

## modelacion\_chile

November 18, 2025

```
[2]: # Limpieza de la base de datos
import pandas as pd
import numpy as np

df = pd.read_csv( # Lectura de datos
    "../data/raw/BBDD Población Chile (1b).xlsx - Tabla ajustada población .csv",
    index_col=0,
)

df = df.T # Transposicion

# notacion europea a float
for col in df.columns:
    df[col] = (
        df[col]
        .astype(str)
        .str.replace(".", "", regex=False)
        .str.replace(",", ".", regex=False)
    )

df = df.replace(["NaN", "nan", "NAN", "None", ""], np.nan)
df = df.fillna(0)

df = df.apply(pd.to_numeric)

print(df)

df.to_csv("../data/output.csv")
```

	Población total de jóvenes (0-14)	Población total de adultos (15-64) \
2002	4156812.0	10288683.0
2003	4116188.0	10456445.0
2004	4057629.0	10640163.0
2005	3991476.0	10826184.0
2006	3921521.0	11017529.0
2007	3864328.0	11200341.0

2008	3823083.0	11375175.0
2009	3792259.0	11541067.0
2010	3767767.0	11697176.0
2011	3748933.0	11851404.0
2012	3732105.0	11998670.0
2013	3712426.0	12125123.0
2014	3698929.0	12251374.0
2015	3695756.0	12369304.0
2016	3692751.0	12488678.0
2017	3689702.0	12658694.0
2018	3696140.0	12890070.0
2019	3714172.0	13132822.0
2020	3738038.0	13361656.0
2021	3745665.0	13473999.0
2022	3739366.0	13528576.0
2023	3722867.0	13573894.0
2024	3698025.0	13619165.0

	Poblacion total mayores (65+ años)	dJ	dA	dM
2002	1246206.0	0	0.0	0.0
2003	1284344.0	-40624	167762.0	38138.0
2004	1324341.0	-58559	183718.0	39997.0
2005	1365829.0	-66153	186021.0	41488.0
2006	1408840.0	-69955	191345.0	43011.0
2007	1453264.0	-57193	182812.0	44424.0
2008	1499496.0	-41245	174834.0	46232.0
2009	1547752.0	-30824	165892.0	48256.0
2010	1598984.0	-24492	156109.0	51232.0
2011	1653822.0	-18834	154228.0	54838.0
2012	1712716.0	-16828	147266.0	58894.0
2013	1774353.0	-19679	126453.0	61637.0
2014	1837314.0	-13497	126251.0	62961.0
2015	1906363.0	-3173	117930.0	69049.0
2016	1985718.0	-3005	119374.0	79355.0
2017	2070796.0	-3049	170016.0	85078.0
2018	2165195.0	6438	231376.0	94399.0
2019	2260222.0	18032	242752.0	95027.0
2020	2358616.0	23866	228834.0	98394.0
2021	2458699.0	7627	112343.0	100083.0
2022	2560621.0	-6299	54577.0	101922.0
2023	2664128.0	-16499	45318.0	103507.0
2024	2769187.0	-24842	45271.0	105059.0

```
[26]: # regresion
import statsmodels.api as sm

# Modelo juvenes
```

```

x_J = df[["Población total de adultos (15-64)", "Población total de jóvenes (0-14)"]]
y_J = df["dJ"]
#x_J=sm.add_constant(x_J,prepend=False)
modelo_J = sm.OLS(y_J, x_J).fit()
coef_j = modelo_J.params.to_dict()

# Modelo adultos

x_A = df[["Población total de jóvenes (0-14)", "Población total de adultos (15-64)"]]
y_A = df["dA"]
# x_A=sm.add_constant(x_A,prepend=False)
modelo_A = sm.OLS(y_A, x_A).fit()
coef_A = modelo_A.params.to_dict()

# Modelo mayores

x_E = df[["Población total de adultos (15-64)", "Poblacion total mayores (65+años)"]]
y_E = df["dM"]
#x_E = sm.add_constant(x_E, prepend=False)
modelo_E = sm.OLS(y_E, x_E).fit()
coef_E = modelo_E.params.to_dict()

print(modelo_J.summary())
print(modelo_A.summary())
print(modelo_E.summary())

```

#### OLS Regression Results

```

=====
=====
Dep. Variable:          dJ    R-squared (uncentered):
0.650
Model:                  OLS    Adj. R-squared (uncentered):
0.616
Method:                 Least Squares    F-statistic:
19.48
Date:                   Tue, 18 Nov 2025    Prob (F-statistic):
1.64e-05
Time:                   12:35:23    Log-Likelihood:
-259.39
No. Observations:      23    AIC:
522.8
Df Residuals:          21    BIC:
525.1

```

```

Df Model:                2
Covariance Type:         nonrobust
=====
=====
                                coef    std err          t      P>|t|
-----
[0.025    0.975]
-----
Población total de adultos (15-64)    0.0113    0.003    3.872    0.001
0.005    0.017
Población total de jóvenes (0-14)   -0.0411    0.009   -4.423    0.000
-0.060   -0.022
=====
Omnibus:                4.824    Durbin-Watson:           0.477
Prob(Omnibus):          0.090    Jarque-Bera (JB):        2.800
Skew:                   0.684    Prob(JB):                0.247
Kurtosis:               4.024    Cond. No.                29.6
=====

```

Notes:

[1]  $R^2$  is computed without centering (uncentered) since the model does not contain a constant.  
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

#### OLS Regression Results

```

=====
=====
Dep. Variable:            dA    R-squared (uncentered):
0.847
Model:                    OLS    Adj. R-squared (uncentered):
0.832
Method:                   Least Squares    F-statistic:
57.93
Date:                     Tue, 18 Nov 2025    Prob (F-statistic):
2.83e-09
Time:                     12:35:23    Log-Likelihood:
-286.27
No. Observations:         23    AIC:
576.5
Df Residuals:             21    BIC:
578.8
Df Model:                 2
Covariance Type:         nonrobust
=====
=====
                                coef    std err          t      P>|t|
-----
[0.025    0.975]
-----

```

```

-----
Población total de jóvenes (0-14)      0.0392      0.030      1.313      0.203
-0.023      0.101
Población total de adultos (15-64)    -0.0004      0.009      -0.043      0.966
-0.020      0.019
=====
Omnibus:                               3.462      Durbin-Watson:          0.614
Prob(Omnibus):                         0.177      Jarque-Bera (JB):      2.013
Skew:                                 -0.702      Prob(JB):              0.365
Kurtosis:                             3.364      Cond. No.              29.6
=====

```

Notes:

[1]  $R^2$  is computed without centering (uncentered) since the model does not contain a constant.  
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

OLS Regression Results

```

=====
=====
Dep. Variable:          dM      R-squared (uncentered):
0.985
Model:                  OLS      Adj. R-squared (uncentered):
0.983
Method:                 Least Squares      F-statistic:
677.0
Date:                   Tue, 18 Nov 2025      Prob (F-statistic):
8.53e-20
Time:                   12:35:23      Log-Likelihood:
-241.67
No. Observations:      23      AIC:
487.3
Df Residuals:          21      BIC:
489.6
Df Model:               2
Covariance Type:       nonrobust
=====
=====

```

```

-----
-----
coef      std err      t      P>|t|
-----
[0.025      0.975]
-----
Población total de adultos (15-64)    -0.0046      0.001      -4.581      0.000
-0.007      -0.003
Poblacion total mayores (65+ años)    0.0655      0.006      10.294      0.000
0.052      0.079
=====
Omnibus:                               31.255      Durbin-Watson:          0.834

```

Prob(Omnibus):	0.000	Jarque-Bera (JB):	69.701
Skew:	-2.444	Prob(JB):	7.32e-16
Kurtosis:	9.988	Cond. No.	41.0

=====

Notes:

[1]  $R^2$  is computed without centering (uncentered) since the model does not contain a constant.

[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[28]: *# Graficación por Runge-Kutta 4 (consistente en el orden del estado)*

```
import numpy as np
import matplotlib.pyplot as plt

def sistema_edos(t, y):
    # y = [J, A, E]
    jt, at, et = y

    # COEFICIENTES
    c1 = 0.0113 # tasa de natalidad
    c2 = 0.0411 # tasa de mortalidad jóvenes
    #const1 = -2.929e+05
    tj = 0.0392 # transicion jovenes
    c3 = -0.0004 # tasa de mortalidad adultos
    const2 = 1.442e+06
    ta = -0.0046 # transicion adultos
    c4 = -0.0655 # tasa de mortalidad mayores
    #const3 = -2.048e+05

    # ECUACIONES
    dyJ = c1*at - c2*jt + const1
    dyA = tj*jt - c3*at + const2
    dyE = ta*at - c4*et + const3
    return [dyJ, dyA, dyE]

def RK4(func, y0, t0, tf, h):
    t_values = np.arange(t0, tf + h, h)
    n = len(t_values)
    y_values = np.zeros((n, len(y0)), dtype=float)
    y_values[0] = y0

    for i in range(1, n):
        k1 = np.array(func(t_values[i-1], y_values[i-1]))
        k2 = np.array(func(t_values[i-1] + h/2, y_values[i-1] + h*k1/2))
        k3 = np.array(func(t_values[i-1] + h/2, y_values[i-1] + h*k2/2))
        k4 = np.array(func(t_values[i-1] + h, y_values[i-1] + h*k3))
```

```

        y_values[i] = y_values[i-1] + h*(k1 + 2*k2 + 2*k3 + k4)/6

    return t_values, y_values

# y0 = [J, A, E] en 2002
y0 = [4156812.0, 10288683.0, 1246206.0]
t0, tf, h = 0, 22, 0.01

t, y = RK4(sistema_edos, y0, t0, tf, h)

years = 2002 + t
fig, ax = plt.subplots(1, 1, figsize=(12, 8))
# Datos del modelo

ax.plot(years, y[:, 0], 'b-', linewidth=2, label='Población de jóvenes')
ax.plot(years, y[:, 1], 'r-', linewidth=2, label='Población de adultos')
ax.plot(years, y[:, 2], 'g-', linewidth=2, label='Población de adultos mayores')

# Datos reales

years_real = df.index.astype(int)

ax.plot(years_real, df["Población total de jovenes_
    ↳(0-14)"], 'bo', markersize=6, label="Real jovenes")
ax.plot(years_real, df["Población total de adultos_
    ↳(15-64)"], 'ro', markersize=6, label="Real adultos")
ax.plot(years_real, df["Poblacion total mayores (65+_
    ↳años)"], 'go', markersize=6, label="Real mayores")

ax.set_title('Población total por grupo')
ax.set_xlabel('Año')
ax.set_ylabel('Población')
ax.grid(True)
ax.legend()
plt.show()

```

