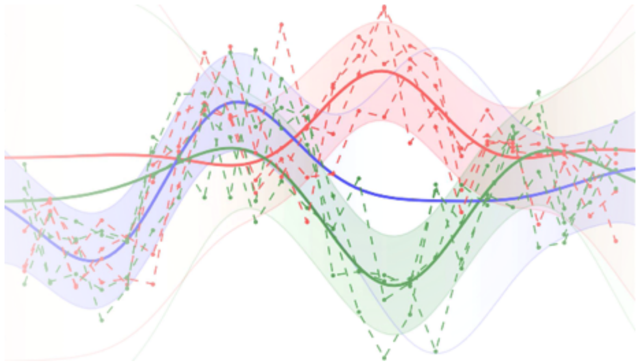
SAPIENZA
UNIVERSITÀ DI ROMA

Giovanni Stilo, Ph.D.
stilo@di.uniroma1.it

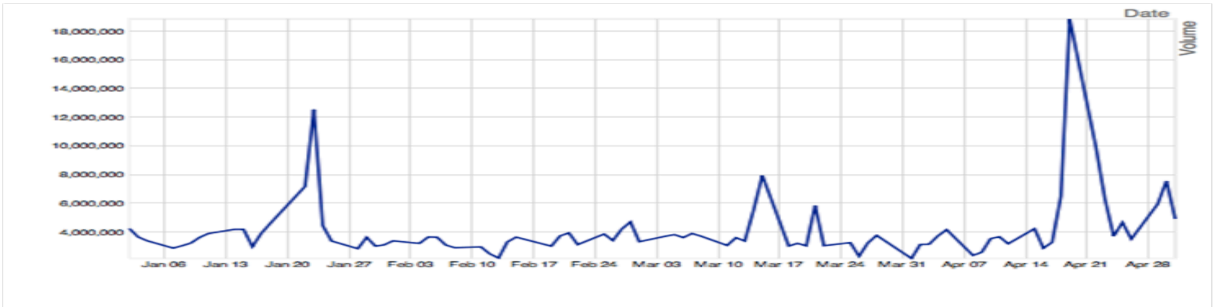


Follow the Flow.

Terms based Timeseries.

Time series

- A **time series** is a sequence of data points; typically measured at successive points in time, spaced at uniform time intervals.
- **Examples:** of time series are the daily closing value of the Dow Jones Industrial Average
- Time series are very frequently **plotted** via line charts.



Daily trading volumes of IBM (Jan-Apr 2013)

- Time series are used in **statistics**, **signal processing**, **pattern recognition**, **econometrics**, **mathematical finance**.

Parts

- Time Series are composed:
  - Time
  - Value

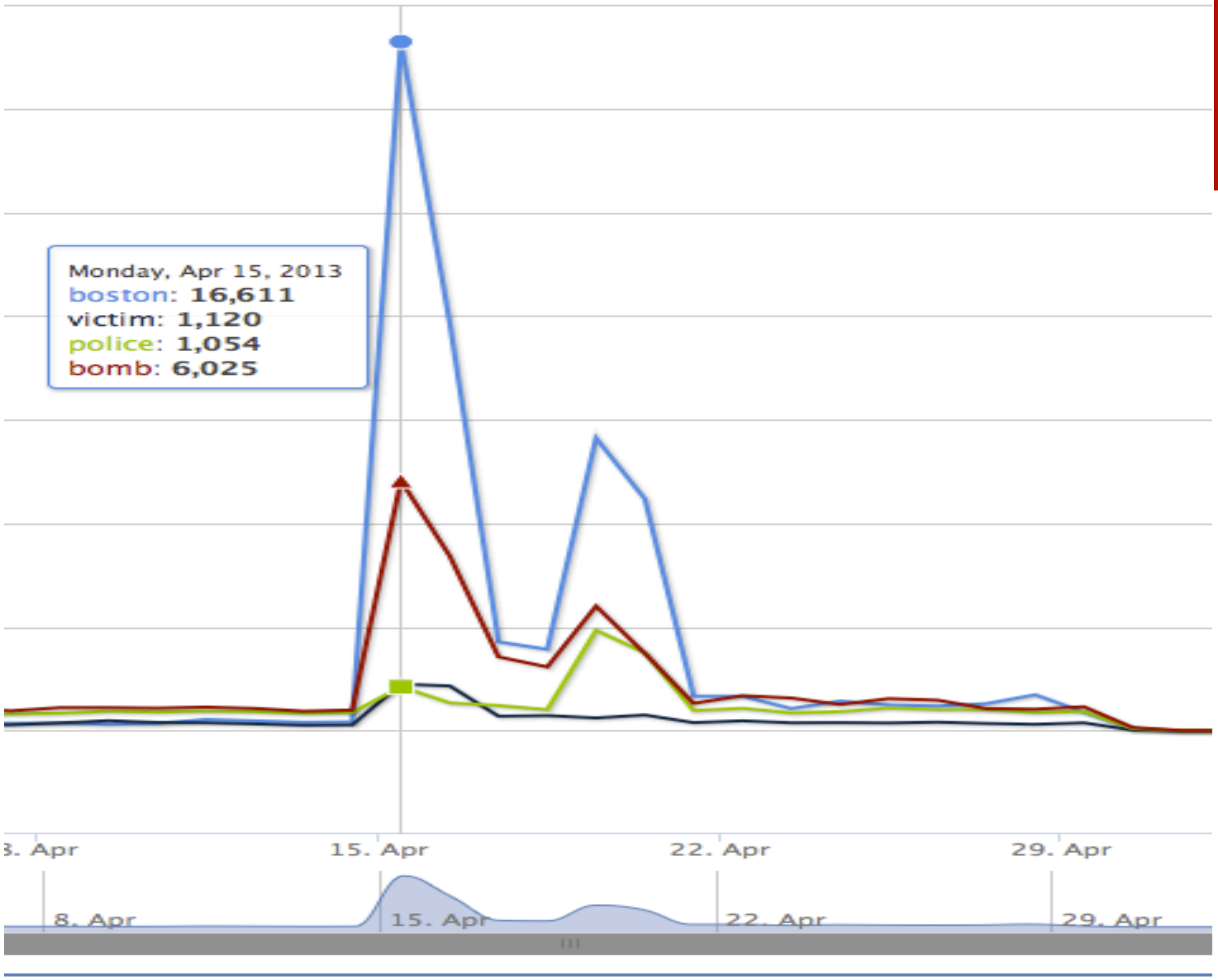
- Two values array(one for time aspect and one for data aspect) is the simplest rappresentation.

Term Time Series

- Term Time Series are time series that keep track of the number of occurences of single term over time.

- Normally are fixed interval Time Series (minutes, hour, day interval).

Term Time Example



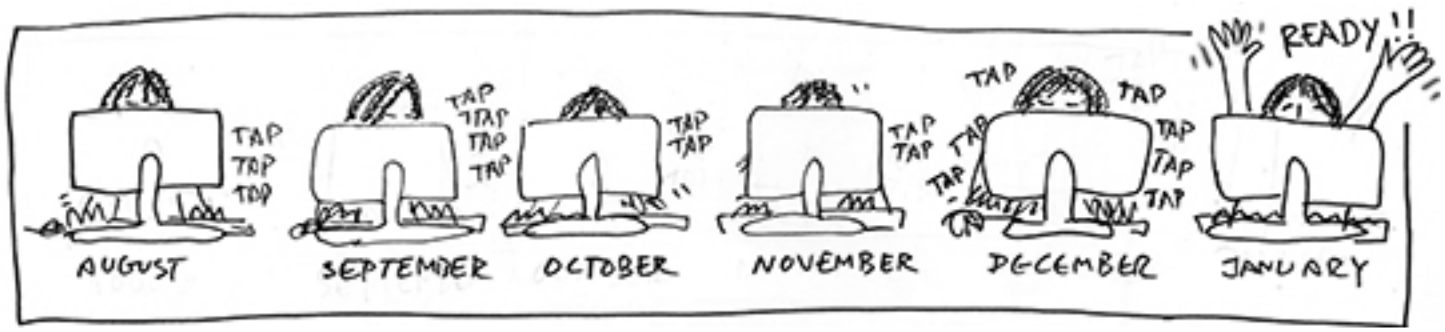
How To

- If we have a temporal aspect in a document (for example Tweets timestamp) we should count all the documents that contain that term $\boldsymbol{w}_j$ in a specific time interval $\boldsymbol{t}_i$.
- That produce the value of $\boldsymbol{w}_j$ in $\boldsymbol{t}_i$.

Class Work (1)

- Using the streaming api (Twitter4J) collect the stream that keep track of the following "target" terms:
  - Renzi
  - Grillo
  - Berlusconi
  - Di Battista
  - Meloni
- For each "target" term create (at realtime) the relative time series.
  Using 2 minutes time fixed interval.
- Design your own implementation for a time series representation.

Let’s Try?!?!


TAP TAP TAP
AUGUST
TAP TAP TAP TAP
SEPTEMBER
TAP TAP
OCTOBER
TAP TAP
NOVEMBER
TAP TAP TAP TAP TAP TAP TAP
DECEMBER
READY!!
JANUARY

Class Work (2)

- Using the streaming api (Twitter4J) collect the stream that keep track of the following "target" terms:
  - Renzi
  - Grillo
  - Berlusconi
  - Di Battista
  - Meloni
- Build a Lucene index for all tweets.
- For each "target" extract from the Lucene index the relative time series.
  Fixed time interval should be chosen later.

SAX String

- To convert a Time Series to a SAX string

```
...
    int alphabetSize = 2;
    double nThreshold = 0.01;
    // instantiate classes
    NormalAlphabet na = new NormalAlphabet();
    SAXProcessor sp = new SAXProcessor();

    // data Vector
    double[] ts = {10, 20, 20, 50, 80, 10, 50, 80, 10, 5};

    // perform the discretization
    SAXRecords res = sp.ts2saxByChunking(ts, ts.length,
                              na.getCuts(alphabetSize), nThreshold);

    // print the output
    String sax = res.getSAXString("");
    System.out.println(sax);
    System.out.println(sax.matches("a+b+a*b*a*"));
...
```

Maven Artifacts

```
<dependency>
   <groupId>net.seninp</groupId>
   <artifactId>jmotif-sax</artifactId>
   <version>1.1.1</version>
</dependency>
```

Class Work (2)

- Using the streaming api (Twitter4J) collect the Public stream (1%) and get only the Italian Tweets.
- For each term create the relative time series. Using 1 minutes time (delta) fixed interval for 10 minutes.
- Discard all time series below a fixed threshold (1000 tweets)
- Convert each time series to SAX and filter it if not satisfy the regex:
  - ***a+b+a*b*a****
- Group together all terms that show the same SAX String.

Class Work (3)

- Reading Tweets from the collection.
  And Create a Lucene Index for any(sliding) window of 10 hours in the dataset.
- For each window consider terms with a minimum number of 10000 occurrences.
- create the relative time series using 1 hour time (delta) fixed interval for 10 hours.
- Convert each time series to SAX and filter it if not satisfy the regex:
  - ***a+b+a*b*a****
- Group together all terms that show the same SAX String.

List Files

```
File folder = new File(dirPath);


File[] listOfFiles = folder.listFiles();

for (File file : listOfFiles) {

      if (file.isFile()) {

            System.out.println(file.getName());
            // DO SOMETHING;
      }
}
```

Reading File

```
FileInputStream fstream =
                        new FileInputStream(file);
GZIPInputStream  gzStream =
                        new GZIPInputStream(fstream);
InputStreamReader isr = new InputStreamReader(gzStream)
BufferedReader br =
                        new BufferedReader(isr );

        String line;

        //Read File Line By Line
        while ((line = br.readLine()) != null) {
            // Print the content on the console
            System.out.println(line);
        }

        br.close();
```

Reading Index Lexicon

```
Fields fields = MultiFields.getFields(index);
Terms terms = fields.terms(defaultField);

TermsEnum iterator = terms.iterator(null);
BytesRef byteRef = null;
while ((byteRef = iterator.next()) != null) {
String t = new String(byteRef.bytes, byteRef.offset,
  byteRef.length);
 if ((t.length() > 2) && (t.length() < 15) &&
  StringUtils.isAlpha(t) && (iterator.docFreq() >
  minFrq)) {

      // CREATE TIMESERIES

  }
}
```

Get Stats

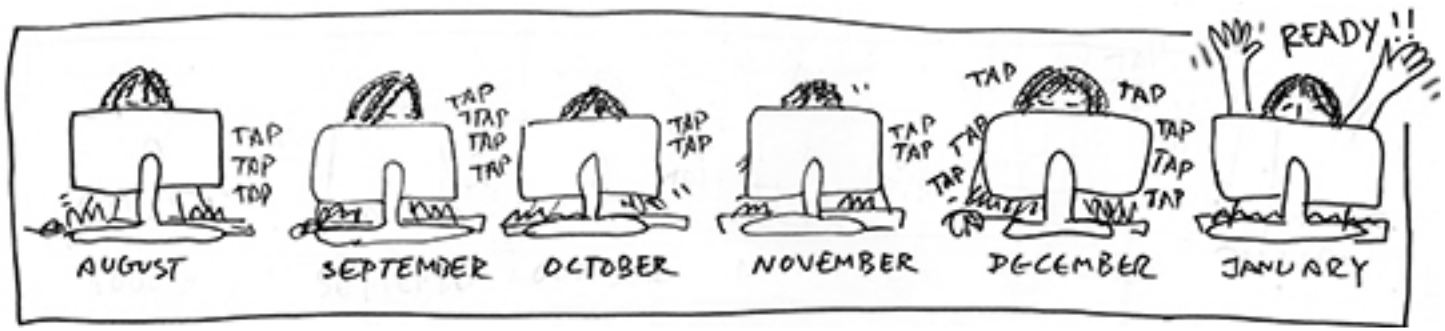
```
TotalHitCountCollector collector = new
  TotalHitCountCollector();

FieldCacheRangeFilter<Long> dateFilter =
  FieldCacheRangeFilter.newLongRange(”TIMEFIELD",
start, end, true, false);

searcher.search(myQuery, dateFilter, collector);

collector.getTotalHits();
```

Let's Try?!?!


TAP TAP TAP
AUGUST
TAP TAP TAP TAP
SEPTEMBER
TAP TAP
OCTOBER
TAP TAP
NOVEMBER
TAP TAP TAP TAP TAP TAP TAP
DECEMBER
READY!!
JANUARY

GIT

- http://rogerdudler.github.io/git-guide/

& clone

- https://github.com/giovanni-stilo/G