Melbourne Bioinformatics Seminar

Hatching a plot on the command line (part 2)

Bernard Pope
Lead Bioinformatician, Cancer and Clinical Genomics
Victorian Health and Medical Research Fellow
Melbourne Bioinformatics
The University of Melbourne, Australia

Recap

- Hatch is a command line tool for analysing and visualising data.
- It takes input from tabular data in CSV or TSV format and produces high-quality plots (charts, graphs) as output.
- It is designed to be fast and convenient, and is particularly suited to data exploration tasks. Input files with large numbers of rows (> millions) are readily supported.
- Hatch plots are highly customisable, however for most cases sensible defaults are applied.
- Hatch is implemented in Python and makes extensive use of the Pandas, Seaborn, and Scikit-learn libraries for data processing and plot generation.

Plot types

From the previous version

- Histogram
- Count
- Scatter
- Box
- Violin
- Line
- Heatmap

New in this version

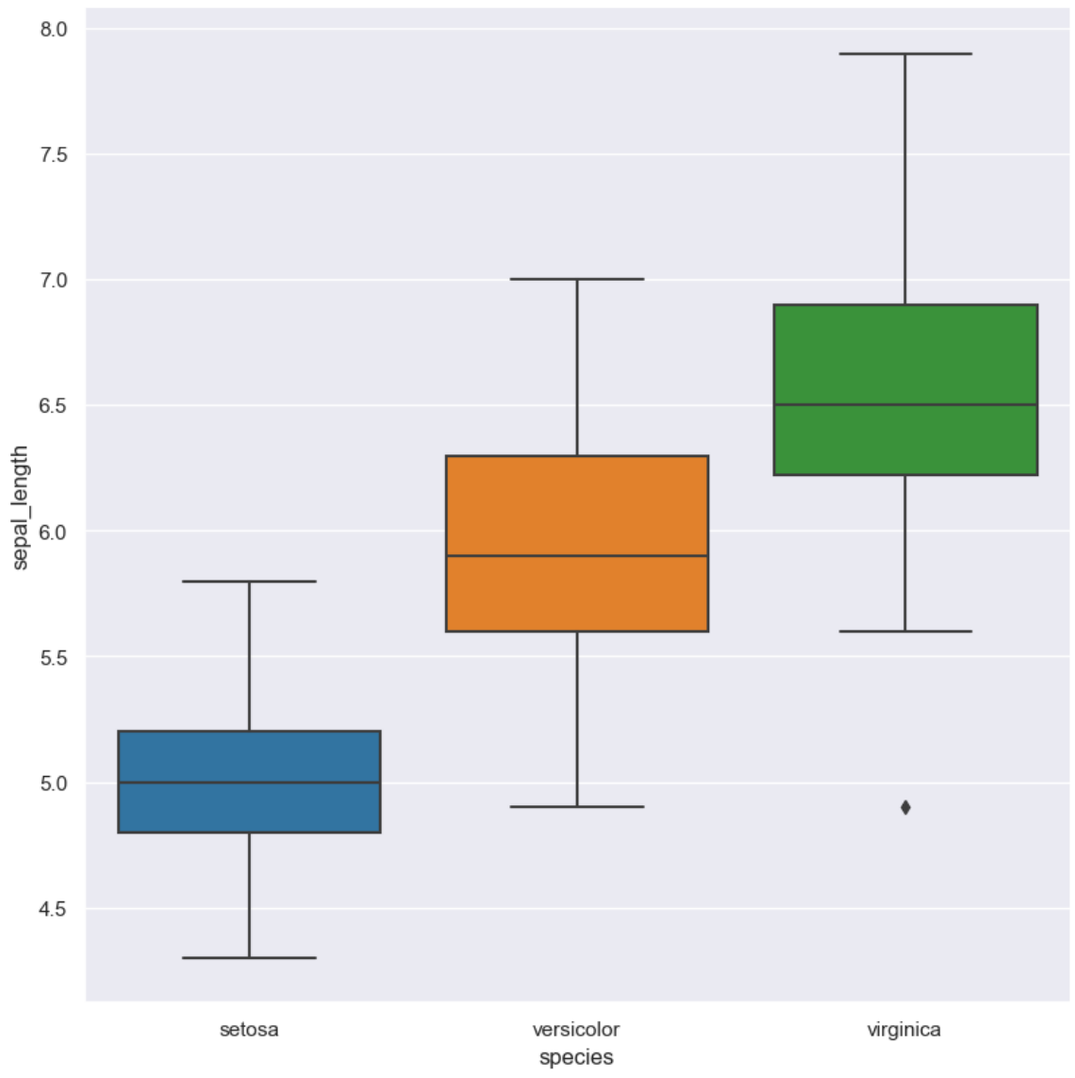
- Swarm
- Strip
- Boxen
- Point
- Bar
- Principal Components Analysis

Simple example

```
$ hatch noplot --info iris.csv
        sepal_length  sepal_width  petal_length  petal_width     species
count     150.000000   150.000000    150.000000   150.000000         150
unique           NaN          NaN           NaN          NaN           3
top              NaN          NaN           NaN          NaN  versicolor
freq             NaN          NaN           NaN          NaN          50
mean        5.843333     3.054000      3.758667     1.198667         NaN
std         0.828066     0.433594      1.764420     0.763161         NaN
min         4.300000     2.000000      1.000000     0.100000         NaN
25%         5.100000     2.800000      1.600000     0.300000         NaN
50%         5.800000     3.000000      4.350000     1.300000         NaN
75%         6.400000     3.300000      5.100000     1.800000         NaN
max         7.900000     4.400000      6.900000     2.500000         NaN
rows: 150, cols: 5
```

Simple example

```
$ hatch box -x species -y sepal_length iris.csv
```



output is written to
iris.sepal_length.species.box.png

Other features

- Input data transformation:
    - Row filtering (better syntax in new version)
    - Dynamic computation of new columns (new)
    - Random subsampling of rows (new)
        - Transformed input data can be saved back to a file (new)
- Facet plots (new)
- Data summarisation via `--info` (new)
- Various plot aesthetic styling options (new)
- Choice of output graphics file formats: png, pdf, svg, jpg (new)
- Optionally open plot in interactive window instead of saving to file (new)

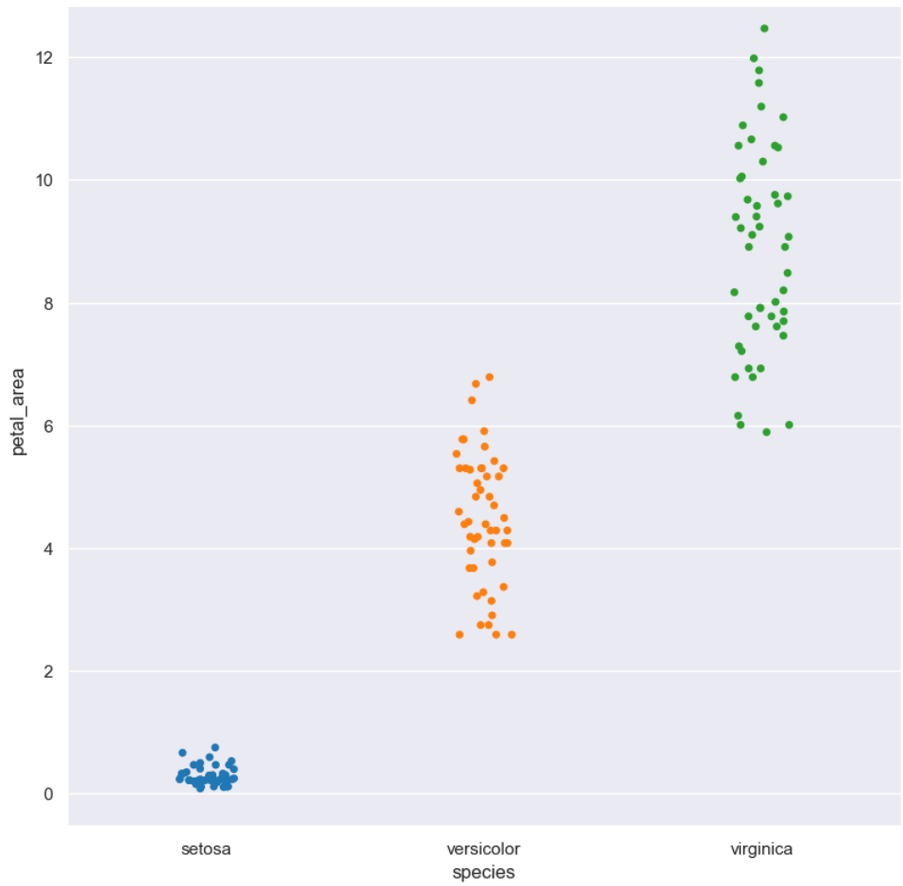
Documentation

- New documentation via Sphinx:

  https://hatch.readthedocs.io/en/latest/index.html

Computation of new rows

```
$ hatch strip \
    --eval 'petal_area = 3.142 * (petal_length / 2) * (petal_width / 2)' \
    -x species -y petal_area \
    iris.csv
```

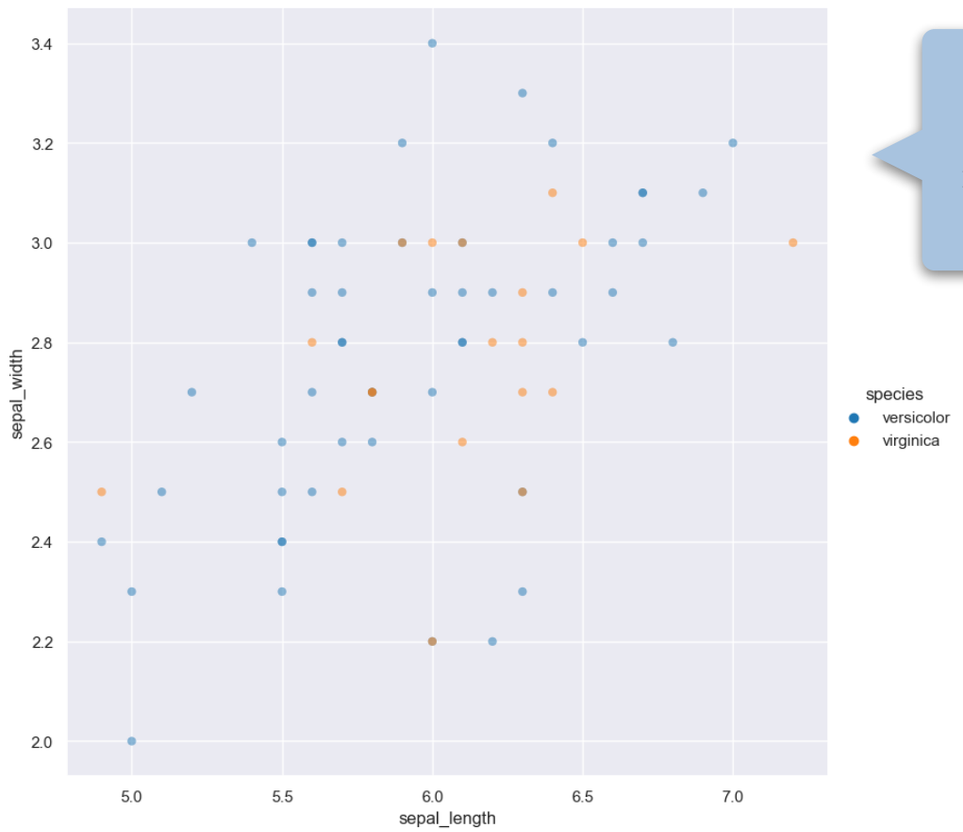


output is written to
iris.petal_area.species.strip.png

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.eval.html

Better filtering syntax

```
$ hatch scatter \
    --eval 'petal_area = 3.142 * (petal_length / 2) * (petal_width / 2)' \
    --filter '2 <= petal_area <= 8' \
    -x sepal_length -y sepal_width --hue species iris.csv
```



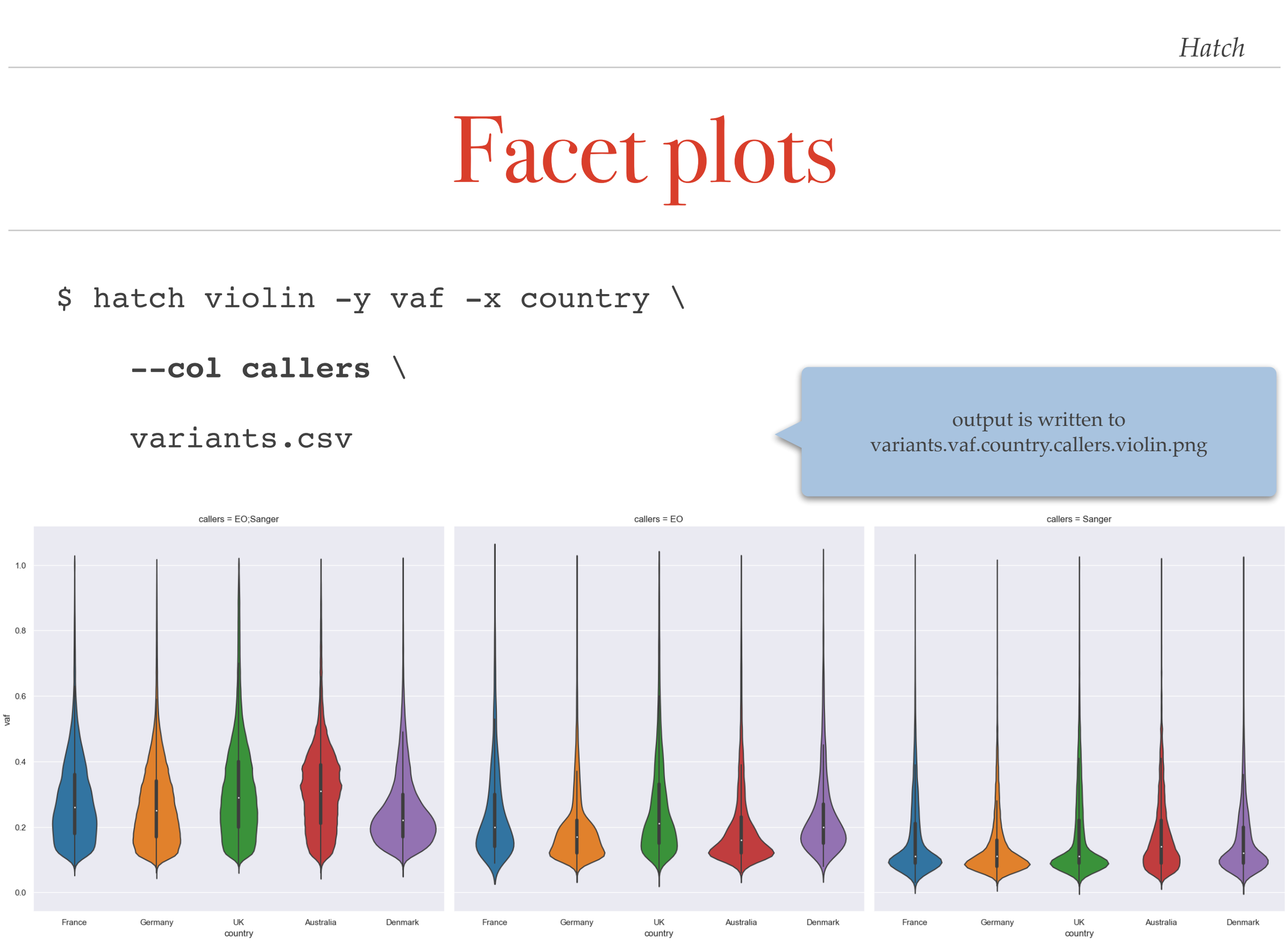
output is written to
iris.sepal_width.sepal_length.species.scatter.png

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.query.html

Facet plots

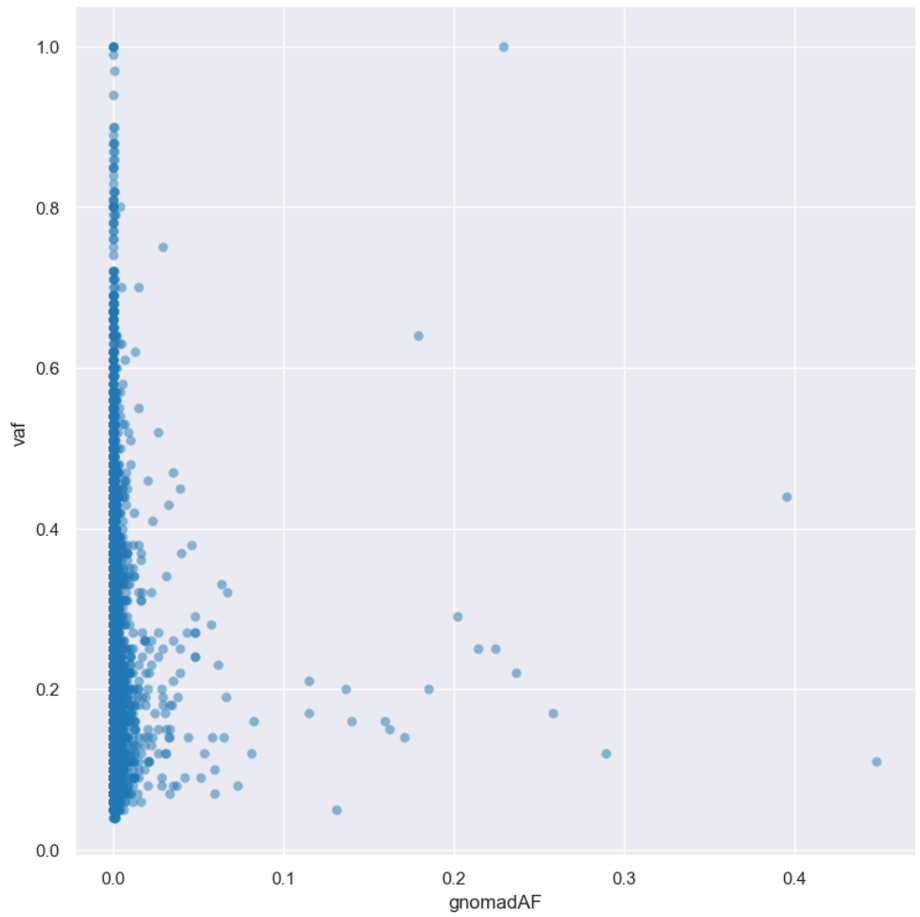
```
$ hatch violin -y vaf -x country \
    --col callers \
    variants.csv
```

output is written to
variants.vaf.country.callers.violin.png



Sampling

```
$ hatch scatter -x gnomadAF -y vaf \
    --sample 0.01 \
    all_variants.pon.csv
```



Sample can be a fraction 0 < n < 1, or absolute number n >= 1

Sampling happens after --eval and --filter

Saving transformed data

```
$ hatch noplot \
    --eval 'is_pop = gnomadAF > 0.1' \
    --filter 'country == "Australia" and is_pop == True' \
    --sample 0.01 \
    --save out.csv \
    all_variants.pon.csv
```

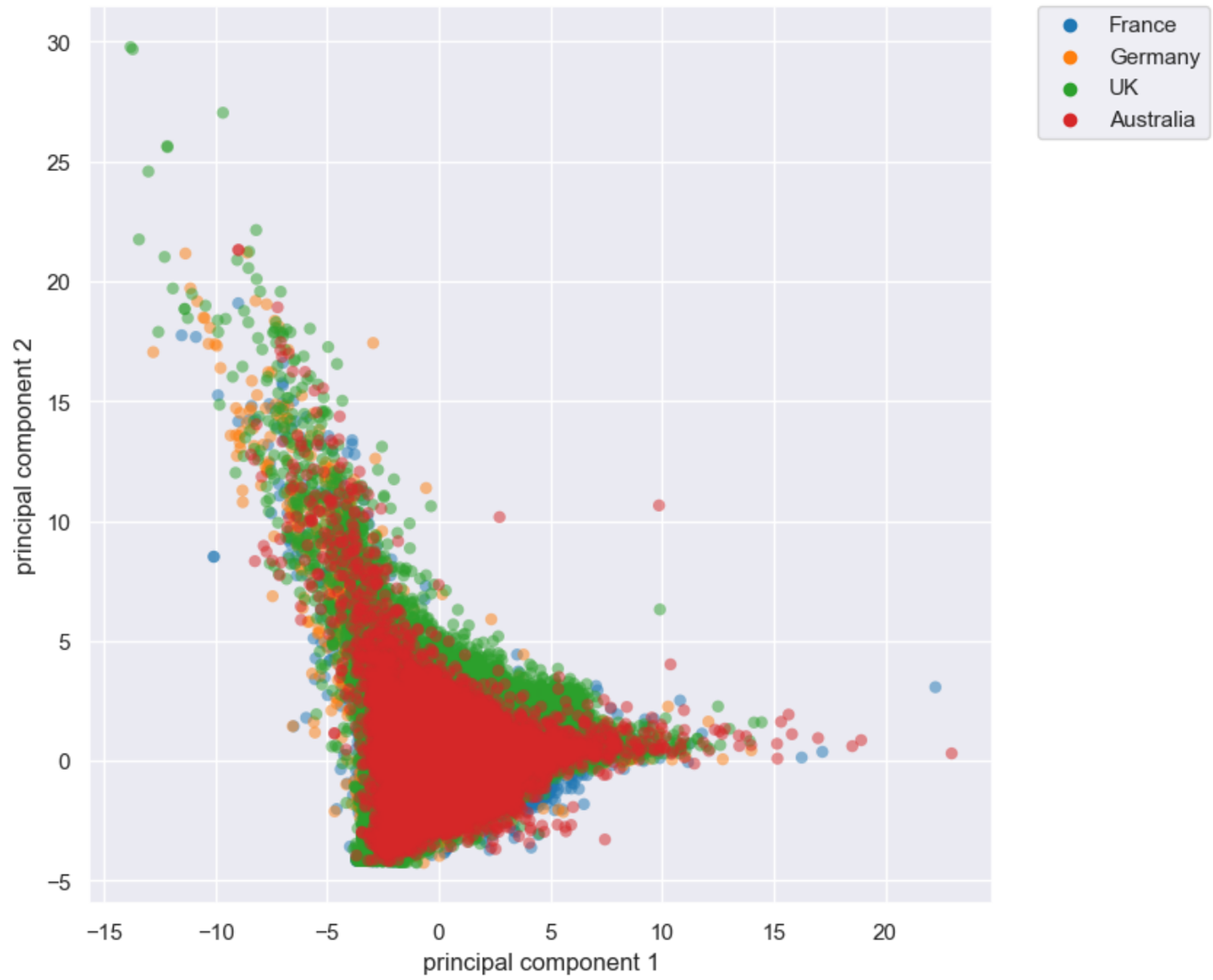
Principal components analysis

```
$ hatch pca \

    --features vaf tumour_strand_bias NORMAL_DP TUMOUR_DP gnomadAF \

    --hue country \

    all_variants.pon.csv
```



Near future work

- More plot types:
    - *clustered* heat maps
    - linear regression
    - pair plots
- Support more options on existing plots
- Better error messages and sanity checking
- Some common statistical calculations:
    - ANOVA, correlation, *etc*
- Official release with multiple installation options:
    - PyPI
    - (bio)conda
    - Docker

Development

- Github:

  https://github.com/bjpop/hatch

- I'm very keen for users and also collaborators.