Four fitting models comparison for SARS-nCoV-2 total cumulative cases curves

Max Pierini*

May 5, 2020

1 Abstract

Four different models have been compared for fitting SARS-nCoV-2 total cumulative cases curves in 187 countries over a period of 104 days. Evaluated models have been: *Simple Logistic Function* (**SLF**), *Simple Gompertz Function* (**SGF**), *Double Logistic Function* (**DLF**) and *Double Gompertz Function* (**DGF**). **DGF** model showed lower MSE, RMSE, NRMSE, MAE, NAE and higher Pearson R compared to the others. R^2 ($p > .99$), R^2_{adj} ($p > .99$), ΔAIC ($p < .55$) and ΔBIC ($p > 0$) showed higher percentages for **DGF** compared with the others.

Results suggest that *Double Gompertz Function* may be a good fitting model for SARS-nCoV-2 cumulative cases curve.

2 Methods

2.1 Data

SARS-nCoV-2 total cumulative cases data have been gathered from Johns Hopkins University GitHub repository [1] and summed into single countries where regional level was provided [2]. Data have been used "as is" without rejecting any outlier and/or error (negative daily Δ). Data and results have been stored in a `pandas` n-dimensional `DataFrame`.

Raw data contained 187 countries and daily cumulative confirmed cases for 104 days, from 2020-01-22 to 2020-05-05.

Data have been collected from: Afghanistan, Albania, Algeria, Andorra, Angola, Antigua and Barbuda, Argentina, Armenia, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bhutan, Bolivia, Bosnia and Herzegovina, Botswana, Brazil, Brunei, Bulgaria, Burkina Faso, Burma, Burundi, Cabo Verde, Cambodia, Cameroon, Canada, Central African Republic, Chad, Chile, China, Colombia, Comoros, Congo (Brazzaville), Congo (Kinshasa), Costa Rica, Cote d'Ivoire, Croatia, Cuba, Cyprus, Czechia, Denmark, Diamond Princess, Djibouti, Dominica, Dominican Republic, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Eswatini, Ethiopia, Fiji, Finland, France, Gabon, Gambia, Georgia, Germany, Ghana, Greece, Grenada, Guatemala, Guinea, Guinea-Bissau, Guyana, Haiti, Holy See, Honduras, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Korea, South, Kosovo, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Liberia,

*info@maxpierini.it

Libya, Liechtenstein, Lithuania, Luxembourg, MS Zaandam, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Mauritania, Mauritius, Mexico, Moldova, Monaco, Mongolia, Montenegro, Morocco, Mozambique, Namibia, Nepal, Netherlands, New Zealand, Nicaragua, Niger, Nigeria, North Macedonia, Norway, Oman, Pakistan, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Poland, Portugal, Qatar, Romania, Russia, Rwanda, Saint Kitts and Nevis, Saint Lucia, Saint Vincent and the Grenadines, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Serbia, Seychelles, Sierra Leone, Singapore, Slovakia, Slovenia, Somalia, South Africa, South Sudan, Spain, Sri Lanka, Sudan, Suriname, Sweden, Switzerland, Syria, Taiwan*, Tajikistan, Tanzania, Thailand, Timor-Leste, Togo, Trinidad and Tobago, Tunisia, Turkey, US, Uganda, Ukraine, United Arab Emirates, United Kingdom, Uruguay, Uzbekistan, Venezuela, Vietnam, West Bank and Gaza, Western Sahara, Yemen, Zambia, Zimbabwe.

2.2 Models

Models have been defined with `lmfit` (implementation of classical `curve_fit` in `scipy`) using Nelder-Mead method for fitting [2].

Total residual from each function have been initially compared (unsorted, sorted, gaussian distribution) to find the model with residual μ closer to 0 and shorter σ. *Akaike Information Criterion* differences (**ΔAIC**) in relative probability density space, *Bayesian Information Criterion* differences (**ΔBIC**), R^2 and R^2_{adj} coefficients mean have been used to find the likely better fitting model that has been finally compared, country by country. See Section 7 for formulae.

Models have been defined as follow:

- Simple Logisic Function (**SLF**):

```python
def simple_logistic_function(x, a, b, k, e):
    d = k * (b - np.array(x))
    return (a / (1 + np.exp(d))) + e
```

$$f(t) = \frac{a}{1 + e^{k(b-t)}} + \varepsilon$$

- Double Logisic Function (**DLF**):

```python
def double_logistic_function(x, a1, b1, k1, a2, b2, k2, e):
    d1 = k1 * (b1 - np.array(x))
    g1 = a1 / (1 + np.exp(d1))
    d2 = k2 * (b2 - np.array(x))
    g2 = (a2 - a1) / (1 + np.exp(d2))
    return g1 + g2 + e
```

$$f(t) = \frac{a_1}{1 + e^{k_1(b_1-t)}} + \frac{a_2 - a_1}{1 + e^{k_2(b_2-t)}} + \varepsilon$$

- Simple Gompertz Function (**SGF**):

```python
def simple_gompertz_function(x, a, b, k, e):
    exp = - np.exp(k * (b - x))
    return a * np.exp(exp) + e
```

$$f(t) = a \cdot e^{-e^{k(b-t)}} + \varepsilon$$

- Double Gompertz Function (**DGF**):

```
def double_gompertz_function(x, a1, b1, k1, a2, b2, k2, e):
    exp1 = - np.exp(k1 * (b1 - x))
    g1 = a1 * np.exp(exp1)
    exp2 = - np.exp(k2 * (b2 - x))
    g2 = (a2 - a1) * np.exp(exp2)
    return g1 + g2 + e
```

$$f(t) = a_1 \cdot e^{-e^{k_1(b_1-t)}} + (a_2 - a_1) \cdot e^{-e^{k_2(b_2-t)}} + \varepsilon$$

3 Model fitting

Fitting has been performed with `lmfit` using Nelder-Mead method

```
model = lmfit.Model(function)
result = model.fit(data=y, params=p, x=x, method='Nelder', nan_policy='omit')
```

initial parameters `p` have been guessed as follows (where y are observed values):

- **SLF** and **SGF**

```
p = model.make_params(
    a=y[-1],
    b=max_y_i,
    k=.1,
    e=y[0]
)
```

$$a = y_{-1}$$

$$b = x_{\mathbf{max}(dy)}$$

$$k = 0.1$$

$$\varepsilon = y_0$$

- **DLF** and **DGF**

```
p = model.make_params(
    a1=y[max_y_i] * 2,
    b1=max_y_i,
    k1=.1,
    a2=max(y),
    b2=len(y),
    k2=.1,
    e=y[0]
)
```

$$a_1 = 2y_{\mathbf{max}(dy)}$$
$$b_1 = x_{\mathbf{max}(dy)}$$
$$k_1 = 0.1$$
$$a_2 = \mathbf{max}(y)$$
$$b_2 = x_{y_{-1}}$$
$$k_2 = 0.1$$
$$\varepsilon = y_0$$

Fitting failed for 0 countries, returning best fit information from 187 countries, for a total of 19448 observed and 77792 predicted values.

Complete `python` backend for data gathering, fitting and analysis along with a `pickle` saved dataframe of all measured data and results is online avalaible [2].

Fitting examples are reported in figures [REF] [REF] [REF].

4 Analysis

Several skill score have been used to evaluate to average skil and skill interpolating extreme values (outliers).

Mean absolute error (**MAE**) is a natural, unambiguous, measure of average error [3]. It shows the errors in the same unit as variables themselves. MAE is bounded below by 0 (best case) and unbounded above. Advantage over Mean Bias Error (**MBE**) is that, taking absolute error values, positive and negative errors can't cancel out [4].

Normalized Absolute Error (**NAE**), bounded below 0 (best case) and "virtually" unbounded above. If more than 1 errors are greater than observed values themselves. Advantage over Normalized Bias Error (**NBE**) is that, taking absolute error values, positive and negative errors can't cancel out. [REF]

Mean Squared Error (**MSE**), variance, taking the square of residual is highly sensitive to large outliers [4]. Bounded below 0 (best case) and unbounded above.

Root Mean Squared Error (**RMSE**) is very commonly used as a measure of deviation from the observed value in the same unit of data mean. Although it has been criticized as being ambiguous [3] and its dependence on the squared error means that it is not resistant to outliers deviating from a Gaussian distribution. It has been included because of its sensitivity to large outliers. Bounded below 0 (best case) and unbounded above.

Normalized Root Mean Squared Error (**NRMSE**) allows to compare **RMSE** of different models, normalized on observed values, on a $(0, 1]$ scale [REF].

Pearson Correlation coefficient (**Pearson R**) depends on squared deviations and so is not a resistant measure. However, this statistic removes the effect of any bias in the interpolated data. Problems with correctly capturing the variance will not be highlighted as the measure normalizes the observed and modeled values by their standard deviations [4]. The statistic is standardized. However, because of its insensitivity to biases and errors in variance, the correlation coefficient should be considered as a measure of potential skill [5].

R^2 coefficient test … Bounded from 0 to 1, $R^2 > .99$ has been fixed to evaluate model H_0 against alternative model H_1 [6].

R^2_{adj} coefficient test … Bounded from 0 to 1, $R^2_{adj} > .99$ has been fixed to evaluate model H_0 against alternative model H_1 [6].

Akaike Information Criterion (**AIC**), its score (or weight) and its relative probability distribution space. Akaike's weights compute the relative probability of two models. Within the probability distribution space, bounded from 0 to 1, if $\Delta\mathbf{AIC} = \mathbf{AIC}_1 - \mathbf{AIC}_0$ an Akaike weight of $\Delta\mathbf{AIC}_p < .5$ means model H_0 has more chances to be better fitting than alternative model H_1 and $1 - \Delta\mathbf{AIC}_p$ is the relative probability of model H_0 against H_1[7].

Bayesian Information Criterion (**BIC**) and its delta … A $\Delta\mathbf{BIC} > 0$ means model H_0 has evidence to be better fitting than alternative model H_1. Look in Appendix for formulae [8].

4.1 Analysis Example

Example of analysis of two fitting models for a noisy observed sample.

Observed o values are generated by

$$o = f(x) = a \cdot x^2 + b + \varepsilon_x$$

where $x = (0, 100]$, $a = 2.3$, $b = 1000$ and ε_x is a random noise $(0, 3000]$.

Observed are fitted with two models using `scipy.optimize.curve_fit` (least squares method):

$$e_1(x) = a \cdot x^3$$

and

$$e_2(x) = a \cdot x^2 + b$$

Model e_2 is taken as null hypothesis H_0.

```python
import numpy as np
from scipy import stats as sts
from scipy.optimize import curve_fit
import sklearn.metrics as skl
from matplotlib import pyplot as plt

def AICp(diff):
    try:
        return np.exp(-.5 * diff) / (1 + np.exp(-.5 * diff))
    except Exception as err:
        if diff > 0:
```

```python
            return 1.
        return 0.

def func_noise(x, a, b, err=False):
    noise = np.random.normal(0, 3e3, len(x))
    return a * np.array(x) ** 2 + b + (noise if err else 0)

def func1(x, a):
    return a * np.array(x) ** 3

def func2(x, a, b):
    return a * np.array(x) ** 2 + b

def report(o, E, P):
    cT = []
    mae, nae, mbe, nbe, mse, rmse, nrmse, r, r2, r2a, aic, bic = (
        [] for _ in range(12)
    )
    for i, e in enumerate(E):
        mae.append(skl.mean_absolute_error(o, e))
        nae.append(np.sum(np.abs(o - e)) / np.sum(o))
        mbe.append(np.sum(o - e) / len(o))
        nbe.append(np.sum(o - e) / np.sum(o))
        _mse = skl.mean_squared_error(o, e)
        mse.append(_mse)
        _rmse = np.sqrt(_mse)
        rmse.append(_rmse)
        _nrmse = _rmse / np.sum(o)
        nrmse.append(_nrmse)
        r.append(sts.pearsonr(o, e)[0])
        r2.append(skl.r2_score(o, e))
        rss = np.sum(np.abs(o - e) ** 2)
        tss = np.sum(np.abs(o - np.mean(o)) ** 2)
        r2a.append(1 - (rss / tss) * ((len(o) - 1) / (len(o) - P[i] - 1)))
        aic.append(len(o) * np.log(rss / len(o)) + 2 * 1)
        bic.append(len(o) * np.log(rss / len(o)) + np.log(len(o)) * P[i])
    cT.append([f"{mae[0]:.0f}", f"{mae[1]:.0f}", f"{(mae[0]-mae[1]):.0f}"])
    cT.append([f"{nae[0]}", f"{nae[1]}", f"{(nae[0]-nae[1])}"])
    cT.append([f"{mbe[0]:.0f}", f"{mbe[1]}", f"{(mbe[0]-mbe[1]):.0f}"])
    cT.append([f"{nbe[0]}", f"{nbe[1]}", f"{(nbe[0]-nbe[1])}"])
    cT.append([f"{mse[0]:.0f}", f"{mse[1]:.0f}", f"{(mse[0]-mse[1]):.0f}"])
    cT.append([f"{rmse[0]:.0f}", f"{rmse[1]:.0f}", f"{(rmse[0]-rmse[1]):.0f}"])
    cT.append([f"{nrmse[0]}", f"{nrmse[1]}", f"{(nrmse[0]-nrmse[1])}"])
    cT.append([f"{r[0]}", f"{r[1]}", f"{(r[0]-r[1])}"])
    cT.append([f"{r2[0]}", f"{r2[1]}", f"{(r2[0]-r2[1])}"])
    cT.append([f"{r2a[0]}", f"{r2a[1]}", f"{(r2a[0]-r2a[1])}"])
    cT.append([f"{aic[0]:.0f}", f"{aic[1]:.0f}", f"$p$: {AICp(aic[0]-aic[1])}"])
    cT.append([f"{bic[0]:.0f}", f"{bic[1]:.0f}", f"{bic[0] - bic[1]:.0f}"])
```

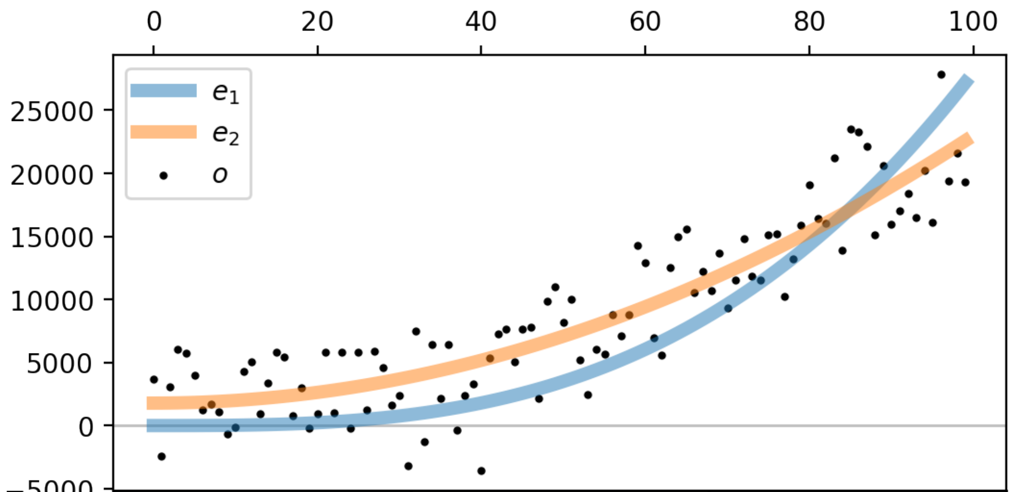
```
    return cT

x = np.arange(0, 100)
o = func_noise(x, 2.3, 1e3, err=True)
popt1, _ = curve_fit(func1, x, o)
e1 = func1(x, *popt1)
popt2, _ = curve_fit(func2, x, o)
e2 = func2(x, *popt2)

cellText = report(o, [e1, e2], [1, 2])

fig, ax = plt.subplots(figsize=(6, 3), dpi=200)
ax.xaxis.tick_top()
ax.axhline(0, lw=1, alpha=.25, c="k")
ax.plot(x, e1, label="$e_1$", lw=5, alpha=.5)
ax.plot(x, e2, label="$e_2$", lw=5, alpha=.5)
ax.scatter(x, o, label="$o$", s=4, c="k")
ax.legend(loc="best")
ax.table(
    cellText=cellText,
    colLabels=[
        "$e_1$  $\mathbf{H_1}$", "$e_2$  $\mathbf{H_0}$",
        "$\Delta = \mathbf{H_1} - \mathbf{H_0}$"
    ],
    colColours=["lightblue", "wheat", "lightgrey"],
    rowColours=["lightgrey" for _ in range(12)],
    rowLabels=[
        "MAE", "NAE", "MBE", "NBE", "MSE", "RMSE", "NRMSE",
        "$R$",  "$R^2$", "$R^2_{adj}$", "AIC", "BIC"
    ],
)
plt.show()
```

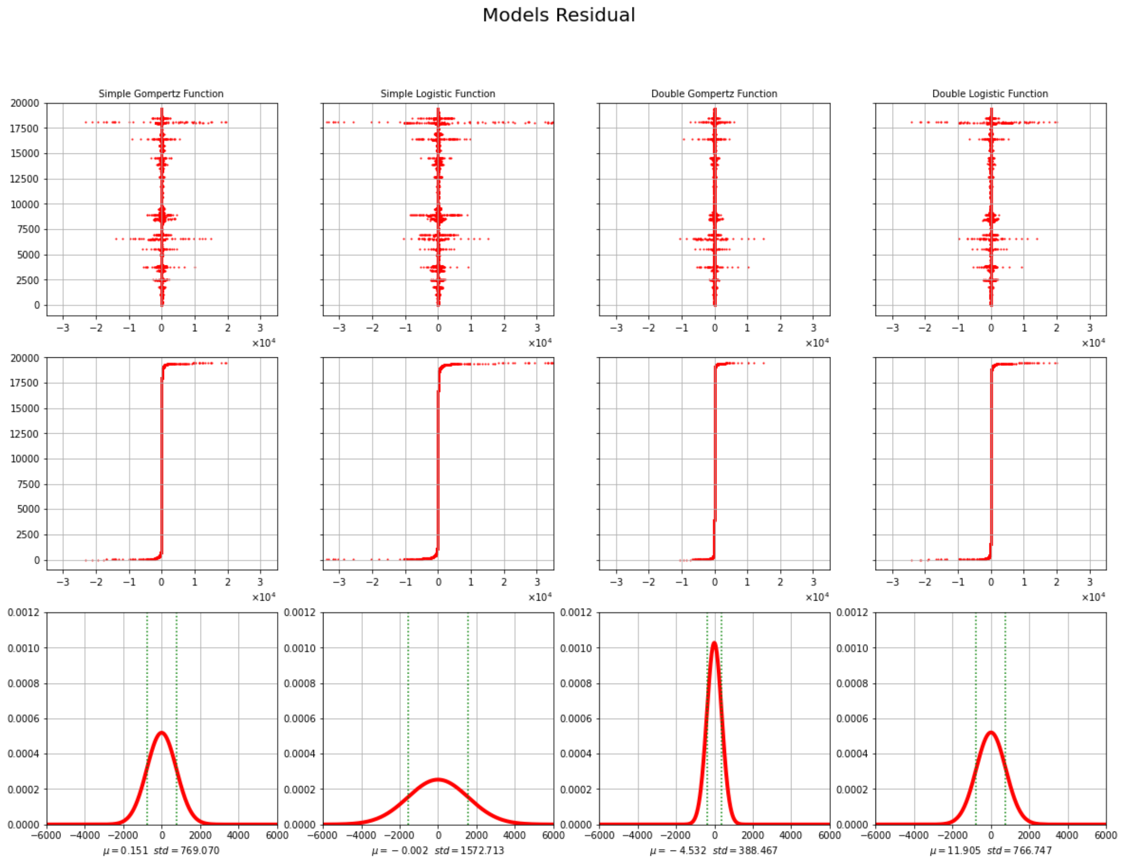
Fitting and analysis report (figure [REF]).



	e_1 $\mathbf{H_1}$	e_2 $\mathbf{H_0}$	$\Delta = \mathbf{H_1} - \mathbf{H_0}$
MAE	3454	2558	895
NAE	0.39499386222658633	0.2925789093424907	0.10241495288409563
MBE	1881	-1.5578944476146718e-05	1881
NBE	0.21517007633240906	-1.7817711649018681e-09	0.21517007811418024
MSE	16843458	9315996	7527462
RMSE	4104	3052	1052
NRMSE	0.004693853402217489	0.003490828709814412	0.0012030246924030767
R	0.8844354093909971	0.8989744187757587	-0.014539009384761603
R^2	0.6531414236973713	0.8081550056132131	-0.1550135819158418
R^2_{adj}	0.6496020504697935	0.8041994387186402	-0.15459738824884672
AIC	1666	1607	p: 1.380072001855801e-13
BIC	1669	1614	55

4.2 Residual

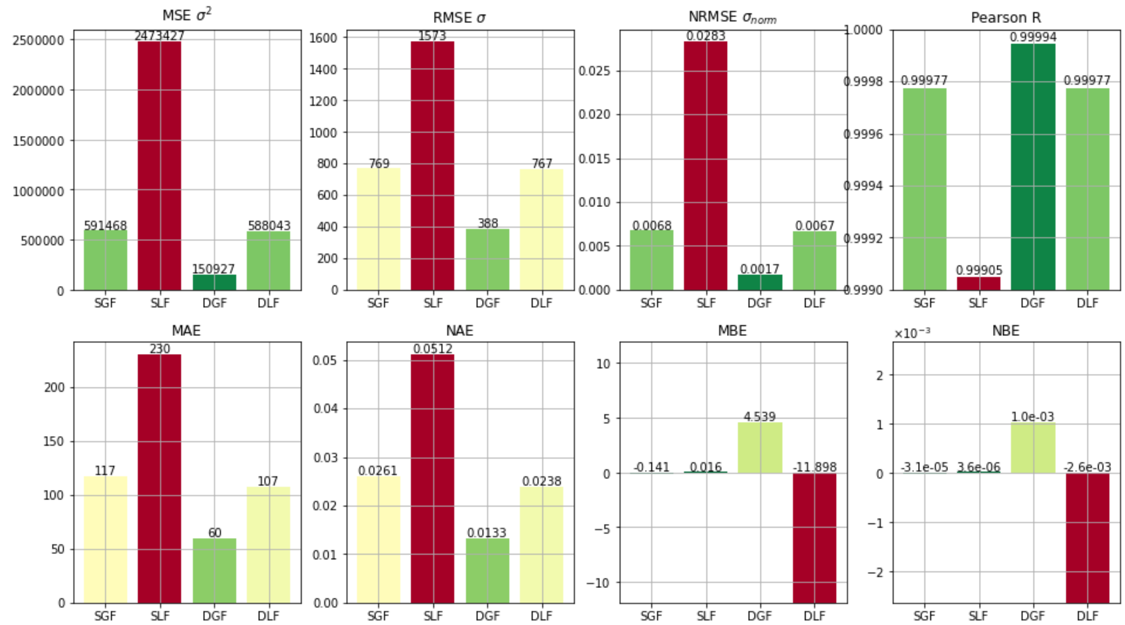
Total residual from each model have been collected and compared to get a first “rough” evidence of the most likely better fitting model [FIG].



DGF showed the lower residual standard deviation while the mean of all four models has been found very close to 0.

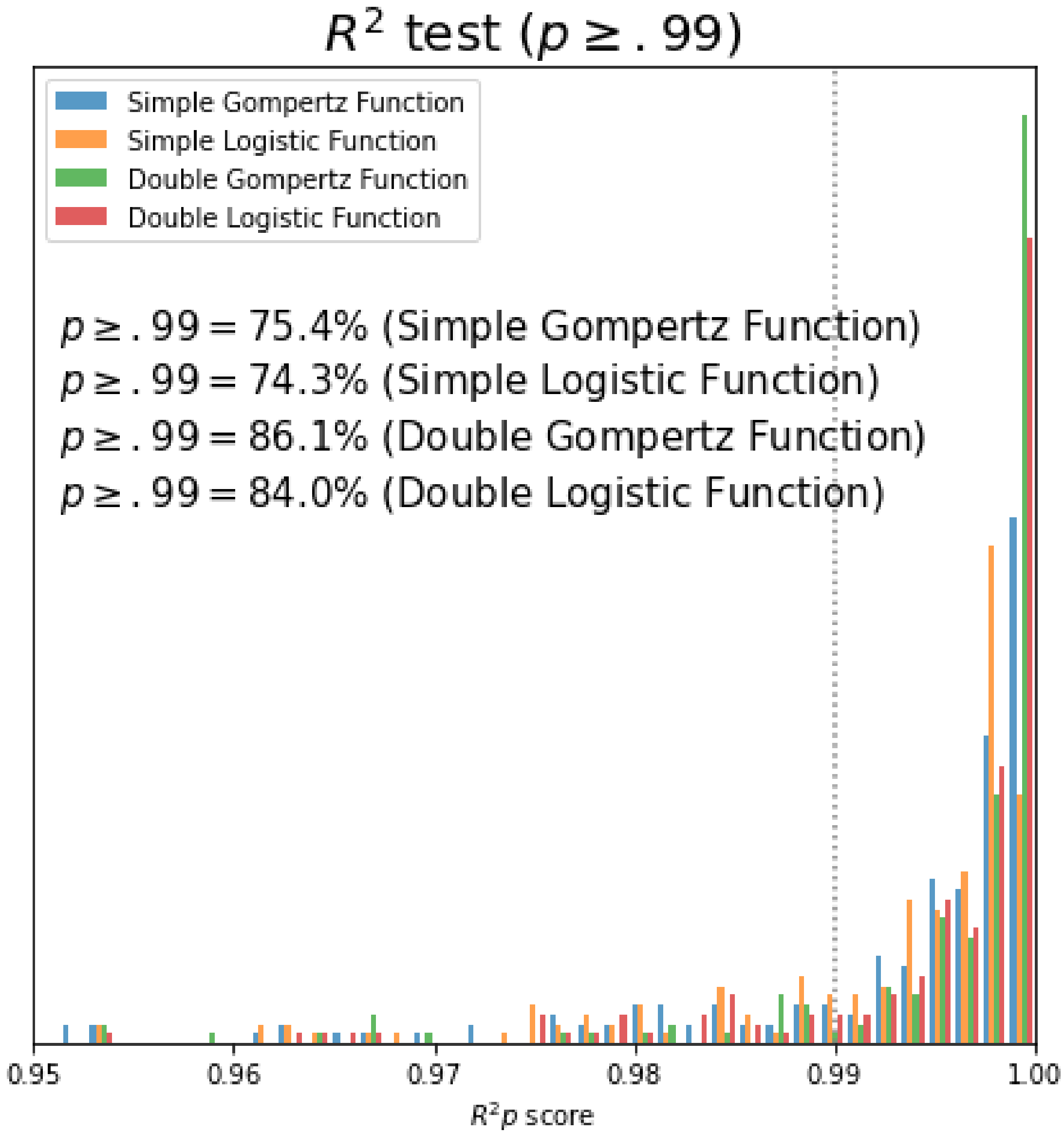
Mean Squared Error (**MSE**, Variance), Root Mean Squared Error (**RMSE**, Standard Deviation), Normalize Root Mean Squared Error (**NRMSE**), Normalized Standard Deviation), Mean Absolute Error (**MAE**), Normalized Absolute Error (**NAE**), Mean Bias Error (**MBE**), Normalized Bias Error (**NBE**) and Pearson Correlation Coefficient (**Pearson R**) have been computed for all models residual Section 4.3 (see Appendix for formulae Section 7). **DGF** showed the best results for all values, but MBE and NBE, confirming the first null hypothesis that could have been the best fitting model among the chosen ones.

4.3 Errors tests



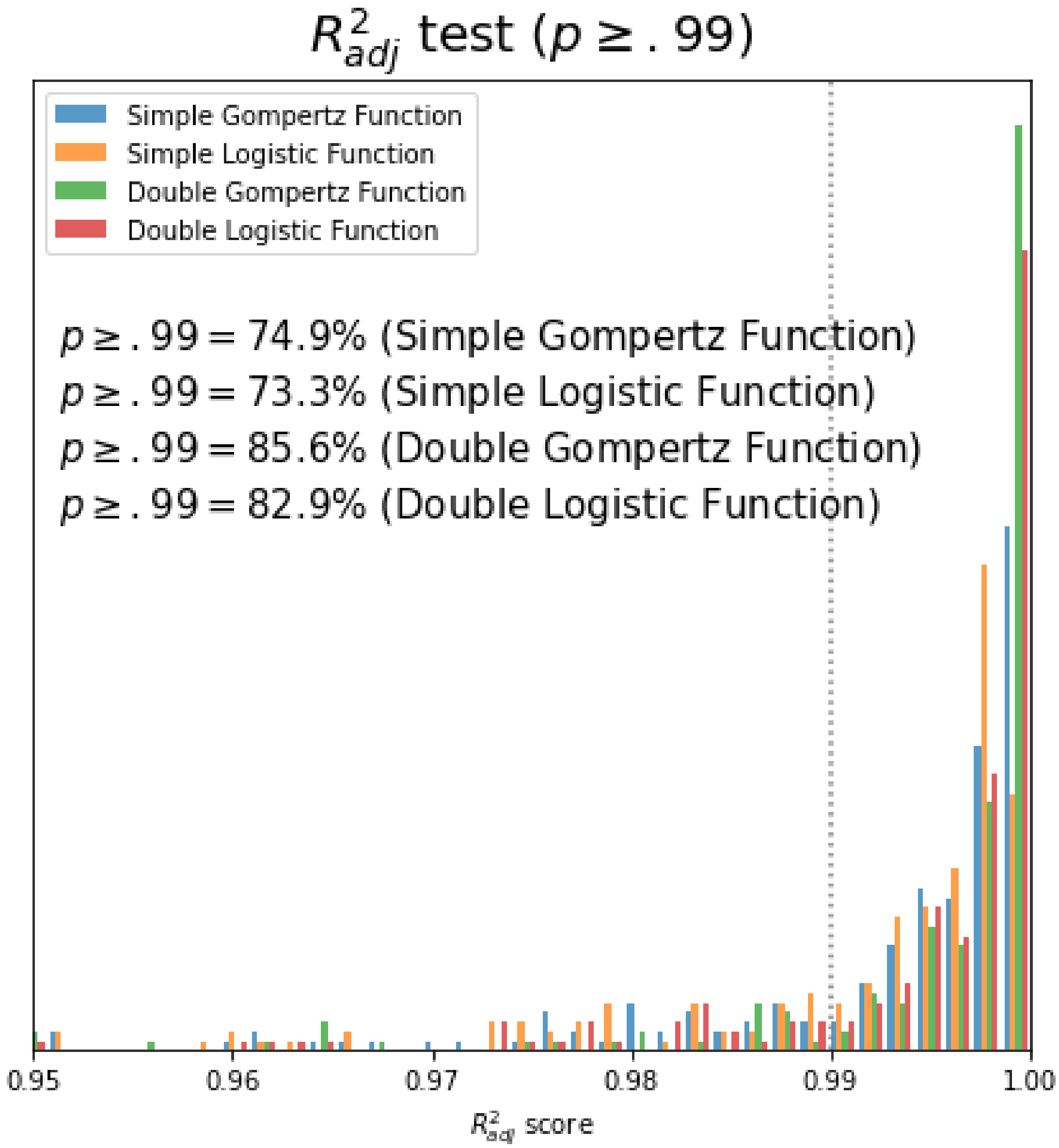
Coefficient of Determination R^2, Adjusted Coefficient of Determination R^2_{adj}, Akaike Information Criterion (**AIC**) and Bayesian Information Criterion (**BIC**) have computed and collected from all fits and compared with each other.

4.4 R2

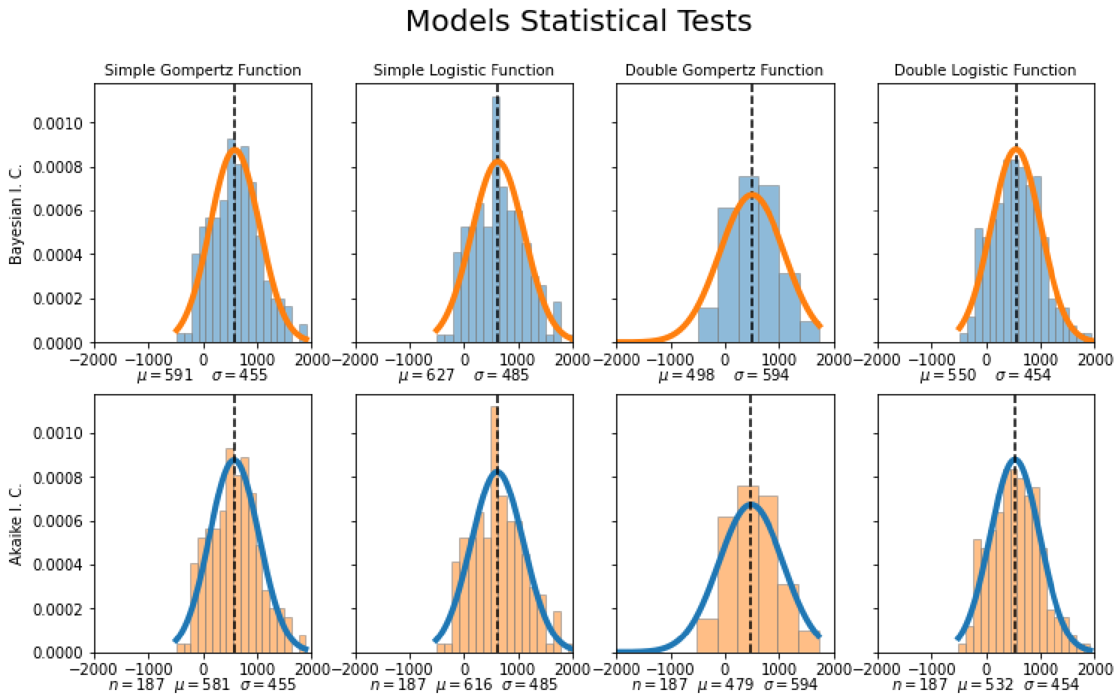


R^2 test ($p \geq .99$)
Simple Gompertz Function
Simple Logistic Function
Double Gompertz Function
Double Logistic Function
$p \geq .99 = 75.4\%$ (Simple Gompertz Function)
$p \geq .99 = 74.3\%$ (Simple Logistic Function)
$p \geq .99 = 86.1\%$ (Double Gompertz Function)
$p \geq .99 = 84.0\%$ (Double Logistic Function)
0.95
0.96
0.97
0.98
0.99
1.00
$R^2 p$ score

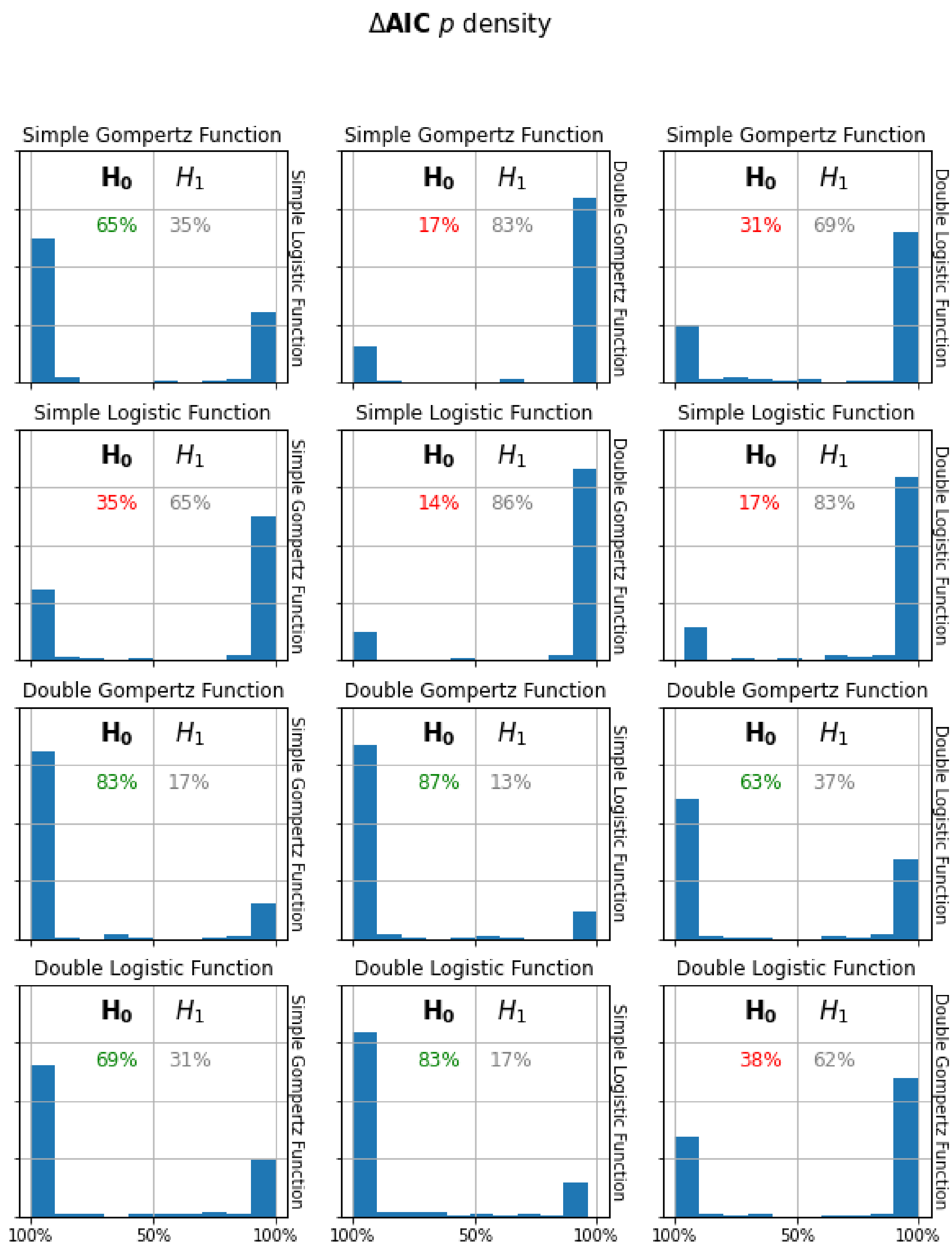
4.5 Adjusted R2



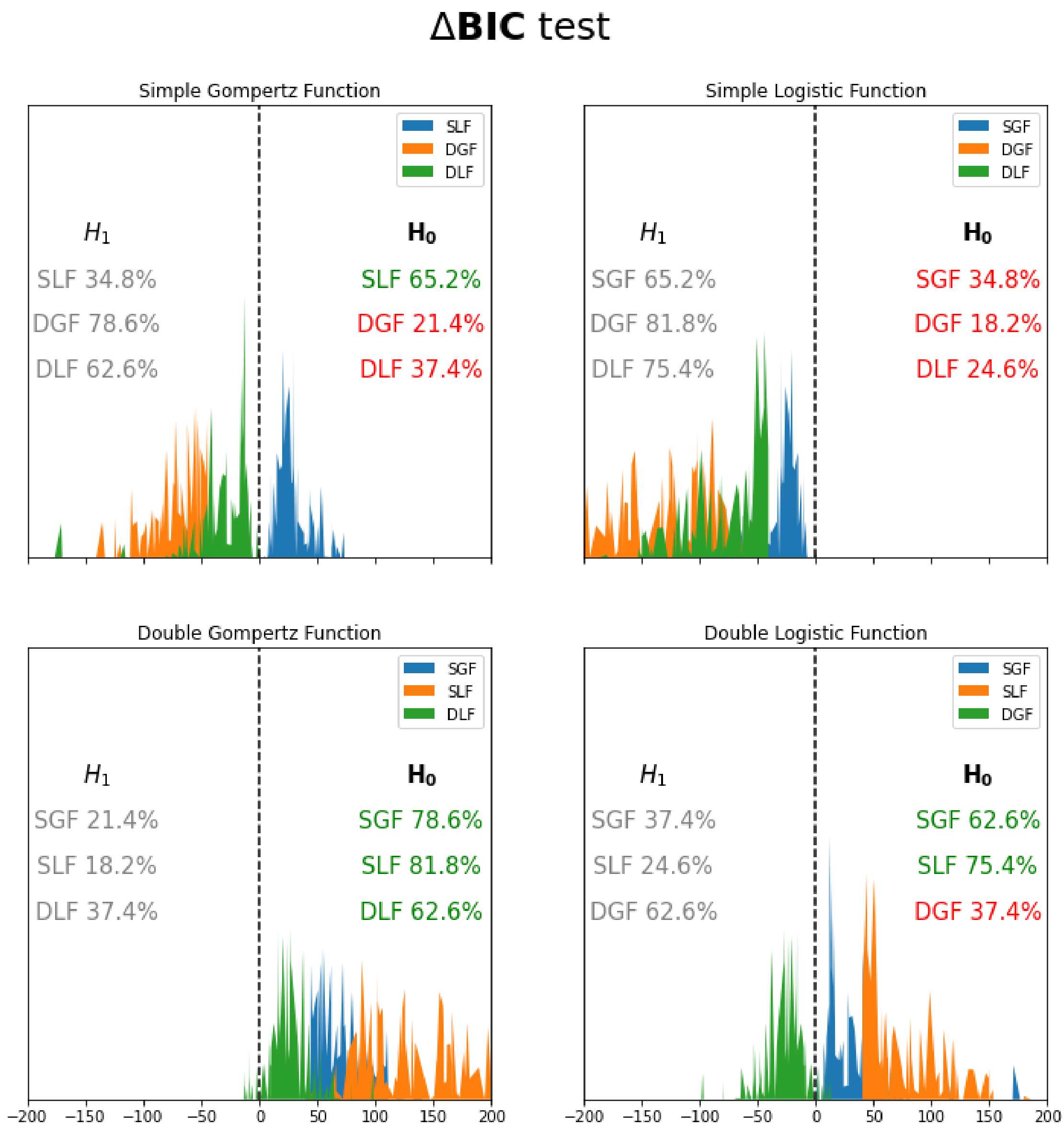
4.6 Information Criteria



4.6.1 AIC


ΔAIC p density
Simple Gompertz Function
Simple Logistic Function
Double Gompertz Function
Double Logistic Function
H0
H1
65%
35%
17%
83%
31%
69%
14%
86%
87%
13%
63%
37%
38%
62%
100%
50%
100%

4.6.2 BIC



Double Gompertz Function showed Δ**BIC** < 0 for:

- ***40 countries*** compared with *Simple Gompertz Function*. 21 have less than 1000 cases: Botswana, Burundi, Central African Republic, Chad, Comoros, Equatorial Guinea, Gabon, Gambia, Guatemala, Guinea-Bissau, Liechtenstein, Maldives, Sao Tome and Principe, Sierra Leone, Sudan, Suriname, Tajikistan, Tanzania, Vietnam, Yemen, Zambia. ***19 have more than 1000 cases***: Bolivia, Bulgaria, China, Cote d'Ivoire, Croatia, Dominican Republic, Ecuador, Finland, Ghana, Guinea, Kuwait, Lithuania, Malaysia, Mexico, Norway, Oman, Senegal, South Africa, Spain.
- ***34 countries*** compared with *Simple Logistic Function*. 22 have less than 1000 cases: Antigua and Barbuda, Burundi, Central African Republic, Chad, Comoros, Diamond Princess, Equatorial Guinea, Ethiopia, Fiji, Gabon, Gambia, Guatemala, Guinea-Bissau, Maldives,

H_0	MAE	NAE	MBE	NBE	MSE	RMSE	NRMSE	R	$p > .99$ R^2	$p > .99$ R^2_{adj}	$p < .5$ ΔAIC	$\Delta > 0$ ΔBIC	H_1
SGF	117	0.0261	-0.14	-3.1e-05	591468	769	0.006759	0.999772	75.40%	74.87%	65.24%	65.24%	SLF
											17.11%	21.39%	DGF
											31.02%	37.43%	DLF
SLF	230	0.0512	0.02	3.6e-06	2473427	1573	0.028265	0.999048	74.33%	73.26%	34.76%	34.76%	SGF
											13.90%	18.18%	DGF
											17.11%	24.60%	DLF
DGF	60	0.0133	4.54	1.0e-03	150927	388	0.001725	0.999942	86.10%	85.56%	83.42%	78.61%	SGF
											86.63%	81.82%	SLF
											62.57%	62.57%	DLF
DLF	107	0.0238	-11.90	-2.6e-03	588043	767	0.006720	0.999774	83.96%	82.89%	68.98%	62.57%	SGF
											82.89%	75.40%	SLF
											37.97%	37.43%	DGF

Sao Tome and Principe, Sudan, Suriname, Tanzania, Vietnam, West Bank and Gaza, Yemen, Zambia. ***12 have more than 1000 cases***: Bolivia, China, Djibouti, Ecuador, France, Ghana, Guinea, Iceland, Oman, Qatar, Senegal, South Africa.

- ***70 countries*** compared with *Double Logistic Function.* 41 have less than 1000 cases: Albania, Antigua and Barbuda, Bahamas, Belize, Bhutan, Burundi, Cabo Verde, Cambodia, Central African Republic, Chad, Comoros, Diamond Princess, Dominica, Equatorial Guinea, Eswatini, Fiji, Gambia, Georgia, Guatemala, Guinea-Bissau, Holy See, Kosovo, Laos, Liberia, Liechtenstein, Maldives, Mauritania, Monaco, Montenegro, Mozambique, Namibia, Niger, Papua New Guinea, Sao Tome and Principe, Sudan, Tanzania, Togo, Vietnam, West Bank and Gaza, Yemen, Zambia. ***29 have more than 1000 cases***: Australia, Austria, Azerbaijan, Belarus, Belgium, Bulgaria, China, Czechia, Djibouti, Ecuador, France, Greece, Guinea, Iceland, Iraq, Israel, Korea, South, Norway, Oman, Peru, Qatar, Russia, Senegal, Singapore, Spain, Sweden, Tunisia, Ukraine, Uzbekistan.

4.7 Results

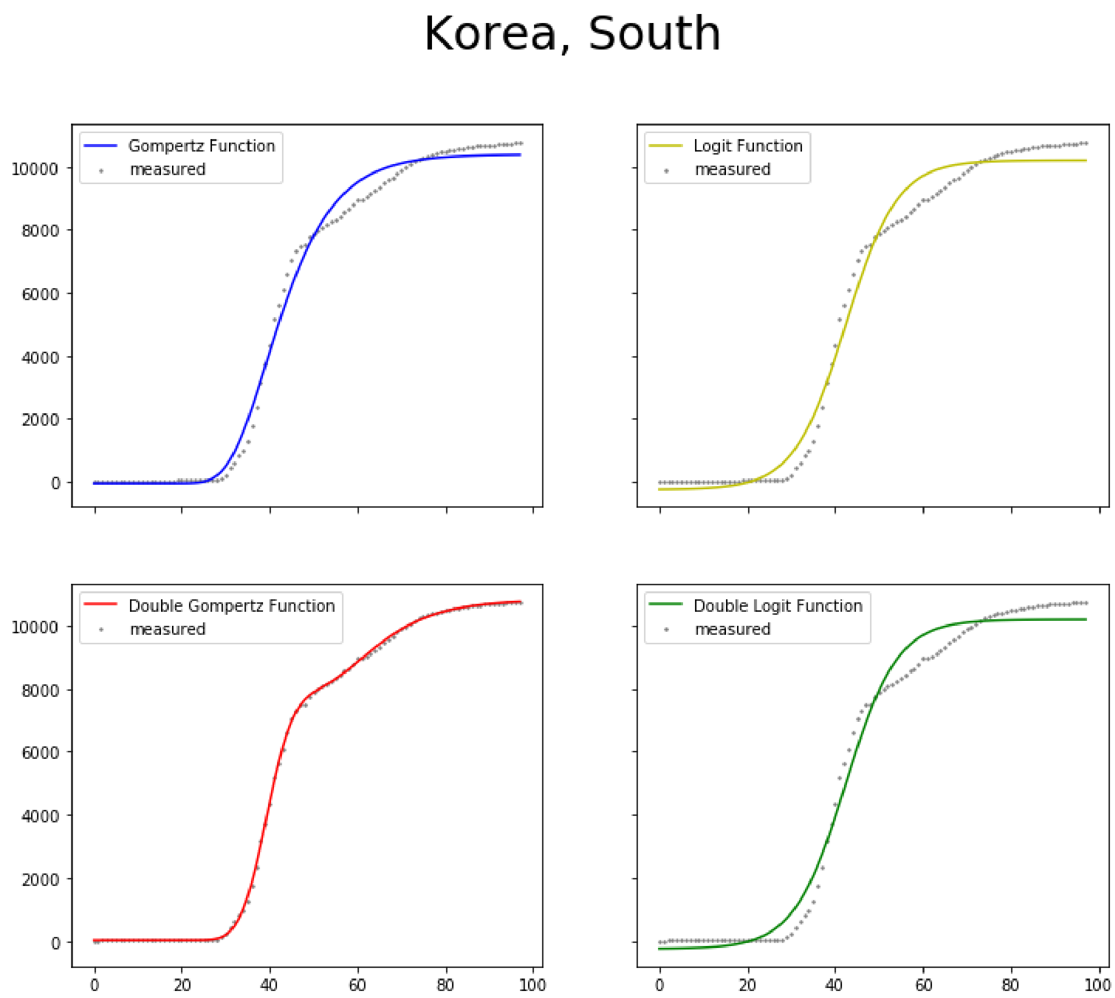
All tests strongly confirmed *Double Gompertz Function* as the better fitting model for SARS-nCoV-2 cumulative cases curve fitting. Results also showed that **DGF** is not only much more fitting than models with less parameters (**SLF** and **SGF**) as expected but also compared to *Double Logistic Function* with the same degrees of freedom.

5 Conclusions

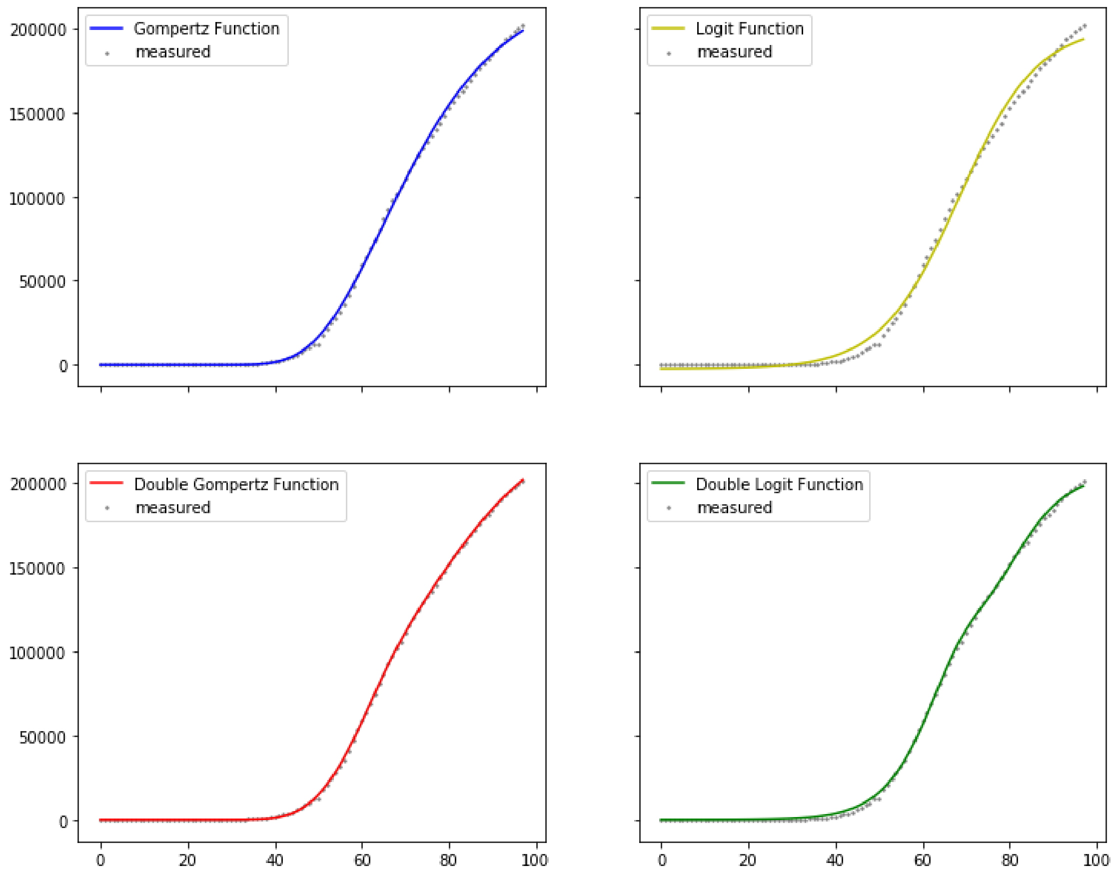
Among the compared models *Double Gompertz Function* has showed the best results and scores fitting data of SARS-nCoV-2 cumulative cases, suggesting that this model should be studied more deeply (possibly improved) and compared to other existing models for further analysis, including forecasting capabilities.

6 Plots

6.1 Fit examples

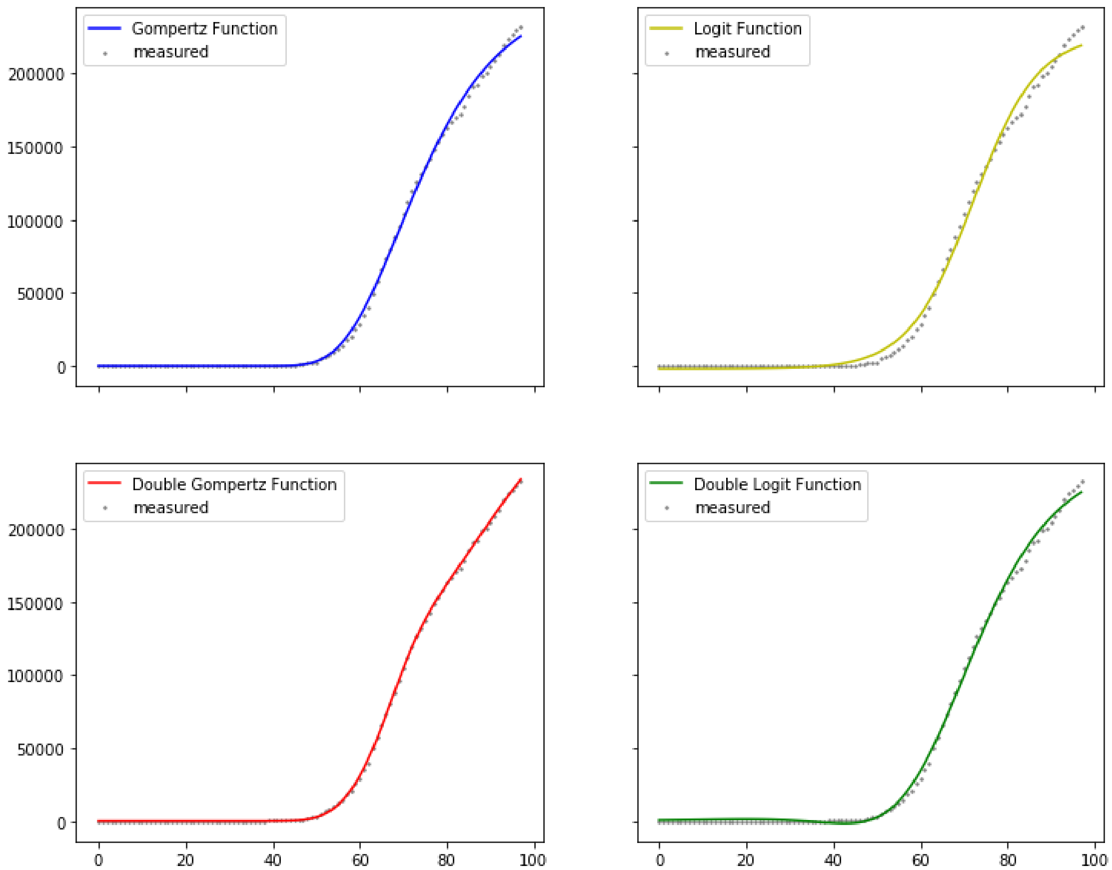


Italy

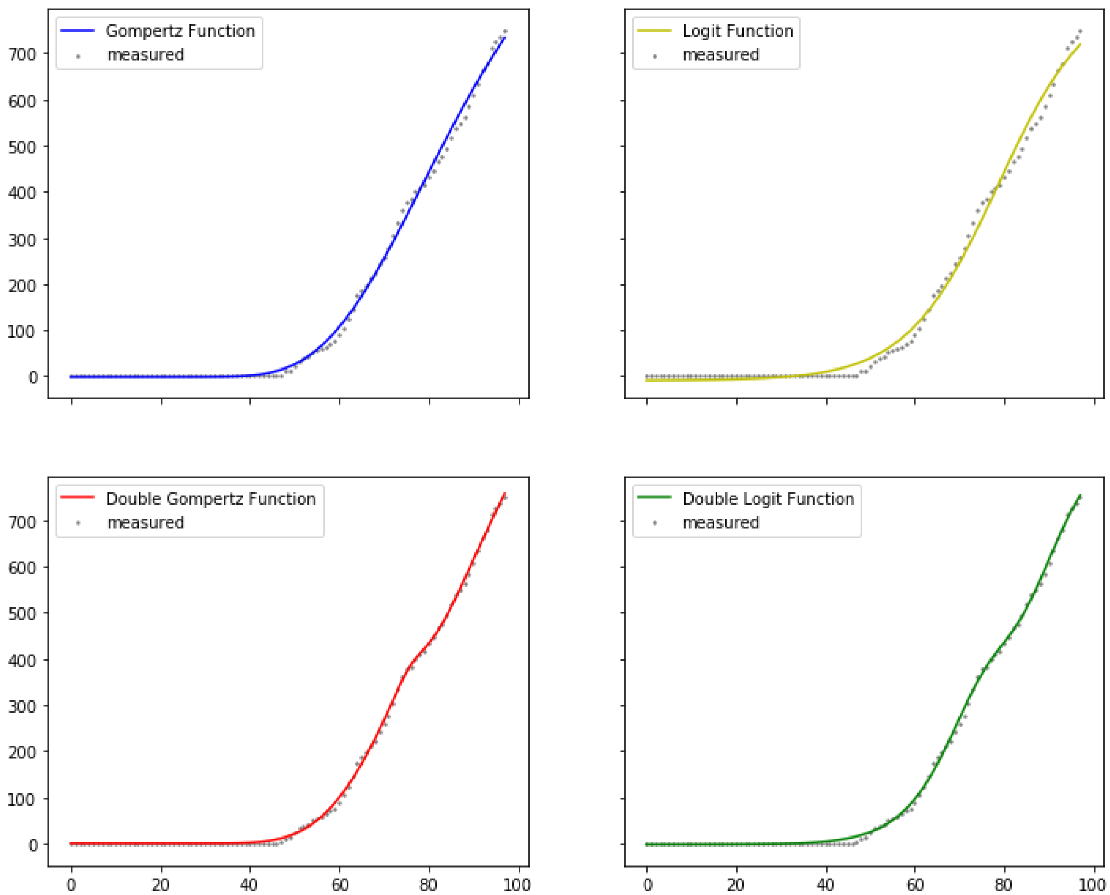


Gompertz Function
measured
Logit Function
measured
Double Gompertz Function
measured
Double Logit Function
measured
200000
150000
100000
50000
0
0
20
40
60
80
100

Spain

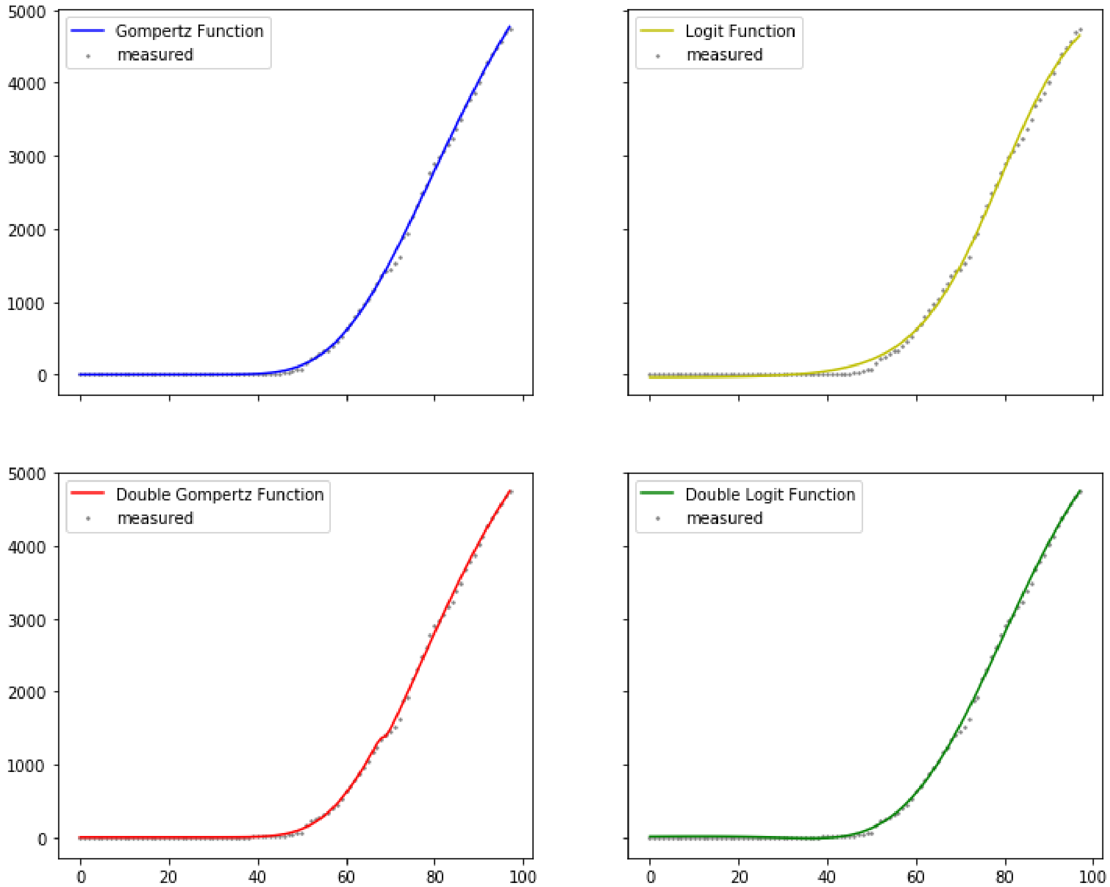

Gompertz Function
measured
Logit Function
measured
Double Gompertz Function
measured
Double Logit Function
measured
200000
150000
100000
50000
0
0
20
40
60
80
100

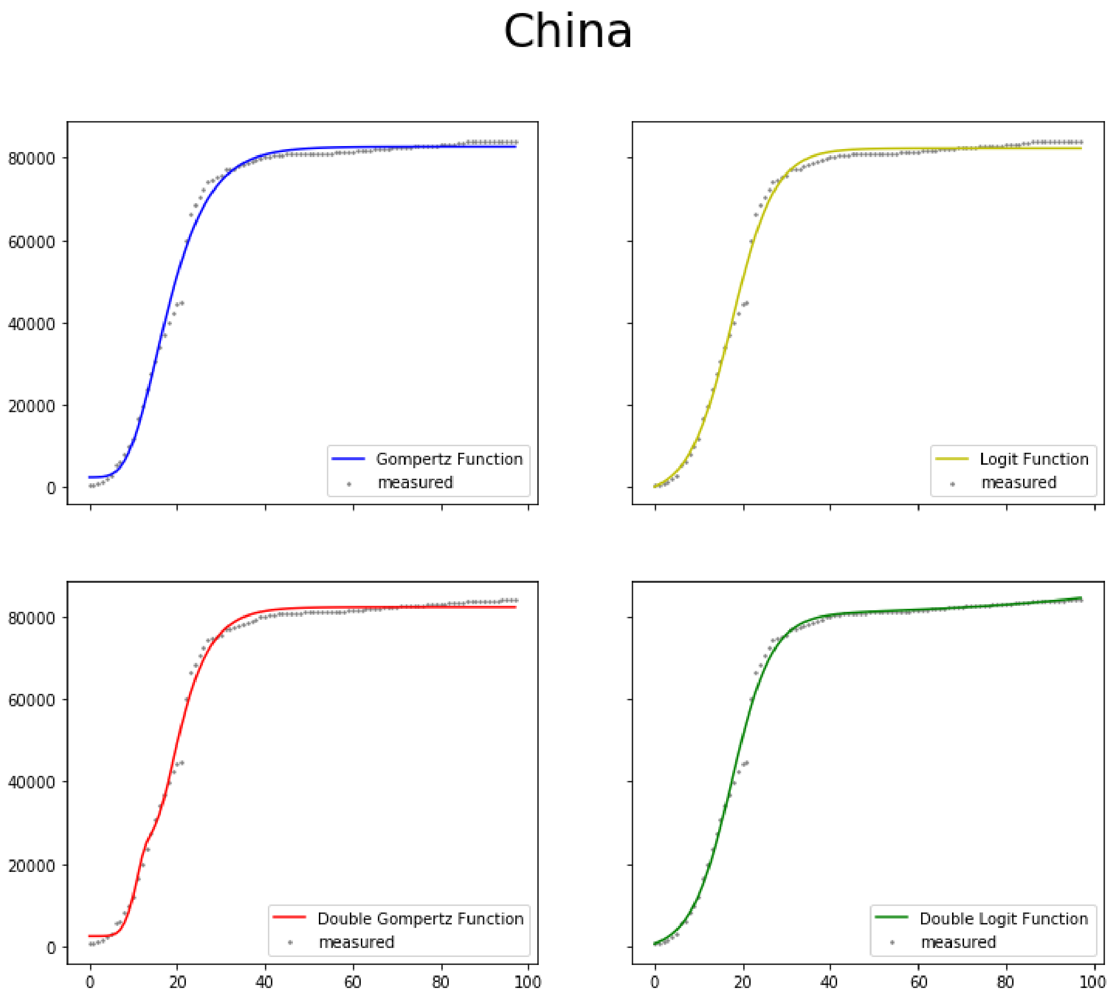
Albania



Gompertz Function
measured
Logit Function
measured
Double Gompertz Function
measured
Double Logit Function
measured
0
100
200
300
400
500
600
700
0
20
40
60
80
100

Finland


Gompertz Function
measured
Logit Function
measured
Double Gompertz Function
measured
Double Logit Function
measured
5000
4000
3000
2000
1000
0
0
20
40
60
80
100



7 Appendix

7.1 Statistics formulae

In all formulae we assume: y as the observed (measured) values and $\hat{y}$ as expected (predicted by fitting model) values; n is the number of values and n_{var} the number of model's variable parameters.

- Unbiased Mean:

$$\bar{y} = \frac{\sum y}{n}$$

- Unbiased Variance:

$$S_{n-1}^2 = \frac{\sum (y - \bar{y})^2}{n - 1}$$

- Biased Variance:

$$S_n^2 = \frac{\sum (y - \bar{y})^2}{n}$$

- Standard Deviation

$$S = \sqrt{S^2}$$

- **CV** Coefficiente of Variation or **RSD** Relative Standard Deviation

$$\mathbf{CV} = \frac{S}{|\bar{y}|}$$

- **MBE**: Mean Bias Error

$$\mathbf{MBE} = \frac{\sum (y - \hat{y})}{n}$$

- **MAE**: Mean Absolute Error (aka Mean Deviation)

$$\mathbf{MAE} = \frac{\sum |y - \hat{y}|}{n}$$

- **NBE**: Normalized Bias Error

$$\mathbf{NBE} = \frac{\sum (y - \hat{y})}{\sum y}$$

- **NAE**: Normalized Absolute Error (aka Normalized Mean Deviation)

$$\mathbf{NAE} = \frac{\sum |y - \hat{y}|}{\sum y}$$

- **RSS**: Residual Sum of Squares

$$\mathbf{RSS} = \sum (y - \hat{y})^2$$

- **TSS**: Total Sum of Squares

$$\mathbf{TSS} = \sum \left(y - \frac{\sum y}{n} \right)^2$$

- **MSE**: Mean Squared Error

$$\mathbf{MSE} = \sigma^2 = \frac{\mathbf{RSS}}{n}$$

- **RMSE**: Root Mean Squared Error (aka Standard Deviation)

$$\mathbf{RMSE} = \sigma = \sqrt{\mathbf{MSE}}$$

- **NRMSE**: Normalized Root Mean Squared Error (aka Normalized Standard Deviation)

$$\mathbf{NRMSE} = \sigma_\nu = \frac{\mathbf{RMSE}}{\sum y}$$

- **RRMSE**: Relative Root Mean Squared Error (aka Relative Standard Deviation or Coefficient of Variation)

$$\mathbf{RRMSE} = \frac{\mathbf{RMSE}}{|\bar{y}|} = \frac{\mathbf{RMSE}}{\left|\frac{\sum y}{n}\right|}$$

- R^2: Coefficient of Determination:

$$R^2 = 1 - \frac{\mathbf{RSS}}{\mathbf{TSS}}$$

- R^2_{adj}: Adjusted Coefficient of Determination:

$$R^2_{adj} = 1 - \frac{\mathbf{RSS}}{\mathbf{TSS}}\left(\frac{n-1}{n-n_{var}-1}\right)$$

- **Pearson R**: Pearson Correlation Coefficient

$$\mathbf{R} = \frac{\sum y\hat{y} - \frac{1}{n}\sum y \sum \hat{y}}{\sqrt{\sum y^2 - \frac{1}{n}\left(\sum y\right)^2}\sqrt{\sum \hat{y}^2 - \frac{1}{n}\left(\sum \hat{y}\right)^2}}$$

- **AIC**: Aikake Information Criterion

$$\mathbf{AIC} = n\ln\left(\frac{\mathbf{RSS}}{n}\right) + 2n_{var}$$

- **AIC p**: Aikake Information Criterion score (or weight) in **AIC** relative probability distribution space [FIG]:

$$\mathbf{AIC_p} = \frac{e^{-0.5\cdot(\mathbf{AIC_1}-\mathbf{AIC_0})}}{1+e^{-0.5\cdot(\mathbf{AIC_1}-\mathbf{AIC_0})}}$$

$$\mathbf{AIC_p} < .5 \Rightarrow P(H_0) > P(H_1)$$

- **BIC**: Bayesian Information Criterion

$$\mathbf{BIC} = n\ln\left(\frac{\mathbf{RSS}}{n}\right) + \ln(n)n_{var}$$

- Δ**BIC**: Bayesian Information Criterion difference [FIG]:

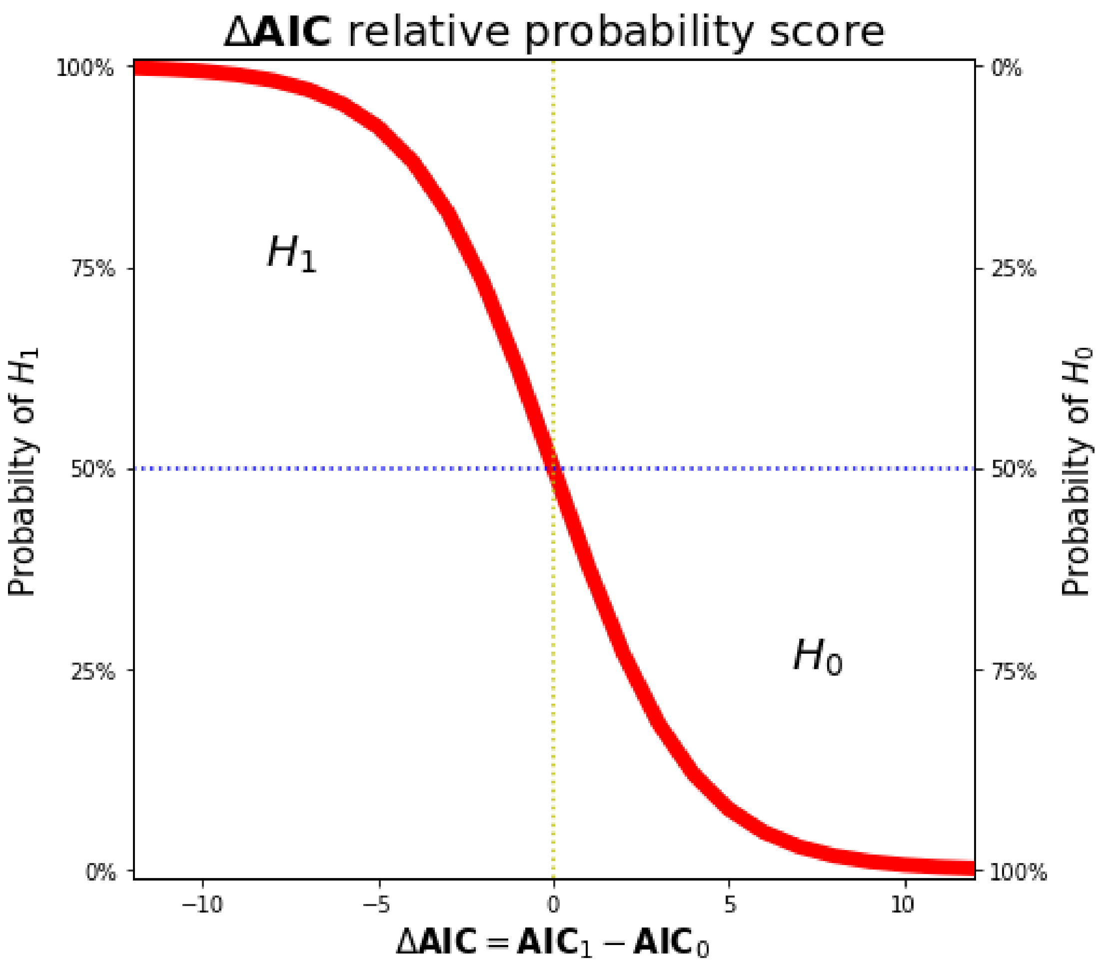
$$\Delta \mathbf{BIC} = \mathbf{BIC}_1 - \mathbf{BIC}_0$$

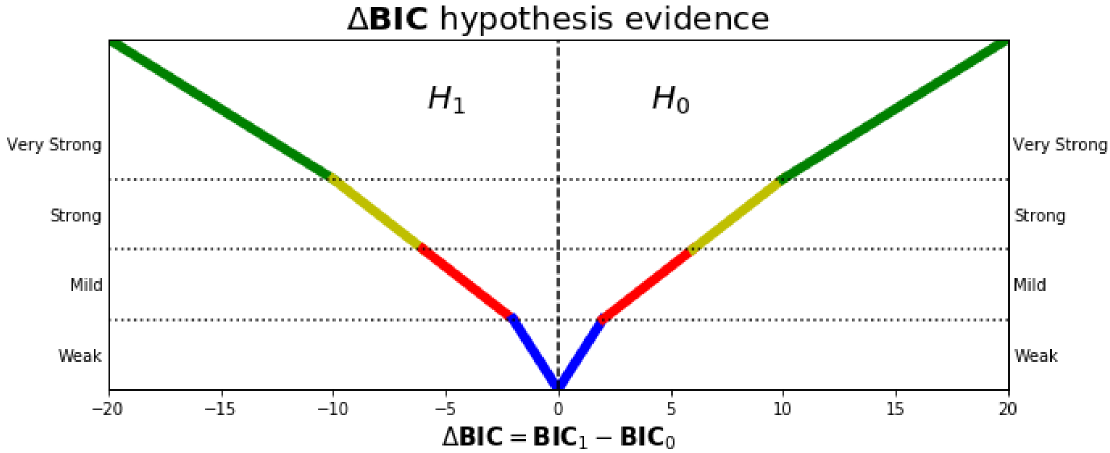
$$\Delta \mathbf{BIC} > 10 \Rightarrow H_0 \text{ Very strong evidence}$$

$$\Delta \mathbf{BIC} = (6, 10] \Rightarrow H_0 \text{ Strong evidence}$$

$$\Delta \mathbf{BIC} = (2, 6] \Rightarrow H_0 \text{ Mild evidence}$$

$$\Delta \mathbf{BIC} = (0, 2] \Rightarrow H_0 \text{ Weak evidence}$$





References

[1] J. H. University, "Johns hopkins university covid-19 github repository," *GitHub*, 2020. [Online]. Available: https://github.com/CSSEGISandData/COVID-19

[2] M. Pierini, "Max pierini covid-19 github repository," *GitHub*, 2020. [Online]. Available: https://github.com/maxdevblock/covid-19-time-series

[3] C. J. Willmott and K. Matsuura, "On the use of dimensioned measures of error to evaluate the performance of spatial interpolators," *International Journal of Geographical Information Science*, vol. 20, no. 1, pp. 89–102, 2006.

[4] N. Hofstra, M. Haylock, M. New, P. Jones, and C. Frei, "Comparison of six methods for the interpolation of daily, european climate data," *Journal of Geophysical Research: Atmospheres*, vol. 113, no. D21, 2008.

[5] A. H. Murphy and E. S. Epstein, "Skill scores and correlation coefficients in model verification," *Monthly weather review*, vol. 117, no. 3, pp. 572–582, 1989.

[6] A. Zhab Bhat, "Curve fitting: Linear, cubic, polynomial (1-5), pie- cewise, goodness of fit and regression analysis in python." [Online]. Available: https://projects.skill-lync.com/projects/Curve-FItting-In-Python-Assignment-89826

[7] H. Motulsky and A. Christopoulos, "Comparing models and curves, graphpad software," 2002.

[8] F. J. Fabozzi, S. M. Focardi, S. T. Rachev, and B. G. Arshanapalli, *The basics of financial econometrics: Tools, concepts, and asset management applications.* John Wiley & Sons, 2014.