1.1先用y=exp(a1*x+a2)+a3作为拟合函数，得到化石燃料排放的二氧化碳随时间变化的曲线，画出真实值和拟合后的图，得到拟合函数里的三个参数，然后在不考虑buffer effect的情况下设置碳循环函数，画出没有buffer effect的二氧化碳大气浓度。
1.2重新定义有buffer effect的函数，画图。
1.3利用二氧化碳年平均数据，再把上面两个图结合起来，得到类似文献中的图2的二氧化碳模拟与观测值的变化趋势图。
Bonus 先选取需要使用的数据，计算各种k值和初始值，然后积分计算box每部分的变化，最后画图。

In [1]:
```
# 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
from scipy import optimize
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
import xlrd
%matplotlib inline
from scipy.integrate import odeint
```

1.1build a two-box model to compute the atmospheric CO2 level in ppm (parts per million) from 1987 to 2004(without the buffer effect)

In [2]:
```
df1=pd.read_csv('global.1751_2014.csv')
df1
```

Out[2]:

	Year	Total carbon emissions from fossil fuel consumption and cement production (million metric tons of C)	Carbon emissions from gas fuel consumption	Carbon emissions from liquid fuel consumption	Carbon emissions from solid fuel consumption	Carbon emissions from cement production	Carbon emissions from gas flaring	Pe em af
0	1751	3	0	0	3	0	0	
1	1752	3	0	0	3	0	0	
2	1753	3	0	0	3	0	0	
3	1754	3	0	0	3	0	0	
4	1755	3	0	0	3	0	0	
...	...	...	...	...	...	...	...	
259	2010	9128	1696	3107	3812	446	67	
260	2011	9503	1756	3134	4055	494	64	
261	2012	9673	1783	3200	4106	519	65	
262	2013	9773	1806	3220	4126	554	68	
263	2014	9855	1823	3280	4117	568	68	

264 rows × 8 columns

In [3]:

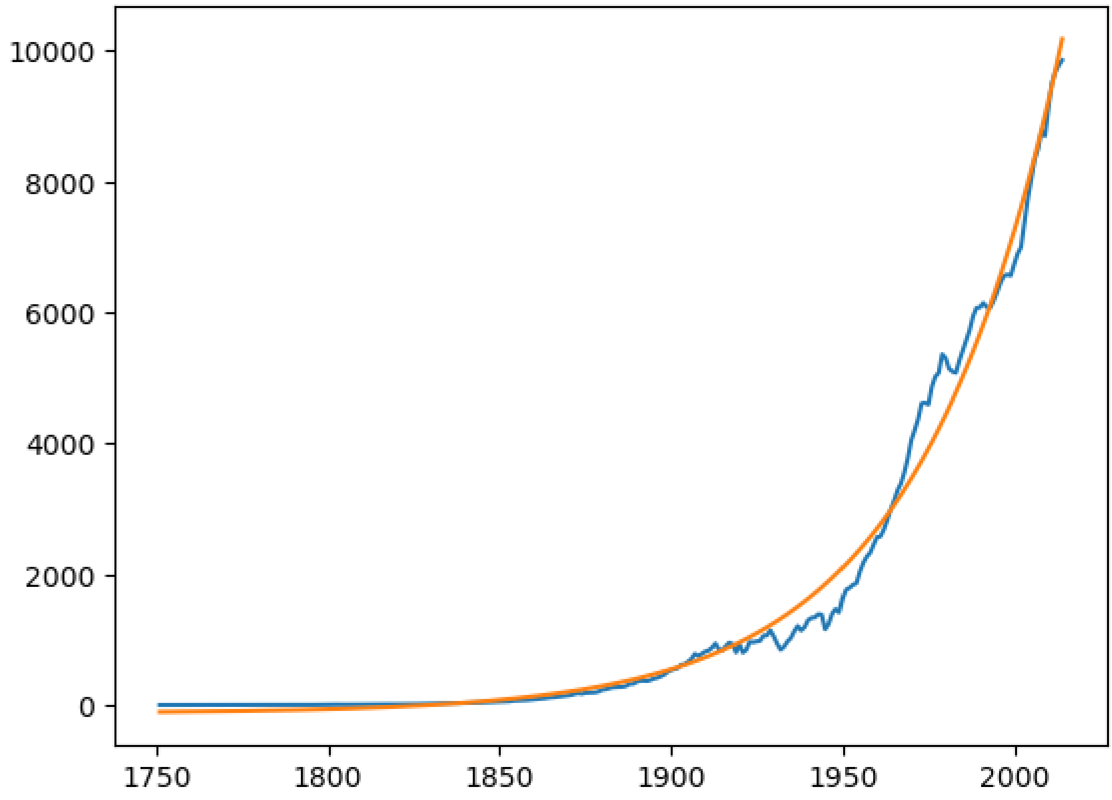
```
# got help from my senior fellow apprentice Zhao Wangchao
# 用y=exp(a1*x+a2)+a3作为拟合函数，得到化石燃料排放的二氧化碳随时间变化的曲线
def fossil_emiss(x,a1,a2,a3):
    return np.exp(a1*x+a2)+a3

# 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
df1['Year'].astype(int)
df1['Total carbon emissions from fossil fuel consumption and cement production (milli
para,cov = optimize.curve_fit(fossil_emiss, df1['Year'], df1['Total carbon emissions

# 画真实值和拟合后的图
plt.plot(df1['Year'],df1['Total carbon emissions from fossil fuel consumption and cem
plt.plot(df1['Year'], fossil_emiss(df1['Year'], *para), '-', label='fit' )

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



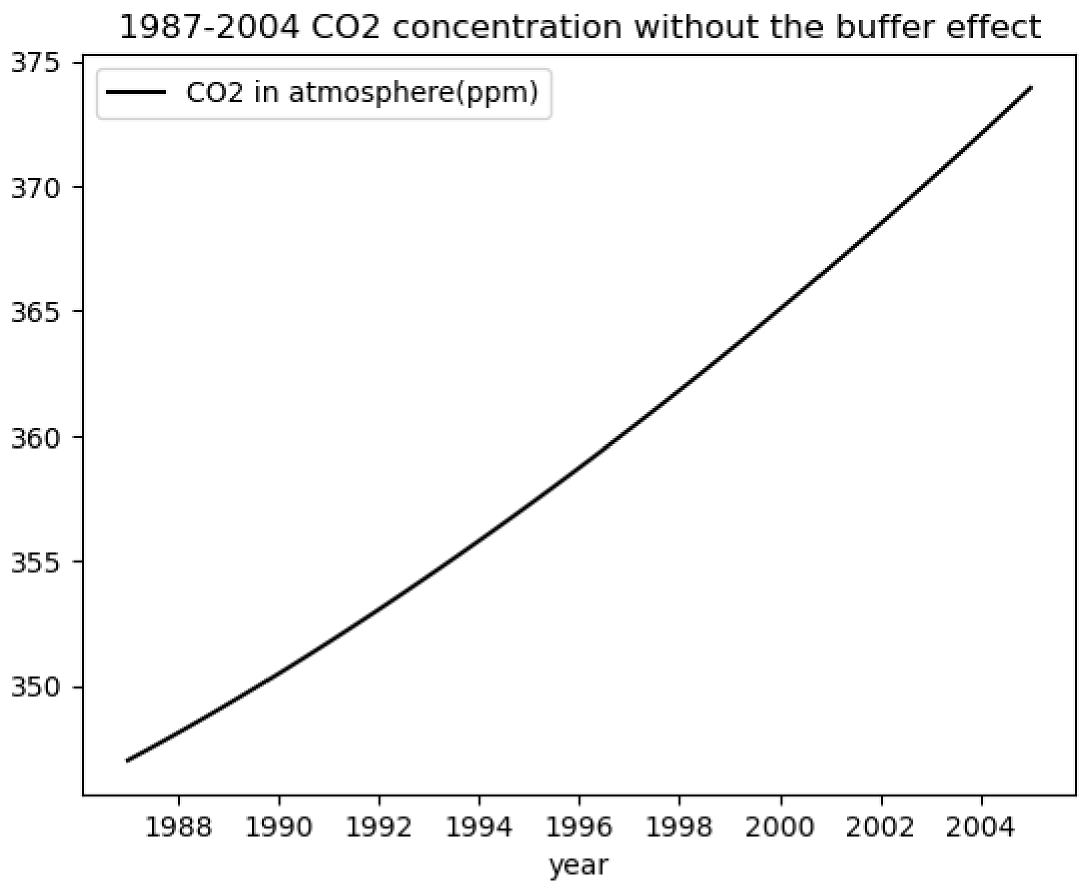

```
[ 2.39596800e-02 -3.90147674e+01 -1.22229282e+02]
```

In [4]:

```
#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*1.22229282e
    return dydt
```

In [5]:
```
# 设置初始值
t1=np.linspace(1987, 2005)
a=fossil_emiss(t1,*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 [6]:
```
# 画图
from matplotlib.ticker import MaxNLocator
plt.plot(t1, sol[:, 0], 'k', 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()
```



1.2 build a two-box model to compute the atmospheric CO2 level in ppm from 1987 to 2004(with the buffer effect)

In [7]:
```python
# 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*1.22229282e+02]
    return dydt
```

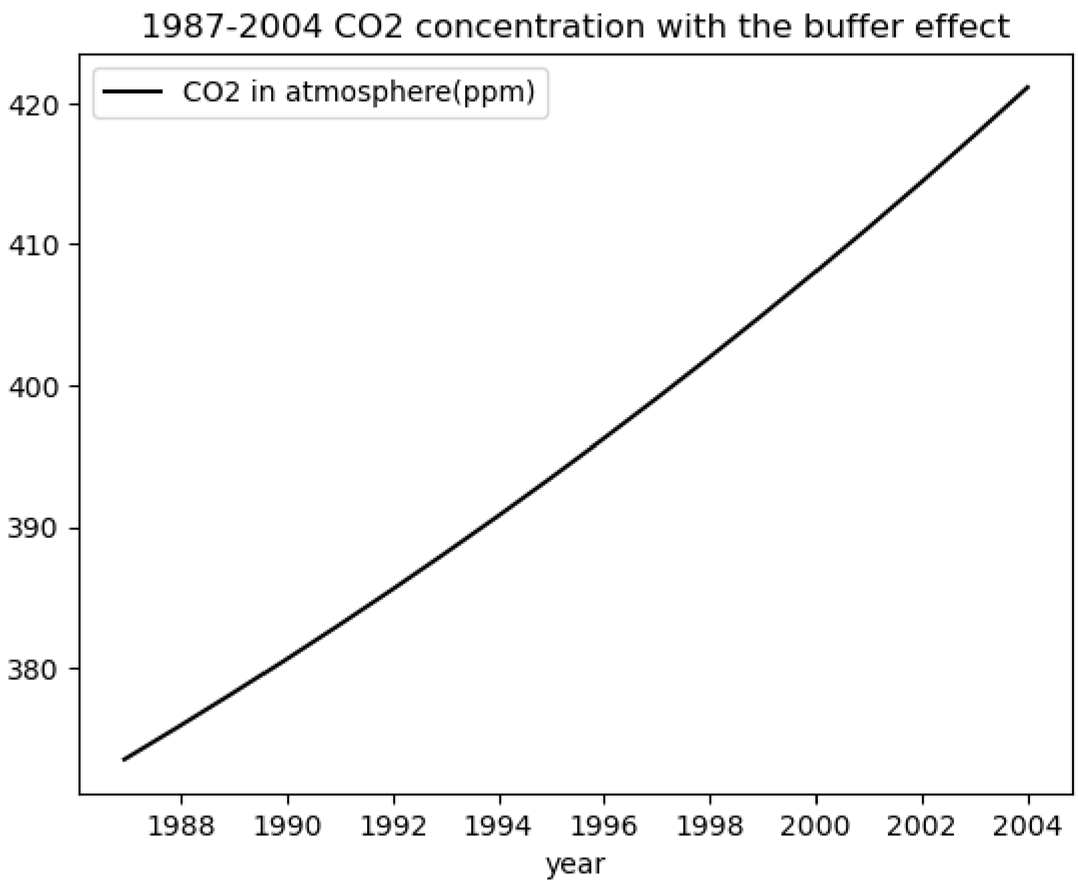
In [8]:
```python
# 设置初始值
t2=np.linspace(0, 253, 253)+1751
a2=fossil_emiss(t2,*para)

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
```

In [9]:
```python
sol2
```

Out[9]:
```
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],
       [ 2.91226034e+02,  3.82712838e+02, -4.36580724e-02],
```

In [10]:
```
# 画图
plt.plot(t2[235:253], sol2[235:253,0], 'k', 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()
```



1.3 reproduce Figure 2 in Tomizuka (2009)

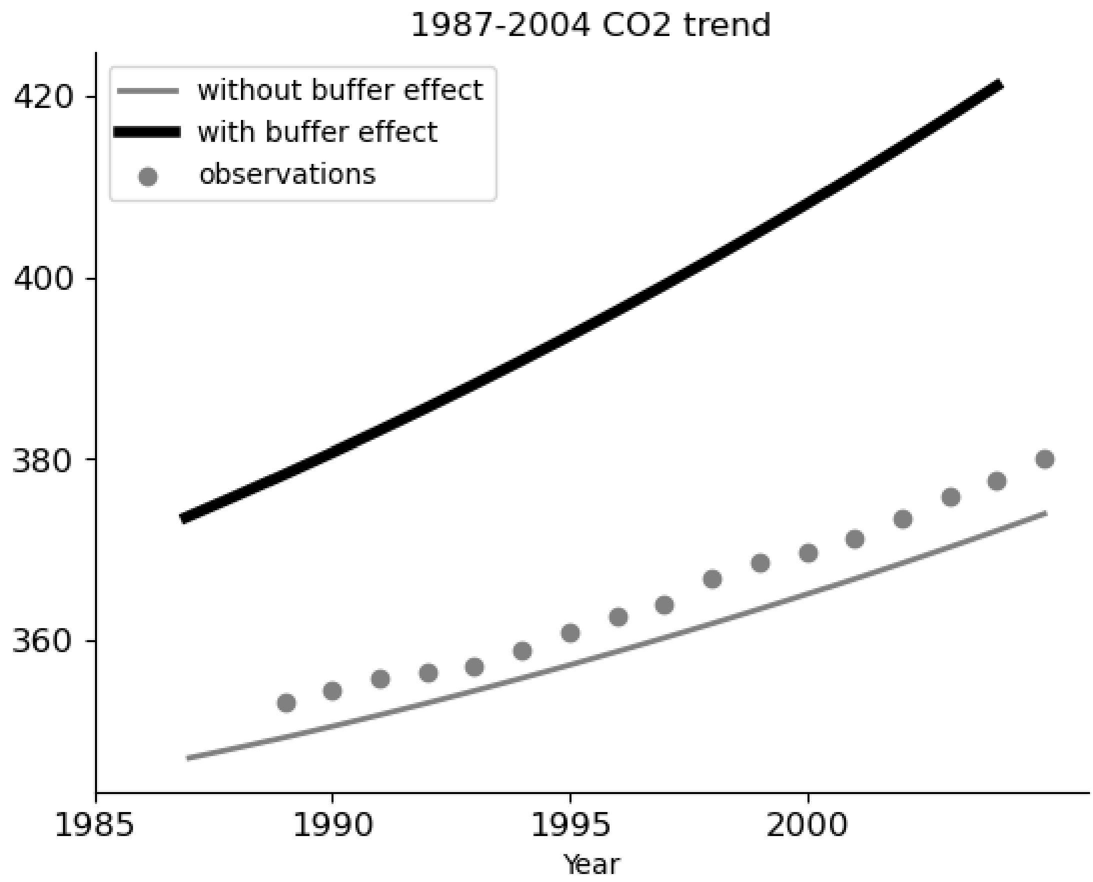
In [11]:
```
# 导入观测值数据
df2=pd.read_csv('co2_annmean_mlo.csv')
df2
```

Out[11]:

	year	mean	unc
0	1959	315.98	0.12
1	1960	316.91	0.12
2	1961	317.64	0.12
3	1962	318.45	0.12
4	1963	318.99	0.12
...	...	...	...
59	2018	408.72	0.12
60	2019	411.65	0.12
61	2020	414.21	0.12
62	2021	416.41	0.12
63	2022	418.53	0.12

64 rows × 3 columns

In [12]:
```
# 画图
plt.plot(t1, sol[:, 0], linewidth=2, color='grey') # without buffer effect
plt.plot(t2[235:253], sol2[235:253,0], linewidth=4, color='black') # with buffer effe
plt.scatter(df2['year'][30:47],df2['mean'][30:47], color='gray') # observed data
# 设置图例
plt.legend(['without buffer effect','with buffer effect','observations'],loc='best')
plt.xlabel('Year')
# 设置横纵坐标
plt.gca().xaxis.set_major_locator(MaxNLocator(integer=True))
plt.xticks([1985, 1990, 1995, 2000], fontsize=12)
plt.yticks([360, 380, 400, 420], fontsize=12)
# 隐藏右边和上边的边框
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
plt.title('1987-2004 CO2 trend')
plt.show()
# 不考虑buffer effect的曲线与文献中图2略有区别，可能是函数拟合不够好
```



[Bonus] Following equation 5-13, compute the atmospheric CO2 level in ppm and reproduce Figure 4 in Tomizuka (2009).

In [13]:

```
# got help from my classmate Xu Jiwen
co2_observations = pd.read_csv('1750-2000CO2.csv')
land_use_data = pd.read_excel('Global_land-use_flux-1750_2005.xls')
ff_emissions = pd.read_csv('global.1751_2014.csv')


land_use_data = land_use_data[['Year', 'Global']]
land_use_data['LandUseChange'] = land_use_data['Global'] / (1000 * 2.13)

ff_emissions = ff_emissions[['Year', 'Total carbon emissions from fossil fuel consump
ff_emissions['FossilFuelEmissions'] = ff_emissions.iloc[:, 1] - ff_emissions.iloc[:,
ff_emissions['EmissionFactor'] = ff_emissions['FossilFuelEmissions'] / (1000 * 2.13)
```

```
In [14]: k12, k21, k23, k24, k32, k34, k43, k45, k51, k67, k71 = [60 / 615, 60 / 842, 9 / 842,
N2_0 = 842 / 2.13


initial_conditions = [615 / 2.13, 842 / 2.13, 9744 / 2.13, 26280 / 2.13, 90000000 / 2
f0 = 62 / 2.13
P0 = 615 / 2.13

# Explore the Beta value
beta_values = [0.38, 0.5]
results = []

for beta in beta_values:

    N1, N2, N3, N4, N5, N6, N7 = initial_conditions.copy()
    atmosphere = [N1]

    for year in range(1751, 2001):
        gamma = ff_emissions[ff_emissions['Year'] == year]['EmissionFactor'].values[0
        delta = land_use_data[land_use_data['Year'] == year]['LandUseChange'].values[

        xi = 3.69 + 0.0186 * N1 - 0.0000018 * N1**2

        f = f0 * (1 + beta * np.log(N1 / P0))

        # Calculate the rate of change for each part
        dN1_dt = -k12 * N1 + k21 * (N2_0 + xi * (N2 - N2_0)) + gamma - f + delta + k
        dN2_dt = k12 * N1 - k21 * (N2_0 + xi * (N2 - N2_0)) - k23 * N2 + k32 * N3 -
        dN3_dt = k23 * N2 - k32 * N3 - k34 * N3 + k43 * N4
        dN4_dt = k34 * N3 - k43 * N4 + k24 * N2 - k45 * N4
        dN5_dt = k45 * N4 - k51 * N5
        dN6_dt = f - k67 * N6 - 2 * delta
        dN7_dt = k67 * N6 - k71 * N7 + delta

        # Update the values of each section
        N1 += dN1_dt
        N2 += dN2_dt
        N3 += dN3_dt
        N4 += dN4_dt
        N5 += dN5_dt
        N6 += dN6_dt
        N7 += dN7_dt

        atmosphere.append(N1)

    results.append(atmosphere)
```

In [15]:
```python
plt.figure(figsize=(12, 8))

plt.scatter(co2_observations['year'], co2_observations['mean'], label='Observations',

plt.text(1850, 300, 'Calculations', fontsize=12)
plt.text(1900, 290, 'Observations', fontsize=12)
plt.text(1950, 345, 'β=0.38', fontsize=12, color='red')
plt.text(1980, 320, 'β=0.50', fontsize=12, color='blue')

# Draw the results of each beta value directly, with the colors red and blue
plt.plot(range(1750, 2001), results[0], color='red', label='β=0.38')
plt.plot(range(1750, 2001), results[1], color='blue', label='β=0.50')


plt.xlabel('Year', fontsize=14)
plt.ylabel('CO2 Concentration (ppm)', fontsize=14)

plt.legend()

plt.show()
```