RoughPy

An excuse for me to play with words

Sam Morley
University of Oxford

DataSig

A rough path between
mathematics and data science

The Alan Turing Institute

Imperial College London

Engineering and Physical Sciences Research Council

Which of the following has the largest proportion of anagrams?

- English
- French
- Spanish
- German
- Lithuanian
- Russian

Python + Rough Paths = RoughPy

RoughPy is a toolkit for working with streaming data through the lens of rough paths.

Design goals

1. Provide a class that represents streaming data as a rough path - we call it a stream.
   - Query over intervals to get a signature.
   - Intelligently cache intermediate results.
   - Provide an abstraction over the underlying data.
2. Provide classes for the algebraic objects: free tensors, shuffle tensors, and objects from the free Lie algebra
3. Provide utilities for working with intervals and other useful tools.
4. Should provide easy interoperability with standard libraries.

An Example

First, import RoughPy:

```
>>> import roughpy as rp
```

Make some data - converted the word “stream” into a stream of letters:

```
>>> import numpy as np
>>> data = np.array([
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]],
    dtype='int8')
```

Get an algebra context

- Width 26 - size of the alphabet
- Depth 2 - moderate depth, is usually enough
- Rational coefficients for infinite precision

```
>>> ctx = rp.get_context(width=26,
                         depth=2,
                         coeffs=rp.Rational)
```

Construct the stream

Construct the stream using the `LieIncrementStream.from_increments` constructor:

```
>>> stream = rp.LieIncrementStream.
                  from_increments(data, ctx=ctx)
```

Compute the signature of the whole stream:

```
>>> sig = stream.signature()
>>> sig
FreeTensor(width=26, depth=2, ctype=Rational)
```

```
>>> print(sig)
{ 1() 1(1) 1(5) 1(13) 1(18) 1(19) 1(20) 1/2(1,1)
  1(1,13) 1(5,1) 1/2(5,5) 1(5,13) 1/2(13,13)
  1(18,1) 1(18,5) 1(18,13) 1/2(18,18) 1(19,1)
  1(19,5) 1(19,13) 1(19,18) 1/2(19,19) 1(19,20)
  1(20,1) 1(20,5) 1(20,13) 1(20,18) 1/2(20,20) }
```

Or the log signature

```
>>> print(stream.log_signature())
{ 1(1) 1(5) 1(13) 1(18) 1(19) 1(20) -1/2([1,5])
  1/2([1,13]) -1/2([1,18]) -1/2([1,19]) -1/2([1,20])
  1/2([5,13]) -1/2([5,18]) -1/2([5,19]) -1/2([5,20])
  -1/2([13,18]) -1/2([13,19]) -1/2([13,20])
  -1/2([18,19]) -1/2([18,20]) 1/2([19,20]) }
```

Expanding slightly


```
>>> happy = word_to_stream("happy")
>>> birthday = word_to_stream("birthday")
>>> terry = word_to_stream("terry")
```

Make a new stream whose increments are the the log signatures of the three above streams.

```
>>> data = some_magic_to_make_an_array_from_logsigs(
...     happy.log_signature(),
...     birthday.log_signature(),
...     terry.log_signature())
>>> new_ctx = rp.get_context(26, 4, rp.Rational)
>>> stream = from_increments(data, ctx=new_ctx)
>>> sig = stream.signature()
>>> print(sig.size(), '/'. sig.dimension())
6344 / 475255
```

Which of the following has the largest proportion of anagrams?

- English $\approx 23.8\%$
- French $\approx 17.3\%$
- Spanish $\approx 17.7\%$
- German $\approx 8.7\%$
- Lithuanian $\approx 29.5\%$
- Russian $\approx 12.9\%$