About [covid19\_inference\_forecast](https://github.com/Priesemann-Group/covid19_inference_forecast)/[scripts](https://github.com/Priesemann-Group/covid19_inference_forecast/tree/master/scripts)/[paper](https://github.com/Priesemann-Group/covid19_inference_forecast/tree/master/scripts/paper)/Corona\_germany\_simple\_model.ipynb

It is a simple model (without change points) used in the paper **Inferring COVID-19 spreading rates and potential change points for case number forecasts.**

“the simpler model,which assumed a stationary λ and was fitted only with data until March 15th (Fig. 1 A).” (p. 9)

Structure of the code:

Run the simulation:

In[5]

[ **Import** common Python libraries and modules : pandas, numpy, matplotlib.pyplot, scipy.stats, matplotlib, pickle

[**Input data**: Load the data (confirmed\_cases\_url, deaths\_url) from John Hopkins datacentre.

**Output & data**: Variable to save figures and data.

path\_to\_save = '../../figures/'

path\_data = '../../data/'

In[2] Construction of inferred new reported case

[**def** delay\_cases() with:

Arguments:

And output **inferred\_cases**

In[3]: Forecasting

[Imported libraries and modules : pymc3 (on Bayesian statistical modeling), theano.tensor, theano (for manipulating and evaluating mathematical expressions, especially matrix-valued ones), datetime, time

[Initial and final time conditions of the fitting and the prediction.:

Output (run the sim):

Plotting:

In [38]

Output (plotting):

2 lines: effective m: 1.283 +- 0.043

λ: 0.407 [0.317, 0.513]

and 6 figures:

Figs joined:

In [42]

Output (Figs joined):

2 lines: 3012.8599625379575

(25, 25)

and 10 figures.

Main construction of In[2]: delay\_cases

Arguments/parameters of delay\_cases

new\_I\_t,

len\_new\_I\_t,

len\_new\_cases\_obs ,

delay,

delay\_arr

Initial values of the parameters:

Used predefined commands:

1/ theano.tensor.maximum(*a*, *b*) Returns a variable representing the maximum element by element of a and b [[source]](https://github.com/Theano/theano/blob/d395439aec5a6ddde8ef5c266fd976412a5c5695/theano/tensor.py)

2/ theano.tensor.abs\_(*a*) Returns a variable representing the absolute of a, ie |a|.

[[source]](https://github.com/Theano/theano/blob/d395439aec5a6ddde8ef5c266fd976412a5c5695/theano/tensor.py)

3/ theano.tensor.dot(*X*, *Y*) For 2-D arrays it is equivalent to matrix multiplication, and for 1-D arrays to inner product of vectors (without complex conjugation). For N dimensions it is a sum product over the last axis of a and the second-to-last of b: [[source]](https://github.com/Theano/theano/blob/d395439aec5a6ddde8ef5c266fd976412a5c5695/theano/tensor/basic.py#L6042-L6086)

4/The '**with**' statement clarifies code that previously would use try...finally blocks to ensure that clean-up code is executed. In this section, I'll discuss the statement as it will commonly be used. In the next section, I'll examine the implementation details and show how to write objects for use with this statement.

The 'with' statement is a new control-flow structure whose basic structure is:

with expression [as variable]:

with-block

The expression is evaluated, and it should result in an object that supports the context management protocol (that is, has \_\_enter\_\_() and \_\_exit\_\_() methods.

The object's \_\_enter\_\_() is called before with-block is executed and therefore can run set-up code. It also may return a value that is bound to the name variable, if given. (Note carefully that variable is *not* assigned the result of expression.)

After execution of the with-block is finished, the object's \_\_exit\_\_() method is called, even if the block raised an exception, and can therefore run clean-up code. (docs.python.org)

5/ The *theano*.*scan*() function is the public-facing interface for looping in Theano. (deeplearning.net)

fn: is a function that describes the operations involved in one step of scan.

6/ theano.tensor.zeros\_like(*x*, *dtype=None*)[[source]](https://github.com/Theano/theano/blob/d395439aec5a6ddde8ef5c266fd976412a5c5695/theano/tensor/basic.py#L2506-L2528)

|  |  |
| --- | --- |
| **Parameters:** | * **x** – tensor that has the same shape as output * **dtype** – data-type, optional By default, it will be x.dtype. |

Returns a tensor the shape of x filled with zeros of the type of dtype.

In[2]

Delays the input new\_I\_t by delay and return and array with length len\_new\_cases\_obs

inferred\_cases = interpolate(new\_I\_t, delay, delay\_mat)

= tt.dot(new\_I\_t, tt.maximum(1-tt.abs\_(delay\_matrix - delay), 0)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| New cases\date i | d1 | d2 | d3 | d4 | d5 |
| Reported obs |  |  |  |  |  |
| Reported inf. |  |  |  |  |  |