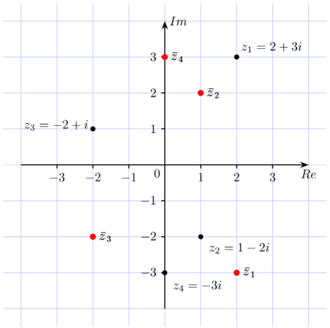
Combining LaTeX graphics with JSXGraph in Numbas

Jim Pettigrew and Don Shearman

Mathematics Education Support Hub, Western Sydney University



Introduction

In this talk I will demonstrate a workflow that enables LaTeX-generated graphics to be ported to Numbas as svg files, embedded in a JSXGraph object and made interactive using basic tools in the extension.

A feature of the workflow will be the ‘syncing’ of the LaTeX and JSXGraph coordinate systems, allowing easy geometric placements and adjustments to be made in JSXGraph.

LaTeX graphics engines

The workflow I will demonstrate uses TikZ to generate the graphics.

I have successfully experimented with a workflow using PSTricks, though this was a little more complicated due to my inability to simplify the synching of coordinate systems (I ended up using a vector graphics editor to manually determine all required distances and dimensions).

Motivation

Many of those who regularly use LaTeX will have accumulated a significant number of high-precision figures that could be useful as illustrative or instructive tools in Numbas.

By using a dvi-to-svg conversion tool, it is possible to faithfully convert LaTeX-generated figures to vector graphics files that can be included as images in Numbas questions.

But this can be taken further: the converted images can be embedded in JSXGraph objects within Numbas and hence animated using the dynamic geometric capabilities of this extension.

Preparation

The workflow uses a dvi-to-svg conversion tool and a vector graphics editor. I use **dvisvgm** (that ships with the MiKTeX distribution) and **Inkscape** respectively.

I use a template LaTeX file for generating the TikZ graphics and some boilerplate JessieCode for creating the JSXGraph object in Numbas (links below).

tikz_template.tex
https://drive.google.com/file/d/1smusSKWwP82VyK-H8LQJn-dP___imQ0UK/view?usp=sharing

jessie_code_template.txt
https://drive.google.com/file/d/10MFS4WziIB0zm3beYG5in598RA2Q_rZI/view?usp=sharing

tikz_code_for_clock.txt
https://drive.google.com/file/d/1-p_w1IlST6qBJBVJrFchDg0eeLBu7V1-/view?usp=sharing

Graphics generation

Compile demo.tex to dvi.

Using a console:

```
cd C:\NumbasUserMeeting\WorkflowDemo
dvisvgm --no-fonts demo.dvi
```

Graphics modification (if needed)

Use Inkscape to modify the svg file (demo.svg) and save the modified file as ‘Plain SVG (*svg)’.

Do not add elements that breach the bounding rectangle.

Upload graphics to Numbas and sync

Create a new question in Numbas, load demo.svg as a resource, include the JSXGraph extension and add a variable named ‘diagram’ for the JSXGraph object.

My example is here.

Copy the JessieCode boilerpate into the value of the diagram variable.

For the image dimensions, use the demo.output file in the same directory as demo.tex.

Add JSXGraph geometry

Check axis and grid alignment before switching these elements off.

Add desired geometry using variable inputs.

Insert variable inputs (to the left of the 'safe' command):

```
‘‘hourHandCoordsX = {hourHandCoords[0]};hourHandCoordsY = {hourHandCoords[1]};
minuteHandCoordsX = {minuteHandCoords[0]};minuteHandCoordsY = {minuteHandCoords[1]};”+
```

Add geometry:

```
segment([0,0],[hourHandCoordsX,hourHandCoordsY])
<<strokeColor: 'black', highlight: false, lastArrow: true, strokeWidth: 3.5, fixed: true>>;
segment([0,0],[minuteHandCoordsX,minuteHandCoordsY])
<<strokeColor: 'black', highlight: false, lastArrow: true, strokeWidth: 3.5, fixed: true>>;
```