Concentration de CO2 dans l'atmosphère depuis 1958

Miguel Arpa Perozo

30 novembre 2020

Table des matières

1 Introduction

1.1 Objectifs

1. Proposer un modèle de la variation de CO2 dans l'atmosphère en fonction du temps depuis 1958 à l'aide de différentes mesures.
2. À l'aide du modèle, prédire la concentration future de CO2 en 2025.

1.2 Généralités sur les données utilisées

Voici quelques points généraux sur les données utilisées dans cette étude :
- Le lien de téléchargement des données est le suivant : link
- Les données ont été téléchargées le 2020-11-16 (date au format ISO).
- D'après le site de téléchargement, la concentration de CO2 est donnée en [micro-mol CO2 per more] (ppm).
- Finalement, d'après le site : "The weekly values have been adjusted to 12 :00 hours at middle day of each weekly period."

1.3 Description des démarches adoptées.

Deux démarches ont été adoptées afin de déterminer un modèle de la concentration de C02 en fonction des mesures :

1. Dans un premier temps, effectuer une analyse fréquentielle en utilisant la transformée de Fourier des mesures afin de caractériser certains phénomènes.
2. Utiliser un outil d'optimisation pour déterminer les paramètres d'un modèle proposé par simple observation des mesures.

1.4 Outils utilisés

1. Python 3.8.6 (GCC 10.2.0)
2. numpy 1.19.2
3. scipy 1.5.3
4. matplotlib 3.3.2

2 Lecture ou téléchargement des données brutes

Les données ont été téléchargées le 2020-11-16. Le lien de téléchargement utilisé est le suivant :

https://scrippsco2.ucsd.edu/assets/data/atmospheric/stations/in_situ_co2/weekly/weekly_in_situ_co2_mlo.csv

On télécharge les données si celles-ci ne sont pas déjà disponibles localement.

```
data_file ="weekly_in_situ_co2_mlo.csv"
import os
import urllib.request

if not os.path.exists(data_file):
    urllib.request.urlretrieve(data_url, data_file)
```

Les données commencent à la ligne 45, donc on ne prend pas en compte les 45 premières lignes du fichier.

```
data = open(data_file, 'rb').read()
lines = data.decode('latin-1').strip().split('\n')
data_lines = lines[45:]
table = [line.split(',') for line in data_lines]
```

On visualise les valeurs du tableau :

```
table[:5]
```

3 Traitement et affichage des données.

Dans cette partie, on convertit les données dans des formats qui permettront de les traiter plus facilement.

3.1 Conversion des *string* en valeurs numériques.

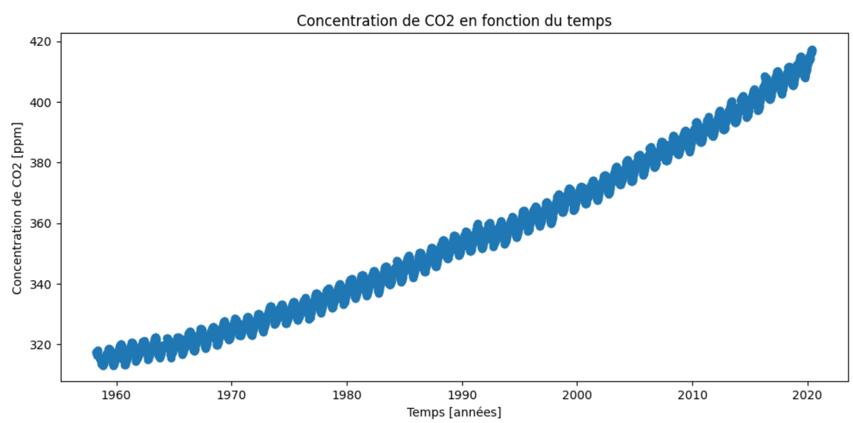
Les données dans `table` sont des string. On va convertir la première colonne en objets `datetime` de Python, et la deuxième colonne en `float`.

```
import datetime
convertedData = [(datetime.datetime.strptime(yearWeekDay, "%Y-%m-%d"), float(co2)) for

dates = [dates for dates, concentration in convertedData]
concentration = [concentration for dates, concentration in convertedData]
```

3.2 Affichage des données.

Ensuite, on convertit les `datetime` dans un format qui peut être utilisé pour l'affichage des données, et on affiche les données brutes.

```
import matplotlib.pyplot as plt
import matplotlib.dates as pltDates

plotDates = pltDates.date2num(dates)

plt.figure(figsize=(10,5))
plt.plot_date(plotDates,concentration)
plt.title("Concentration de CO2 en fonction du temps")
plt.xlabel("Temps [années]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



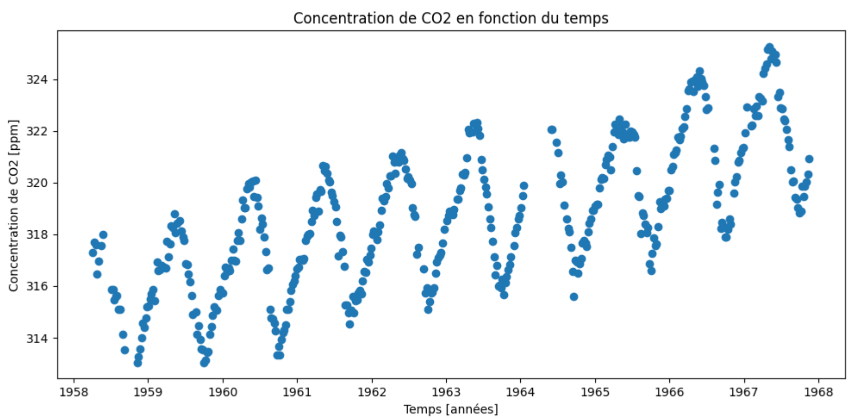
On remarque sur la figure la superposition de deux phénomènes :

1. Un phénomène périodique de faible amplitude à haute fréquence.
2. Une croissance globale, "lente" de forme parabolique.

3.3 *Zoom* sur l'affichage des données

Effectuons un *zoom* sur une période de temps plus courte pour caractériser les oscillations rapides observées précédemment.

```
# Window's size where we want to zoom.
windowSize = 450;
plotDatesZoom = pltDates.date2num(dates[:windowSize])

plt.figure(figsize=(10,5))
plt.plot_date(plotDatesZoom,concentration[:windowSize])
plt.title("Concentration de CO2 en fonction du temps")
plt.xlabel("Temps [années]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



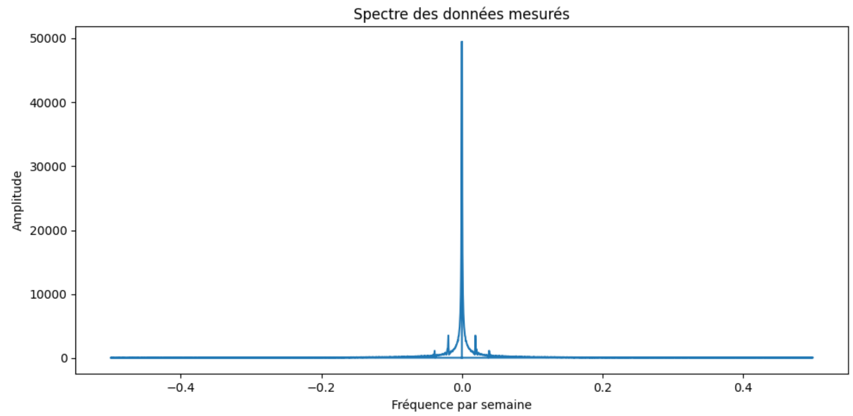
Sur cette figure, on met en évidence le phénomène périodique sur le niveau de concentration de CO2. On peut faire les premières hypothèses suivantes :

1. La période des oscillations est de l'ordre d'une année.
2. L'amplitude varie entre cinq et 7 ppm.

4 Analyse fréquentielle de la variation de la concentration de CO2.

Dans cette partie, on souhaite faire une analyse fréquentielle des données en effectuant une transformée de Fourier sur les mesures afin de mieux caractériser les phénomènes observés précédemment. Pour cela, on utilise le module `numpy` et l'on affiche les résultats avec `matplotlib`.

```
import numpy as np

# Conversion of the data to numpy array
co2 = np.array(concentration)

#Remove mean value
co2NoMean = co2 - np.mean(co2)
# Concentration FFT
concentrationFFT = np.fft.fft(co2NoMean)
# We take the norm of the FFT
concentrationFFTNorm  = np.absolute(concentrationFFT)
concentrationFFTAngle = np.angle(concentrationFFT)

# Frequency axis
N  = co2.shape[0] # Number of samples
Te = 1            # Sampling interval
t  = np.arange(N)  # time reference

#freq = np.arange(N)/(N*Te)
freq = np.fft.fftfreq(N)*N*(1/N*Te)


plt.figure(figsize=(10,5))
plt.plot(freq,concentrationFFTNorm)
plt.title("Spectre des données mesurés")
plt.xlabel("Fréquence par semaine")
plt.ylabel("Amplitude")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```

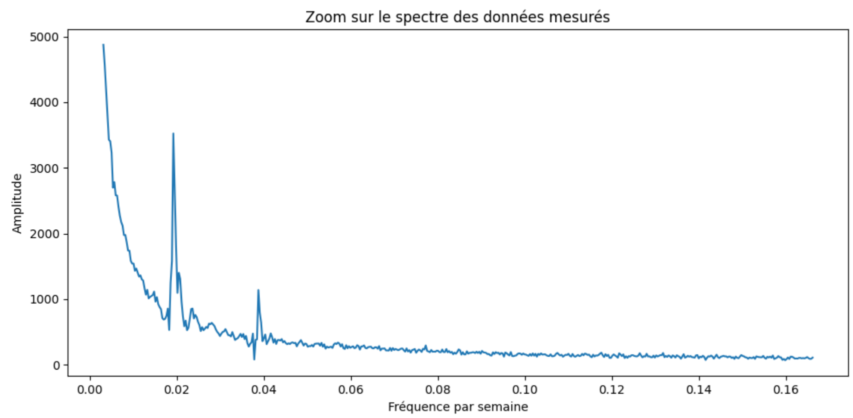


Sur ce graphique, on peut voir l'ensemble du spectre des mesures. On remarque quatres pics qui devraient correpondre au phénomène périodique. On tentera dans la prochaine partie de caractériser plus précisément ces pics.

4.1 *Zoom* sur le spectre du signal

On effectue un zoom sur la partie positive du graphique pour tenter de caractériser les différentes fréquences d'oscillations du signal.

```
startIndx = 10
fftzoomIndx = int(N/6)

plt.figure(figsize=(10,5))
plt.plot(freq[startIndx:fftzoomIndx],concentrationFFTNorm[startIndx:fftzoomIndx])
plt.title("Zoom sur le spectre des données mesurés")
plt.xlabel("Fréquence par semaine")
plt.ylabel("Amplitude")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```

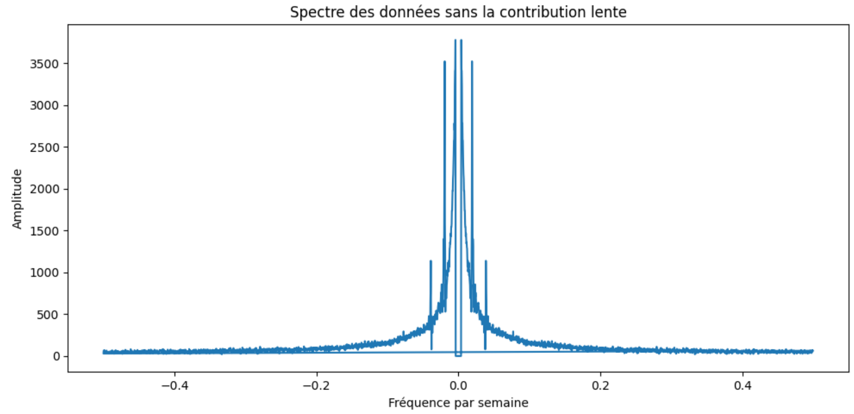


On remarque sur le spectre du signal que le phénomène périodique possède deux pics d'amplitude d'environ 1400 et 4000. On suppose que le premier pic avec une plus grande amplitude correspond au phénomène périodique que l'on a mis en évidence dans l'observation des données brutes. En effet, on a supposé précédemment que les oscillations avaient une période d'une année, notre unité de temps étant une semaine, on peut déduire la fréquence des oscillations en considérant qu'une année a 52 semaine : $f = \frac{1}{52} \approx 0.019$ [/semaines] ce qui reste cohérent avec les résultats dans le graphique.

4.2 Filtrage de la contribution lente

On élimine toutes les valeurs supérieures au seuil établi précédemment (4000) dans le spectre pour éliminer la contribution lente dans le signal.

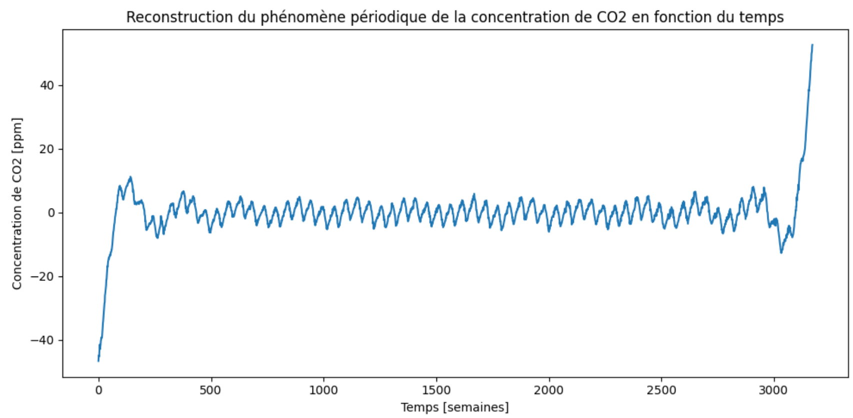
```
# Extraction of the small oscillations
smallAmplitude = 4000
smallNorm = np.copy(concentrationFFTNorm)
smallNorm[smallNorm > smallAmplitude] = 0

plt.figure(figsize=(10,5))
plt.plot(freq,smallNorm)
plt.title("Spectre des données sans la contribution lente")
plt.xlabel("Fréquence par semaine")
plt.ylabel("Amplitude")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



4.3 Reconstruction du phénomène périodique

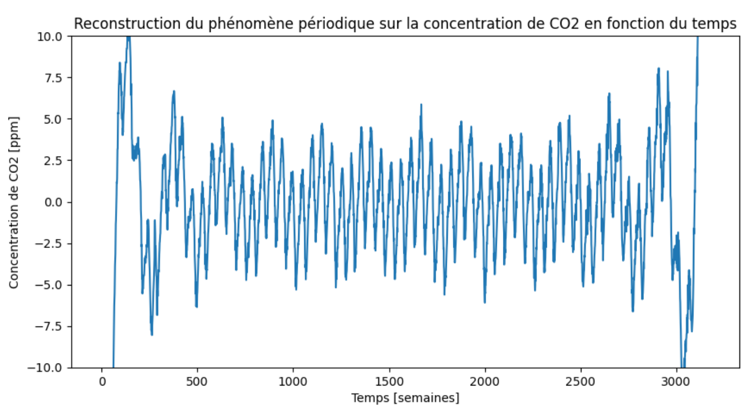
L'objectif est de déterminer l'amplitude des oscillations en prenant en compte toutes les mesures et non pas juste sur une fenêtre d'observation comme on l'a fait précédemment.

```
smallFreqSignal = smallNorm*np.exp(1j*concentrationFFTAngle)
smallSignal = np.fft.ifft(smallFreqSignal)
smallSignal = np.real(smallSignal)

plt.figure(figsize=(10,5))
plt.plot(t,smallSignal)
plt.title("Reconstruction du phénomène périodique de la concentration de CO2 en fonctio
plt.xlabel("Temps [semaines]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



Effectuons un zoom sur le signal reconstruit :

```
plt.figure(figsize=(10,5))
plt.plot(t,smallSignal)
plt.ylim(-10,10)
plt.title("Reconstruction du phénomène périodique sur la concentration de CO2 en fonct
plt.xlabel("Temps [semaines]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.savefig(matplot_lib_filename)
matplot_lib_filename
```

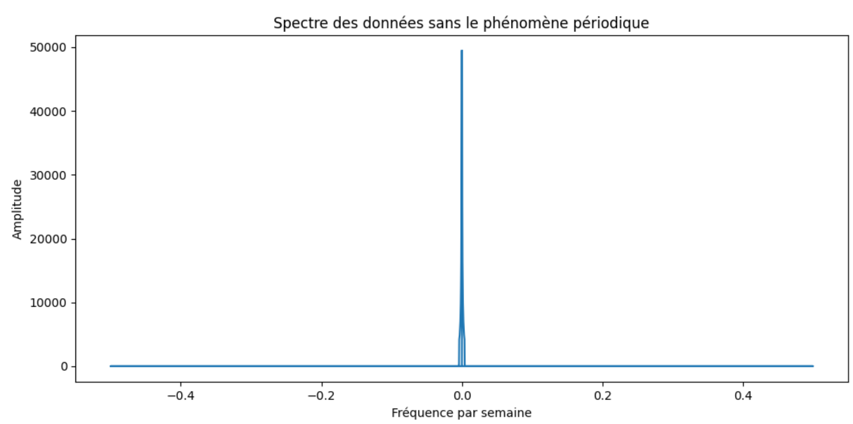


On peut déduire que le phénomène périodique a une amplitude de cinq d'après le graphique.

4.4 Filtrage du phénomène périodique

On souhaite maintenant filtrer le phénomène périodique ce qui pourrait nous aider a mieux comprendre la contribution lente. On élimine donc toutes les valeurs inférieures au seuil établit précédemment.

```
# Extraction of the big oscillations
bigAmplitude = 4000
bigNorm = np.copy(concentrationFFTNorm)
bigNorm[bigNorm < bigAmplitude] = 0

plt.figure(figsize=(10,5))
plt.plot(freq,bigNorm)
plt.title("Spectre des données sans le phénomène périodique")
plt.xlabel("Fréquence par semaine")
plt.ylabel("Amplitude")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```

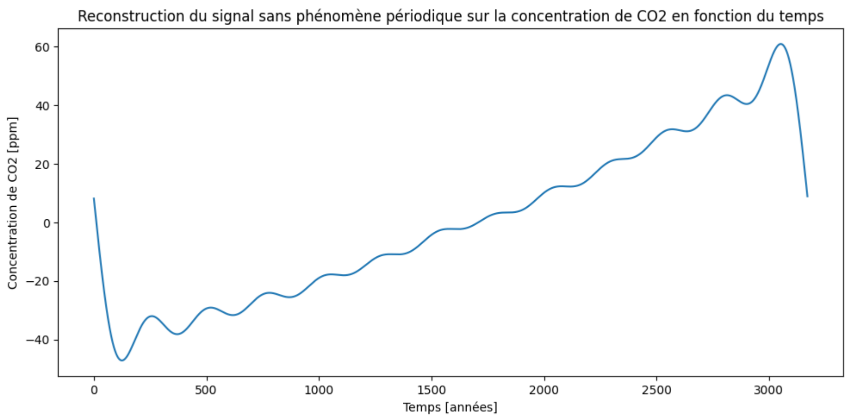


4.5 Reconstruction de la contribution lente du signal

Finalement on reconstruit le signal temporel puis on l'affiche afin de mieux le caractériser.

```
bigFreqSignal = bigNorm*np.exp(1j*concentrationFFTAngle)
bigSignal = np.fft.ifft(bigFreqSignal)
```

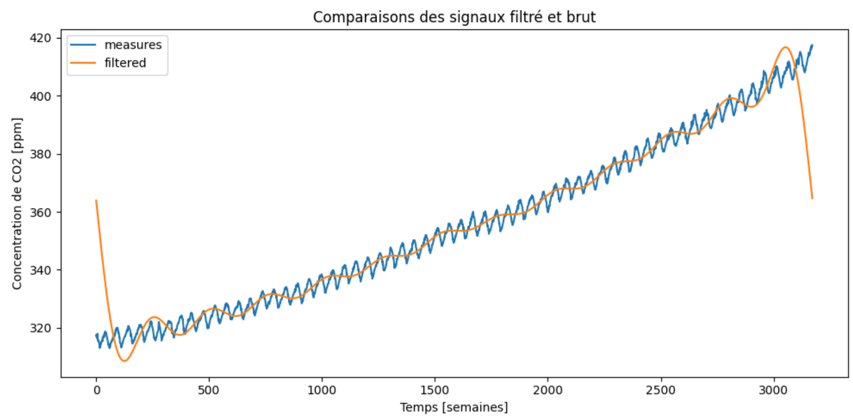
```
bigSignal = np.real(bigSignal)

plt.figure(figsize=(10,5))
plt.plot(t,bigSignal)
plt.title("Reconstruction du signal sans phénomène périodique sur la concentration de C
plt.xlabel("Temps [années]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



Ce graphique ne nous permet pas de déduire grand chose quant à cette contribution lente. Tentons de le comparer avec les mesures.

4.6 Comparaison des signaux filtrés et brut

```
plt.figure(figsize=(10,5))
plt.plot(t,co2,label="measures")
plt.plot(t,bigSignal+np.mean(co2),label="filtered")
plt.legend()
plt.title("Comparaisons des signaux filtré et brut")
plt.xlabel("Temps [semaines]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



Malheuresement, cette approche ne nous permet pas d'avoir plus d'informations pertinentes quant à cette contribution lente. Donc par la suite, on modélisera cette contribution par un polynôme d'ordre 2.

5 Modélisation des phénomènes et extrapolations

5.1 Oscillations périodiques

L'analyse fréquentielle nous a permis caractériser le phénomène périodique par un signal de fréquence $f = 0.02$ par unité de temps (par semaine), et avec une amplitude de 5 [ppm]. Ainsi, soit $s_p(t)$ le signal périodique avec t en semaine, on a :

$$s_p(t) = 5\cos(0.02t) \tag{1}$$

5.2 Contribution lente du signal

L'analyse fréquentielle ne nous a pas permis d'avoir plus d'information quant à la contribution lente du signal. Cependant, si l'on regarde l'allure générale des données, on peut supposer que le concentration de CO2 varie de façon quadratique par rapport au temps. Donc, on va modéliser cette contribution lente par un polynôme d'ordre deux $y(t) = at^2 + bt + c$. On utilisera un outil de *curve fitting* pour caractériser ce polynôme.

Compte tenu de la modélisation du phénomène périodique, celui-ci possède une valeur moyenne nulle. Or les outils de *curve fitting* calculent la norme deux de l'erreur entre le modèle et les données, donc il est inutile de filtrer les données brute pour tenter d'enlever le phénomène périodique pour caractériser la contribution lente.

Dans un premier temps, on définit une fonction de notre modèle pour la contribution lente

```
def slowContributionModel(t,a,b,c):
    return a*t*t + b*t + c
```

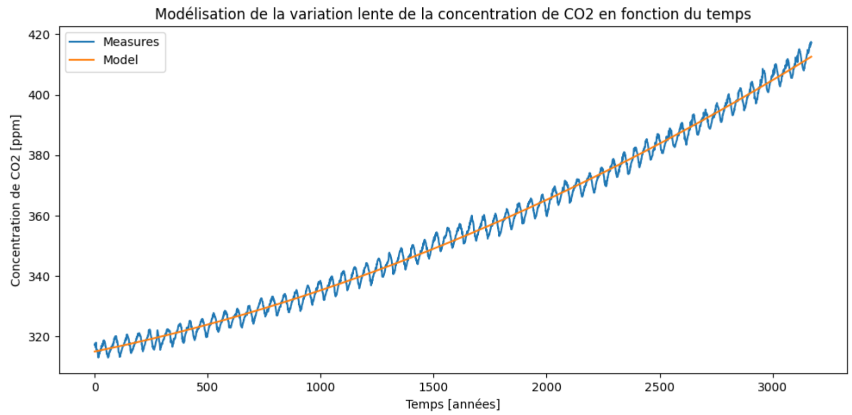
Puis l'on utilise le module `scipy` plus spécifiquement le module `scipy.optimize` pour déterminer les paramètres de notre modèle.

```
from scipy import optimize
params, params_covariance = optimize.curve_fit(slowContributionModel,t,co2, p0=[1, 1, 1
print("Parameters :\t",params)
print("Parameters covariance :\t", params_covariance)
```

```
Parameters :  [4.83379437e-06 1.54227807e-02 3.15061312e+02]
Parameters covariance :  [[ 2.85416803e-15 -9.05056674e-12  4.78171606e-09]
 [-9.05056674e-12  3.06138429e-08 -1.81982548e-05]
 [ 4.78171606e-09 -1.81982548e-05  1.44289419e-02]]
```

Finalement on compare le modèle obtenu avec les données des mesures :

```
co2Model = slowContributionModel(t,params[0], params[1], params[2])

plt.figure(figsize=(10,5))
plt.plot(t,co2,label="Measures")
plt.plot(t,co2Model,label="Model")
plt.legend()
plt.title("Modélisation de la variation lente de la concentration de CO2 en fonction du
plt.xlabel("Temps [années]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



On constate que l'on obtient des résultats satisfaisants pour la caractérisation de la contribution lente.

5.3 Extrapolation de la concentration de CO2 jusqu'à 2025

Dans un premier temps, on crée une liste qui contient l'ensemble des données et l'on rajoute des dates jusqu'à la première semaine de 2025.

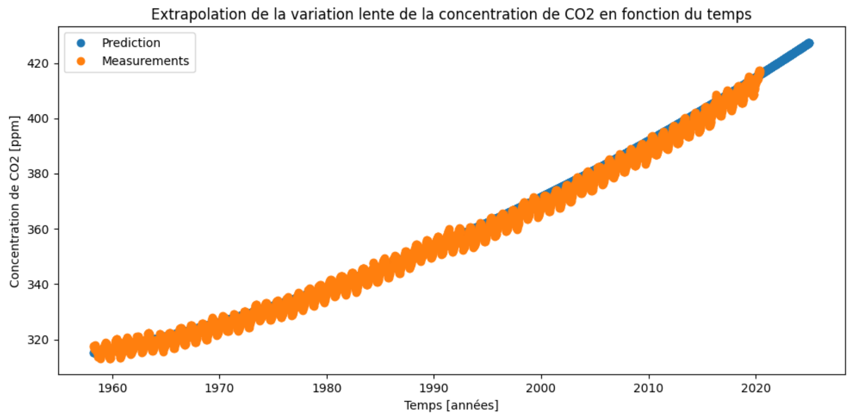
```
initialDate = dates[0]
finalDate   = datetime.datetime(year=2025,month=1,day=1)
deltaWeeks = finalDate -initialDate
deltaWeeks = int(deltaWeeks.days/7) + 2

newDates = [initialDate + datetime.timedelta(weeks=i) for i in range(deltaWeeks)]
newDates[-1]
```

```
2025-01-04 00:00:00
```

Finalement, on extrapole à l'aide du modèle puis on affiche les résultats.

```
timePrediction = np.arange(len(newDates))
co2Prediction  = slowContributionModel(timePrediction,params[0], params[1], params[2])
newPlotDates   = pltDates.date2num(newDates)

plt.figure(figsize=(10,5))
plt.plot_date(newPlotDates,co2Prediction,label="Prediction")
plt.plot_date(plotDates,concentration,label="Measurements")
plt.legend()
```

```
plt.title("Extrapolation de la variation lente de la concentration de CO2 en fonction d
plt.xlabel("Temps [années]")
plt.ylabel("Concentration de CO2 [ppm]")
plt.tight_layout()

plt.savefig(matplot_lib_filename)
matplot_lib_filename
```



On finit par calculer l'augmentation de concentration en pourcentage par rapport à notre dernière mesure :

```
augmentation = (co2Prediction[-1] - concentration[-1])/ concentration[-1] *100
print("Augmentation de la concentration en % : ", augmentation)
```

```
Augmentation de la concentration en % :  2.4518838408355386
```

6 Conclusion

Cette étude propose un modèle de la variation de concentration de CO2, puis utilise ce modèle afin de prédire l'augmentation de la concentration de C02 en 2025 par rapport à la dernière mesure.

On considère que l'on peut caractériser la variation de la concentration de C02 en deux phénomènes dont on propose un modèle mathématique :

1. Une variation périodique dont la période correspond à une année et d'amplitude 5 ppm.
2. Une variation "globale" modélisée par un polynôme d'ordre 2.