In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import scipy
from scipy.integrate import odeint
from scipy.optimize import curve_fit
import math
```

In [2]:

```
# 导入数据
CO2 = pd.read_csv("D:\data\\global.1751_2008.csv")
CO2
```

Out[2]:

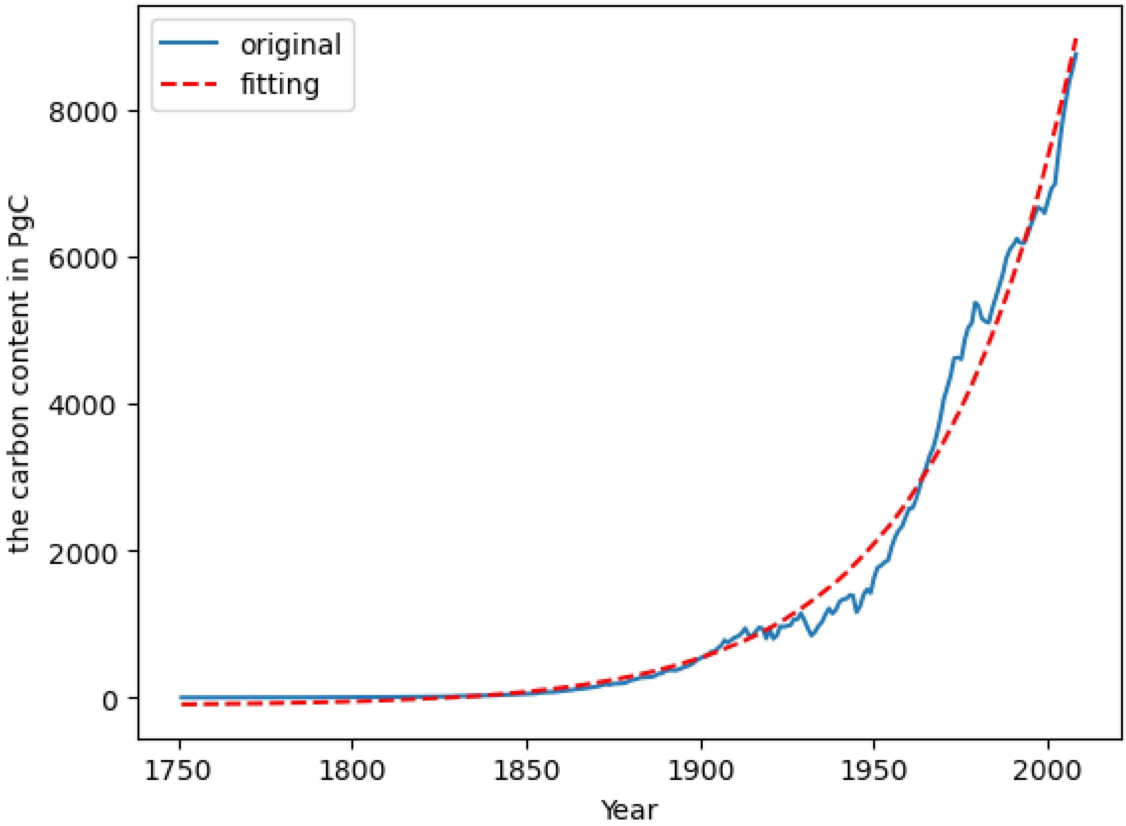
	Year	Total	Gas	Liquids	Solids	Cement Production	Gas Flaring	Per Capita
0	1751	3	0	0	3	0	0	NaN
1	1752	3	0	0	3	0	0	NaN
2	1753	3	0	0	3	0	0	NaN
3	1754	3	0	0	3	0	0	NaN
4	1755	3	0	0	3	0	0	NaN
...	...	...	...	...	...	...	...	...
253	2004	7782	1431	3027	2971	298	55	1.21
254	2005	8086	1473	3071	3162	320	61	1.24
255	2006	8350	1519	3080	3333	355	62	1.27
256	2007	8543	1551	3074	3468	382	68	1.28
257	2008	8749	1616	3095	3578	386	73	1.30

258 rows × 8 columns

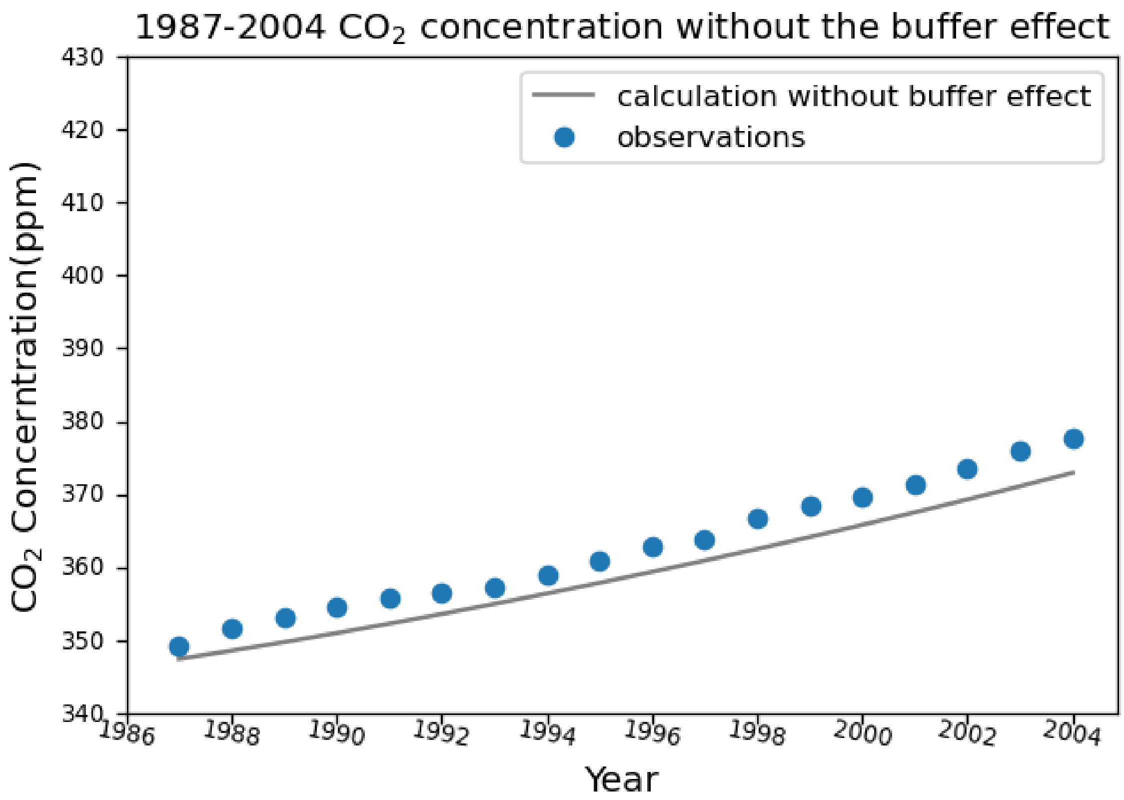
In [3]:

```
# 1.1
# 定义一个指数函数
def r_func(x, a, b, c):
    return a * np.exp(b * x) + c

# 没加maxfev时，报错提示：Optimal parameters not found: Number of calls to function h
# 为了解决问题，参考：https://blog.csdn.net/Jacktuo/article/details/120682582
# 发现改变最大迭代次数后，还是报错，为了减少计算量，用p0来设置参数的初始值，参考：htt
popt,pcov = curve_fit(r_func, CO2['Year'], CO2['Total'],maxfev=1500, p0=[1,0,1])
# 输出拟合系数a,b,c
print(popt)
y_fit= [r_func(i, popt[0], popt[1], popt[2]) for i in CO2['Year']]

#输出原曲线和拟合曲线
plt.plot(CO2['Year'], CO2['Total'],label='original')
plt.plot(CO2['Year'], y_fit,'r--',label='fitting')
plt.legend(loc=2)
plt.xlabel('Year');plt.ylabel('the carbon content in PgC')
plt.show()
# 注意这里单位为百万公吨（1 Tg C=0.001 Pg C）
```

```
[ 4.82316522e-18  2.43954472e-02 -1.12313831e+02]
```

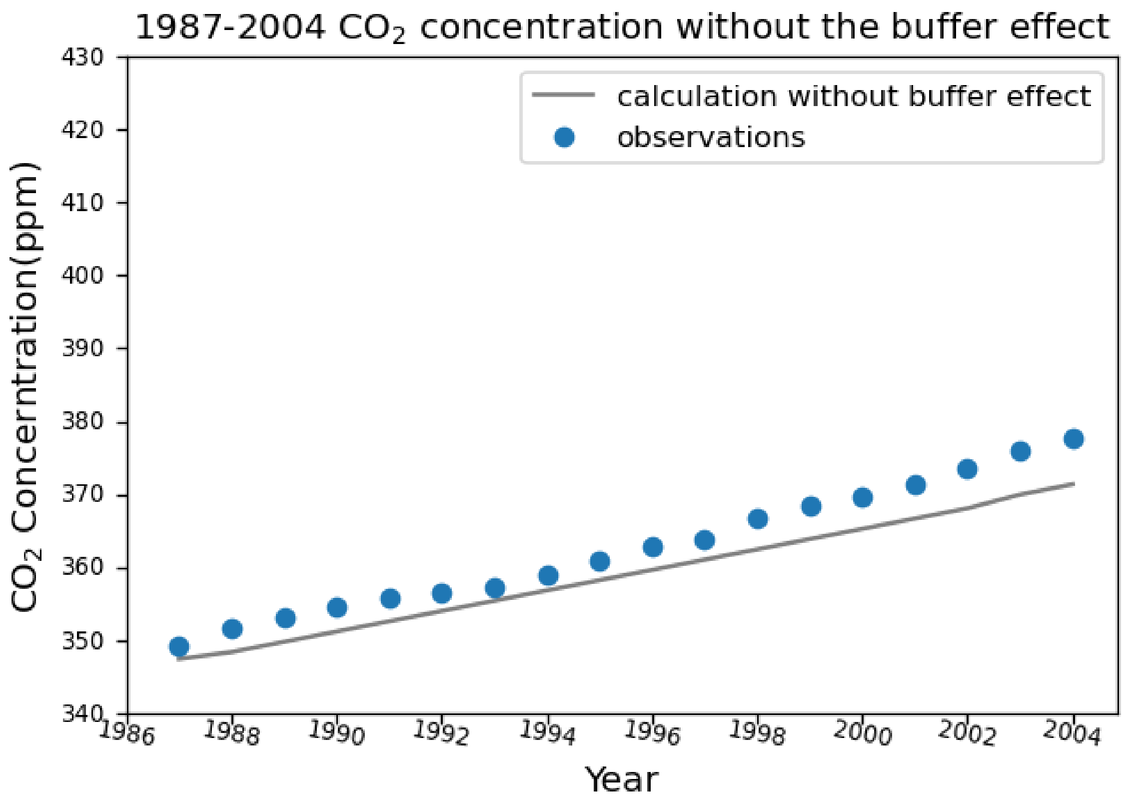


In [4]:
```
#排放速率r=a*e^(bx)+c, r'=a*b*e^(bx)
a=popt[0]
b=popt[1]
c=popt[2]
# 从拟合的函数选取1987-2004时间段
x1=np.arange(1987,2005)
# r1为拟合每年产生的多少百万公吨碳
r1=r_func(x1, a,b,c)
# 初始条件，N1和N2单位均为Tg C
k12=105/740
k21=102/900
N1=740*1000
N2=900*1000
y0=[N1,N2,r1[0]]

# 函数定义了一个包含三个微分方程的微分方程组,得到N1, N2 和 r 随时间的变化率: dN1/dt,
def pend(y,x1,k12,k21):
    N1, N2, r1 = y
    return [-k12*N1+k21*N2+r1,k12*N1-k21*N2, b*(r1-c)]

# 对以上微分方程积分，得到的函数sol的三列分别为N1, N2 和 r
sol=odeint(pend, y0,x1, args=(k12,k21))
# 文章中说1986年的740 PgC对应的浓度时347 ppm(文中说将PgC单位数据除以2.13就得到 ppm),
# 因此按照这个系数将得到的大气浓度转化为 ppm单位
sol_N1 = sol[:,0]/2.13/1000

# 作图，画拟合曲线
plt.figure(figsize=(6,4),dpi=120)
plt.plot(x1, sol_N1, color='gray', label='calculation without buffer effect')
# 画实际观测图，仅选取1987-2004时间段的数据
observation = pd.read_csv("D:\data\\co2_annmean_mlo.csv")
observation_1=observation.iloc[28:46,]
plt.plot(observation_1['year'],observation_1['mean'],"o",label='observations')

plt.yticks(ticks=np.arange(340,440,10), fontsize=8, rotation=0, ha='right', va='cent
plt.xticks(ticks=np.arange(1986,2005,2), fontsize=8, rotation=-10, ha='center', va='
plt.legend()
plt.xlabel('Year',fontsize=12)
plt.ylabel('CO$_2$ Concerntration(ppm)',fontsize=12)
plt.title('1987-2004 CO$_2$ concentration without the buffer effect',fontsize=12)
plt.show()
```


1987-2004 CO_2 concentration without the buffer effect
calculation without buffer effect
observations
CO_2 Concerntration(ppm)
Year
430
420
410
400
390
380
370
360
350
340
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004

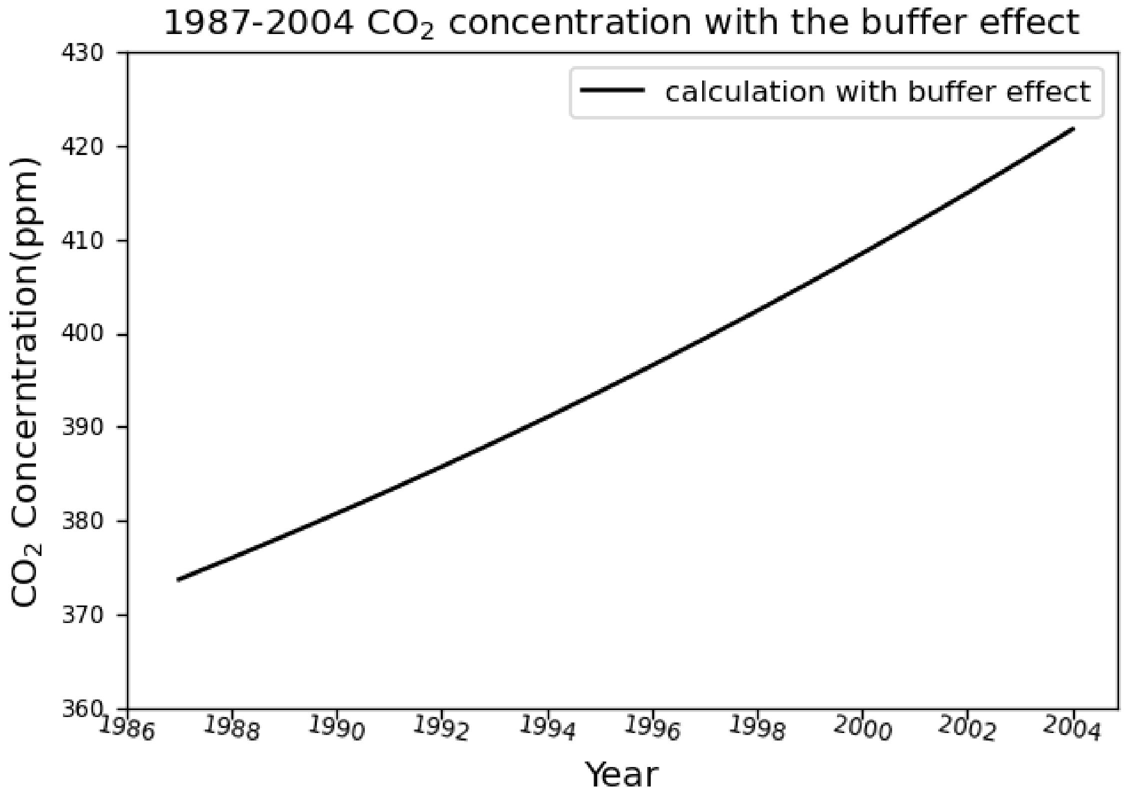
In [5]:

```
# 感觉计算值和观测值不像文中重合，因此采用另一种方式计算
# 选取到1987-2004的数据，并将单位PgC换算成百万公吨
r1 = np.loadtxt("D:\data\\global.1751_2008.csv", delimiter=",", skiprows = 237)[0:-4
# 定义参数
x1 = np.arange(1987,2005)
k12 = 105/740
k21 = 102/900
N1_pend_1 = np.empty_like(x1)
N2_pend_1 = np.empty_like(x1)

# 函数定义了一个包含两个微分方程的微分方程组,得到N1, N2 随时间的变化率
def pend_1(N,x1,r1):
    N1,N2 = N
    dN1dt = -k12 * N1 + k21 * N2 + r1
    dN2dt = k12 * N1 - k21 * N2
    return [dN1dt, dN2dt]

# 设置初始值
N0 = [740, 900]
N1_pend_1[0],N2_pend_1[0] = N0

# 对以上微分方程积分，2004-1987+1=18
for i in range(1,18):
    dx = [0,1]
    N = odeint(pend_1,N0,dx,args=(r1[i-1],))
    N1_pend_1[i] = N[1][0]
    N2_pend_1[i] = N[1][1]
    N0 = N[1]

# 单位换算成ppm
sol_N11 = N1_pend_1/2.13

# 作图，画拟合曲线
plt.figure(figsize=(6,4),dpi=120)
plt.plot(x1, sol_N11, color='gray', label='calculation without buffer effect')

# 画实际观测图，仅选取1987-2004时间段的数据
plt.plot(observation_1['year'],observation_1['mean'],"o",label='observations')

plt.yticks(ticks=np.arange(340,440,10), fontsize=8, rotation=0, ha='right', va='cent
plt.xticks(ticks=np.arange(1986,2005,2), fontsize=8, rotation=-10, ha='center', va='
plt.legend()
plt.xlabel('Year',fontsize=12)
plt.ylabel('CO$_2$ Concerntration(ppm)',fontsize=12)
plt.title('1987-2004 CO$_2$ concentration without the buffer effect',fontsize=12)
plt.show()

# 观测值和计算值接近了一些（1987和1993年重合了），但还是无法完全符合文章中的图
# 但以上两种方法思路相同，只是计算方法不同也会有误差，因此与文中图片的差异可能是数据
```



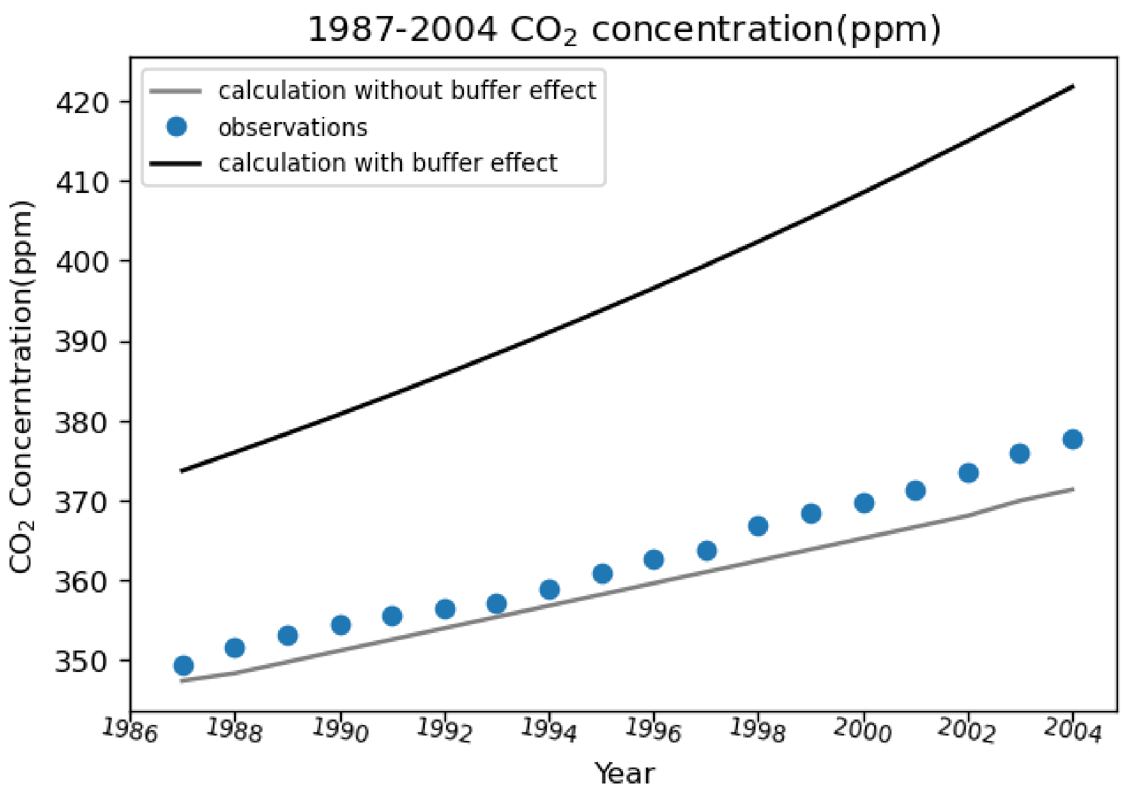
1987-2004 CO_2 concentration without the buffer effect
calculation without buffer effect
observations
CO_2 Concerntration(ppm)
Year
430
420
410
400
390
380
370
360
350
340
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004

In [6]:
```
# 1.2

#从前工业时代开始计算（1750年）
x2=np.arange(1750,2005)
r2=r_func(x2, a,b,c)

# 初始条件
k12=105/740
k21=102/900
N1=615*1000
N2=842*1000
y0=[N1,N2,r2[0]]

# 函数定义了一个包含三个微分方程的微分方程组,得到N1, N2 和 r 随时间的变化率: dN1/dt,
def pend_2(y,x2,k12,k21):
    N1,N2,r2 = y
    z = N1/1000/2.13                          # 将大气中二氧化碳浓度转化为 p
    N2_0 = 842*1000                           # 工业化前海洋二氧化碳的初始值
    ξ = 3.69+1.86e-2*(z)-1.8e-6*(z**2)        # 缓冲因子
    return [-k12*N1+k21*(N2_0+ξ*(N2-N2_0))+r2, k12*N1-k21*(N2_0+ξ*(N2-N2_0)),b*(

# 积分并将单位换算成 ppm
sol_1= odeint(pend_2, y0, x2, args=(k12,k21))
sol_N1_1 = sol_1/2.13/1000

# 作图
plt.figure(figsize=(6,4),dpi=120)
plt.plot(x2[237:255],sol_N1_1[237:255,0], color='k', label='calculation with buffer (

plt.yticks(ticks=np.arange(360,440,10), fontsize=8, rotation=0, ha='right', va='cent
plt.xticks(ticks=np.arange(1986,2006,2), fontsize=8, rotation=-10, ha='center', va='
plt.legend()
plt.xlabel('Year',fontsize=12)
plt.ylabel('CO$_2$ Concerntration(ppm)',fontsize=12)
plt.title('1987-2004 CO$_2$ concentration with the buffer effect',fontsize=12)
plt.show()
```



1987-2004 CO2 concentration with the buffer effect
calculation with buffer effect
CO2 Concerntration(ppm)
Year
430
420
410
400
390
380
370
360
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004

In [7]:
```
#1.3
plt.figure(figsize=(6,4),dpi=120)
# 对三种数据作图
plt.plot(x1, sol_N11, color='gray', label='calculation without buffer effect')
plt.plot(observation_1['year'],observation_1['mean'],"o",label='observations')
plt.plot(x2[237:255],sol_N1_1[237:255,0], color='k', label='calculation with buffer (

#调整作图参数
plt.xticks(ticks=np.arange(1986,2006,2), fontsize=8, rotation=-10, ha='center', va='
plt.legend(loc='best',fontsize=8)
plt.xlabel('Year');plt.ylabel('CO$_2$ Concerntration(ppm)')
plt.title('1987-2004 CO$_2$ concentration(ppm)',fontsize=12)
plt.show()
```



In [8]:

```
# Bonus
seven_box=pd.read_csv("D:\data\\Global_land-use_flux-1850_2005.csv")
# 新建dataframe.将全球土地变化引起的二氧化碳变化数据提取出来
# 并按照文中给出的1750年对应的0.2 PgC数据转化为百万公吨插入表格第一行
df = pd.DataFrame()
df['Year'] = seven_box['Year']
df['Global'] = seven_box['Global']
row=['1750','200']
df.iloc[0]=row
df = df.astype(int)
df
```

```
C:\Users\lenovo\AppData\Local\Temp\ipykernel_12368\370091035.py:9: FutureWarning: Setting an item of incompatible dtype is deprecated and will raise in a future error of pandas. Value '1750' has dtype incompatible with int64, please explicitly cast to a compatible dtype first.
  df.iloc[0]=row
C:\Users\lenovo\AppData\Local\Temp\ipykernel_12368\370091035.py:9: FutureWarning: Setting an item of incompatible dtype is deprecated and will raise in a future error of pandas. Value '200' has dtype incompatible with float64, please explicitly cast to a compatible dtype first.
  df.iloc[0]=row
```

Out[8]:

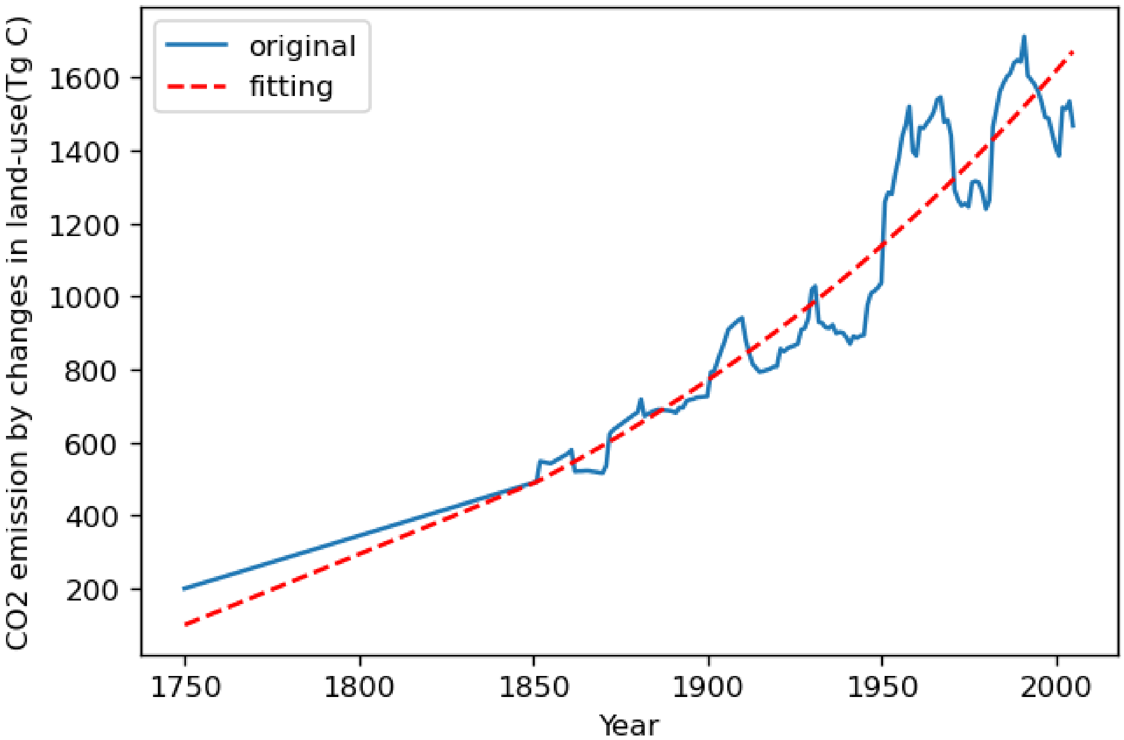
	Year	Global
0	1750	200
1	1851	492
2	1852	548
3	1853	546
4	1854	544
...	...	...
151	2001	1385
152	2002	1517
153	2003	1513
154	2004	1534
155	2005	1467

156 rows × 2 columns

In [9]:

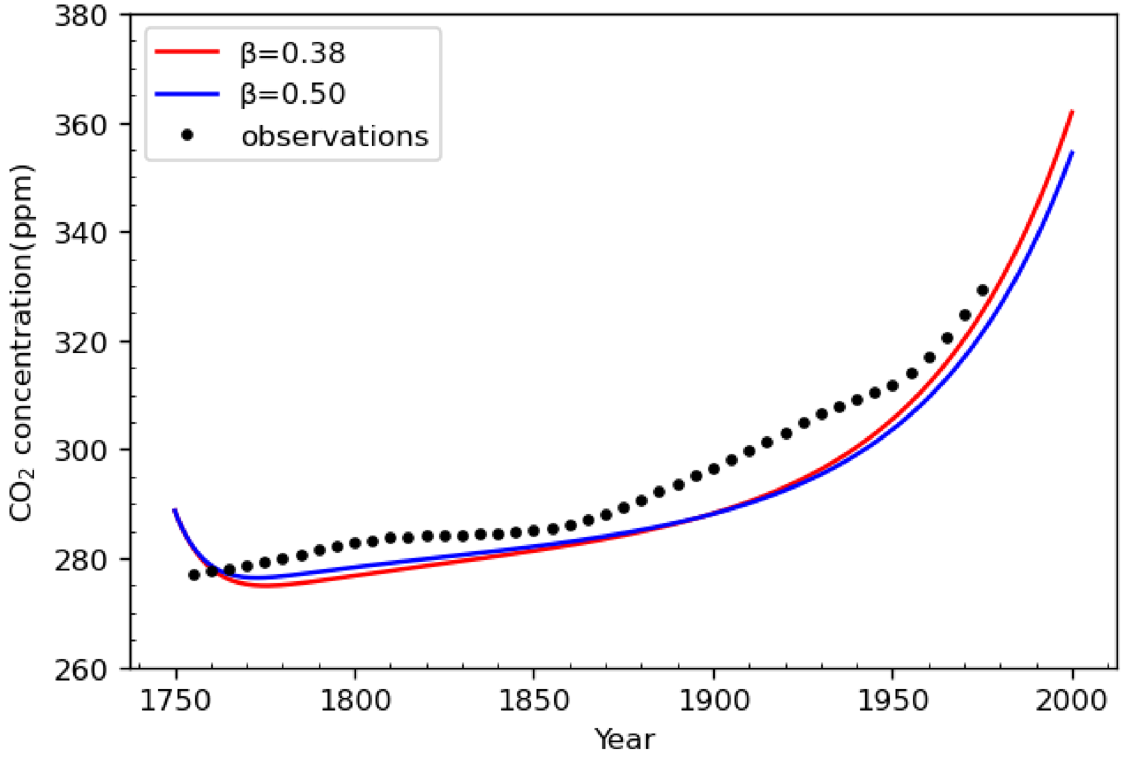
```
# 定义一个指数函数（土地利用变化对大气的排放率）
def  δ_func(x, a, b, c):
    return a * np.exp(b * x) + c

popt1,pcov = curve_fit(δ_func, df['Year'], df['Global'], p0=[1,0,1], maxfev=2000)
y_fit= [δ_func(i, popt1[0], popt1[1], popt1[2]) for i in df['Year']]

# 作图
#输出原曲线和拟合曲线
plt.figure(figsize=(6,4),dpi=120)
plt.plot(df['Year'], df['Global'],label='original')
plt.plot(df['Year'], y_fit,'r--',label='fitting')
plt.legend(loc=2)
plt.xlabel('Year')
plt.ylabel('CO2 emission by changes in land-use(Tg C)')

#输出拟合得到的a1,b1,c1值
print(popt)
```

```
[ 4.82316522e-18  2.43954472e-02 -1.12313831e+02]
```



In [10]:
```
a1 = popt1[0]
b1 = popt1[1]
c1 = popt1[2]

# 输入系数和各参数初始值
N02 = 821
k12 = 60/615
k21 = 60/842
k23 = 9/842
k24 = 43/842
k32 = 52/9744
k34 = 162/9744
k43 = 205/26280
k45 = 0.2/26280
k51 = 0.2/90000000
k67 = 62/731
k71 = 62/1328

N1=615*1000
N2=842*1000
N3=9744*1000
N4=26280*1000
N5=90000000*1000
N6=731*1000
N7=1238*1000

# 选取1750-2000年
x3=np.arange(1750,2001)
r3=r_func(x3,a,b,c)
 δ = δ _func(x3,a1,b1,c1)                        # δ 是土地利用变化对大气的排放率
y0=[N1,N2,N3,N4,N5,N6,N7,r3[0], δ [0]]

#β=0.38
def pend_3(y,x3):
    N1,N2,N3,N4,N5,N6,N7,r3, δ  = y
    z =N1/1000/2.13                              # 将大气中二氧化碳浓度转化为 ppm
    N2_0 = 842*1000
    p=N1
    p0=615*1000                                  # p为大气二氧化碳浓度，p0对应于工业化
    f0=62*1000                                   # f为净初级生产率，f0对应于工业化前的
     ξ  = 3.69+1.86e-2*(z)-1.8e-6*(z**2)         # 缓冲因子
    f=f0*(1+0.38*math.log(p/p0,math.e))
    dydt = [-k12*N1+k21*(N2_0+ ξ *(N2-N2_0))+r3-f+ δ +k51*N5+k71*N7,     # 以下是N1 到
            k12*N1-k21*(N2_0+ ξ *(N2-N2_0))-k23*N2+k32*N3-k24*N2,
            k23*N2-k32*N3-k34*N3+k43*N4,
            k34*N3-k43*N4+k24*N2-k45*N4,
            k45*N4-k51*N5,
            f-k67*N6-2* δ ,
            k67*N6-k71*N7+ δ ,
            b*(r3-c),
            b1*(r3-c1)]
    return dydt

#β=0.50
def pend_4(y,x3):
    N1,N2,N3,N4,N5,N6,N7,r3, δ  = y
    z =N1/1000/2.13
    N2_0 = 842*1000
    p=N1
    p0=615*1000
    f0=62*1000
```

```
    ξ  = 3.69+1.86e-2*(z)-1.8e-6*(z**2)
    f=f0*(1+0.50*math.log( p/p0,math.e))
    dydt = [-k12*N1+k21*(N2_0+ξ*(N2-N2_0))+r3-f+δ+k51*N5+k71*N7,
            k12*N1-k21*(N2_0+ξ*(N2-N2_0))-k23*N2+k32*N3-k24*N2,
            k23*N2-k32*N3-k34*N3+k43*N4,
            k34*N3-k43*N4+k24*N2-k45*N4,
            k45*N4-k51*N5,
            f-k67*N6-2*δ,
            k67*N6-k71*N7+δ,
            b*(r3-c),
            b1*(r3-c1)]
    return dydt

# 积分完将单位转化为ppm
sol3=odeint(pend_3, y0,x3)/2.13/1000
sol4=odeint(pend_4, y0,x3)/2.13/1000

# 作图
# 画不同β值对应的积分曲线
plt.figure(figsize=(6,4),dpi=120)
axes = plt.subplot()
plt.plot(x3, sol3[:,0], 'r', label='β=0.38')
plt.plot(x3, sol4[:,0], 'b', label='β=0.50')

# 根据观测值作图
observation_2=pd.read_csv("D:\data\\lawdome.smoothed yr75.csv")
observation_3 = observation_2.iloc[149:195]
plt.plot(observation_3['Year'],observation_3['CO2_ppm'],"k .",label='observations')

# 显示主副刻度线
axes.minorticks_on()
plt.yticks(ticks=np.linspace(260,380,7), fontsize=10)
axes.tick_params(axis="y", which="minor", direction="in", width=0.5, length=2)
plt.xticks(ticks=np.linspace(1750,2000,6), fontsize=10)
axes.tick_params(axis="x", which="minor", direction="in", width=0.5, length=2)

plt.legend()
plt.xlabel('Year')
plt.ylabel('CO$_2$ concentration(ppm)')
plt.show()
```



380
360
340
320
300
280
260
CO2 concentration(ppm)
β=0.38
β=0.50
observations
1750
1800
1850
1900
1950
2000
Year