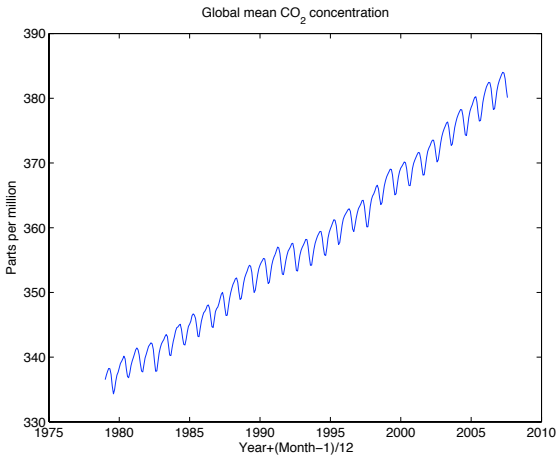
Bayesian linear and Gaussian process regression. The following time series of monthly mean global CO_2 concentrations can be obtained from the file `co2.txt` (original data obtained from http://www.esrl.noaa.gov/gmd/ccgg/trends):



We will apply Bayesian linear and Gaussian process regression to predict the CO_2 concentration $f(t)$ as a function of time t, where $t = \text{Year} + (\text{Month} - 1)/12$.

(a) First we model the function using linear regression, that is, using the functional form

$$f(t) = at + b + \epsilon(t),$$

with i.i.d. noise residual $\epsilon(t) \sim \mathcal{N}(0, 1)$ and prior $a \sim \mathcal{N}(0, 10^2)$, $b \sim \mathcal{N}(360, 100^2)$. Compute (using MATLAB or another package) the posterior mean and covariance over a and b given the CO_2 data.

(b) Let $a_{\text{MAP}}, b_{\text{MAP}}$ be the MAP estimate in the question above. The residual is the difference between the observed function values and the predicted mean function values

$$g_{\text{obs}}(t) = f_{\text{obs}}(t) - (a_{\text{MAP}}t + b_{\text{MAP}}),$$

where $f_{\text{obs}}(t)$ is the observed value of the CO_2 concentration at time t.
Plot $g_{\text{obs}}(t)$. Do you think these residuals conform to our prior over $\epsilon(t)$? State, with justifications, which characteristics of the residual you think do or do not conform to our prior belief.

(c) Write a MATLAB (or other language) function to generate samples drawn from a GP. Specifically, given a covariance kernel function $k(\cdot, \cdot)$ and a vector of input points $\mathbf{x}$, return a function $f(\mathbf{x})$ evaluated on the input points $\mathbf{x}$ drawn randomly from a GP with the given covariance kernel and with zero mean.

(d) Test your function by plotting sample functions drawn from the following kernel, for various settings of the hyperparameters

$$k(s,t) = \theta^2 \left(\exp\left(-\frac{2\sin^2(\pi(s-t)/\tau)}{\sigma^2} \right) + \phi^2 \exp\left(-\frac{(s-t)^2}{2\eta^2} \right) \right) + \zeta^2 \delta_{s=t} \qquad (1)$$

Describe the characteristics of the drawn functions, and how the characteristics of the functions depend on the parameters. [5 marks]

covariance_mtx = Σ

sig = chol $\left(\Sigma + \begin{pmatrix} 0.001 & & 0 \\ & \ddots & \\ 0 & & 0.001 \end{pmatrix} \right)$.

Sample = sig $\times \begin{pmatrix} N(0,1) \\ \vdots \end{pmatrix} + \begin{pmatrix} 0 \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \end{pmatrix}$

(e) Suppose we were to consider modelling the residual function $g(t)$ using a zero mean GP with the covariance kernel above. Based on the plot of $g(t)$ and your explorations in the preceding part, what do you think will be suitable values for the hyperparameters of k?

(f) [**Bonus**] Extrapolate the CO_2 concentration levels to 2020 using the GP with covariance kernel k of eqn 1, and your chosen parameter values. Specifically, compute the predictive mean and variance of the residual $g(t)$ for every month between September 2007 and December 2020 given the observed residuals $g_{\text{obs}}(t)$. Plot the means and one standard deviation error bars of the extrapolated CO_2 concentration levels

$$f(t) = a_{\text{MAP}} t + b_{\text{MAP}} + g(t)$$

along with the observed CO_2 levels. Does the behaviour of the extrapolation conform to your expectations? How sensitive are your conclusions to settings of the kernel hyperparameters?

(g) [**Bonus**] Why is the above procedure not fully Bayesian? How would we go about modelling $f(t)$ in a Bayesian framework?

(a) Consider a general Bayesian Regression problem:

$$y = \phi^T(x) w + \varepsilon,$$
$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$
$$w \sim \mathcal{N}(m_0, S_0).$$

i.e. we have a prior that $p(w) = \mathcal{N}(m_0, S_0)$.

By the Baye's theorem, the posterior over parameters is:

$$p(w|X,y) = \frac{p(y|X,w)\cdot p(w)}{p(y|X)} \quad \cdots \circledast$$

where $p(y|X) = \int p(y|X,w)\cdot p(w)\, dw$. is the evidence (marginal likelihood). ← independent of w

$$p(y|X,w) = \mathcal{N}(y|\phi w, \sigma^2 I).$$

CLAIM: posterior of parameters: $p(w|X,y) = \mathcal{N}(w|S_N, m_N)$,
where $S_N = (S_0^{-1} + \sigma^{-2}\phi^T\phi)^{-1}$
$m_N = S_N(S_0^{-1}m_0 + \sigma^{-2}\phi^T y)$.

Proof: From $\circledast$, $p(w|X,y) \propto p(y|X,w)\cdot p(w)$
Gaussian Gaussian Gaussian

Take log of RHS:

$$\log \mathcal{N}(y|\phi w, \sigma^2 I) + \log \mathcal{N}(w|m_0, S_0).$$ ← Independent of w

$$= -\frac{1}{2}\left[\sigma^{-2}(y-\phi w)^T(y-\phi w) + (w-m_0)^T S_0^{-1}(w-m_0)\right] + \text{const}$$

$$= -\frac{1}{2}\left[\sigma^{-2}y^Ty - 2\sigma^{-2}y^T\phi w - \sigma^{-2}w^T\phi^T\phi w + w^TS_0^{-1}w - 2m_0^TS_0^{-1}w + m_0^TS_0^{-1}m_0\right] + \text{const}$$

$$= -\frac{1}{2}\left[w^T(\sigma^{-2}\phi^T\phi + S_0^{-1})w - 2(\sigma^{-2}y^T\phi + m_0^TS_0^{-1})w\right] + \text{const}'$$

$$= -\frac{1}{2}\left[w^T(\sigma^{-2}\phi^T\phi + S_0^{-1})w - 2(\sigma^{-2}\phi^Ty + S_0^{-1}m_0)^Tw\right] + \text{const}'$$

Hence $p(w|X,y) \propto \exp\left\{-\frac{1}{2}\left[w^T(\sigma^{-2}\phi^T\phi + S_0^{-1})w - 2(\sigma^{-2}\phi^T y + S_0^{-1}m_0)^T w\right]\right\}$

compare $p(w|X,y) \propto \exp\left\{-\frac{1}{2}(w-m_N)^T S_N^{-1}(w-m_N)\right\}$

$$= \exp\left\{-\frac{1}{2}\left[w^T S_N^{-1} w - 2m_N^T S_N^{-1} w + m_N^T S_N^{-1} m_N\right]\right\}$$

$$\Rightarrow \begin{cases} S_N^{-1} = (\sigma^{-2}\phi^T\phi + S_0^{-1}) \\ m_N^T S_N^{-1} = (\sigma^{-2}\phi^T y + S_0^{-1}m_0)^T \end{cases} \Leftrightarrow \begin{cases} S_N = (\sigma^{-2}\phi^T\phi + S_0^{-1})^{-1} \\ m_N = S_N(\sigma^{-2}\phi^T y + S_0^{-1}m_0) \end{cases} \quad \square$$

---------Proof completed ---------

In this case,

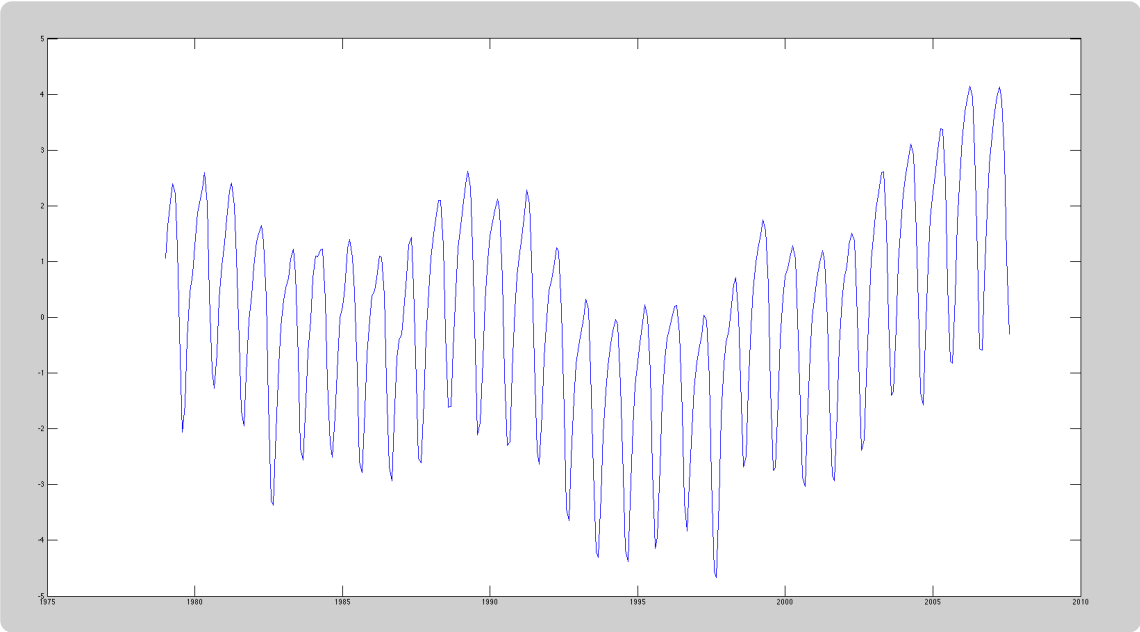
$\sigma^2 = 1$, $S_0 = \begin{pmatrix} 10^2 & 0 \\ 0 & 100^2 \end{pmatrix}$, $m_0 = \begin{pmatrix} 0 \\ 360 \end{pmatrix}$, $\phi = \begin{pmatrix} \cdots & t & \cdots \\ \cdots & 1 & \cdots \end{pmatrix}$

Plug in matlab, we get m_N (posterior_mean)

S_N (posterior_covariance)

```
posterior_covariance =
   1.0e+02 *
   0.000000417453539  -0.000832106417939
  -0.000832106417939   1.658659376255700
posterior_mean =
   1.0e+03 *
   0.001572033536338
  -2.775586453638971
>>
```

(b)



The residuals DO NOT confirm our prior over $\varepsilon(t)$.

It is given $\varepsilon(t) \sim \mathcal{N}(0,1)$.

But the variance we get for g_{obs} is 3.324, which is much greater than 1.

Also this can be spotted from the plot.

```
g_obs_variance =
    3.323762842551880
>>
```

(c) Function 1 : covarianceKernelFunction

covarianceKernelFunction.m | SampleFromGP.m | generateCovarianceMatrix.m

```
function [ cov ] = covarianceKernelFunction(X1,X2,parameters)

    % split the parameters
    theta = parameters(1);
    tau = parameters(2);
    sigma = parameters(3);
    phi = parameters(4);
    eta = parameters(5);
    zeta = parameters(6);

    if X1 == X2
       tail = zeta^2 * 1;
    else
       tail = 0;
    end

    cov =  theta^2 * (exp(( (-2) * (sin(pi*(X1-X2)/tau)^2 ))/(sigma^2)) ...
           + phi^2 * exp((-(X1-X2)^2)/(2*eta^2))) + tail;

end
```

Function 2 : generateCovarianceMatrix

covarianceKernelFunction.m | SampleFromGP.m | generateCovarianceMatrix.m

```
function covariance_matrix = generateCovarianceMatrix(t,parameters)
    covariance_matrix = zeros(length(t),length(t));
    for i = 1:length(t)
        for j = 1:length(t)
            covariance_matrix(i,j) = covarianceKernelFunction(t(i),t(j),parameters);
        end
    end
```

Function 3 : SampleFromGP

covarianceKernelFunction.m | SampleFromGP.m | generateCovarianceMatrix.m

```
function sample = SampleFromGP(mean,covariance_matrix)

    % Cholesky decomposition
    sig = chol(covariance_matrix + eye(size(covariance_matrix))*0.00000001, 'lower');
    % generate a vector (seed) with mean 0, variance 1
    unit_vector = randn(size(covariance_matrix,1),1);
    % project to generate the final sample
    sample = sig * unit_vector + mean;

end
```

(d) θ : determines the whole amplitude

large $\theta \Rightarrow$ large osscilation amplitude.

σ : determines length of the "wiggles" in the function.

τ : determines the repitition of the peak.

i.e. the "small" period. In this case, $\tau = 1$ as it represents each year as a period.

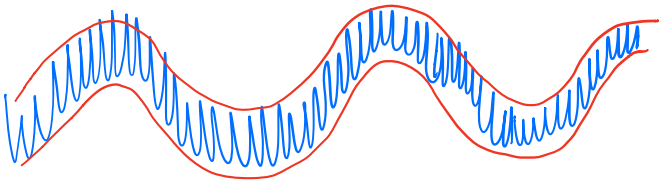
ϕ : the magnitude of how "twisted" the function is.

η : the period of twist. "large" period.

ξ : magnitude of noise.

Consider an example :

τ, σ determines the "small" osscilation in BLUE



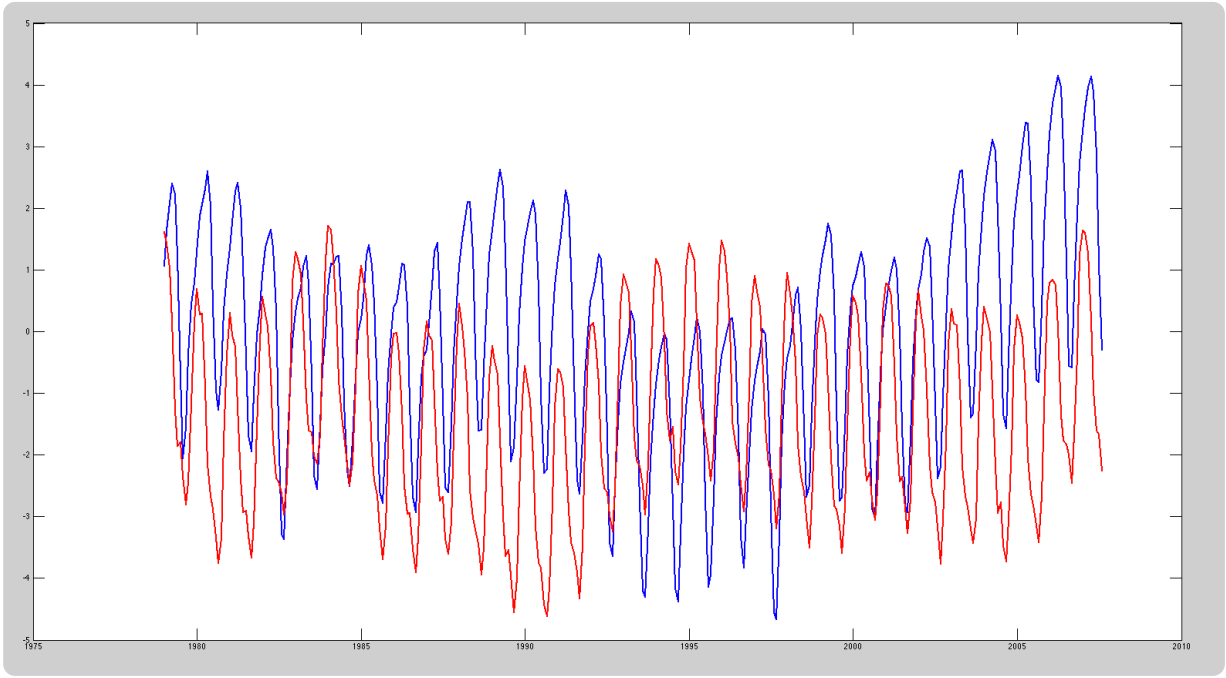
ϕ, η determine the "large" osscilation in RED

(e)

```
19 -    theta = 1.85;
20 -    tau = 1;
21 -    sigma = 1.3;
22 -    phi = 0.52;
23 -    eta = 1.36;
24 -    zeta = 0.08;
25 -    parameters = [theta tau sigma phi eta zeta];
```

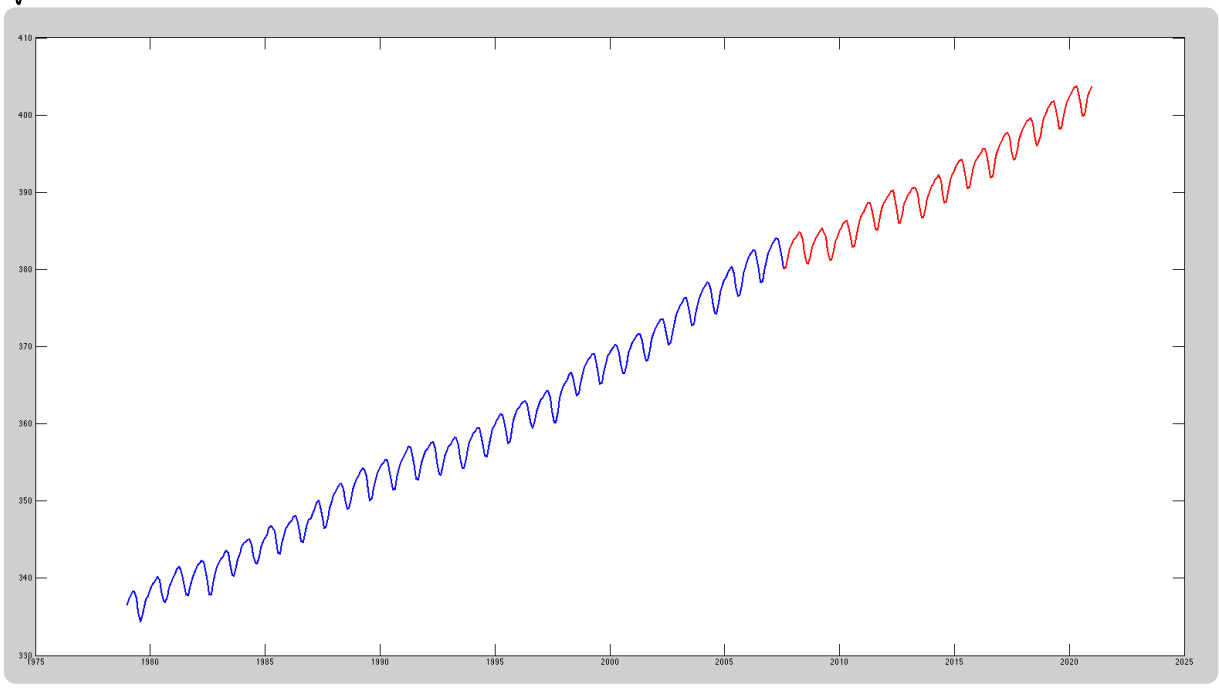
My choice : $\theta = 1.85$, $\tau = 1$ (one year), $\sigma = 1.3$, $\phi = 0.52$, $\eta = 1.36$

$\xi = 0.08$



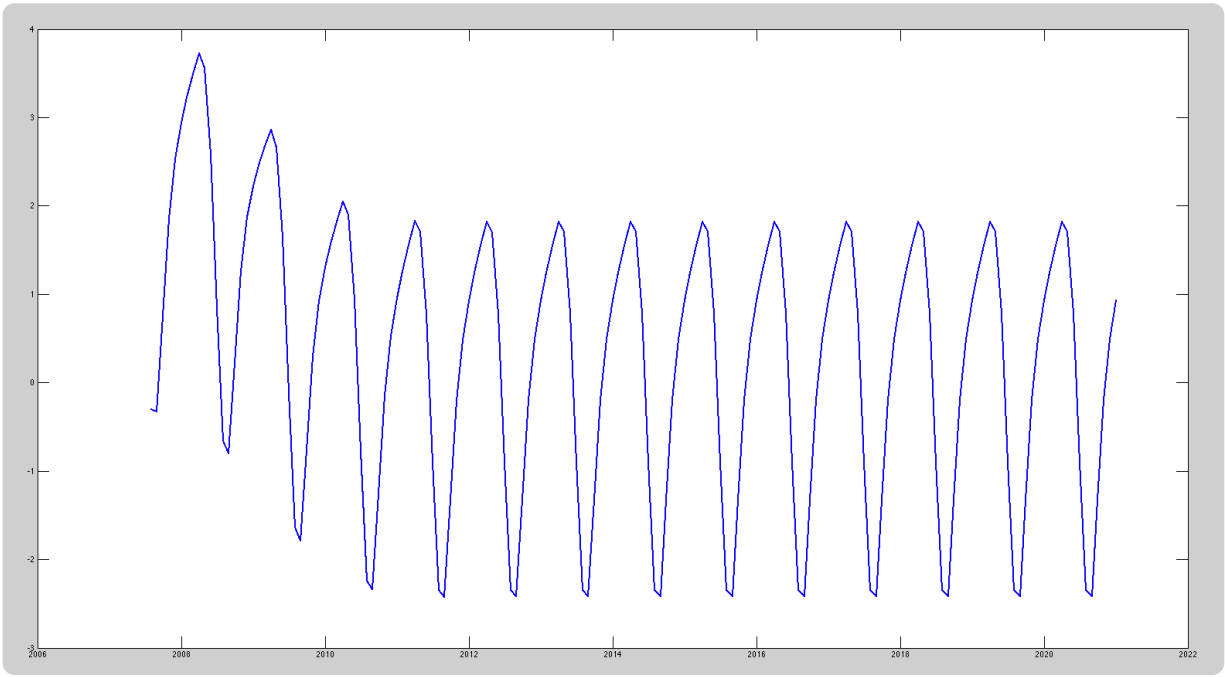
Plot modelled error (Red) with true error (Blue) with the given hyperparameters.

(f)

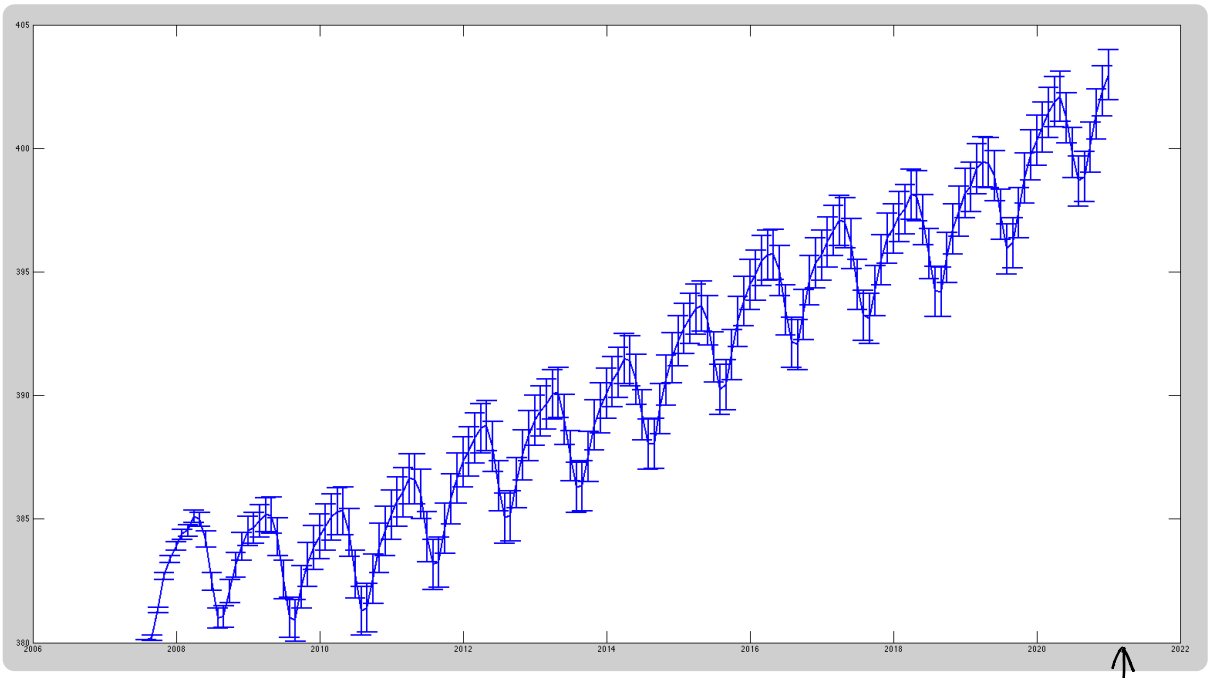


BLUE : orginal data Red : predicteed data

Predictive mean plot



With error bar ↓ (errors are diagnol of covariance matrix)



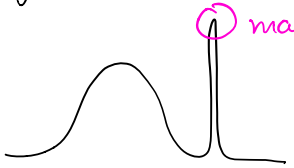
Narrow error bar at beginning ⋮ Wide error bar, further from data

The extrapolation conforms my expectation.

* it oscillates every year in the "small" period.
* there is a "large" period, which slightly twists the curve.
  the "large period" is around 18 years, which has been shown by the prediction.

How sensitive: It is very sensitive to the parameters.
if we significantly change the parameters,
the shape will be changed significantly.

(9). The procedure is NOT fully Bayesian since MAP is a point estimate, it might find the highest peak not the largest density.



How to modify to make it fully Bayesian?

Starting with a non-linear regression problem $f(x) = w^T \phi(x)$.

$$Y^T | X \sim \mathcal{N}(0_N, \tau^2 \phi^T \phi + \sigma^2 I_N)$$

Predict a single new test data by:

$$y | x, \underline{Y}, \underline{\underline{X}} \sim \mathcal{N}(\tilde{K}_{xx} \tilde{K}_{xx}^{-1} Y^T, \tilde{K}_{xx} - \tilde{K}_{xx}^{-1} \tilde{K}_{xx})$$

and choose a specific kernel $k(x_1, x_2) = \phi(x_1)^T \phi(x_2)$.

Q(3) Appendix: the MATLAB script

```
% load data
clear, clc, close all, load co2.txt -ascii; data = co2; format long
t = data(:,1) + (data(:,2) - 1) / 12;
f_obs = data(:,3); X = [t'; ones(1,length(t))];
w_covariance = [10^2 0; 0 100^2];
w_mean = [0 360]'; sigma = 1; % variance of error
posterior_covariance = (w_covariance^(-1) + sigma^(-2) * X * X')^(-1);
posterior_mean = posterior_covariance * (w_covariance^(-1) * w_mean + sigma^(-2) * X * f_obs );
posterior_covariance
posterior_mean
a_map = posterior_mean(1); b_map = posterior_mean(2);
figure();
g_obs = f_obs - (a_map * t + b_map);
g_obs_variance = std(g_obs)^2
plot(t, g_obs); hold on
% hyperparameters:
theta = 1.85; tau = 1; sigma = 1.3;phi = 0.52; eta = 1.36;  zeta = 0.08;
parameters = [theta tau sigma phi eta zeta];
covariance_matrix = generateCovarianceMatrix(t,parameters);
mean = zeros(length(t),1); % mean is zero, given in the question
error = SampleFromGP(mean,covariance_matrix);
% sample = mvnrnd(mean,covariance_matrix);
plot(t,error,'r')
% generate date to 2020
t_future = (t(length(t)):1/12:2021)';
t_full = [t;t_future];
n = length(t); n_future = length(t_future); n_full = length(t_full);
covariance_matrix_full = generateCovarianceMatrix(t_full,parameters);
% split the full covariance matrix to use the conditional property:
Kaa = covariance_matrix;
Kab = covariance_matrix_full(1:n,n+1:n_full);
Kba = covariance_matrix_full(n+1:n_full,1:n);
Kbb = covariance_matrix_full(n+1:n_full,n+1:n_full);
% distribution of the predicted data:
prediction_mean = Kba * Kaa^(-1) * g_obs;
prediction_covariance = Kbb - Kba * Kaa^(-1) * Kab;
prediction_error = SampleFromGP(prediction_mean, prediction_covariance);
prediction_data = a_map * t_future + ones(n_future,1) * b_map + prediction_error;
figure()
plot(t,data(:,3)); hold on
plot(t_future,prediction_data,'r'); hold on
figure()
% plot errorbars
plot(t_future,prediction_data,'r');
err = sqrt(diag(prediction_covariance)); % diagnol of the predction covariance matrix
errorbar(t_future,prediction_data,  err)
```