, though it will start to look less clear. For example, the input

```
\nimgame{1,2,3,4,5,6,7,8,9,10,9,8,7,6,5,4,3,2,1}
```

will produce the output:

Similarly, if you have a lot of sticks in the same heap, it will wrap and look confusing, for example the input `\nimgame{128}` will produce the output:



7 Changes to usage or output

(for full change log, see GitHub `README.md`)

- v2.0.1: documentation tweaks.
- v2.0: rewrite, now based on ex and with options to customise size and colour, though basic usage should remain the same. More detailed documentation. Fixed a bug in v1.2 where a block display `\nimgame` didn't start itself in a new paragraph.
- v1.2:
  - switched `\begin{center}` to `\centering` (because the former doesn't work in `standalone` documents and the latter doesn't add vertical space);
  - removed some whitespace that appeared to the right of the last heap.
- v1.1: added option to make inline Nim game.