NumPy

In [2]:
```
import numpy as np
```

In [2]:
```
a = np.array([1,2,3,4,])
b = np.array([2,3,4,4])
```

In [3]:
```
print(a)
```

```
[1 2 3 4]
```

In [4]:
```
print (b)
```

```
[2 3 4 4]
```

In [5]:
```
print(type(a))
print(type(b))
```

```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

In [9]:
```
c = np.array((24,5,6,6),dtype = "f")
```

In [10]:
```
type(c)
```

Out[10]:
```
numpy.ndarray
```

In [11]:
```
a.dtype
```

Out[11]:
```
dtype('int32')
```

In [12]:
```
c.dtype
```

Out[12]:
```
dtype('float32')
```

In [3]:
```
a = np.array([[1,2,3] , [4,5,6]])
```

In [4]:
```
print(a[0])
```

```
[1 2 3]
```

In [5]:
```
print(a[1])
```

```
[4 5 6]
```

In [6]:
```
print(a)
```

```
[[1 2 3]
 [4 5 6]]
```

In [7]:
```
print(a[1,2])
```

```
6
```

In [10]:
```
a.ndim
```

Out[10]:
```
2
```

In [24]:
```
b = np.array([[1,2,3],[2,4,5],[5,6,8]])   ## All should have same size of array
```

```
In [25]: b.ndim

Out[25]: 2

In [27]: b[2,2]

Out[27]: 8

In [32]: c = np.array([[[1,2,3],[2,4,5],[5,6,8]],[[1,2,3],[2,4,5],[5,6,8]]])

In [33]: c.ndim

Out[33]: 3

In [34]: c[1,1,2]

Out[34]: 5

In [35]: type(c)

Out[35]: numpy.ndarray

In [36]: c.shape  ## Tells the amount of elements in each dimension of the array

Out[36]: (2, 3, 3)

In [37]: c.shape[0]

Out[37]: 2

In [38]: c.shape[1]

Out[38]: 3

In [39]: c.shape[2]

Out[39]: 3

In [40]: B = np.array([3])

In [41]: B.ndim

Out[41]: 1

In [42]: C = np.array(3)

In [43]: C.ndim

Out[43]: 0

In [44]: c.size

Out[44]: 18

In [45]: c.nbytes

Out[45]: 72
```

In [46]:
```
c.astype
```

Out[46]:
```
<function ndarray.astype>
```

In [48]:
```
np.arange(100)
```

Out[48]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [11]:
```
np.arange(20,100,3)
```

Out[11]:
```
array([20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53, 56, 59, 62, 65, 68,
       71, 74, 77, 80, 83, 86, 89, 92, 95, 98])
```

In [54]:
```
x = np.random.permutation(np.arange(10))
print(x)
```

```
[5 8 3 9 0 2 6 4 7 1]
```

In [12]:
```
np.random.randint??
```

In [14]:
```
v = np.random.randint(20,300)
```

In [15]:
```
type(v)
```

Out[15]:
```
int
```

In [16]:
```
A = np.random.rand(1000)
```

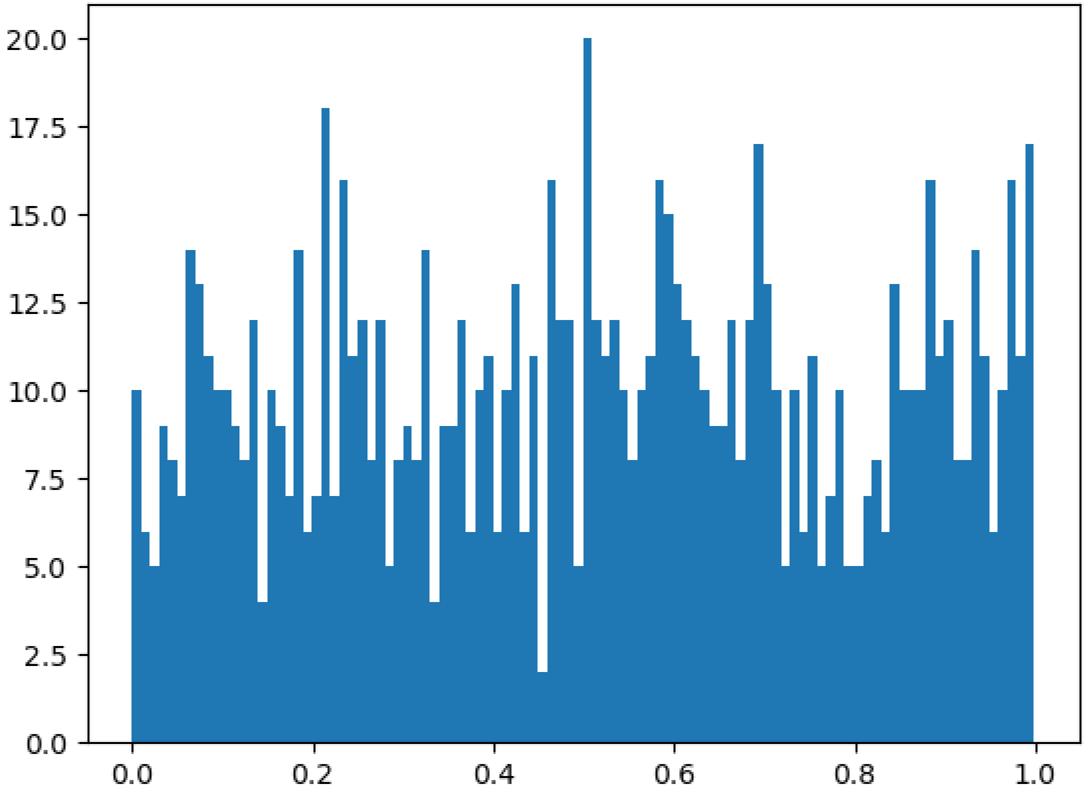
In [17]:
```
A
```

Out[17]:
```
array([0.77026434, 0.40507102, 0.82739175, 0.80165012, 0.79582249,
       0.44563875, 0.70079419, 0.95915981, 0.06336659, 0.44502076,
       0.08046927, 0.21248426, 0.05006767, 0.03861415, 0.83843742,
       0.16428498, 0.95149733, 0.67893428, 0.10729968, 0.59773908,
       0.39773992, 0.6762144 , 0.93053297, 0.57264117, 0.34827608,
       0.17016352, 0.28163911, 0.52619929, 0.32795093, 0.01719621,
       0.99807243, 0.72065194, 0.37330331, 0.22582597, 0.03221685,
       0.50306103, 0.99915883, 0.67082695, 0.34859884, 0.7419035 ,
       0.21506281, 0.38042846, 0.39730284, 0.66578939, 0.85951578,
       0.15609597, 0.80585962, 0.68891486, 0.40752759, 0.76922046,
       0.5459678 , 0.11661917, 0.46538612, 0.90002384, 0.45009817,
       0.3036959 , 0.53701378, 0.71072402, 0.68800907, 0.6768769 ,
       0.07093665, 0.59723077, 0.93128264, 0.77320752, 0.38670208,
       0.92548779, 0.05469635, 0.57731992, 0.56078631, 0.79682288,
       0.83943488, 0.71079411, 0.03272602, 0.1845831 , 0.33466574,
       0.13177198, 0.58917854, 0.71865744, 0.38664412, 0.39656753,
       0.9726834 , 0.45619181, 0.15009946, 0.00664431, 0.86880712,
       0.43971233, 0.59865833, 0.05184361, 0.06835135, 0.5476431 ,
       0.56505554, 0.10635394, 0.03041463, 0.15427435, 0.62984902,
       0.32532535, 0.88566533, 0.86274006, 0.80474479, 0.43310639,
       0.01113339, 0.52228207, 0.55626764, 0.79941111, 0.40858765,
       0.36309494, 0.84652078, 0.77130556, 0.35118661, 0.12630472,
       0.61131511, 0.78436036, 0.02187398, 0.8821381 , 0.14820929,
       0.62132901, 0.04160285, 0.45415074, 0.48040499, 0.02968544,
       0.76892217, 0.15960765, 0.54333057, 0.89546309, 0.49097332,
       0.89181999, 0.4002134 , 0.27950533, 0.57660697, 0.22399838,
       0.89113894, 0.06202619, 0.67738049, 0.2189735 , 0.96132543,
       0.66724779, 0.91956189, 0.56135538, 0.46448946, 0.20772168,
       0.16533419, 0.64071731, 0.32854969, 0.03313501, 0.66830293,
       0.92968405, 0.34565633, 0.56980084, 0.60015455, 0.52792745,
       0.03221387, 0.73872159, 0.98222792, 0.74834081, 0.40344283,
       0.22137389, 0.61918964, 0.86706534, 0.69453103, 0.50264366,
       0.46944902, 0.54062536, 0.57922464, 0.91692377, 0.8502546 ,
       0.57561869, 0.97964561, 0.74793079, 0.77662448, 0.42084402,
       0.91608794, 0.75798824, 0.81317426, 0.89295721, 0.73196381,
       0.39057679, 0.44385356, 0.36249796, 0.8163277 , 0.12973078,
       0.80292414, 0.86849003, 0.00305526, 0.03311551, 0.3274769 ,
       0.80250346, 0.31574251, 0.5834477 , 0.90368409, 0.82249467,
       0.17613196, 0.39943043, 0.15543877, 0.43734016, 0.24187376,
       0.0207098 , 0.47960053, 0.77502043, 0.67403023, 0.75175448,
       0.33521551, 0.39317558, 0.86648268, 0.57511779, 0.3211227 ,
       0.09388846, 0.53128582, 0.3196715 , 0.8948441 , 0.11543881,
       0.07502195, 0.23562092, 0.35526579, 0.42338107, 0.65987276,
       0.9458198 , 0.19813737, 0.23454272, 0.80976373, 0.65030447,
       0.58829512, 0.59614768, 0.12364225, 0.05108577, 0.95054349,
       0.83156593, 0.7508333 , 0.4908456 , 0.57272094, 0.18534069,
       0.70872537, 0.79684665, 0.80734308, 0.22086194, 0.98707729,
       0.54350514, 0.98698071, 0.1634944 , 0.93123307, 0.03516003,
       0.12480067, 0.88501056, 0.76124202, 0.71133314, 0.92186819,
       0.0984569 , 0.14002596, 0.52685751, 0.9348532 , 0.68483143,
       0.77777127, 0.37179014, 0.00101482, 0.33798839, 0.25488407,
       0.17940112, 0.4582296 , 0.55539555, 0.40155011, 0.13243722,
       0.71221076, 0.78066395, 0.0451028 , 0.59012042, 0.29009066,
       0.01550795, 0.5680643 , 0.21184647, 0.79465273, 0.84561147,
       0.83776413, 0.12406843, 0.27328467, 0.38280974, 0.41557354,
       0.37428249, 0.18040376, 0.17722441, 0.74369917, 0.95549687,
       0.62176684, 0.23377041, 0.18610681, 0.89133129, 0.81734758,
       0.39934752, 0.99078844, 0.91535474, 0.68868698, 0.98641573,
       0.61446797, 0.61915585, 0.0821788 , 0.18735551, 0.89593094,
       0.13320986, 0.18964452, 0.29168532, 0.58562655, 0.9007303 ,
       0.74748187, 0.75784881, 0.49809807, 0.59516157, 0.5309023 ,
       0.5444152 , 0.08044721, 0.3587078 , 0.11096725, 0.98586966,
       0.55370418, 0.25323834, 0.58796262, 0.11854738, 0.42307719,
       0.14371448, 0.39497885, 0.49191962, 0.96347628, 0.72937202,
       0.06403757, 0.00281294, 0.84401628, 0.11015288, 0.42184299,
       0.72858699, 0.03181857, 0.00549775, 0.93522377, 0.85360242,
       0.40934745, 0.75967722, 0.23237176, 0.30783779, 0.38083604,
       0.64504614, 0.78448414, 0.85793528, 0.74364784, 0.71694176,
       0.21874662, 0.5829091 , 0.00216026, 0.28234685, 0.32484946,
       0.87654152, 0.05778186, 0.23986183, 0.72784559, 0.56468132,
       0.58877686, 0.74461306, 0.52372973, 0.90390773, 0.64241911,
       0.38912183, 0.74236341, 0.81270334, 0.15291816, 0.2179114 ,
       0.36557544, 0.19128929, 0.09787099, 0.52103552, 0.74609938,
       0.74704883, 0.28833853, 0.22230386, 0.37901205, 0.62284781,
       0.74274434, 0.54673389, 0.9247873 , 0.07220789, 0.40379822,
       0.00321185, 0.7707281 , 0.75528171, 0.91552023, 0.82637023,
       0.72081189, 0.55080113, 0.43285147, 0.26957187, 0.1287156 ,
       0.76492495, 0.94866782, 0.40590627, 0.30724746, 0.18526157,
       0.11305754, 0.33233995, 0.35577268, 0.39860508, 0.3610915 ,
       0.0197637 , 0.53531827, 0.68734563, 0.40692234, 0.752333  ,
       0.55344811, 0.52349067, 0.48972942, 0.59234471, 0.64470558,
       0.19969348, 0.41637094, 0.10145397, 0.30038728, 0.86322153,
       0.61245651, 0.54221913, 0.13231569, 0.21725342, 0.49543469,
       0.51571603, 0.55567722, 0.79902712, 0.18801767, 0.77864357,
       0.71620067, 0.82508605, 0.37897017, 0.76916622, 0.4968434 ,
       0.21082632, 0.12843107, 0.55803396, 0.54255385, 0.45612616,
       0.8643673 , 0.55078441, 0.70101893, 0.75206069, 0.25656095,
       0.44158434, 0.31668665, 0.93793681, 0.61016887, 0.89518854,
       0.88179065, 0.79163753, 0.91437028, 0.78040349, 0.28076133,
       0.3125848 , 0.78246519, 0.19802284, 0.73692095, 0.13412536,
       0.19473968, 0.33514031, 0.81481623, 0.06657801, 0.18153464,
       0.96099967, 0.61072978, 0.64140646, 0.04787651, 0.51080476,
       0.21366256, 0.82620408, 0.35346929, 0.97815673, 0.98574454,
       0.01852125, 0.54702506, 0.17412098, 0.58993917, 0.16075766,
       0.46706649, 0.03945326, 0.00556098, 0.47046335, 0.35187013,
       0.76963591, 0.86598345, 0.25654795, 0.78166727, 0.20008687,
       0.95158774, 0.19504981, 0.12031078, 0.56562543, 0.63167019,
       0.13039601, 0.24834131, 0.73291108, 0.46025951, 0.52468383,
       0.89458431, 0.7251878 , 0.10212392, 0.45675276, 0.52814727,
       0.71022203, 0.95077294, 0.47691967, 0.48797424, 0.67143379,
       0.28838933, 0.17608335, 0.67602893, 0.11877644, 0.55982733,
       0.49185095, 0.24215387, 0.96712727, 0.92921557, 0.17344438,
       0.99893376, 0.83641023, 0.33145612, 0.06495258, 0.1313175 ,
       0.9334529 , 0.84238332, 0.90462646, 0.29556886, 0.89716631,
       0.55735077, 0.75730473, 0.85088651, 0.05210105, 0.5987189 ,
       0.08575429, 0.30737407, 0.77808789, 0.04870349, 0.62213512,
       0.85497262, 0.09149005, 0.88290139, 0.48642862, 0.37341661,
       0.26461634, 0.83892533, 0.90150673, 0.74177554, 0.16544769,
       0.30924436, 0.26000785, 0.30718214, 0.33930344, 0.3595137 ,
       0.69050947, 0.53737603, 0.31402295, 0.92611605, 0.58126293,
       0.7128423 , 0.46069748, 0.83159169, 0.90133098, 0.14156844,
       0.99412722, 0.4003222 , 0.83400307, 0.89296823, 0.56693528,
       0.93791179, 0.7295903 , 0.21680376, 0.03023662, 0.24987043,
       0.45744411, 0.98539981, 0.60552957, 0.95997753, 0.29864328,
       0.35487218, 0.2785915 , 0.99425794, 0.57027735, 0.14846725,
       0.68645951, 0.24996491, 0.70920362, 0.01802463, 0.90618863,
       0.78822182, 0.03439745, 0.17098786, 0.90062284, 0.55759463,
       0.65090474, 0.76078372, 0.6156222 , 0.03949754, 0.53142454,
       0.6660122 , 0.72110339, 0.82038361, 0.43979756, 0.76723517,
       0.51466468, 0.95240351, 0.32591193, 0.35581948, 0.28982349,
       0.28155262, 0.3791768 , 0.99991711, 0.57294071, 0.72834231,
       0.26727547, 0.55026965, 0.41561861, 0.130108  , 0.14947367,
       0.94318616, 0.84141256, 0.61125782, 0.20076601, 0.37723203,
       0.71550222, 0.74631845, 0.25641215, 0.58707558, 0.8915191 ,
       0.52402826, 0.59082763, 0.38369041, 0.77195483, 0.08379363,
       0.99143988, 0.48410359, 0.23453366, 0.15984446, 0.6611381 ,
       0.23009663, 0.85679251, 0.53701024, 0.38346419, 0.48398161,
       0.02139921, 0.33987909, 0.57306707, 0.55903524, 0.22541239,
       0.5433488 , 0.65665161, 0.64513601, 0.01086025, 0.6363805 ,
       0.26106137, 0.14283195, 0.2212328 , 0.43911422, 0.14326517,
       0.16940877, 0.3051302 , 0.95694932, 0.28974524, 0.06775832,
       0.52426102, 0.90066524, 0.78327545, 0.65552164, 0.89475755,
       0.38757643, 0.99942716, 0.96929306, 0.81315311, 0.42127219,
       0.95447909, 0.9485322 , 0.25191399, 0.41298029, 0.08356748,
       0.35699957, 0.76147383, 0.24654228, 0.44702527, 0.39543099,
       0.76375907, 0.10350306, 0.57479016, 0.26475278, 0.45239705,
       0.57399287, 0.43883622, 0.95797268, 0.56189312, 0.63406183,
       0.69896909, 0.54180878, 0.69539069, 0.51217082, 0.0791552 ,
       0.40329509, 0.58711983, 0.34352711, 0.76357385, 0.85889392,
       0.20858037, 0.31247134, 0.02821499, 0.30620094, 0.26956411,
       0.45249765, 0.46820338, 0.41123821, 0.2385119 , 0.28585965,
       0.7044065 , 0.04301278, 0.84487672, 0.01554331, 0.7558361 ,
       0.31750202, 0.10272648, 0.47439107, 0.85133984, 0.72660572,
       0.77437171, 0.6579632 , 0.8285537 , 0.75146201, 0.12710523,
       0.10974601, 0.15685506, 0.10435412, 0.99799332, 0.08797291,
       0.023285  , 0.16900223, 0.68035166, 0.39657139, 0.09699613,
       0.54358796, 0.63927201, 0.03852865, 0.26244892, 0.48556747,
       0.60761164, 0.2923162 , 0.84314885, 0.47807776, 0.42245979,
       0.3692393 , 0.36259325, 0.938093  , 0.52352374, 0.06041312,
       0.28618762, 0.59324481, 0.11122876, 0.17665263, 0.85603611,
       0.66648752, 0.83041434, 0.93691942, 0.57302868, 0.10031208,
       0.7231644 , 0.72306096, 0.22564804, 0.26747159, 0.35128705,
       0.26904183, 0.93798709, 0.98467751, 0.08053577, 0.42785663,
       0.24477443, 0.34463432, 0.41054992, 0.11005976, 0.55634607,
       0.41797423, 0.90942059, 0.01630103, 0.10980735, 0.50713154,
       0.34158415, 0.49517174, 0.05949008, 0.22019957, 0.22683409,
       0.36144508, 0.93052363, 0.02060985, 0.18730465, 0.0895462 ,
       0.29408234, 0.9971096 , 0.00807379, 0.90225716, 0.75815662,
       0.393601  , 0.4368155 , 0.4848258 , 0.60886119, 0.16369496,
       0.50411674, 0.60411394, 0.16080552, 0.69989061, 0.9750209 ,
       0.71714874, 0.02547146, 0.3197432 , 0.1804226 , 0.72200816,
       0.95923484, 0.33592716, 0.84075338, 0.96677097, 0.32890546,
       0.69846445, 0.35793347, 0.28035536, 0.21609835, 0.75824999,
       0.69317712, 0.99541505, 0.78613672, 0.84186527, 0.23098779,
       0.70934822, 0.10990493, 0.51272095, 0.81523947, 0.20305838,
       0.89867288, 0.87895235, 0.23278529, 0.46560109, 0.79283648,
       0.05907371, 0.2005117 , 0.88205476, 0.98576741, 0.48421941,
       0.75853757, 0.16132439, 0.6208909 , 0.96713647, 0.62261134,
       0.1267604 , 0.60160098, 0.51943485, 0.1320871 , 0.68762388,
       0.81593362, 0.56345199, 0.53957389, 0.45564863, 0.75720704,
       0.28454826, 0.83624133, 0.6619841 , 0.01472435, 0.73784258,
       0.26508358, 0.10866101, 0.32932054, 0.10481116, 0.31484632,
       0.56667593, 0.33118006, 0.07076094, 0.74649871, 0.12149285,
       0.84705834, 0.84433588, 0.5633023 , 0.85838867, 0.94614182,
       0.76238   , 0.61936506, 0.17762283, 0.34838319, 0.46469988,
       0.41927307, 0.65291866, 0.36032995, 0.8486697 , 0.79185662,
       0.37574619, 0.95772318, 0.9395549 , 0.13243252, 0.46226081,
       0.02822198, 0.13886106, 0.67696779, 0.42802597, 0.52429046,
       0.05963306, 0.81099852, 0.15558563, 0.40560201, 0.35178079,
       0.23042666, 0.60499805, 0.81950658, 0.23532297, 0.17013271,
       0.5900594 , 0.36423064, 0.59367366, 0.19799652, 0.5282548 ,
       0.41965851, 0.74052732, 0.34619475, 0.49404885, 0.5884885 ,
       0.11429837, 0.96207555, 0.99669683, 0.23875705, 0.23857609,
       0.12361807, 0.93070321, 0.78029815, 0.90123255, 0.43447557,
       0.98303107, 0.82462018, 0.56393797, 0.91809094, 0.16704594,
       0.93953059, 0.90255658, 0.82120986, 0.71364428, 0.31117955,
       0.14048543, 0.83300399, 0.01534628, 0.7127823 , 0.69707532,
       0.27385047, 0.22415178, 0.33882728, 0.3233493 , 0.10991612,
       0.67643549, 0.30454515, 0.16643281, 0.95915194, 0.05198944,
       0.74860861, 0.16091227, 0.20059127, 0.10486334, 0.70465189,
       0.35538512, 0.15083344, 0.72952782, 0.58549388, 0.94187229,
       0.95161371, 0.24352972, 0.21916936, 0.31477357, 0.6620499 ,
       0.21107415, 0.04326296, 0.32184274, 0.52289941, 0.42383991,
       0.684698  , 0.97141687, 0.50670424, 0.08927118, 0.29177106,
       0.96898295, 0.19214148, 0.90692193, 0.54553041, 0.94604721,
       0.85426947, 0.35093023, 0.0508372 , 0.52053189, 0.22179727,
       0.53193353, 0.33010754, 0.7647021 , 0.95228693, 0.83616635,
       0.04026484, 0.54794611, 0.05015064, 0.55025631, 0.13582363,
       0.91697132, 0.55452534, 0.39577479, 0.33044751, 0.07573318,
       0.83024129, 0.38803213, 0.59408462, 0.33378929, 0.43856983])
```

In [18]:
```
import matplotlib.pyplot as plt
```

In [64]:
```
plt.hist(A,bins = 100)
```

Out[64]:
```
(array([10.,  6.,  5.,  9.,  8.,  7., 14., 13., 11., 10., 10.,  9.,  8.,
        12.,  4., 10.,  9.,  7., 14.,  6.,  7., 18.,  7., 16., 11., 12.,
         8., 12.,  5.,  8.,  9.,  8., 14.,  4.,  9.,  9., 12.,  6., 10.,
        11.,  6., 10., 13.,  6., 11.,  2., 16., 12., 12.,  5., 20., 12.,
        11., 12., 10.,  8., 10., 11., 16., 15., 13., 12., 11., 10.,  9.,
         9., 12.,  8., 12., 17., 13., 10.,  5., 10.,  6., 11.,  5.,  7.,
        10.,  5.,  5.,  7.,  8.,  6., 13., 10., 10., 10., 16., 11., 12.,
         8.,  8., 14., 11.,  6., 10., 16., 11., 17.]),
 array([5.02990655e-05, 1.00328617e-02, 2.00154244e-02, 2.99979870e-02,
        3.99805497e-02, 4.99631123e-02, 5.99456750e-02, 6.99282376e-02,
        7.99108003e-02, 8.98933629e-02, 9.98759256e-02, 1.09858488e-01,
        1.19841051e-01, 1.29823614e-01, 1.39806176e-01, 1.49788739e-01,
        1.59771301e-01, 1.69753864e-01, 1.79736427e-01, 1.89718989e-01,
        1.99701552e-01, 2.09684115e-01, 2.19666677e-01, 2.29649240e-01,
        2.39631803e-01, 2.49614365e-01, 2.59596928e-01, 2.69579491e-01,
        2.79562053e-01, 2.89544616e-01, 2.99527179e-01, 3.09509741e-01,
        3.19492304e-01, 3.29474867e-01, 3.39457429e-01, 3.49439992e-01,
        3.59422554e-01, 3.69405117e-01, 3.79387680e-01, 3.89370242e-01,
        3.99352805e-01, 4.09335368e-01, 4.19317930e-01, 4.29300493e-01,
        4.39283056e-01, 4.49265618e-01, 4.59248181e-01, 4.69230744e-01,
        4.79213306e-01, 4.89195869e-01, 4.99178432e-01, 5.09160994e-01,
        5.19143557e-01, 5.29126120e-01, 5.39108682e-01, 5.49091245e-01,
        5.59073808e-01, 5.69056370e-01, 5.79038933e-01, 5.89021495e-01,
        5.99004058e-01, 6.08986621e-01, 6.18969183e-01, 6.28951746e-01,
        6.38934309e-01, 6.48916871e-01, 6.58899434e-01, 6.68881997e-01,
        6.78864559e-01, 6.88847122e-01, 6.98829685e-01, 7.08812247e-01,
        7.18794810e-01, 7.28777373e-01, 7.38759935e-01, 7.48742498e-01,
        7.58725061e-01, 7.68707623e-01, 7.78690186e-01, 7.88672748e-01,
        7.98655311e-01, 8.08637874e-01, 8.18620436e-01, 8.28602999e-01,
        8.38585562e-01, 8.48568124e-01, 8.58550687e-01, 8.68533250e-01,
        8.78515812e-01, 8.88498375e-01, 8.98480938e-01, 9.08463500e-01,
        9.18446063e-01, 9.28428626e-01, 9.38411188e-01, 9.48393751e-01,
        9.58376314e-01, 9.68358876e-01, 9.78341439e-01, 9.88324001e-01,
        9.98306564e-01]),
 <BarContainer object of 100 artists>)
```

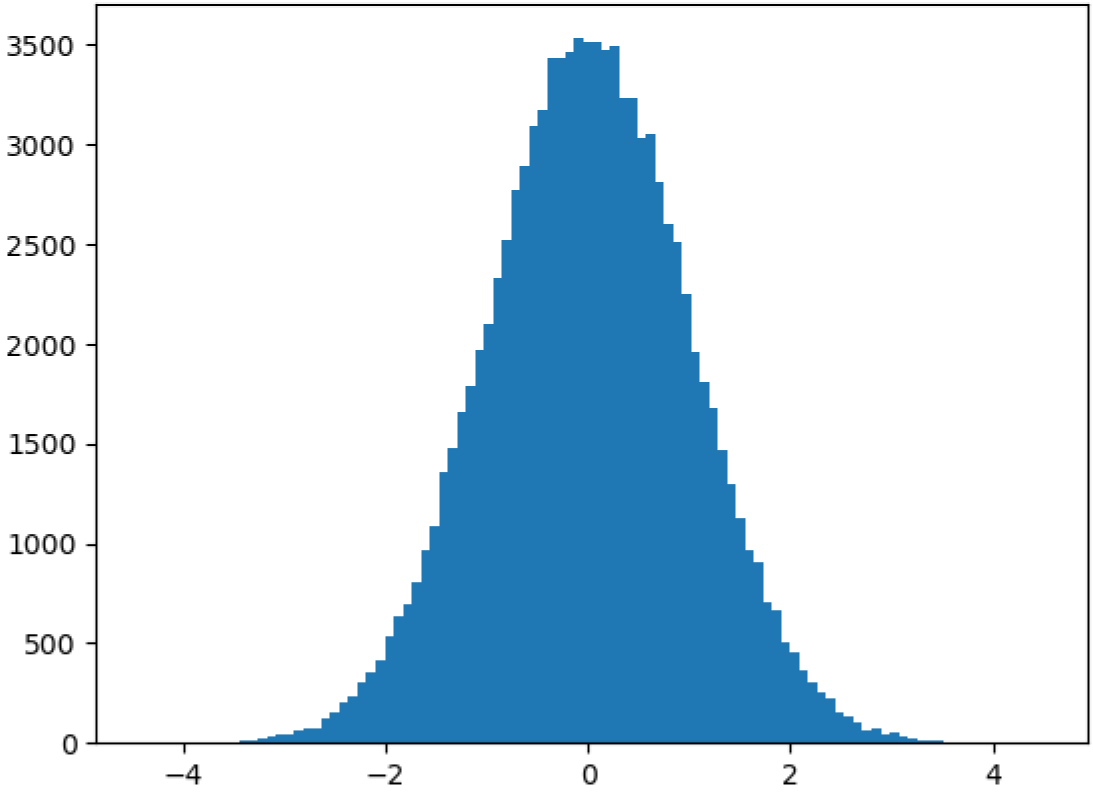


In [65]:

```
B = np.random.randn(100000)
plt.hist(B,bins = 100)
```

Out[65]:

```
(array([1.000e+00, 0.000e+00, 0.000e+00, 2.000e+00, 1.000e+00, 2.000e+00,
        2.000e+00, 4.000e+00, 6.000e+00, 6.000e+00, 3.000e+00, 8.000e+00,
        1.400e+01, 2.300e+01, 3.500e+01, 4.600e+01, 4.300e+01, 6.000e+01,
        7.100e+01, 7.100e+01, 1.240e+02, 1.560e+02, 2.010e+02, 2.310e+02,
        3.050e+02, 3.540e+02, 4.140e+02, 5.370e+02, 6.370e+02, 6.980e+02,
        8.050e+02, 9.690e+02, 1.086e+03, 1.358e+03, 1.480e+03, 1.657e+03,
        1.791e+03, 1.963e+03, 2.102e+03, 2.333e+03, 2.522e+03, 2.768e+03,
        2.887e+03, 3.087e+03, 3.174e+03, 3.433e+03, 3.435e+03, 3.459e+03,
        3.529e+03, 3.516e+03, 3.507e+03, 3.469e+03, 3.488e+03, 3.230e+03,
        3.232e+03, 3.032e+03, 3.048e+03, 2.810e+03, 2.603e+03, 2.505e+03,
        2.249e+03, 1.957e+03, 1.808e+03, 1.675e+03, 1.465e+03, 1.296e+03,
        1.126e+03, 9.650e+02, 9.090e+02, 7.080e+02, 6.650e+02, 5.030e+02,
        4.580e+02, 3.670e+02, 3.050e+02, 2.550e+02, 2.210e+02, 1.580e+02,
        1.320e+02, 1.020e+02, 6.600e+01, 7.100e+01, 4.200e+01, 5.300e+01,
        3.100e+01, 2.600e+01, 1.400e+01, 1.000e+01, 1.100e+01, 3.000e+00,
        4.000e+00, 4.000e+00, 2.000e+00, 1.000e+00, 2.000e+00, 1.000e+00,
        1.000e+00, 0.000e+00, 0.000e+00, 1.000e+00]),
 array([-4.42129726, -4.33211611, -4.24293497, -4.15375382, -4.06457268,
        -3.97539153, -3.88621039, -3.79702924, -3.7078481 , -3.61866695,
        -3.52948581, -3.44030466, -3.35112352, -3.26194237, -3.17276123,
        -3.08358008, -2.99439894, -2.90521779, -2.81603665, -2.7268555 ,
        -2.63767436, -2.54849321, -2.45931207, -2.37013092, -2.28094978,
        -2.19176863, -2.10258749, -2.01340634, -1.9242252 , -1.83504405,
        -1.74586291, -1.65668176, -1.56750061, -1.47831947, -1.38913832,
        -1.29995718, -1.21077603, -1.12159489, -1.03241374, -0.9432326 ,
        -0.85405145, -0.76487031, -0.67568916, -0.58650802, -0.49732687,
        -0.40814573, -0.31896458, -0.22978344, -0.14060229, -0.05142115,
         0.03776   ,  0.12694114,  0.21612229,  0.30530343,  0.39448458,
         0.48366572,  0.57284687,  0.66202801,  0.75120916,  0.8403903 ,
         0.92957145,  1.01875259,  1.10793374,  1.19711488,  1.28629603,
         1.37547717,  1.46465832,  1.55383946,  1.64302061,  1.73220175,
         1.8213829 ,  1.91056404,  1.99974519,  2.08892633,  2.17810748,
         2.26728862,  2.35646977,  2.44565091,  2.53483206,  2.6240132 ,
         2.71319435,  2.8023755 ,  2.89155664,  2.98073779,  3.06991893,
         3.15910008,  3.24828122,  3.33746237,  3.42664351,  3.51582466,
         3.6050058 ,  3.69418695,  3.78336809,  3.87254924,  3.96173038,
         4.05091153,  4.14009267,  4.22927382,  4.31845496,  4.40763611,
         4.49681725]),
 <BarContainer object of 100 artists>)
```



```
In [20]: C = np.random.rand(2,3)
```

```
In [21]: C
```

```
Out[21]: array([[0.23763739, 0.12293626, 0.41030879],
                [0.20186687, 0.851891  , 0.31724362]])
```

```
In [22]: C.ndim
```

```
Out[22]: 2
```

```
In [23]: C =  np.random.rand(2,3,4,2)
```

```
In [24]: C.ndim
```

```
Out[24]: 4
```

```
In [25]: C
```

```
Out[25]: array([[[[0.39715539, 0.82869715],
                   [0.48726109, 0.80176536],
                   [0.23552154, 0.29771621],
                   [0.6284648 , 0.76802856]],

                  [[0.81670575, 0.50630161],
                   [0.92127226, 0.78794814],
                   [0.73711478, 0.3160002 ],
                   [0.35947594, 0.56355416]],

                  [[0.59598059, 0.59118275],
                   [0.68083136, 0.11148099],
                   [0.41831554, 0.38834297],
                   [0.42500367, 0.89655028]]],


                 [[[0.80154771, 0.11066008],
                   [0.91363518, 0.27241838],
                   [0.2978877 , 0.8370984 ],
                   [0.37085133, 0.92248519]],

                  [[0.44040239, 0.94651039],
                   [0.21118516, 0.32041444],
                   [0.21899407, 0.25082022],
                   [0.07007234, 0.80929466]],

                  [[0.76733774, 0.35497098],
                   [0.15149692, 0.75081669],
                   [0.06541053, 0.85892429],
                   [0.8007893 , 0.01532184]]]])
```

```
In [26]: D = np.arange(100).reshape(4,25)
```

```
In [27]: D.shape
```

```
Out[27]: (4, 25)
```

```
In [28]: D = np.arange(100).reshape(4,5,5)
```

```
In [29]: D.shape
```

```
Out[29]: (4, 5, 5)
```

```
In [30]: np.zeros
```

```
Out[30]: <function numpy.zeros>
```

```
In [31]: np.ones
```

```
Out[31]: <function numpy.ones(shape, dtype=None, order='C', *, like=None)>
```

Slicing Slicing in numpy does not create a copy but acquires the same memory whereas in normal slicing in list it creates a copy

```
In [84]: A = np.arange(100)
```

```
In [85]: b = A[3:9]
```

```
In [86]: b[0] = -1299
In [87]: b
```

```
Out[87]: array([-1299,     4,     5,     6,     7,     8])
```

```
In [34]: A
```

```
Out[34]: array([0.77026434, 0.40507102, 0.82739175, 0.80165012, 0.79582249,
                0.44563875, 0.70079419, 0.95915981, 0.06336659, 0.44502076,
                0.08046927, 0.21248426, 0.05006767, 0.03861415, 0.83843742,
                0.16428498, 0.95149733, 0.67893428, 0.10729968, 0.59773908,
                0.39773992, 0.6762144 , 0.93053297, 0.57264117, 0.34827608,
                0.17016352, 0.28163911, 0.52619929, 0.32795093, 0.01719621,
                0.99807243, 0.72065194, 0.37330331, 0.22582597, 0.03221685,
                0.50306103, 0.99915883, 0.67082695, 0.34859884, 0.7419035 ,
                0.21506281, 0.38042846, 0.39730284, 0.66578939, 0.85951578,
                0.15609597, 0.80585962, 0.68891486, 0.40752759, 0.76922046,
                0.5459678 , 0.11661917, 0.46538612, 0.90002384, 0.45009817,
                0.3036959 , 0.53701378, 0.71072402, 0.68800907, 0.6768769 ,
                0.07093665, 0.59723077, 0.93128264, 0.77320752, 0.38670208,
                0.92548779, 0.05469635, 0.57731992, 0.56078631, 0.79682288,
                0.83943488, 0.71079411, 0.03272602, 0.1845831 , 0.33466574,
                0.13177198, 0.58917854, 0.71865744, 0.38664412, 0.39656753,
                0.9726834 , 0.45619181, 0.15009946, 0.00664431, 0.86880712,
                0.43971233, 0.59865833, 0.05184361, 0.06835135, 0.5476431 ,
                0.56505554, 0.10635394, 0.03041463, 0.15427435, 0.62984902,
                0.32532535, 0.88566533, 0.86274006, 0.80474479, 0.43310639,
                0.01113339, 0.52228207, 0.55626764, 0.79941111, 0.40858765,
                0.36309494, 0.84652078, 0.77130556, 0.35118661, 0.12630472,
                0.61131511, 0.78436036, 0.02187398, 0.8821381 , 0.14820929,
                0.62132901, 0.04160285, 0.45415074, 0.48040499, 0.02968544,
                0.76892217, 0.15960765, 0.54333057, 0.89546309, 0.49097332,
                0.89181999, 0.4002134 , 0.27950533, 0.57660697, 0.22399838,
                0.89113894, 0.06202619, 0.67738049, 0.2189735 , 0.96132543,
                0.66724779, 0.91956189, 0.56135538, 0.46448946, 0.20772168,
                0.16533419, 0.64071731, 0.32854969, 0.03313501, 0.66830293,
                0.92968405, 0.34565633, 0.56980084, 0.60015455, 0.52792745,
                0.03221387, 0.73872159, 0.98222792, 0.74834081, 0.40344283,
                0.22137389, 0.61918964, 0.86706534, 0.69453103, 0.50264366,
                0.46944902, 0.54062536, 0.57922464, 0.91692377, 0.8502546 ,
                0.57561869, 0.97964561, 0.74793079, 0.77662448, 0.42084402,
                0.91608794, 0.75798824, 0.81317426, 0.89295721, 0.73196381,
                0.39057679, 0.44385356, 0.36249796, 0.8163277 , 0.12973078,
                0.80292414, 0.86849003, 0.00305526, 0.03311551, 0.3274769 ,
                0.80250346, 0.31574251, 0.5834477 , 0.90368409, 0.82249467,
                0.17613196, 0.39943043, 0.15543877, 0.43734016, 0.24187376,
                0.0207098 , 0.47960053, 0.77502043, 0.67403023, 0.75175448,
                0.33521551, 0.39317558, 0.86648268, 0.57511779, 0.3211227 ,
                0.09388846, 0.53128582, 0.3196715 , 0.8948441 , 0.11543881,
                0.07502195, 0.23562092, 0.35526579, 0.42338107, 0.65987276,
                0.9458198 , 0.19813737, 0.23454272, 0.80976373, 0.65030447,
                0.58829512, 0.59614768, 0.12364225, 0.05108577, 0.95054349,
                0.83156593, 0.7508333 , 0.4908456 , 0.57272094, 0.18534069,
                0.70872537, 0.79684665, 0.80734308, 0.22086194, 0.98707729,
                0.54350514, 0.98698071, 0.1634944 , 0.93123307, 0.03516003,
                0.12480067, 0.88501056, 0.76124202, 0.71133314, 0.92186819,
                0.0984569 , 0.14002596, 0.52685751, 0.9348532 , 0.68483143,
                0.77777127, 0.37179014, 0.00101482, 0.33798839, 0.25488407,
                0.17940112, 0.4582296 , 0.55539555, 0.40155011, 0.13243722,
                0.71221076, 0.78066395, 0.0451028 , 0.59012042, 0.29009066,
                0.01550795, 0.5680643 , 0.21184647, 0.79465273, 0.84561147,
                0.83776413, 0.12406843, 0.27328467, 0.38280974, 0.41557354,
                0.37428249, 0.18040376, 0.17722441, 0.74369917, 0.95549687,
                0.62176684, 0.23377041, 0.18610681, 0.89133129, 0.81734758,
                0.39934752, 0.99078844, 0.91535474, 0.68868698, 0.98641573,
                0.61446797, 0.61915585, 0.0821788 , 0.18735551, 0.89593094,
                0.13320986, 0.18964452, 0.29168532, 0.58562655, 0.9007303 ,
                0.74748187, 0.75784881, 0.49809807, 0.59516157, 0.5309023 ,
                0.5444152 , 0.08044721, 0.3587078 , 0.11096725, 0.98586966,
                0.55370418, 0.25323834, 0.58796262, 0.11854738, 0.42307719,
                0.14371448, 0.39497885, 0.49191962, 0.96347628, 0.72937202,
                0.06403757, 0.00281294, 0.84401628, 0.11015288, 0.42184299,
                0.72858699, 0.03181857, 0.00549775, 0.93522377, 0.85360242,
                0.40934745, 0.75967722, 0.23237176, 0.30783779, 0.38083604,
                0.64504614, 0.78448414, 0.85793528, 0.74364784, 0.71694176,
                0.21874662, 0.5829091 , 0.00216026, 0.28234685, 0.32484946,
                0.87654152, 0.05778186, 0.23986183, 0.72784559, 0.56468132,
                0.58877686, 0.74461306, 0.52372973, 0.90390773, 0.64241911,
                0.38912183, 0.74236341, 0.81270334, 0.15291816, 0.2179114 ,
                0.36557544, 0.19128929, 0.09787099, 0.52103552, 0.74609938,
                0.74704883, 0.28833853, 0.22230386, 0.37901205, 0.62284781,
                0.74274434, 0.54673389, 0.9247873 , 0.07220789, 0.40379822,
                0.00321185, 0.7707281 , 0.75528171, 0.91552023, 0.82637023,
                0.72081189, 0.55080113, 0.43285147, 0.26957187, 0.1287156 ,
                0.76492495, 0.94866782, 0.40590627, 0.30724746, 0.18526157,
                0.11305754, 0.33233995, 0.35577268, 0.39860508, 0.3610915 ,
                0.0197637 , 0.53531827, 0.68734563, 0.40692234, 0.752333  ,
                0.55344811, 0.52349067, 0.48972942, 0.59234471, 0.64470558,
                0.19969348, 0.41637094, 0.10145397, 0.30038728, 0.86322153,
                0.61245651, 0.54221913, 0.13231569, 0.21725342, 0.49543469,
                0.51571603, 0.55567722, 0.79902712, 0.18801767, 0.77864357,
                0.71620067, 0.82508605, 0.37897017, 0.76916622, 0.4968434 ,
                0.21082632, 0.12843107, 0.55803396, 0.54255385, 0.45612616,
                0.8643673 , 0.55078441, 0.70101893, 0.75206069, 0.25656095,
                0.44158434, 0.31668665, 0.93793681, 0.61016887, 0.89518854,
                0.88179065, 0.79163753, 0.91437028, 0.78040349, 0.28076133,
                0.3125848 , 0.78246519, 0.19802284, 0.73692095, 0.13412536,
                0.19473968, 0.33514031, 0.81481623, 0.06657801, 0.18153464,
                0.96099967, 0.61072978, 0.64140646, 0.04787651, 0.51080476,
                0.21366256, 0.82620408, 0.35346929, 0.97815673, 0.98574454,
                0.01852125, 0.54702506, 0.17412098, 0.58993917, 0.16075766,
                0.46706649, 0.03945326, 0.00556098, 0.47046335, 0.35187013,
                0.76963591, 0.86598345, 0.25654795, 0.78166727, 0.20008687,
                0.95158774, 0.19504981, 0.12031078, 0.56562543, 0.63167019,
                0.13039601, 0.24834131, 0.73291108, 0.46025951, 0.52468383,
                0.89458431, 0.7251878 , 0.10212392, 0.45675276, 0.52814727,
                0.71022203, 0.95077294, 0.47691967, 0.48797424, 0.67143379,
                0.28838933, 0.17608335, 0.67602893, 0.11877644, 0.55982733,
                0.49185095, 0.24215387, 0.96712727, 0.92921557, 0.17344438,
                0.99893376, 0.83641023, 0.33145612, 0.06495258, 0.1313175 ,
                0.9334529 , 0.84238332, 0.90462646, 0.29556886, 0.89716631,
                0.55735077, 0.75730473, 0.85088651, 0.05210105, 0.5987189 ,
                0.08575429, 0.30737407, 0.77808789, 0.04870349, 0.62213512,
                0.85497262, 0.09149005, 0.88290139, 0.48642862, 0.37341661,
                0.26461634, 0.83892533, 0.90150673, 0.74177554, 0.16544769,
                0.30924436, 0.26000785, 0.30718214, 0.33930344, 0.3595137 ,
                0.69050947, 0.53737603, 0.31402295, 0.92611605, 0.58126293,
                0.7128423 , 0.46069748, 0.83159169, 0.90133098, 0.14156844,
                0.99412722, 0.4003222 , 0.83400307, 0.89296823, 0.56693528,
                0.93791179, 0.7295903 , 0.21680376, 0.03023662, 0.24987043,
                0.45744411, 0.98539981, 0.60552957, 0.95997753, 0.29864328,
                0.35487218, 0.2785915 , 0.99425794, 0.57027735, 0.14846725,
                0.68645951, 0.24996491, 0.70920362, 0.01802463, 0.90618863,
                0.78822182, 0.03439745, 0.17098786, 0.90062284, 0.55759463,
                0.65090474, 0.76078372, 0.6156222 , 0.03949754, 0.53142454,
                0.6660122 , 0.72110339, 0.82038361, 0.43979756, 0.76723517,
                0.51466468, 0.95240351, 0.32591193, 0.35581948, 0.28982349,
                0.28155262, 0.3791768 , 0.99991711, 0.57294071, 0.72834231,
                0.26727547, 0.55026965, 0.41561861, 0.130108  , 0.14947367,
                0.94318616, 0.84141256, 0.61125782, 0.20076601, 0.37723203,
                0.71550222, 0.74631845, 0.25641215, 0.58707558, 0.8915191 ,
                0.52402826, 0.59082763, 0.38369041, 0.77195483, 0.08379363,
                0.99143988, 0.48410359, 0.23453366, 0.15984446, 0.6611381 ,
                0.23009663, 0.85679251, 0.53701024, 0.38346419, 0.48398161,
                0.02139921, 0.33987909, 0.57306707, 0.55903524, 0.22541239,
                0.5433488 , 0.65665161, 0.64513601, 0.01086025, 0.6363805 ,
                0.26106137, 0.14283195, 0.2212328 , 0.43911422, 0.14326517,
                0.16940877, 0.3051302 , 0.95694932, 0.28974524, 0.06775832,
                0.52426102, 0.90066524, 0.78327545, 0.65552164, 0.89475755,
                0.38757643, 0.99942716, 0.96929306, 0.81315311, 0.42127219,
                0.95447909, 0.9485322 , 0.25191399, 0.41298029, 0.08356748,
                0.35699957, 0.76147383, 0.24654228, 0.44702527, 0.39543099,
                0.76375907, 0.10350306, 0.57479016, 0.26475278, 0.45239705,
                0.57399287, 0.43883622, 0.95797268, 0.56189312, 0.63406183,
                0.69896909, 0.54180878, 0.69539069, 0.51217082, 0.0791552 ,
                0.40329509, 0.58711983, 0.34352711, 0.76357385, 0.85889392,
                0.20858037, 0.31247134, 0.02821499, 0.30620094, 0.26956411,
                0.45249765, 0.46820338, 0.41123821, 0.2385119 , 0.28585965,
                0.7044065 , 0.04301278, 0.84487672, 0.01554331, 0.7558361 ,
                0.31750202, 0.10272648, 0.47439107, 0.85133984, 0.72660572,
                0.77437171, 0.6579632 , 0.8285537 , 0.75146201, 0.12710523,
                0.10974601, 0.15685506, 0.10435412, 0.99799332, 0.08797291,
                0.023285  , 0.16900223, 0.68035166, 0.39657139, 0.09699613,
                0.54358796, 0.63927201, 0.03852865, 0.26244892, 0.48556747,
                0.60761164, 0.2923162 , 0.84314885, 0.47807776, 0.42245979,
                0.3692393 , 0.36259325, 0.938093  , 0.52352374, 0.06041312,
                0.28618762, 0.59324481, 0.11122876, 0.17665263, 0.85603611,
                0.66648752, 0.83041434, 0.93691942, 0.57302868, 0.10031208,
                0.7231644 , 0.72306096, 0.22564804, 0.26747159, 0.35128705,
                0.26904183, 0.93798709, 0.98467751, 0.08053577, 0.42785663,
                0.24477443, 0.34463432, 0.41054992, 0.11005976, 0.55634607,
                0.41797423, 0.90942059, 0.01630103, 0.10980735, 0.50713154,
                0.34158415, 0.49517174, 0.05949008, 0.22019957, 0.22683409,
                0.36144508, 0.93052363, 0.02060985, 0.18730465, 0.0895462 ,
                0.29408234, 0.9971096 , 0.00807379, 0.90225716, 0.75815662,
                0.393601  , 0.4368155 , 0.4848258 , 0.60886119, 0.16369496,
                0.50411674, 0.60411394, 0.16080552, 0.69989061, 0.9750209 ,
                0.71714874, 0.02547146, 0.3197432 , 0.1804226 , 0.72200816,
                0.95923484, 0.33592716, 0.84075338, 0.96677097, 0.32890546,
                0.69846445, 0.35793347, 0.28035536, 0.21609835, 0.75824999,
                0.69317712, 0.99541505, 0.78613672, 0.84186527, 0.23098779,
                0.70934822, 0.10990493, 0.51272095, 0.81523947, 0.20305838,
                0.89867288, 0.87895235, 0.23278529, 0.46560109, 0.79283648,
                0.05907371, 0.2005117 , 0.88205476, 0.98576741, 0.48421941,
                0.75853757, 0.16132439, 0.6208909 , 0.96713647, 0.62261134,
                0.1267604 , 0.60160098, 0.51943485, 0.1320871 , 0.68762388,
                0.81593362, 0.56345199, 0.53957389, 0.45564863, 0.75720704,
                0.28454826, 0.83624133, 0.6619841 , 0.01472435, 0.73784258,
                0.26508358, 0.10866101, 0.32932054, 0.10481116, 0.31484632,
                0.56667593, 0.33118006, 0.07076094, 0.74649871, 0.12149285,
                0.84705834, 0.84433588, 0.5633023 , 0.85838867, 0.94614182,
                0.76238   , 0.61936506, 0.17762283, 0.34838319, 0.46469988,
                0.41927307, 0.65291866, 0.36032995, 0.8486697 , 0.79185662,
                0.37574619, 0.95772318, 0.9395549 , 0.13243252, 0.46226081,
                0.02822198, 0.13886106, 0.67696779, 0.42802597, 0.52429046,
                0.05963306, 0.81099852, 0.15558563, 0.40560201, 0.35178079,
                0.23042666, 0.60499805, 0.81950658, 0.23532297, 0.17013271,
                0.5900594 , 0.36423064, 0.59367366, 0.19799652, 0.5282548 ,
                0.41965851, 0.74052732, 0.34619475, 0.49404885, 0.5884885 ,
                0.11429837, 0.96207555, 0.99669683, 0.23875705, 0.23857609,
                0.12361807, 0.93070321, 0.78029815, 0.90123255, 0.43447557,
                0.98303107, 0.82462018, 0.56393797, 0.91809094, 0.16704594,
                0.93953059, 0.90255658, 0.82120986, 0.71364428, 0.31117955,
                0.14048543, 0.83300399, 0.01534628, 0.7127823 , 0.69707532,
                0.27385047, 0.22415178, 0.33882728, 0.3233493 , 0.10991612,
                0.67643549, 0.30454515, 0.16643281, 0.95915194, 0.05198944,
                0.74860861, 0.16091227, 0.20059127, 0.10486334, 0.70465189,
                0.35538512, 0.15083344, 0.72952782, 0.58549388, 0.94187229,
                0.95161371, 0.24352972, 0.21916936, 0.31477357, 0.6620499 ,
                0.21107415, 0.04326296, 0.32184274, 0.52289941, 0.42383991,
                0.684698  , 0.97141687, 0.50670424, 0.08927118, 0.29177106,
                0.96898295, 0.19214148, 0.90692193, 0.54553041, 0.94604721,
                0.85426947, 0.35093023, 0.0508372 , 0.52053189, 0.22179727,
                0.53193353, 0.33010754, 0.7647021 , 0.95228693, 0.83616635,
                0.04026484, 0.54794611, 0.05015064, 0.55025631, 0.13582363,
                0.91697132, 0.55452534, 0.39577479, 0.33044751, 0.07573318,
                0.83024129, 0.38803213, 0.59408462, 0.33378929, 0.43856983])
```

In [35]:

```
b = A[3:10].copy()
```

In [36]:

```
b
```

Out[36]:

```
array([0.80165012, 0.79582249, 0.44563875, 0.70079419, 0.95915981,
       0.06336659, 0.44502076])
```

In [37]:

```
A[::5]
```

Out[37]:

```
array([0.77026434, 0.44563875, 0.08046927, 0.16428498, 0.39773992,
       0.17016352, 0.99807243, 0.50306103, 0.21506281, 0.15609597,
       0.5459678 , 0.3036959 , 0.07093665, 0.92548779, 0.83943488,
       0.13177198, 0.9726834 , 0.43971233, 0.56505554, 0.32532535,
       0.01113339, 0.36309494, 0.61131511, 0.62132901, 0.76892217,
       0.89181999, 0.89113894, 0.66724779, 0.16533419, 0.92968405,
       0.03221387, 0.22137389, 0.46944902, 0.57561869, 0.91608794,
       0.39057679, 0.80292414, 0.80250346, 0.17613196, 0.0207098 ,
       0.33521551, 0.09388846, 0.07502195, 0.9458198 , 0.58829512,
       0.83156593, 0.70872537, 0.54350514, 0.12480067, 0.0984569 ,
       0.77777127, 0.17940112, 0.71221076, 0.01550795, 0.83776413,
       0.37428249, 0.62176684, 0.39934752, 0.61446797, 0.13320986,
       0.74748187, 0.5444152 , 0.55370418, 0.14371448, 0.06403757,
       0.72858699, 0.40934745, 0.64504614, 0.21874662, 0.87654152,
       0.58877686, 0.38912183, 0.36557544, 0.74704883, 0.74274434,
       0.00321185, 0.72081189, 0.76492495, 0.11305754, 0.0197637 ,
       0.55344811, 0.19969348, 0.61245651, 0.51571603, 0.71620067,
       0.21082632, 0.8643673 , 0.44158434, 0.88179065, 0.3125848 ,
       0.19473968, 0.96099967, 0.21366256, 0.01852125, 0.46706649,
       0.76963591, 0.95158774, 0.13039601, 0.89458431, 0.71022203,
       0.28838933, 0.49185095, 0.99893376, 0.9334529 , 0.55735077,
       0.08575429, 0.85497262, 0.26461634, 0.30924436, 0.69050947,
       0.7128423 , 0.99412722, 0.93791179, 0.45744411, 0.35487218,
       0.68645951, 0.78822182, 0.65090474, 0.6660122 , 0.51466468,
       0.28155262, 0.26727547, 0.94318616, 0.71550222, 0.52402826,
       0.99143988, 0.23009663, 0.02139921, 0.5433488 , 0.26106137,
       0.16940877, 0.52426102, 0.38757643, 0.95447909, 0.35699957,
       0.76375907, 0.57399287, 0.69896909, 0.40329509, 0.20858037,
       0.45249765, 0.7044065 , 0.31750202, 0.77437171, 0.10974601,
       0.023285  , 0.54358796, 0.60761164, 0.3692393 , 0.28618762,
       0.66648752, 0.7231644 , 0.26904183, 0.24477443, 0.41797423,
       0.34158415, 0.36144508, 0.29408234, 0.393601  , 0.50411674,
       0.71714874, 0.95923484, 0.69846445, 0.69317712, 0.70934822,
       0.89867288, 0.05907371, 0.75853757, 0.1267604 , 0.81593362,
       0.28454826, 0.26508358, 0.56667593, 0.84705834, 0.76238   ,
       0.41927307, 0.37574619, 0.02822198, 0.05963306, 0.23042666,
       0.5900594 , 0.41965851, 0.11429837, 0.12361807, 0.98303107,
       0.93953059, 0.14048543, 0.27385047, 0.67643549, 0.74860861,
       0.35538512, 0.95161371, 0.21107415, 0.684698  , 0.96898295,
       0.85426947, 0.53193353, 0.04026484, 0.91697132, 0.83024129])
```

In [38]:

```
A[::-5]
```

Out[38]:

```
array([0.43856983, 0.07573318, 0.13582363, 0.83616635, 0.22179727,
       0.94604721, 0.29177106, 0.42383991, 0.6620499 , 0.94187229,
       0.70465189, 0.05198944, 0.10991612, 0.69707532, 0.31117955,
       0.16704594, 0.43447557, 0.23857609, 0.5884885 , 0.5282548 ,
       0.17013271, 0.35178079, 0.52429046, 0.46226081, 0.79185662,
       0.46469988, 0.94614182, 0.12149285, 0.31484632, 0.73784258,
       0.75720704, 0.68762388, 0.62261134, 0.48421941, 0.79283648,
       0.20305838, 0.23098779, 0.75824999, 0.32890546, 0.72200816,
       0.9750209 , 0.16369496, 0.75815662, 0.0895462 , 0.22683409,
       0.50713154, 0.55634607, 0.42785663, 0.35128705, 0.10031208,
       0.85603611, 0.06041312, 0.42245979, 0.48556747, 0.09699613,
       0.08797291, 0.12710523, 0.72660572, 0.7558361 , 0.28585965,
       0.26956411, 0.85889392, 0.0791552 , 0.63406183, 0.45239705,
       0.39543099, 0.08356748, 0.42127219, 0.89475755, 0.06775832,
       0.14326517, 0.6363805 , 0.22541239, 0.48398161, 0.6611381 ,
       0.08379363, 0.8915191 , 0.37723203, 0.14947367, 0.72834231,
       0.28982349, 0.76723517, 0.53142454, 0.55759463, 0.90618863,
       0.14846725, 0.29864328, 0.24987043, 0.56693528, 0.14156844,
       0.58126293, 0.3595137 , 0.16544769, 0.37341661, 0.62213512,
       0.5987189 , 0.89716631, 0.1313175 , 0.17344438, 0.55982733,
       0.67143379, 0.52814727, 0.52468383, 0.63167019, 0.20008687,
       0.35187013, 0.16075766, 0.98574454, 0.51080476, 0.18153464,
       0.13412536, 0.28076133, 0.89518854, 0.25656095, 0.45612616,
       0.4968434 , 0.77864357, 0.49543469, 0.86322153, 0.64470558,
       0.752333  , 0.3610915 , 0.18526157, 0.1287156 , 0.82637023,
       0.40379822, 0.62284781, 0.74609938, 0.2179114 , 0.64241911,
       0.56468132, 0.32484946, 0.71694176, 0.38083604, 0.85360242,
       0.42184299, 0.72937202, 0.42307719, 0.98586966, 0.5309023 ,
       0.9007303 , 0.89593094, 0.98641573, 0.81734758, 0.95549687,
       0.41557354, 0.84561147, 0.29009066, 0.13243722, 0.25488407,
       0.68483143, 0.92186819, 0.03516003, 0.98707729, 0.18534069,
       0.95054349, 0.65030447, 0.65987276, 0.11543881, 0.3211227 ,
       0.75175448, 0.24187376, 0.82249467, 0.3274769 , 0.12973078,
       0.73196381, 0.42084402, 0.8502546 , 0.50264366, 0.40344283,
       0.52792745, 0.66830293, 0.20772168, 0.96132543, 0.22399838,
       0.49097332, 0.02968544, 0.14820929, 0.12630472, 0.40858765,
       0.43310639, 0.62984902, 0.5476431 , 0.86880712, 0.39656753,
       0.33466574, 0.79682288, 0.38670208, 0.6768769 , 0.45009817,
       0.76922046, 0.85951578, 0.7419035 , 0.03221685, 0.01719621,
       0.34827608, 0.59773908, 0.83843742, 0.44502076, 0.79582249])
```

In [39]:

```
A[::-1]
```

Out[39]:

```
array([0.43856983, 0.33378929, 0.59408462, 0.38803213, 0.83024129,
       0.07573318, 0.33044751, 0.39577479, 0.55452534, 0.91697132,
       0.13582363, 0.55025631, 0.05015064, 0.54794611, 0.04026484,
       0.83616635, 0.95228693, 0.7647021 , 0.33010754, 0.53193353,
       0.22179727, 0.52053189, 0.0508372 , 0.35093023, 0.85426947,
       0.94604721, 0.54553041, 0.90692193, 0.19214148, 0.96898295,
       0.29177106, 0.08927118, 0.50670424, 0.97141687, 0.684698  ,
       0.42383991, 0.52289941, 0.32184274, 0.04326296, 0.21107415,
       0.6620499 , 0.31477357, 0.21916936, 0.24352972, 0.95161371,
       0.94187229, 0.58549388, 0.72952782, 0.15083344, 0.35538512,
       0.70465189, 0.10486334, 0.20059127, 0.16091227, 0.74860861,
       0.05198944, 0.95915194, 0.16643281, 0.30454515, 0.67643549,
       0.10991612, 0.3233493 , 0.33882728, 0.22415178, 0.27385047,
       0.69707532, 0.7127823 , 0.01534628, 0.83300399, 0.14048543,
       0.31117955, 0.71364428, 0.82120986, 0.90255658, 0.93953059,
       0.16704594, 0.91809094, 0.56393797, 0.82462018, 0.98303107,
       0.43447557, 0.90123255, 0.78029815, 0.93070321, 0.12361807,
       0.23857609, 0.23875705, 0.99669683, 0.96207555, 0.11429837,
       0.5884885 , 0.49404885, 0.34619475, 0.74052732, 0.41965851,
       0.5282548 , 0.19799652, 0.59367366, 0.36423064, 0.5900594 ,
       0.17013271, 0.23532297, 0.81950658, 0.60499805, 0.23042666,
       0.35178079, 0.40560201, 0.15558563, 0.81099852, 0.05963306,
       0.52429046, 0.42802597, 0.67696779, 0.13886106, 0.02822198,
       0.46226081, 0.13243252, 0.9395549 , 0.95772318, 0.37574619,
       0.79185662, 0.8486697 , 0.36032995, 0.65291866, 0.41927307,
       0.46469988, 0.34838319, 0.17762283, 0.61936506, 0.76238   ,
       0.94614182, 0.85838867, 0.5633023 , 0.84433588, 0.84705834,
       0.12149285, 0.74649871, 0.07076094, 0.33118006, 0.56667593,
       0.31484632, 0.10481116, 0.32932054, 0.10866101, 0.26508358,
       0.73784258, 0.01472435, 0.6619841 , 0.83624133, 0.28454826,
       0.75720704, 0.45564863, 0.53957389, 0.56345199, 0.81593362,
       0.68762388, 0.1320871 , 0.51943485, 0.60160098, 0.1267604 ,
       0.62261134, 0.96713647, 0.6208909 , 0.16132439, 0.75853757,
       0.48421941, 0.98576741, 0.88205476, 0.2005117 , 0.05907371,
       0.79283648, 0.46560109, 0.23278529, 0.87895235, 0.89867288,
       0.20305838, 0.81523947, 0.51272095, 0.10990493, 0.70934822,
       0.23098779, 0.84186527, 0.78613672, 0.99541505, 0.69317712,
       0.75824999, 0.21609835, 0.28035536, 0.35793347, 0.69846445,
       0.32890546, 0.96677097, 0.84075338, 0.33592716, 0.95923484,
       0.72200816, 0.1804226 , 0.3197432 , 0.02547146, 0.71714874,
       0.9750209 , 0.69989061, 0.16080552, 0.60411394, 0.50411674,
       0.16369496, 0.60886119, 0.4848258 , 0.4368155 , 0.393601  ,
       0.75815662, 0.90225716, 0.00807379, 0.9971096 , 0.29408234,
       0.0895462 , 0.18730465, 0.02060985, 0.93052363, 0.36144508,
       0.22683409, 0.22019957, 0.05949008, 0.49517174, 0.34158415,
       0.50713154, 0.10980735, 0.01630103, 0.90942059, 0.41797423,
       0.55634607, 0.11005976, 0.41054992, 0.34463432, 0.24477443,
       0.42785663, 0.08053577, 0.98467751, 0.93798709, 0.26904183,
       0.35128705, 0.26747159, 0.22564804, 0.72306096, 0.7231644 ,
       0.10031208, 0.57302868, 0.93691942, 0.83041434, 0.66648752,
       0.85603611, 0.17665263, 0.11122876, 0.59324481, 0.28618762,
       0.06041312, 0.52352374, 0.938093  , 0.36259325, 0.3692393 ,
       0.42245979, 0.47807776, 0.84314885, 0.2923162 , 0.60761164,
       0.48556747, 0.26244892, 0.03852865, 0.63927201, 0.54358796,
       0.09699613, 0.39657139, 0.68035166, 0.16900223, 0.023285  ,
       0.08797291, 0.99799332, 0.10435412, 0.15685506, 0.10974601,
       0.12710523, 0.75146201, 0.8285537 , 0.6579632 , 0.77437171,
       0.72660572, 0.85133984, 0.47439107, 0.10272648, 0.31750202,
       0.7558361 , 0.01554331, 0.84487672, 0.04301278, 0.7044065 ,
       0.28585965, 0.2385119 , 0.41123821, 0.46820338, 0.45249765,
       0.26956411, 0.30620094, 0.02821499, 0.31247134, 0.20858037,
       0.85889392, 0.76357385, 0.34352711, 0.58711983, 0.40329509,
       0.0791552 , 0.51217082, 0.69539069, 0.54180878, 0.69896909,
       0.63406183, 0.56189312, 0.95797268, 0.43883622, 0.57399287,
       0.45239705, 0.26475278, 0.57479016, 0.10350306, 0.76375907,
       0.39543099, 0.44702527, 0.24654228, 0.76147383, 0.35699957,
       0.08356748, 0.41298029, 0.25191399, 0.9485322 , 0.95447909,
       0.42127219, 0.81315311, 0.96929306, 0.99942716, 0.38757643,
       0.89475755, 0.65552164, 0.78327545, 0.90066524, 0.52426102,
       0.06775832, 0.28974524, 0.95694932, 0.3051302 , 0.16940877,
       0.14326517, 0.43911422, 0.2212328 , 0.14283195, 0.26106137,
       0.6363805 , 0.01086025, 0.64513601, 0.65665161, 0.5433488 ,
       0.22541239, 0.55903524, 0.57306707, 0.33987909, 0.02139921,
       0.48398161, 0.38346419, 0.53701024, 0.85679251, 0.23009663,
       0.6611381 , 0.15984446, 0.23453366, 0.48410359, 0.99143988,
       0.08379363, 0.77195483, 0.38369041, 0.59082763, 0.52402826,
       0.8915191 , 0.58707558, 0.25641215, 0.74631845, 0.71550222,
       0.37723203, 0.20076601, 0.61125782, 0.84141256, 0.94318616,
       0.14947367, 0.130108  , 0.41561861, 0.55026965, 0.26727547,
       0.72834231, 0.57294071, 0.99991711, 0.3791768 , 0.28155262,
       0.28982349, 0.35581948, 0.32591193, 0.95240351, 0.51466468,
       0.76723517, 0.43979756, 0.82038361, 0.72110339, 0.6660122 ,
       0.53142454, 0.03949754, 0.6156222 , 0.76078372, 0.65090474,
       0.55759463, 0.90062284, 0.17098786, 0.03439745, 0.78822182,
       0.90618863, 0.01802463, 0.70920362, 0.24996491, 0.68645951,
       0.14846725, 0.57027735, 0.99425794, 0.2785915 , 0.35487218,
       0.29864328, 0.95997753, 0.60552957, 0.98539981, 0.45744411,
       0.24987043, 0.03023662, 0.21680376, 0.7295903 , 0.93791179,
       0.56693528, 0.89296823, 0.83400307, 0.4003222 , 0.99412722,
       0.14156844, 0.90133098, 0.83159169, 0.46069748, 0.7128423 ,
       0.58126293, 0.92611605, 0.31402295, 0.53737603, 0.69050947,
       0.3595137 , 0.33930344, 0.30718214, 0.26000785, 0.30924436,
       0.16544769, 0.74177554, 0.90150673, 0.83892533, 0.26461634,
       0.37341661, 0.48642862, 0.88290139, 0.09149005, 0.85497262,
       0.62213512, 0.04870349, 0.77808789, 0.30737407, 0.08575429,
       0.5987189 , 0.05210105, 0.85088651, 0.75730473, 0.55735077,
       0.89716631, 0.29556886, 0.90462646, 0.84238332, 0.9334529 ,
       0.1313175 , 0.06495258, 0.33145612, 0.83641023, 0.99893376,
       0.17344438, 0.92921557, 0.96712727, 0.24215387, 0.49185095,
       0.55982733, 0.11877644, 0.67602893, 0.17608335, 0.28838933,
       0.67143379, 0.48797424, 0.47691967, 0.95077294, 0.71022203,
       0.52814727, 0.45675276, 0.10212392, 0.7251878 , 0.89458431,
       0.52468383, 0.46025951, 0.73291108, 0.24834131, 0.13039601,
       0.63167019, 0.56562543, 0.12031078, 0.19504981, 0.95158774,
       0.20008687, 0.78166727, 0.25654795, 0.86598345, 0.76963591,
       0.35187013, 0.47046335, 0.00556098, 0.03945326, 0.46706649,
       0.16075766, 0.58993917, 0.17412098, 0.54702506, 0.01852125,
       0.98574454, 0.97815673, 0.35346929, 0.82620408, 0.21366256,
       0.51080476, 0.04787651, 0.64140646, 0.61072978, 0.96099967,
       0.18153464, 0.06657801, 0.81481623, 0.33514031, 0.19473968,
       0.13412536, 0.73692095, 0.19802284, 0.78246519, 0.3125848 ,
       0.28076133, 0.78040349, 0.91437028, 0.79163753, 0.88179065,
       0.89518854, 0.61016887, 0.93793681, 0.31668665, 0.44158434,
       0.25656095, 0.75206069, 0.70101893, 0.55078441, 0.8643673 ,
       0.45612616, 0.54255385, 0.55803396, 0.12843107, 0.21082632,
       0.4968434 , 0.76916622, 0.37897017, 0.82508605, 0.71620067,
       0.77864357, 0.18801767, 0.79902712, 0.55567722, 0.51571603,
       0.49543469, 0.21725342, 0.13231569, 0.54221913, 0.61245651,
       0.86322153, 0.30038728, 0.10145397, 0.41637094, 0.19969348,
       0.64470558, 0.59234471, 0.48972942, 0.52349067, 0.55344811,
       0.752333  , 0.40692234, 0.68734563, 0.53531827, 0.0197637 ,
       0.3610915 , 0.39860508, 0.35577268, 0.33233995, 0.11305754,
       0.18526157, 0.30724746, 0.40590627, 0.94866782, 0.76492495,
       0.1287156 , 0.26957187, 0.43285147, 0.55080113, 0.72081189,
       0.82637023, 0.91552023, 0.75528171, 0.7707281 , 0.00321185,
       0.40379822, 0.07220789, 0.9247873 , 0.54673389, 0.74274434,
       0.62284781, 0.37901205, 0.22230386, 0.28833853, 0.74704883,
       0.74609938, 0.52103552, 0.09787099, 0.19128929, 0.36557544,
       0.2179114 , 0.15291816, 0.81270334, 0.74236341, 0.38912183,
       0.64241911, 0.90390773, 0.52372973, 0.74461306, 0.58877686,
       0.56468132, 0.72784559, 0.23986183, 0.05778186, 0.87654152,
       0.32484946, 0.28234685, 0.00216026, 0.5829091 , 0.21874662,
       0.71694176, 0.74364784, 0.85793528, 0.78448414, 0.64504614,
       0.38083604, 0.30783779, 0.23237176, 0.75967722, 0.40934745,
       0.85360242, 0.93522377, 0.00549775, 0.03181857, 0.72858699,
       0.42184299, 0.11015288, 0.84401628, 0.00281294, 0.06403757,
       0.72937202, 0.96347628, 0.49191962, 0.39497885, 0.14371448,
       0.42307719, 0.11854738, 0.58796262, 0.25323834, 0.55370418,
       0.98586966, 0.11096725, 0.3587078 , 0.08044721, 0.5444152 ,
       0.5309023 , 0.59516157, 0.49809807, 0.75784881, 0.74748187,
       0.9007303 , 0.58562655, 0.29168532, 0.18964452, 0.13320986,
       0.89593094, 0.18735551, 0.0821788 , 0.61915585, 0.61446797,
       0.98641573, 0.68868698, 0.91535474, 0.99078844, 0.39934752,
       0.81734758, 0.89133129, 0.18610681, 0.23377041, 0.62176684,
       0.95549687, 0.74369917, 0.17722441, 0.18040376, 0.37428249,
       0.41557354, 0.38280974, 0.27328467, 0.12406843, 0.83776413,
       0.84561147, 0.79465273, 0.21184647, 0.5680643 , 0.01550795,
       0.29009066, 0.59012042, 0.0451028 , 0.78066395, 0.71221076,
       0.13243722, 0.40155011, 0.55539555, 0.4582296 , 0.17940112,
       0.25488407, 0.33798839, 0.00101482, 0.37179014, 0.77777127,
       0.68483143, 0.9348532 , 0.52685751, 0.14002596, 0.0984569 ,
       0.92186819, 0.71133314, 0.76124202, 0.88501056, 0.12480067,
       0.03516003, 0.93123307, 0.1634944 , 0.98698071, 0.54350514,
       0.98707729, 0.22086194, 0.80734308, 0.79684665, 0.70872537,
       0.18534069, 0.57272094, 0.4908456 , 0.7508333 , 0.83156593,
       0.95054349, 0.05108577, 0.12364225, 0.59614768, 0.58829512,
       0.65030447, 0.80976373, 0.23454272, 0.19813737, 0.9458198 ,
       0.65987276, 0.42338107, 0.35526579, 0.23562092, 0.07502195,
       0.11543881, 0.8948441 , 0.3196715 , 0.53128582, 0.09388846,
              0.3211227 , 0.57511779, 0.86648268, 0.39317558, 0.33521551,
              0.75175448, 0.67403023, 0.77502043, 0.47960053, 0.0207098 ,
              0.24187376, 0.43734016, 0.15543877, 0.39943043, 0.17613196,
              0.82249467, 0.90368409, 0.5834477 , 0.31574251, 0.80250346,
              0.3274769 , 0.03311551, 0.00305526, 0.86849003, 0.80292414,
              0.12973078, 0.8163277 , 0.36249796, 0.44385356, 0.39057679,
              0.73196381, 0.89295721, 0.81317426, 0.75798824, 0.91608794,
              0.42084402, 0.77662448, 0.74793079, 0.97964561, 0.57561869,
              0.8502546 , 0.91692377, 0.57922464, 0.54062536, 0.46944902,
              0.50264366, 0.69453103, 0.86706534, 0.61918964, 0.22137389,
              0.40344283, 0.74834081, 0.98222792, 0.73872159, 0.03221387,
              0.52792745, 0.60015455, 0.56980084, 0.34565633, 0.92968405,
              0.66830293, 0.03313501, 0.32854969, 0.64071731, 0.16533419,
              0.20772168, 0.46448946, 0.56135538, 0.91956189, 0.66724779,
              0.96132543, 0.2189735 , 0.67738049, 0.06202619, 0.89113894,
              0.22399838, 0.57660697, 0.27950533, 0.4002134 , 0.89181999,
              0.49097332, 0.89546309, 0.54333057, 0.15960765, 0.76892217,
              0.02968544, 0.48040499, 0.45415074, 0.04160285, 0.62132901,
              0.14820929, 0.8821381 , 0.02187398, 0.78436036, 0.61131511,
              0.12630472, 0.35118661, 0.77130556, 0.84652078, 0.36309494,
              0.40858765, 0.79941111, 0.55626764, 0.52228207, 0.01113339,
              0.43310639, 0.80474479, 0.86274006, 0.88566533, 0.32532535,
              0.62984902, 0.15427435, 0.03041463, 0.10635394, 0.56505554,
              0.5476431 , 0.06835135, 0.05184361, 0.59865833, 0.43971233,
              0.86880712, 0.00664431, 0.15009946, 0.45619181, 0.9726834 ,
              0.39656753, 0.38664412, 0.71865744, 0.58917854, 0.13177198,
              0.33466574, 0.1845831 , 0.03272602, 0.71079411, 0.83943488,
              0.79682288, 0.56078631, 0.57731992, 0.05469635, 0.92548779,
              0.38670208, 0.77320752, 0.93128264, 0.59723077, 0.07093665,
              0.6768769 , 0.68800907, 0.71072402, 0.53701378, 0.3036959 ,
              0.45009817, 0.90002384, 0.46538612, 0.11661917, 0.5459678 ,
              0.76922046, 0.40752759, 0.68891486, 0.80585962, 0.15609597,
              0.85951578, 0.66578939, 0.39730284, 0.38042846, 0.21506281,
              0.7419035 , 0.34859884, 0.67082695, 0.99915883, 0.50306103,
              0.03221685, 0.22582597, 0.37330331, 0.72065194, 0.99807243,
              0.01719621, 0.32795093, 0.52619929, 0.28163911, 0.17016352,
              0.34827608, 0.57264117, 0.93053297, 0.6762144 , 0.39773992,
              0.59773908, 0.10729968, 0.67893428, 0.95149733, 0.16428498,
              0.83843742, 0.03861415, 0.05006767, 0.21248426, 0.08046927,
              0.44502076, 0.06336659, 0.95915981, 0.70079419, 0.44563875,
              0.79582249, 0.80165012, 0.82739175, 0.40507102, 0.77026434])
```

```
In [40]: B = (A ==-1299)*np.arange(A.size)
```

```
In [33]: B
```

```
Out[33]: array([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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 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,
       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, 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, 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, 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, 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, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

In [103…

```
A
```

Out[103]:

```
array([    0,     1,     2, -1299,     4,     5,     6,     7,     8,
           9,    10,    11,    12,    13,    14,    15,    16,    17,
          18,    19,    20,    21,    22,    23,    24,    25,    26,
          27,    28,    29,    30,    31,    32,    33,    34,    35,
          36,    37,    38,    39,    40,    41,    42,    43,    44,
          45,    46,    47,    48,    49,    50,    51,    52,    53,
          54,    55,    56,    57,    58,    59,    60,    61,    62,
          63,    64,    65,    66,    67,    68,    69,    70,    71,
          72,    73,    74,    75,    76,    77,    78,    79,    80,
          81,    82,    83,    84,    85,    86,    87,    88,    89,
          90,    91,    92,    93,    94,    95,    96,    97,    98,
          99])
```

In [104…

```
idx = np.argwhere(A==-1299)[0][0]  ## To find the index of a particular element
```

In [105…

```
idx
```

Out[105]:

```
3
```

In [106…

```
A[idx] = 3 ## To replace the values on that element
```

In [107…

```
A
```

Out[107]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [41]:

```
A = np.round(10*np.random.rand(5,4))
```

In [42]:

```
A
```

```
Out[42]: array([[ 4.,  9.,  5.,  1.],
                [ 5.,  3.,  9.,  9.],
                [ 7.,  1.,  8.,  2.],
                [ 8.,  7.,  8., 10.],
                [ 4.,  7.,  0.,  9.]])
```

```
In [45]: A[1,2] = 8
```

```
In [46]: A
```

```
Out[46]: array([[ 4.,  9.,  5.,  1.],
                [ 5.,  3.,  8.,  9.],
                [ 7.,  1.,  8.,  2.],
                [ 8.,  7.,  8., 10.],
                [ 4.,  7.,  0.,  9.]])
```

```
In [117… A[1,2]
```

```
Out[117]: 7.0
```

```
In [118… A[1,:]
```

```
Out[118]: array([6., 6., 7., 9.])
```

```
In [119… A[:,1]
```

```
Out[119]: array([3., 6., 4., 1., 6.])
```

```
In [121… A[1:3,2:4]
```

```
Out[121]: array([[7., 9.],
                 [1., 6.]])
```

```
In [122… A.T #Transpose
```

```
Out[122]: array([[4., 6., 6., 4., 6.],
                 [3., 6., 4., 1., 6.],
                 [9., 7., 1., 4., 6.],
                 [7., 9., 6., 9., 1.]])
```

```
In [48]: import numpy.linalg as la
```

```
In [49]: la.inv(np.random.rand(3,3))
```

```
Out[49]: array([[ 18.57192609,   8.61785398, -23.59470737],
                [  4.20808187,   0.54888623,  -3.02211905],
                [-22.95885659,  -6.66891505,  25.79930309]])
```

```
In [125… A
```

```
Out[125]: array([[4., 3., 9., 7.],
                 [6., 6., 7., 9.],
                 [6., 4., 1., 6.],
                 [4., 1., 4., 9.],
                 [6., 6., 6., 1.]])
```

```
In [126… A.sort(axis = 0 )
```

```
In [127… A
```

```
Out[127]: array([[4., 1., 1., 1.],
                 [4., 3., 4., 6.],
                 [6., 4., 6., 7.],
                 [6., 6., 7., 9.],
                 [6., 6., 9., 9.]])
```

In [129…

```
A.sort(axis = 1)
```

In [130…

```
A
```

Out[130]:

```
array([[1., 1., 1., 4.],
       [3., 4., 4., 6.],
       [4., 6., 6., 7.],
       [6., 6., 7., 9.],
       [6., 6., 9., 9.]])
```

In [131…

```
#Index_array
```

In [132…

```
A = np.arange(100)
```

In [133…

```
B = A[[3,5,6]]
```

In [134…

```
B
```

Out[134]:

```
array([3, 5, 6])
```

In [136…

```
B[0] = -4
```

In [137…

```
B
```

Out[137]:

```
array([-4,  5,  6])
```

In [138…

```
A
```

Out[138]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [139…

```
B = A[A<40]
```

In [140…

```
B
```

Out[140]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39])
```

In [141…

```
B = A[(A<40) & (A>30)]
```

In [142…

```
B
```

Out[142]:

```
array([31, 32, 33, 34, 35, 36, 37, 38, 39])
```

In [143…

```
# & , and
# | , or
# ~, not
```

In [144…

```
#Broadcasting
```

In [145…

```
A = np.arange(100)
```

In [146…

```
A
```

Out[146]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [147…

```
A = A + 5
```

In [148…

```
A
```

Out[148]:

```
array([  5,   6,   7,   8,   9,  10,  11,  12,  13,  14,  15,  16,  17,
        18,  19,  20,  21,  22,  23,  24,  25,  26,  27,  28,  29,  30,
        31,  32,  33,  34,  35,  36,  37,  38,  39,  40,  41,  42,  43,
        44,  45,  46,  47,  48,  49,  50,  51,  52,  53,  54,  55,  56,
        57,  58,  59,  60,  61,  62,  63,  64,  65,  66,  67,  68,  69,
        70,  71,  72,  73,  74,  75,  76,  77,  78,  79,  80,  81,  82,
        83,  84,  85,  86,  87,  88,  89,  90,  91,  92,  93,  94,  95,
        96,  97,  98,  99, 100, 101, 102, 103, 104])
```

In [152…

```
A = np.round(A +1/3,1)*10
```

In [153…

```
A
```

Out[153]:

```
array([  57.,   67.,   77.,   87.,   97.,  107.,  117.,  127.,  137.,
        147.,  157.,  167.,  177.,  187.,  197.,  207.,  217.,  227.,
        237.,  247.,  257.,  267.,  277.,  287.,  297.,  307.,  317.,
        327.,  337.,  347.,  357.,  367.,  377.,  387.,  397.,  407.,
        417.,  427.,  437.,  447.,  457.,  467.,  477.,  487.,  497.,
        507.,  517.,  527.,  537.,  547.,  557.,  567.,  577.,  587.,
        597.,  607.,  617.,  627.,  637.,  647.,  657.,  667.,  677.,
        687.,  697.,  707.,  717.,  727.,  737.,  747.,  757.,  767.,
        777.,  787.,  797.,  807.,  817.,  827.,  837.,  847.,  857.,
        867.,  877.,  887.,  897.,  907.,  917.,  927.,  937.,  947.,
        957.,  967.,  977.,  987.,  997., 1007., 1017., 1027., 1037.,
       1047.])
```

In [159…

```
#np.hstack = concat two arrays horizontally
#np.vstack = concat two arrays vertically
#np.sort
#Universal
#Faster
```

In [161…

```
A+(np.arange(2).reshape(2,1))
```

Out[161]:

```
array([[  57.,   67.,   77.,   87.,   97.,  107.,  117.,  127.,  137.,
         147.,  157.,  167.,  177.,  187.,  197.,  207.,  217.,  227.,
         237.,  247.,  257.,  267.,  277.,  287.,  297.,  307.,  317.,
         327.,  337.,  347.,  357.,  367.,  377.,  387.,  397.,  407.,
         417.,  427.,  437.,  447.,  457.,  467.,  477.,  487.,  497.,
         507.,  517.,  527.,  537.,  547.,  557.,  567.,  577.,  587.,
         597.,  607.,  617.,  627.,  637.,  647.,  657.,  667.,  677.,
         687.,  697.,  707.,  717.,  727.,  737.,  747.,  757.,  767.,
         777.,  787.,  797.,  807.,  817.,  827.,  837.,  847.,  857.,
         867.,  877.,  887.,  897.,  907.,  917.,  927.,  937.,  947.,
         957.,  967.,  977.,  987.,  997., 1007., 1017., 1027., 1037.,
        1047.],
       [  58.,   68.,   78.,   88.,   98.,  108.,  118.,  128.,  138.,
         148.,  158.,  168.,  178.,  188.,  198.,  208.,  218.,  228.,
         238.,  248.,  258.,  268.,  278.,  288.,  298.,  308.,  318.,
         328.,  338.,  348.,  358.,  368.,  378.,  388.,  398.,  408.,
         418.,  428.,  438.,  448.,  458.,  468.,  478.,  488.,  498.,
         508.,  518.,  528.,  538.,  548.,  558.,  568.,  578.,  588.,
         598.,   608.,   618.,   628.,   638.,   648.,   658.,   668.,   678.,
         688.,   698.,   708.,   718.,   728.,   738.,   748.,   758.,   768.,
         778.,   788.,   798.,   808.,   818.,   828.,   838.,   848.,   858.,
         868.,   878.,   888.,   898.,   908.,   918.,   928.,   938.,   948.,
         958.,   968.,   978.,   988.,   998.,  1008.,  1018.,  1028.,  1038.,
        1048.]])
```

```
In [4]: B = (np.round(10*np.random.rand(2,3)))
```

```
In [5]: B
```

```
Out[5]: array([[3., 2., 5.],
               [2., 7., 4.]])
```

```
In [6]: B + 3
```

```
Out[6]: array([[ 6.,  5.,  8.],
               [ 5., 10.,  7.]])
```

```
In [7]: B+np.arange(2).reshape(2,1)
```

```
Out[7]: array([[3., 2., 5.],
               [3., 8., 5.]])
```

```
In [8]: B
```

```
Out[8]: array([[3., 2., 5.],
               [2., 7., 4.]])
```

```
In [9]: C = np.round(10*np.random.rand(2,2))
```

```
In [10]: C
```

```
Out[10]: array([[8., 6.],
                [7., 3.]])
```

```
In [14]: C = np.hstack((B,C))
```

```
In [15]: C
```

```
Out[15]: array([[3., 2., 5., 8., 6.],
                [2., 7., 4., 7., 3.]])
```

```
In [16]: A = np.random.permutation(np.arange(10))
```

```
In [17]: A
```

```
Out[17]: array([3, 5, 7, 2, 8, 0, 6, 1, 9, 4])
```

```
In [18]: A.sort()
```

```
In [19]: A
```

```
Out[19]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [20]: np.sort(A)
```

```
Out[20]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [21]: A = A[::-1]
```

```
In [22]: A
```

```
Out[22]: array([9, 8, 7, 6, 5, 4, 3, 2, 1, 0])

In [23]: A = np.array(["abc","How are you","fih","dheui"])

In [24]: A

Out[24]: array(['abc', 'How are you', 'fih', 'dheui'], dtype='<U11')

In [26]: A.sort()

In [27]: A

Out[27]: array(['How are you', 'abc', 'dheui', 'fih'], dtype='<U11')

In [28]: B = np.random.rand(1000000)

In [29]: %timeit sum(B)
         %timeit np.sum(B)

         66.9 ms ± 1.19 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
         725 µs ± 42.4 µs per loop (mean ± std. dev. of 7 runs, 1,000 loops each)

In [30]: def mySum(G):
             s = 0
             for x in G:
                 s+=x
             return s

In [31]: %timeit mySum(B)

         116 ms ± 6 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

Pandas

```
In [51]: import pandas as pd

In [33]: print(pd.__version__)

         2.0.3

In [34]: A = pd.Series([2,3,4,5],index = ["a","b","c","d"])

In [35]: A.values

Out[35]: array([2, 3, 4, 5], dtype=int64)

In [37]: type(A.values)

Out[37]: numpy.ndarray

In [38]: A.index

Out[38]: Index(['a', 'b', 'c', 'd'], dtype='object')

In [39]: A["a"]

Out[39]: 2
```

```
In [40]: A["a":"c"]

Out[40]: a    2
         b    3
         c    4
         dtype: int64
```

```
In [56]: grades_dict = {"A":4,"B":4.5,"C":3,"D":2.5}
         grades = pd.Series(grades_dict)
```

```
In [45]: grades.values

Out[45]: array([4. , 4.5, 3. , 2.5])
```

```
In [46]: grades.index

Out[46]: Index(['A', 'B', 'C', 'D'], dtype='object')
```

```
In [54]: marks_dict = {"A":85,"B":75,"C":65,"D":55}
         marks= pd.Series(marks_dict)
```

```
In [49]: marks

Out[49]: A    85
         B    75
         C    65
         D    55
         dtype: int64
```

```
In [50]: marks[0:2]

Out[50]: A    85
         B    75
         dtype: int64
```

```
In [51]: #Data frame
         marks

Out[51]: A    85
         B    75
         C    65
         D    55
         dtype: int64
```

```
In [57]: grades

Out[57]: A    4.0
         B    4.5
         C    3.0
         D    2.5
         dtype: float64
```

```
In [58]: D = pd.DataFrame({"Marks":marks,"Grades":grades})
```

```
In [55]: D
```

Out[55]:

	Marks	Grades
A	85	4.0
B	75	4.5
C	65	3.0
D	55	2.5

In [56]:

```
D.T
```

Out[56]:

	A	B	C	D
Marks	85.0	75.0	65.0	55.0
Grades	4.0	4.5	3.0	2.5

In [57]:

```
D
```

Out[57]:

	Marks	Grades
A	85	4.0
B	75	4.5
C	65	3.0
D	55	2.5

In [58]:

```
D.values
```

Out[58]:

```
array([[85. ,  4. ],
       [75. ,  4.5],
       [65. ,  3. ],
       [55. ,  2.5]])
```

In [59]:

```
D.values[2,0]
```

Out[59]:

```
65.0
```

In [60]:

```
D.columns
```

Out[60]:

```
Index(['Marks', 'Grades'], dtype='object')
```

In [59]:

```
D["ScaledMarks"] = 100*D["Marks"]/90
```

In [60]:

```
D
```

Out[60]:

	Marks	Grades	ScaledMarks
A	85	4.0	94.444444
B	75	4.5	83.333333
C	65	3.0	72.222222
D	55	2.5	61.111111

In [64]:

```
del D["ScaledMarks"]
```

In [65]:

```
D
```

Out[65]:

	Marks	Grades
A	85	4.0
B	75	4.5
C	65	3.0
D	55	2.5

```
In [61]: G = D[D["Marks"]>70]
```

```
In [62]: G
```

Out[62]:

	Marks	Grades	ScaledMarks
A	85	4.0	94.444444
B	75	4.5	83.333333

```
In [63]: A = pd.DataFrame([{"a":1,"b":4},{"b":-3,"c":9}])
```

```
In [64]: A
```

Out[64]:

	a	b	c
0	1.0	4	NaN
1	NaN	-3	9.0

```
In [65]: A.fillna(0)
```

Out[65]:

	a	b	c
0	1.0	4	0.0
1	0.0	-3	9.0

```
In [66]: A.dropna?
```

```
In [67]: A = pd.Series(["a","b","c"], index = [1,3,5])
```

```
In [68]: A[1]
```

```
Out[68]: 'a'
```

```
In [69]: A[1:3]
```

```
Out[69]: 3    b
         5    c
         dtype: object
```

```
In [70]: A.loc[1:3]
         # For label indexing
```

```
Out[70]: 1    a
         3    b
         dtype: object
```

```
In [71]: A.iloc[1:3]
         # For Interger indexing
```

```
Out[71]: 3    b
         5    c
         dtype: object
```

```
In [72]: D
```

Out[72]:

	Marks	Grades	ScaledMarks

A	85	4.0	94.444444
B	75	4.5	83.333333
C	65	3.0	72.222222
D	55	2.5	61.111111

In [73]:
```
D.iloc[2,:]
```

Out[73]:
```
Marks           65.000000
Grades           3.000000
ScaledMarks     72.222222
Name: C, dtype: float64
```

In [74]:
```
D.iloc[::-1,:]
```

Out[74]:

	Marks	Grades	ScaledMarks
D	55	2.5	61.111111
C	65	3.0	72.222222
B	75	4.5	83.333333
A	85	4.0	94.444444

In [75]:
```
import numpy as np
import pandas as pd
```

In [78]:
```
from sklearn.impute import SimpleImputer
```

In [79]:
```
df = pd.read_csv("F:\DOWNLOADS NEW/covid_19_data.csv")
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[79], line 1
----> 1 df = pd.read_csv("F:\DOWNLOADS NEW/covid_19_data.csv")

File ~\anaconda3\Lib\site-packages\pandas\io\parsers\readers.py:912, in read_csv(filepat
h_or_buffer, sep, delimiter, header, names, index_col, usecols, dtype, engine, converter
s, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values,
 keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_form
at, keep_date_col, date_parser, date_format, dayfirst, cache_dates, iterator, chunksize,
 compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escap
echar, comment, encoding, encoding_errors, dialect, on_bad_lines, delim_whitespace, low_
memory, memory_map, float_precision, storage_options, dtype_backend)
    899 kwds_defaults = _refine_defaults_read(
    900     dialect,
    901     delimiter,
   (...)
    908     dtype_backend=dtype_backend,
    909 )
    910 kwds.update(kwds_defaults)
--> 912 return _read(filepath_or_buffer, kwds)

File ~\anaconda3\Lib\site-packages\pandas\io\parsers\readers.py:577, in _read(filepath_o
r_buffer, kwds)
    574 _validate_names(kwds.get("names", None))
    576 # Create the parser.
--> 577 parser = TextFileReader(filepath_or_buffer, **kwds)
    579 if chunksize or iterator:
    580     return parser
```

```
File ~\anaconda3\Lib\site-packages\pandas\io\parsers\readers.py:1407, in TextFileReader.
__init__(self, f, engine, **kwds)
   1404     self.options["has_index_names"] = kwds["has_index_names"]
   1406 self.handles: IOHandles | None = None
-> 1407 self._engine = self._make_engine(f, self.engine)

File ~\anaconda3\Lib\site-packages\pandas\io\parsers\readers.py:1661, in TextFileReader.
_make_engine(self, f, engine)
   1659     if "b" not in mode:
   1660         mode += "b"
-> 1661 self.handles = get_handle(
   1662     f,
   1663     mode,
   1664     encoding=self.options.get("encoding", None),
   1665     compression=self.options.get("compression", None),
   1666     memory_map=self.options.get("memory_map", False),
   1667     is_text=is_text,
   1668     errors=self.options.get("encoding_errors", "strict"),
   1669     storage_options=self.options.get("storage_options", None),
   1670 )
   1671 assert self.handles is not None
   1672 f = self.handles.handle

File ~\anaconda3\Lib\site-packages\pandas\io\common.py:859, in get_handle(path_or_buf, m
ode, encoding, compression, memory_map, is_text, errors, storage_options)
    854 elif isinstance(handle, str):
    855     # Check whether the filename is to be opened in binary mode.
    856     # Binary mode does not support 'encoding' and 'newline'.
    857     if ioargs.encoding and "b" not in ioargs.mode:
    858         # Encoding
--> 859         handle = open(
    860             handle,
    861             ioargs.mode,
    862             encoding=ioargs.encoding,
    863             errors=errors,
    864             newline="",
    865         )
    866     else:
    867         # Binary mode
    868         handle = open(handle, ioargs.mode)

FileNotFoundError: [Errno 2] No such file or directory: 'F:\\DOWNLOADS NEW/covid_19_dat
a.csv'
```

In [6]:
```
df.head(30)
```

Out[6]:

	SNo	ObservationDate	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	1	01/22/2020	Anhui	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
1	2	01/22/2020	Beijing	Mainland China	1/22/2020 17:00	14.0	0.0	0.0
2	3	01/22/2020	Chongqing	Mainland China	1/22/2020 17:00	6.0	0.0	0.0
3	4	01/22/2020	Fujian	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
4	5	01/22/2020	Gansu	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
5	6	01/22/2020	Guangdong	Mainland China	1/22/2020 17:00	26.0	0.0	0.0
6	7	01/22/2020	Guangxi	Mainland China	1/22/2020 17:00	2.0	0.0	0.0
7	8	01/22/2020	Guizhou	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
8	9	01/22/2020	Hainan	Mainland China	1/22/2020 17:00	4.0	0.0	0.0
9	10	01/22/2020	Hebei	Mainland China	1/22/2020 17:00	1.0	0.0	0.0

10	11	01/22/2020	Heilongjiang	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
11	12	01/22/2020	Henan	Mainland China	1/22/2020 17:00	5.0	0.0	0.0
12	13	01/22/2020	Hong Kong	Hong Kong	1/22/2020 17:00	0.0	0.0	0.0
13	14	01/22/2020	Hubei	Mainland China	1/22/2020 17:00	444.0	17.0	28.0
14	15	01/22/2020	Hunan	Mainland China	1/22/2020 17:00	4.0	0.0	0.0
15	16	01/22/2020	Inner Mongolia	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
16	17	01/22/2020	Jiangsu	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
17	18	01/22/2020	Jiangxi	Mainland China	1/22/2020 17:00	2.0	0.0	0.0
18	19	01/22/2020	Jilin	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
19	20	01/22/2020	Liaoning	Mainland China	1/22/2020 17:00	2.0	0.0	0.0
20	21	01/22/2020	Macau	Macau	1/22/2020 17:00	1.0	0.0	0.0
21	22	01/22/2020	Ningxia	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
22	23	01/22/2020	Qinghai	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
23	24	01/22/2020	Shaanxi	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
24	25	01/22/2020	Shandong	Mainland China	1/22/2020 17:00	2.0	0.0	0.0
25	26	01/22/2020	Shanghai	Mainland China	1/22/2020 17:00	9.0	0.0	0.0
26	27	01/22/2020	Shanxi	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
27	28	01/22/2020	Sichuan	Mainland China	1/22/2020 17:00	5.0	0.0	0.0
28	29	01/22/2020	Taiwan	Taiwan	1/22/2020 17:00	1.0	0.0	0.0
29	30	01/22/2020	Tianjin	Mainland China	1/22/2020 17:00	4.0	0.0	0.0

```
In [8]: df.drop(["SNo","Last Update"],axis=1,inplace=True)
```

```
In [9]: df.head()
```

Out[9]:

	ObservationDate	Province/State	Country/Region	Confirmed	Deaths	Recovered
0	01/22/2020	Anhui	Mainland China	1.0	0.0	0.0
1	01/22/2020	Beijing	Mainland China	14.0	0.0	0.0
2	01/22/2020	Chongqing	Mainland China	6.0	0.0	0.0
3	01/22/2020	Fujian	Mainland China	1.0	0.0	0.0
4	01/22/2020	Gansu	Mainland China	0.0	0.0	0.0

```
In [17]: df.rename(columns={"ObservationDate":"Date",
                           "Province/State":"Province","Country/Region":"Country"},inplace = Tru
```

```
In [18]: df.head()
```

Out[18]:

	Date	Province	Country	Confirmed	Deaths	Recovered
0	01/22/2020	Anhui	Mainland China	1.0	0.0	0.0
1	01/22/2020	Beijing	Mainland China	14.0	0.0	0.0
2	01/22/2020	Chongqing	Mainland China	6.0	0.0	0.0

3	01/22/2020	Fujian	Mainland China	1.0	0.0	0.0
4	01/22/2020	Gansu	Mainland China	0.0	0.0	0.0

In [19]:
```
df["Date"]=pd.to_datetime(df["Date"])
```

In [22]:
```
df.head()
```

Out[22]:

	Date	Province	Country	Confirmed	Deaths	Recovered
0	2020-01-22	Anhui	Mainland China	1.0	0.0	0.0
1	2020-01-22	Beijing	Mainland China	14.0	0.0	0.0
2	2020-01-22	Chongqing	Mainland China	6.0	0.0	0.0
3	2020-01-22	Fujian	Mainland China	1.0	0.0	0.0
4	2020-01-22	Gansu	Mainland China	0.0	0.0	0.0

In [8]:
```
df.describe()
```

Out[8]:

	SNo	Confirmed	Deaths	Recovered
count	4247.000000	4247.000000	4247.000000	4247.000000
mean	2124.000000	586.884624	17.530257	187.914528
std	1226.147626	5033.596411	190.278672	1976.388824
min	1.000000	0.000000	0.000000	0.000000
25%	1062.500000	1.000000	0.000000	0.000000
50%	2124.000000	9.000000	0.000000	1.000000
75%	3185.500000	99.500000	1.000000	16.000000
max	4247.000000	67707.000000	2986.000000	45235.000000

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4247 entries, 0 to 4246
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   SNo              4247 non-null   int64  
 1   ObservationDate  4247 non-null   object 
 2   Province/State   2746 non-null   object 
 3   Country/Region   4247 non-null   object 
 4   Last Update      4247 non-null   object 
 5   Confirmed        4247 non-null   float64
 6   Deaths           4247 non-null   float64
 7   Recovered        4247 non-null   float64
dtypes: float64(3), int64(1), object(4)
memory usage: 265.6+ KB
```

In [11]:
```
df = df.fillna("NA")
```

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4247 entries, 0 to 4246
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   SNo              4247 non-null   int64  
 1   ObservationDate  4247 non-null   object 
 2   Province/State   4247 non-null   object 
 3   Country/Region   4247 non-null   object 
 4   Last Update      4247 non-null   object 
 5   Confirmed        4247 non-null   float64
 6   Deaths           4247 non-null   float64
 7   Recovered        4247 non-null   float64
dtypes: float64(3), int64(1), object(4)
memory usage: 265.6+ KB
```

In [14]:

```
df.head(10)
```

Out[14]:

	SNo	ObservationDate	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
0	1	01/22/2020	Anhui	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
1	2	01/22/2020	Beijing	Mainland China	1/22/2020 17:00	14.0	0.0	0.0
2	3	01/22/2020	Chongqing	Mainland China	1/22/2020 17:00	6.0	0.0	0.0
3	4	01/22/2020	Fujian	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
4	5	01/22/2020	Gansu	Mainland China	1/22/2020 17:00	0.0	0.0	0.0
5	6	01/22/2020	Guangdong	Mainland China	1/22/2020 17:00	26.0	0.0	0.0
6	7	01/22/2020	Guangxi	Mainland China	1/22/2020 17:00	2.0	0.0	0.0
7	8	01/22/2020	Guizhou	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
8	9	01/22/2020	Hainan	Mainland China	1/22/2020 17:00	4.0	0.0	0.0
9	10	01/22/2020	Hebei	Mainland China	1/22/2020 17:00	1.0	0.0	0.0

In [22]:

```
df2= df.groupby("Country/Region")[["Confirmed","Deaths","Recovered"]].sum().reset_index(
```

In [23]:

```
df2
```

Out[23]:

	Country/Region	Confirmed	Deaths	Recovered
0	Azerbaijan	1.0	0.0	0.0
1	Afghanistan	17.0	0.0	0.0
2	Algeria	91.0	0.0	0.0
3	Andorra	7.0	0.0	0.0
4	Argentina	25.0	1.0	0.0
...	...	...	...	...
106	US	2660.0	90.0	150.0
107	Ukraine	6.0	0.0	0.0
108	United Arab Emirates	524.0	0.0	108.0
109	Vatican City	3.0	0.0	0.0
110	Vietnam	560.0	0.0	342.0

111 rows × 4 columns

In [28]:

```
df2= df.groupby(["Country/Region","ObservationDate"])[["Confirmed","Deaths","Recovered"]
```

In [29]:

```
df2
```

Out[29]:

	Country/Region	ObservationDate	Confirmed	Deaths	Recovered
0	Azerbaijan	02/28/2020	1.0	0.0	0.0
1	Afghanistan	02/24/2020	1.0	0.0	0.0
2	Afghanistan	02/25/2020	1.0	0.0	0.0
3	Afghanistan	02/26/2020	1.0	0.0	0.0
4	Afghanistan	02/27/2020	1.0	0.0	0.0
...	...	...	...	...	...
1856	Vietnam	03/04/2020	16.0	0.0	16.0
1857	Vietnam	03/05/2020	16.0	0.0	16.0
1858	Vietnam	03/06/2020	16.0	0.0	16.0
1859	Vietnam	03/07/2020	18.0	0.0	16.0
1860	Vietnam	03/08/2020	30.0	0.0	16.0

1861 rows × 5 columns

In [30]:

```
df3 = df2[df2["Confirmed"]>100]
```

In [31]:

```
df3
```

Out[31]:

	Country/Region	ObservationDate	Confirmed	Deaths	Recovered
106	Austria	03/08/2020	104.0	0.0	0.0
171	Belgium	03/06/2020	109.0	0.0	1.0
172	Belgium	03/07/2020	169.0	0.0	1.0
173	Belgium	03/08/2020	200.0	0.0	1.0
461	France	03/01/2020	130.0	2.0	12.0
...	...	...	...	...	...
1761	US	03/04/2020	153.0	11.0	8.0
1762	US	03/05/2020	221.0	12.0	8.0
1763	US	03/06/2020	278.0	14.0	8.0
1764	US	03/07/2020	417.0	17.0	8.0
1765	US	03/08/2020	537.0	21.0	8.0

202 rows × 5 columns

In [32]:

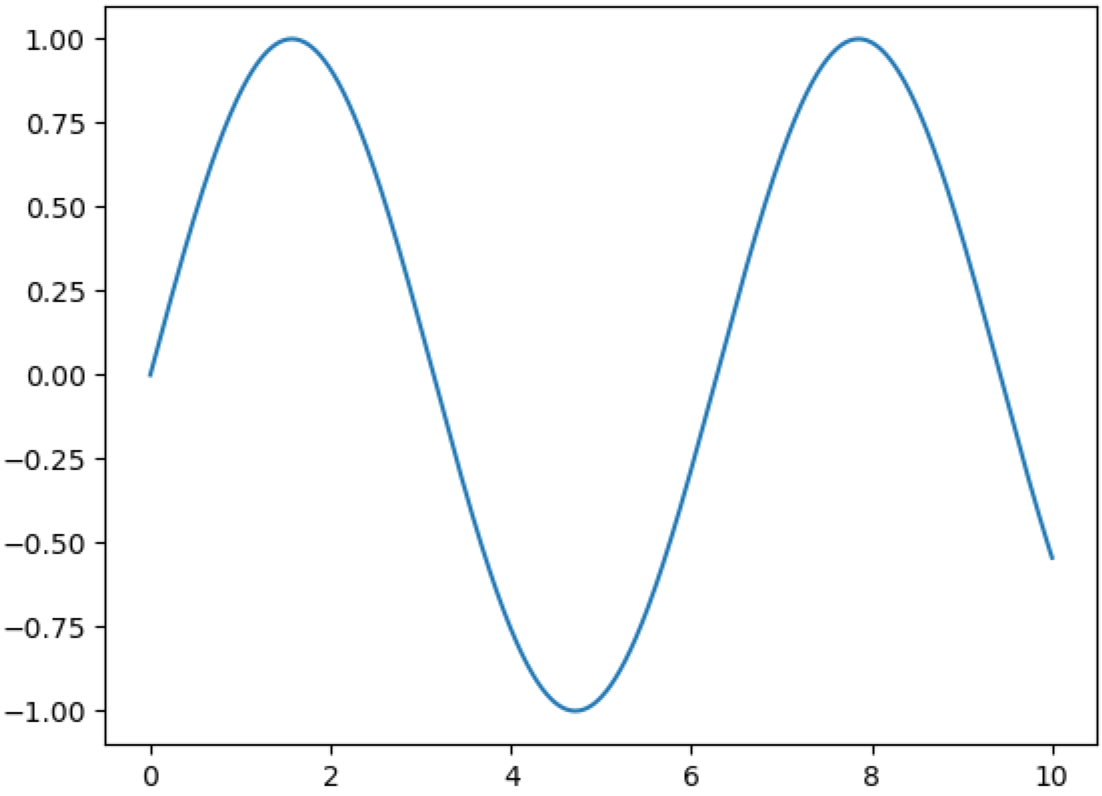
```
import matplotlib.pyplot as plt
```

In [34]:

```
x = np.linspace(0,10,1000)
y = np.sin(x)
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x27f283fa650>]
```

Out[34]:

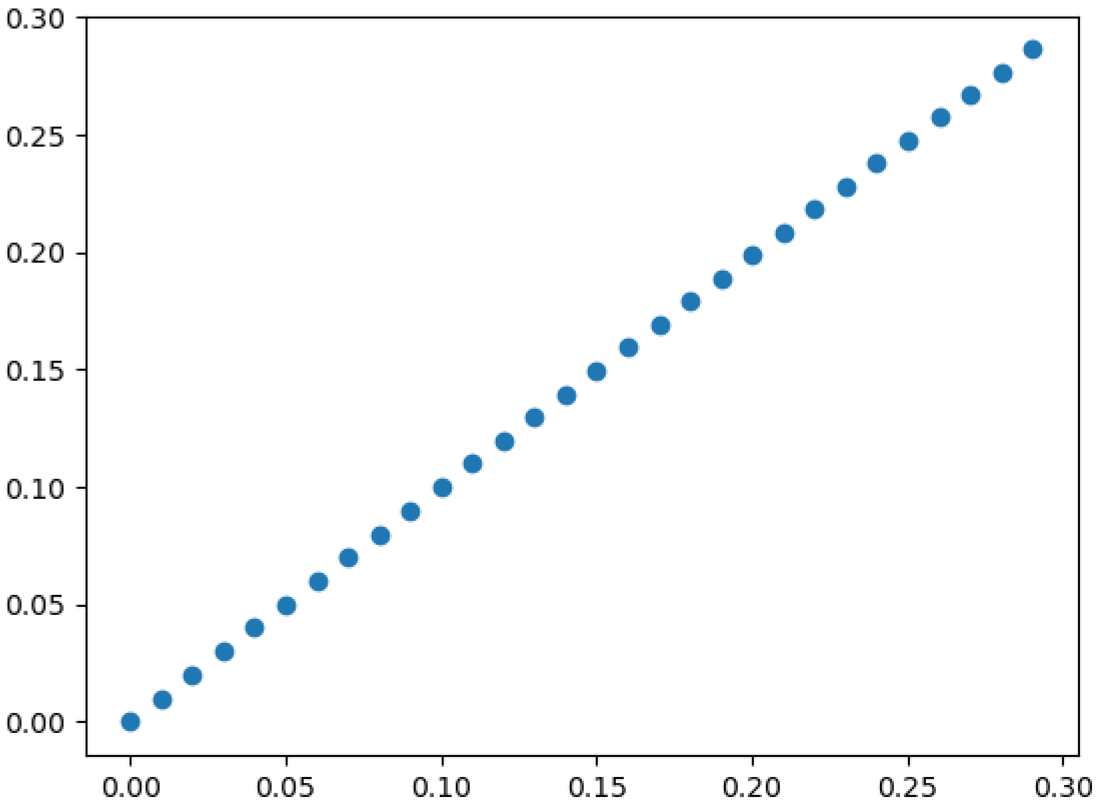


In [36]:

```
plt.scatter(x[:30],y[:30])
```

Out[36]:

```
<matplotlib.collections.PathCollection at 0x27f2868b250>
```

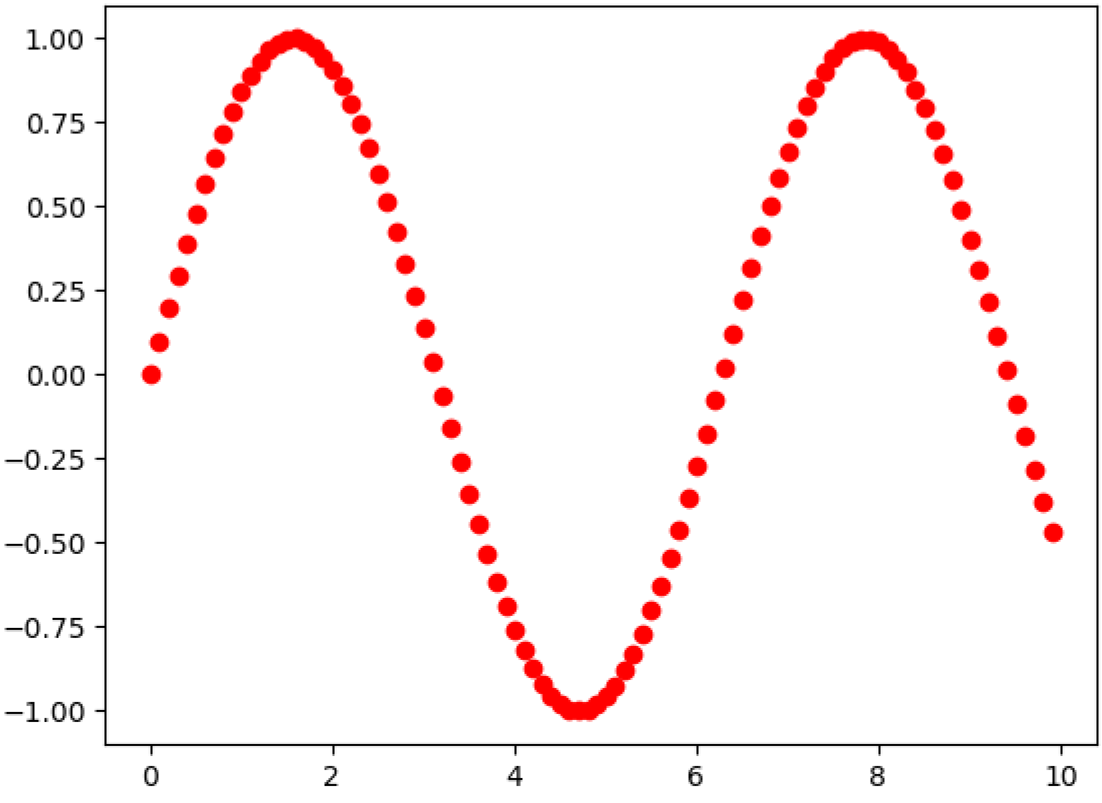


In [39]:

```
plt.scatter(x[::10],y[::10],color = "red")
```

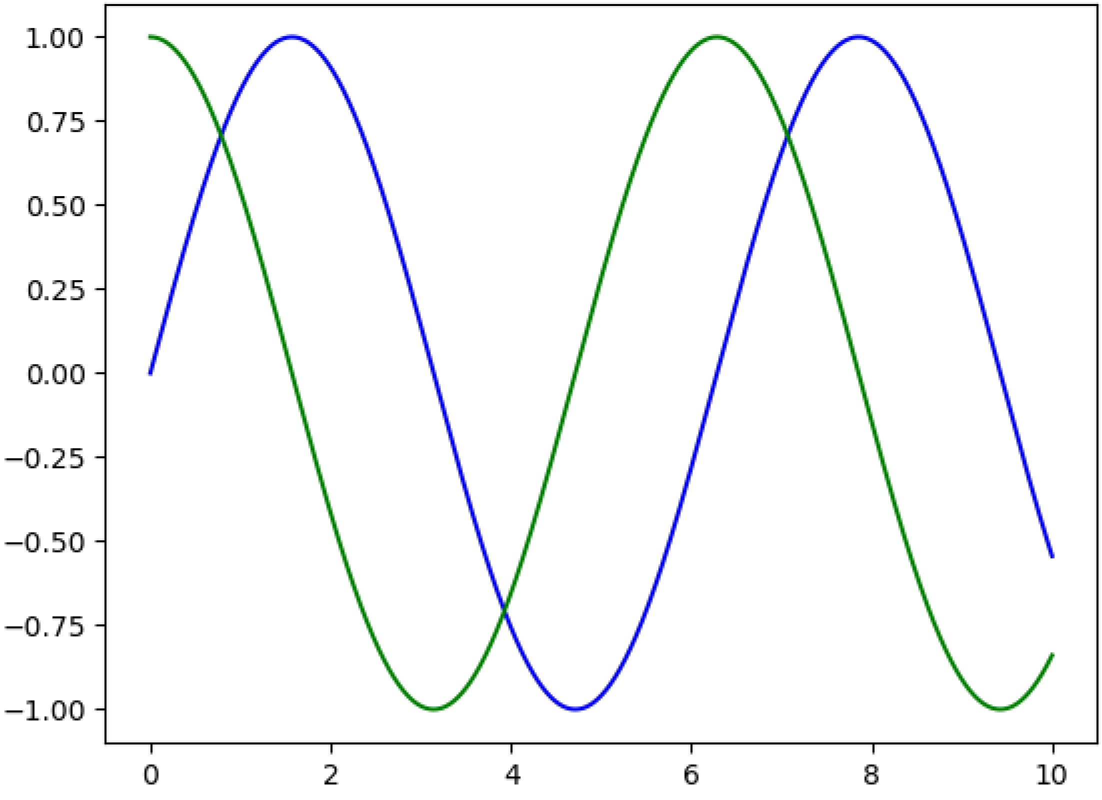
Out[39]:

```
<matplotlib.collections.PathCollection at 0x27f2860a450>
```



```
In [40]: plt.plot(x,y,color="b")
         plt.plot(x,np.cos(x),color="g")
```

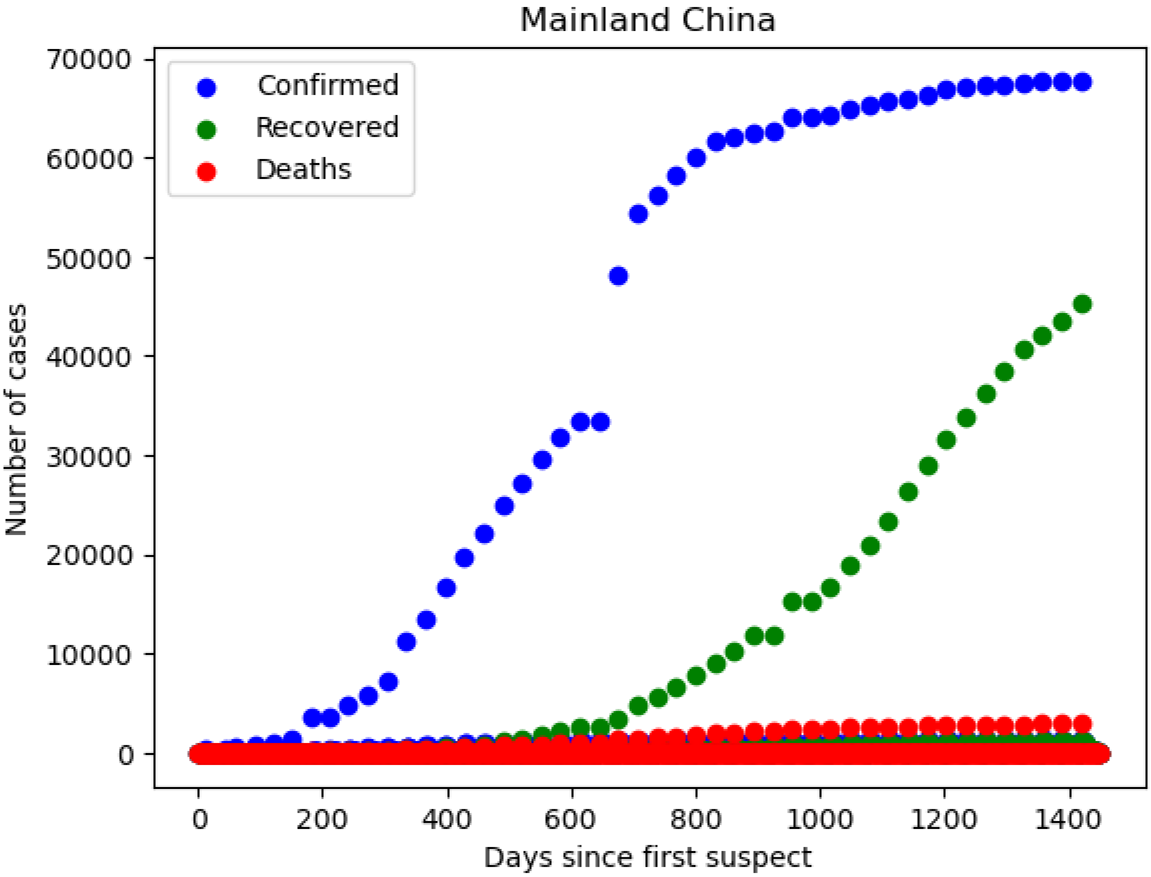
```
Out[40]: [<matplotlib.lines.Line2D at 0x27f286fd590>]
```

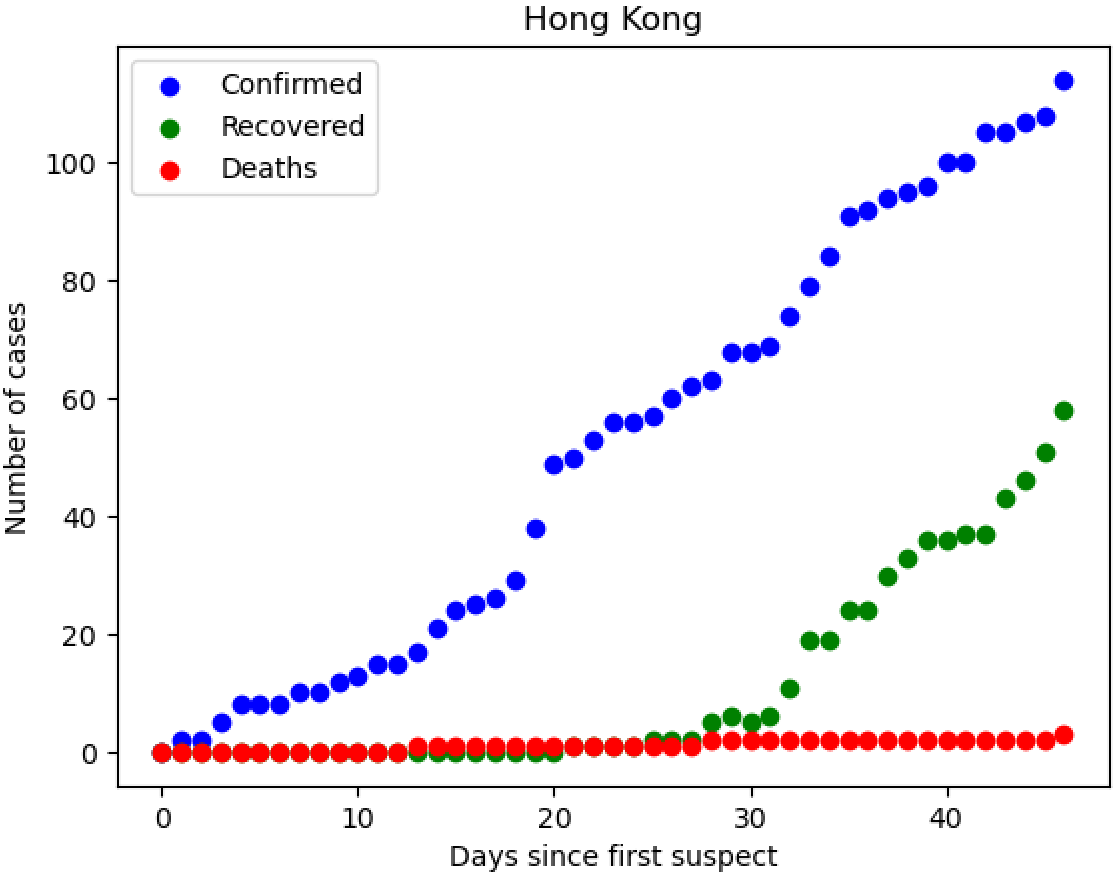


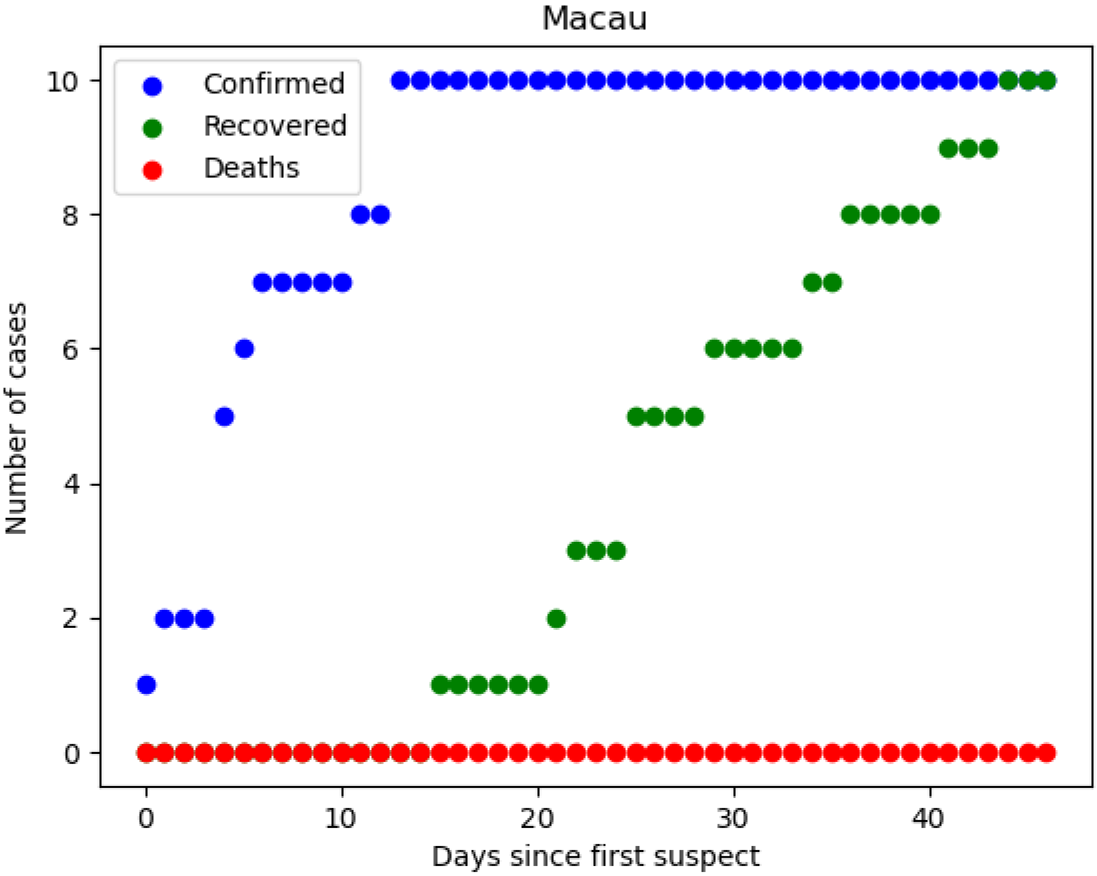
```
In [42]: countries = df["Country/Region"].unique()
         len(countries)
```

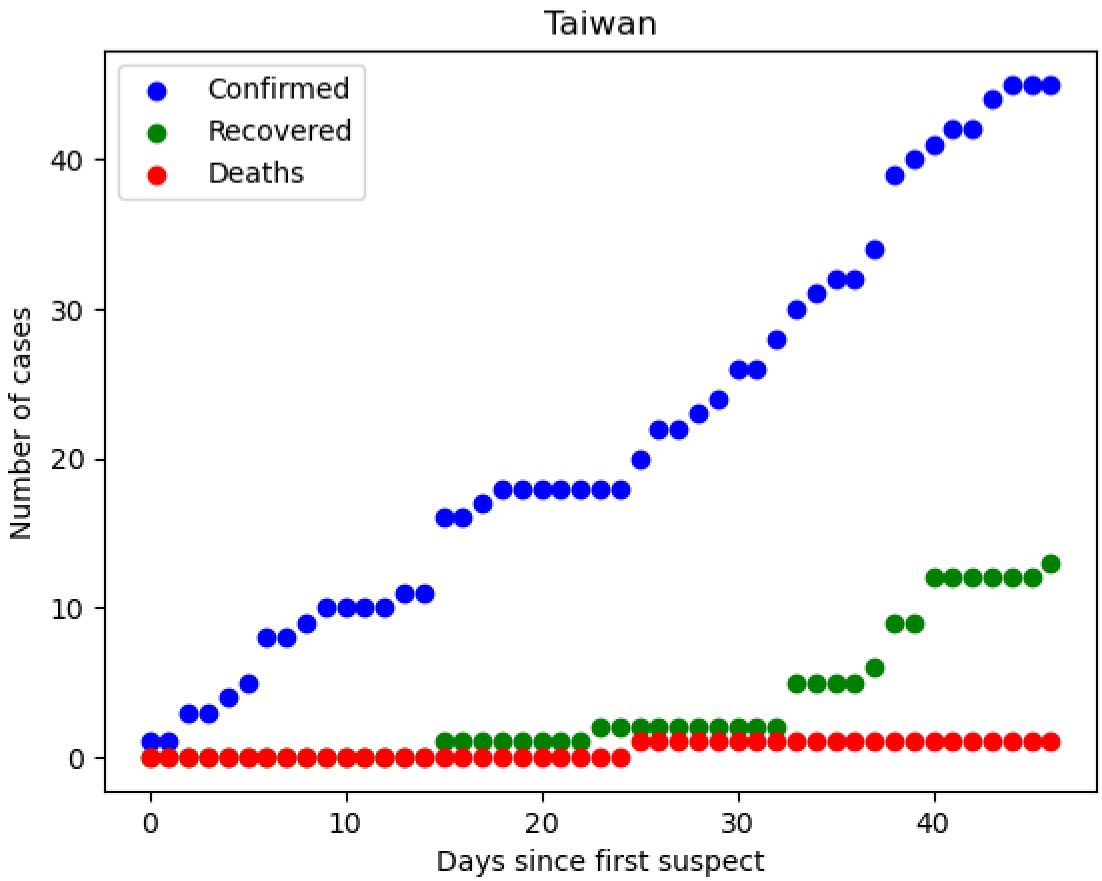
```
Out[42]: 111
```

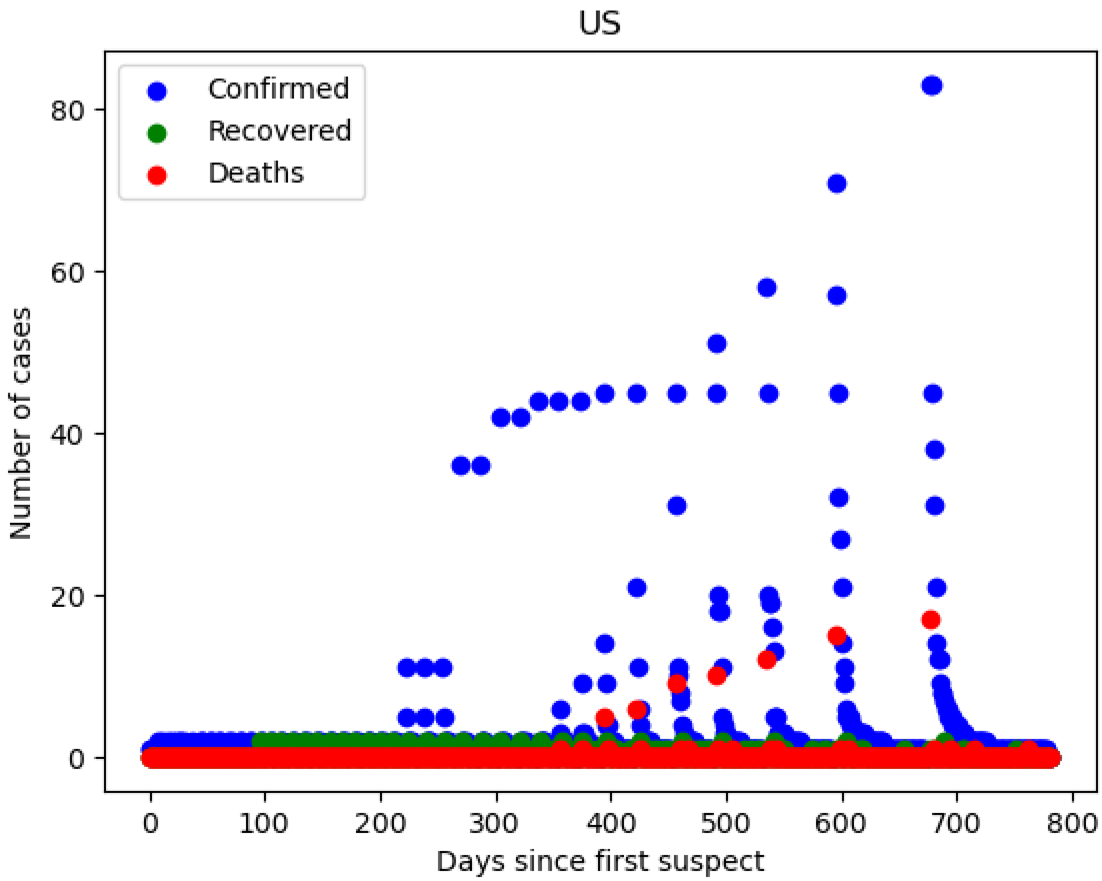
```
In [46]: for idx in range(0,len(countries)):
    C= df[df["Country/Region"]== countries[idx]].reset_index()
    plt.scatter(np.arange(0,len(C)),C["Confirmed"],color = "blue",label = "Confirmed")
    plt.scatter(np.arange(0,len(C)),C["Recovered"],color = "green",label = "Recovered")
    plt.scatter(np.arange(0,len(C)),C["Deaths"],color = "red",label = "Deaths")
    plt.title(countries[idx])
    plt.xlabel("Days since first suspect")
    plt.ylabel("Number of cases")
    plt.legend()
    plt.show()
```

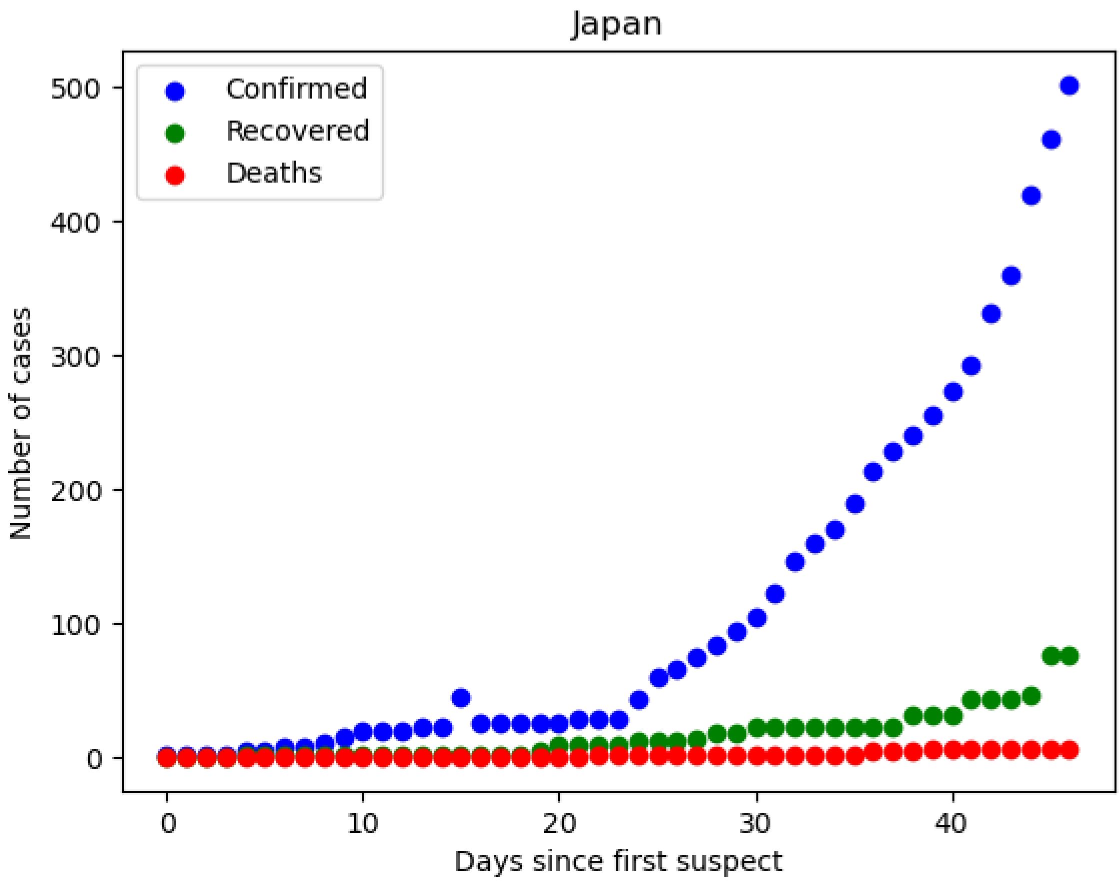


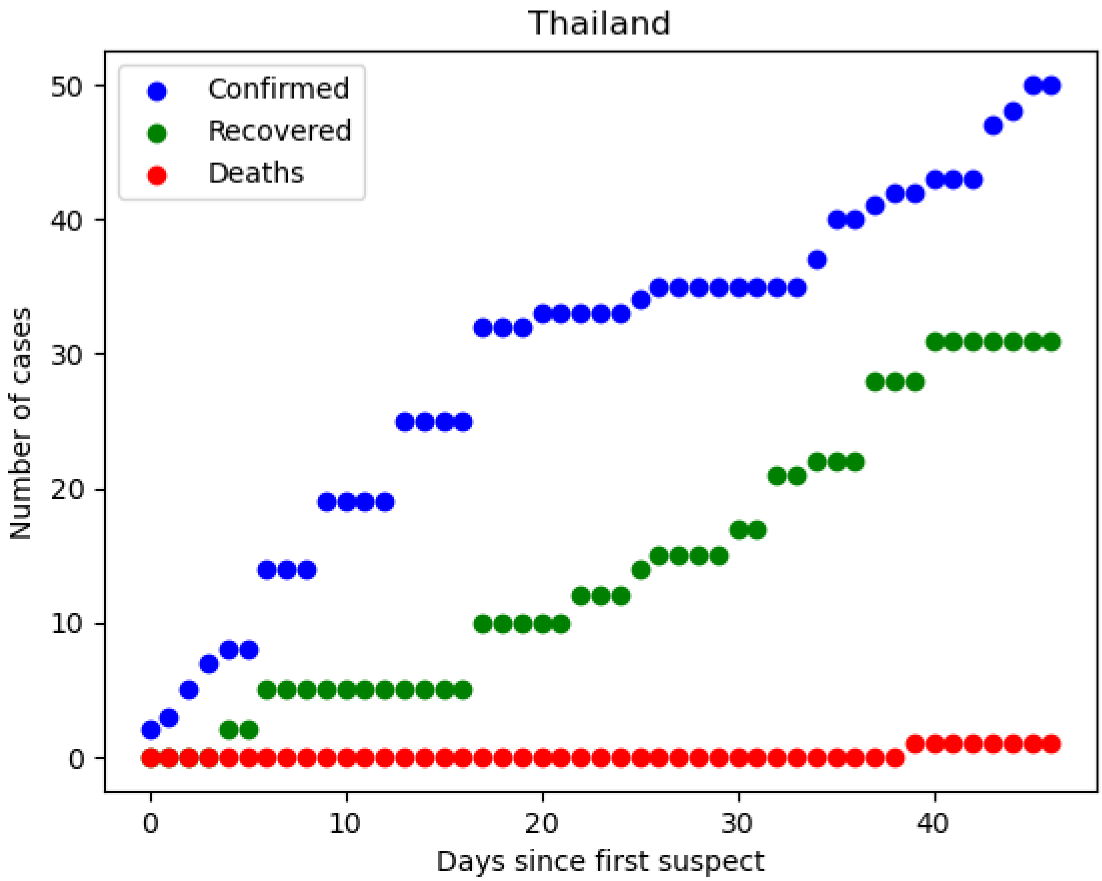

Hong Kong

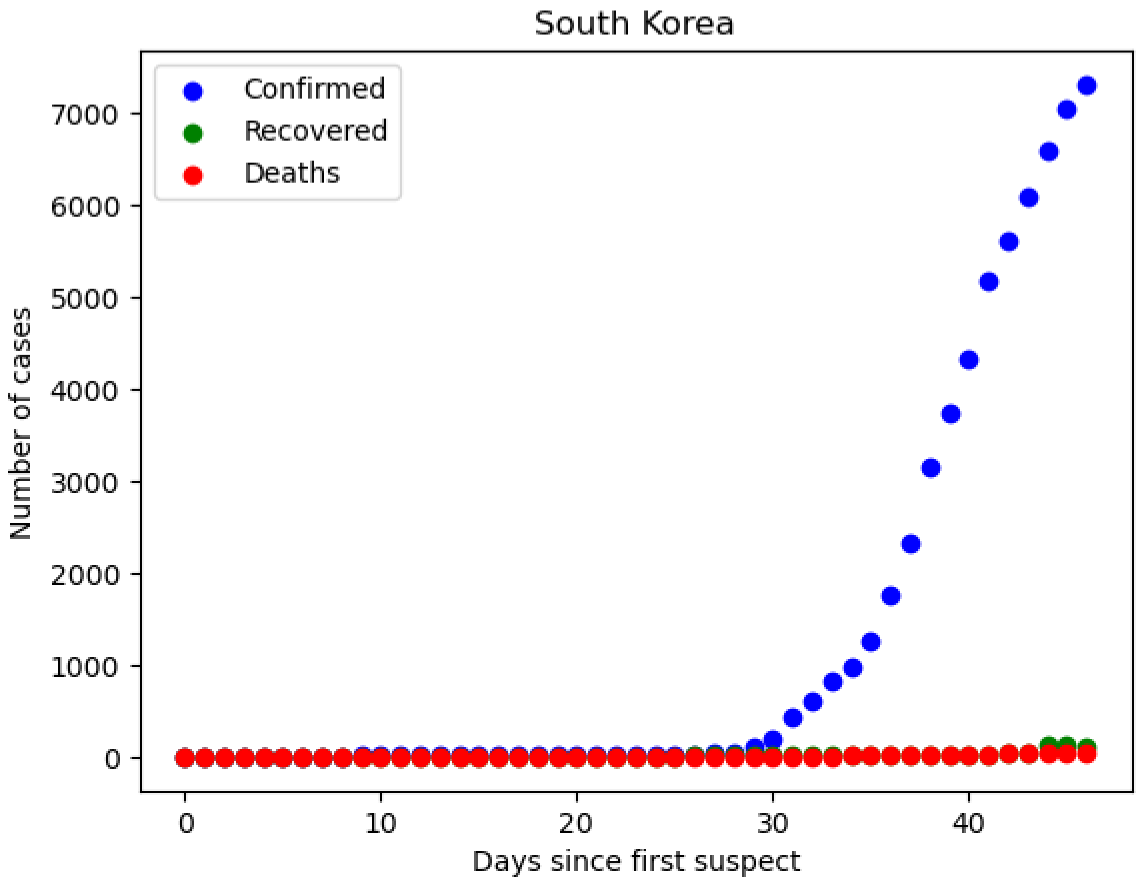

Macau

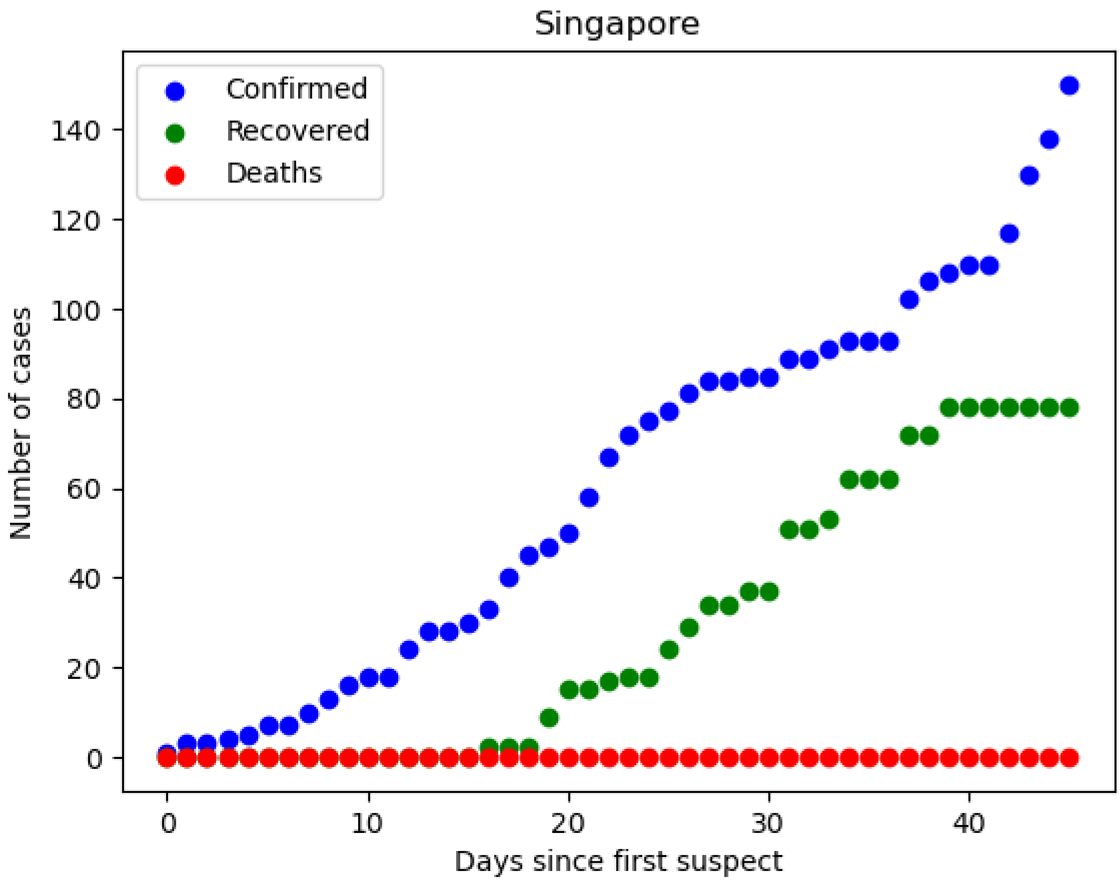

Taiwan
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
Days since first suspect

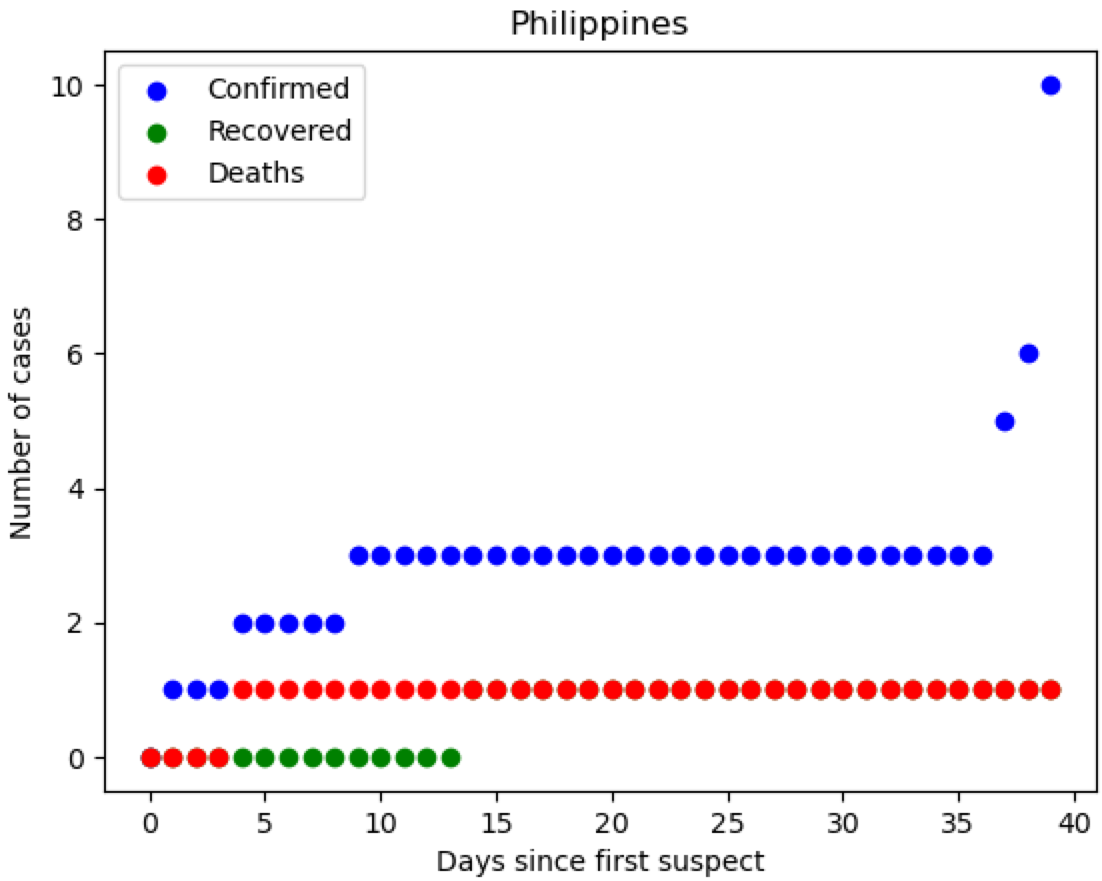

US
Confirmed
Recovered
Deaths
Number of cases
0
20
40
60
80
0
100
200
300
400
500
600
700
800
Days since first suspect

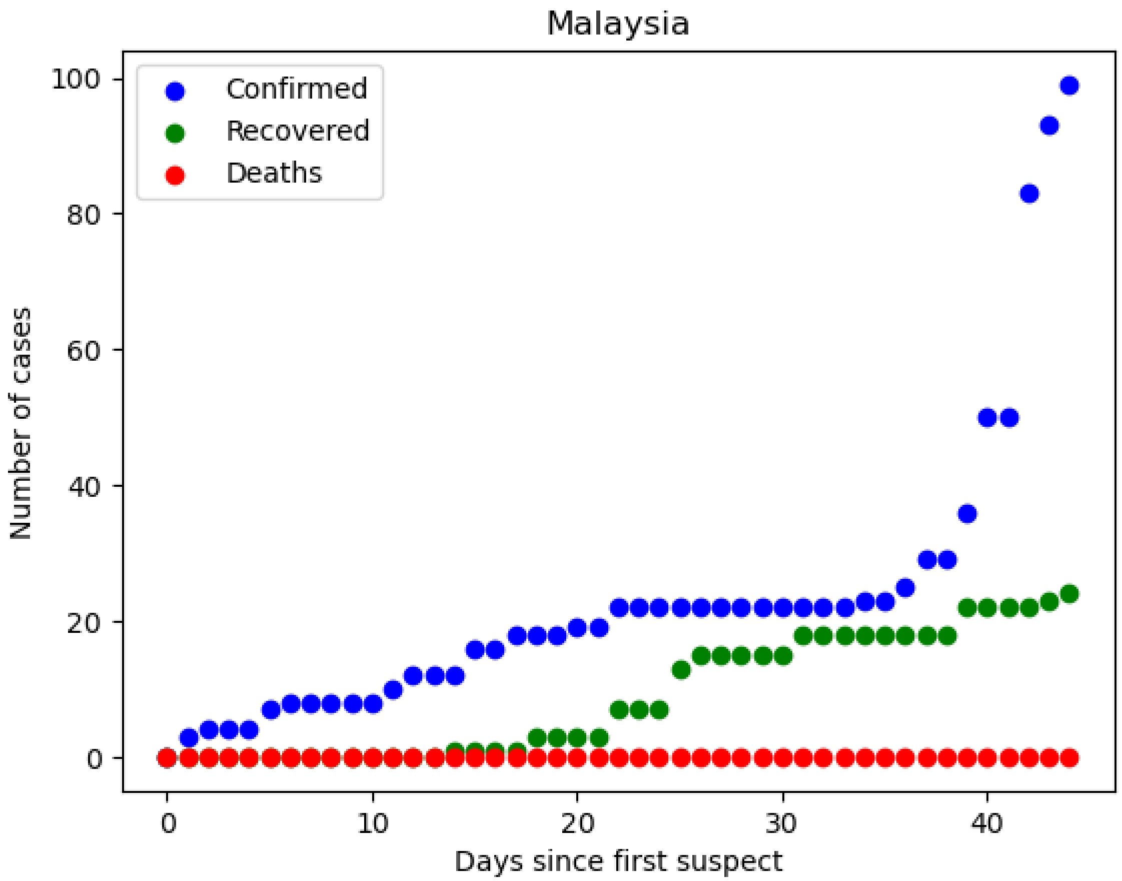

Japan
Confirmed
Recovered
Deaths
Number of cases
500
400
300
200
100
0
0
10
20
30
40
Days since first suspect

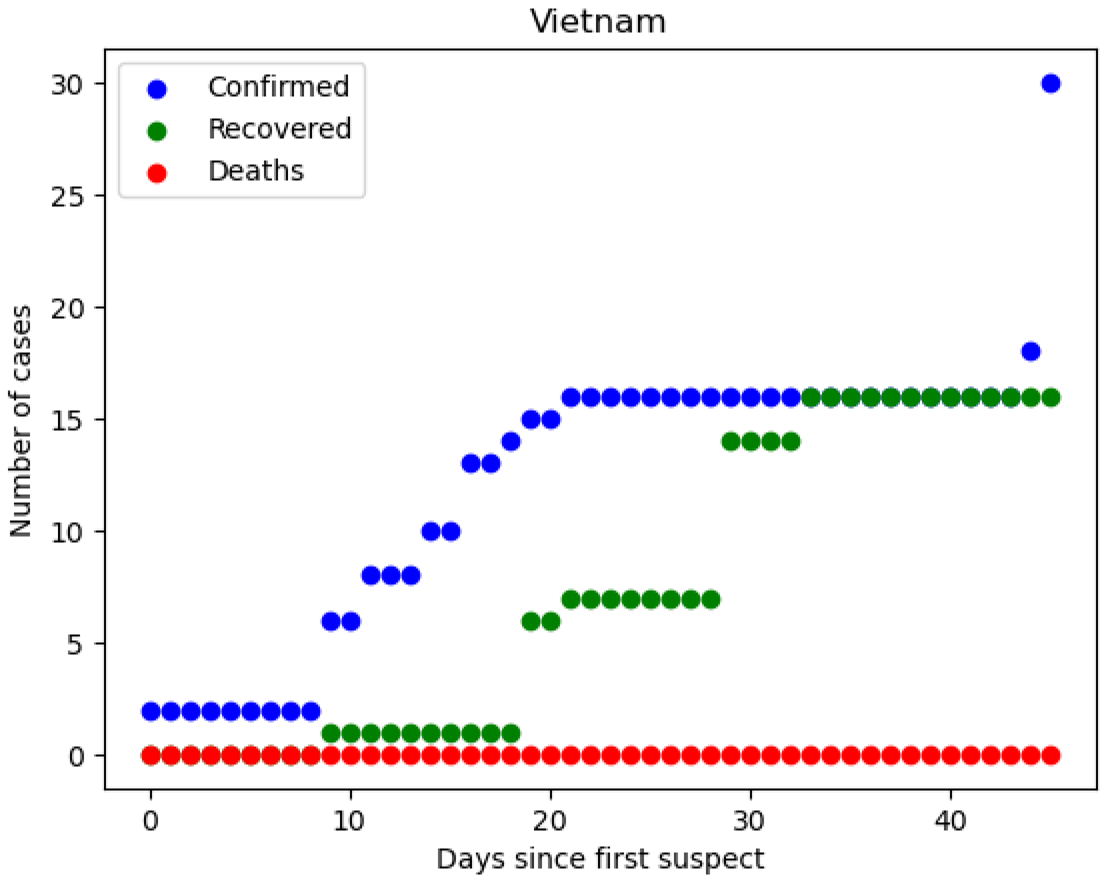

Thailand
Confirmed
Recovered
Deaths
Number of cases
50
40
30
20
10
0
0
10
20
30
40
Days since first suspect

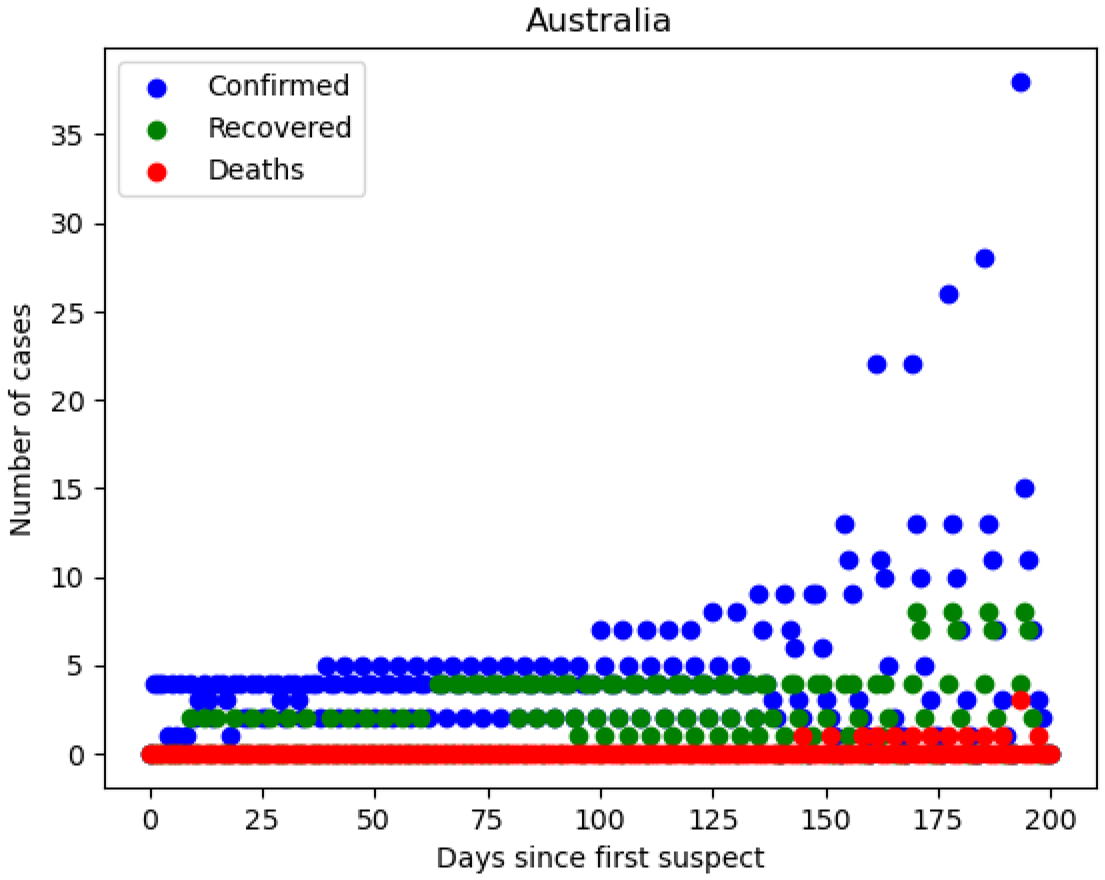

South Korea
Confirmed
Recovered
Deaths
Number of cases
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
Days since first suspect

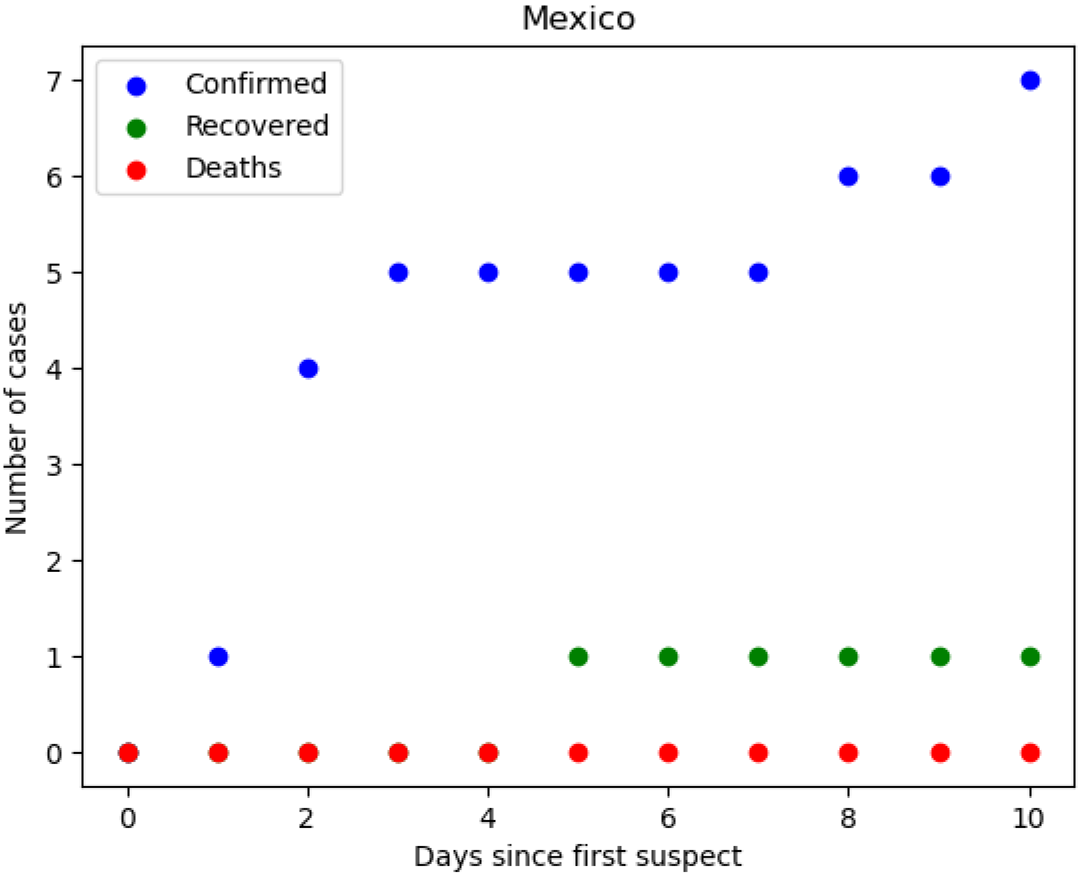

Singapore
Confirmed
Recovered
Deaths
Number of cases
140
120
100
80
60
40
20
0
0
10
20
30
40
Days since first suspect

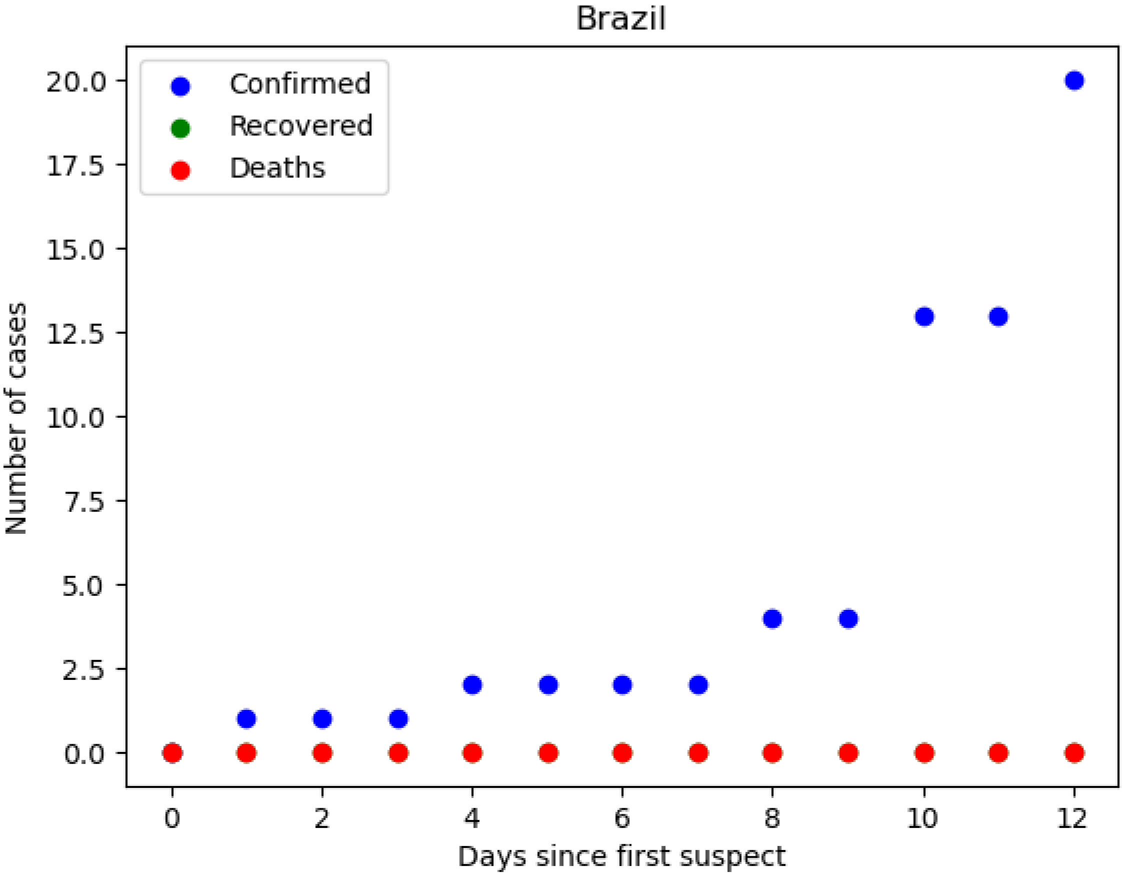

Philippines
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
Days since first suspect
0
5
10
15
20
25
30
35
40

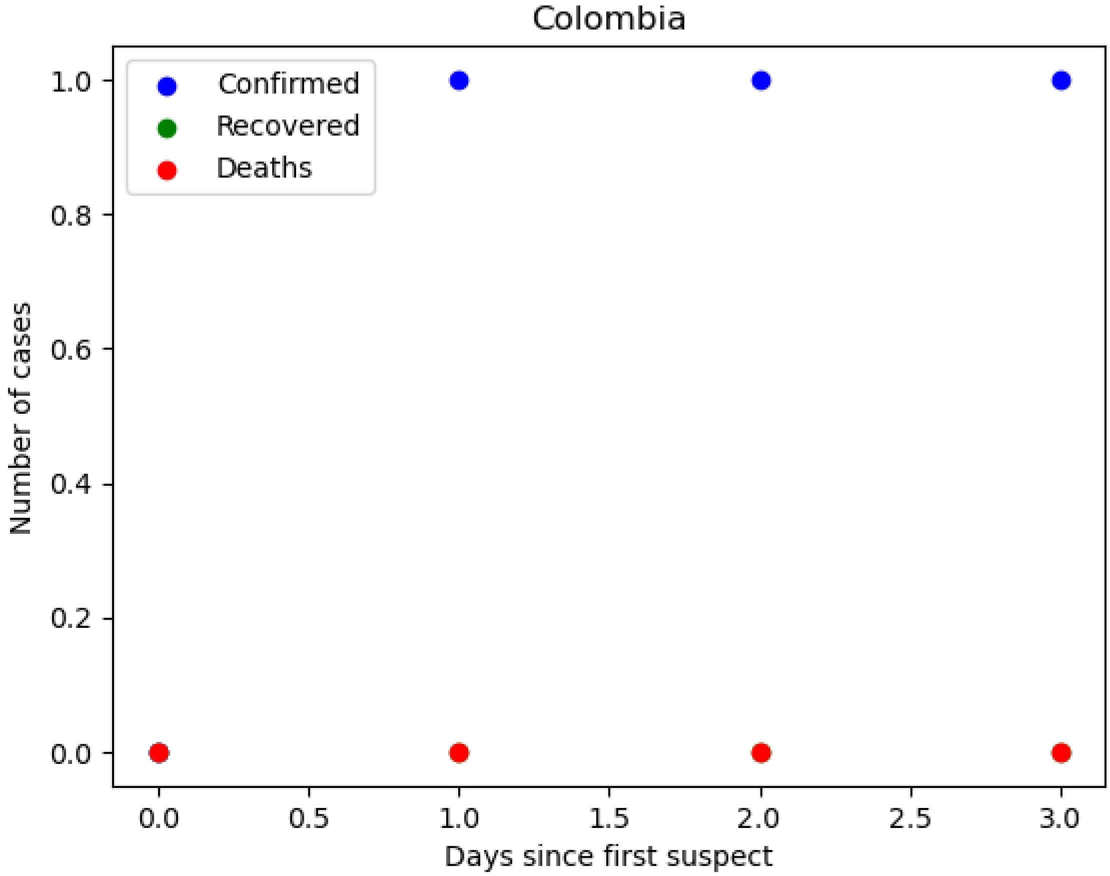

Malaysia
Confirmed
Recovered
Deaths
Number of cases
0
20
40
60
80
100
Days since first suspect
0
10
20
30
40

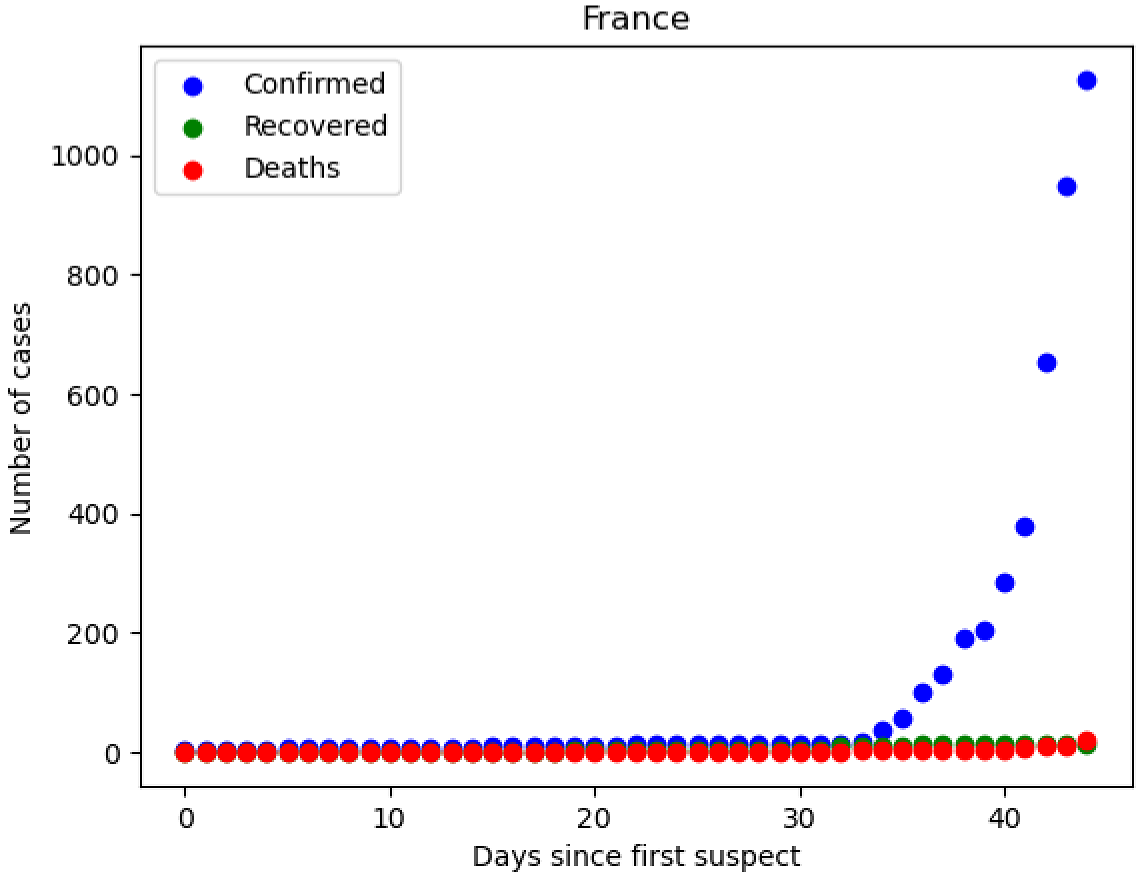

Vietnam
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
0
10
20
30
40
Days since first suspect

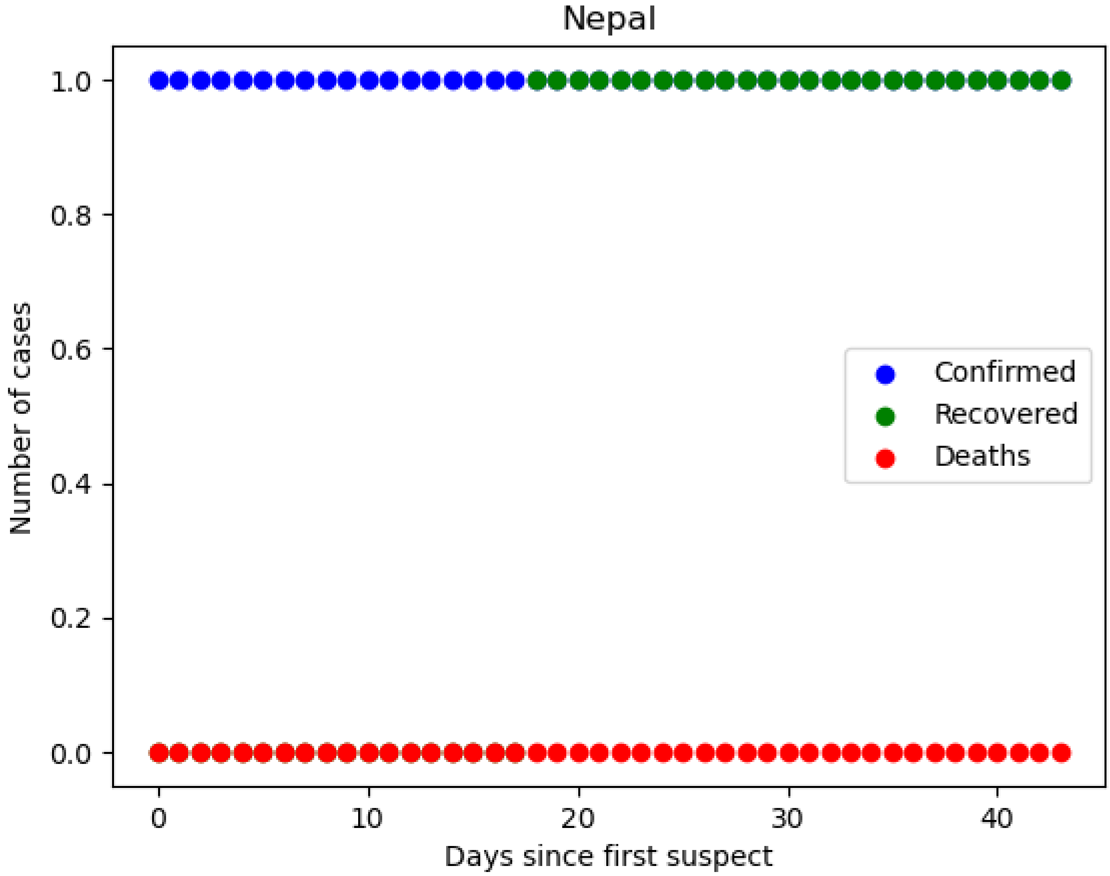

Australia
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
35
0
25
50
75
100
125
150
175
200
Days since first suspect

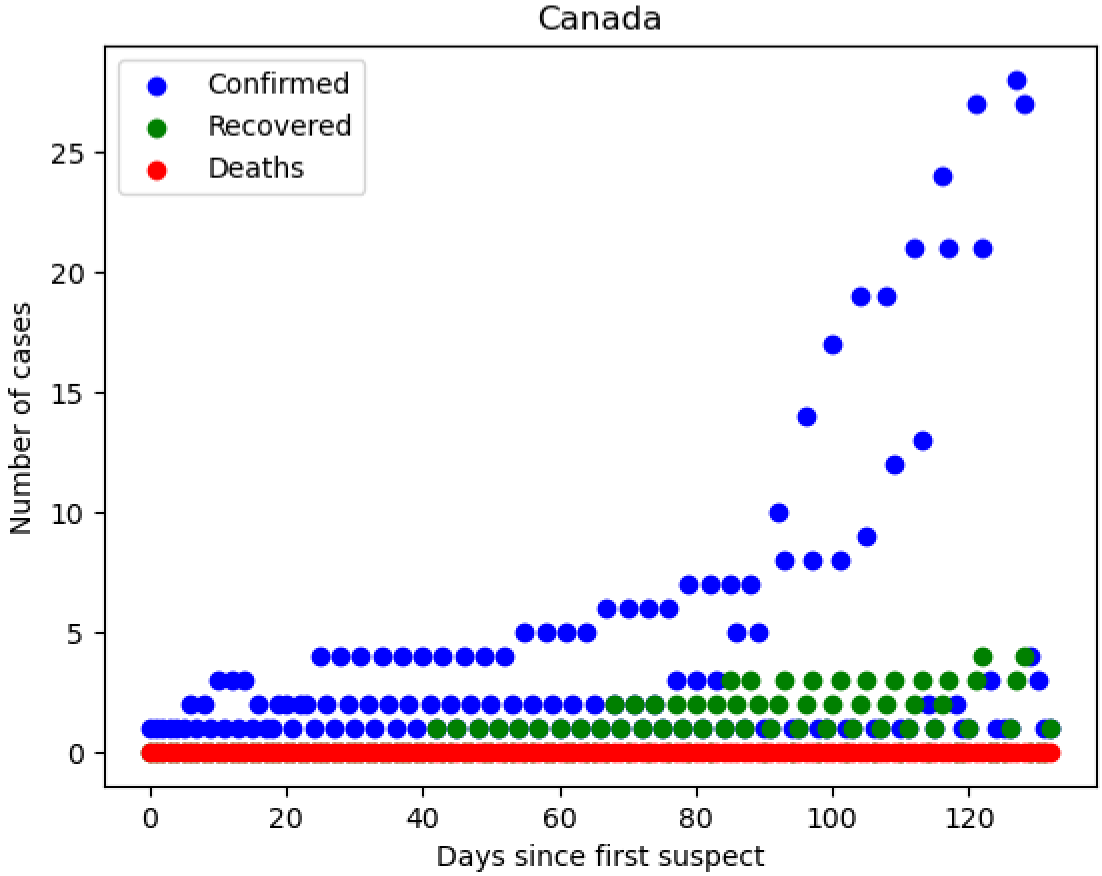

Mexico
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
6
7
0
2
4
6
8
10
Days since first suspect

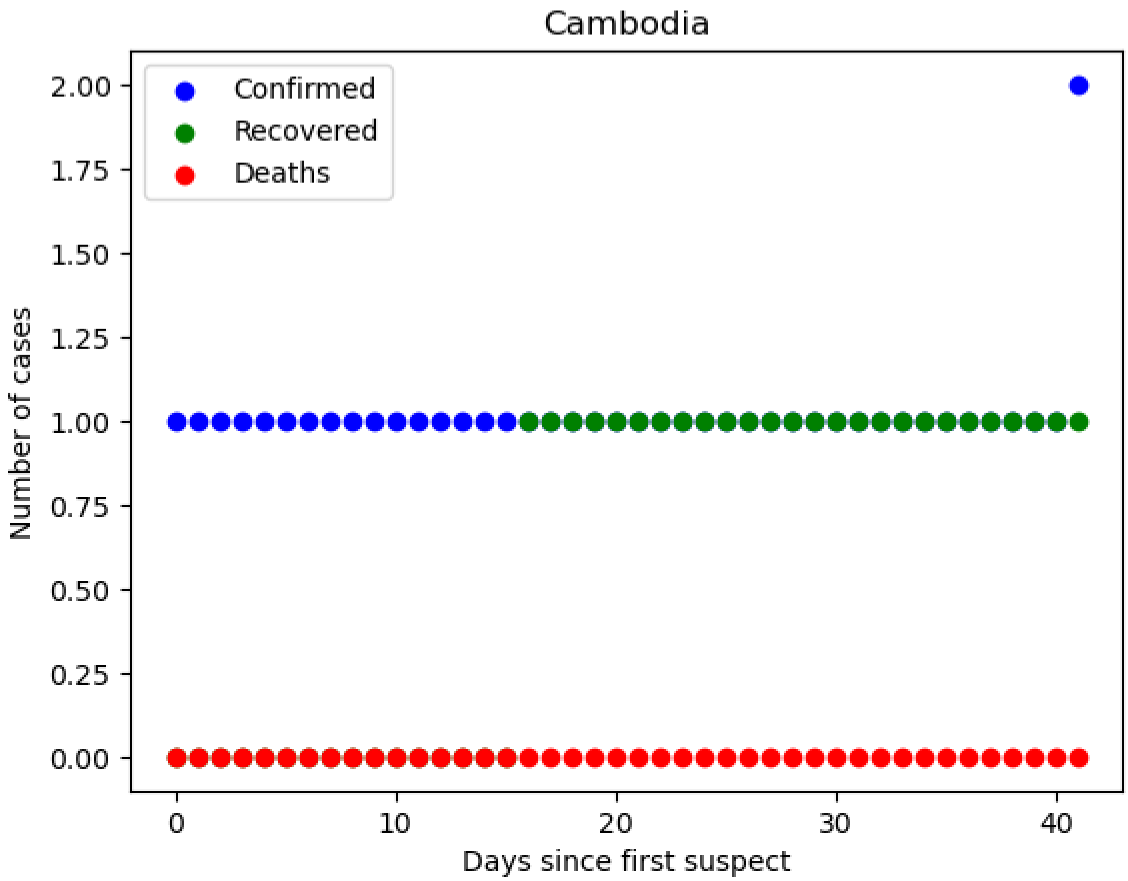

Brazil
Confirmed
Recovered
Deaths
Number of cases
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0
2
4
6
8
10
12
Days since first suspect

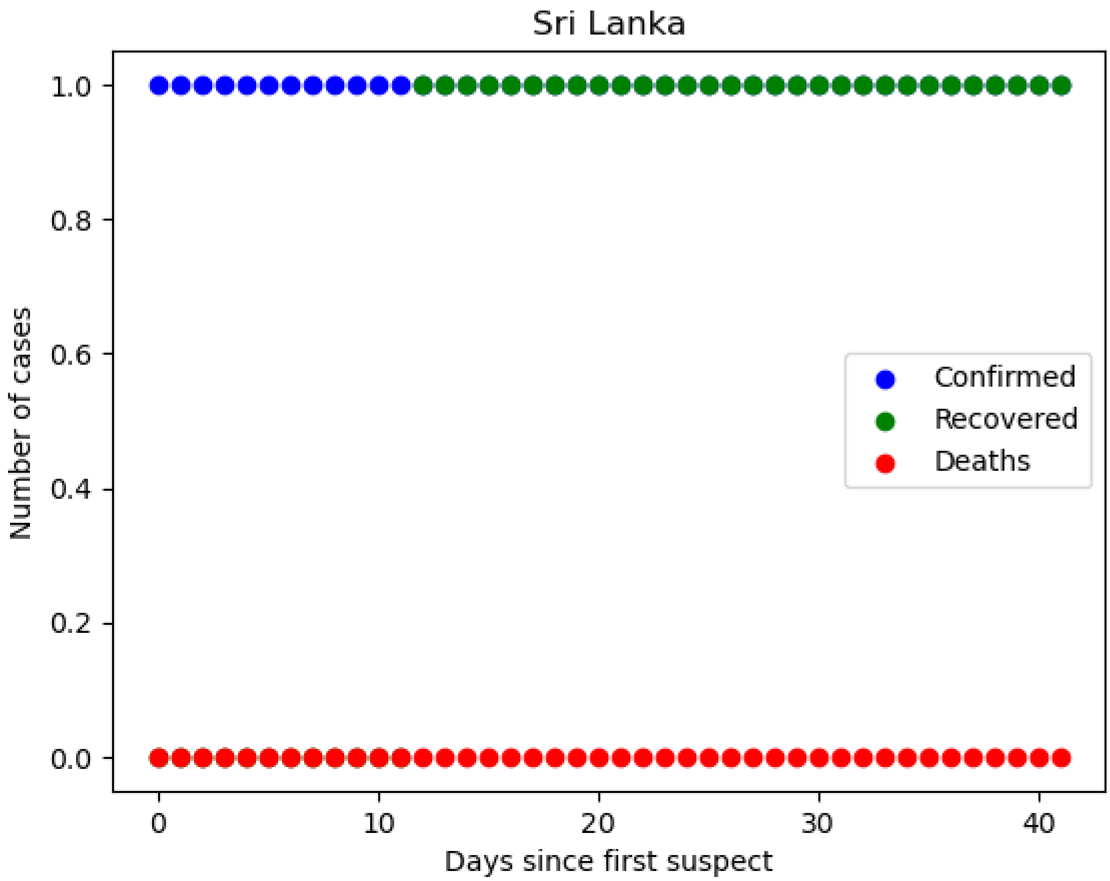

Colombia
Confirmed
Recovered
Deaths
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Days since first suspect

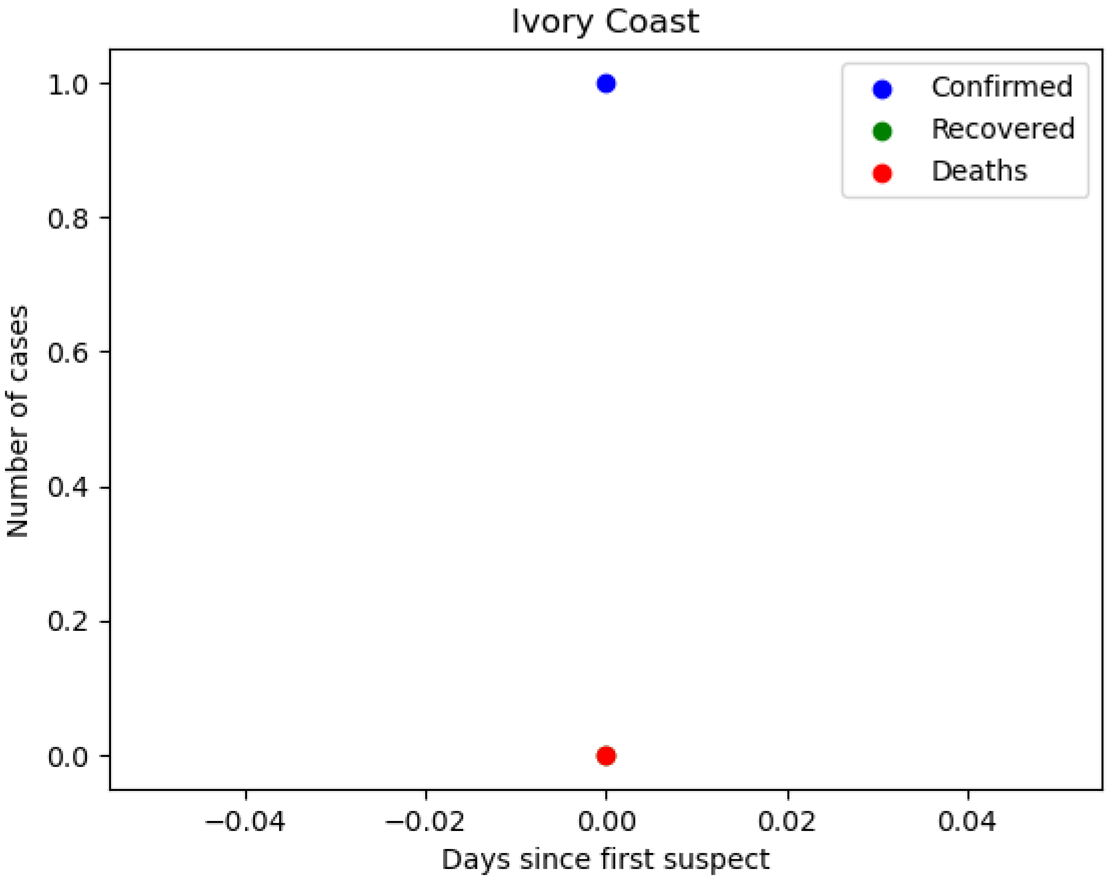

France
Confirmed
Recovered
Deaths
1000
800
600
400
200
0
Number of cases
0
10
20
30
40
Days since first suspect

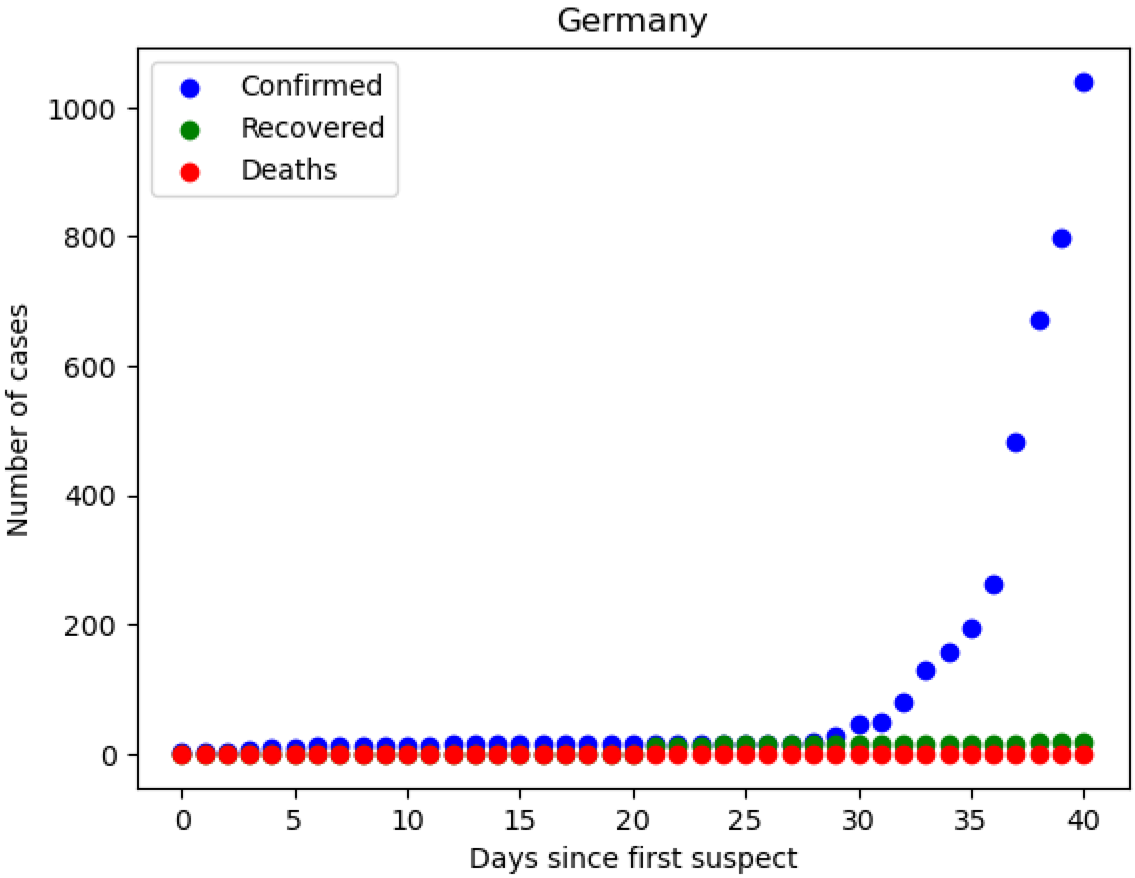

Nepal
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0
10
20
30
40
Days since first suspect
Confirmed
Recovered
Deaths

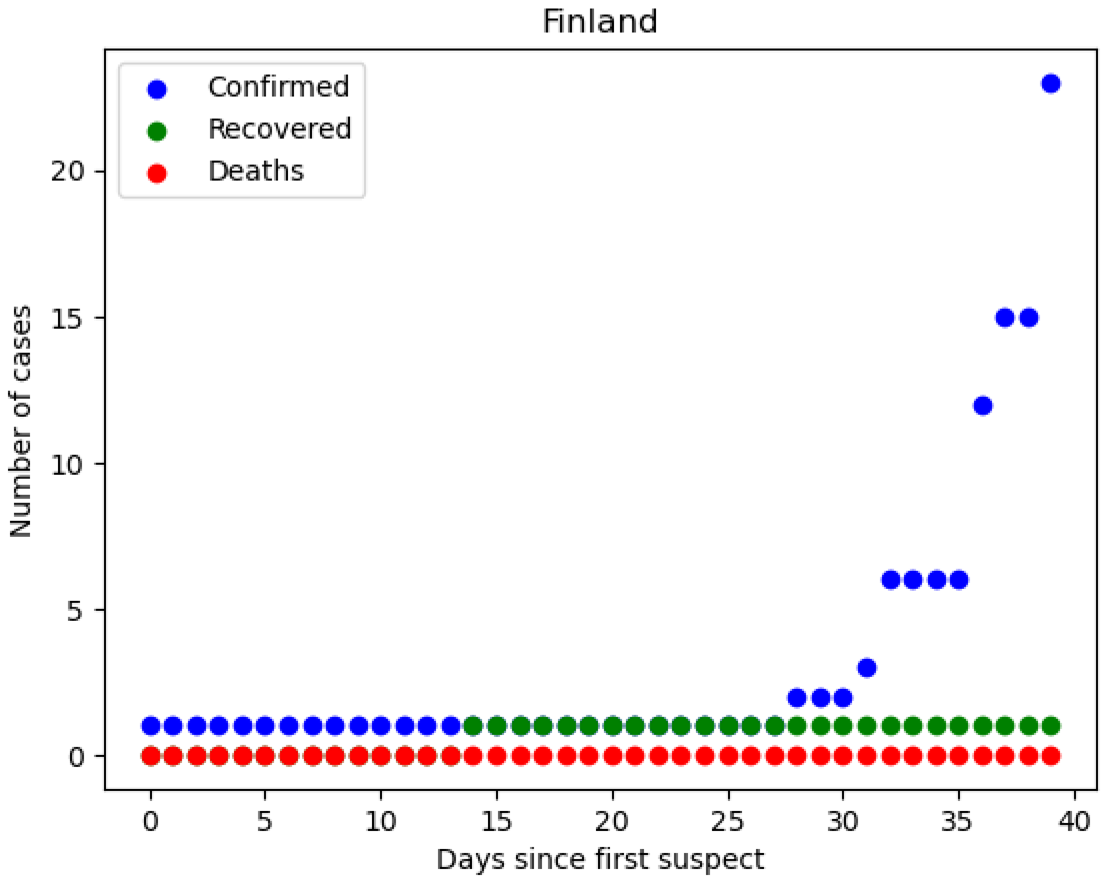

Canada
25
20
15
10
5
0
Number of cases
0
20
40
60
80
100
120
Days since first suspect
Confirmed
Recovered
Deaths

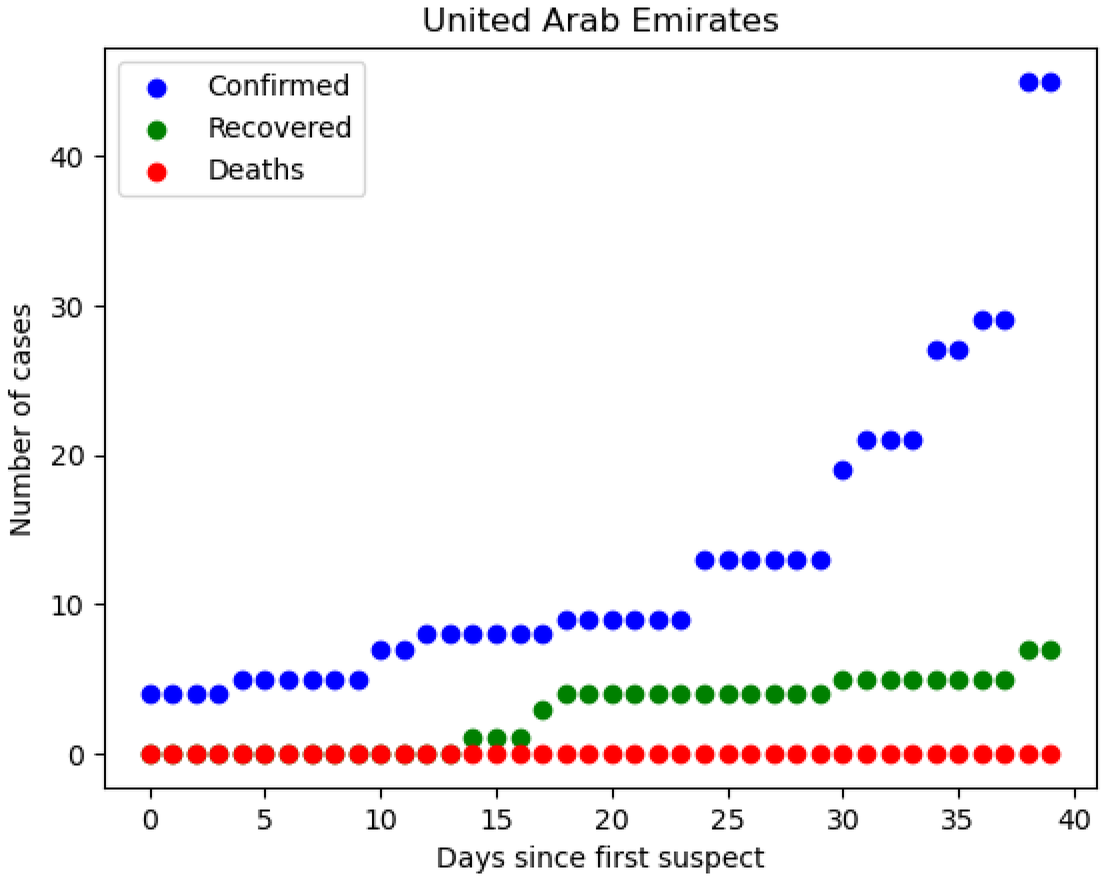

Cambodia
Confirmed
Recovered
Deaths
Number of cases
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
10
20
30
40
Days since first suspect

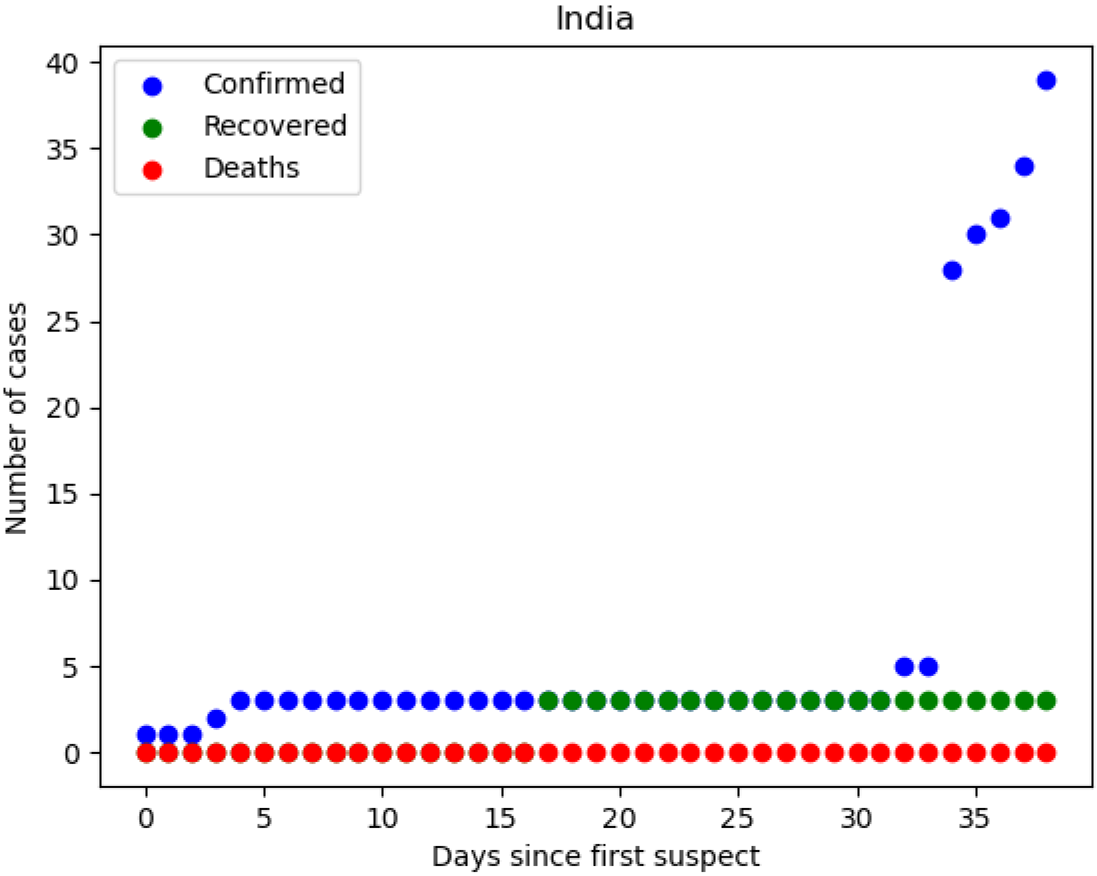

Sri Lanka
Confirmed
Recovered
Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
Days since first suspect

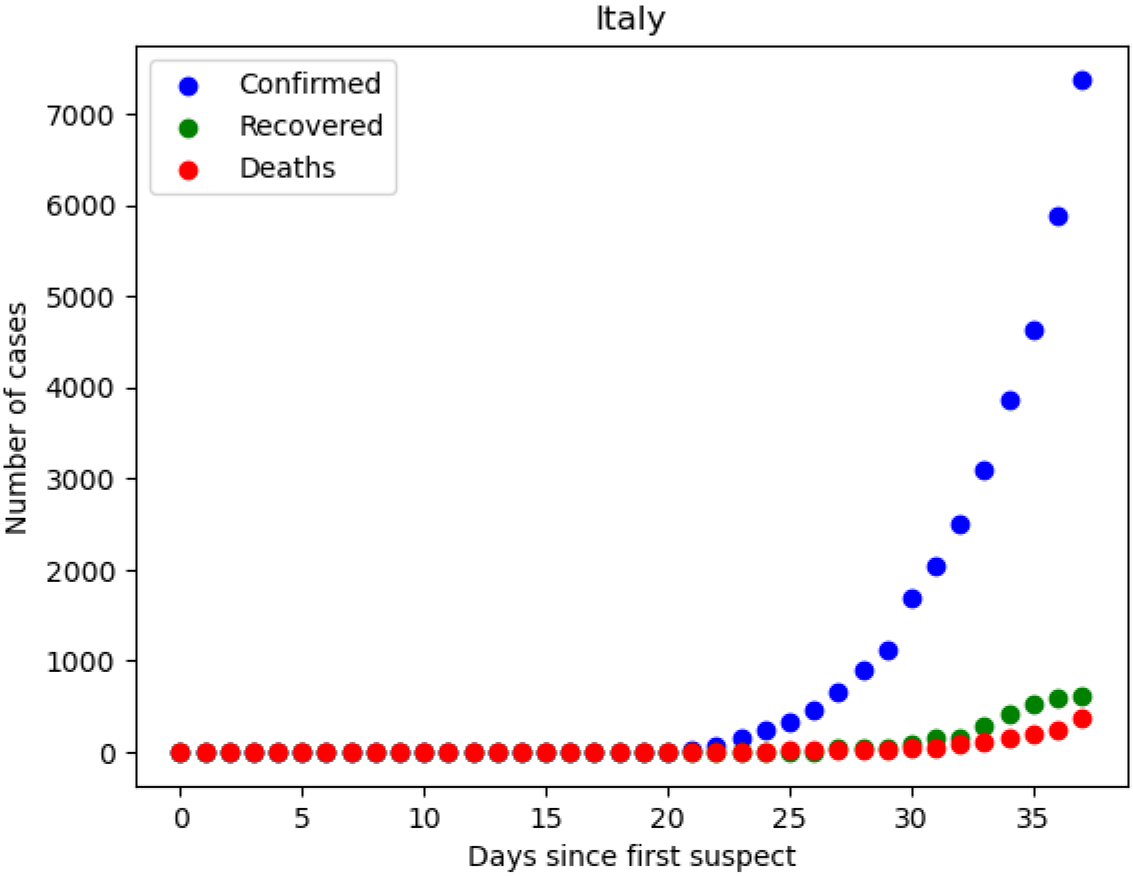

Ivory Coast
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect
Confirmed
Recovered
Deaths

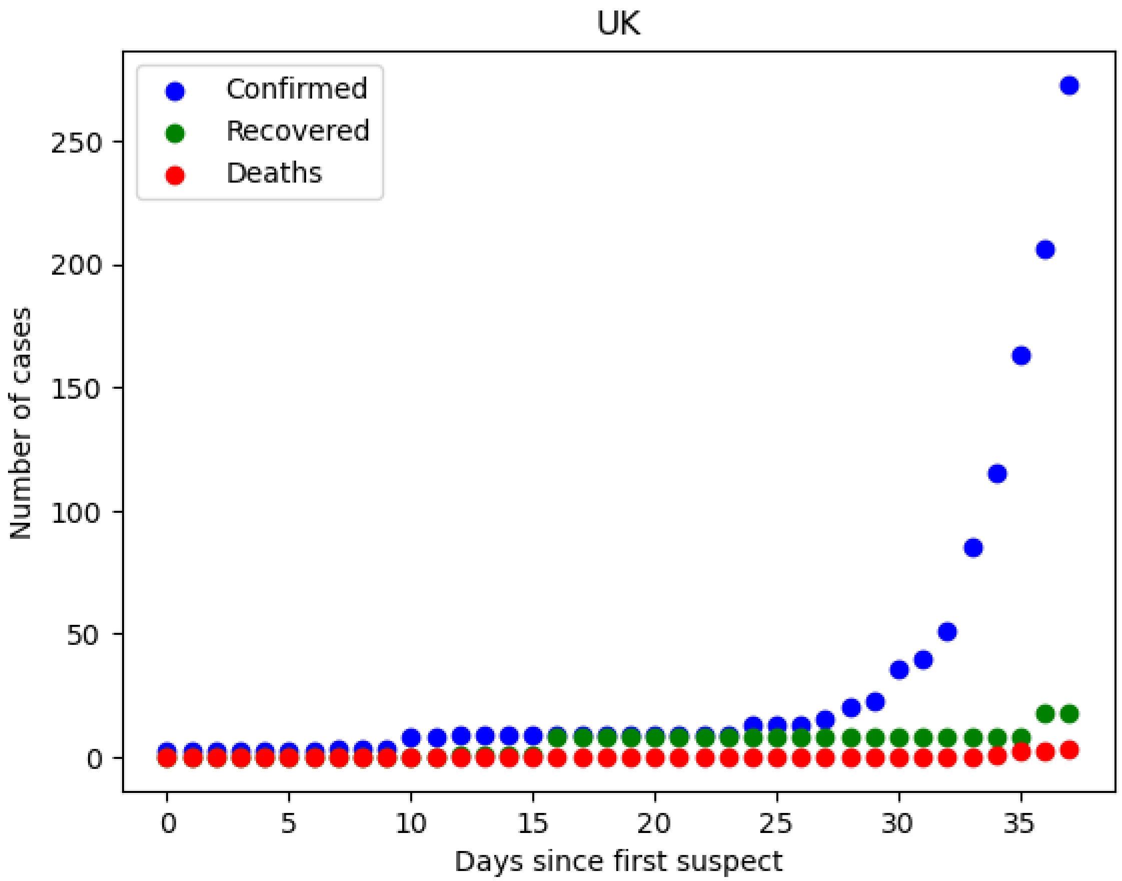

Germany
Number of cases
1000
800
600
400
200
0
0
5
10
15
20
25
30
35
40
Days since first suspect
Confirmed
Recovered
Deaths

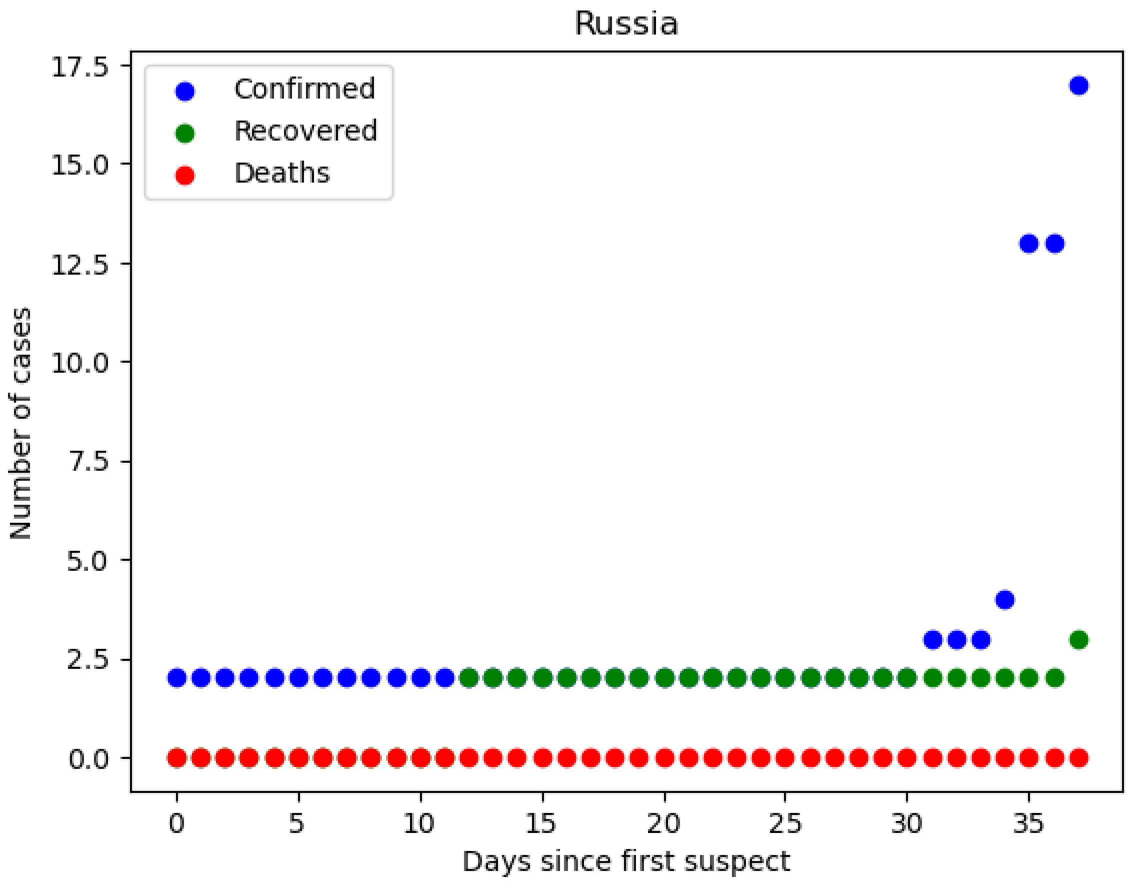

Finland
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
0
5
10
15
20
25
30
35
40
Days since first suspect

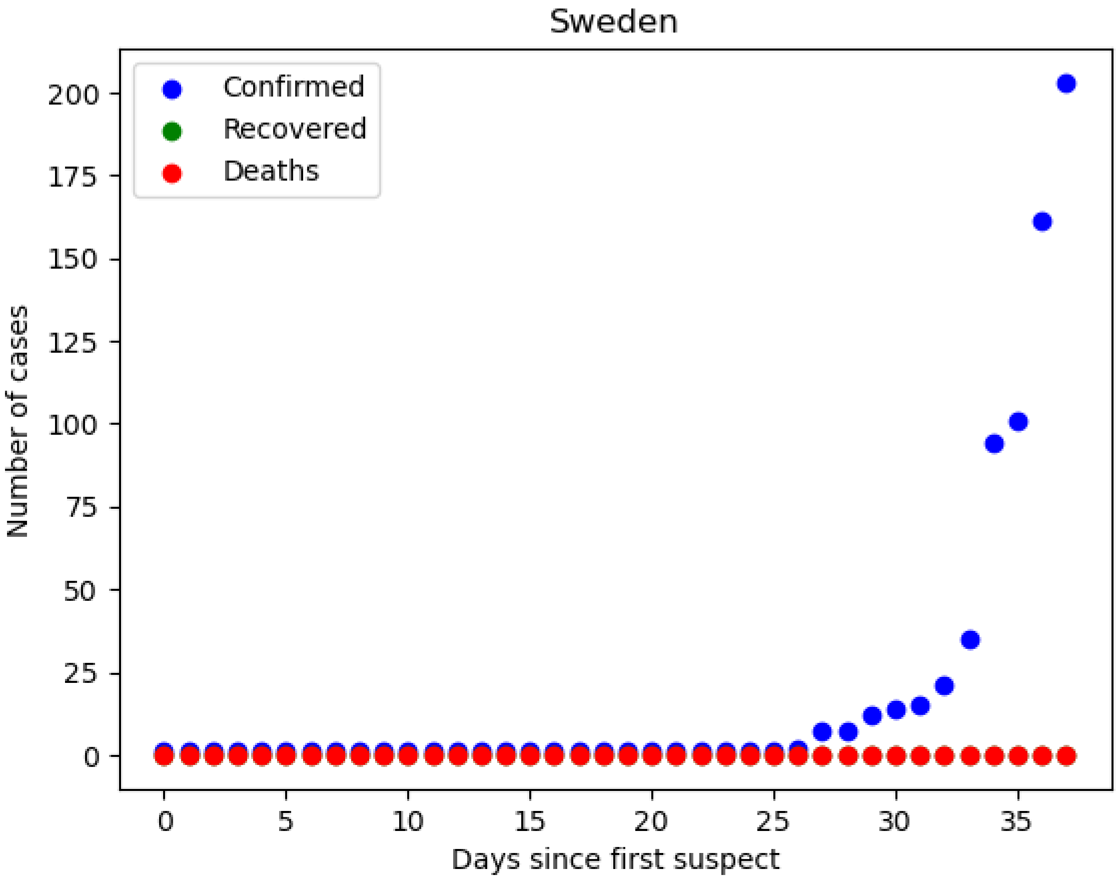

United Arab Emirates
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
0
5
10
15
20
25
30
35
40
Days since first suspect

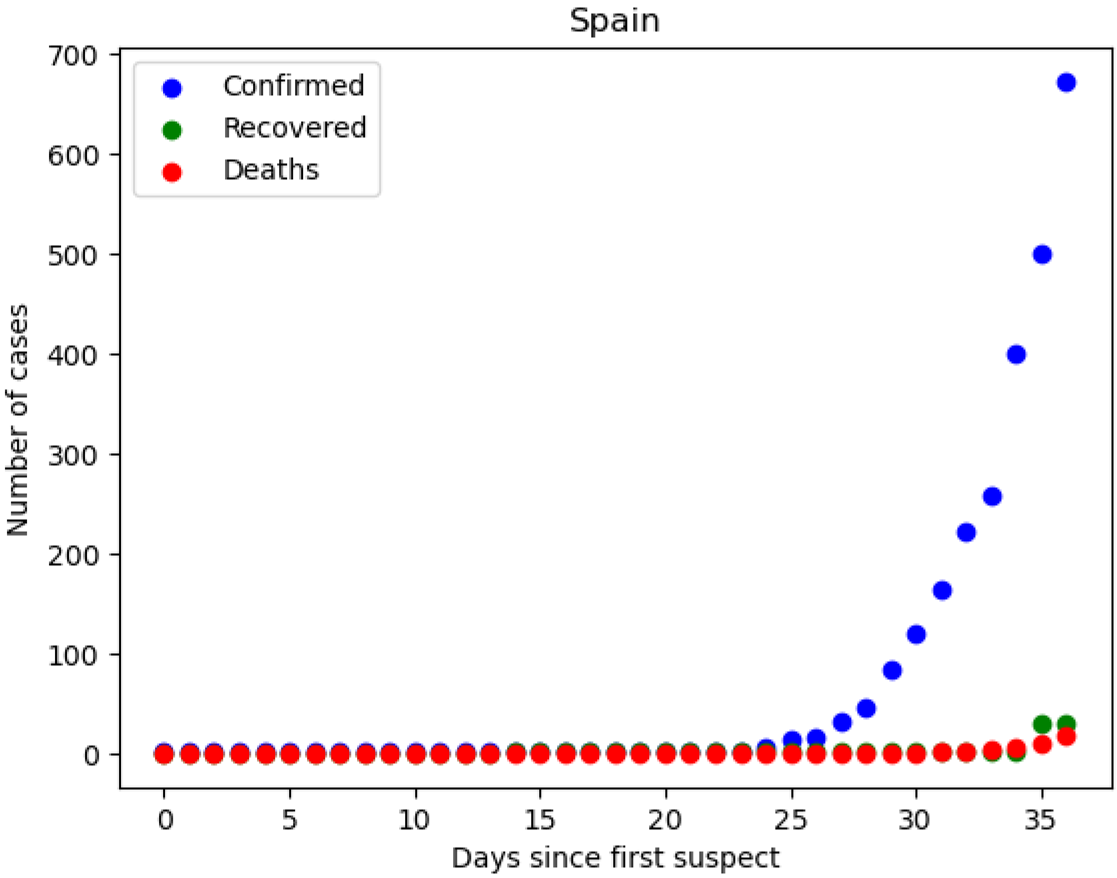

India
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
35
40
Days since first suspect
0
5
10
15
20
25
30
35

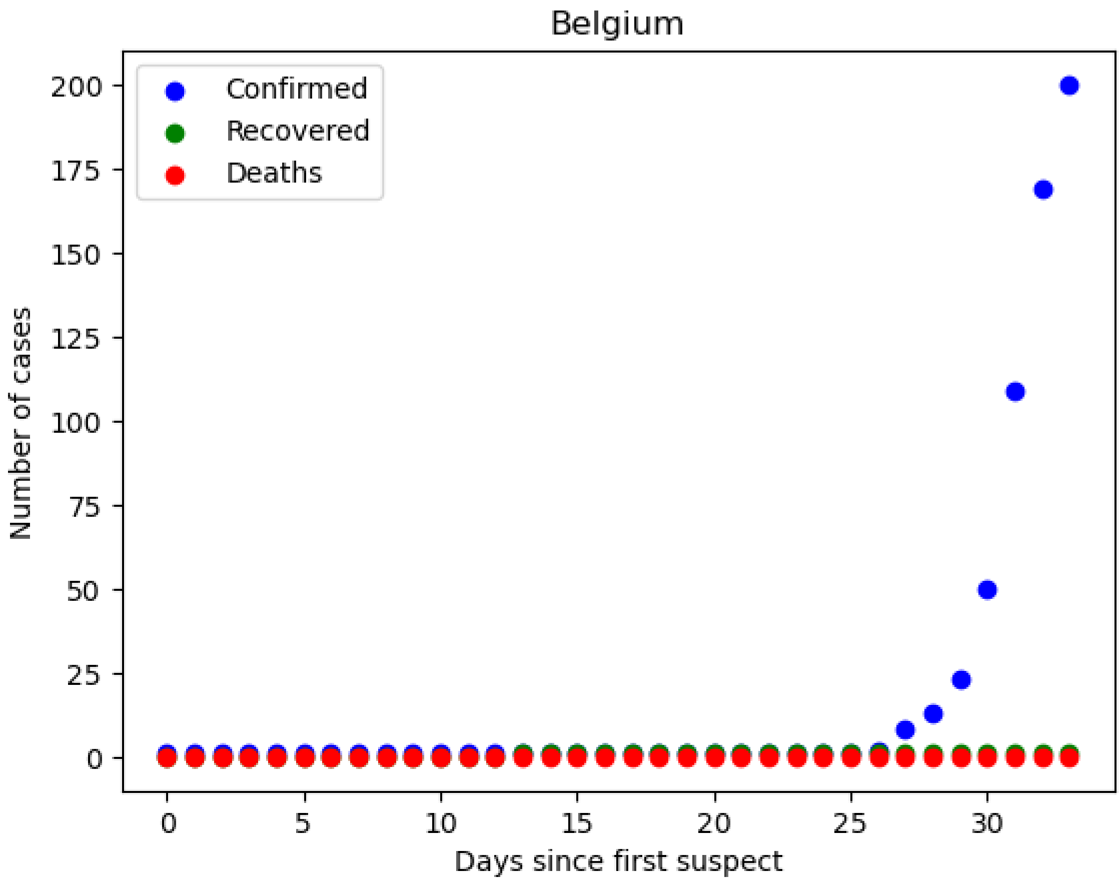

Italy
Confirmed
Recovered
Deaths
Number of cases
0
1000
2000
3000
4000
5000
6000
7000
Days since first suspect
0
5
10
15
20
25
30
35


UK
Confirmed
Recovered
Deaths
Number of cases
0
50
100
150
200
250
0
5
10
15
20
25
30
35
Days since first suspect

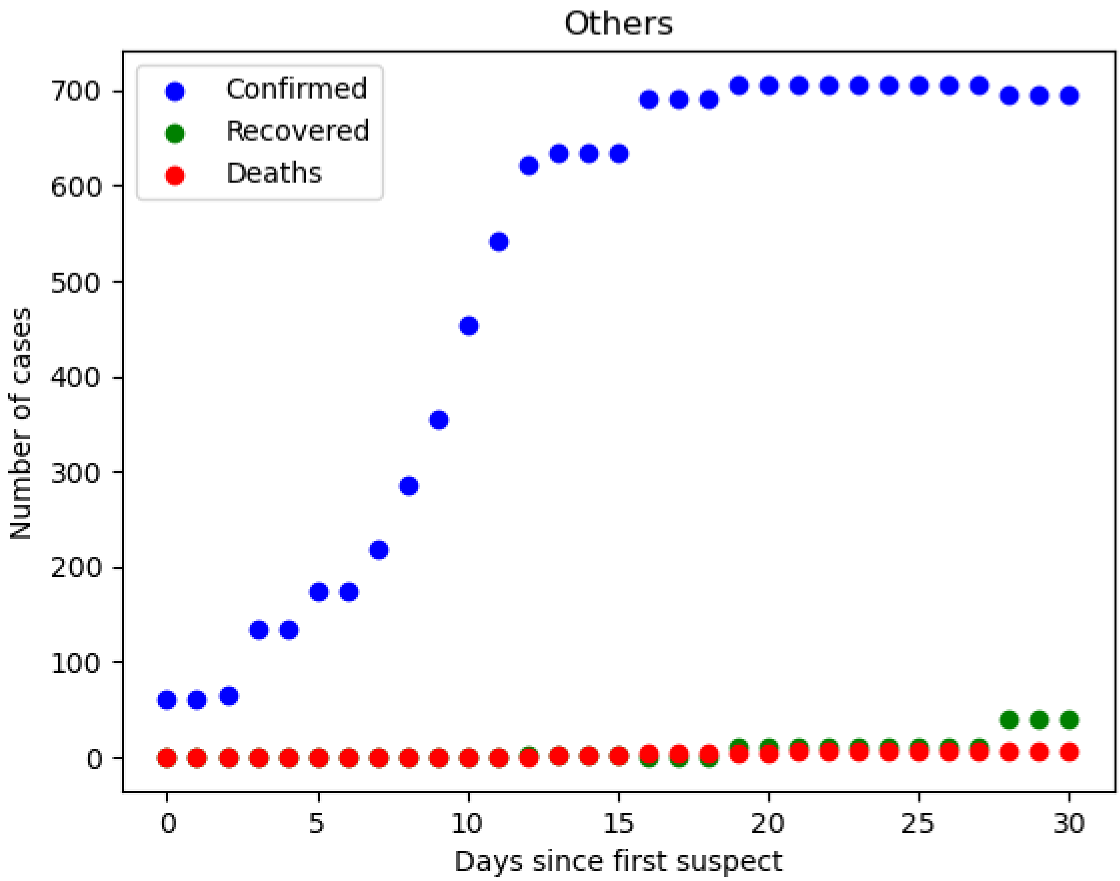

Russia
Confirmed
Recovered
Deaths
Number of cases
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0
5
10
15
20
25
30
35
Days since first suspect


Sweden
Confirmed
Recovered
Deaths
Number of cases
0
25
50
75
100
125
150
175
200
0
5
10
15
20
25
30
35
Days since first suspect

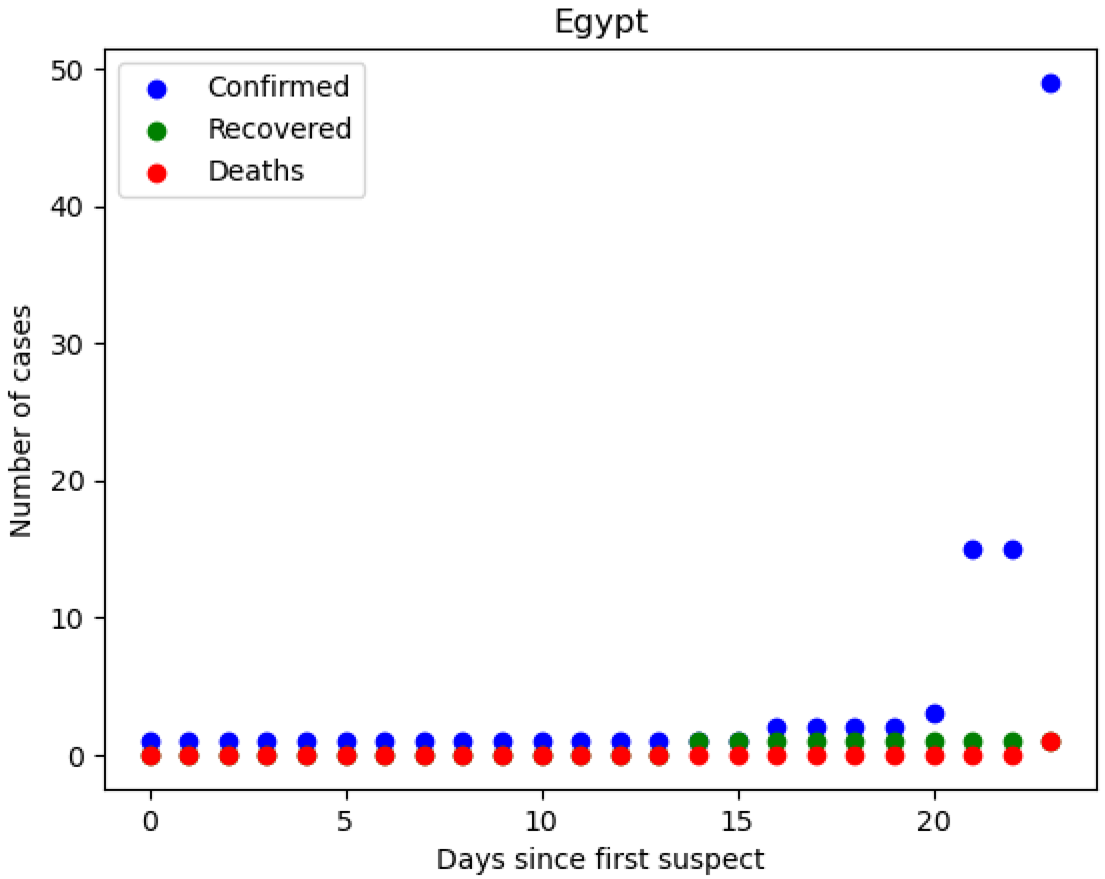

Spain
Confirmed
Recovered
Deaths
Number of cases
0
100
200
300
400
500
600
700
0
5
10
15
20
25
30
35
Days since first suspect

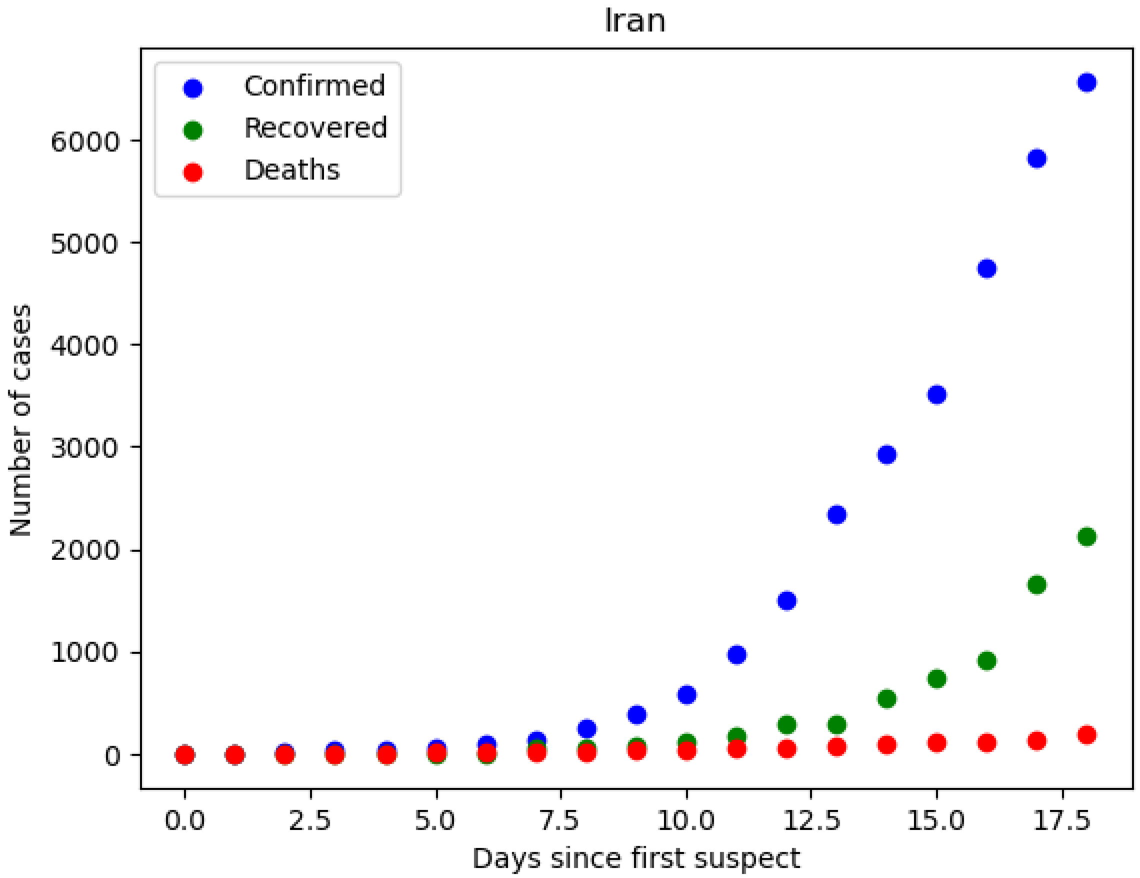


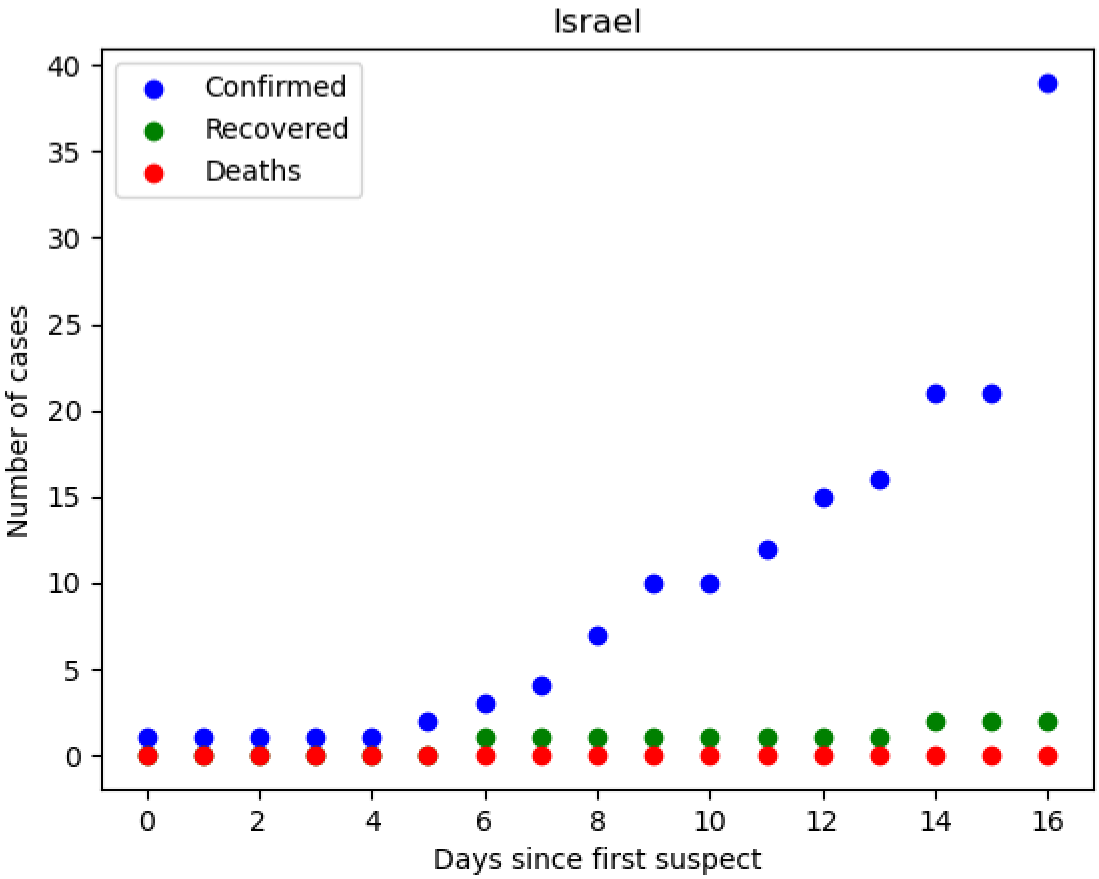
Belgium

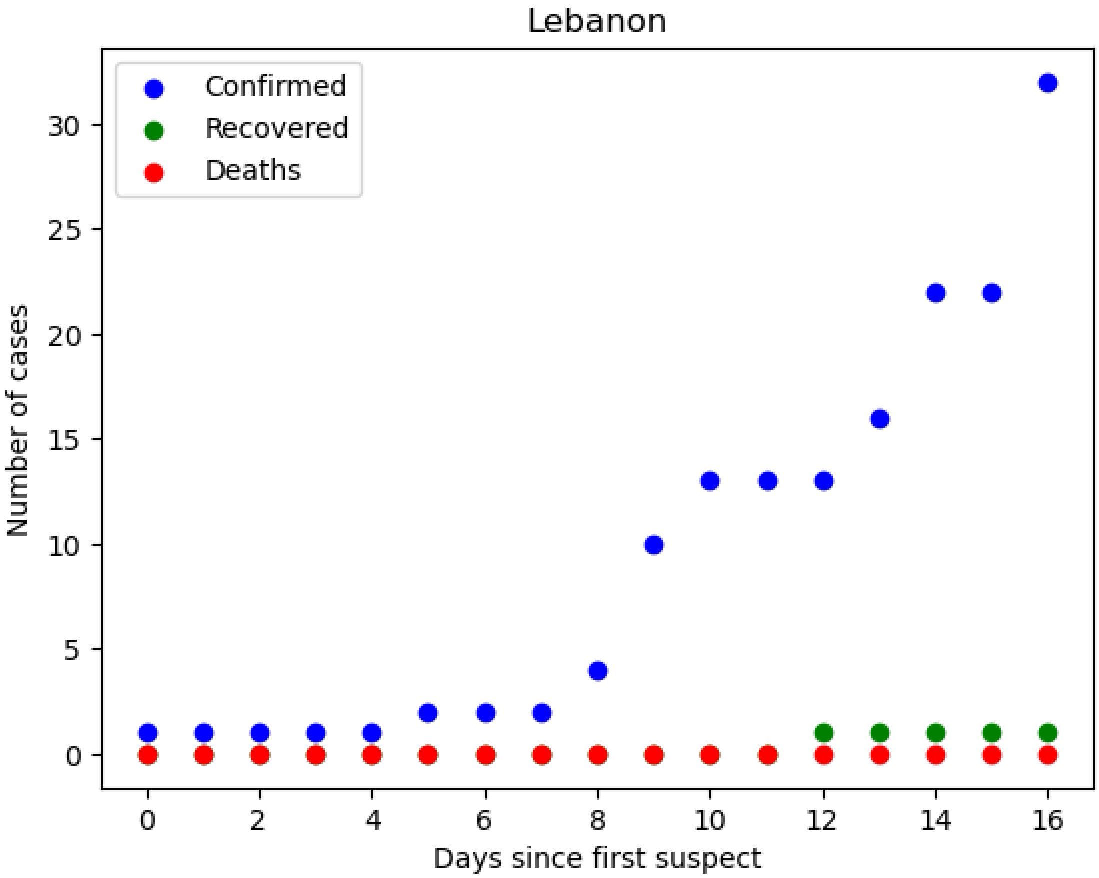


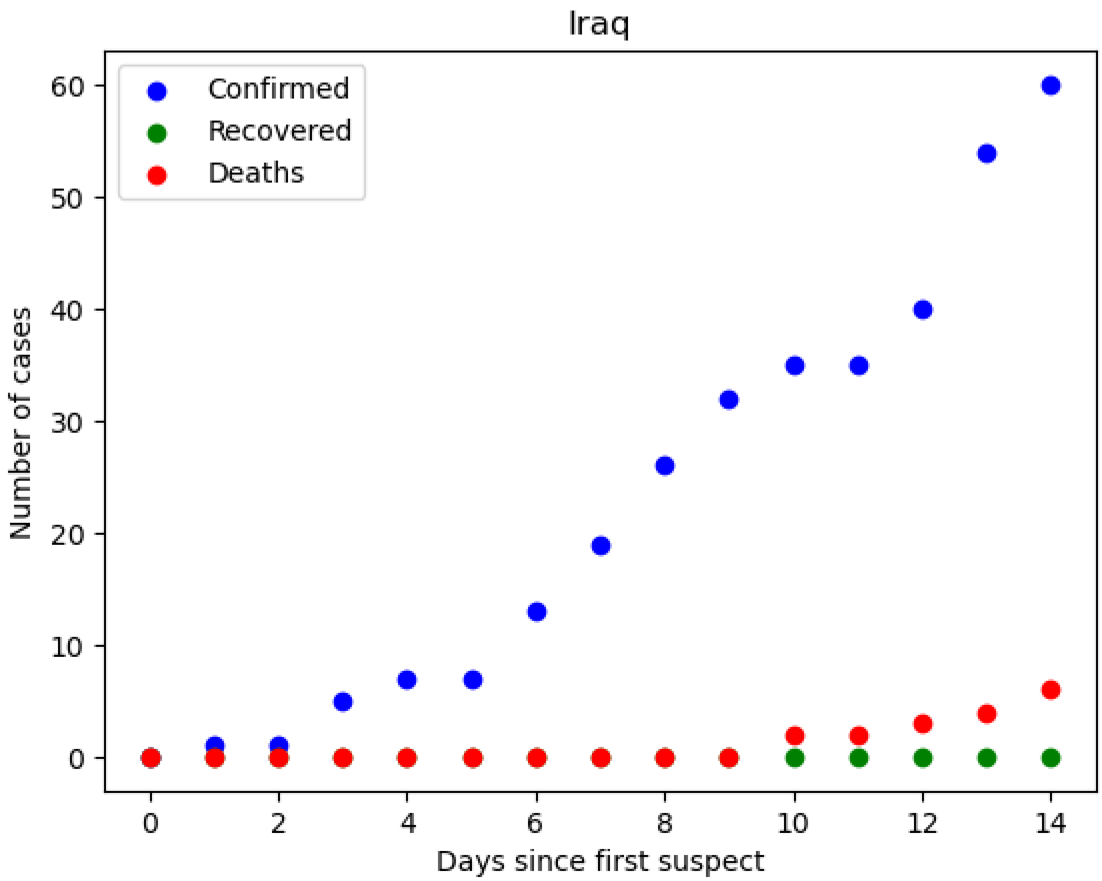
Others

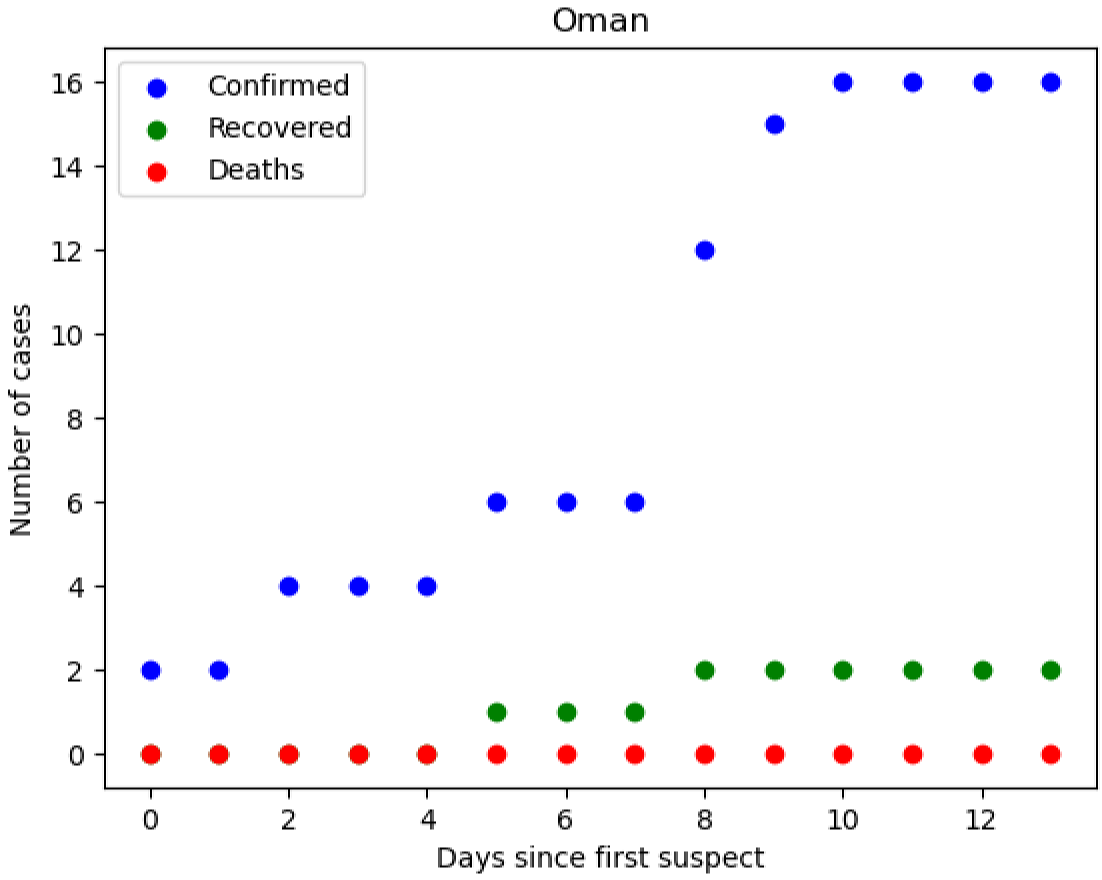

Egypt
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
50
0
5
10
15
20
Days since first suspect

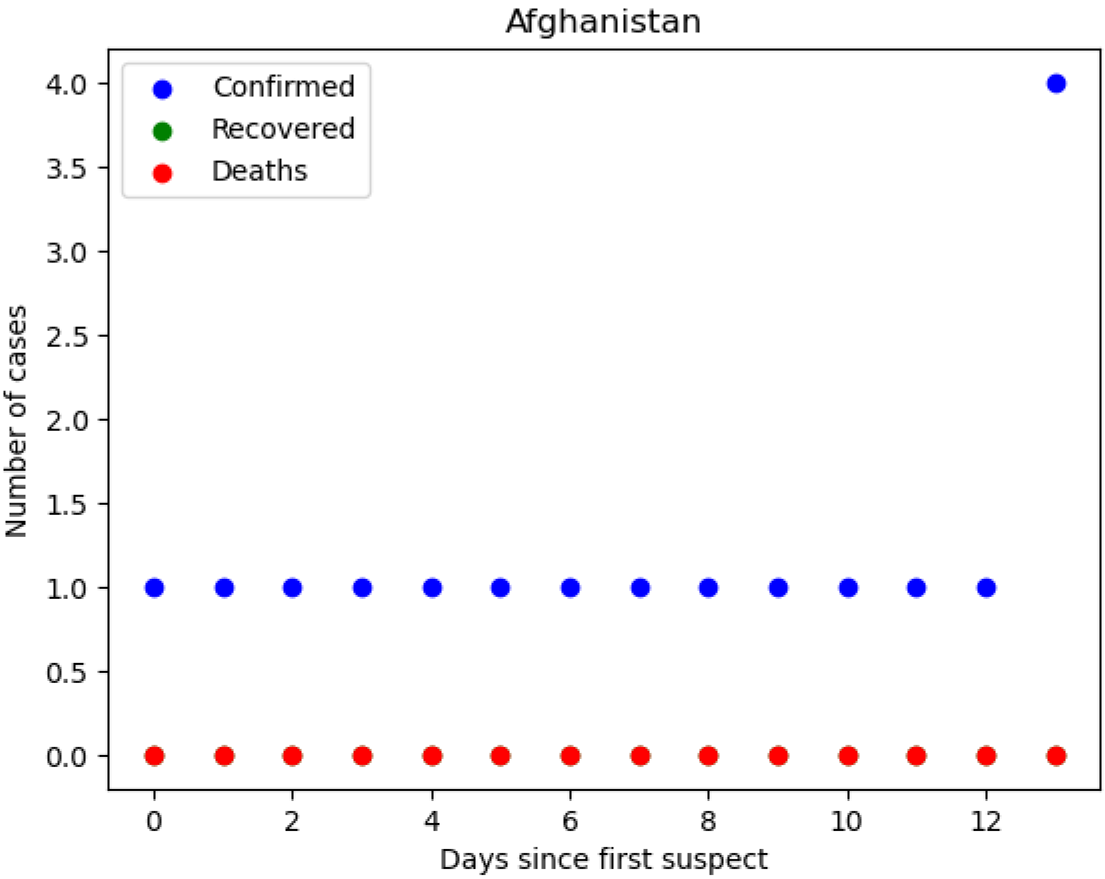

Iran
Confirmed
Recovered
Deaths
Number of cases
0
1000
2000
3000
4000
5000
6000
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Days since first suspect

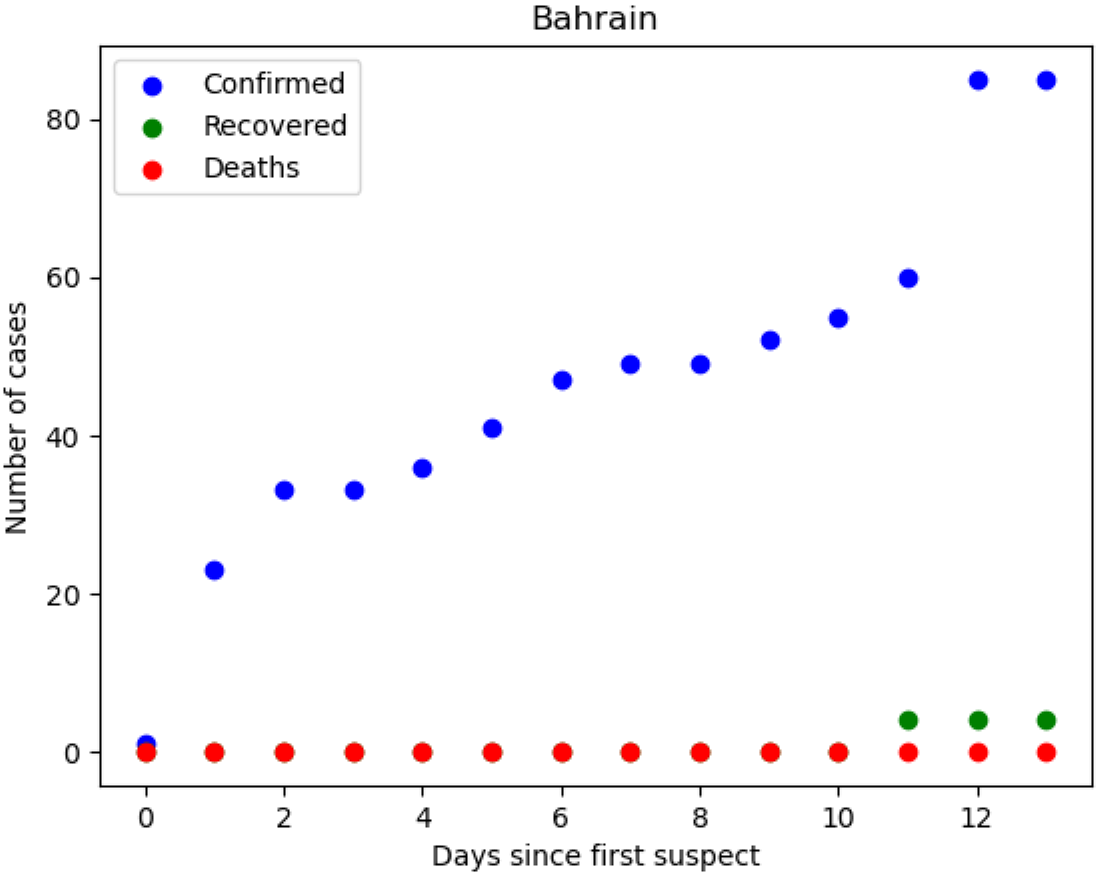

Israel
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
35
40
0
2
4
6
8
10
12
14
16
Days since first suspect

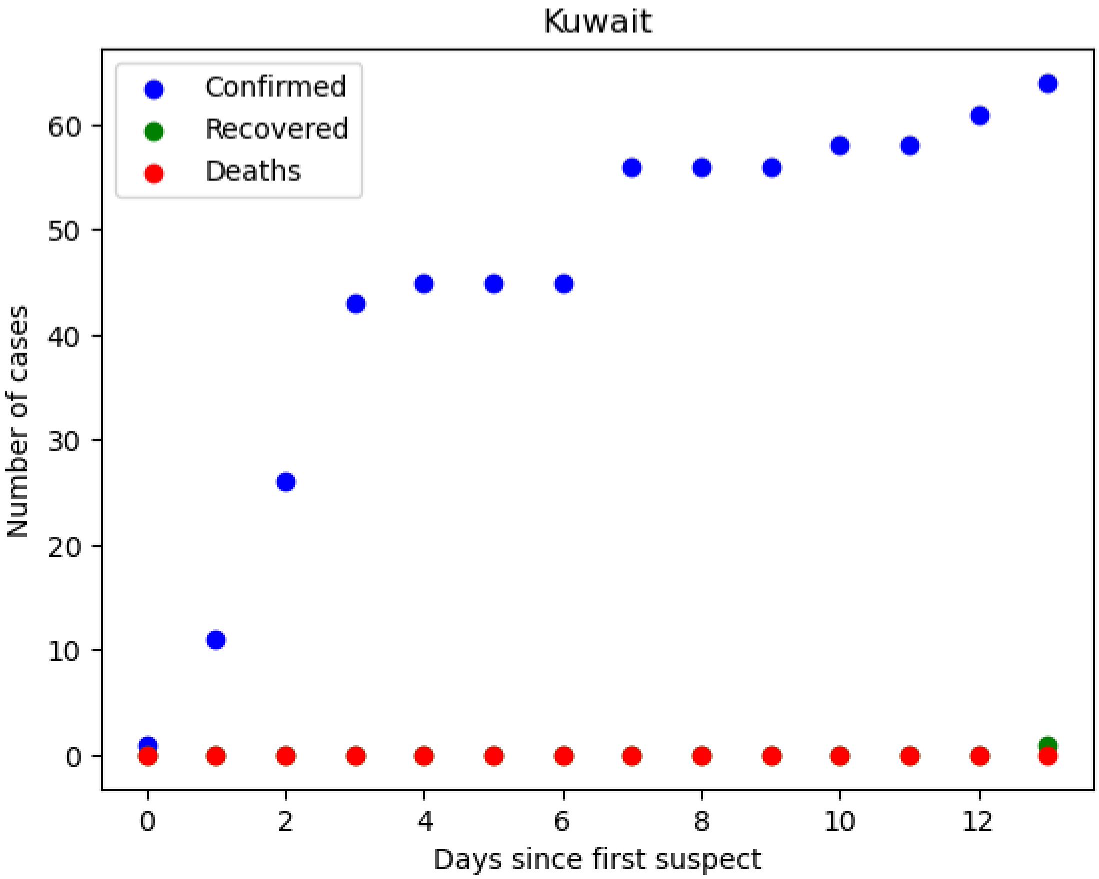

Lebanon
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
0
2
4
6
8
10
12
14
16
Days since first suspect

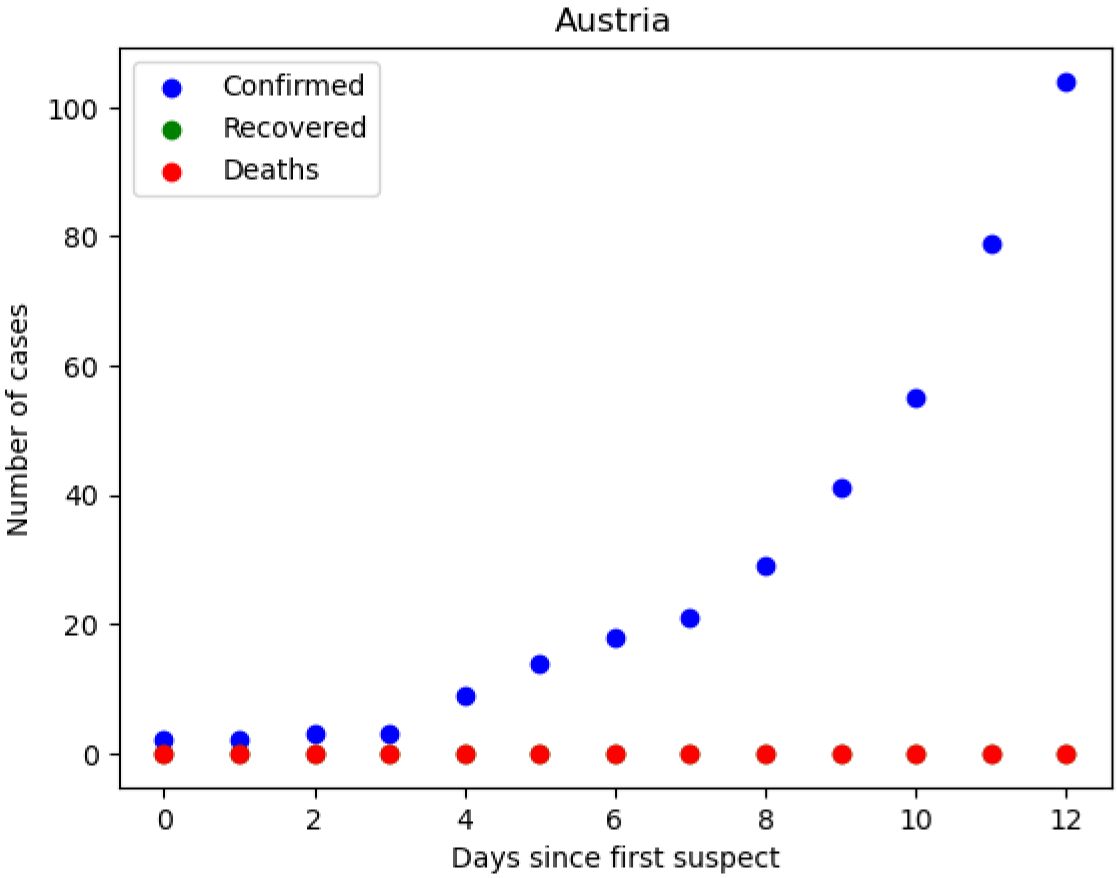

Iraq
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
50
60
0
2
4
6
8
10
12
14
Days since first suspect

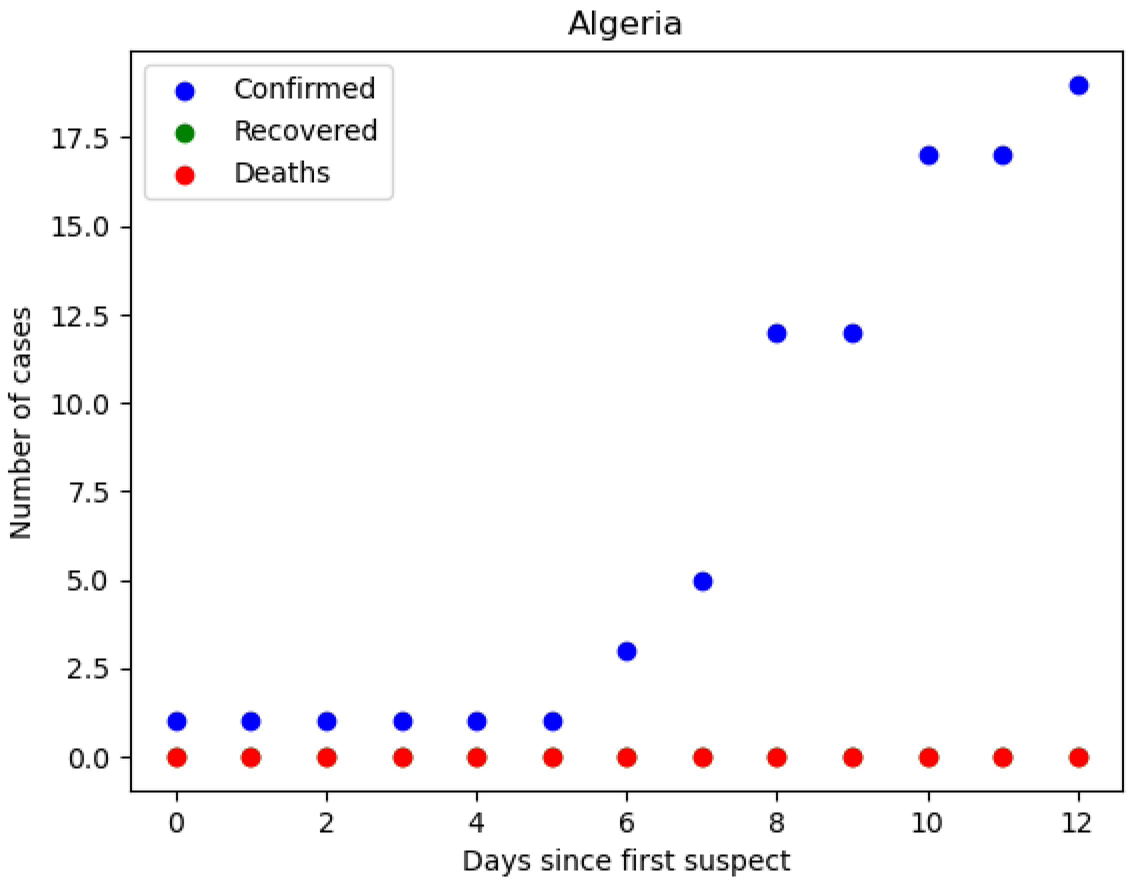

Oman
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
12
14
16
0
2
4
6
8
10
12
Days since first suspect

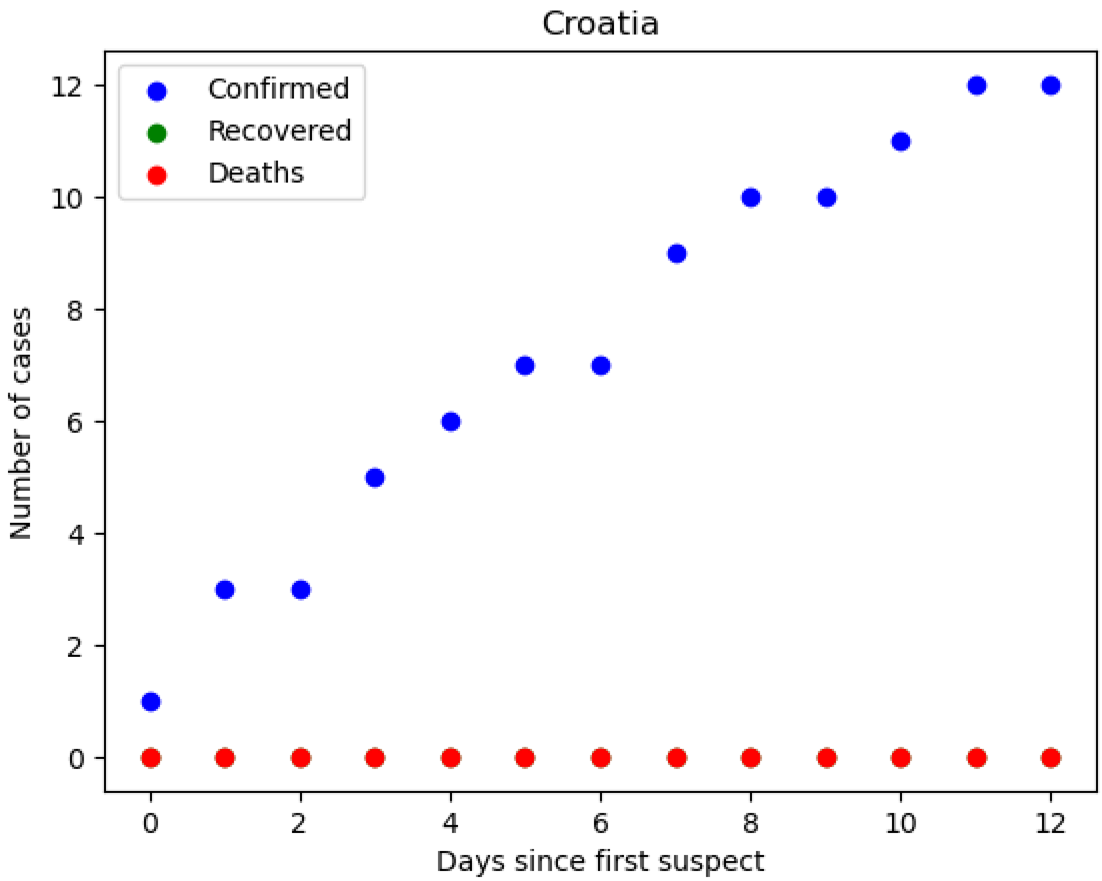

Afghanistan
Confirmed
Recovered
Deaths
Number of cases
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
2
4
6
8
10
12
Days since first suspect

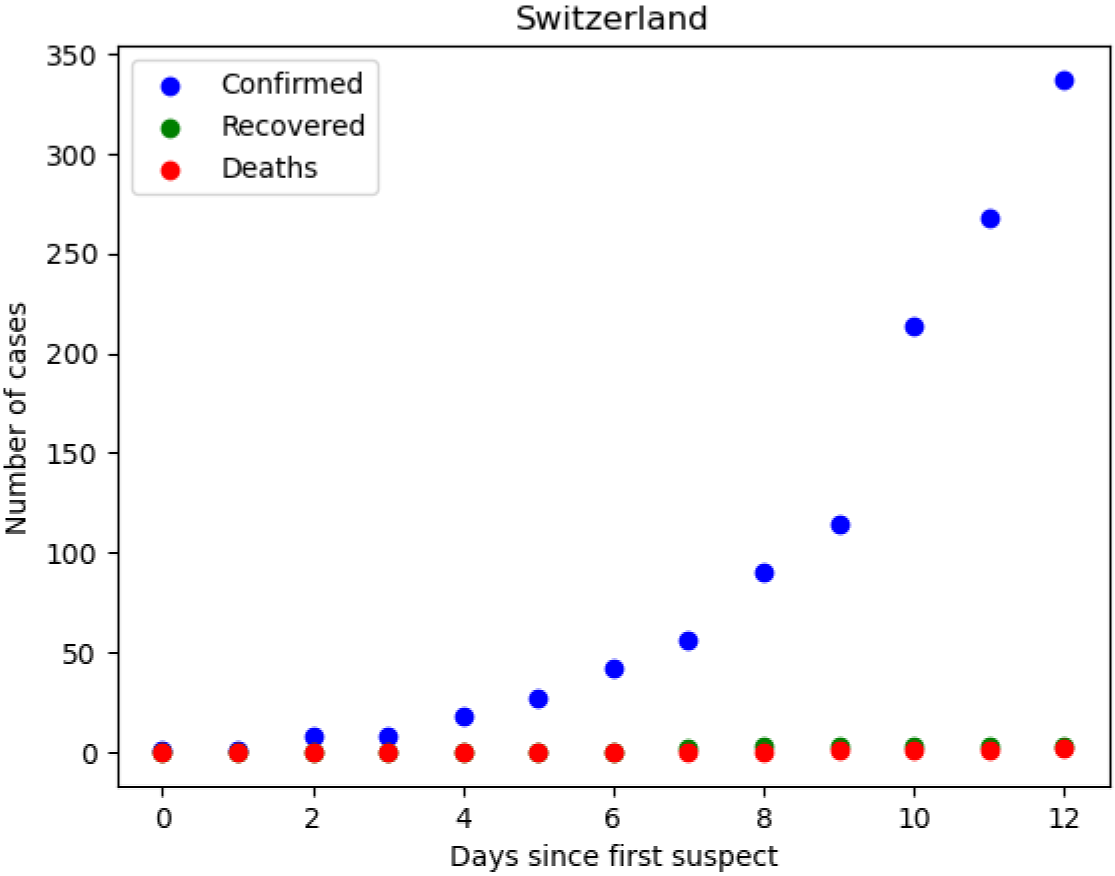

Bahrain
Confirmed
Recovered
Deaths
Number of cases
80
60
40
20
0
0
2
4
6
8
10
12
Days since first suspect

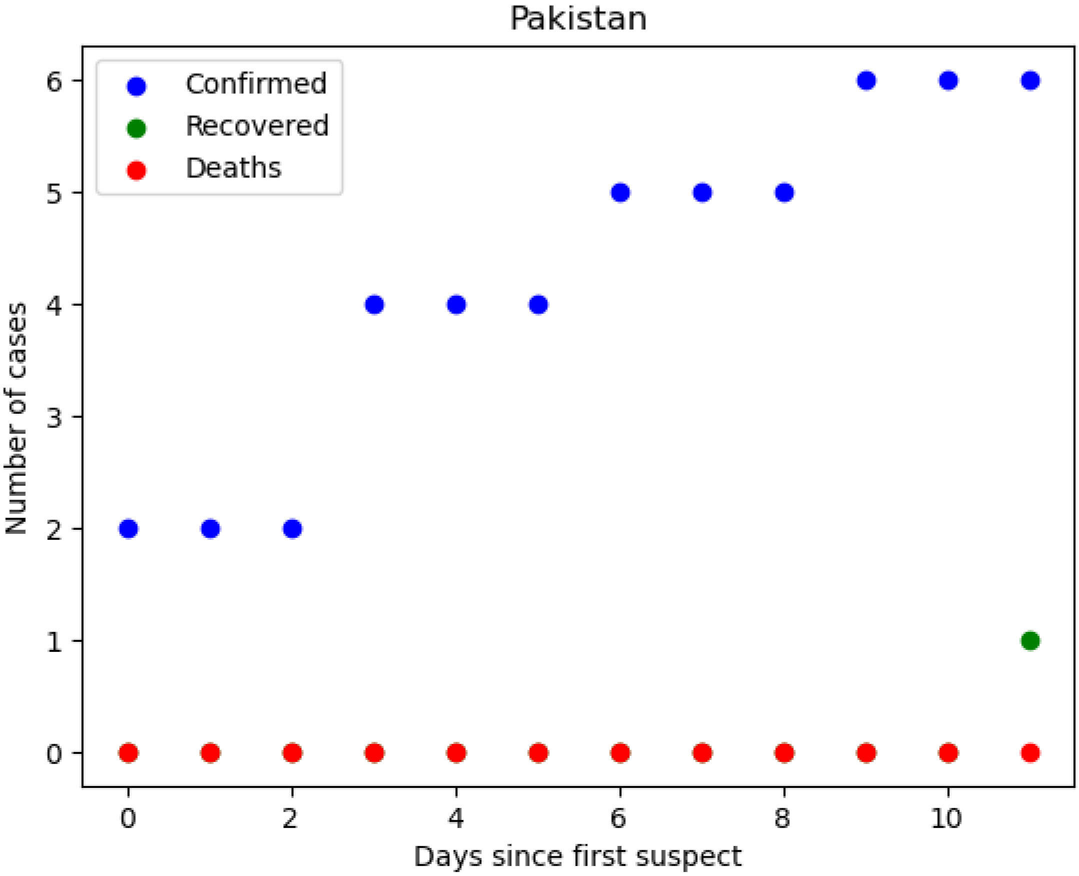

Kuwait
● Confirmed
● Recovered
● Deaths
Number of cases
0
10
20
30
40
50
60
0
2
4
6
8
10
12
Days since first suspect

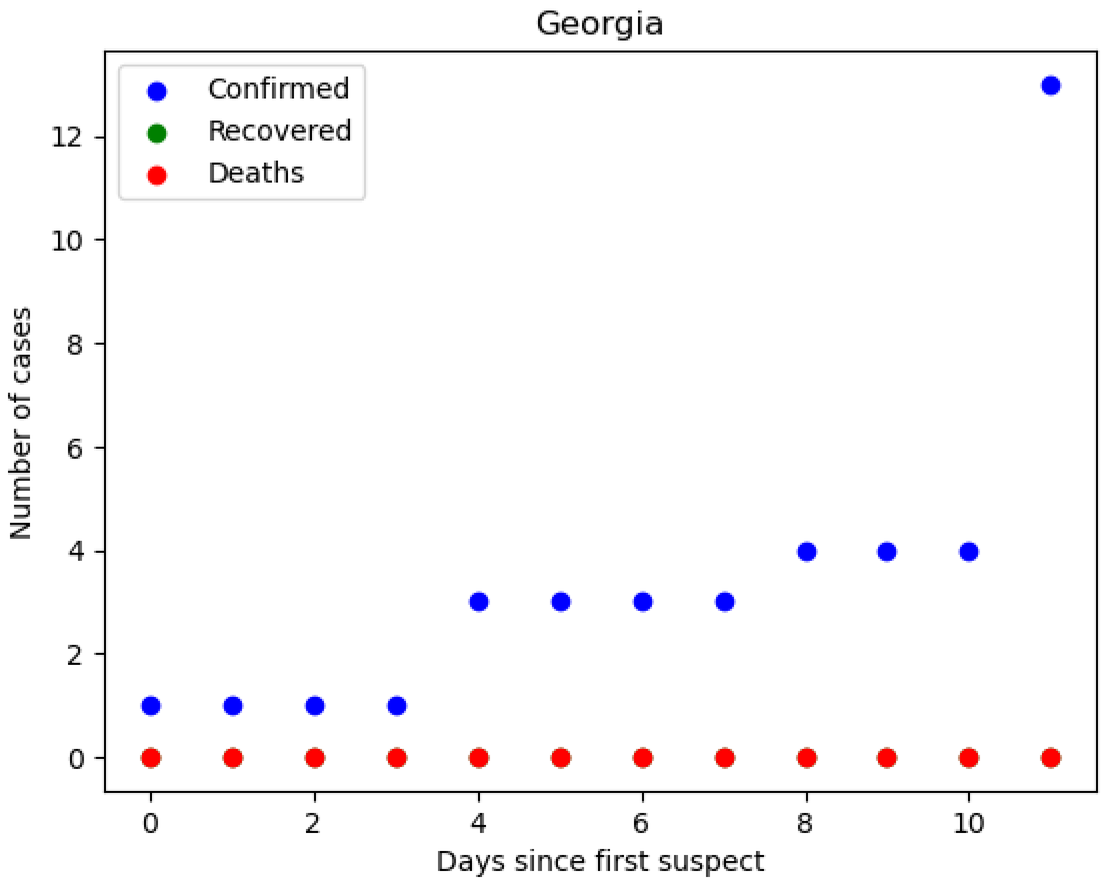

Austria
● Confirmed
● Recovered
● Deaths
Number of cases
0
20
40
60
80
100
0
2
4
6
8
10
12
Days since first suspect

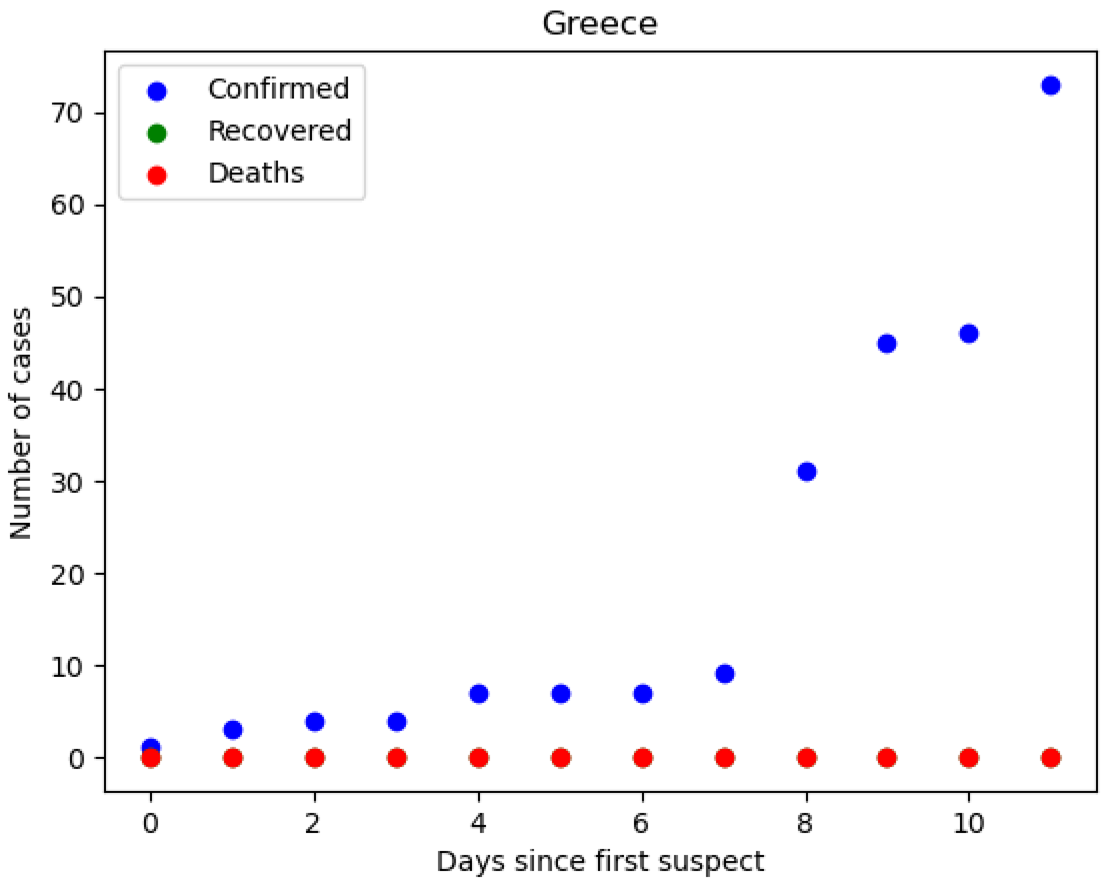

Algeria
Confirmed
Recovered
Deaths
Number of cases
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Days since first suspect
0
2
4
6
8
10
12

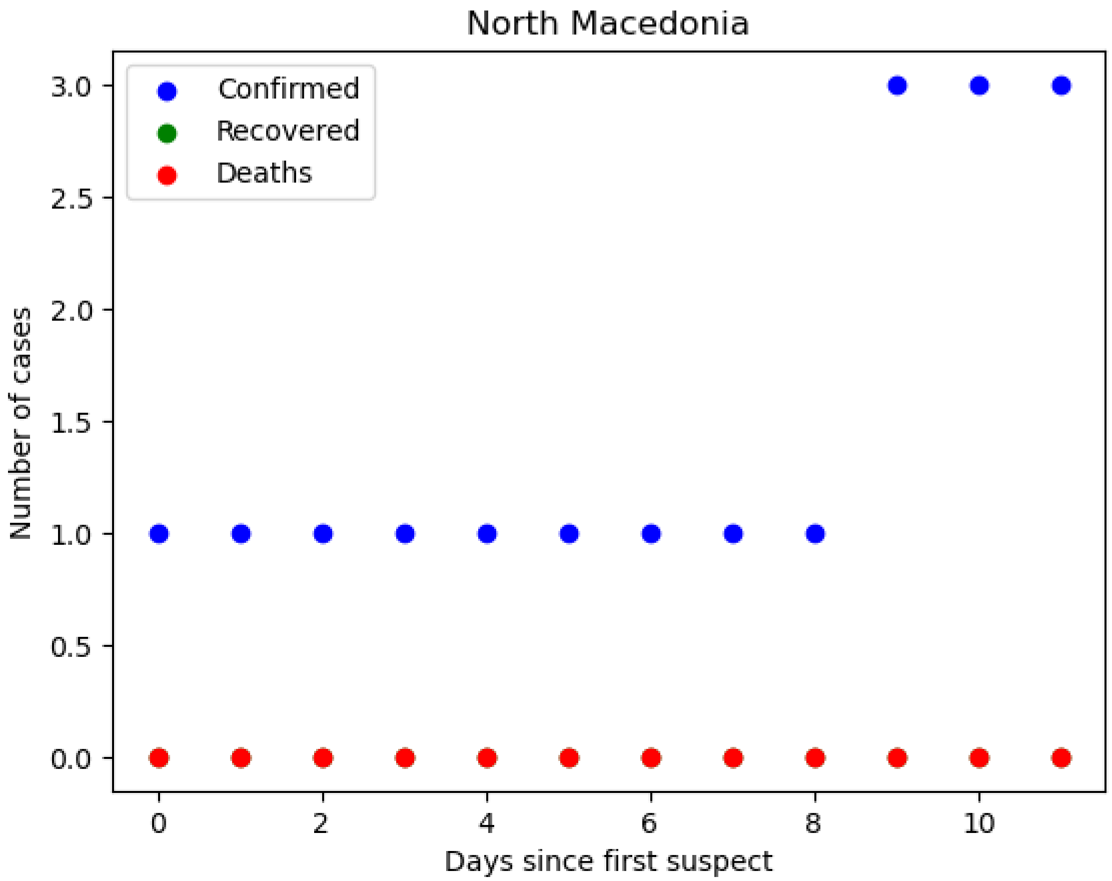

Croatia
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
12
Days since first suspect
0
2
4
6
8
10
12

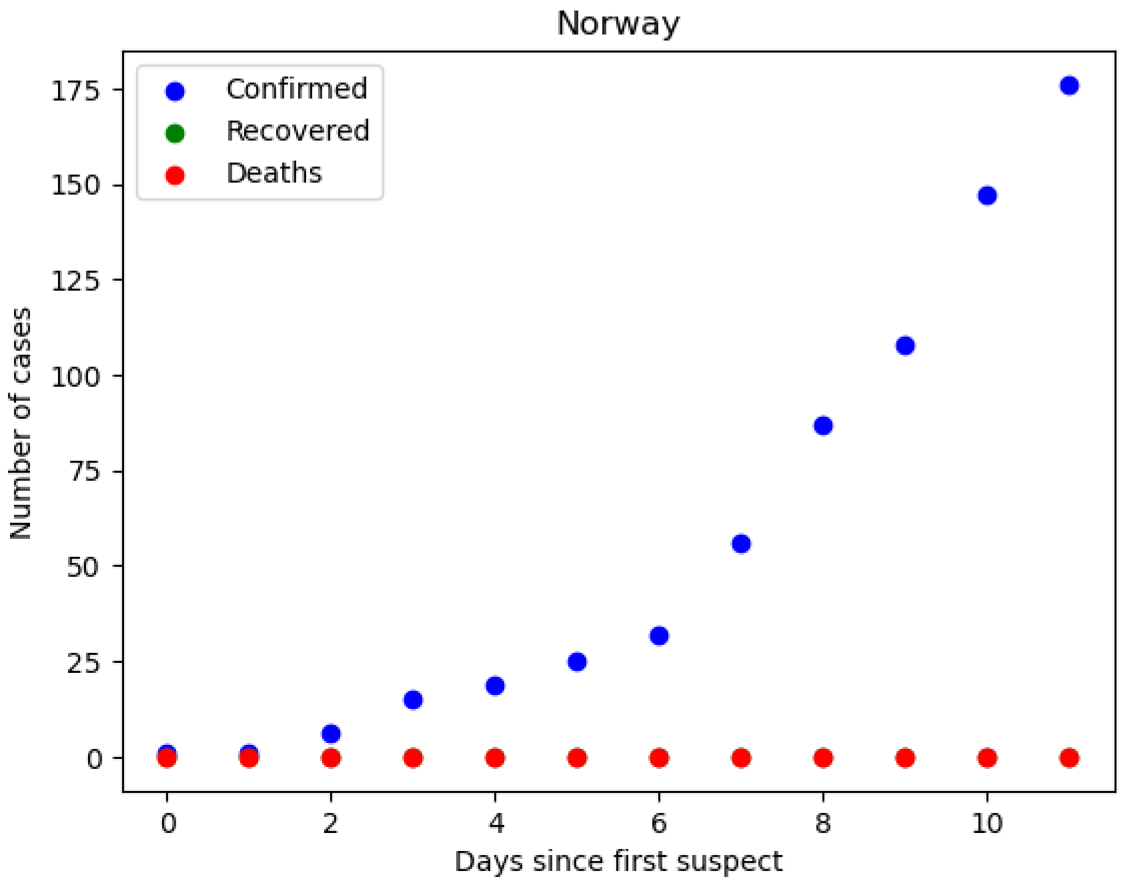

Switzerland
Confirmed
Recovered
Deaths
Number of cases
0
50
100
150
200
250
300
350
Days since first suspect
0
2
4
6
8
10
12

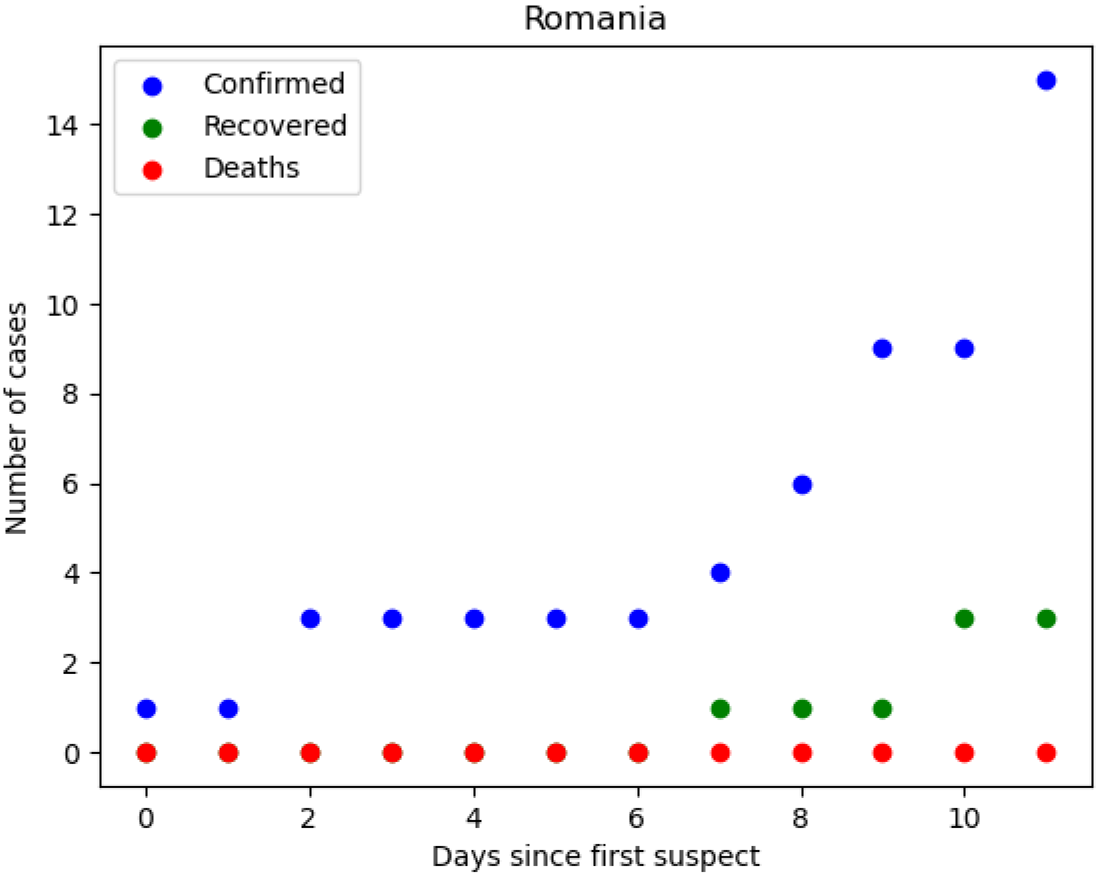

Pakistan
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
6
Days since first suspect
0
2
4
6
8
10

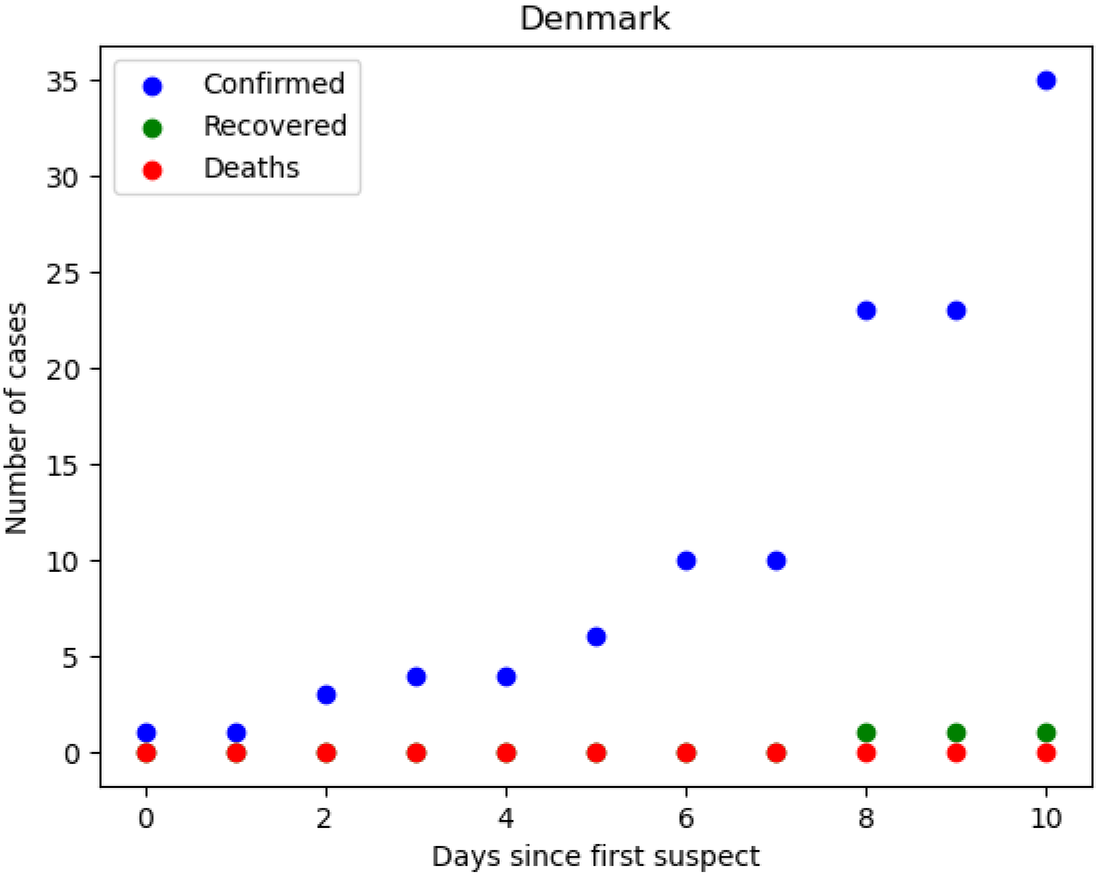

Georgia
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
12
Days since first suspect
0
2
4
6
8
10

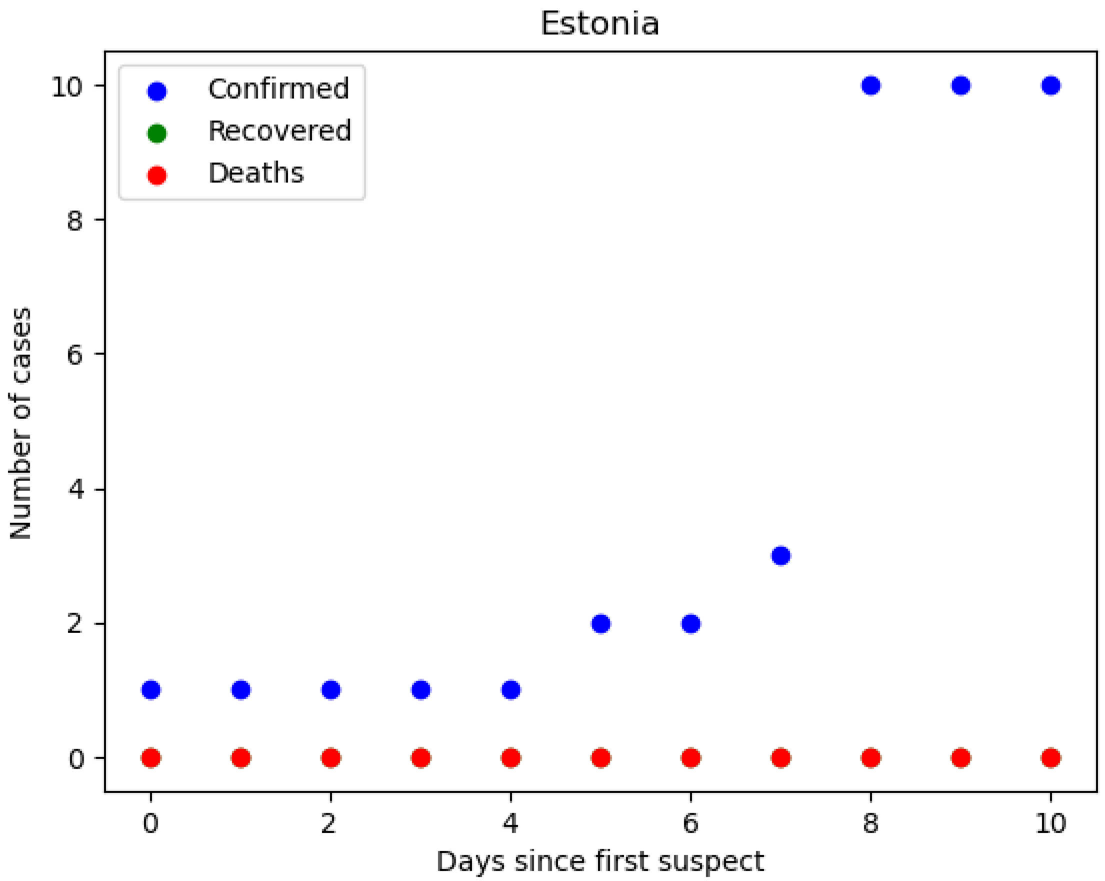

Greece
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
50
60
70
Days since first suspect
0
2
4
6
8
10


North Macedonia
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
2
4
6
8
10
Days since first suspect

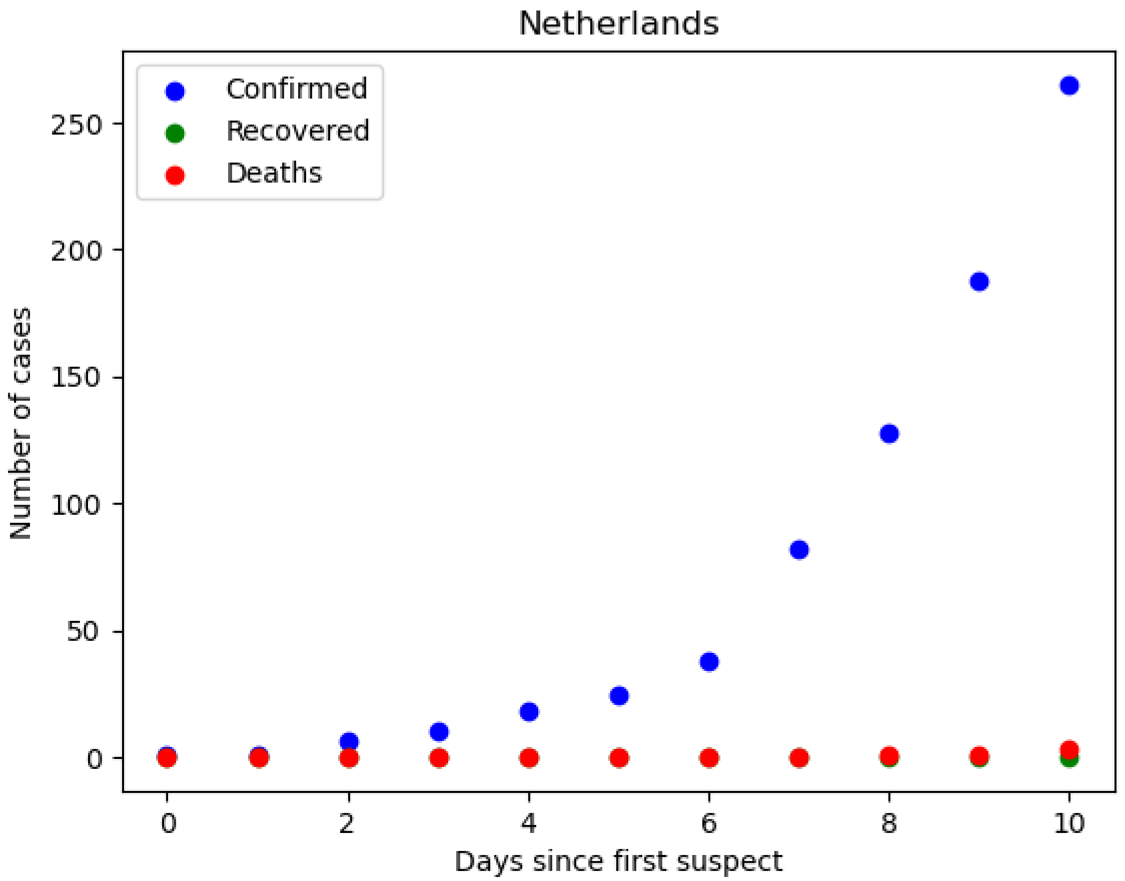

Norway
Confirmed
Recovered
Deaths
Number of cases
175
150
125
100
75
50
25
0
0
2
4
6
8
10
Days since first suspect


Romania
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
12
14
Days since first suspect
0
2
4
6
8
10

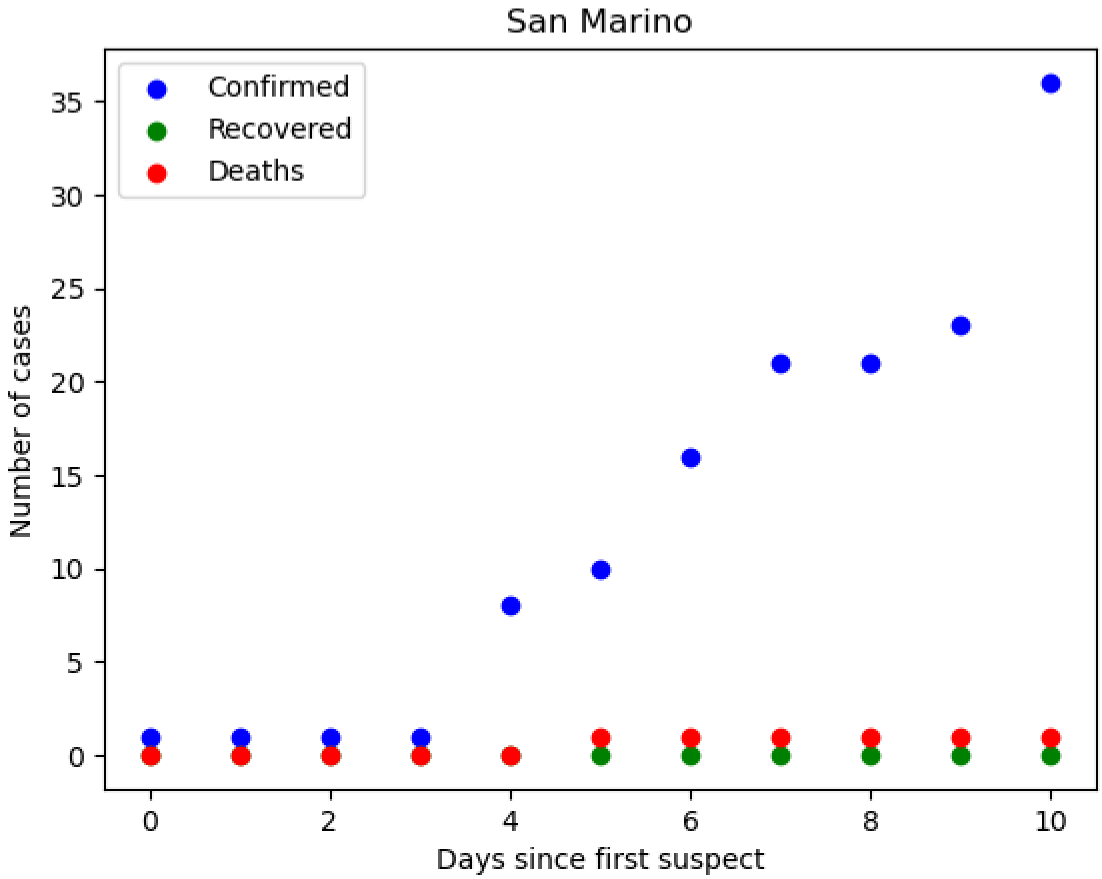

Denmark
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
35
Days since first suspect
0
2
4
6
8
10

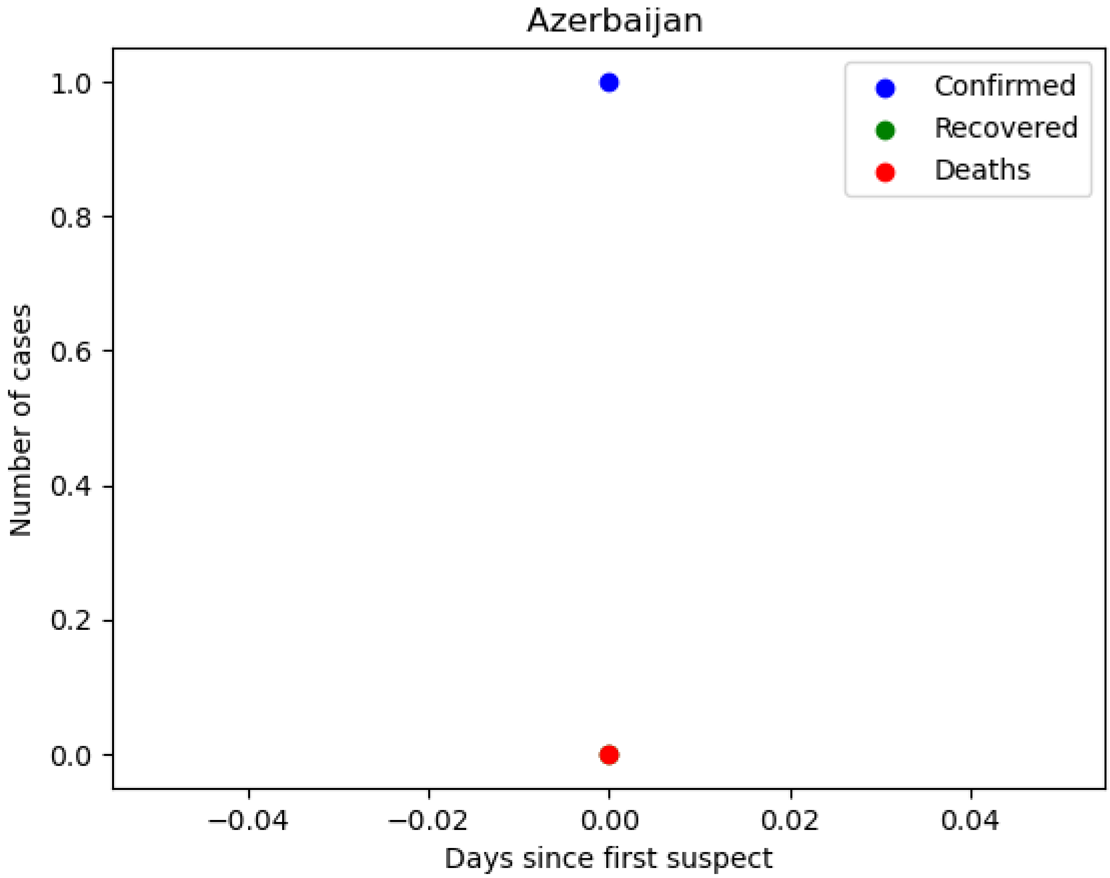


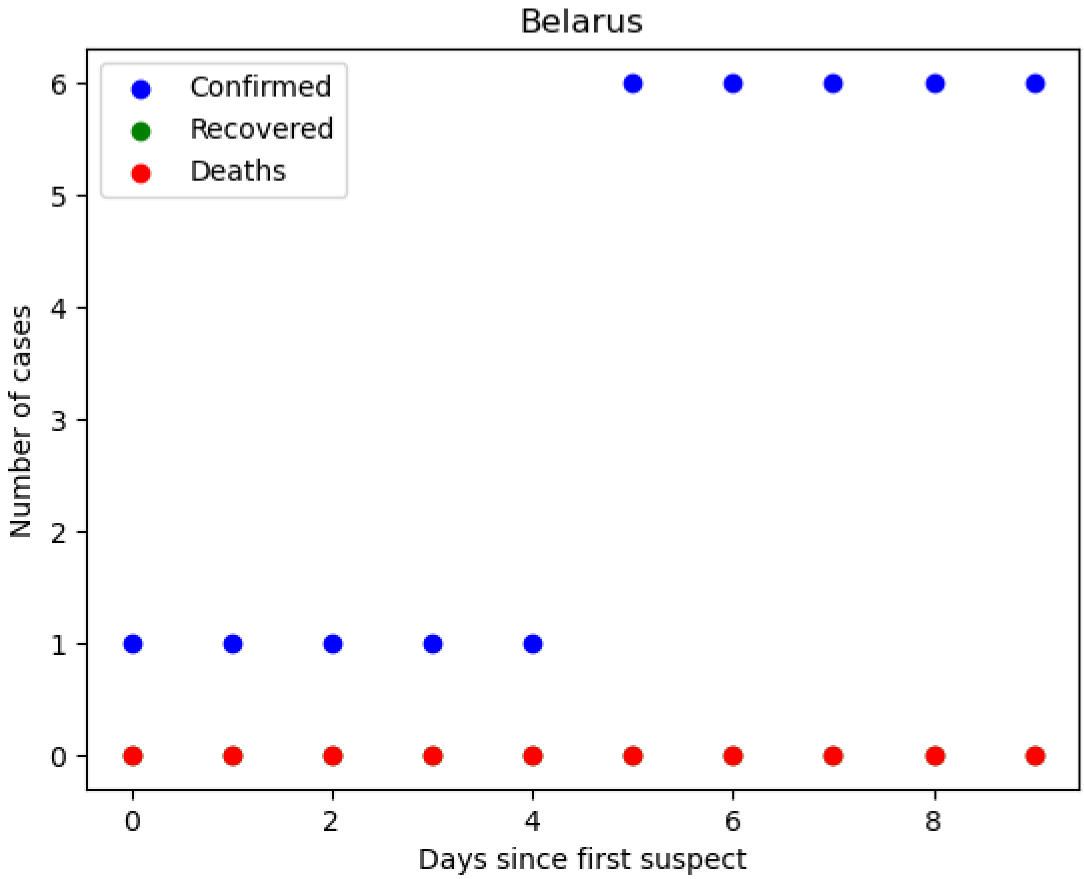
Estonia
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
Days since first suspect
0
2
4
6
8
10

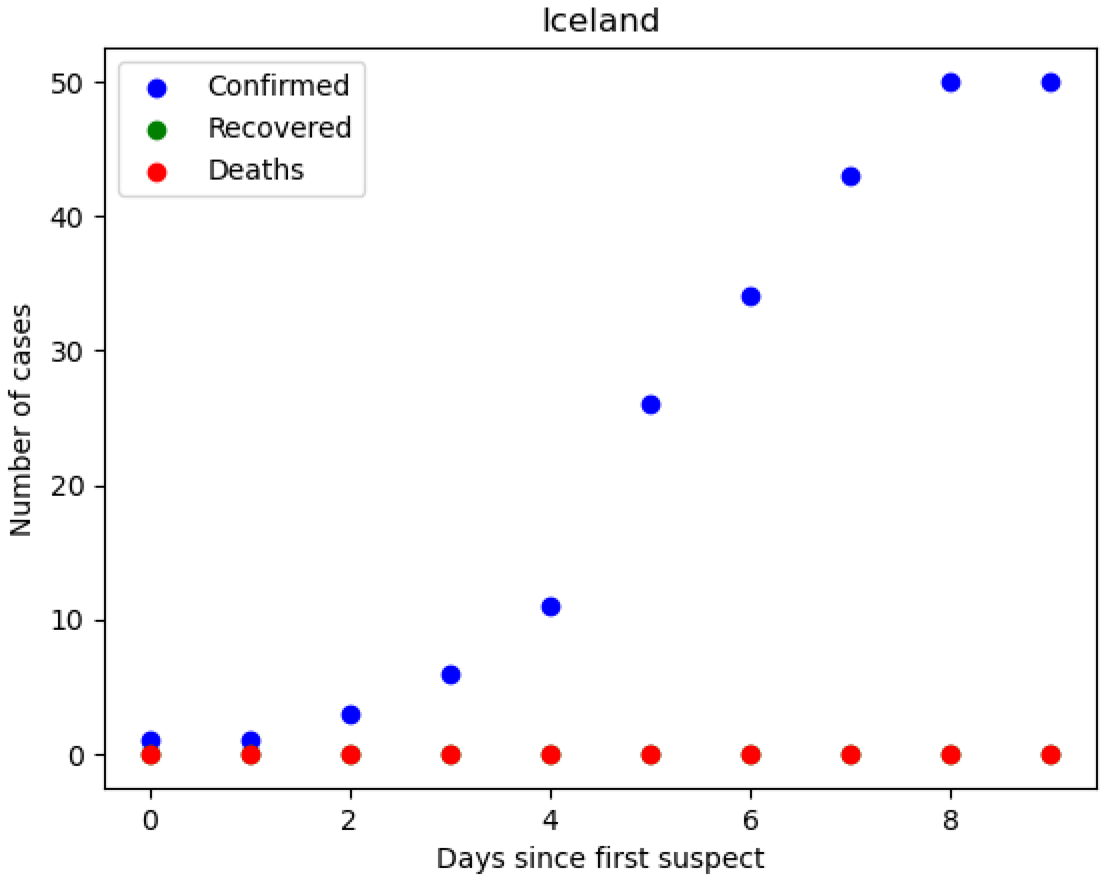


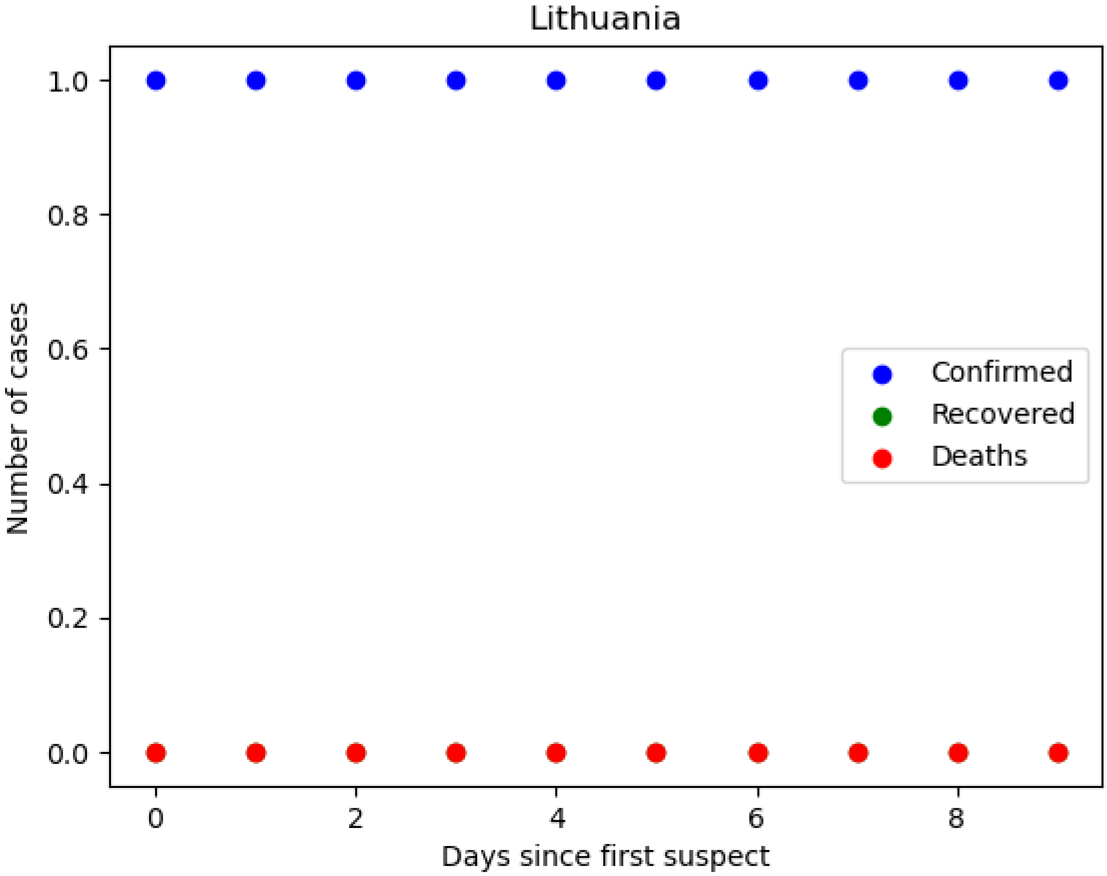
Netherlands
Confirmed
Recovered
Deaths
Number of cases
0
50
100
150
200
250
Days since first suspect
0
2
4
6
8
10

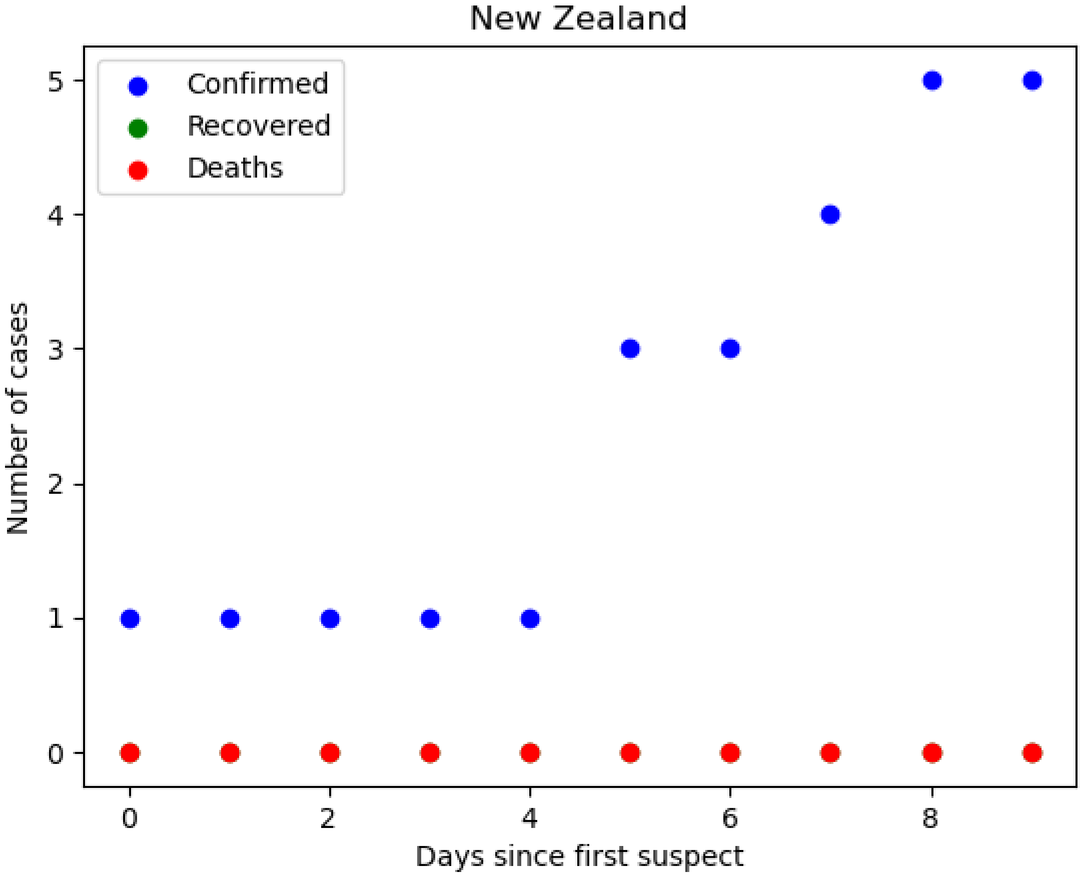

San Marino
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
35
0
2
4
6
8
10
Days since first suspect

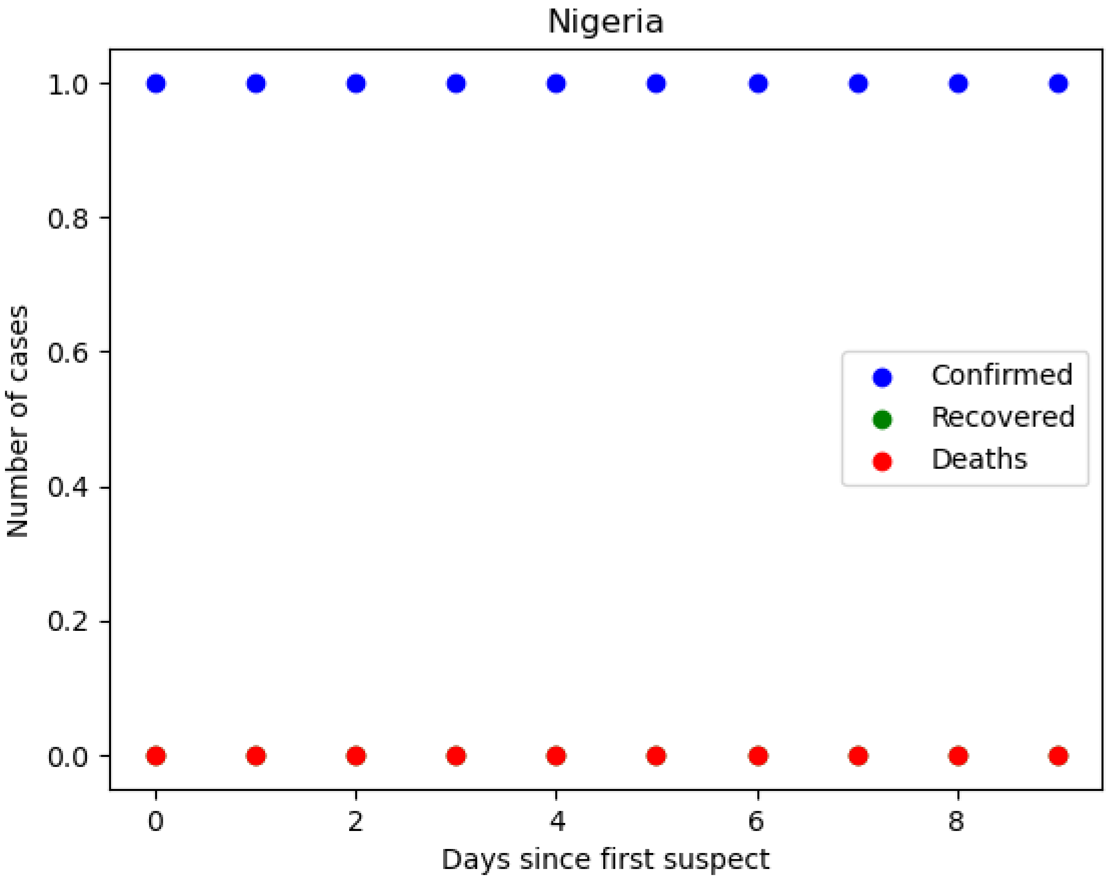

Azerbaijan
Confirmed
Recovered
Deaths
Number of cases
0.0
0.2
0.4
0.6
0.8
1.0
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect

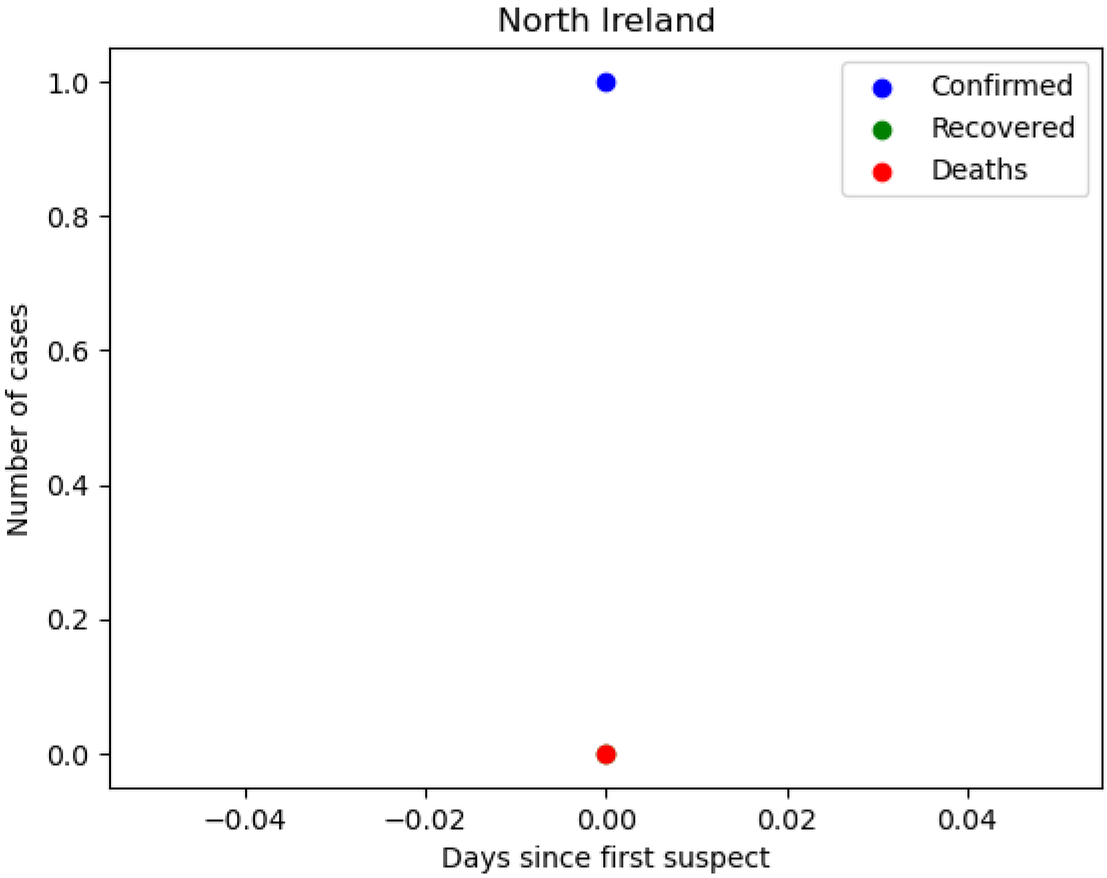

Belarus
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
6
0
2
4
6
8
Days since first suspect

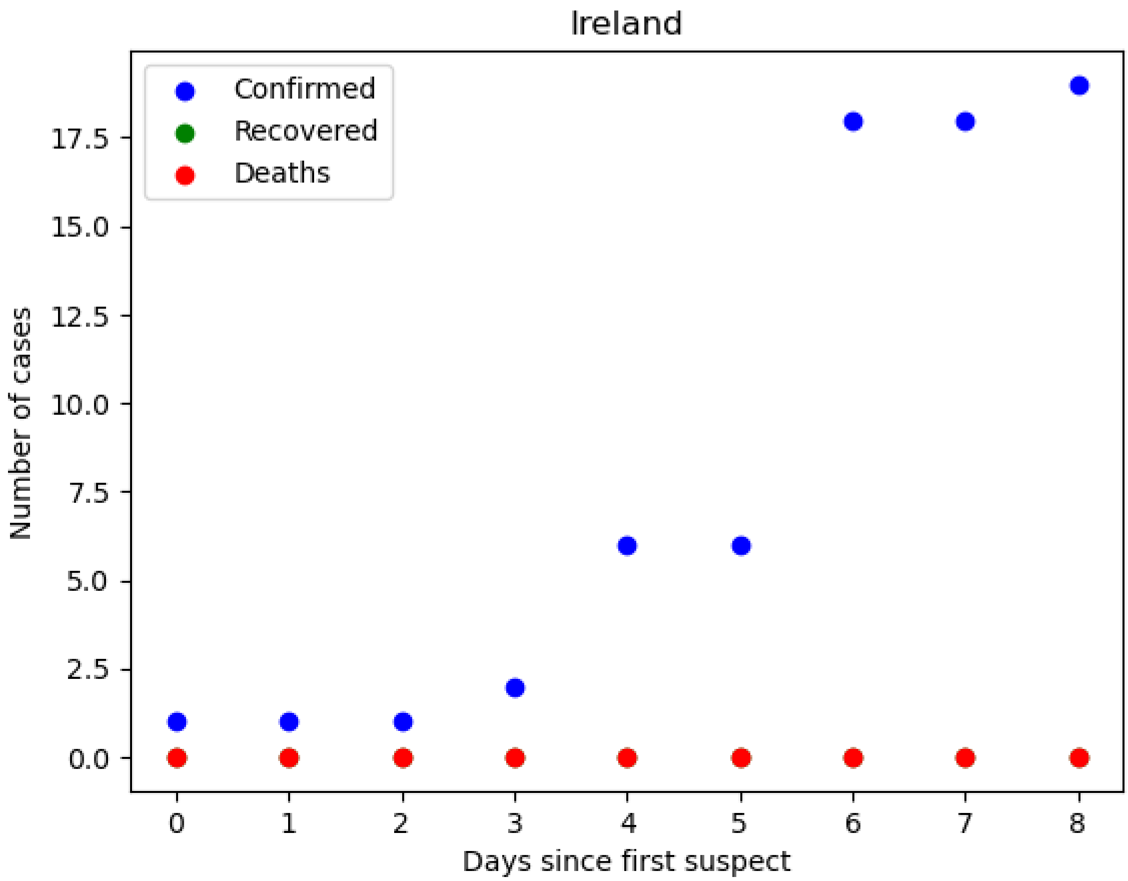

Iceland
Confirmed
Recovered
Deaths
Number of cases
0
10
20
30
40
50
0
2
4
6
8
Days since first suspect

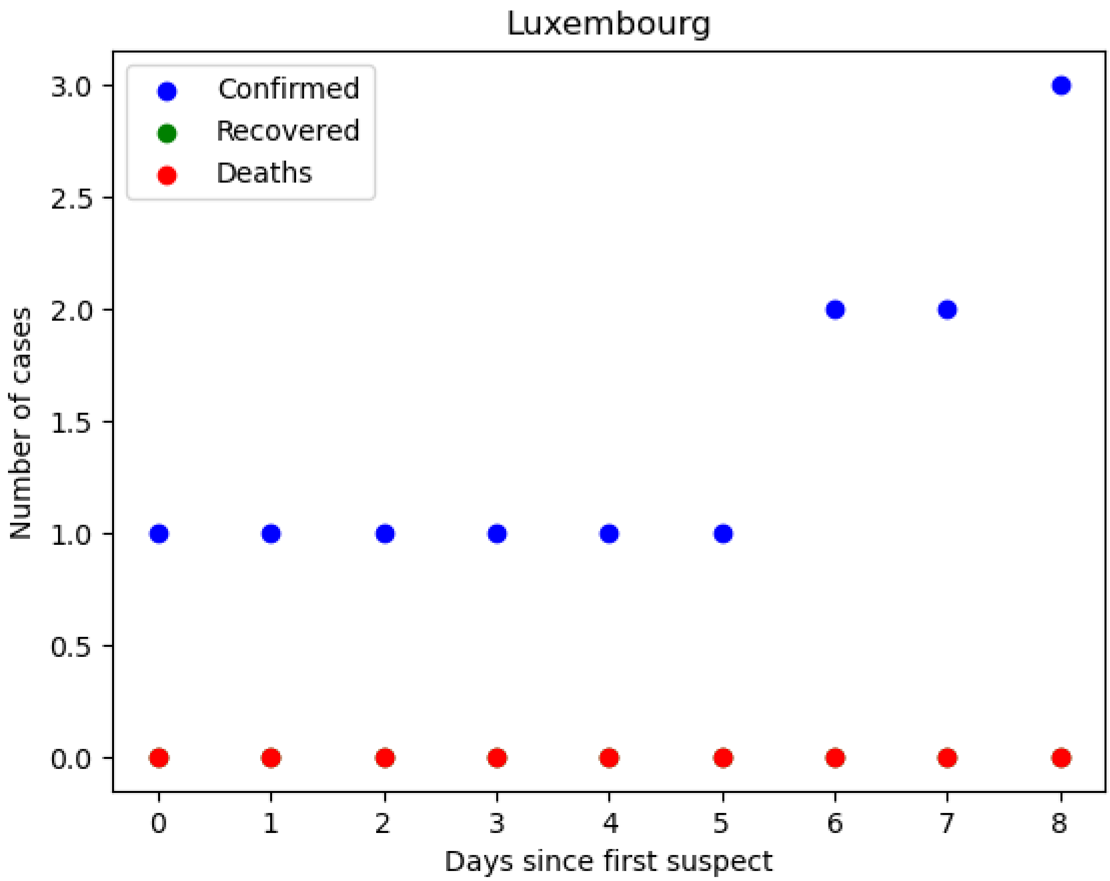

Lithuania
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0
2
4
6
8
Days since first suspect
Confirmed
Recovered
Deaths

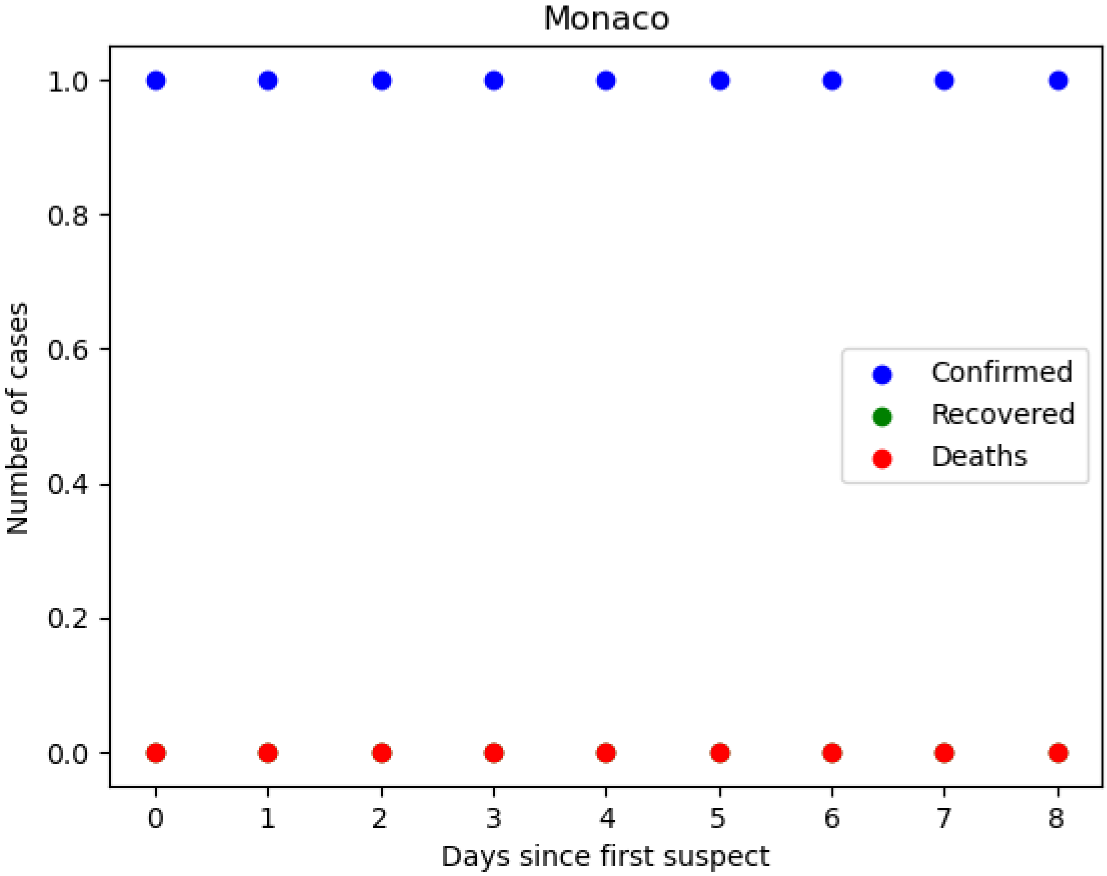

New Zealand
5
4
3
2
1
0
Number of cases
0
2
4
6
8
Days since first suspect
Confirmed
Recovered
Deaths

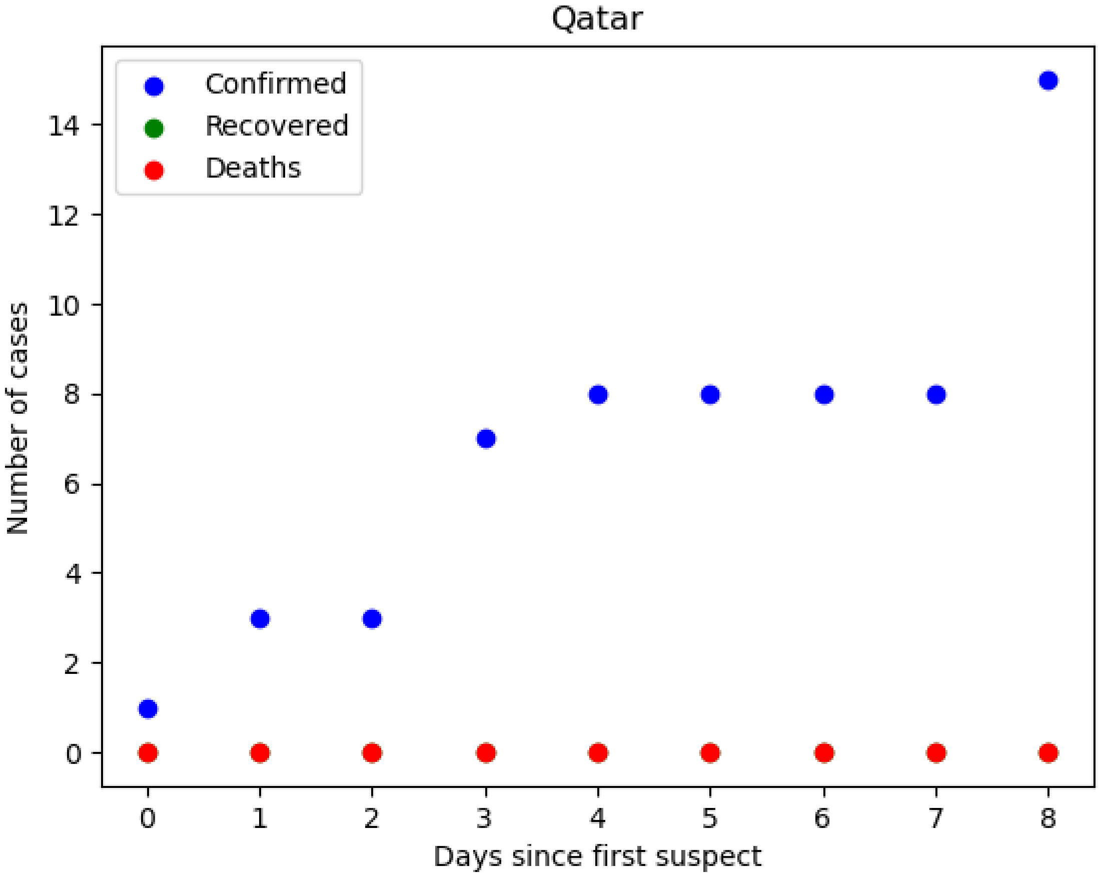

Nigeria
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
Days since first suspect
● Confirmed
● Recovered
● Deaths

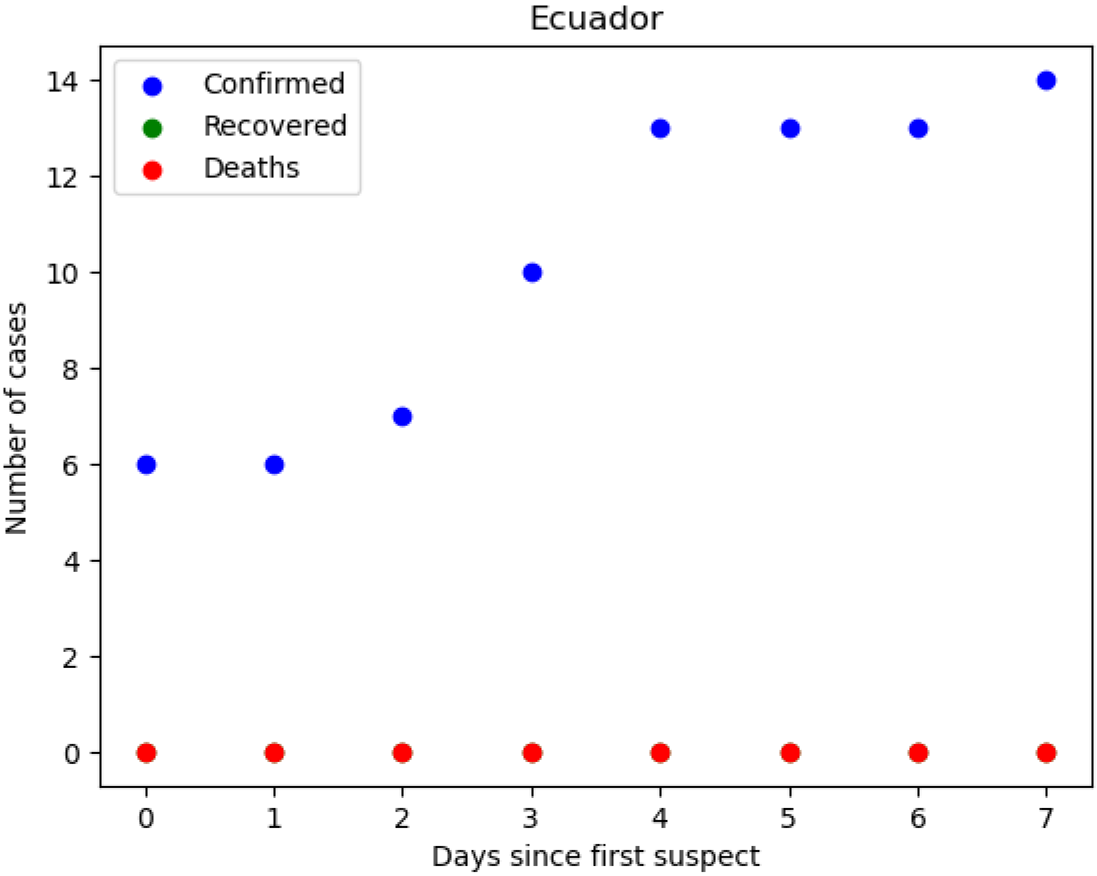

North Ireland
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect
● Confirmed
● Recovered
● Deaths

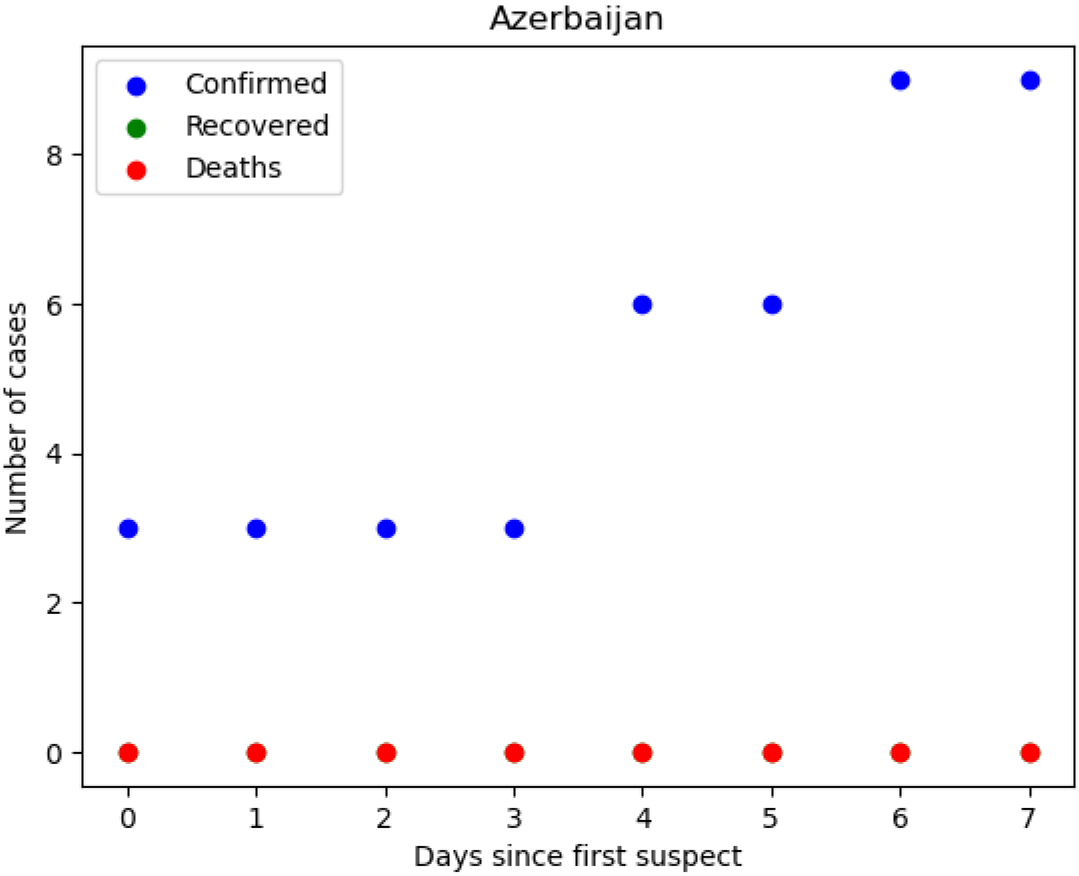

Ireland
Confirmed
Recovered
Deaths
Number of cases
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1
2
3
4
5
6
7
8
Days since first suspect

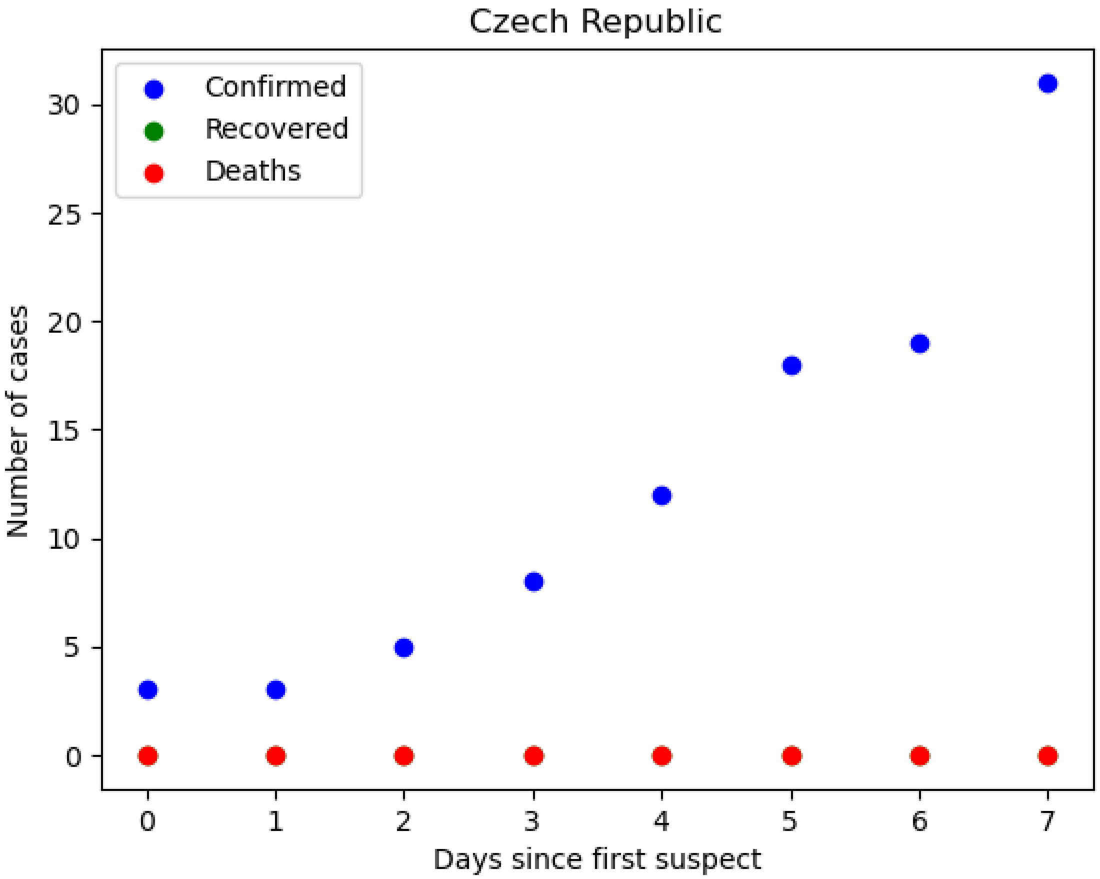

Luxembourg
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5
6
7
8
Days since first suspect

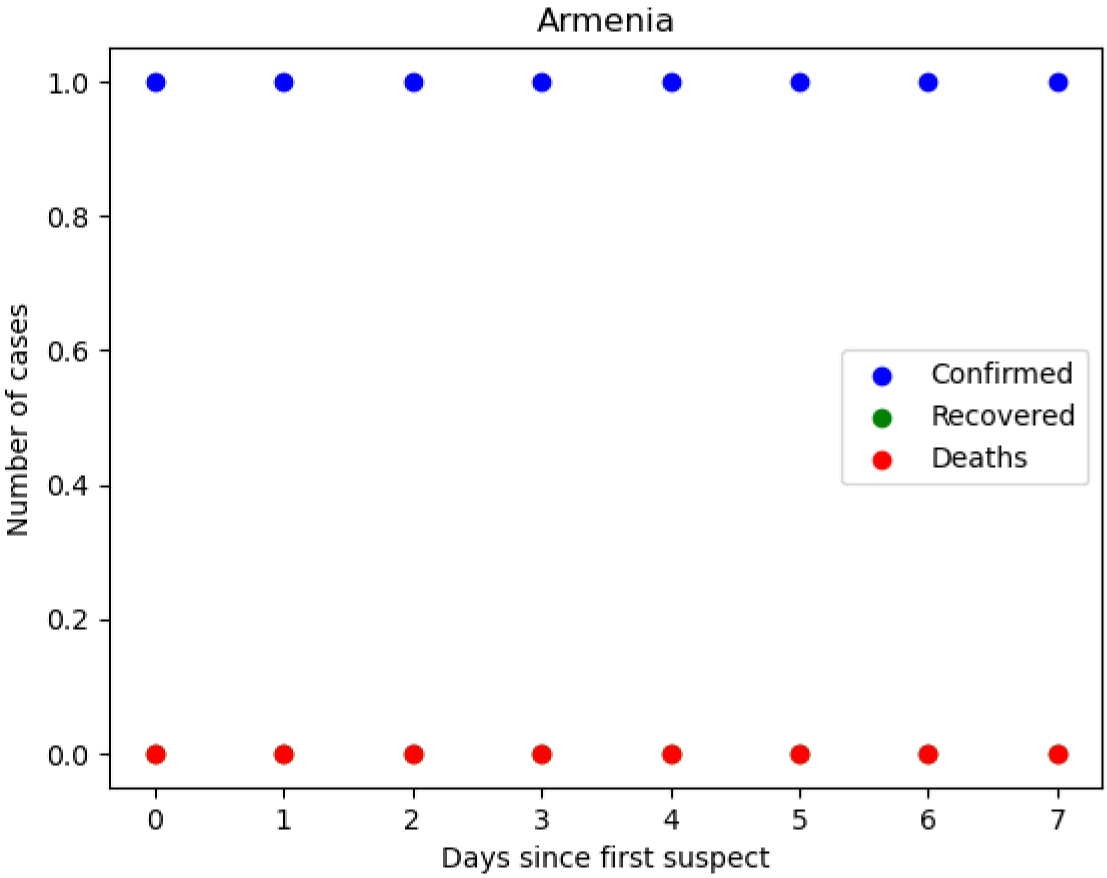

Monaco
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
Confirmed
Recovered
Deaths
0
1
2
3
4
5
6
7
8
Days since first suspect

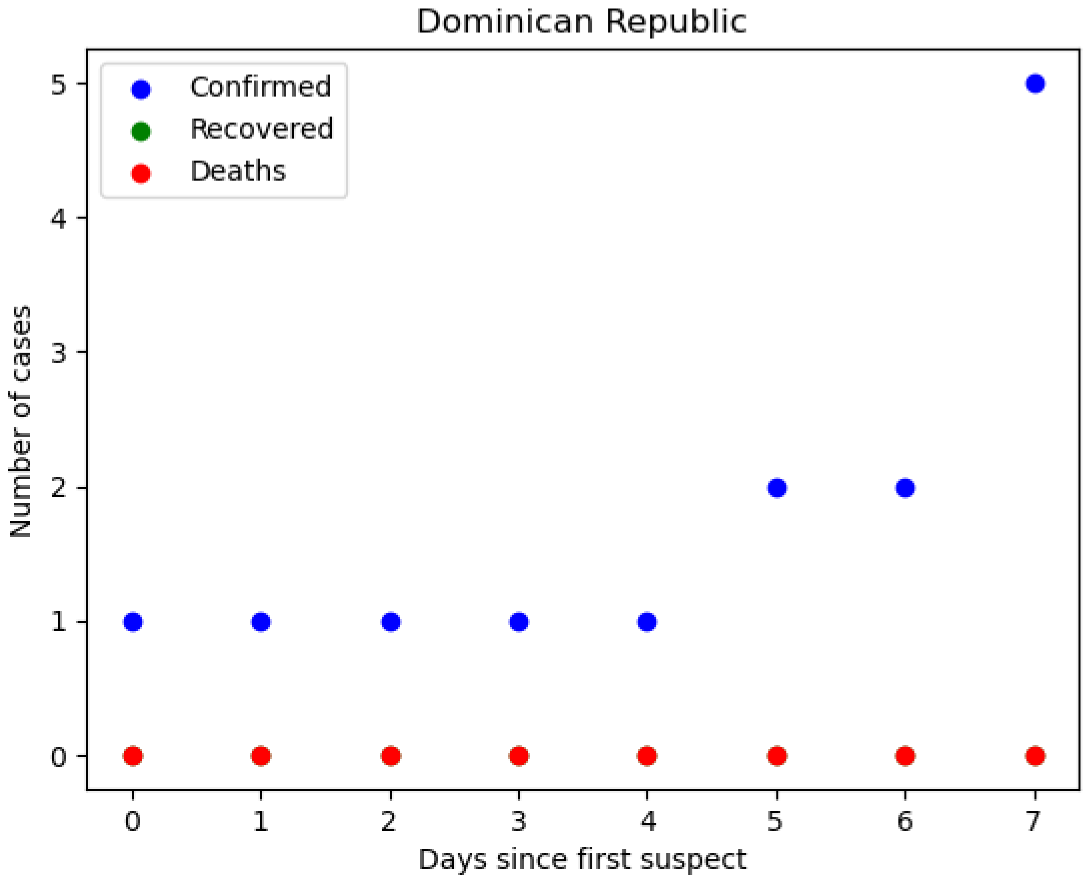

Qatar
Number of cases
14
12
10
8
6
4
2
0
Confirmed
Recovered
Deaths
0
1
2
3
4
5
6
7
8
Days since first suspect

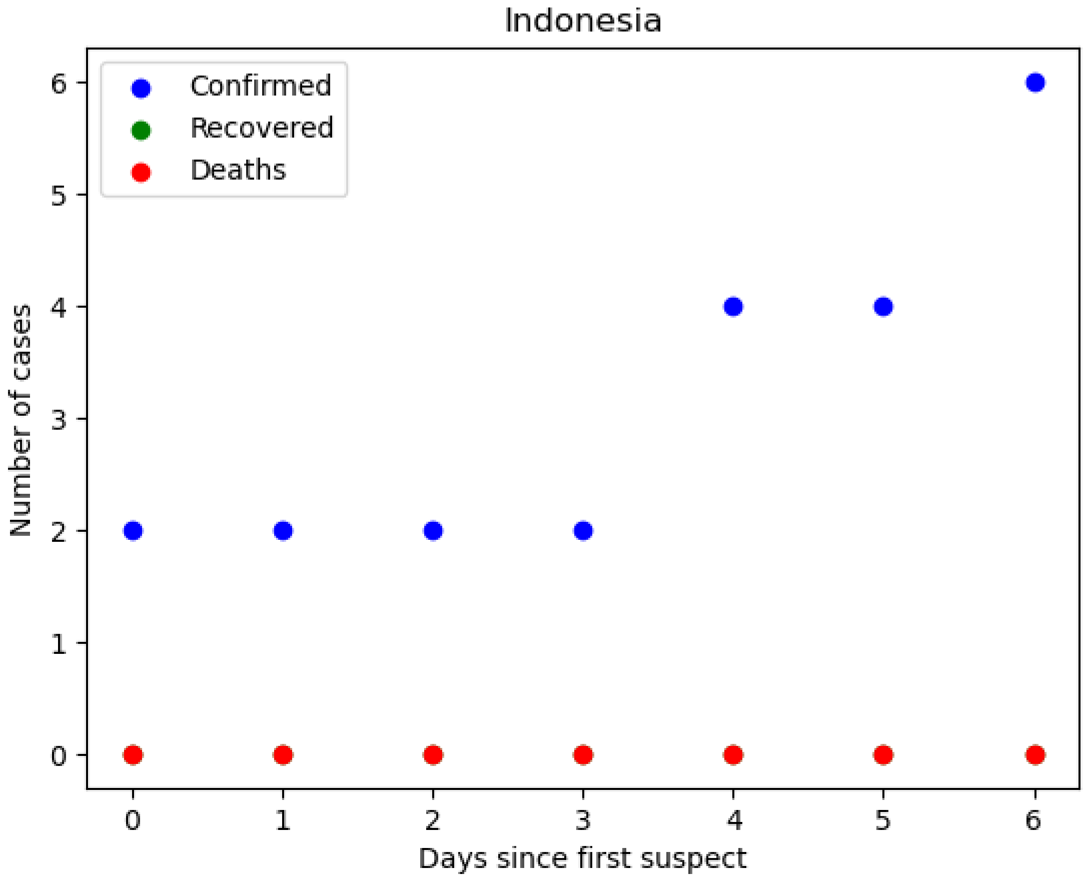

Ecuador
Confirmed
Recovered
Deaths
Number of cases
14
12
10
8
6
4
2
0
0
1
2
3
4
5
6
7
Days since first suspect

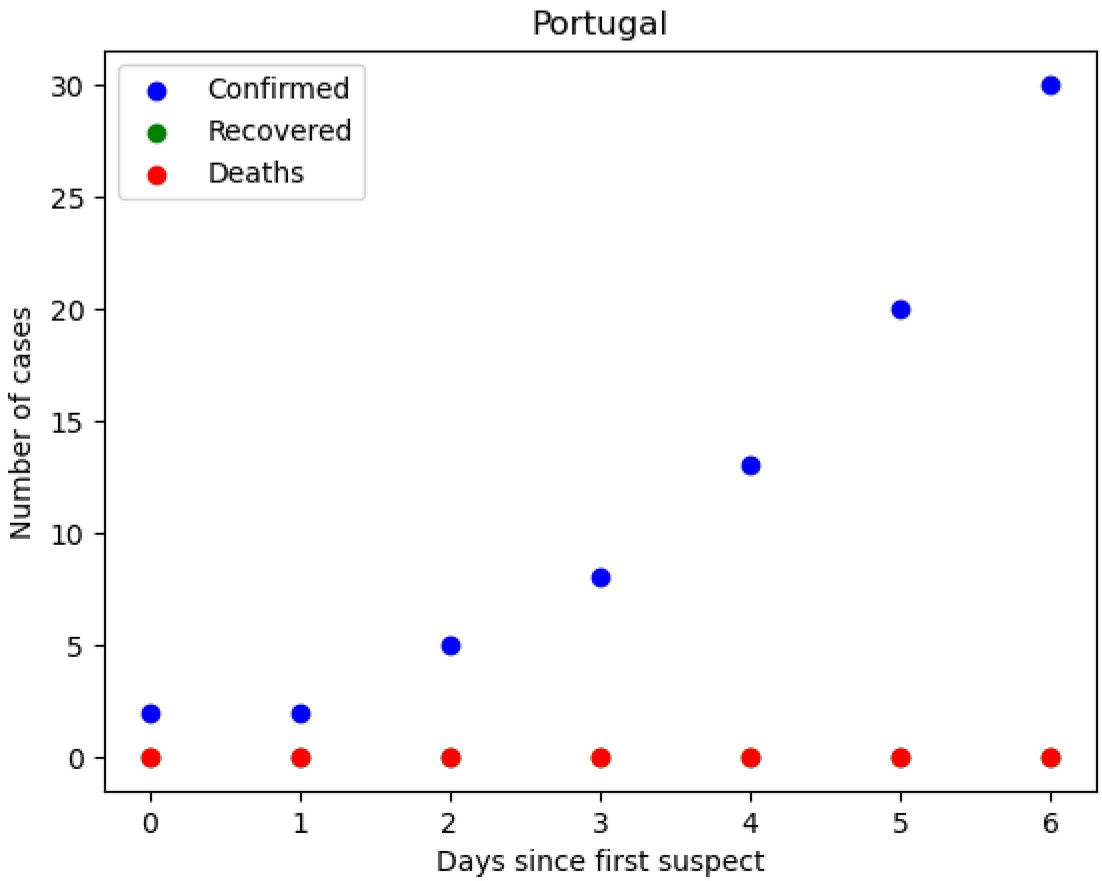

Azerbaijan
Confirmed
Recovered
Deaths
Number of cases
8
6
4
2
0
0
1
2
3
4
5
6
7
Days since first suspect

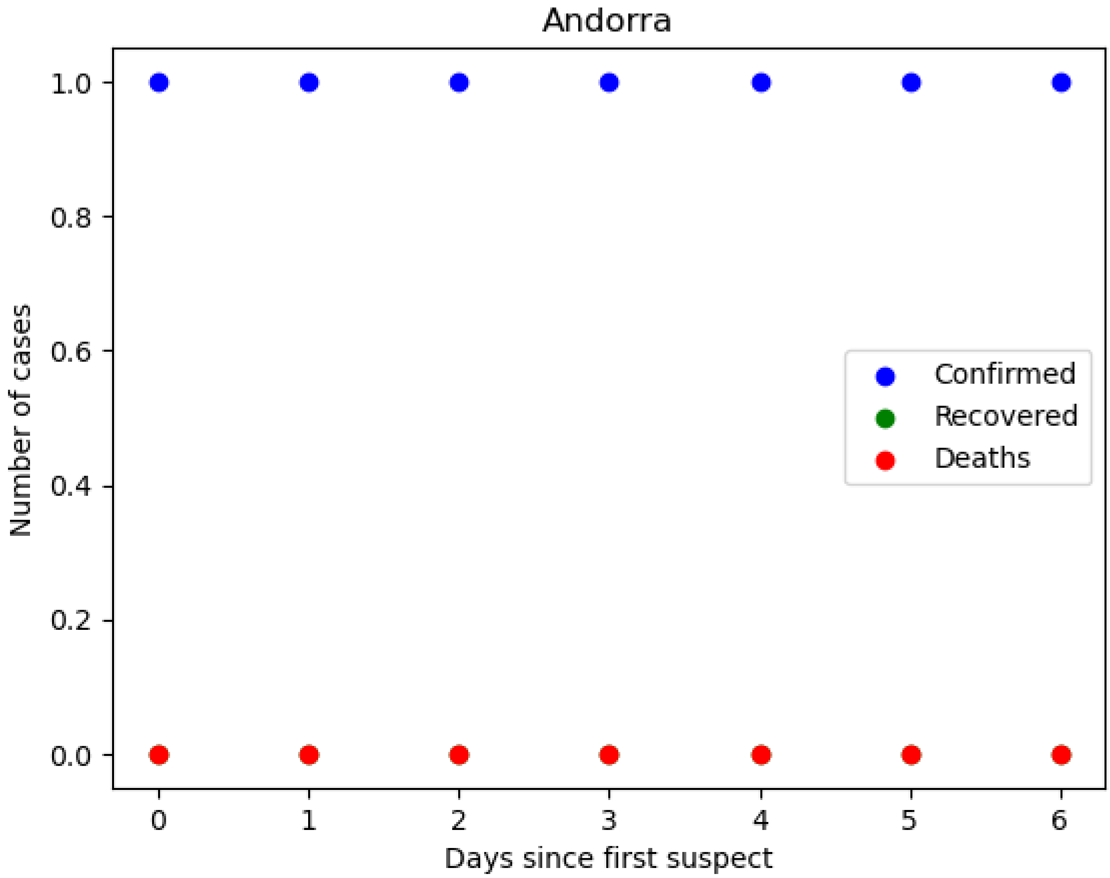

Czech Republic
Confirmed
Recovered
Deaths
Number of cases
0
5
10
15
20
25
30
0
1
2
3
4
5
6
7
Days since first suspect

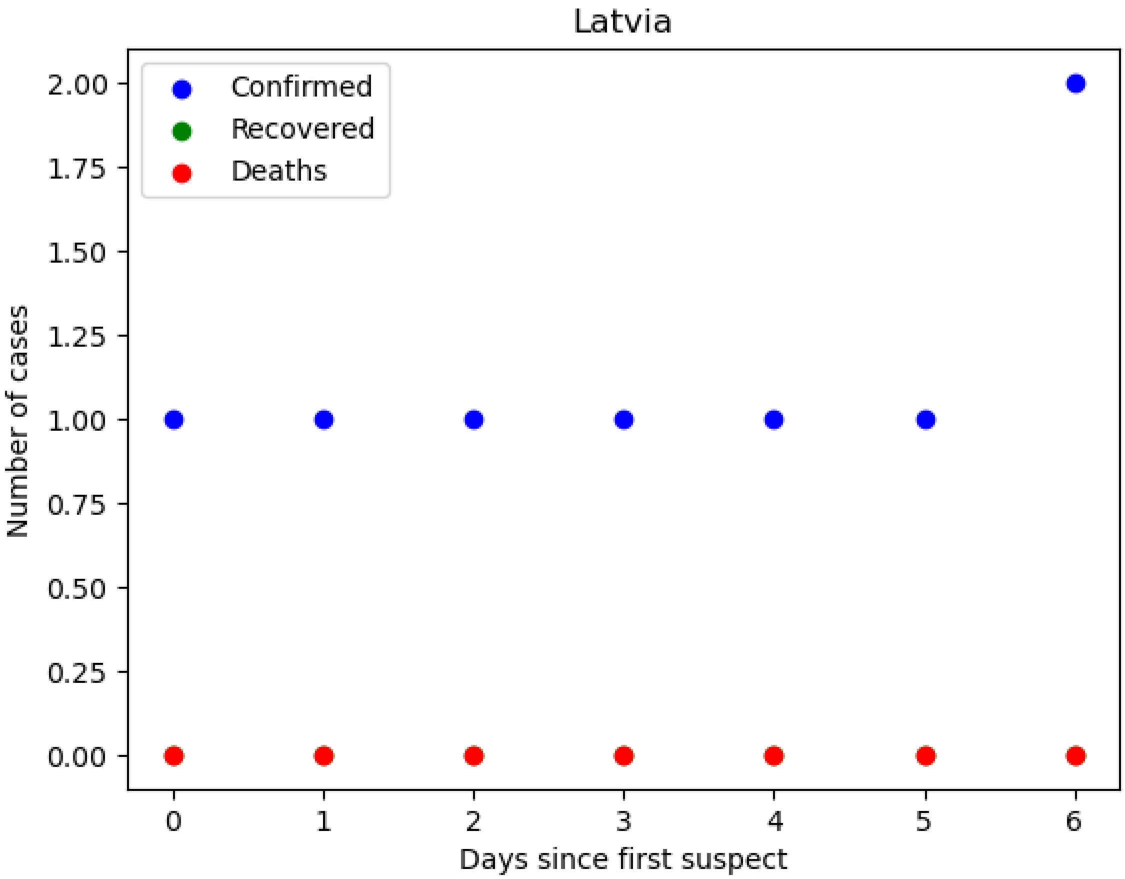

Armenia
Confirmed
Recovered
Deaths
Number of cases
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
7
Days since first suspect

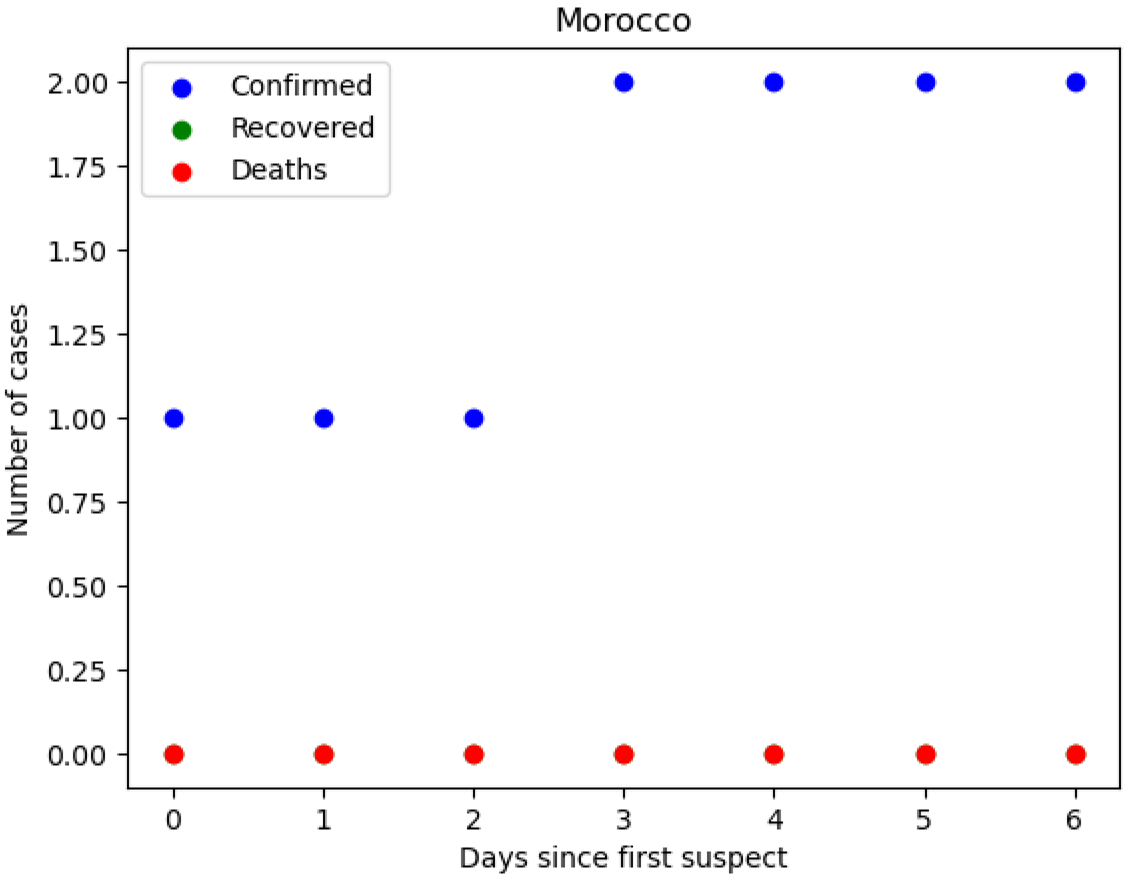

Dominican Republic
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
Days since first suspect
0
1
2
3
4
5
6
7

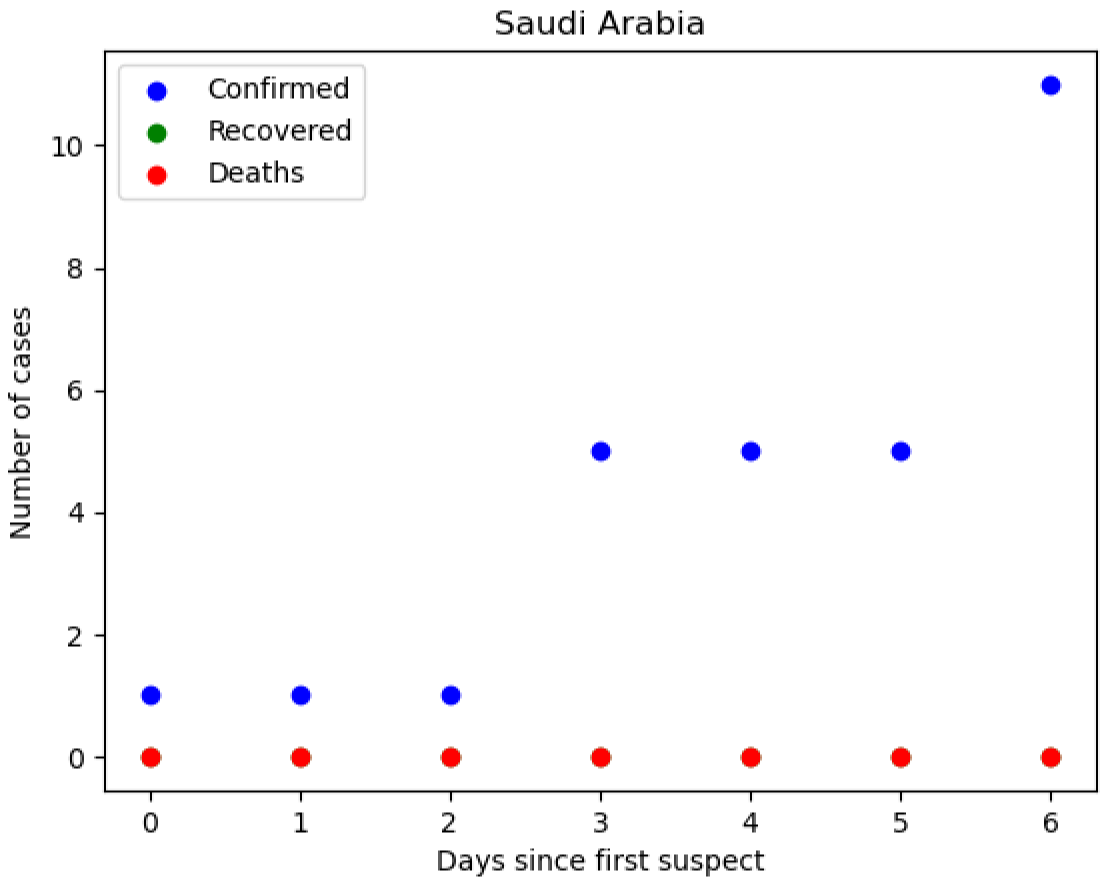

Indonesia
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
6
Days since first suspect
0
1
2
3
4
5
6

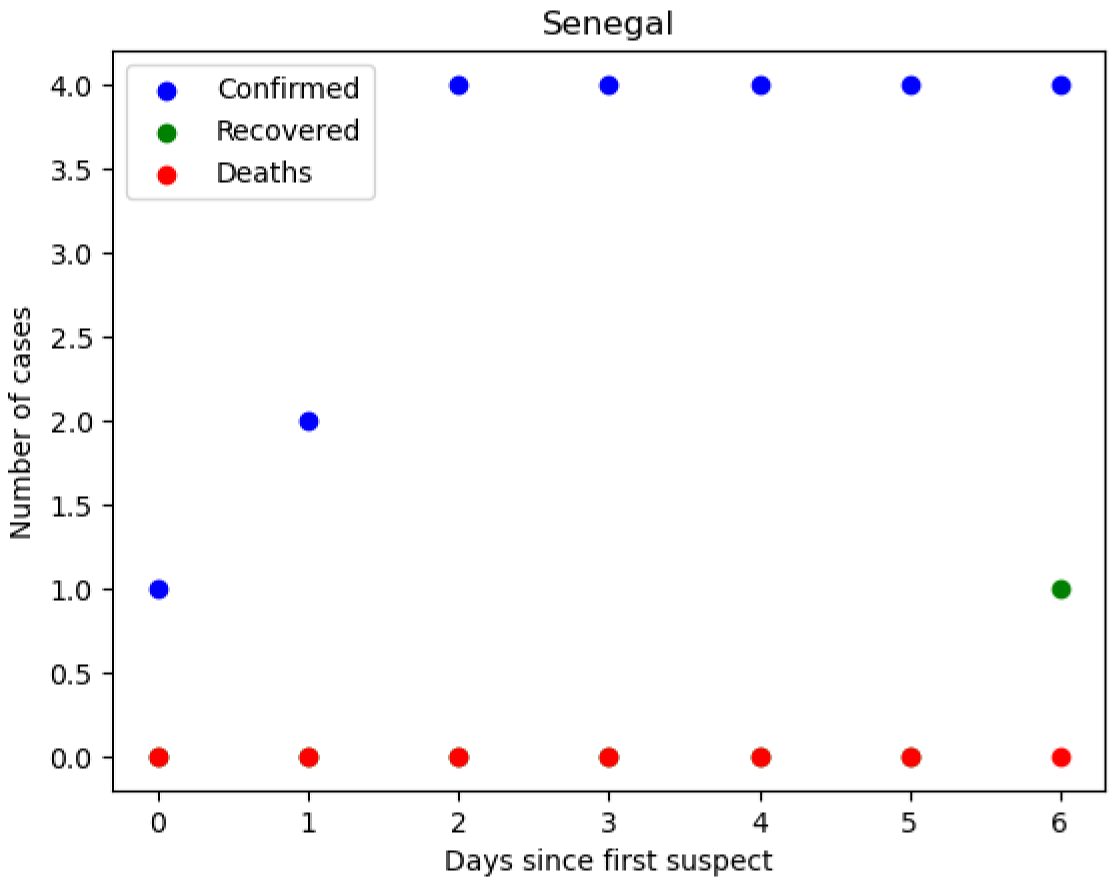

Portugal
● Confirmed
● Recovered
● Deaths
Number of cases
30
25
20
15
10
5
0
0
1
2
3
4
5
6
Days since first suspect

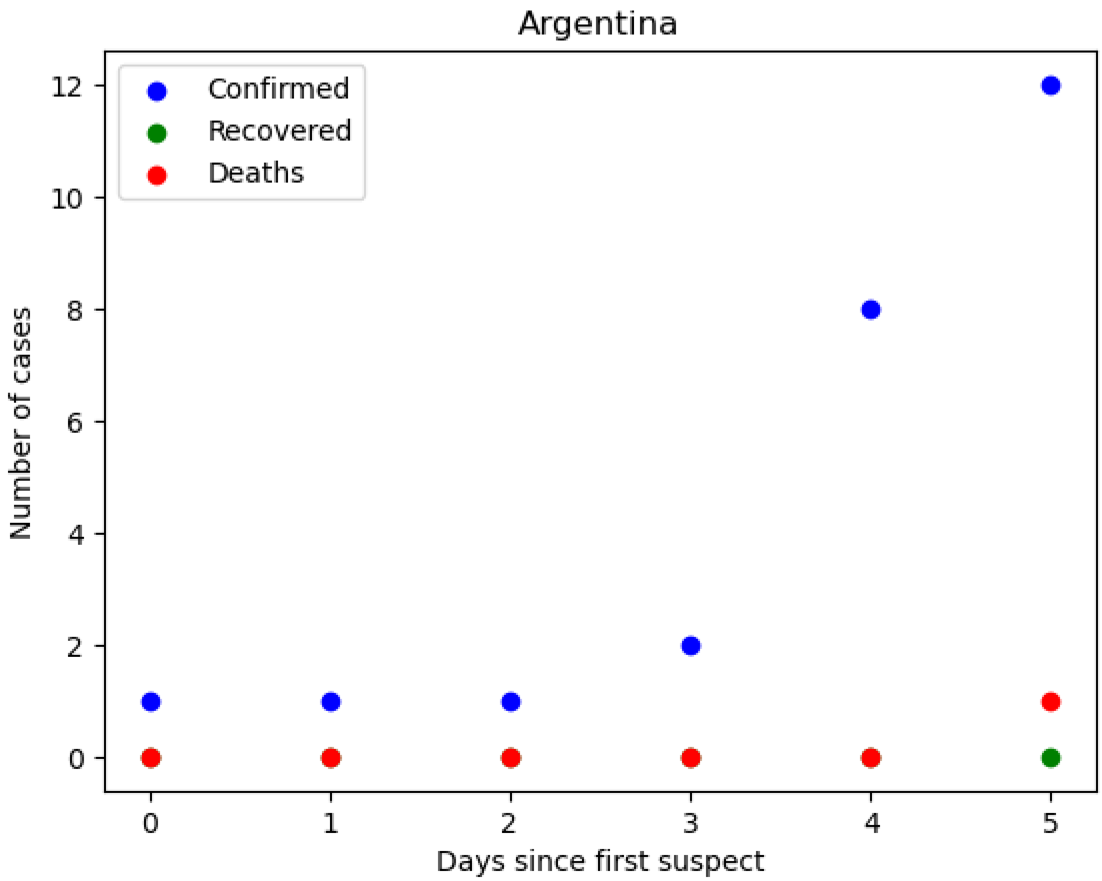

Andorra
● Confirmed
● Recovered
● Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
Days since first suspect

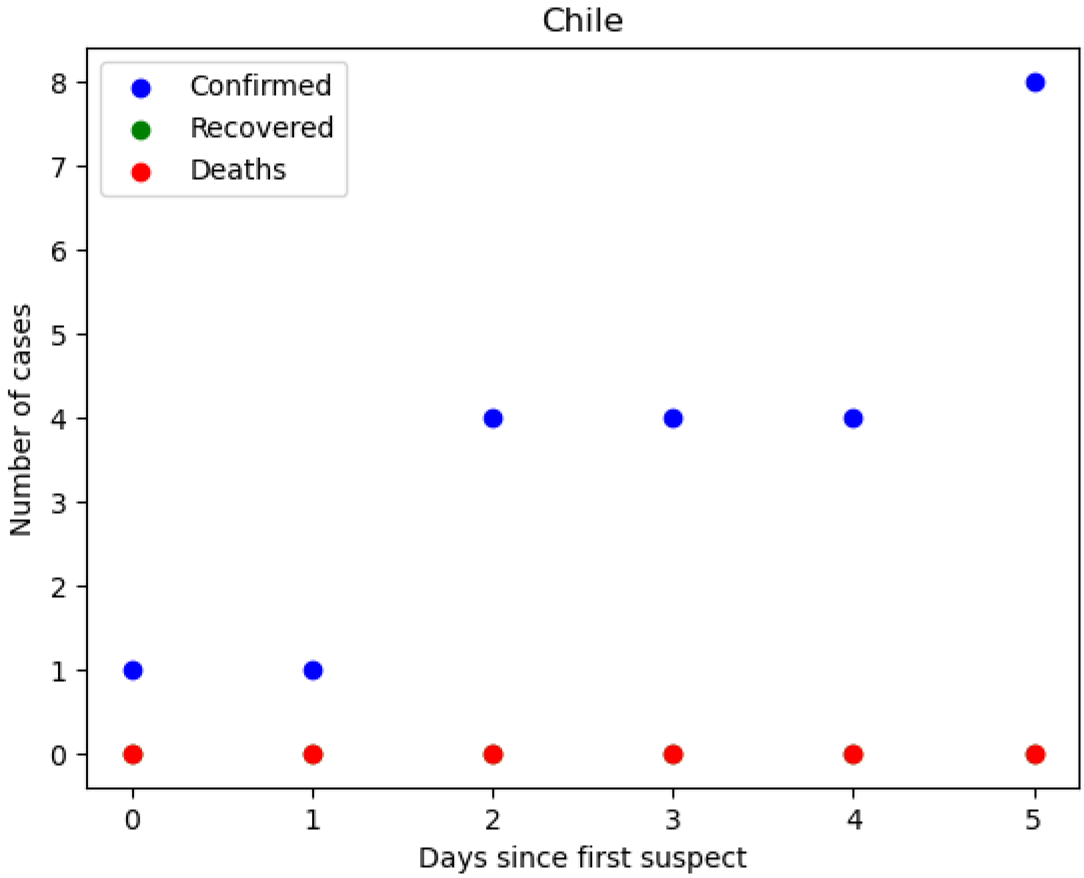

Latvia
Confirmed
Recovered
Deaths
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of cases
0
1
2
3
4
5
6
Days since first suspect

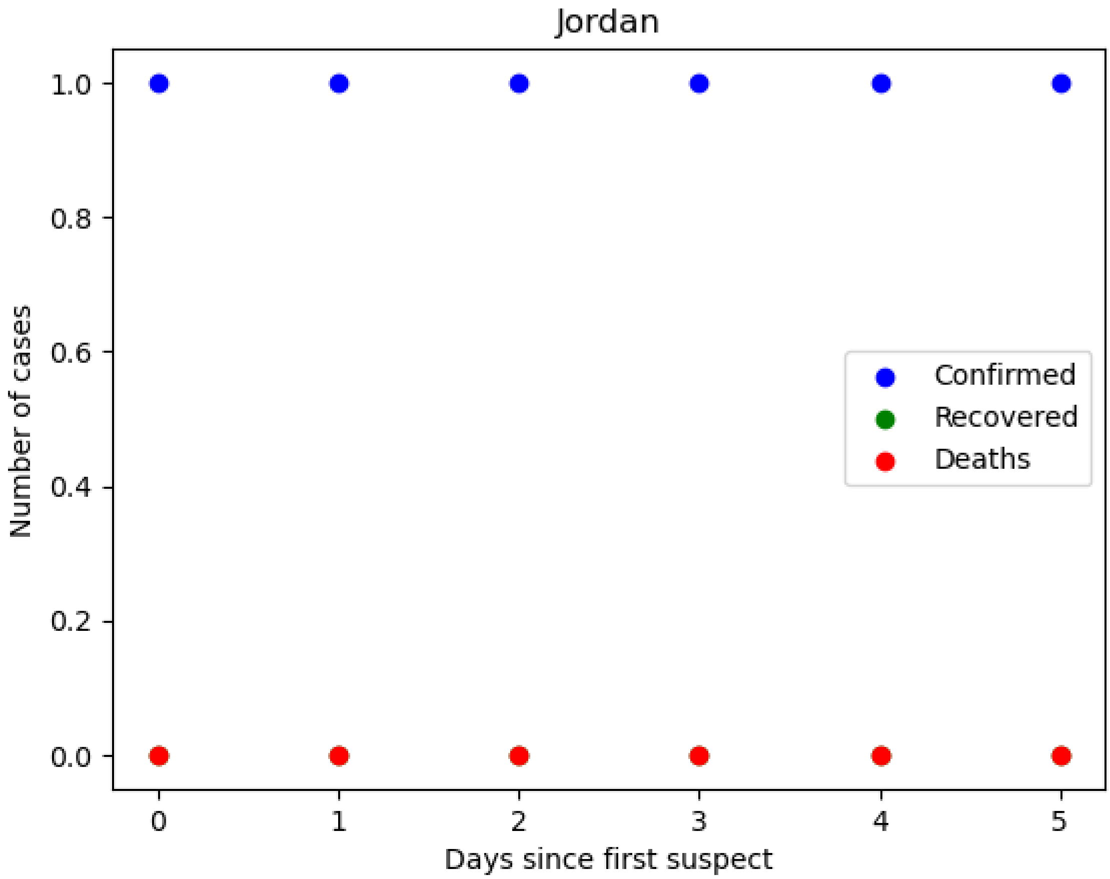

Morocco
Confirmed
Recovered
Deaths
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of cases
0
1
2
3
4
5
6
Days since first suspect

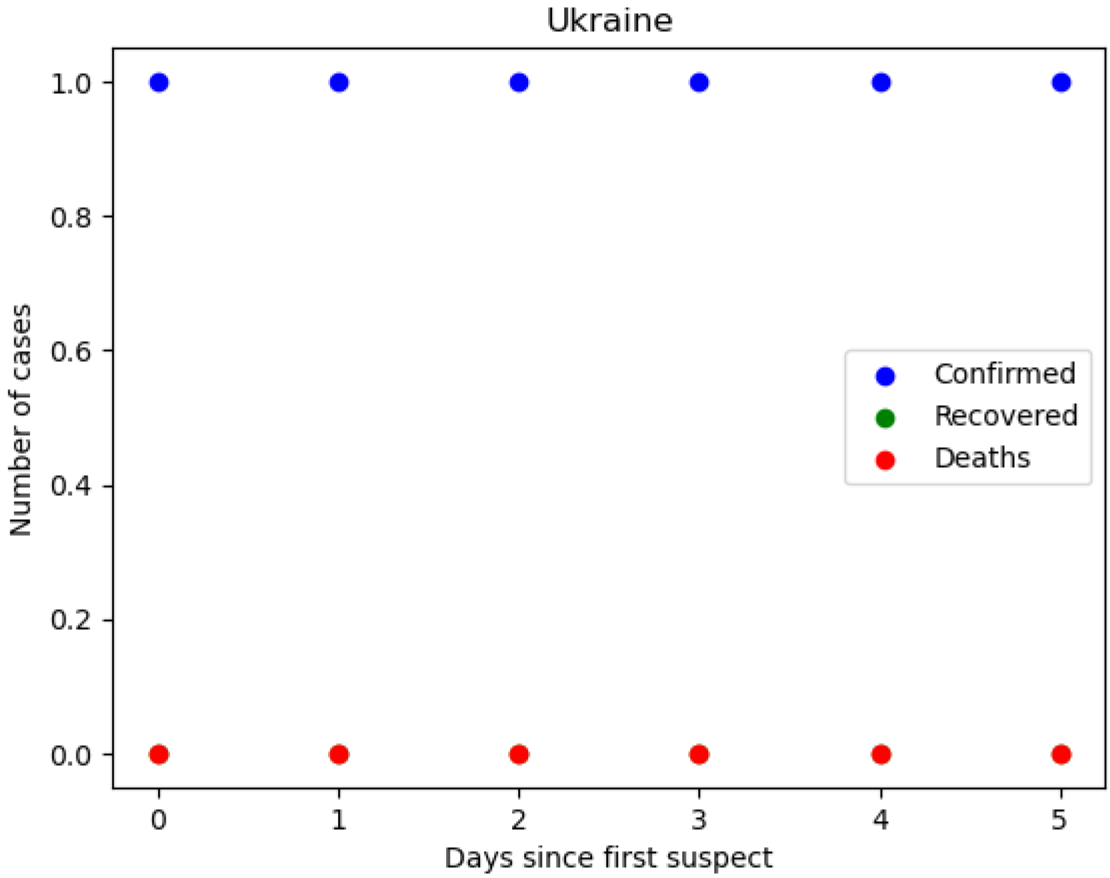

Saudi Arabia
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
0
1
2
3
4
5
6
Days since first suspect

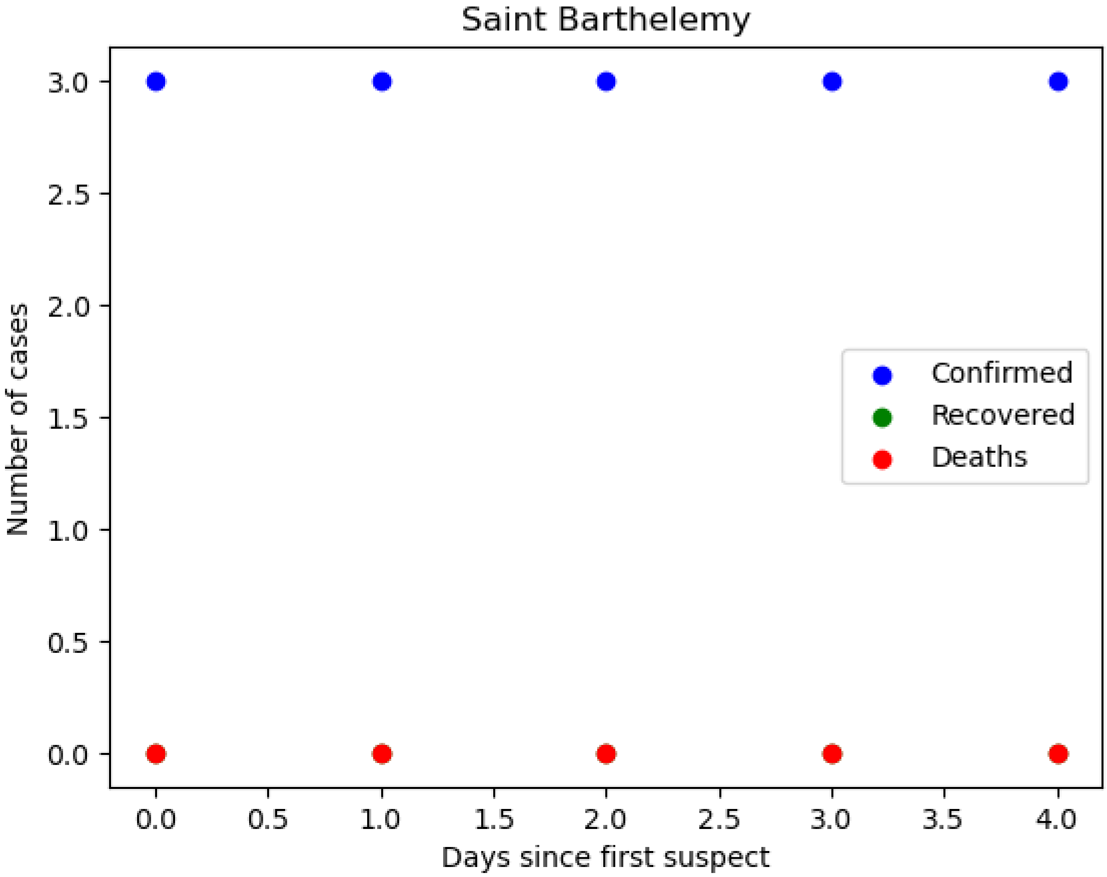

Senegal
Confirmed
Recovered
Deaths
Number of cases
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
1
2
3
4
5
6
Days since first suspect

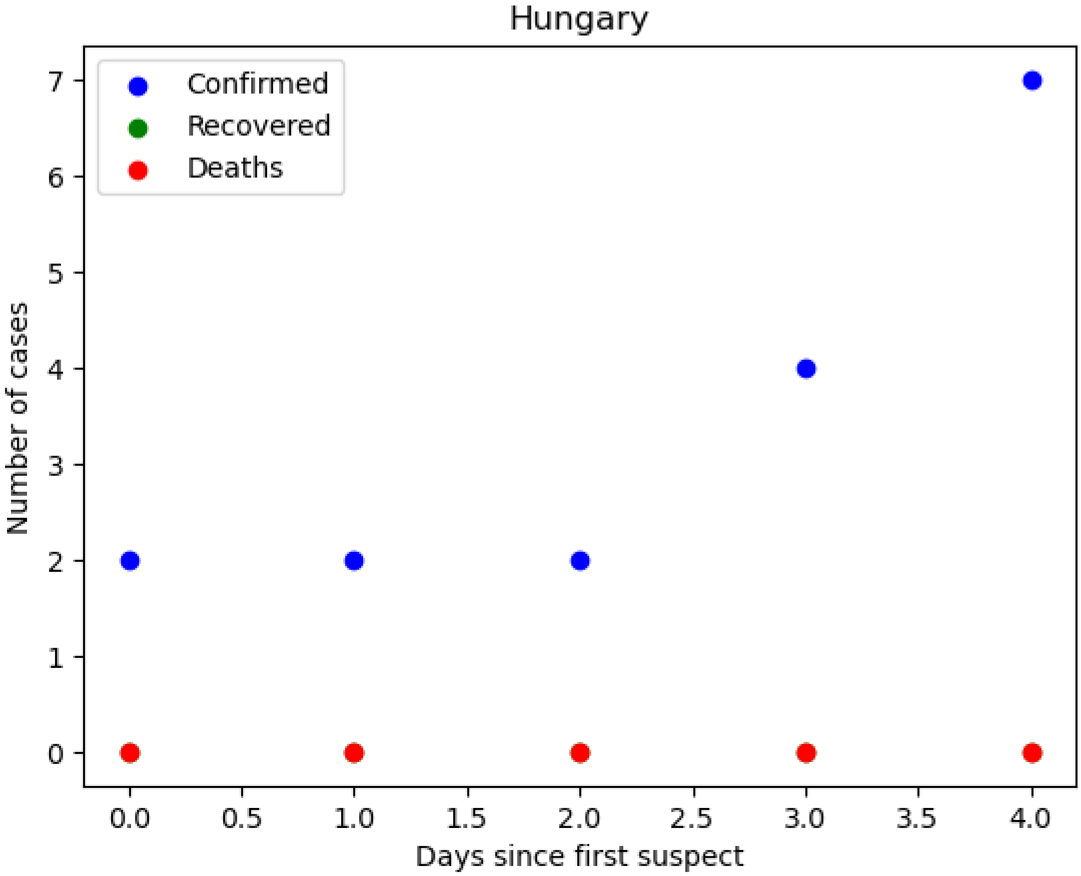

Argentina
Confirmed
Recovered
Deaths
Number of cases
0
2
4
6
8
10
12
0
1
2
3
4
5
Days since first suspect

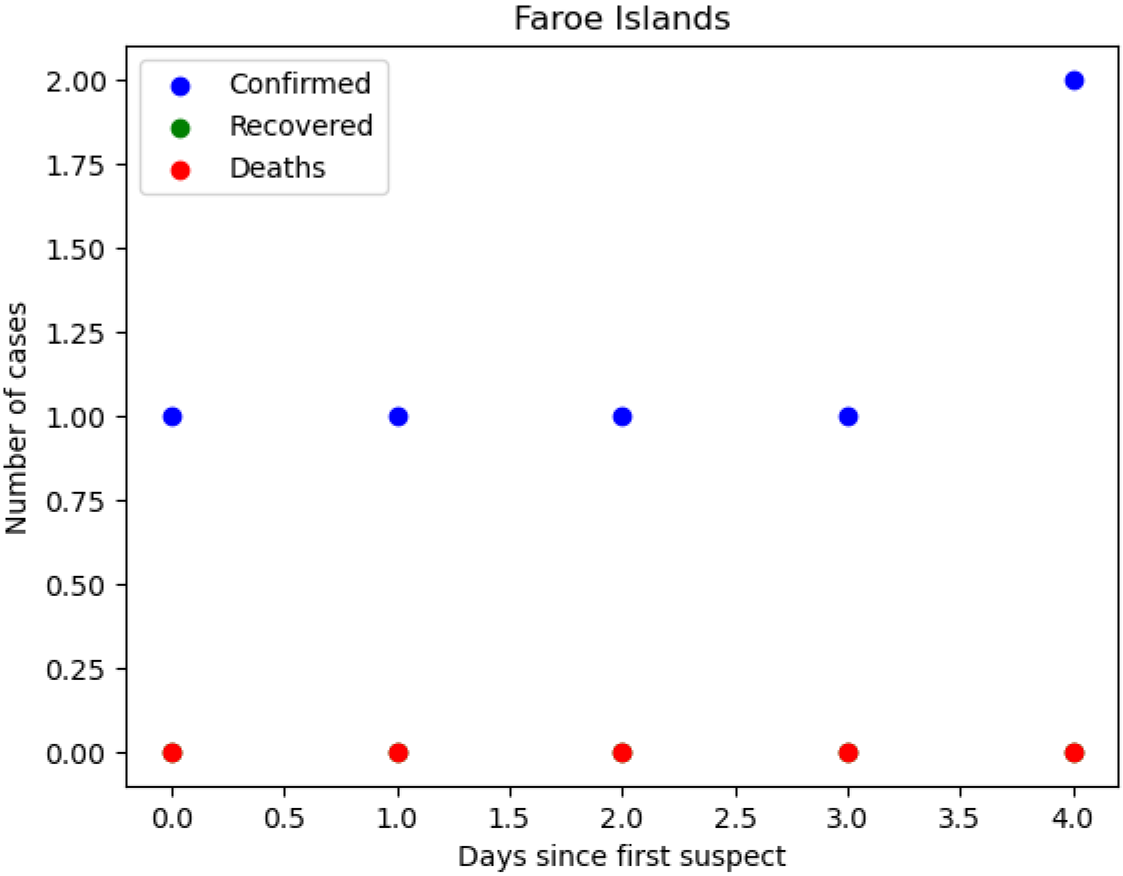

Chile
Confirmed
Recovered
Deaths
Number of cases
0
1
2
3
4
5
6
7
8
0
1
2
3
4
5
Days since first suspect

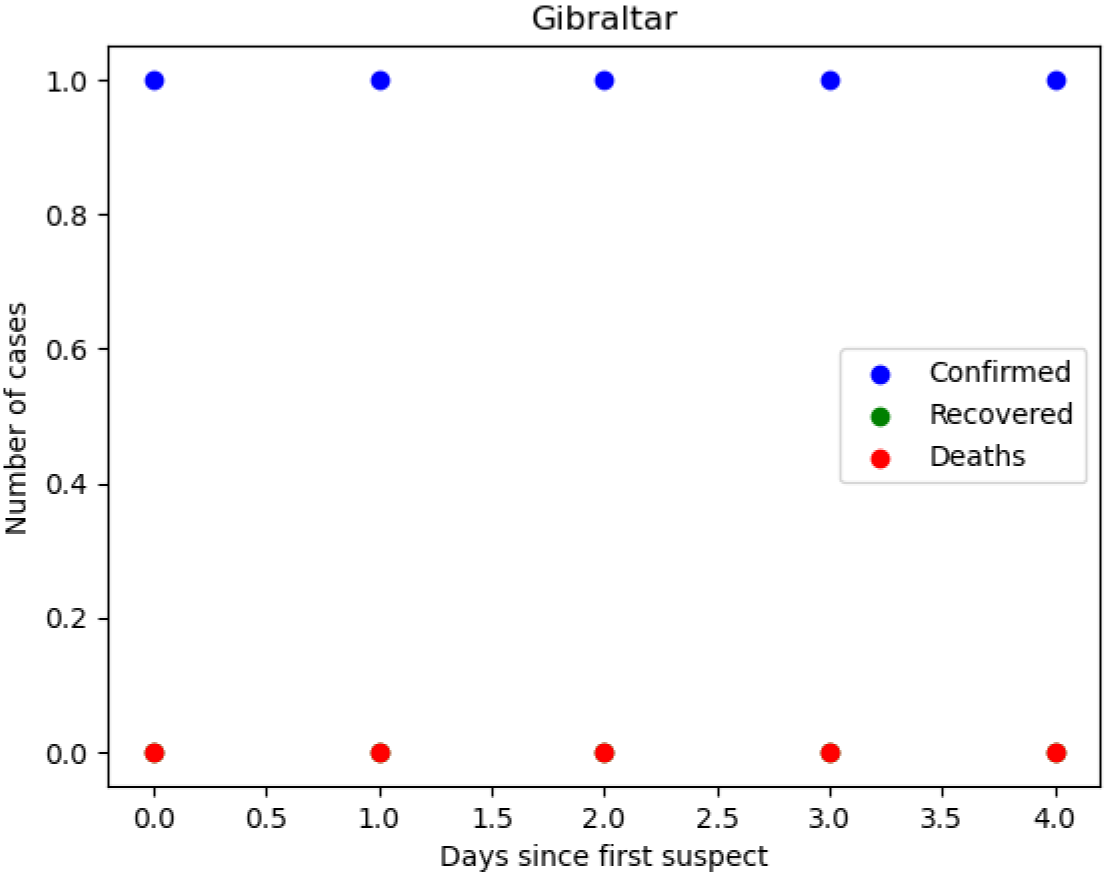

Jordan
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0
1
2
3
4
5
Days since first suspect
● Confirmed
● Recovered
● Deaths

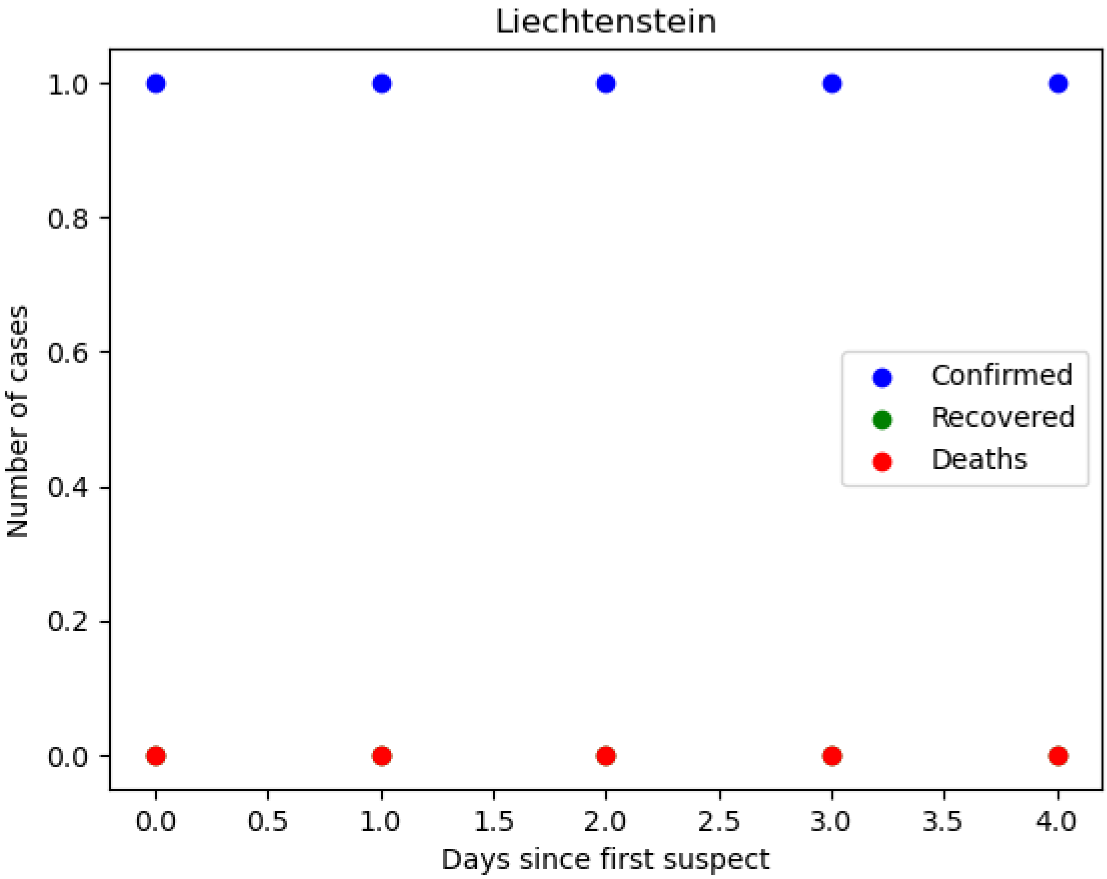

Ukraine
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0
1
2
3
4
5
Days since first suspect
● Confirmed
● Recovered
● Deaths

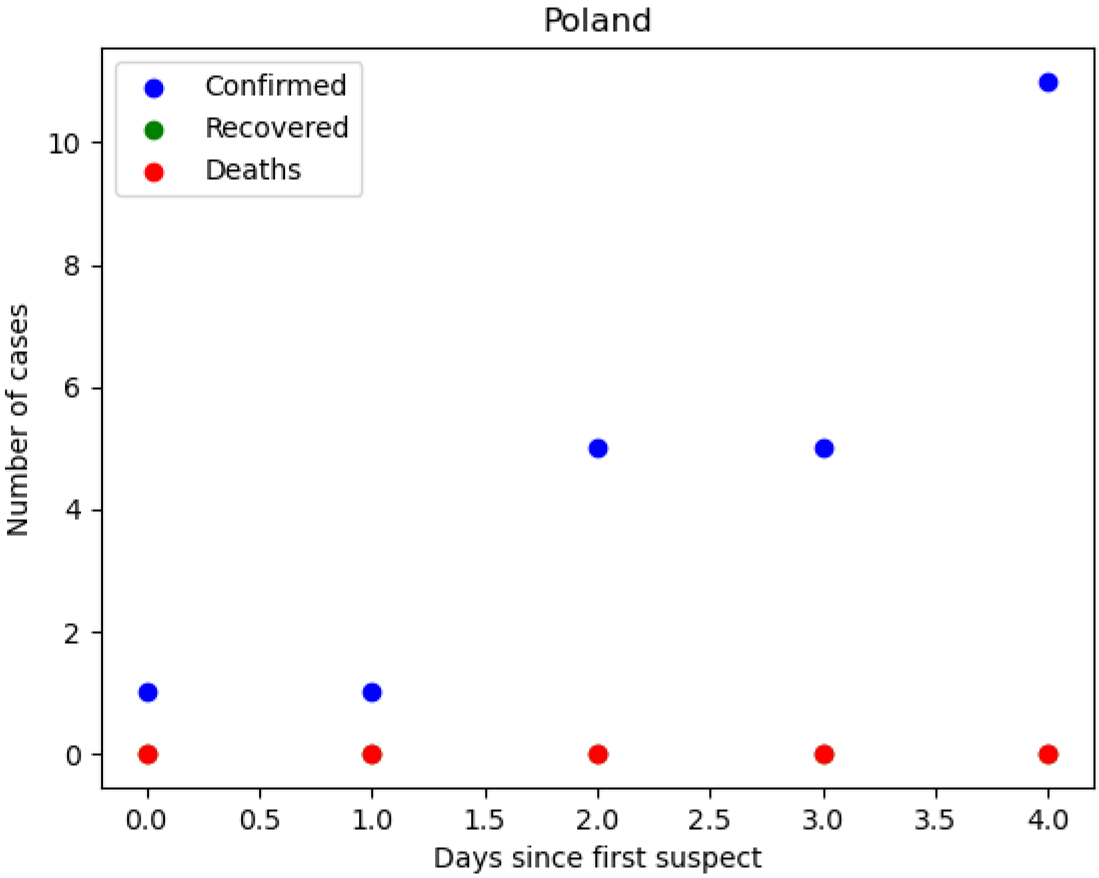

Saint Barthelemy
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Number of cases
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days since first suspect
Confirmed
Recovered
Deaths

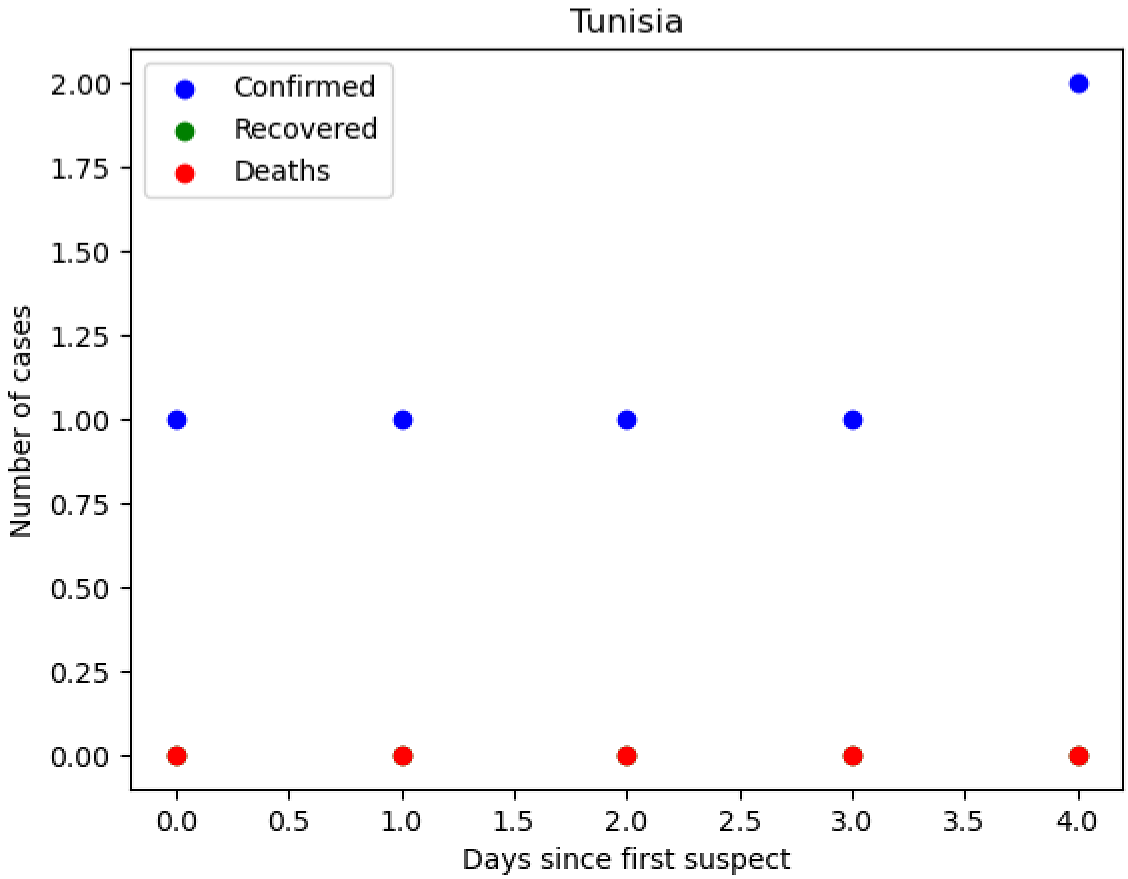

Hungary
7
6
5
4
3
2
1
0
Number of cases
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days since first suspect
Confirmed
Recovered
Deaths

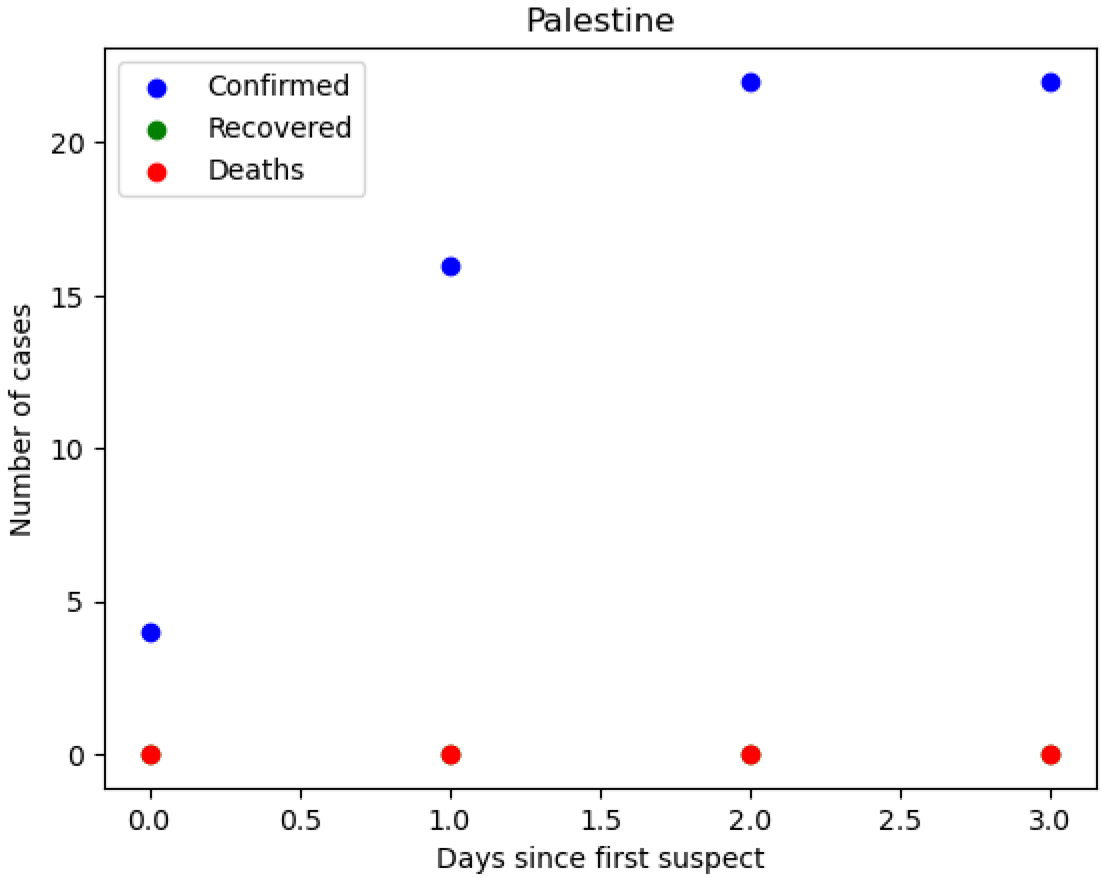

Faroe Islands
Confirmed
Recovered
Deaths
Number of cases
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days since first suspect

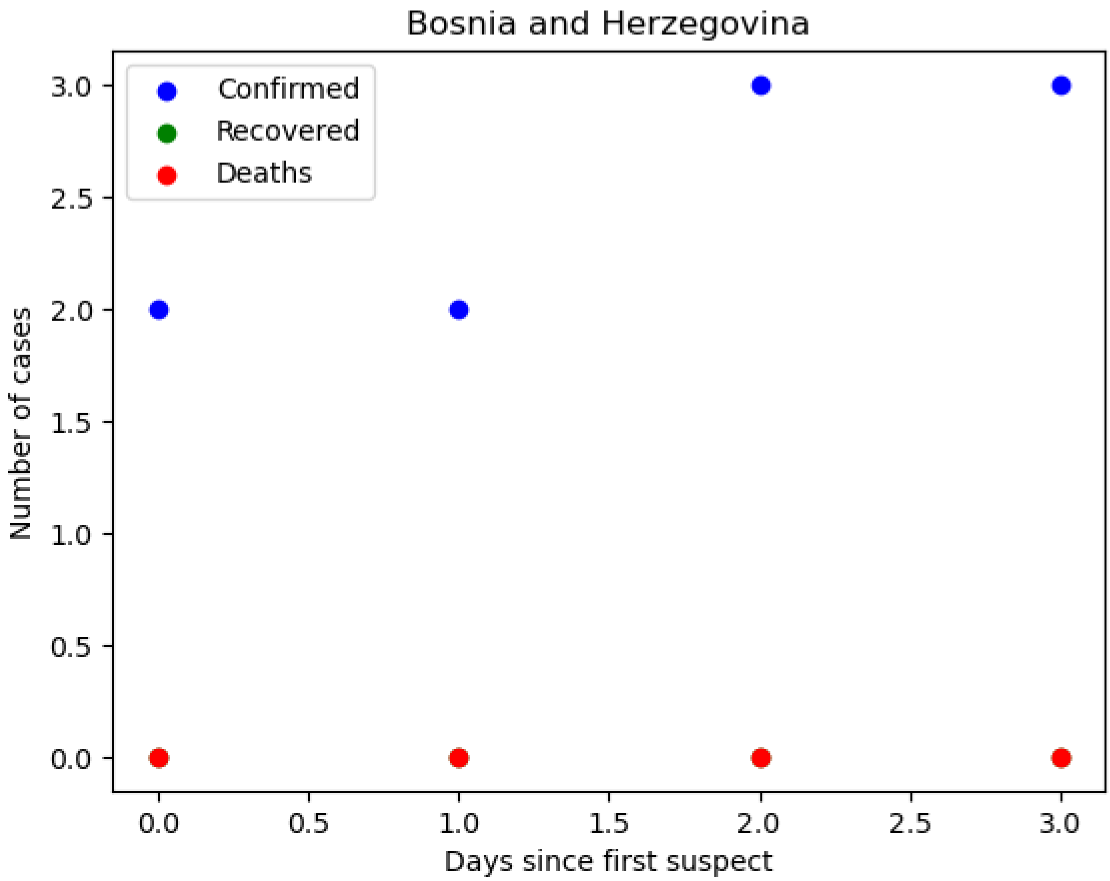

Gibraltar
Confirmed
Recovered
Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days since first suspect

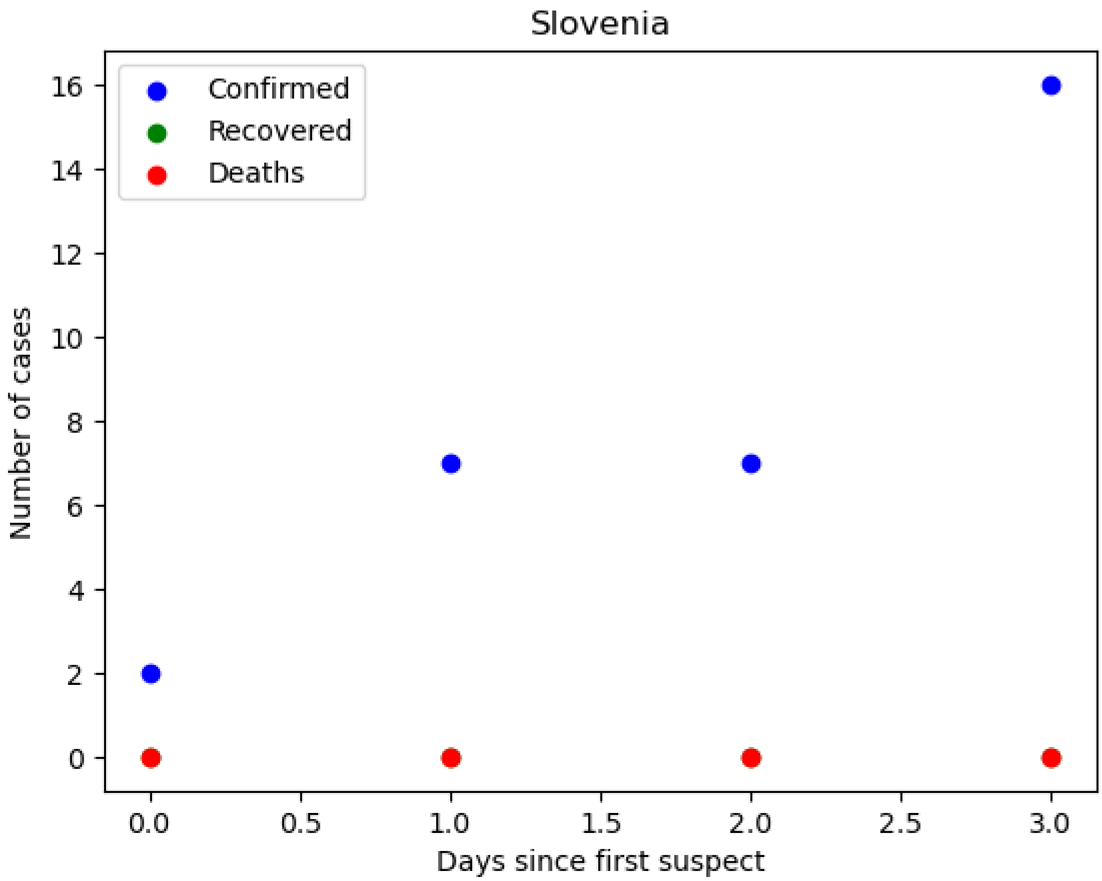

Liechtenstein
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
Days since first suspect
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
● Confirmed
● Recovered
● Deaths

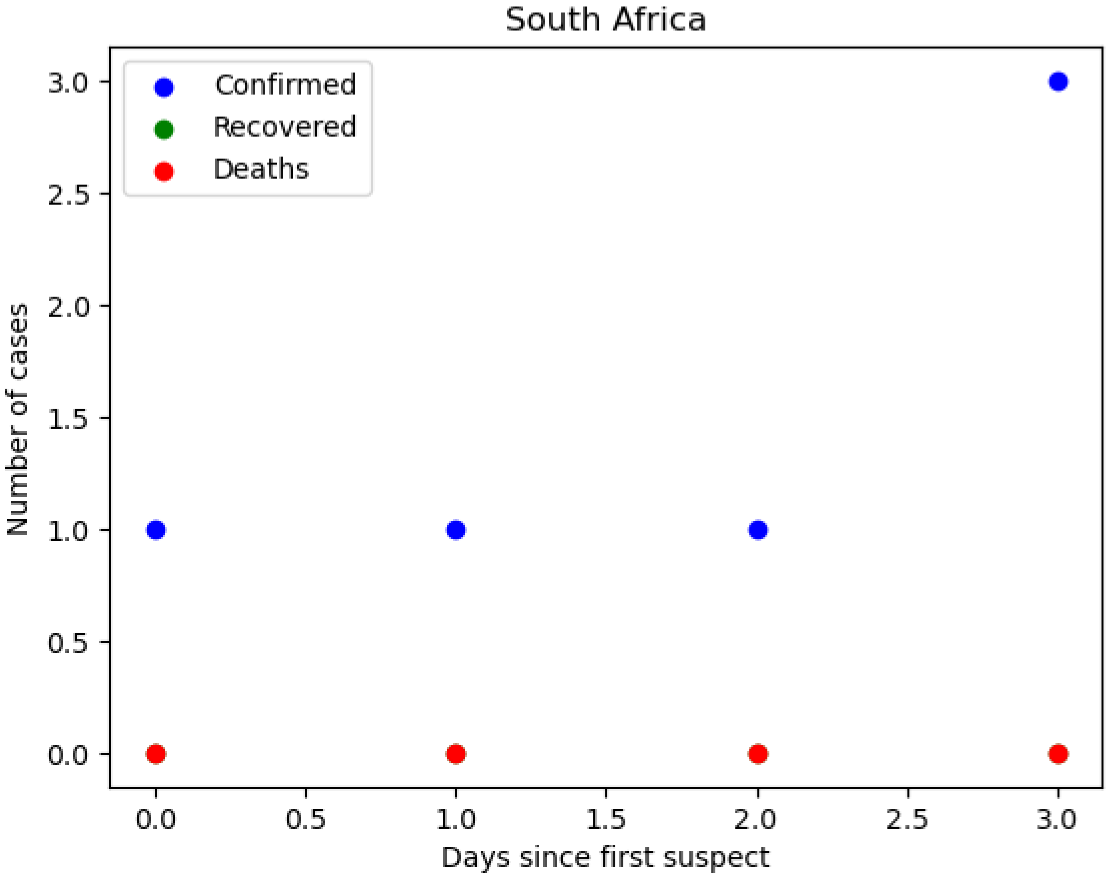

Poland
Number of cases
10
8
6
4
2
0
Days since first suspect
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
● Confirmed
● Recovered
● Deaths

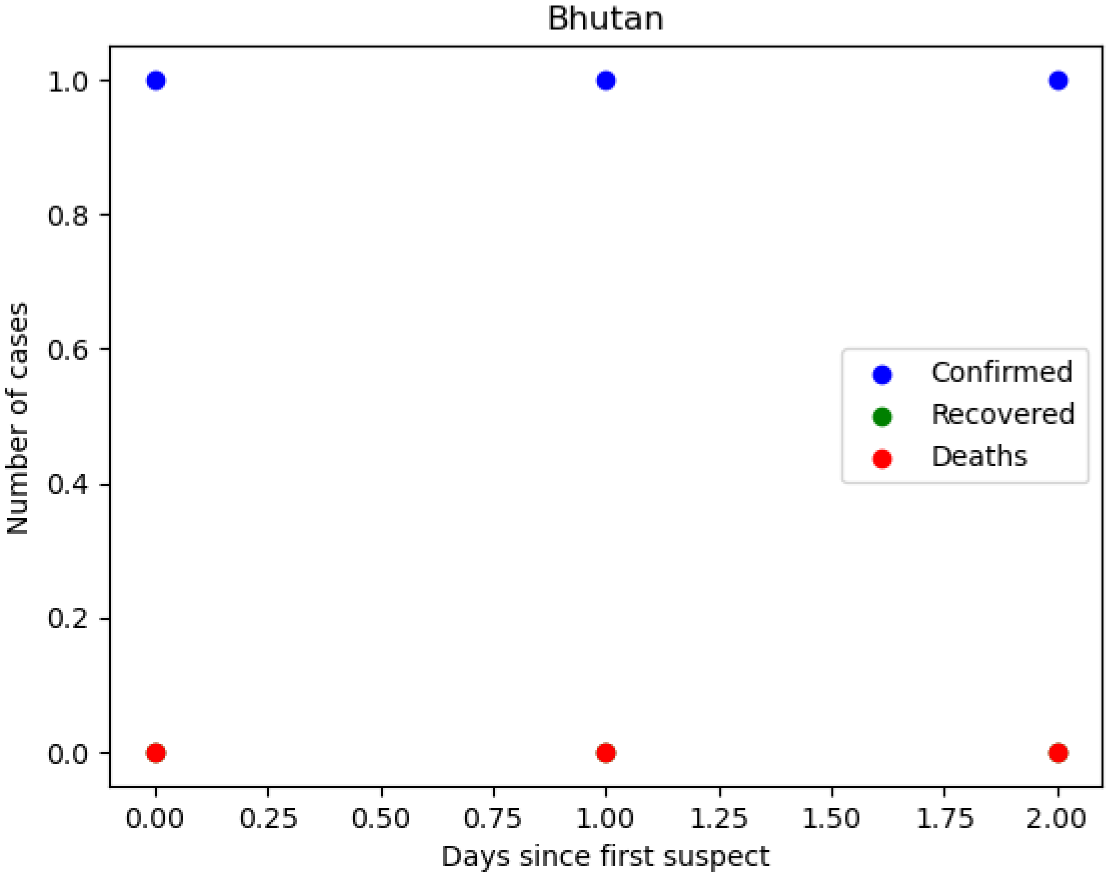

Tunisia
Confirmed
Recovered
Deaths
Number of cases
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Days since first suspect

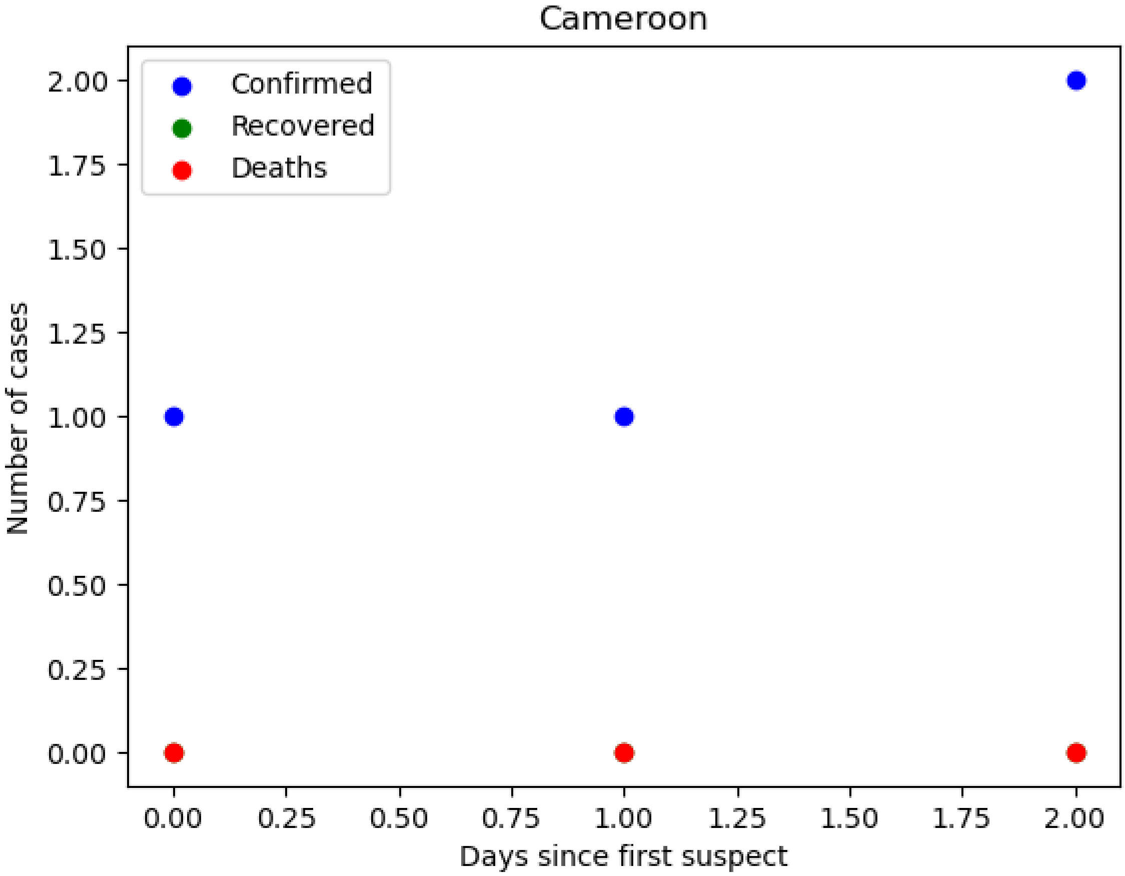

Palestine
Confirmed
Recovered
Deaths
Number of cases
20
15
10
5
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Days since first suspect

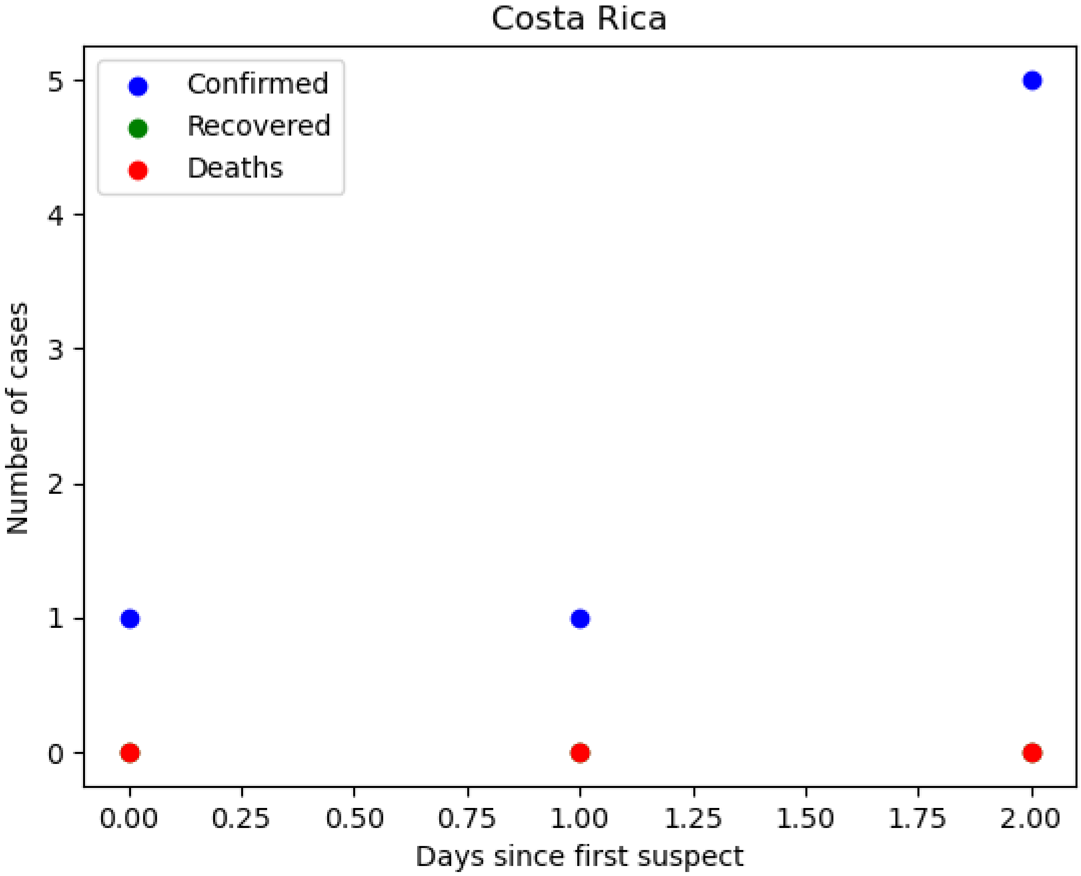

Bosnia and Herzegovina
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Days since first suspect

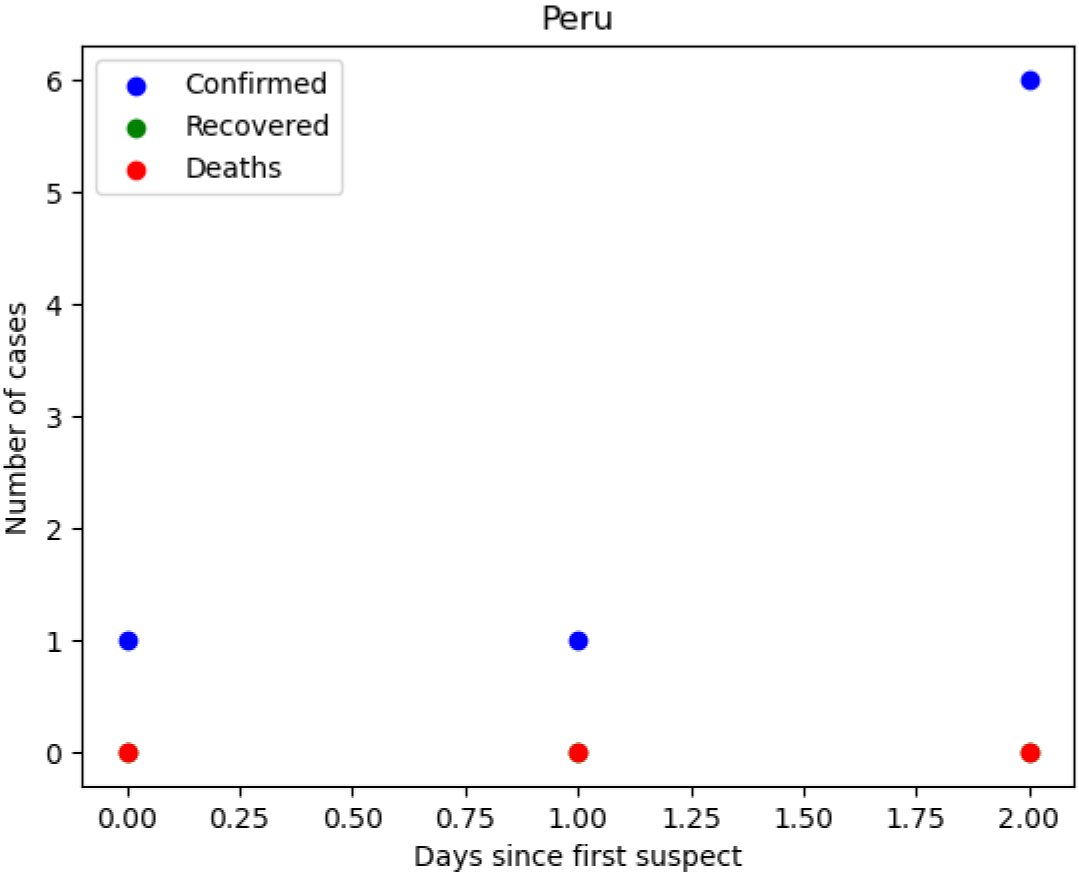

Slovenia
Confirmed
Recovered
Deaths
Number of cases
16
14
12
10
8
6
4
2
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Days since first suspect

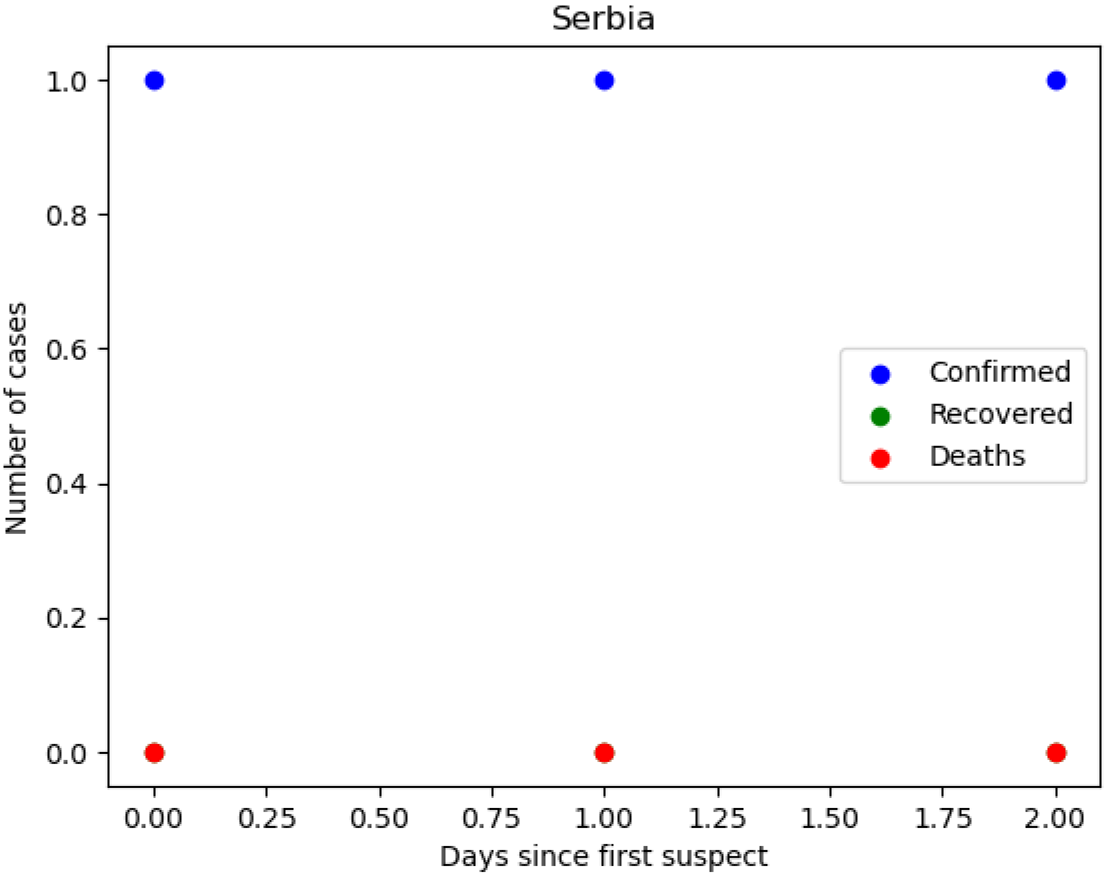

South Africa
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Days since first suspect

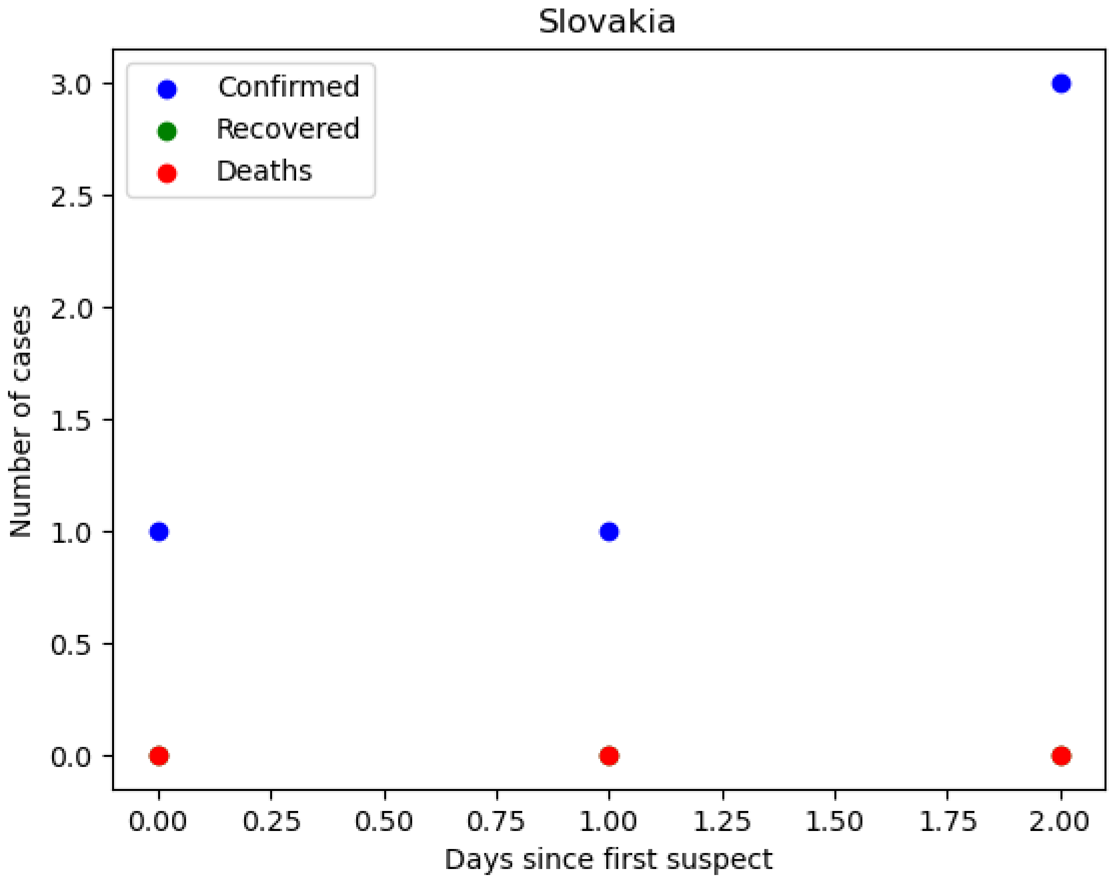

Bhutan
Confirmed
Recovered
Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

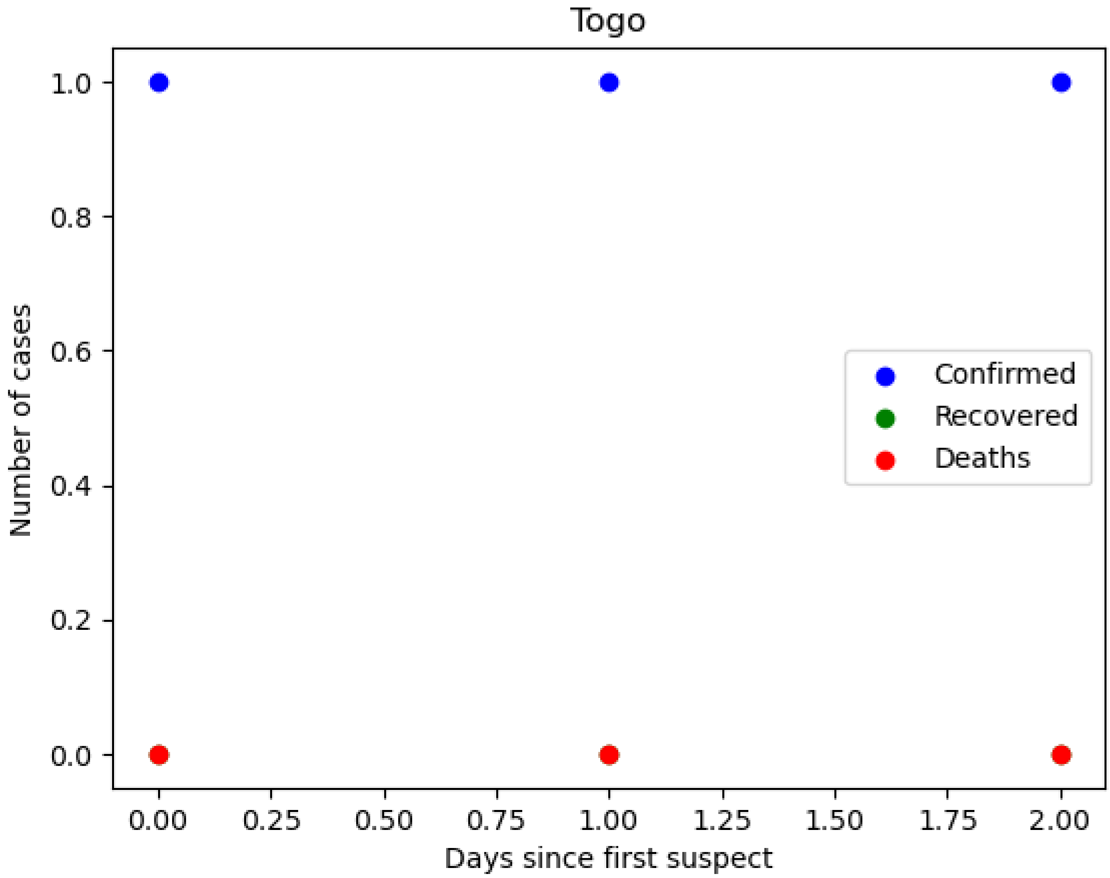

Cameroon
● Confirmed
● Recovered
● Deaths
Number of cases
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

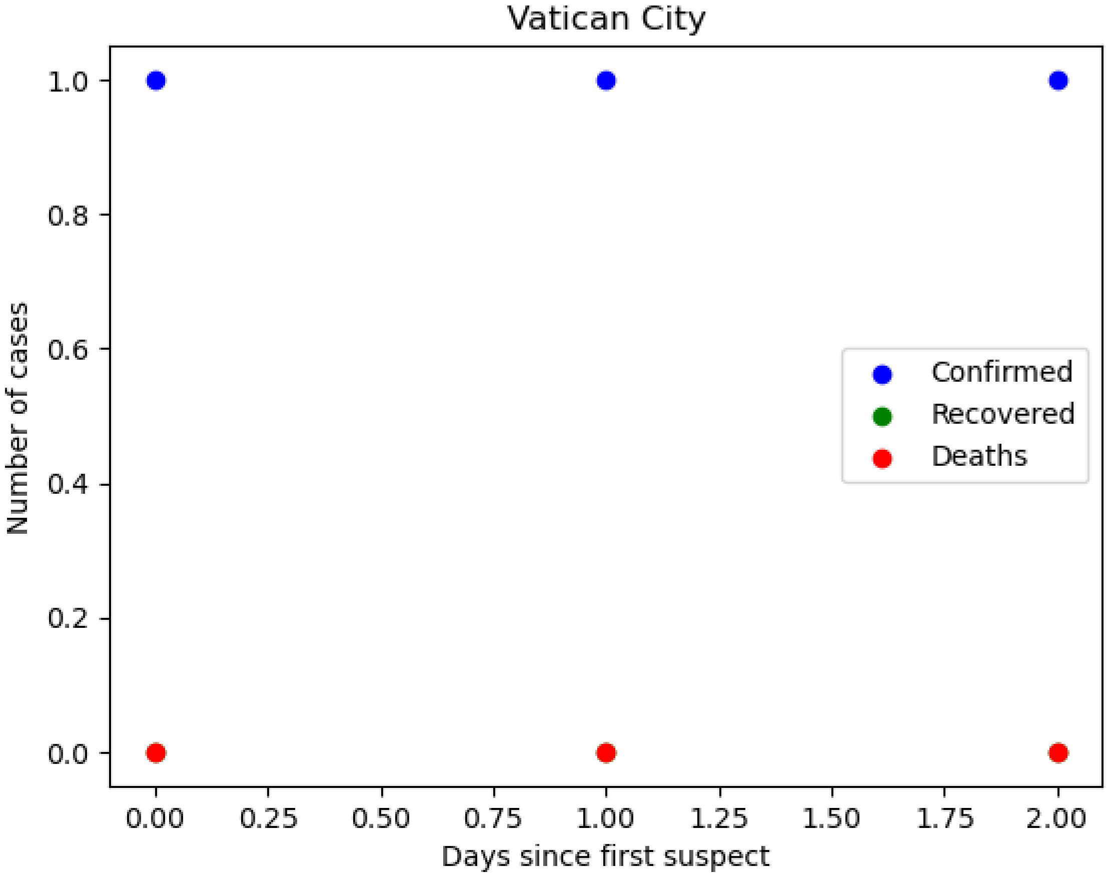

Costa Rica
● Confirmed
● Recovered
● Deaths
Number of cases
5
4
3
2
1
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

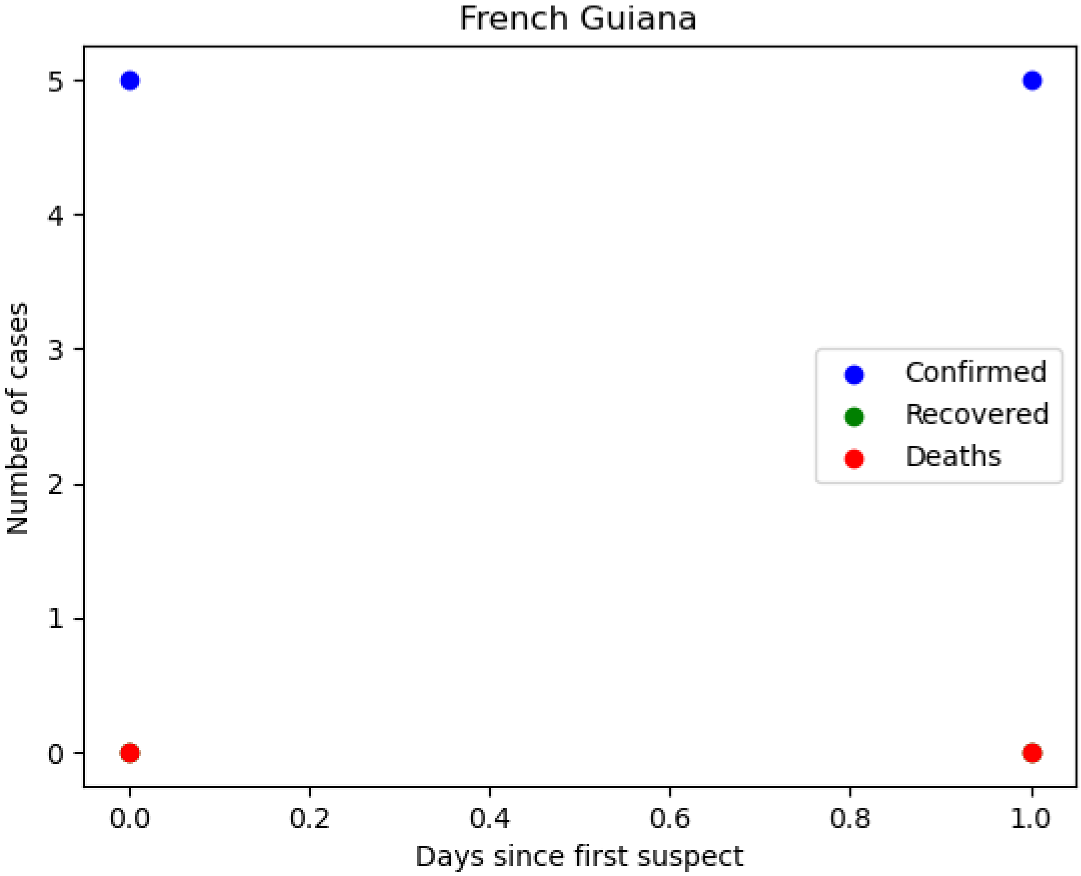

Peru
Confirmed
Recovered
Deaths
Number of cases
6
5
4
3
2
1
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

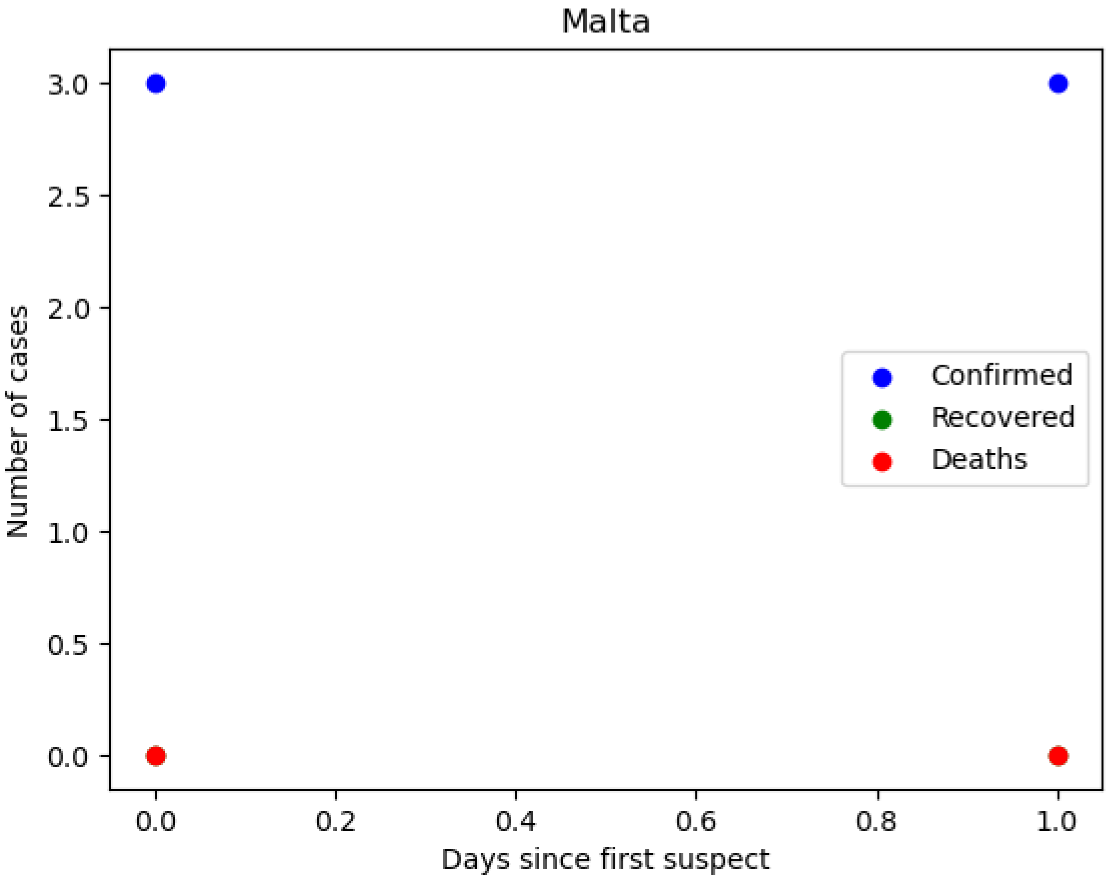

Serbia
Confirmed
Recovered
Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

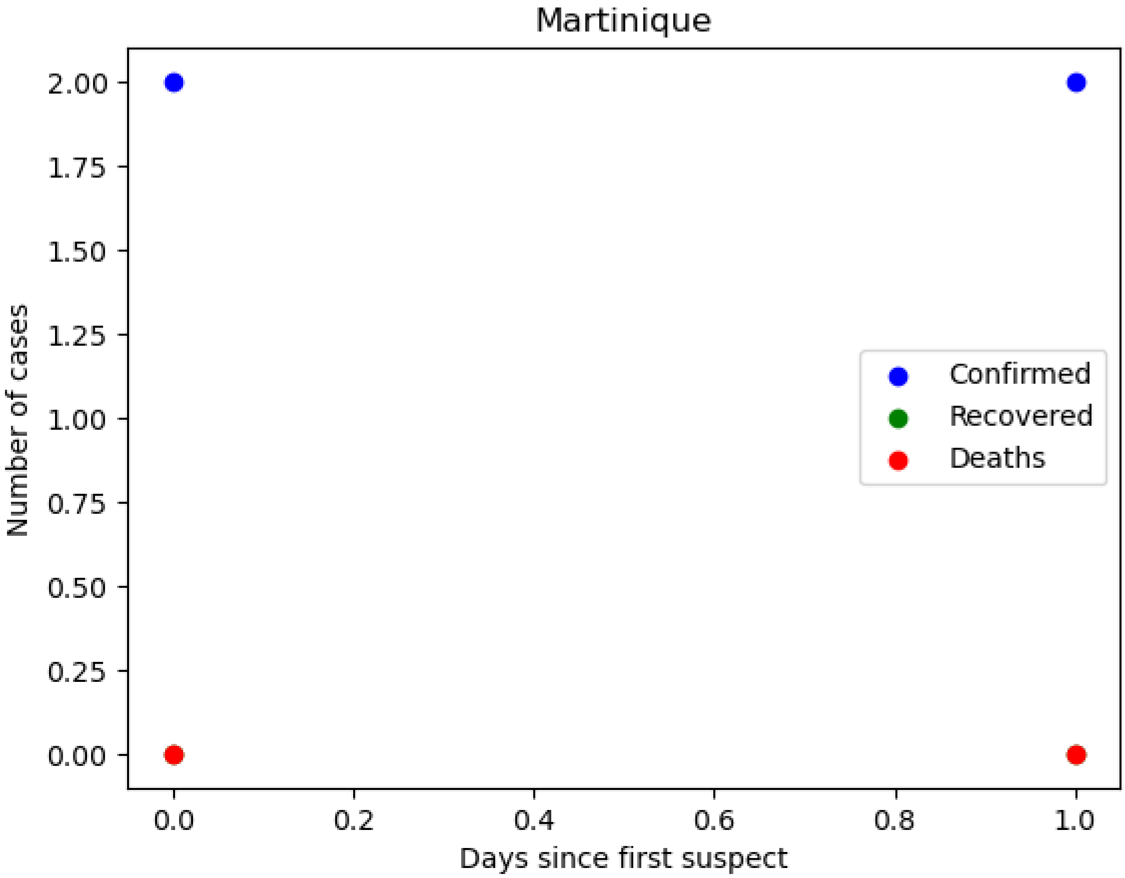

Slovakia
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

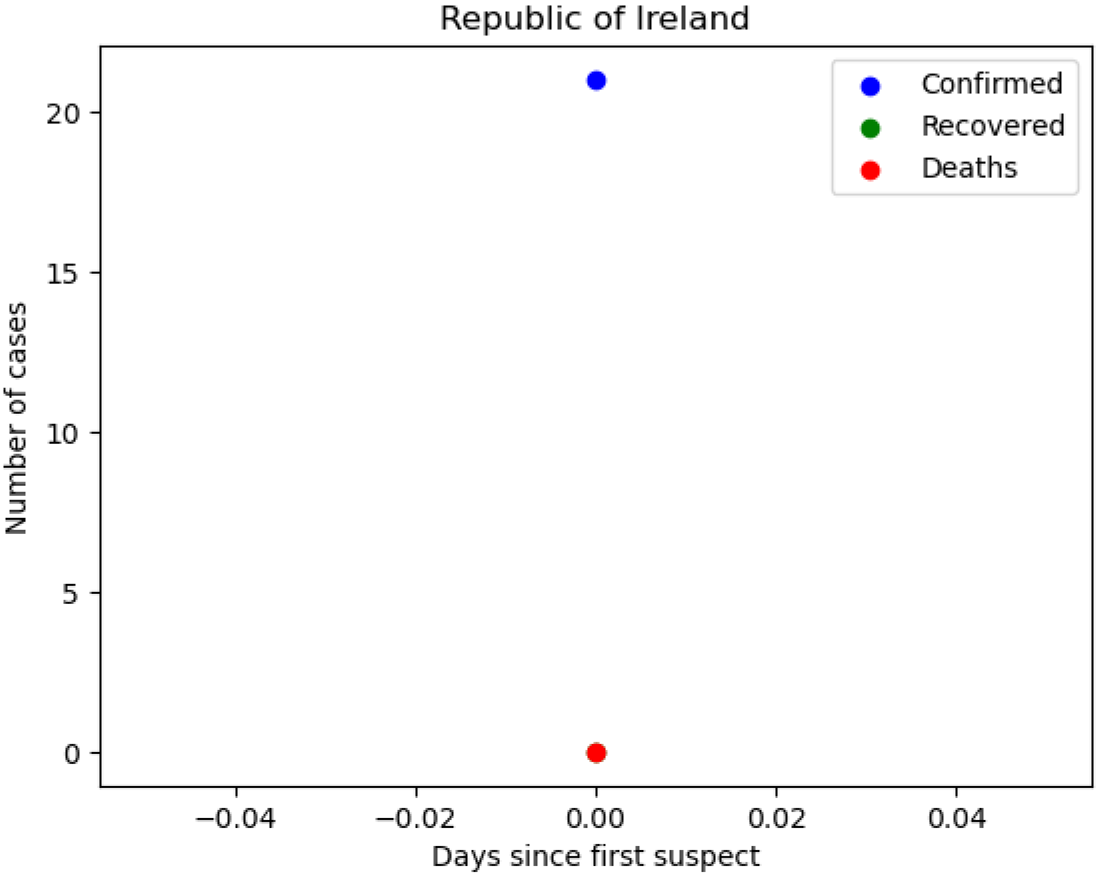

Togo
Confirmed
Recovered
Deaths
Number of cases
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect

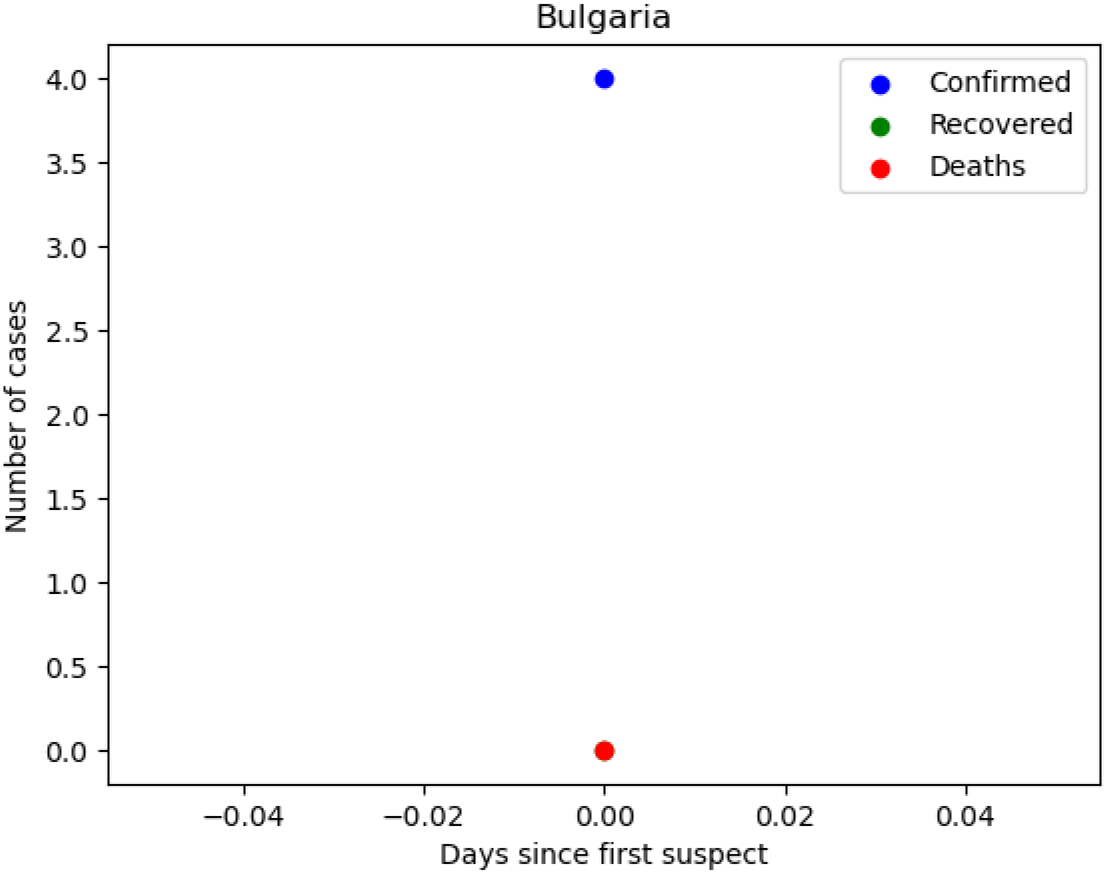

Vatican City
1.0
0.8
0.6
0.4
0.2
0.0
Number of cases
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Days since first suspect
● Confirmed
● Recovered
● Deaths

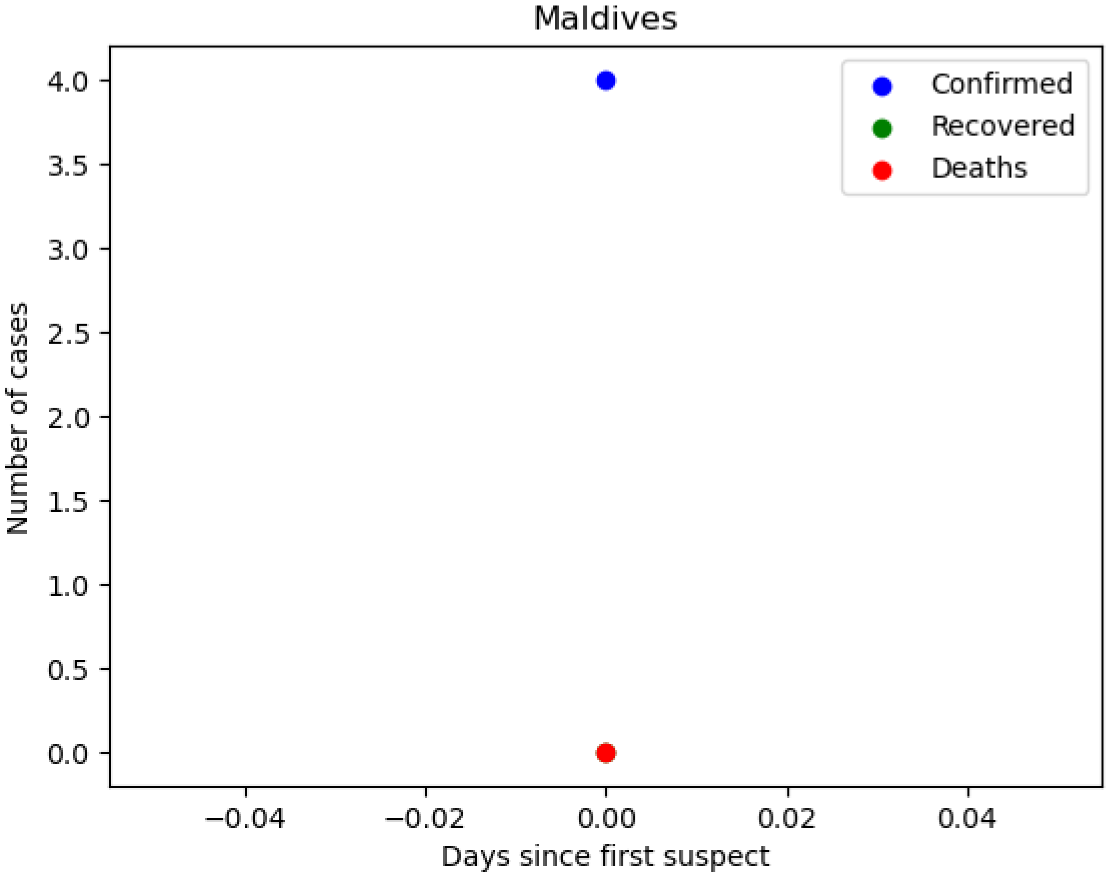

French Guiana
5
4
3
2
1
0
Number of cases
0.0
0.2
0.4
0.6
0.8
1.0
Days since first suspect
● Confirmed
● Recovered
● Deaths

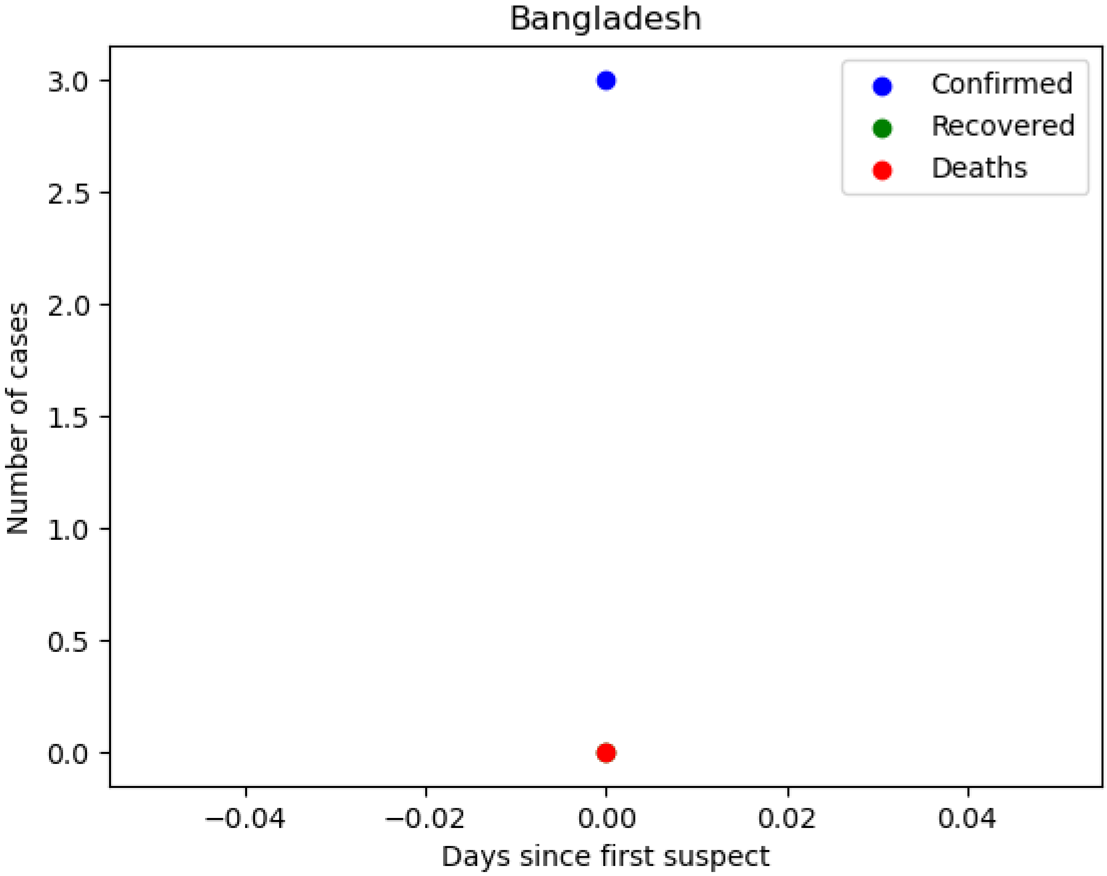

Malta
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Number of cases
0.0
0.2
0.4
0.6
0.8
1.0
Days since first suspect
Confirmed
Recovered
Deaths

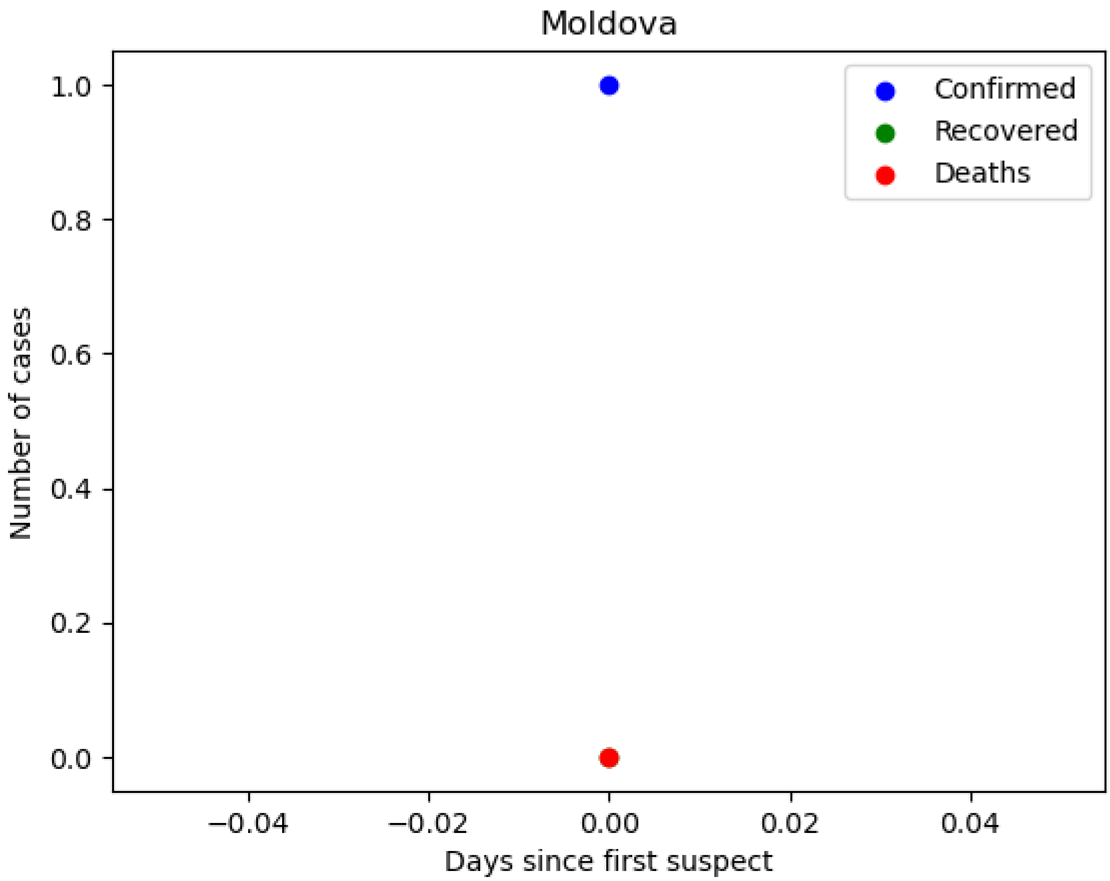

Martinique
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of cases
0.0
0.2
0.4
0.6
0.8
1.0
Days since first suspect
Confirmed
Recovered
Deaths

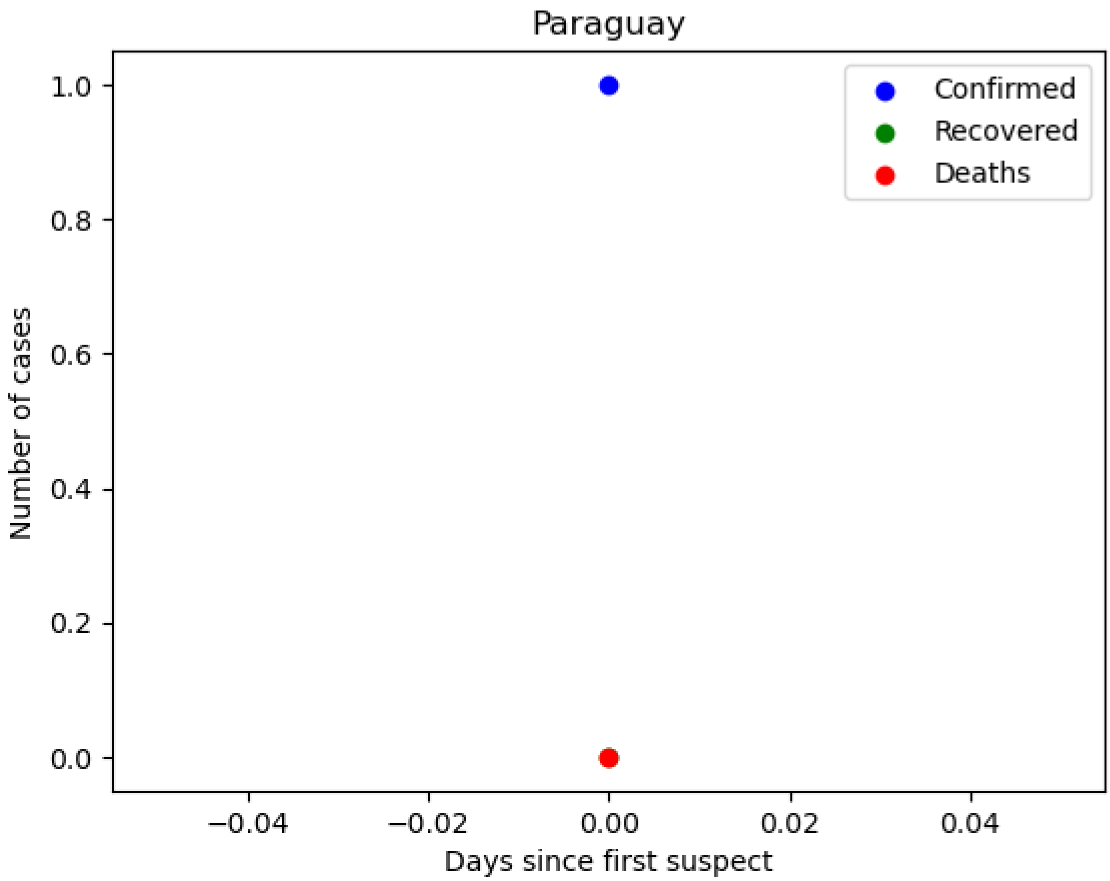

Republic of Ireland
Confirmed
Recovered
Deaths
20
15
10
5
0
Number of cases
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect


Bulgaria
Confirmed
Recovered
Deaths
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Number of cases
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect


Maldives
Confirmed
Recovered
Deaths
Number of cases
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect


Bangladesh
Confirmed
Recovered
Deaths
Number of cases
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.04
−0.02
0.00
0.02
0.04
Days since first suspect





In [48]:

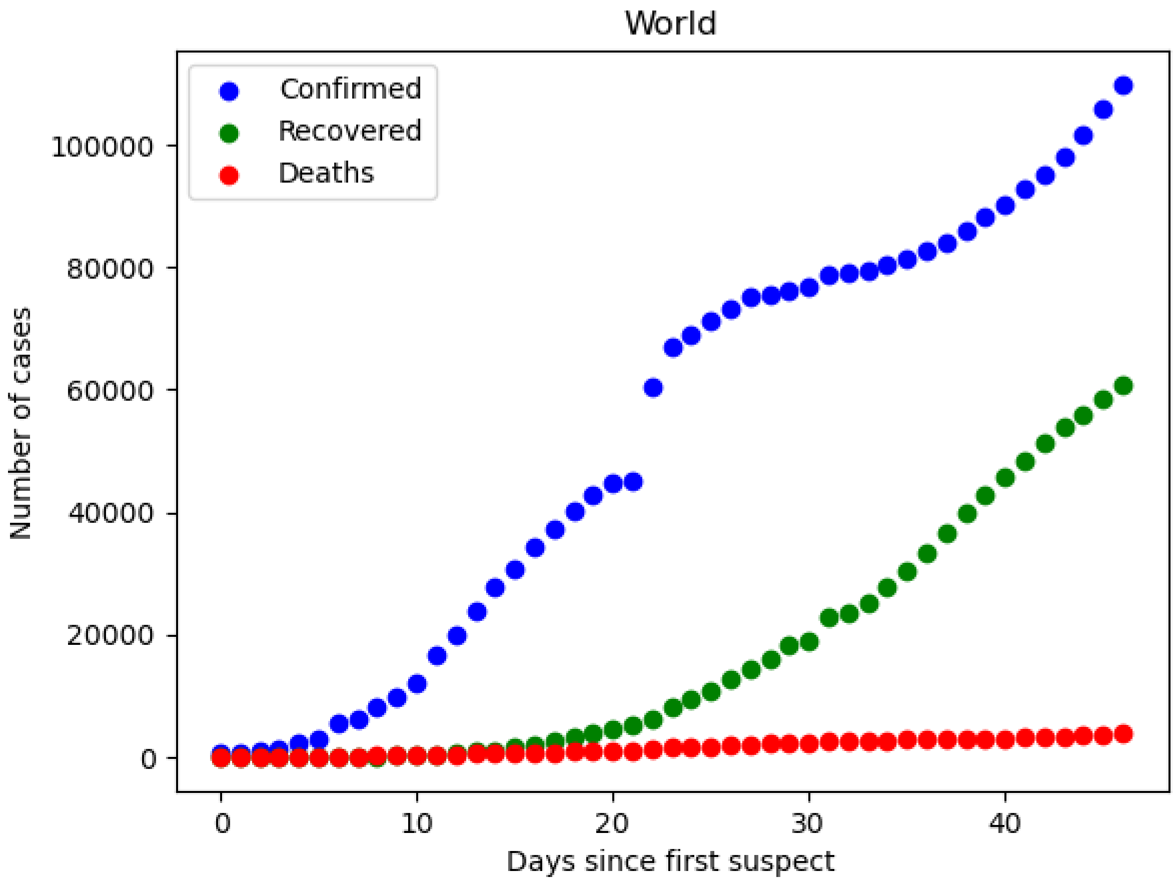
```
df4 = df.groupby(["ObservationDate"])[["Confirmed","Deaths","Recovered"]].sum().reset_in
```

In [50]:

```
C = df4
plt.scatter(np.arange(0,len(C)),C["Confirmed"],color = "blue",label = "Confirmed")
plt.scatter(np.arange(0,len(C)),C["Recovered"],color = "green", label = "Recovered")
plt.scatter(np.arange(0,len(C)),C["Deaths"],color = "red", label = "Deaths")
plt.title("World")
plt.xlabel("Days since first suspect")
plt.ylabel("Number of cases")
plt.legend()
plt.show
```

Out[50]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [ ]: