Towards Geographic, Demographic, and Climatic Hypotheses Exploration on COVID-19 Spread - An Open source Johns Hopkins University Time Series Normalisation, and Integration Layer

Version 1.1 - June 10, 2020

Karim Baïna

Alqualsadi research team (Innovation on Digital and Enterprise Architectures)
ADMIR Laboratory, Rabat IT Center,
ENSIAS, University Mohammed V in Rabat,
BP 713 Agdal, Rabat, Morocco
karim.baina@um5.ac.ma

Abstract. Epidemiologist, Scientists, Statisticians, Historians, Data engineers and Data scientists are working on finding descriptive models and theories to explain COVID-19 expansion phenomena or on building analytics predictive models for learning the apex of COVID-19 confirmed cases, recovered cases, and deaths evolution time series curves. In CRISP-DM life cycle, 75% of time is consumed only by data preparation phase causing lot of pressures and stress on scientists and data scientists building machine learning models. This paper aims to help reducing data preparation efforts by presenting detailed data preparation repository with shell and python scripts for formatting, normalising, and integrating Johns Hopkins University COVID-19 daily data via three normalisation user stories applying data preparation at lexical, syntactic & semantics and pragmatic levels, and four integration user stories through geographic, demographic, climatic, and distance based similarity dimensions, among others. This paper and related open source repository will help data engineers and data scientists aiming to deliver results in an agile analytics life cycle adapted to critical COVID-19 context.

Key words: *Coronavirus, SARS-CoV-2, COVID-19, 2019-nCoV, Data Engineering, Data preparation, Data normalisation, Data integration.*

1 Introduction

Johns Hopkins University has provided a github repository with, among others, daily fresh data about COVID-19 pandemic confirmed cases, recovered cases, and deaths evolution [1]. Epidemiologist, Scientists, Statisticians, Historians, Data engineers and Data scientists are working on finding models and theories to explain and predict COVID-19 expansion phenomena. Our paper aims to help reducing data engineers and data scientists data preparation efforts, by presenting detailed data preparation layer for formatting, normalising, and integrating Johns Hopkins University COVID-19 daily data via three normalisation user stories applying data nomalisation at lexical, syntactic & semantics and pragmatic levels, and four, among others, integration user stories through geographic, demographic, climatic, and distance based similarity. Our data integration and analytics approach for this paper, and for handling COVID-19 crisis in general is building Minimum Viable Model, Platform, and Data Product through agile analytics [2].

2 COVID-19 Data Normalisation

The current data normalisation step is based on a Linux/Unix data preparation Shell script that formats Johns Hopkins University COVID-19 three files : time_series_covid19_confirmed_global.csv, time_series_covid19_deaths_global.csv, or time_series_covid19_recovered_global.csv to be ready for data analytics according to four different user stories.

To set the data normalisation goals and to enable its broad use, requirements will be formalised as user stories.

- Data Engineer **User Story 1 : Data Format Normalisation (***Lexical level***)** : modify countries and dates values to enable time series dates arithmetic, and scalable data post-processing in a notebook or a data flow language.

- Data Engineer/Scientist **User Story 2 : Data Representation Normalisation (***Syntactic & Semantic level***)** : merge province/state with country/region columns and remove not null values to be compliant to unique key database constraint, visual load (and relatively optimise storage), and represent countries evolution shapes of confirmed case, death, or recovered case on a absolute standard real date time axis for temporal relational databases/-multidimensional data warehouses/NoSQL data stores loading and analysis.

- Data Scientist **User Story 3 : Data Temporal Normalisation (***Pragmatic level***)** : shift left all countries COVID-19 time series to their D_0 (Date of first not-null Value either for confirmed, deaths or recovered), and rename all date columns names as $D_0..D_n$ to represent countries evolution shapes of confirmed case, death, or recovered case on a **relative abstract date time axis**.

2.1 User Story 1 - Data Format Normalisation (–*Lexical level*)

As a Data Engineer, I would like to be able to :

1. remove or replace specific characters,
2. and format dates columns names,
3. *with keeping column names line (first data row).*

In order to :

1. enable time series dates arithmetic
2. and enable scalable data post-processing in a notebook or a data flow language

Intent and Motivation The aim of this user story is to transform COVID-19 data set file by (i) removing '"', '*' characters, (ii) replacing non separator ',' by '-' character (e.g. "Korea, South" → Korea-South, Taiwan* → Taiwan) important for normalising country names necessary in further join queries with other country keyed data sets, (iii) normalising column dates names into "%m/%d/%Y" date formatted columns (eg. 3/2/20 → 03/02/2020) enabling further date arithmetic and manipulation (e.g. duration calculations, manipulate programatically Date D_0 of first COVID-19 of a country, or Date of n^{th} death, etc.), with (iv) keeping first columns names line.

Applicability A Data engineer needs format normalisation '*user story 1*' for :

- performing join queries between COVID-19 data set and other country keyed data sets.
- performing date arithmetic operations like difference between two dates or addition/subtraction of a period to a given date necessary for advanced database time windows queries, temporal data warehouse drill-down and roll-up, or time series value predictions.

Listing 1.1: User Story 1 - Data Format Normalisation in Bourne Shell

Code in sh

```
specific=$1

INPUT_JHU_FILE=./COVID-19/csse_covid_19_data/
    csse_covid_19_time_series/time_series_covid19_${
    specific}_global.csv

sed "s/, /-/" ${INPUT_JHU_FILE} |
sed "s/\"//g" |
sed "s/\*//" | sed -E "s/\,(.)\//,0\1\//g" |
sed -E "s/\/(.)\//\/0\1\//g" |
sed -E "s/\/20([^/])/\/2020\1/g" |
sed -E "s/\/20$/\/2020/g"| sed -E "s/,($)/,0\1/g" > ./
    output_data/time_series_covid19_${specific}_global-
    us1-normalisation.csv
```

Output file name The user story output file name is as follows ./output_data/time_series_covid19_$1_global-us1-normalisation.csv depending on script passed parameter either confirmed, deaths or recovered.

Figure 1 shows a sample of last week before April 3rd, 2020 of time_series_covid19_deaths_global-us1-normalisation.csv file sorted alphabetically by Country/Region column (ascending order), and lexically transformed by user story 1.

2.2 User Story 2 - Data Representation Normalisation (*–Syntactic & Semantic level*)

As a Data Engineer/Scientist, I would like to be able to

1. merge the two first data set key columns to create a composite primary key (*–Syntactic level*),
2. remove null values (*–Semantic level*),
3. *with and without column names line (first data row).*

In order to :

1. compare countries evolution shapes of confirmed case, death, or recovered case on a same **absolute date time axis**.
2. application scenarios :
   (a) *sub-Scenario 2.1 :* enable for example to load the data set in a spread sheet with column names line.
   (b) or *sub-Scenario 2.2 :* enable loading databases (see [3] for COVID-19 HBase NoSQL storage, and Hive SQL querying application case studies) with all row except the column names line.

Province/State	Country/Region	Lat	Long	03/28/2020	03/29/2020	03/30/2020	03/31/2020	04/01/2020	04/02/2020	04/03/2020
	Afghanistan	33	65	4	4	4	4	4	6	6
	Albania	41.1533	20.1683	10	10	11	15	15	16	17
	Algeria	28.0339	1.6596	29	31	35	44	58	86	105
	Andorra	42.5063	1.5218	3	6	8	12	14	15	16
	Angola	-11.2027	17.8739	0	2	2	2	2	2	2
	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	0	0	0
	Argentina	-38.4161	-63.6167	18	19	23	27	28	36	39
	Armenia	40.0691	45.0382	1	3	3	3	4	7	7
Australian Capital Territory	Australia	-35.4735	149.0124	0	0	1	1	1	1	1
New South Wales	Australia	-33.8688	151.2093	8	8	8	8	9	10	12
Northern Territory	Australia	-12.4634	130.8456	0	0	0	0	0	0	0
Queensland	Australia	-28.0167	153.4	1	2	2	2	2	4	4
South Australia	Australia	-34.9285	138.6007	0	0	0	0	0	0	0
Tasmania	Australia	-41.4545	145.9707	0	0	0	1	2	2	2
Victoria	Australia	-37.8136	144.9631	3	4	4	4	4	5	7
Western Australia	Australia	-31.9505	115.8605	2	2	2	2	2	2	2
	Austria	47.5162	14.5501	68	86	108	128	146	158	168
	Azerbaijan	40.1431	47.5769	4	4	4	5	5	5	5
	Bahamas	25.0343	-77.3963	0	0	0	0	1	1	1
	Bahrain	26.0275	50.55	4	4	4	4	4	4	4
	Bangladesh	23.685	90.3563	5	5	5	5	6	6	6
	Barbados	13.1939	-59.5432	0	0	0	0	0	0	0
	Belarus	53.7098	27.9534	0	0	0	1	2	4	4
	Belgium	50.8333	4	353	431	513	705	828	1011	1143
	Benin	9.3077	2.3158	0	0	0	0	0	0	0
	Bhutan	27.5142	90.4336	0	0	0	0	0	0	0
	Bolivia	-16.2902	-63.5887	0	1	4	6	7	8	9
	Bosnia and Herzegovina	43.9159	17.6791	5	6	10	13	13	16	17
	Brazil	-14.235	-51.9253	111	136	159	201	240	324	359

Fig. 1: A sample of April 3rd 2020 JHU data transformed by user story 1

Intent and Motivation The aim of this user story is to transform COVID-19 data set file to a sparse file by (i) keeping only not null values (e.g. for better Matrix visualisation in spreadsheets, or optimised NoSQL storage), (ii) merging the two first columns to form a composite key separated by a '~' character (i.e. concatenating province/state with country/region separated by '~' character.), with (iii) keeping first columns names line.

Applicability A Data engineer/scientist needs representation normalisation '*user story 2*' for :

- providing compliance with databases unique key constraint.
- enabling date manipulation for programming languages/scripts and for temporal relational databases/multidimensional data warehouses/NoSQL data stores.
- decreasing visual load when manipulating a COVID-19 numerical matrix, enabling clean curves tracing, and relatively optimised storage on disk and memory.
- representing and comparing countries evolution shapes of confirmed case, death, or recovered case on a absolute standard real date time axis.

Code in sh

Listing 1.2: User Story 2/sub-Scenario 2.1 - Data Representation Normalisation in Bourne Shell

```
sed "s/,0,/,,/g" ./output_data/time_series_covid19_${
    specific}_global-us1-normalisation.csv |
sed -E "s/([,]+)0,/\1,/g" |
sed -E "s/,0($)/,\1/" |
sed "s/^,/~/" |
sed -E "s/([a-z A-Z]+),([a-z A-Z]+)/\1~\2/" > ./
    output_data/time_series_covid19_${specific}_global-
    sparse-with-formatted-column-names-us2-1-
    normalisation.csv
```

Listing 1.3: User Story 2/sub-Scenario 2.2 - Data Representation Normalisation in Bourne Shell

```
tail -n +2 ./output_data/time_series_covid19_${
    specific}_global-sparse-with-formatted-column-names
    -us2-1-normalisation.csv > ./output_data/
    time_series_covid19_${specific}_global-sparse-us2
    -2-normalisation.csv
```

Output file name The user story output file name is as follows :

– for *sub-Scenario 2.1* : ./output_data/time_series_covid19_$1_global-sparse-with-formatted-column-names-us2-1-normalisation.csv depending on script passed parameter either confirmed, deaths or recovered.

– for *sub-Scenario 2.2* : ./output_data/time_series_covid19_$1_global-sparse-us2-2-normalisation.csv depending on script passed parameter either confirmed, deaths or recovered.

Figure 2 shows a sample of last two weeks before April 3rd, 2020 of time_series_covid19_confirmed_global-sparse-with-formatted-column-names-us2-1-normalisation.csv file generated by user story 2/sub-Scenario2.1 syntactic and semantic rules, and sorted on April 3rd, 2020 (last column) (descending order) under a visual spreadsheet.

Figure 3 shows a sample of first two weeks after the detection of first COVID-19 confirmed case in United States January 22nd, 2020 of the same time_series_covid19_confirmed_global-sparse-with-formatted-column-names-us2-1-normalisation.csv file generated by user story 2/sub-Scenario2.1 , and sorted on last April 3rd, 2020 column (descending order). One may remark sparse representation.

2.3 User Story 3 - Data Temporal Normalisation (–*Pragmatic level*)

As a Data Scientist, I would like to be able to

1. shift all countries COVID-19 time series to D_0 (Date of first not-null Value either for confirmed, deaths or recovered)
2. rename all date columns names as $D_0..D_n$.

In order to :

1. compare countries evolution shapes of confirmed case, death, or recovered case on a **relative date time axis**.

Intent and Motivation The aim of this user story is to transform COVID-19 data set file to a relative time axis by shifting left all countries COVID-19 time series (i.e. rows values) to D_0 (Date of first not-null Value either for confirmed, deaths or recovered), and and renaming all date columns names as $D_0..D_n$ necessary for comparing evolution shapes independently of their real first date for confirmed cases, deaths, or recovered cases.

Two different (respectively equivalent) values $v_{i,k}$ and $v_{j,k}$ for two countries (i.e. row) i and j mean that at their same COVID-19 evolution k^{th} logical day D_k, the two countries are at different (respectively equivalent) levels for confirmed, number of deaths or recovered cases. However, D_k real Dates for the two countries i and j are physically different (a different columns indexed values semantics/context for each country (row) dependently to their relative D_0 real date – pragmatics).

Applicability A Data scientist needs temporal normalisation '*user story 3*' for:

- representing, comparing analytically and visually, countries evolution shapes of confirmed case, death, or recovered case on a same **relative abstract date time axis**. *Especially, when one knows that there are reference COVID-19 critical and lucky evolution shapes.*
- compute distances between comparable windows of two countries evolution time series of different D_0 (see section 3.4).

Code in sh [1]

Listing 1.4: User Story 3 - Data Temporal Normalisation in Bourne Shell

```
./days.sh `head -n 1 ./output_data/time_series_covid19_${specific}_global-sparse-with-formatted-column-names-us2-1-normalisation.csv |
sed "s/,/\t/g"` > ./output_data/time_series_covid19_${specific}_global-sparse-shifted-to-D0-us3-normalisation.csv

cat ./output_data/time_series_covid19_${specific}_global-sparse-us2-2-normalisation.csv |
sed -E "s/([,]+)((,[0-9]+)((,[0-9]*)*))/\2\1/" |
sed "s/^~//" |
sed "s/~/-/" >> ./output_data/time_series_covid19_${specific}_global-sparse-shifted-to-D0-us3-normalisation.csv
```

Listing 1.5: days.sh shell script for renaming all date columns names to $D_0..D_n$.

```
suite=""
limit=`expr $# - 4`
for i in $(seq 0 $limit) ; do
  suite="$suite,D$i"
done
echo "$1,$2,$3$suite"
```

[1] simple `sed -E "s/([,]+)((,[0-9]+)+))/\2\1/"` regular expression translation command would be more logical than `sed -E "s/([,]+)((,[0-9]+)((,[0-9]*)*))/\2\1/"`, however some dirty data contain null values (i.e. ",,"

Output file name Depending on script passed parameter either confirmed, deaths or recovered, the user story output file name is as follows ./output_data/time_series_covid19_$1_global-sparse-shifted-to-D0-us3-normalisation.csv.

Figure 4 shows a sample of April 3rd, 2020 time_series_covid19_deaths_global-sparse-shifted-to-D0-us3-normalisation.csv file generated by user story 3 pragmatic rule with regards to each country D_0, and sorted on D24 column (descending order) under a visual spreadsheet.

2.4 Normalisation Technical aspects

- **Normalisation OS & Software Installation Prerequisites :** (1) A Linux/Unix environment is requested for running the data Normalisation shell script, and (2) git tools are necessary for pulling data.
- **Normalisation Data Collection Prerequisites :** COVID-19 Data should be collected each day from Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE) github repository (e.g around 2.00 am GMT+1).

Listing 1.6: Pulling COVID-19 Data

```
#kbaina is my local home directory, change it to your
    home directory or another directory
cd /home/kbaina/

git clone https://github.com/CSSEGISandData/COVID-19.
    git COVID-19/

cd ./COVID-19/

git pull
```

- **Normalisation Input :** The unique data Normalisation script parameter is a value among 'confirmed', 'deaths' or 'recovered', used for naming pulled gihub Johns-Hopkins University data source file :
  ./COVID-19/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_$1_global.csv
  (either time_series_covid19_confirmed_global.csv,
  time_series_covid19_deaths_global.csv,
  or time_series_covid19_recovered_global.csv).
- **Normalisation script Name :** normalisation_flow.sh
- **Normalisation script Language :** Bourne Shell (sh).
- **Normalisation script Calls Scenarios :**

sparse coding) between two not null values (e.g. Iceland, Kazakhstan death values, or Australia-Northern territory confirmed values.

Listing 1.7: Scenario 1. Run all 3 normalisation user stories on time_series_covid19_confirmed_global.csv

```
chmod u+x ./normalisation_flow.sh

./normalisation_flow.sh confirmed
```

Listing 1.8: Scenario 2. Run all 3 normalisation user stories on time_series_covid19_deaths_global.csv

```
./normalisation_flow.sh deaths
```

Listing 1.9: Scenario 3. Run all 3 normalisation user stories on time_series_covid19_recovered_global.csv

```
./normalisation_flow.sh recovered
```

3 COVID-19 Data Integration

After normalising data, Data scientists may need to explain and test eventual correlations hypotheses between COVID-19 evolution and propagation and different features either geographic, demographic, climatic or only visualise COVID-19 data on a specific region either economic, political, organisational or cooperation based. In the following, some integration steps are proposed in order to enrich previously normalised COVID-19 data with with additional features.

3.1 User Story 1 - Geographic Data Integration

As a Data Engineer/Scientist, I would like to be able to :

1. add geographic features to quantitative time series,

In order to :

1. enable time series exploration, sorting, aggregation and analysis by geographic dimension.

Intent and Motivation The aim of this user story is to position previoulsy normalised COVID-19 data within each continent. In fact, each country row representing related times series (either confirmed, recovered or deaths) is cobbled with its continent using United Nations open data source [4]. [2]

[2] To be compliant with JHU data, we added to this list three new items : Taiwan,

Applicability A Data engineer needs geographic integration '*user story 1*' for:

- performing geographic abstraction on COVID-19 evolution.
- performing spread simulation over time between geographic regions according to their interactions (e.g economic exchange, and international mobility, etc.).
- performing geographic analysis about COVID-19 with additional regional based health, social, and economic indicators.

Figures 5 and 6, and 7 show examples, among others, of visualisation based on geographic dimension data integration, that the author published on social networks.

3.2 User Story 2 - Demographic Data Integration

As a Data Engineer/Scientist, I would like to be able to :

1. add demographic features to quantitative time series,

In order to :

1. enable time series exploration, sorting, aggregation and analysis by demographic dimension.

Intent and Motivation The aim of this user story is to integrate each country time series of previoulsy normalised COVID-19 data with demographic information among population size, density, land area, migrants, fertility rate, medium age, urban population rate as provided by [5].

Figure 8 shows an example, among others, of visualisation showing an illustrative demograhic data integration, that the author published on social networks.

3.3 User Story 3 - Climatic Data Integration

As a Data Engineer/Scientist, I would like to be able to :

1. add climatic features to quantitative time series,

In order to :

1. enable time series exploration, sorting, aggregation and analysis by climatic dimension.

Laos, and West Bank and Gaza

Intent and Motivation Knowing that climatic correlation hypothesis on spread and transmission of COVID-19 is an interesting research spot [6–8], the aim of this user story is to convert raw latitude data (as one of the most important factors determining the climate) into useful climatic zones features.

In fact, two climatic zones classifications have been integrated to the COVID-19 previously normalised data (1) seven zones classification according to [9] (*Zone 1 - Polar, Zone 2 - Arctic, Zone 3 - Subarctic, Zone 4 - Midlatitude, Zone 5 - Subtropical, Zone 6 - Tropical, and Zone 7 - Equatorial*), and (2) three zones classification according to [10] (*Zone 1 - Frigid, Zone 2 - Temperate, and Zone 3 - Torrid*). Tables 1 and 2 show used climatic zones altitude conversion rules.

North Polar	[75° (N) , 90° (N)]
Arctic	[60° (N) , 75° (N) [
Subarctic	[55° (N) , 60° (N) [
Midlatitude	[35° (N) , 55° (N) [
Subtropical	[25° (N) , 35° (N) [
Tropical	[10° (N) , 25° (N) [
Equatorial	[10° (S) , 10° (N) [
Tropical	[25° (S) , 10° (S) [
Subtropical	[35° (S) , 25° (S) [
Midlatitude	[55° (S) , 35° (S) [
Subantarctic	[60° (S) , 55° (S) [
Antarctic	[75° (S) , -60° (S) [
South Polar	[-90° (S) , 75° (S) [

Table 1: Climatic Zones conversion rules according to [9]

Climatic Zone	**Latitude Interval**	**Earth Surface rate**
North Frigid	[66.5° (N) Arctic Circle , 90° (N) North Pole]	4.12%
North Temperate	[23.5° (N) Tropic of Cancer , 66.5° (N) Arctic Circle [	25.99%
Torrid	[23.5° (S) Tropic of Capricorn , 23.5° (N) Tropic of Cancer [	39.78%
South Temperate	[66.5° (S) Antarctic Circle , 23.5° (S) Tropic of Capricorn [	25.99%
South Frigid	[90° (S) South Pole , 66.5° (S) Antarctic Circle [	4.12%

Table 2: Climatic Zones conversion rules according to [10]

Figure 9 shows an example, among others, of visualisation based on climatic dimension data integration, that the author published on social networks.

3.4 User Story 4 - Similarity based Data Integration

As a Data Engineer, I would like to be able to :

1. compute distances between quantitative time series,

In order to :

1. enable similarity based analysis of time series sets.

Intent and Motivation The aim of this user story is to compute a distance measure (described in algorithms 1, and 2) between time series enabling Data scientists to analyse eventual clustering polarities in terms of times series evolution shapes focused on monitored countries.

Let ts_1, and ts_2 be two COVID-19 evolution time series either for confirmed, recovered or deaths (i.e. two rows of output file produced by user story 1 - see section 2.1). Figure 1 shows a simple fragment of targeted time series. The function $distance(ts_1, ts_2)$ (see algorithm 1) computes quantitative distance between two countries evolution COVID-19 time series (i) shifted to the same D_0 (Date of first not-null Value - either for confirmed, deaths or recovered) (see section 2.3 for information about this temporal normalisation), and (ii) restricted to the minimum size of both time series.

Algorithm 1 Time series distance algorithm

procedure $distance(ts_{country_1}, ts_{country_2})$ ▷ computes a distance measure between two time series
  $day0_1 = min(\ \arg\min_{day} ts_{country_1}(day)\ )$
  ▷ $\arg\min_d ts(d) = \{x \mid ts(d) = \min_{d'} ts(d')\}$ retuns all day-indexes minimising ts time series. ▷ So $day0_i$ will be the first not null value day-index of a time series $ts_{country_i}$ (i.e. $ts_{country_i}$ day0 index).
  $day0_2 \leftarrow min(\ \arg\min_{day} ts_{country_2}(day)\ )$
  $window \leftarrow |ts_{country_1}| - max(\ day0_1, day0_2\ )$
  ▷ $|ts|$ returns number of ts columns (size of oldest time series), which is the same for all COVID-19 time series of the same family (confirmed, recovered or deaths).
  $sum \leftarrow \sum_{d_1=day0_1, d_2=day0_2}^{d_1=day0_1+window\ -\ 1, d_2=day0_2+window\ -\ 1} (ts_{country_1}(d_1) - ts_{country_2}(d_2))^2$
  $distance[country_1][country_2] \leftarrow \sqrt{sum}$
  $distance[country_2][country_1] \leftarrow distance[country_1][country_2]$
end procedure

Applicability A Data scientist needs integration '*user story 4*' for :

- adding countries names as rows in monitored_countries.csv file, so distance of those monitored countries is computed with all countries.

Algorithm 2 Time series distance Computing for all monitored countries

```
procedure computing_similarities_with(monitored_countries)  ▷ Computing distance for all countries time series with monitored countries time series
    for country_i in countries do
        for country_j in monitored_countries do
            distance(ts_{country_i}, ts_{country_j})
        end for
    end for
end procedure
```

- ◦ tracking COVID-19 evolution in some specific countries that have neither geographic, nor demographic, or climatic relations.
- ◦ discovering countries time series similar to those tracked countries.

Figure 10 show an example, among others, of visualisation based on similarity data integration, that the author published on social networks.

3.5 User Story 5 - Regional, Political, Economic, or Organisational Data Integration

As a Data Engineer/Scientist, I would like to be able to :

1. add additional (Regional, Political, Economic, or Organisational) features to quantitative time series,

In order to :

1. enable time series exploration, sorting, aggregation and analysis by such contextual dimensions.

Intent and Motivation The aim of this user story is to integrate the previously normalised COVID-19 data with additional file sources to supervise specific group of coutries like : the Group of Seven [11], the League of Arab States [12], the Euro Zone [13], the Schengen Area [14], and the Islamic World Educational, Scientific and Cultural Organization (ICESCO) members [15]. Many oher other organisations can be integrated EU countries, commonwealth countries, BRICS (Brazil, Russia, India, China, South Africa), etc.

Applicability A Data scientist needs integration Regional, Political, Economic, or Organisational '*user story 5*' for :

- ◦ analysing geostrategically COVID-19 spread in some countries with specific regional, political, economic, or organisational relations beside geographic, demographic, and climatic or similarity based dimensions.

Figures 11 and 12 show examples, among others, of visualisation based on political, and organisational dimension data integration, that the author published on social networks.

3.6 Integration Technical aspects

- **Integration OS & Software Installation Prerequisites :** (1) Normalisation OS & Software Installation Prerequisites, and (2) python 3 [3].
- **Integration Prerequisites :** COVID-19 Data should be normalised according to user story 1 producing time_series_covid19_$1_global-us1-normalisation.csv output file.
- **Integration Input :** (1) The unique data Integration script parameter is a value among 'confirmed', 'deaths' or 'recovered', used for naming prepared Johns-Hopkins University data source file through our nomalisation user story 1 :
  time_series_covid19_$1_global-us1-normalisation.csv
  (either time_series_covid19_confirmed_global-us1-normalisation.csv,
  time_series_covid19_deaths_global-us1-normalisation.csv,
  or time_series_covid19_recovered_global-us1-normalisation.csv). Also,
  (2) one may update monitored_countries.csv file.
- **Integration script Name :** integration_flow.py, and full_dataflow.sh
- **Integration script Language :** Bourne Shell (sh), and Python.
- **Integration script Calls Scenarios :**

Listing 1.10: Scenario 1. Run all 5 integration user stories on time_series_covid19_confirmed_global-us1-normalisation.csv

```
chmod u+x ./full_dataflow.sh

./full_dataflow.sh  confirmed
```

Listing 1.11: Scenario 2. Run all 5 integration user stories on time_series_covid19_deaths_global-us1-normalisation.csv

```
./full_dataflow.sh  deaths
```

Listing 1.12: Scenario 3. Run all 5 integration user stories on time_series_covid19_recovered_global-us1-normalisation.csv

```
./full_dataflow.sh recovered
```

- **Output file name :**
  Operationally, the five integration user stories are run all together, and the integration user stories output file names reflect this behaviour as follows :

[3] This paper integration python code has been tested with version 3.8.2

In fact, either the five integration user stories are run (i) on top of normalisation user story 1 and produces ./output_data/time_series_covid19_$–specific"_formatted-us1-normalisation-with-full-integration.csv

or (ii) on top of normalisation user story 2 (sub-scenario 2.1) and produces ./output_data/time_series_covid19_$–specific"_formatted-sparse-us2-1-normalisation-with-full-integration.csv

or (iii) on top of normalisation user story 3 and produces , and ./output_data/time_series_covid19_'$–specific"_formatted-sparse-shifted-to-D0-us3-normalisation-with-full-integration.csv depending on ${specific} integration script parameter with values either confirmed, deaths or recovered.

Listing 1.13: full_dataflow.sh shell script running end to end normalisation shell and Python integration dataflow

Code in sh

```sh
#!/bin/sh

if [ $# -eq 0 ]
then
  echo "syntax ./$0 recovered | confirmed | deaths"
  exit
else
  case "$1" in
   "recovered" | "confirmed" | "deaths")
    specific=$1
    ;;
   *)
    echo "syntax ./$0 recovered | confirmed | deaths"
    exit
    ;;
esac
fi

# pre-processing : normalisation
./normalisation_flow.sh ${specific}

# processing : integration
./integration_flow.py ${specific} > ./output_data/time_series_covid19_${specific}_formatted-us1-normalisation-with-full-integration.csv

# post-processing :
head -n 1 ./output_data/time_series_covid19_${specific}_formatted-us1-normalisation-with-full-integration.csv > ./output_data/time_series_covid19_${specific}_formatted-sparse-us2-1-normalisation-with-full-integration.csv
./days_bis.sh `head -n 1 ./output_data/time_series_covid19_${specific}_formatted-us1-normalisation-with-full-integration.csv` > ./output_data/time_series_covid19_${specific}_formatted-sparse-shifted-to-D0-us3-normalisation-with-full-integration.csv
tail -n +2 ./output_data/time_series_covid19_${specific}_formatted-us1-normalisation-with-full-integration.csv |
sed "s/, 0,/,,/g"  |
sed  -E "s/([,]+) 0,/\1,/g" |
sed -E "s/, 0($)/,\1/" | tee ./output_data/time_series_covid19_${specific}_formatted-sparse.tmp.csv |
sed -E "s/([,]+)((,[ 0-9]+)((,[ 0-9]*)*))/\2\1/" >> ./output_data/time_series_covid19_${specific}_formatted-sparse-shifted-to-D0-us3-normalisation-with-full-integration.csv
cat ./output_data/time_series_covid19_${specific}_formatted-sparse.tmp.csv >>  ./output_data/time_series_covid19_${specific}_formatted-sparse-us2-1-normalisation-with-full-integration.csv
```

4 Conclusion & Discussion

Aiming to deliver in an agile analytics life cycle adapted to critical COVID-19 context, this paper presents detailed (i) three user stories applying data preparation at lexical, syntactic & semantics and pragmatic levels as data preparation helpers for data engineers and data scientists, and (ii) four integration user stories through geographic, demographic, climatic, and distance based similarity dimensions, among others. A normalisation and integration repository provides open source Shell and Python code. Table 3 synthesizes the new integration features this paper added to JHU data after normalisation phase. [4]

There are *some limitations to this paper :* (i) using shell and standalone python scripts instead of integrated Notebooks, (ii) executing manually and generating manually graphs on a spread sheet instead of an automated end-to-end devops delivery dataflow, (iii) no statistical or machine learning ideas have been developing neither for describing data nor for predicting times series Apex, or clustering all countries, and (iv) According to Data scientists analysis needs, more other countries related features can be additionally integrated related to massive testing, non-pharmaceutical public health measures, universal BCG vaccination policies, or countries hydroxychloroquine related protocol for treating COVID-19.

The author is aware of all those limitations. In fact, for the two first limitation, an agile analytics style with Minimum Viable Data Product approach has been followed [2]. This approach prioritises results quality, with respecting critical delays delivery (like those of COVID-19 crisis) with no time for rhetoric discussions neither on technical platform versions, or distributions of standalone versus clouded Notebooks, or automated devops delivery design, and/or cloud object storages with different APIs and related security for data file storage, and no time for porting from one environment to another, etc. However, one may need advanced toolkits for advanced data post-processing using data structures, utility packages, and execution kernels not provided by shell scripting languages and runtime environments. With regards to the two last limitations, the author has prepared the normalisation, and integration layer to build, as a perspective, an analytical layers integrating machine, and deep learning capabilities.

This work aims to help scientists and data scientists shortening data preparation phase which is time consuming cording to CRISP-DM life cycle specialists. It is to be taken as a leveraging bootstrap for specific data preparation phase in COVID-19 analytics Big Data projects targeting for instance to integrate COVID-19 evolution time series with medical/biology best practices, COVID-19 mutations, scientific papers results, or to study correlations between COVID-19 time series with humidity data, people telco mobility during countries lockdown phases, or to analyse recurrent COVID-19 contamination causality, or to study similarities with other historical pandemics evolution data like SARS-

[4] Due to integration constraints some country names of the table's cited sources, used as mapping keys, have been adapted to JHU normalised data form (an example among many others : *Czech Republic* becomes *Czechia*, or *Myanmar* becomes *Burma*)

integration dimension	new features	master data file or integration rules set	source
general	#days_since_day_zero : size (age) of the time series		
geographic	continent	countries_by_continents.csv	[4]
demographic	population, density, medium_age, urban_population_rate, rate_by_1M_of_inhabitants	demography.csv	[5]
climatic	climatic_zone_I, climatic_zone_II : 3 & 7 climatic zones classification	climatic zones conversion rules of tables 1 & 2	[9, 10]
regional, organisational, political, or economic coalitions	arab_country : Is a League of Arab States member ?	arab_countries.csv	[12]
	icesco_country : Is an ICESCO Organisation member ?	icesco_countries.csv	[15]
	eurozone_country : Is a Euro Zone member ?	euro_zone_countries.csv	[13]
	G7_country : Is a G7 Zone member ?	G7_countries.csv	[11]
	schengen_country : Is a Schengen Area member ?	schengen_countries.csv	[14]
specific country cluster discovery	monitored_country : Is a monitored member ? D0_greater_or_equal_than_country$_i$: age based order relation distance_with_country$_i$: *computing_similarities_with(monitored_countries)*	monitored_countries.csv	

Table 3: Synthesis of new features resulted from Integration of normalised JHU data with external sources

CoV, MERS-COV, or to compare evolution with spreading information from social networks, etc. The more integration you do on the data generated by the author scripts with other data sets (e.g. continents, median age, population, testing numbers, virus contamination rates, etc.), the more features you will have and the more this work will leverage your COVID-19 data experience. Hurry Up, and share your experience for the world scientists.

5 Appendix : How to download open source normalisation and integration scripts, master data files, and materials of this paper ?

To download continuously data engineering Shell & Python scripts, master data files and reporting graphs discussed in this paper, you can access, and clone the author gitlab repository at [16].

Acknowledgment

Acknowledgement must go to Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE) for keeping up to date world wide COVID-19 data available in a daily frequency.

Acknowledgement must go to The Ministry of National Education, Higher Education, Staff Training, and Scientific Research, Morocco for accepting and supporting my sabbatical leave to do research, and return to ENSIAS refreshed. I also acknowledge my colleagues at ENSIAS maintaining the superb teaching and learning and e-learning culture in the school in my absence especially during COVID-19 crisis.

References

1. Johns-Hopkins, U.: Novel coronavirus (covid-19) cases, provided by jhu csse. https://github.com/CSSEGISandData/COVID-19 (2020)
2. Tran, D.: What is minimum viable (data) product ? https://medium.com/idealo-tech-blog/\what-is-minimum-viable-data-product-49269e338d85 (2018)
3. Baïna, K.: Leveraging Data Preparation, HBase NoSQL Storage, and HiveQL Querying for COVID-19 Big Data Analytics Projects (2020)
4. United-Nation Statistics Division: Standard country or area codes for statistical use (M49). https://unstats.un.org/unsd/methodology/m49/overview/ (2019)
5. Worldometers: World population, population by country. https://www.worldometers.info/world-population/population-by-country/ (2020)
6. Wang, J., Tang, K., Feng, K., Lv, W.: High temperature and high humidity reduce the transmission of covid-19. Available at SSRN 3551767 (2020)

7. Sajadi, M.M., Habibzadeh, P., Vintzileos, A., Shokouhi, S., Miralles-Wilhelm, F., Amoroso, A.: Temperature and latitude analysis to predict potential spread and seasonality for covid-19. Available at SSRN 3550308 (2020)
8. Araujo, M.B., Naimi, B.: Spread of sars-cov-2 coronavirus likely to be constrained by climate. medRxiv (2020)
9. Wikipedia: Geographical Zones. https://en.wikipedia.org/wiki/Geographical_zone (2020)
10. EarthOnlineMedia: Geographical Zones. https://www.earthonlinemedia.com/ebooks/tpe_3e/essentials/geographical_zones.html (2020)
11. Wikipedia: Group of Seven. https://en.wikipedia.org/wiki/Group_of_Seven (2020)
12. Arab-League: League of Arab States. https://en.wikipedia.org/wiki/Arab_League (2020)
13. Wikipedia: Euro Zone. https://en.wikipedia.org/wiki/Eurozone (2020)
14. Wikipedia: Schengen Area. https://en.wikipedia.org/wiki/Schengen_Area (2020)
15. Wikipedia: Islamic World Educational, Scientific and Cultural Organization (ICESCO). https://en.wikipedia.org/wiki/Islamic_Educational,_Scientific_and_Cultural_Organization (2020)
16. Baïna, K.: Novel coronavirus (covid-19) data engineering. https://gitlab.com/kbaina/COVID-19 (2020)

Province/State~Country/Region	Lat	Long	03/22/2020	03/23/2020	03/24/2020	03/25/2020	03/26/2020	03/27/2020	03/28/2020	03/29/2020	03/30/2020	03/31/2020	04/01/2020	04/02/2020	04/03/2020
~US	37.0902	-95.7129	33276	43847	53740	65778	83836	101657	121478	140886	161807	188172	213372	243453	275586
~Italy	43	12	59138	63927	69176	74386	80589	86498	92472	97689	101739	105792	110574	115242	119827
~Spain	40	-4	28768	35136	39885	49515	57786	65719	73235	80110	87956	95923	104118	112065	119199
~Germany	51	9	24873	29056	32986	37323	43938	50871	57695	62095	66885	71808	77872	84794	91159
Hubei~China	30.9756	112.2707	67800	67800	67801	67801	67801	67801	67801	67801	67801	67801	67802	67802	67802
~France	46.2276	2.2137	16018	19856	22304	25233	29155	32964	37575	40174	44550	52128	56989	59105	64338
~Iran	32	53	21638	23049	24811	27017	29406	32332	35408	38309	41495	44605	47593	50468	53183
~United Kingdom	55.3781	-3.436	5683	6650	8077	9529	11658	14543	17089	19522	22141	25150	29474	33718	38168
~Turkey	38.9637	35.2433	1236	1529	1872	2433	3629	5698	7402	9217	10827	13531	15679	18135	20921
~Switzerland	46.8182	8.2275	7474	8795	9877	10897	11811	12928	14076	14829	15922	16605	17768	18827	19606
~Belgium	50.8333	4	3401	3743	4269	4937	6235	7284	9134	10836	11899	12775	13964	15348	16770
~Netherlands	52.1326	5.2913	4204	4749	5560	6412	7431	8603	9762	10866	11750	12595	13614	14697	15723
~Austria	47.5162	14.5501	3582	4474	5283	5588	6909	7657	8271	8788	9618	10180	10711	11129	11524
~Korea-South	36	128	8961	8961	9037	9137	9241	9332	9478	9583	9661	9786	9887	9976	10062
~Portugal	39.3999	-8.2245	1600	2060	2362	2995	3544	4268	5170	5962	6408	7443	8251	9034	9886
~Brazil	-14.235	-51.9253	1546	1924	2247	2554	2985	3417	3904	4256	4579	5717	6836	8044	9056

Fig. 2: A sample of April 3rd 2020 JHU data transformed by user story 2 with last two weeks column names line and sorted on April 3rd, 2020

Province/State~Country/Region	Lat	Long	01/22/2020	01/23/2020	01/24/2020	01/25/2020	01/26/2020	01/27/2020	01/28/2020	01/29/2020	01/30/2020	01/31/2020	02/01/2020	02/02/2020	02/03/2020
~US	37.0902	-95.7129	1	1	2	2	5	5	5	5	5	7	8	8	11
~Italy	43	12										2	2	2	2
~Spain	40	-4											1	1	1
~Germany	51	9						1	4	4	4	5	8	10	12
Hubei~China	30.9756	112.2707	444	444	549	761	1058	1423	3554	3554	4903	5806	7153	11177	13522
~France	46.2276	2.2137			2	3	3	3	4	5	5	5	6	6	6
~Iran	32	53													
~United Kingdom	55.3781	-3.436										2	2	2	2
~Turkey	38.9637	35.2433													
~Switzerland	46.8182	8.2275													
~Belgium	50.8333	4													
~Netherlands	52.1326	5.2913													
~Austria	47.5162	14.5501													
~Korea-South	36	128	1	1	2	2	3	4	4	4	4	11	12	15	15
~Portugal	39.3999	-8.2245													
~Brazil	-14.235	-51.9253													

Fig. 3: A sample of April 3rd 2020 JHU data transformed by user story 2 with last two weeks column names line and sorted on April 3rd, 2020

Province/State~Country/Region	Lat	Long	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25
Spain	40	-4	1	2	3	5	10	17	28	35	54	55	133	195	289	342	533	623	830	1043	1375	1772	2311	2808	3647	4365	5138	5982
Italy	43	12	1	2	3	7	10	12	17	21	29	34	52	79	107	148	197	233	366	463	631	827	827	1266	1441	1809	2158	2503
Hubei-China	30.9756	112.2707	17	17	24	40	52	76	125	125	162	204	249	350	414	479	549	618	699	780	871	974	1068	1068	1310	1457	1596	1696
United Kingdom	55.3781	-3.436	1	2	2	3	4	6	8	8	8	21	21	55	55	71	137	177	233	281	335	422	465	578	759	1019	1228	1408
Germany	51	9	2	2	3	3	7	9	11	17	24	28	44	67	84	94	123	157	206	267	342	433	533	645	775	920	1107	1275
Netherlands	52.1326	5.2913	1	1	3	3	4	5	5	10	12	20	24	43	58	76	106	136	179	213	276	356	434	546	639	771	864	1039
US	37.0902	-95.7129	1	1	6	7	11	12	14	17	21	22	28	36	40	47	54	63	85	108	118	200	244	307	417	557	706	942
Iran	32	53	2	2	4	5	8	12	16	19	26	34	43	54	66	77	92	107	124	145	194	237	291	354	429	514	611	724
Switzerland	46.8182	8.2275	1	1	1	2	2	3	4	4	11	13	14	14	27	28	41	54	75	98	120	122	153	191	231	264	300	359
Korea-South	36	128	1	2	2	6	8	10	12	13	13	16	17	28	28	35	35	42	44	50	53	54	60	66	66	72	75	75
Egypt	26	30	1	1	1	1	1	2	2	2	2	4	6	6	8	10	14	19	20	21	24	30	36	40	41	46	52	58
Morocco	**31.7917**	**-7.0926**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**2**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**6**	**11**	**23**	**25**	**26**	**33**	**36**	**39**	**44**	**48**	
Iraq	33	44	2	2	3	4	6	6	7	7	8	9	10	10	10	11	12	13	17	17	20	23	27	29	36	40	42	42
France	46.2276	2.2137	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	4	4	6	9	11	19	19	33	48
British Columbia-Canada	49.2827	-123.1207	1	1	1	1	1	1	1	4	4	7	7	8	10	10	13	13	13	14	14	17	17	19	24	24	31	31
Argentina	-38.4161	-63.6167	1	1	1	1	1	2	2	2	2	2	2	3	3	4	4	4	6	8	9	13	18	19	23	27	28	36
San Marino	43.9424	12.4578	1	1	1	1	1	1	1	2	2	3	5	5	5	7	7	11	11	14	20	20	20	21	21	21	21	22
Henan-China	33.882	113.614	1	1	1	2	2	2	2	2	2	2	2	2	3	4	6	6	7	8	10	11	13	13	16	19	19	19
Lebanon	33.8547	35.8623	1	3	3	3	3	3	3	3	3	4	4	4	4	4	4	6	6	8	8	10	11	12	14	16	17	
Heilongjiang-China	47.862	127.7615	1	1	1	1	1	1	2	2	2	2	2	2	2	3	3	5	6	7	8	8	9	11	11	11	11	11

Fig. 4: A sample of April 3rd 2020 JHU data transformed by user story 3 and sorted on D24

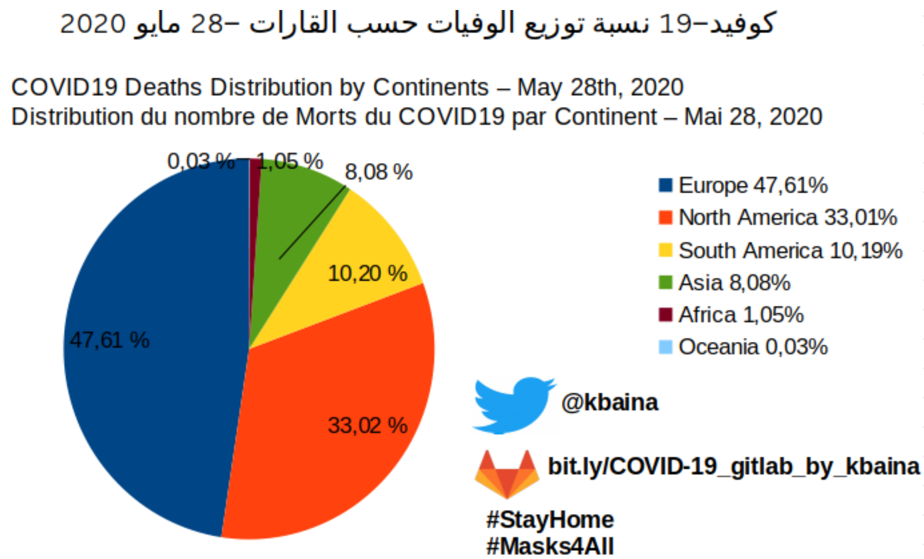


Fig. 5: An example of visualisation based on geographic dimension : COVID19 deaths distribution by continents on May 28th, 2020

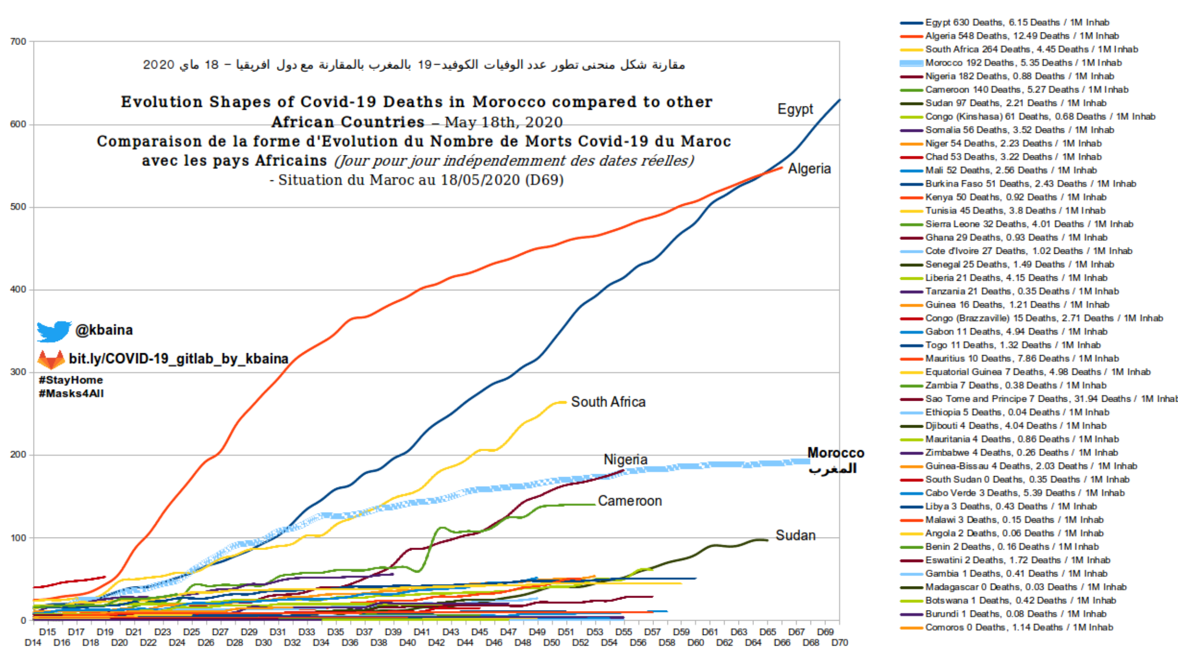


Fig. 6: An example of visualisation based on geographic dimension : Evolution Shapes of COVID19 deaths of Africa continent countries on May 28th, 2020

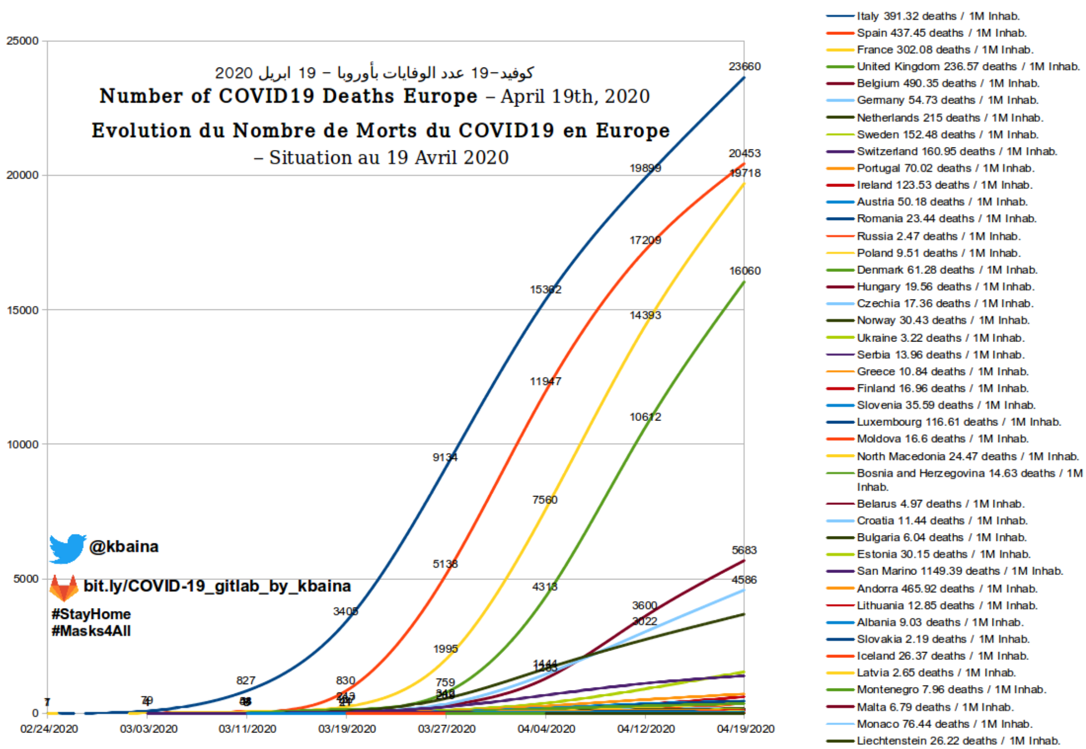
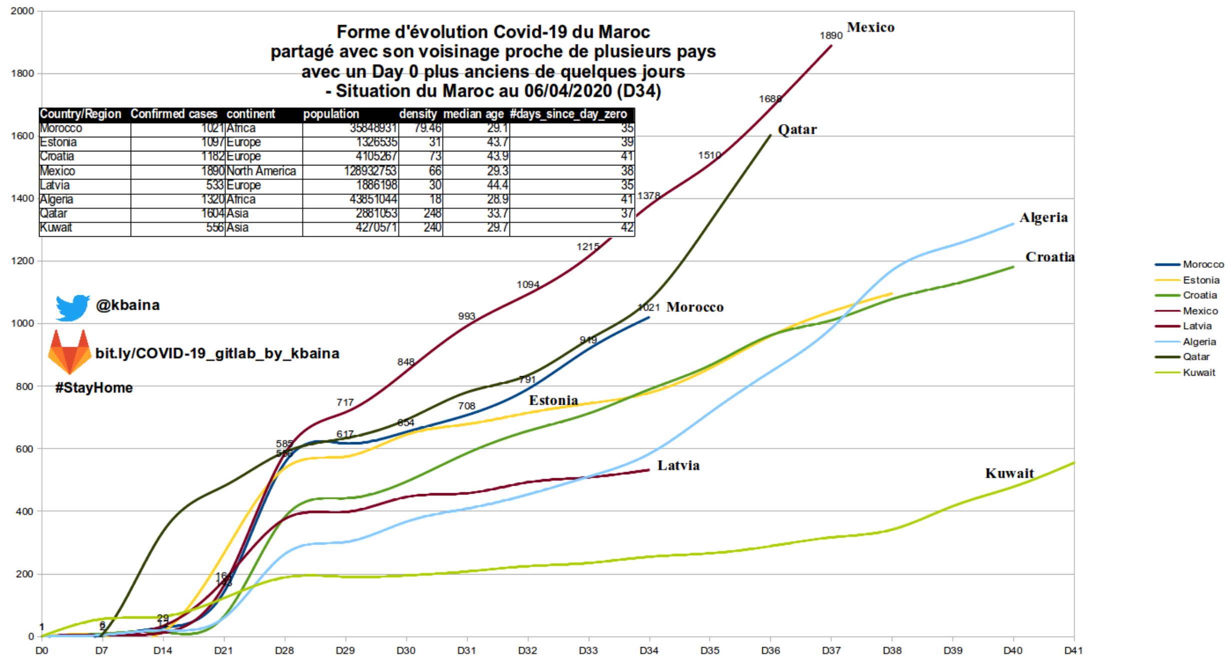


Fig. 7: An example of visualisation based on geographic dimension : Evolution Shapes of COVID19 deaths of European continent on April 19th, 2020



Country/Region	Confirmed cases	continent	population	density	median age	#days_since_day_zero
Morocco	1021	Africa	35848931	79.46	29.1	35
Estonia	1097	Europe	1326535	31	43.7	39
Croatia	1182	Europe	4105267	73	43.9	41
Mexico	1890	North America	128932753	66	29.3	38
Latvia	533	Europe	1886198	30	44.4	35
Algeria	1320	Africa	43851044	18	28.9	41
Qatar	1604	Asia	2881053	248	33.7	37
Kuwait	556	Asia	4270571	240	29.7	42

Fig. 8: An example of visualisation showing an illustrative demograhic data : COVID19 confirmed cases with Morocco similar time series on April 6th, 2020

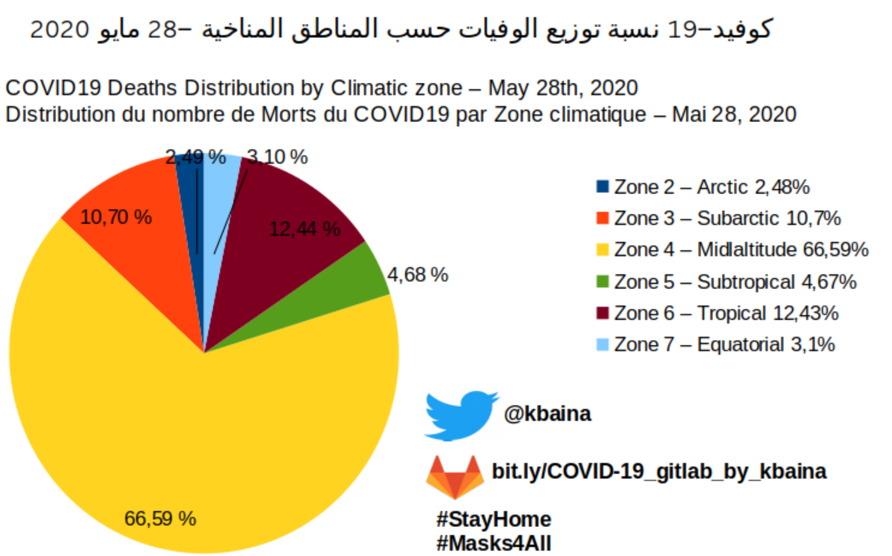


Fig. 9: An example of visualisation based on climatic dimension : COVID19 deaths distribution by climatic zones on May 28th, 2020

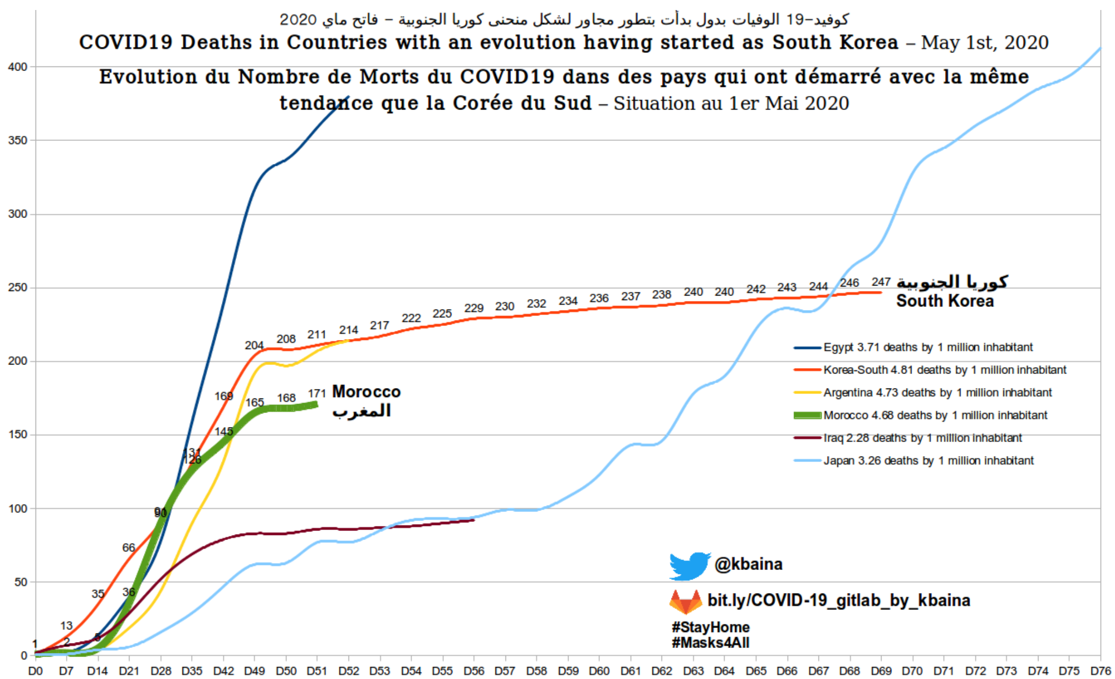


Fig. 10: An example of visualisation based on similarity based data integration : COVID-19 deaths in countries like Morocco and Argentina among others with the same shape (distance based) as South Korea on May 1st, 2020

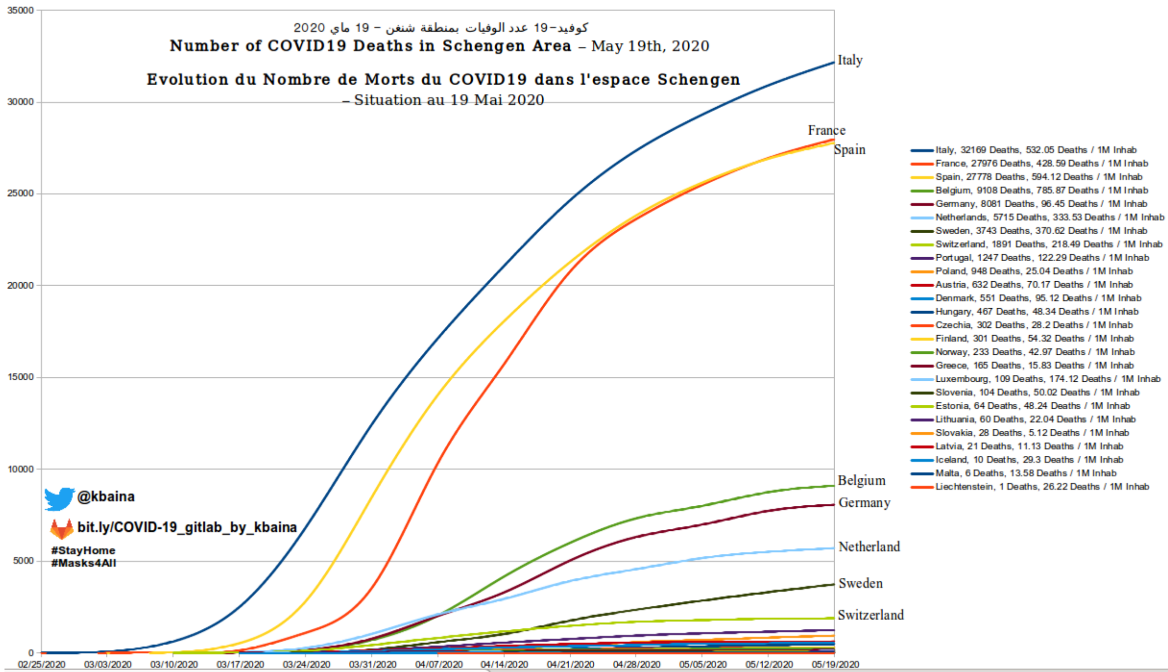


Fig. 11: An example of visualisation based on geographic dimension : COVID-19 deaths evolution in Schengen Area on May 19th, 2020

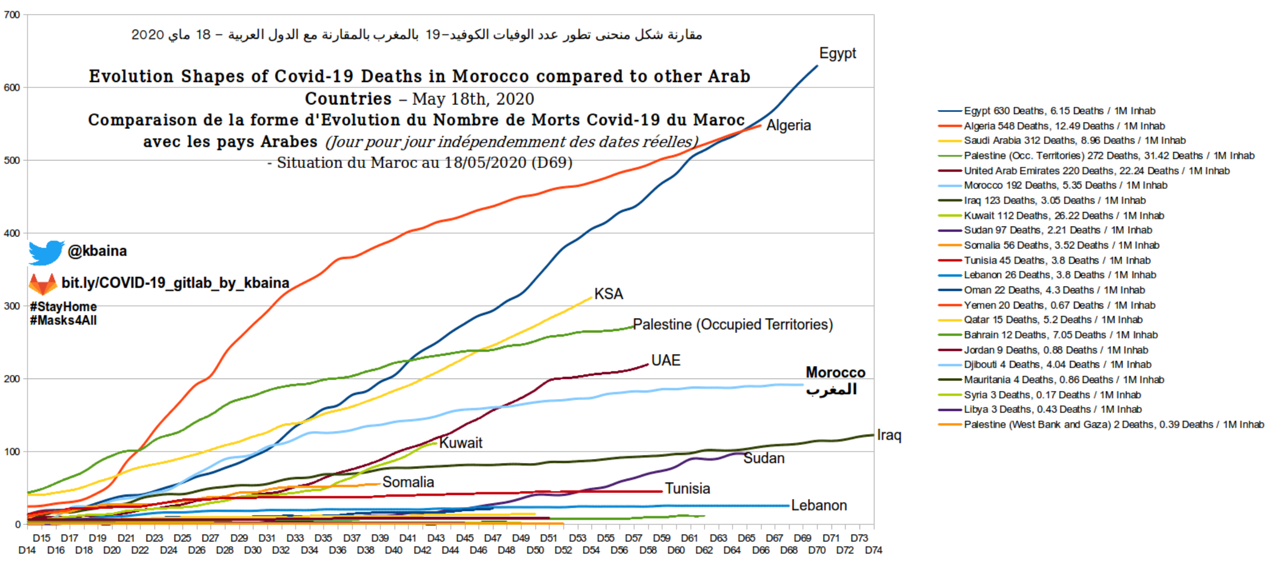


Fig. 12: An example of visualisation based on geographic dimension : COVID-19 deaths evolution in Schengen Area on May 19th, 2020