Using the Chickens and Eggs Model to conduct sensitivy analysis in Python

Developed by: Robinson Salazar Rua and Peter Hovmand

Last updated: December 16, 2023

Install the packages

In [ ]:
```
!pip install pysd
!pip install netCDF4
```

Load the libraries

In [28]:
```
import os
import pysd
import itertools
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Set the working directory

In [29]:
```
os.chdir("C:/Users/rss188/Desktop/Chicken model/SFD chicken model")
os.getcwd()
```

Out[29]:
```
'C:\\Users\\rss188\\Desktop\\Chicken model\\SFD chicken model'
```

Read the original model and print the results

In [30]:
```
model= pysd.read_vensim("C:/Users/rss188/Desktop/Chicken model/SFD chicken model/SFD chicken model V2.mdl")

results = model.run()

print(results.head())
print(results.tail())
```

```
   FINAL TIME  INITIAL TIME  SAVEPER  TIME STEP  Births  Chickens  \
0          50             0        1          1   200.0    1000.0
1          50             0        1          1   200.0    1000.0
2          50             0        1          1   200.0    1000.0
3          50             0        1          1   200.0    1000.0
4          50             0        1          1   200.0    1000.0

   Cross roading  Death risk  Deaths  Egg production    Eggs  \
0            1.0         0.2   200.0           200.0  1000.0
1            1.0         0.2   200.0           200.0  1000.0
2            1.0         0.2   200.0           200.0  1000.0
3            1.0         0.2   200.0           200.0  1000.0
4            1.0         0.2   200.0           200.0  1000.0

   Fertility effect  Incubation time  Initial chicken population  \
0               0.2               5                        1000
1               0.2               5                        1000
2               0.2               5                        1000
3               0.2               5                        1000
4               0.2               5                        1000

   Initial number of eggs  Max chicken capacity  Normal death risk  \
0                    1000                  1000                0.2
1                    1000                  1000                0.2
2                    1000                  1000                0.2
3                    1000                  1000                0.2
4                    1000                  1000                0.2

   Normal fertility rate  Stress
0                    0.2     1.0
1                    0.2     1.0
2                    0.2     1.0
3                    0.2     1.0
4                    0.2     1.0
    FINAL TIME  INITIAL TIME  SAVEPER  TIME STEP  Births  Chickens  \
46          50             0        1          1   200.0    1000.0
47          50             0        1          1   200.0    1000.0
48          50             0        1          1   200.0    1000.0
49          50             0        1          1   200.0    1000.0
50          50             0        1          1   200.0    1000.0

    Cross roading  Death risk  Deaths  Egg production    Eggs  \
46            1.0         0.2   200.0           200.0  1000.0
47            1.0         0.2   200.0           200.0  1000.0
48            1.0         0.2   200.0           200.0  1000.0
49            1.0         0.2   200.0           200.0  1000.0
50            1.0         0.2   200.0           200.0  1000.0

    Fertility effect  Incubation time  Initial chicken population  \
46               0.2               5                        1000
47               0.2               5                        1000
48               0.2               5                        1000
49               0.2               5                        1000
50               0.2               5                        1000

    Initial number of eggs  Max chicken capacity  Normal death risk  \
46                    1000                  1000                0.2
47                    1000                  1000                0.2
48                    1000                  1000                0.2
49                    1000                  1000                0.2
50                    1000                  1000                0.2

    Normal fertility rate  Stress
46                    0.2     1.0
47                    0.2     1.0
48                    0.2     1.0
49                    0.2     1.0
50                    0.2     1.0
```

Set new conditions and print the results

In [31]:

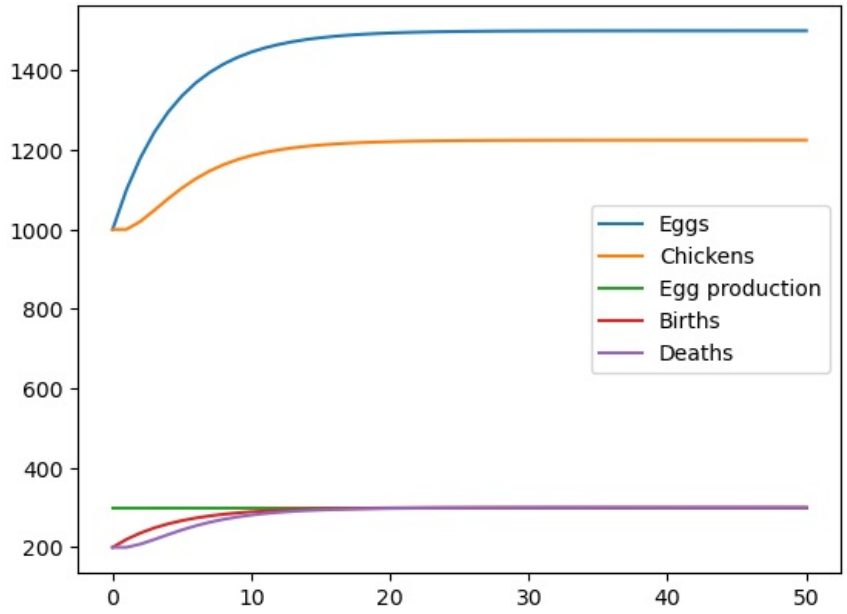
```
model= pysd.read_vensim("C:/Users/rss188/Desktop/Chicken model/SFD chicken model/SFD chicken model V2.mdl")

params = {'Normal fertility rate': 0.3, 'Normal death risk': 0.2}
return_columns = ['Eggs', 'Chickens', 'Egg production', 'Births', 'Deaths']
return_timestamps = range(51)

results = model.run(params=params, return_columns=return_columns, return_timestamps=return_timestamps, initial_

print(results.head())
print(results.tail())
```

```
     Eggs     Chickens  Egg production  Births      Deaths
0  1000.0  1000.000000           300.0  200.00  200.000000
1  1100.0  1000.000000           300.0  220.00  200.000000
2  1180.0  1020.000000           300.0  236.00  208.080000
3  1244.0  1047.920000           300.0  248.80  219.627265
4  1295.2  1077.092735           300.0  259.04  232.025752
           Eggs     Chickens  Egg production      Births      Deaths
46  1499.982578  1224.732851           300.0  299.996516  299.994111
47  1499.986062  1224.735255           300.0  299.997212  299.995289
48  1499.988850  1224.737179           300.0  299.997770  299.996231
49  1499.991080  1224.738717           300.0  299.998216  299.996985
50  1499.992864  1224.739948           300.0  299.998573  299.997588
```

Plot the results from the new conditions

In [32]:
```
results.plot()
```

Out[32]:
```
<Axes: >
```



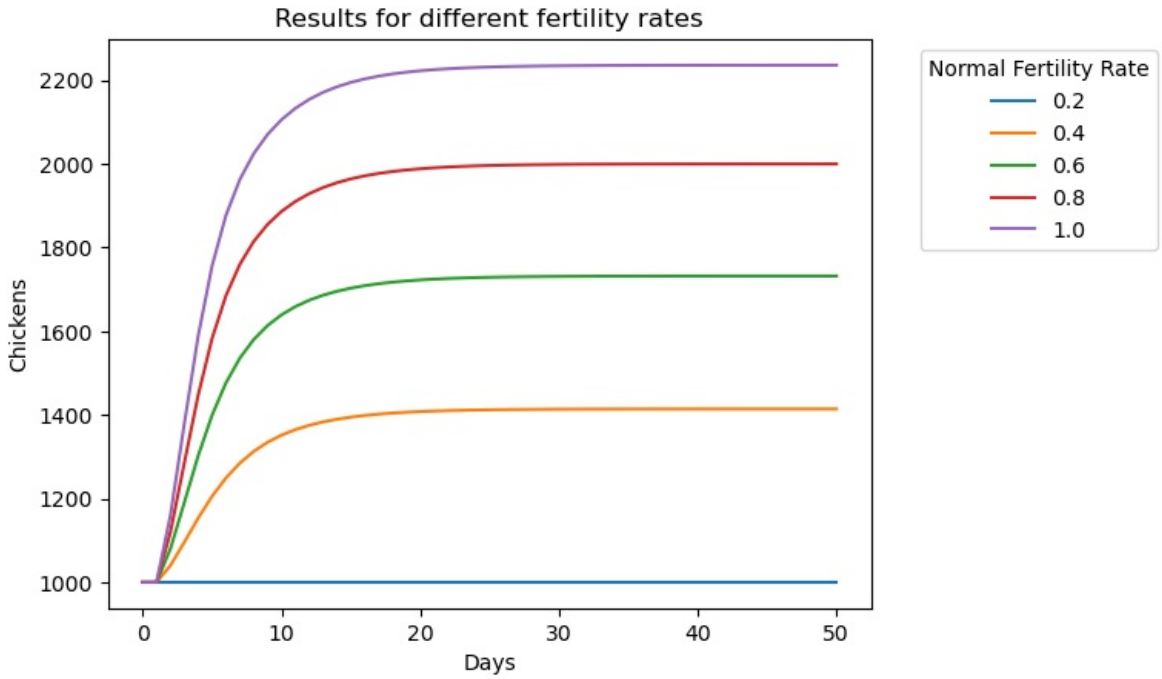
First sensitivy analysis (Normal fertility rate)

In [33]:
```
Fertility_Rate_Values = np.linspace(start=0.2, stop=1, num=5)

results = pd.DataFrame()

for rate in Fertility_Rate_Values:

    single_run_result = model.run(params={'Normal Fertility Rate': rate})

    results[str(round(rate, 2))] = single_run_result['Chickens']

print(results.head())
```

```
      0.2          0.4          0.6          0.8          1.0
0  1000.0  1000.000000  1000.000000  1000.000000  1000.000000
1  1000.0  1000.000000  1000.000000  1000.000000  1000.000000
2  1000.0  1040.000000  1080.000000  1120.000000  1160.000000
3  1000.0  1095.680000  1190.720000  1285.120000  1378.880000
4  1000.0  1153.177068  1302.357176  1447.613317  1589.017989
```

Plot the results from the first sensitivity analysis

In [34]:
```
results.plot()
plt.xlabel('Days')
plt.ylabel('Chickens')
plt.title('Results for different fertility rates')
plt.legend(title='Normal Fertility Rate', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



Second sensitivy analysis (Normal fertility rate and Normal death risk)

In [35]:

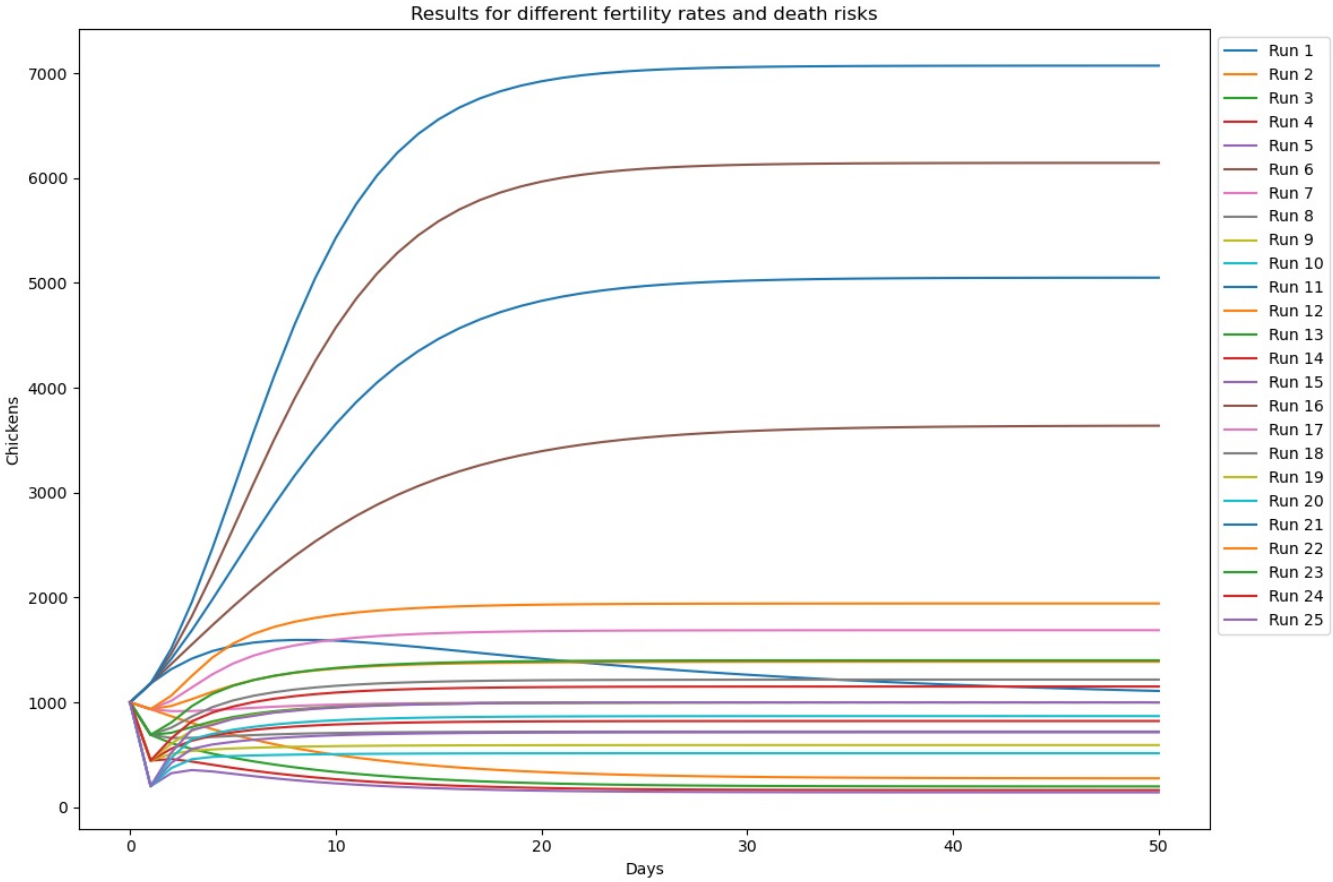
```
Fertility_Rate_Values = np.linspace(start=0.02, stop=1, num=5)
Death_Risk_Values = np.linspace(start=0.02, stop=1, num=5)


parameter_combinations = list(itertools.product(Fertility_Rate_Values, Death_Risk_Values))

results_list = []
run_number = 1  # Initialize run number

for Fertility_Rate, Death_Risk in parameter_combinations:

    single_run_result = model.run(params={'Normal Fertility Rate': Fertility_Rate, 'Normal Death Risk': Death_R

    # Store only necessary data with run number

    results_list.append({
        'Run Number': f'Run {run_number}',
        'Normal Fertility Rate': Fertility_Rate,
        'Normal Death Risk': Death_Risk,
        'Eggs': single_run_result['Eggs'],
        'Chickens': single_run_result['Chickens']
    })

    run_number += 1  # Increment run number for each iteration
    del single_run_result #delete variables that are no longer needed to free up memory


results = pd.DataFrame(results_list)

csv_file_path = r'C:/Users/rss188/Desktop/Chicken model/SFD chicken model/model_results.csv'
results.to_csv(csv_file_path, index=False)

print(results.head())
```

```
  Run Number  Normal Fertility Rate  Normal Death Risk  \
0      Run 1                   0.02              0.020
1      Run 2                   0.02              0.265
2      Run 3                   0.02              0.510
3      Run 4                   0.02              0.755
4      Run 5                   0.02              1.000

                                                 Eggs  \
0  0     1000.000000
1      820.000000
2      676...
1  0     1000.000000
1      820.000000
2      676...
2  0     1000.000000
1      820.000000
2      676...
3  0     1000.000000
1      820.000000
2      676...
4  0     1000.000000
1      820.000000
2      676...

                                             Chickens
0  0     1000.000000
1     1180.000000
2     1316...
1  0     1000.000000
1      935.000000
2      867...
2  0     1000.000000
1      690.000000
2      611...
3  0     1000.000000
1      445.000000
2      459...
4  0     1000.000000
1      200.000000
2      324...
```

Plot the results from the second sensitivity analysis

In [36]:

```
plt.figure(figsize=(12, 8))

for index, row in results.iterrows():
    plt.plot(row['Chickens'], label=row['Run Number'])

plt.xlabel('Days')
plt.ylabel('Chickens')
plt.title('Results for different fertility rates and death risks')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()
```



Third sensitivy analysis (Normal fertility rate, Normal death risk, Max chicken capacity)

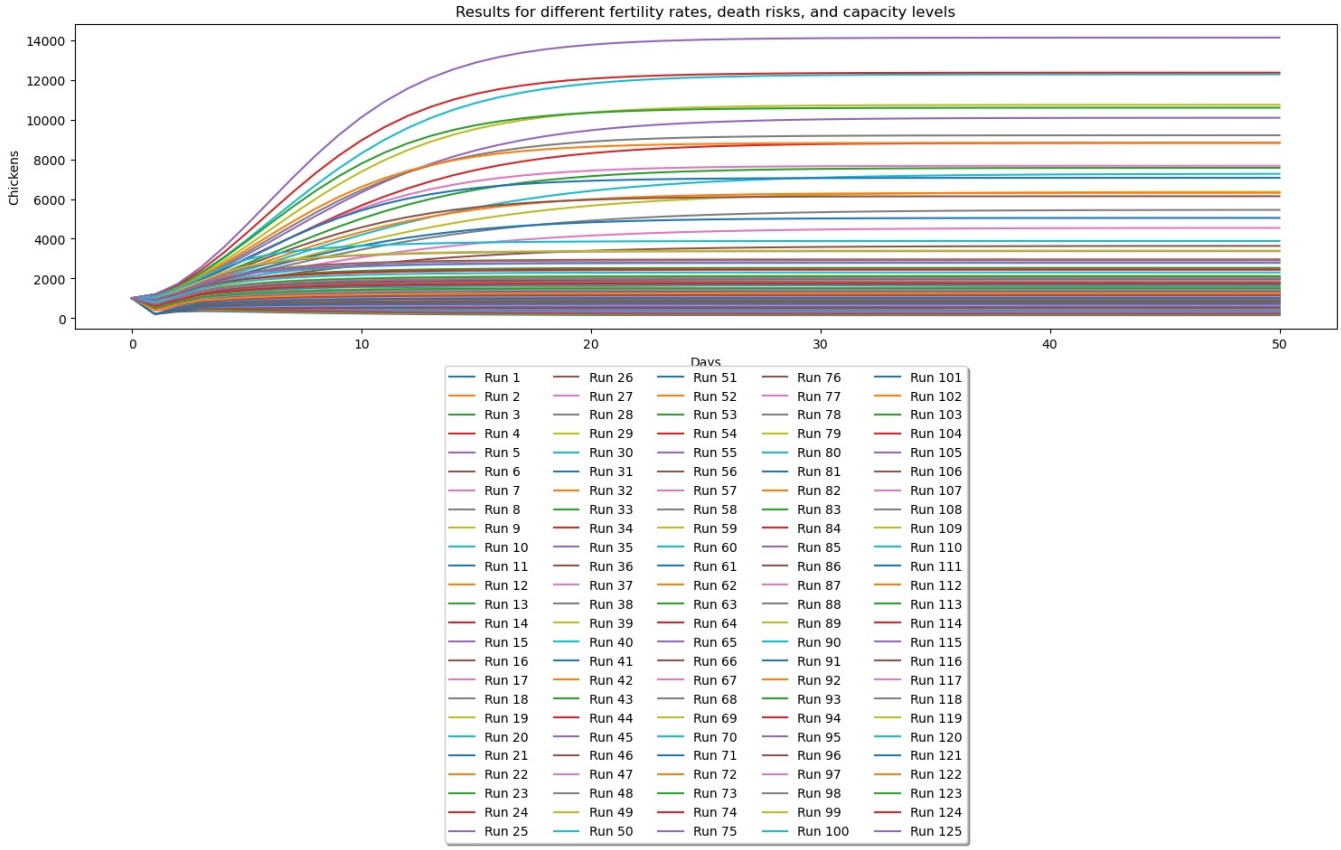
In [37]:

```
Fertility_Rate_Values = np.linspace(start=0.02, stop=1, num=5)
Death_Risk_Values = np.linspace(start=0.02, stop=1, num=5)
Chicken_Capacity_Values= np.linspace(start=1000, stop=2000, num=5)


parameter_combinations = list(itertools.product(Fertility_Rate_Values, Death_Risk_Values, Chicken_Capacity_Valu

results_list = []
run_number = 1  # Initialize run number

for Fertility_Rate, Death_Risk, Chicken_Capacity in parameter_combinations:

    single_run_result = model.run(params={'Normal Fertility Rate': Fertility_Rate, 'Normal Death Risk': Death_R

    # Store only necessary data with run number

    results_list.append({
        'Run Number': f'Run {run_number}',
        'Normal Fertility Rate': Fertility_Rate,
        'Normal Death Risk': Death_Risk,
        'Max chicken capacity': Chicken_Capacity,
        'Eggs': single_run_result['Eggs'],
        'Chickens': single_run_result['Chickens']
    })

    run_number += 1  # Increment run number for each iteration
    del single_run_result #delete variables that are no longer needed to free up memory


results = pd.DataFrame(results_list)

csv_file_path = r'C:/Users/rss188/Desktop/Chicken model/SFD chicken model/model_results.csv'
results.to_csv(csv_file_path, index=False)

print(results.head())
```

```
  Run Number  Normal Fertility Rate  Normal Death Risk  Max chicken capacity  \
0      Run 1                   0.02               0.02                1000.0
1      Run 2                   0.02               0.02                1250.0
2      Run 3                   0.02               0.02                1500.0
3      Run 4                   0.02               0.02                1750.0
4      Run 5                   0.02               0.02                2000.0

                                            Eggs  \
0  0     1000.000000
1      820.000000
2      676...
1  0     1000.000000
1      825.000000
2      685...
2  0     1000.000000
1      830.000000
2      694...
3  0     1000.000000
1      835.000000
2      703...
4  0     1000.000000
1      840.000000
2      712...

                                        Chickens
0  0     1000.000000
1     1180.000000
2     1316...
1  0     1000.000000
1     1184.000000
2     1326...
2  0     1000.000000
1     1186.666667
2     1333...
3  0     1000.000000
1     1188.571429
2     1339...
4  0     1000.000000
1     1190.000000
2     1343...
```

Plot the results from the third sensitivity analysis

In [38]:
```
plt.figure(figsize=(15, 10))

for index, row in results.iterrows():
    plt.plot(row['Chickens'], label=row['Run Number'])

plt.xlabel('Days')
plt.ylabel('Chickens')
plt.title('Results for different fertility rates, death risks, and capacity levels')
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.1), fancybox=True, shadow=True, ncol=5)
plt.tight_layout()
plt.show()
```



Fourth sensitivy analysis (Normal fertility rate, Normal death risk, Max chicken capacity, Initial number of eggs, Initial number of chickens)

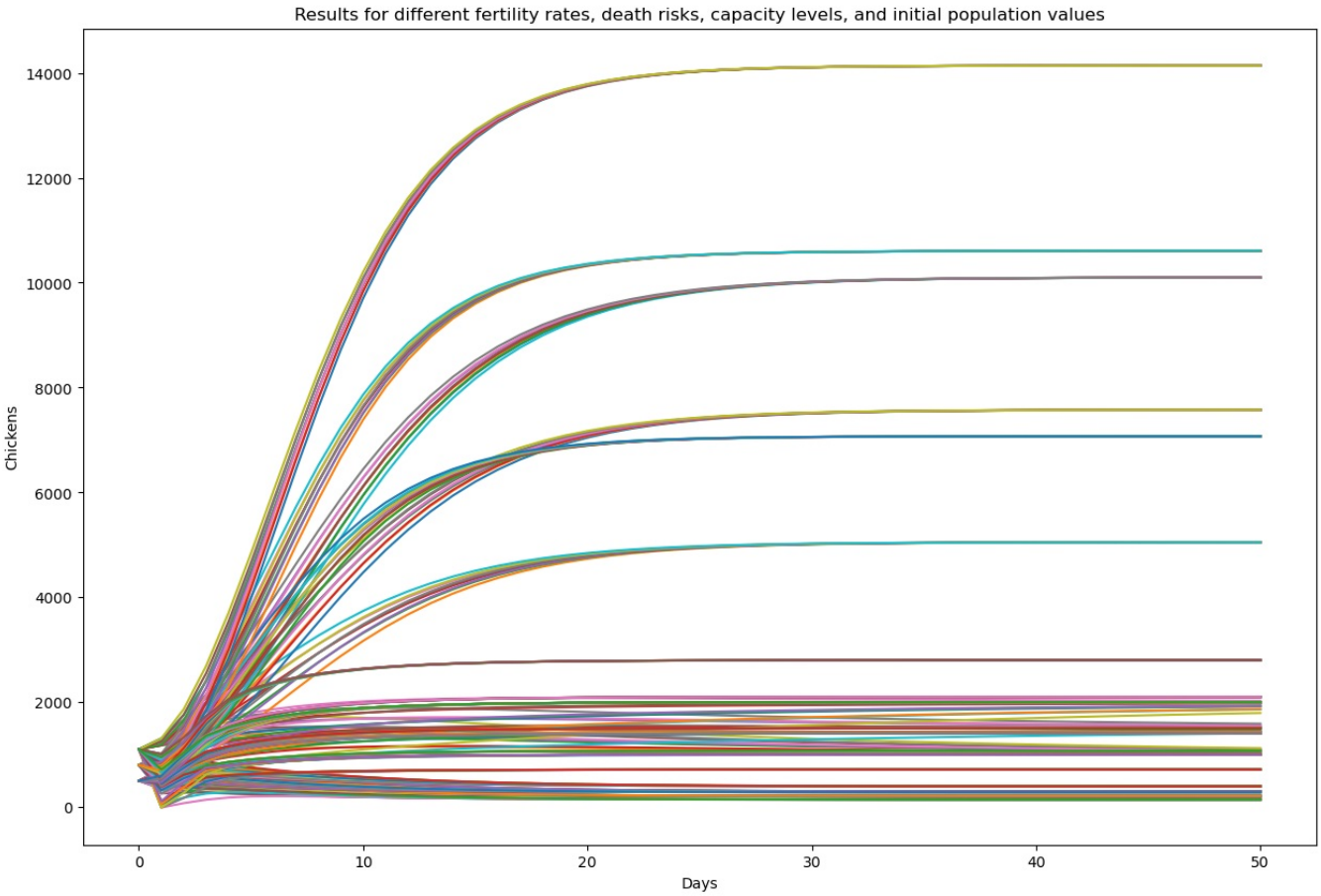
In [39]:

```
Fertility_Rate_Values = np.linspace(start=0.02, stop=1, num=3)
Death_Risk_Values = np.linspace(start=0.02, stop=1, num=3)
Chicken_Capacity_Values= np.linspace(start=1000, stop=2000, num=3)
Initial_Eggs_Values= np.linspace(start=500, stop=1100, num=3)
Initial_Chickens_Values= np.linspace(start=500, stop=1100, num=3)


parameter_combinations = list(itertools.product(Fertility_Rate_Values, Death_Risk_Values, Chicken_Capacity_Valu

results_list = []
run_number = 1  # Initialize run number

for Fertility_Rate, Death_Risk, Chicken_Capacity, Initial_Eggs, Initial_Chickens in parameter_combinations:

    single_run_result = model.run(params={'Normal Fertility Rate': Fertility_Rate, 'Normal Death Risk': Death_R

    # Store only necessary data with run number

    results_list.append({
        'Run Number': f'Run {run_number}',
        'Normal Fertility Rate': Fertility_Rate,
        'Normal Death Risk': Death_Risk,
        'Max chicken capacity': Chicken_Capacity,
        'Initial number of eggs': Initial_Eggs,
        'Initial chicken population': Initial_Chickens,
        'Eggs': single_run_result['Eggs'],
        'Chickens': single_run_result['Chickens']
    })

    run_number += 1  # Increment run number for each iteration
    del single_run_result #delete variables that are no longer needed to free up memory


results = pd.DataFrame(results_list)

csv_file_path = r'C:/Users/rss188/Desktop/Chicken model/SFD chicken model/model_results.csv'
results.to_csv(csv_file_path, index=False)

print(results.head())
```

```
  Run Number  Normal Fertility Rate  Normal Death Risk  Max chicken capacity  \
0      Run 1                   0.02               0.02                1000.0
1      Run 2                   0.02               0.02                1000.0
2      Run 3                   0.02               0.02                1000.0
3      Run 4                   0.02               0.02                1000.0
4      Run 5                   0.02               0.02                1000.0

   Initial number of eggs  Initial chicken population  \
0                   500.0                       500.0
1                   500.0                       800.0
2                   500.0                      1100.0
3                   800.0                       500.0
4                   800.0                       800.0

                                              Eggs  \
0  0     500.000000
1     420.000000
2     356.00...
1  0     500.000000
1     420.000000
2     356.00...
2  0     500.000000
1     420.000000
2     356.00...
3  0     800.000000
1     660.000000
2     548.00...
4  0     800.000000
1     660.000000
2     548.00...

                                          Chickens
0  0     500.000000
1     595.000000
2     671.91...
1  0      800.000000
1      887.200000
2      955...
2  0     1100.000000
1     1175.800000
2     1232...
3  0      500.000000
1      655.000000
2      778...
4  0      800.000000
1      947.200000
2     1061...
```

Plot the results from the fourth sensitivity analysis

In [40]:
```
plt.figure(figsize=(15, 10))

for index, row in results.iterrows():
    plt.plot(row['Chickens'], label=row['Run Number'])

plt.xlabel('Days')
plt.ylabel('Chickens')
plt.title('Results for different fertility rates, death risks, capacity levels, and initial population values')
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.1), fancybox=True, shadow=True, ncol=5)
plt.show()
```



Run 1 Run 2 Run 3 Run 4 Run 5 Run 6 Run 7 Run 8 Run 9 Run 10 Run 11 Run 12 Run 13 Run 14 Run 15 Run 16 Run 17 Run 18 Run 19 Run 20 Run 21 Run 22 Run 23 Run 24 Run 25 Run 26 Run 27 Run 28 Run 29 Run 30 Run 31 Run 32 Run 33 Run 34 Run 35 Run 36 Run 37 Run 38 Run 39 Run 40 Run 41 Run 42 Run 43 Run 44 Run 45 Run 46 Run 47 Run 48 Run 49

Run 50 Run 51 Run 52 Run 53 Run 54 Run 55 Run 56 Run 57 Run 58 Run 59 Run 60 Run 61 Run 62 Run 63 Run 64 Run 65 Run 66 Run 67 Run 68 Run 69 Run 70 Run 71 Run 72 Run 73 Run 74 Run 75 Run 76 Run 77 Run 78 Run 79 Run 80 Run 81 Run 82 Run 83 Run 84 Run 85 Run 86 Run 87 Run 88 Run 89 Run 90 Run 91 Run 92 Run 93 Run 94 Run 95 Run 96 Run 97 Run 98

Run 99 Run 100 Run 101 Run 102 Run 103 Run 104 Run 105 Run 106 Run 107 Run 108 Run 109 Run 110 Run 111 Run 112 Run 113 Run 114 Run 115 Run 116 Run 117 Run 118 Run 119 Run 120 Run 121 Run 122 Run 123 Run 124 Run 125 Run 126 Run 127 Run 128 Run 129 Run 130 Run 131 Run 132 Run 133 Run 134 Run 135 Run 136 Run 137 Run 138 Run 139 Run 140 Run 141 Run 142 Run 143 Run 144 Run 145 Run 146 Run 147

Run 148 Run 149 Run 150 Run 151 Run 152 Run 153 Run 154 Run 155 Run 156 Run 157 Run 158 Run 159 Run 160 Run 161 Run 162 Run 163 Run 164 Run 165 Run 166 Run 167 Run 168 Run 169 Run 170 Run 171 Run 172 Run 173 Run 174 Run 175 Run 176 Run 177 Run 178 Run 179 Run 180 Run 181 Run 182 Run 183 Run 184 Run 185 Run 186 Run 187 Run 188 Run 189 Run 190 Run 191 Run 192 Run 193 Run 194 Run 195

Run 196 Run 197 Run 198 Run 199 Run 200 Run 201 Run 202 Run 203 Run 204 Run 205 Run 206 Run 207 Run 208 Run 209 Run 210 Run 211 Run 212 Run 213 Run 214 Run 215 Run 216 Run 217 Run 218 Run 219 Run 220 Run 221 Run 222 Run 223 Run 224 Run 225 Run 226 Run 227 Run 228 Run 229 Run 230 Run 231 Run 232 Run 233 Run 234 Run 235 Run 236 Run 237 Run 238 Run 239 Run 240 Run 241 Run 242 Run 243

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js