

In [14]:

```
# Import SciPy
import scipy
import pandas as pd
import numpy as np
from scipy import constants
from scipy.interpolate import UnivariateSpline
from numpy import exp
from scipy import integrate
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
from scipy.integrate import odeint
```

In [89]:

```
#1.1
# 导入γ的数据
df1=pd.read_csv('global.1751_2014.csv')
df1
```

Out[89]:

	year	total	gas	liquid	solid	cement production	gas flaring	per capita carbon emissions
0	1751	3	0	0	3	0	0	NaN
1	1752	3	0	0	3	0	0	NaN
2	1753	3	0	0	3	0	0	NaN
3	1754	3	0	0	3	0	0	NaN
4	1755	3	0	0	3	0	0	NaN
...
259	2010	9128	1696	3107	3812	446	67	1.32
260	2011	9503	1756	3134	4055	494	64	1.36
261	2012	9673	1783	3200	4106	519	65	1.36
262	2013	9773	1806	3220	4126	554	68	1.36

In [90]:

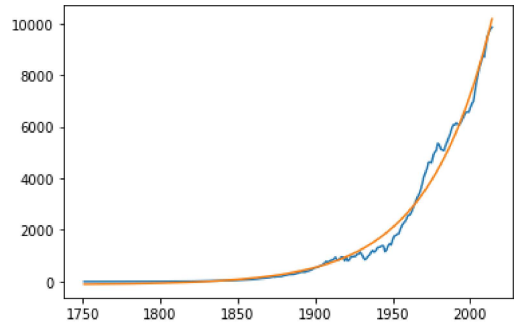
```
#拟合部分参考焦小乔同学
import scipy.optimize as optimize
# 注意到排放是随时间变化，对其进行指数拟合y=exp(a1*x+a2)+a3，得到化石燃料排放的二氧化碳随时间变化的曲线
def fossil_emiss(x,a1,a2,a3): # 定义拟合的函数
    return np.exp(a1*x+a2)+a3
# 设置拟合的初始值
a1=0.1
a2=0.1
a3=0
p0=[a1,a2,a3]

# 调用拟合函数
para,cov=optimize.curve_fit(fossil_emiss,df1['year'],df1['total'],p0=p0)

# 计算拟合后的结果
y_fit=[fossil_emiss(a,*para) for a in df['year']]

# 画真实值和拟合后的图
plt.plot(df1['year'],df1['total'])
plt.plot(df1['year'],y_fit)

plt.show()
print(para)
```



[2.39596800e-02 -3.90147674e+01 -1.22229282e+02]

In [91]:

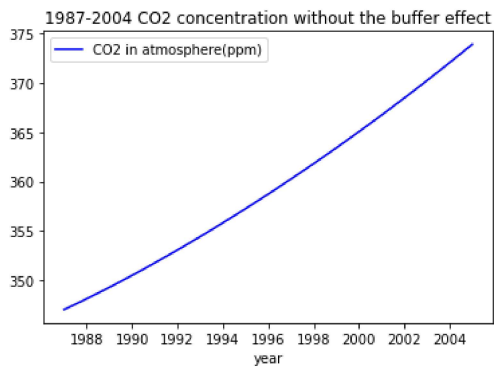
```
#https://zhuanlan.zhihu.com/p/484167038
# 定义没有buffer效应的碳循环函数
def model1(y, t, k12, k21):
    N1, N2, a = y#a是γ
    dydt = [-k12*N1+k21*N2+a, k12*N1-k21*N2, 2.39596800e-02*a+2.39596800e-02*1.2229282e+02]
    return dydt
```

In [92]:

```
# 设置初始值
t1=np.linspace(1987, 2005)
a=fossil_emiss(t,*para)
k12=105/740
k21=102/900
N1=740*1000
N2=900*1000
y0=[N1,N2,a[0]]
#转化单位
sol = odeint(model1, y0, t1, args=(k12,k21))/1000/740*347
```

In [93]:

```
# 画图
from matplotlib.ticker import MaxNLocator
plt.plot(t, sol[:, 0], 'b', label='CO2 in atmosphere(ppm)')
plt.legend(loc='best')
plt.xlabel(' year')
# 设置横坐标轴刻度为整数
plt.gca().xaxis.set_major_locator(MaxNLocator(integer=True))
plt.title('1987-2004 CO2 concentration without the buffer effect')
plt.show()
```



In [94]:

```
# 定义有buffer效应的函数
def model2(y, t, k12, k21, N0):
    N1, N2, a2 = y
    bf=3.69+1.86e-2*(N1/740/1000*347)-1.8e-6*((N1/740/1000*347)**2) #buffer factor
    dydt = [-k12*N1+k21*(N0+bf*(N2-N0))+a2,
            k12*N1-k21*(N0+bf*(N2-N0)),
            2.39596800e-02*a2+2.39596800e-02*1.2229282e+02]
    return dydt
```

In [95]:

```
# 设置初始值
t2=np.linspace(0, 253, 253)+1751
a2=fossil_emiss(t2,*para)

k12=105/740
k21=102/900
N0=821*1000
N1=618*1000
N2=821*1000
y0=[N1,N2,a2[0]]
sol2 = odeint(model2, y0, t2, args=(k12,k21,N0))/740/1000*347 #计算微分方程，并且转化为ppm的单位
```

In [96]:

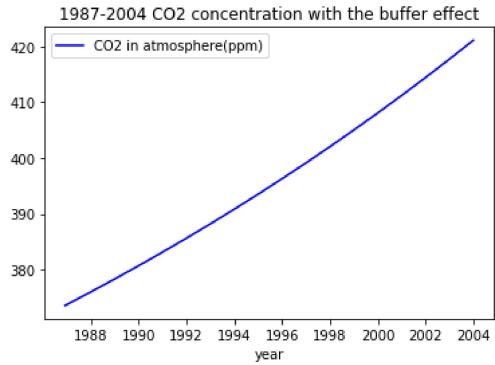
```
sol2
```

Out[96]:

```
array([[ 2.89791892e+02,  3.84982432e+02, -4.84577482e-02],
       [ 2.91238045e+02,  3.83487737e+02, -4.82420907e-02],
       [ 2.91660746e+02,  3.83016713e+02, -4.80211828e-02],
       [ 2.91763603e+02,  3.82865757e+02, -4.77948966e-02],
       [ 2.91766902e+02,  3.82814589e+02, -4.75631011e-02],
       [ 2.91739337e+02,  3.82794521e+02, -4.73256623e-02],
       [ 2.91702311e+02,  3.82784156e+02, -4.70824426e-02],
       [ 2.91662489e+02,  3.82776834e+02, -4.68333014e-02],
       [ 2.91621951e+02,  3.82770480e+02, -4.65780946e-02],
       [ 2.91581350e+02,  3.82764449e+02, -4.63166743e-02],
       [ 2.91540891e+02,  3.82758540e+02, -4.60488894e-02],
       [ 2.91500644e+02,  3.82752693e+02, -4.57745849e-02],
       [ 2.91460633e+02,  3.82746889e+02, -4.54936021e-02],
       [ 2.91420870e+02,  3.82741121e+02, -4.52057783e-02],
       [ 2.91381363e+02,  3.82735390e+02, -4.49109471e-02],
       [ 2.91342120e+02,  3.82729695e+02, -4.46089378e-02],
       [ 2.91303146e+02,  3.82724038e+02, -4.42995755e-02],
       [ 2.91264449e+02,  3.82718418e+02, -4.39826815e-02].
```

In [97]:

```
# 画图
plt.plot(t2[235:253], sol2[235:253,0], 'b', label='CO2 in atmosphere(ppm)')
plt.legend(loc='best')
plt.xlabel('year')
plt.gca().xaxis.set_major_locator(MaxNLocator(integer=True))
plt.title('1987-2004 CO2 concentration with the buffer effect')
plt.show()
```



In [98]:

```
# 导入观测值数据
df2=pd.read_csv('co2_annmean_g1.csv')
df2
```

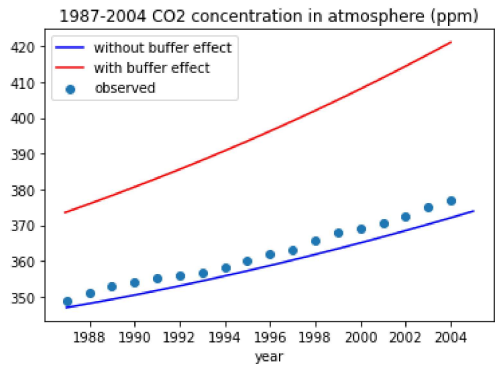
Out[98]:

	year	mean	unc
0	1979	336.85	0.10
1	1980	338.91	0.07
2	1981	340.11	0.08
3	1982	340.86	0.03
4	1983	342.53	0.05
5	1984	344.07	0.07
6	1985	345.54	0.07
7	1986	346.97	0.07
8	1987	348.68	0.09
9	1988	351.16	0.07
10	1989	352.79	0.06
11	1990	354.05	0.07
12	1991	355.39	0.06
13	1992	356.09	0.06
14	1993	356.83	0.07
15	1994	358.33	0.07
16	1995	360.17	0.05
17	1996	361.93	0.04
18	1997	363.05	0.05
19	1998	365.70	0.04
20	1999	367.80	0.07
21	2000	368.96	0.06
22	2001	370.57	0.05
23	2002	372.58	0.04
24	2003	375.14	0.04
25	2004	376.95	0.06
26	2005	378.98	0.05
27	2006	381.15	0.05
28	2007	382.90	0.05
29	2008	385.02	0.05
30	2009	386.50	0.04
31	2010	388.76	0.06
32	2011	390.63	0.06
33	2012	392.65	0.07
34	2013	395.40	0.06
35	2014	397.34	0.05
36	2015	399.65	0.05
37	2016	403.06	0.07
38	2017	405.22	0.07
39	2018	407.61	0.07
40	2019	410.08	0.08
41	2020	412.44	0.06
42	2021	414.71	0.06

In [99]:

```
# 画图
plt.plot(t, sol[:, 0], 'b') # without buffer effect
plt.plot(t2[235:253], sol2[235:253,0], 'r') # with buffer effect
plt.scatter(df2['year'][8:26], df2['mean'][8:26]) # observed data

plt.legend(['without buffer effect', 'with buffer effect', 'observed'], loc='best')
plt.xlabel(' year')
plt.gca().xaxis.set_major_locator(MaxNLocator(integer=True)) # 设置横坐标轴刻度为整数
plt.title('1987-2004 CO2 concentration in atmosphere (ppm)')
plt.show()
# 不考虑buffer效应时比观测值低估，可能是由于人为源排放拟合时有一定的低估，因此积分之后会出现比观测值低
```



In []:

In []:

In []: