CO2 Project Report

Vu Luong

1. CONTEXT

The quantity that I am going to model is the weekly-recorded CO_2 levels from the Mauna Loa Observatory in Hawaii. The measurements span all the way back to 1958, and new measurements are still being added to this day.

This is an important quantity to learn more about, since it (and its predictions) can be used in larger, more complex climate change models. These models can in turn inspire policies that help reduce CO_2 emissions and combat global climate change.

2. MODEL

A simple plot of the data looks like this:

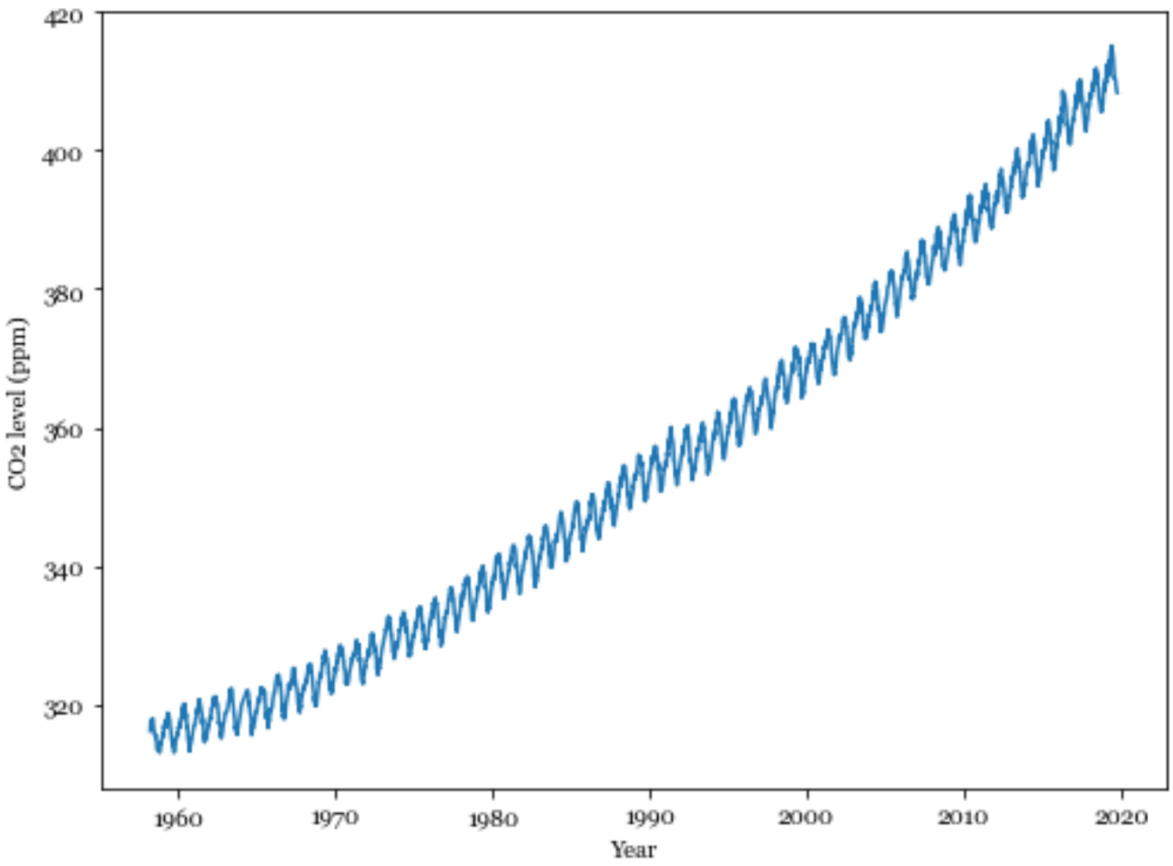


Figure 1. Original data on CO2 levels

There are two components to model: the underlying trend (we see a clear upward trend throughout the years), and seasonal variations (the yearly fluctuations that gives rise to the cyclical movements in the graph).

2.1. UNDERLYING TREND

We can consider 4 candidate functions to model the underlying trend:

$$p_1(x) = \alpha + \beta x$$

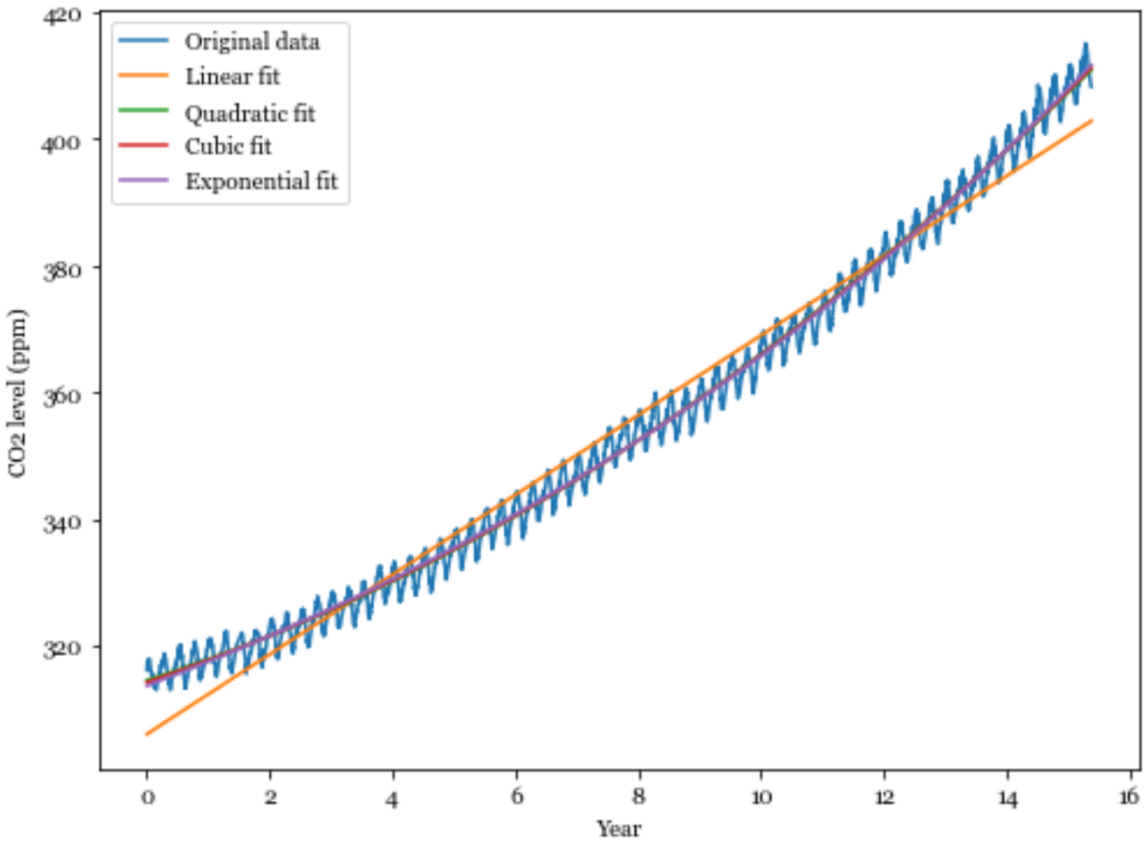
$$p_2(x) = \alpha + \beta x + \gamma x^2$$

$$p_3(x) = \alpha + \beta x + \gamma x^2 + \delta x^3$$

$$\mathrm{e}(x) = \alpha \cdot e^{\beta(x-\gamma)} + \delta$$

The first three are polynomials, and the last is an exponential function.

Using Python, we can find the parameters $(\alpha, \beta, ...)$ that provide the best fit for the data (i.e. result in the smallest sum of squared errors - SSE) (see section 2.1 in Jupyter notebook). The results are as follows:




```
SSE for the best linear fit is: 57141
SSE for the best quadratic fit is: 15871
SSE for the best cubic fit is: 15823
SSE for the best exponential fit is: 15875
```

Figure 2. Best fit lines, and their corresponding SSE measures, for candidate trend functions

We find that the last three options are all much better than the first (the linear fit). We can spot the linear function's inadequacy right from the graph: it fails to account for the small yet not insignificant curvature in the data.

Among the last three options, however, we don't see much of a difference. The cubic fit is ever so slightly better than the quadratic fit, bit the difference (48 points) is negligible. The exponential fit is 4 points worse than the quadratic fit, which is again negligible.

It is best to use the quadratic function. It has approximately the same performance as the other two options, but has the benefit of being less complex. This means that there are either fewer parameters to model, or that the function's form is just simpler (e.g. sums and products vs exponentials). This simplicity will lead to a more robust (i.e. less resistant to changes in its variables and assumptions) model, and an overall smoother inference process (e.g. easier Stan convergence).

2.2. SEASONAL VARIATIONS

We consider 3 candidate functions to model the seasonal variations:

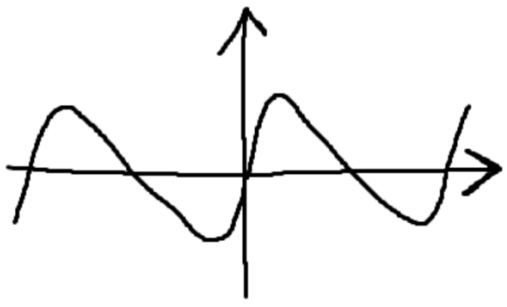
$$s_{\cos}(x) = A \cdot cos(2\pi x + \phi)$$

$$s_{\text{tilted cos}}(x) = A \cdot cos(2\pi x - \delta\cos(2\pi x) + \phi)$$

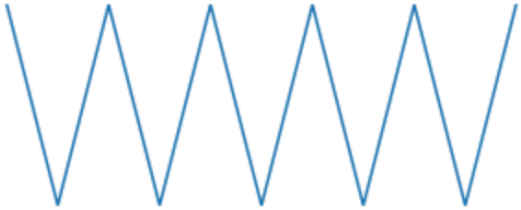
$$s_{\text{mod}}(x) = A \cdot (4 \cdot |(x \bmod 1 + \phi) \bmod 1 - 0.5| - 1)$$

The first function is just a regular weighted cosine function, with amplitude A and phase ϕ.

The second function is slightly more complex. It is the same cosine function, but with a $-\delta\cos(2\pi x)$ component added inside the cosine. This adds a "tilt" to the resulting function (below), with δ determining the direction and degree of the "tilt".



The third function (below) makes use of the peculiar shape of the modulo function, resulting in what looks like the cosine function, but with angular (rather than smooth) spikes and valleys.



Using Python, we can find the parameters (A, ϕ, ...) that produce the best fit for the data (by minimizing the sum of squared errors - SSE) (section 2.2 in the Jupyter notebook). The results are as follows:

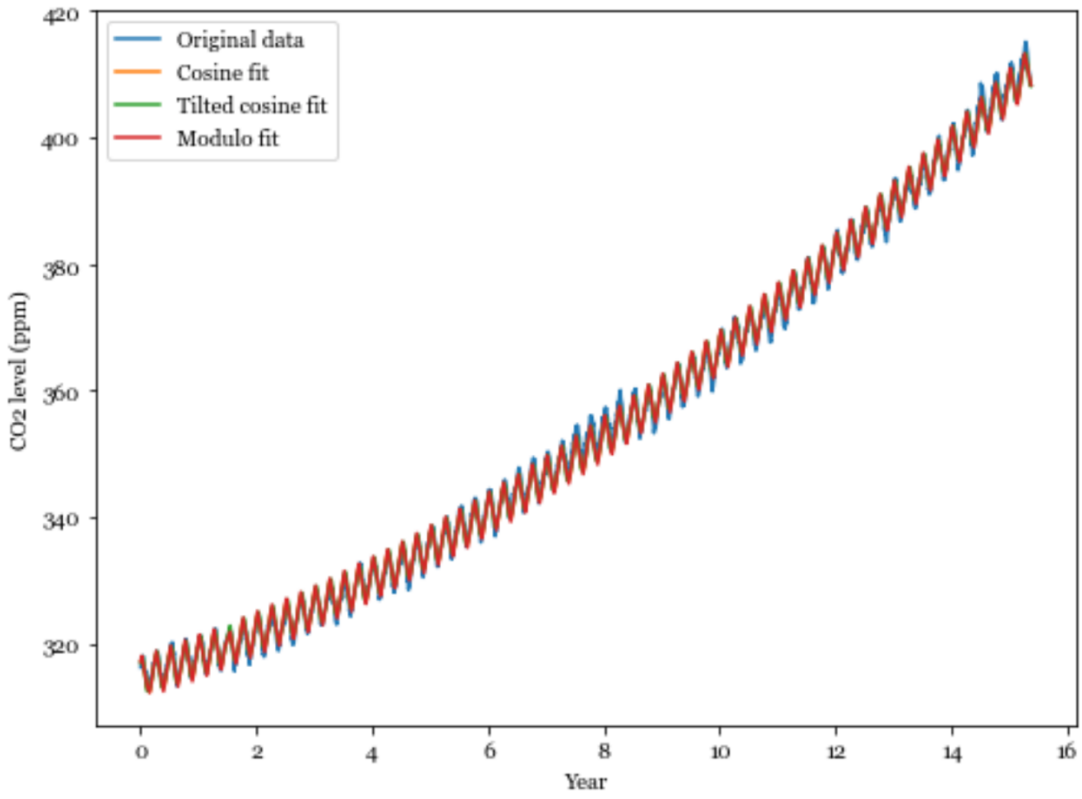


Figure 3. Best fit lines for candidate seasonal variation functions

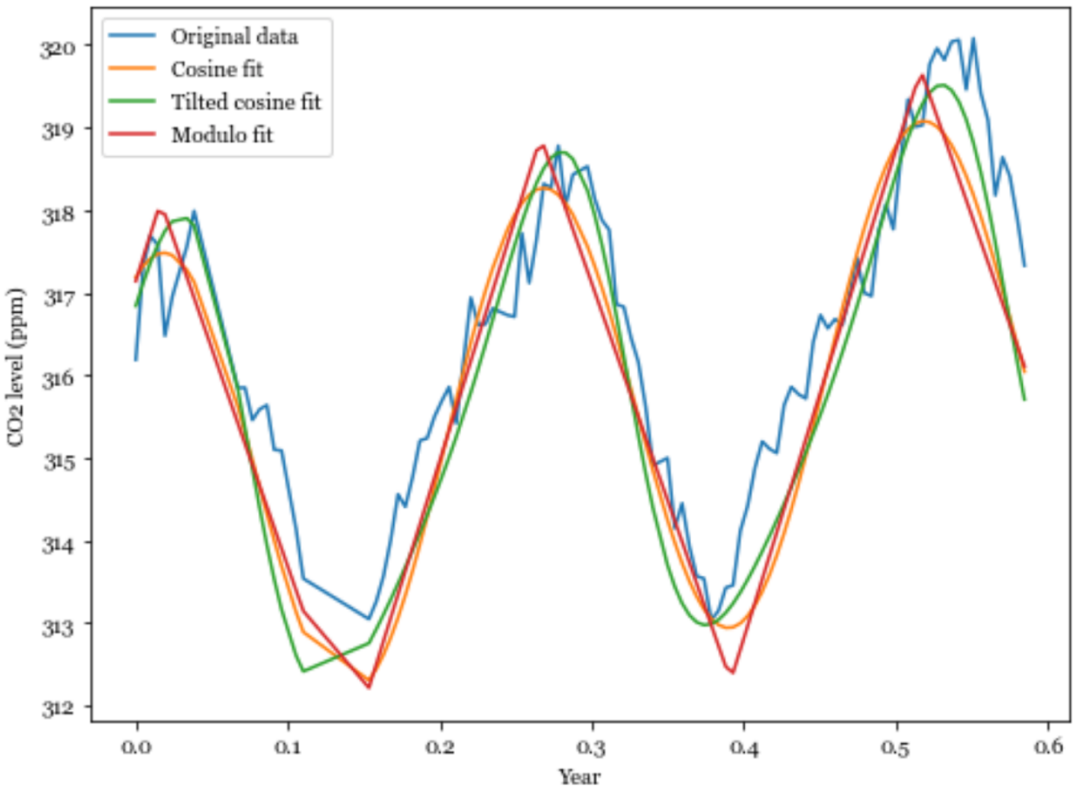


Figure 4. Best fit lines for candidate seasonal variation functions, zoomed in to show only the first 2 years

```
SSE for the best cosine fit is: 3041
SSE for the best tilted cosine fit is: 2439
SSE for the best modulo fit is: 3146
```

The best performer is the tilted cosine function. The second-best is the cosine function. And the worst is the modulo function.

It is highly preferable to use the tilted cosine function, since its best fit version gives us an SSE that is about 400 points lower than that of the cosine function. This is some noticeable and meaningful difference.

However, I tried performing inference with the tilted cosine function multiple times, and Stan's chains failed to converge every time (for more on this, see section 3.2). Thus, we must make do with the second-best option: the cosine function.

2.3. COMPLETE MODEL

The skeleton for our model is the function:

$$\mathrm{f}(t) = \alpha + \beta t + \gamma t^2 + A \cdot cos(2\pi t + \phi)$$

To model the noise, we introduce a Normal distribution with mean $\mathrm{f}(t)$ and standard deviation σ:

$$level \sim Normal(\alpha + \beta t + \gamma t^2 + A \cdot cos(2\pi t + \phi), \sigma)$$

2.4. ASSUMPTIONS

A few basic assumptions include:

- Quadratic trend and cosine seasonal variations (discussed above)
- Homoscedasticity (i.e. σ does not change through time)
- Normality (i.e. the likelihood above follows the Normal distribution).

Another crucial assumption is *stationariness*. This means that the parameters that govern how the model behaves ($\alpha, \beta, \gamma, A, \phi$) don't change through time. If they do, then we'll have to estimate α_0, α_1, ..., α_n, β_0, β_1, ..., β_n, and so on, each for a single time step (with n being the total number of time steps we have in the dataset; at the time of this report, $n = 3139$). This is clearly not computationally feasible. Furthermore, Stan has a limit on the number of parameters it can estimate. So, the stationariness assumption is necessary in pratice.

One way to test if this assumption is plausible is to calculate the ADF (Augmented Dickey-Fuller) statistics. This is used to check if a time series has a unit root. If it does, then the series is non-stationary. This serves as the null hypothesis. The alternate hypothesis is that there is no unit root, so the series is stationary.

Here we have the test result (Jupyter notebook section 2.3):

```
ADF Statistic: -12.2683
p-value: 0.0000
```

The p-value is very close to 0. We can safely reject the null hypothesis. Thus, the data is likely stationary. The stationariness assumption is safe to make.

3. INFERENCE & RESULTS

3.1. QUANTITIES

We have the following model:

$$level \sim Normal(\alpha + \beta t + \gamma t^2 + A \cdot cos(2\pi t + \phi), \sigma)$$

In this model, only $level$ is observed. All the parameters (α, β, ...) are unobserved, so we'll have to use Stan to estimate them.

One hidden variable in this model would be the underlying trend: $\alpha + \beta t + \gamma t^2$. It might be beneficial to separate this trend from the seasonal variation, since these are two different quantities that we're interested in. However, we can't have a hierarchical model (see section 3.2), so the underlying trend can only be a function (and not a Normal distribution like $x_i \sim Normal(\alpha + \beta t_i + \gamma t_i^2, \sigma_x)$). Thus, it is still best to merge the trend and seasonal variation components together like above.

3.2. DIFFICULTIES

Divergence due to ϕ symmetry ($cos(\phi) = cos\,(2\pi - \phi)$)

The Stan chains diverged almost every time with the cosine function. A pair plot of the posterior samples shows that the parameter ϕ has two separate peaks, one very close to zero, and one to the left of 2π. This suggests some sort of symmetry in the sampling.

Indeed, for cosine functions, we have $cos(\phi) = cos\ (2\pi - \phi)$. This means that the posterior samples could be either ϕ or $2\pi - \phi$.

Shifting the bounds of ϕ to the left (from $[0, 2\pi]$ to $[-\pi, \pi]$) fixes this problem. Now, all four chains converge towards the same value for ϕ (around -0.42).

Divergence with tilted cosine function

I attempted to work with the tilted cosine function ($A \cdot cos(2\pi t - \delta \cos\ (2\pi t) + \phi)$), but there were many divergences among the posterior samples. My first intuition was to change the bounds of ϕ and δ, since, like above, there might be some symmetry between the two. However, even the most restrictive bounds ($[0, \pi]$ for both ϕ and δ) failed to curb the divergences. I eventually had to use the second-best option – the normal cosine function – for the Stan model.

Divergence with normal time units

I originally ran the Stan model with normal time units (each unit = 1 day, so consecutive measurements are 7 units apart). However, there were divergences across all parameters. After looking around for a while, I finally found the culprits: β and γ, or their minuscule, close-to-zero values.

With the normal time unit, the best fit values (i.e. the ones that lead to the smallest SSE) for β was ~2.1e-3, and for γ was ~9.7e-8. These are very small, close-to-zero values. Stan has a difficult time converging to these. That's because the posterior samples can easily cross to the negative side, in which case we'll see a negative trend line. This trend line would then be an awful fit for our data, so in these cases Stan adjusts the other parameter values to make up for the difference. Thus, we have a

divergence with multi-modal posterior distributions. There would be one true mode, and one falsy mode that happens when Stan tries to over-compensate.

So, I changed the time unit, making it 1 unit = 4 years. This meant dividing the time column in the data by 365.25*4 = 1461. Now, the best fit values are ~3.1 for β and ~0.21 for γ. This works because in a quadratic function, changing the scale of the input variable (here, it is the time) would lead to a different slope (or steepness) for the best fit line, and thus different coefficient values. β and γ are those coefficients.

Limit (on number of parameters) reached with hierarchical model

I also considered working with a hierarchical model. It would look like:

$$x_i \sim Normal(\alpha + \beta t_i + \gamma t_i^2, \sigma_x)$$

$$y_i \sim Normal(x_i + A \cdot cos(2\pi t_i + \phi), \sigma_y)$$

where x_i is the underlying CO_2 level, and y_i is the observed level, with any seasonal variation added on top. σ_x would account for any variation in the CO_2 level that was not captured by the quadratic model (since α, β, and γ might have changed from t_{i-1} to t_i). σ_y, on the other hand, would account for variations that were not captured by the cosine model, and any measurement errors.

This could be a more complex but "truthful" model, in the sense that it can better capture the randomness in the data (since we have two random variables, each with its own σ, rather than just one).

However, since I would have to estimate all $x_i's$ in my model, and there are, at the time of this report, 3139 time-steps (and thus 3139 x_i's), it would be computationally infeasible to do this. Stan has a limit on the number of parameters it could estimate, and this easily surpassed that limit (I tried). So, I had to settle with a non-hierarchical model.

3.3. PRIORS

I ended up using the following model:

$$level \sim Normal(\alpha + \beta t + \gamma t^2 + A \cdot cos(2\pi t + \phi), \sigma)$$

There are 6 parameters to estimate, so I had to come up with 6 priors. Below are my choices:

- $\alpha \sim Normal(300, 100)$
    - α is the trend offset. It could be either positive or negative, so a Normal would work well.
    - Looking at Figure 1, we see that the data starts at around 300. So, a mean of 300 would be reasonable. We are not sure about this, so the standard deviation could be 100.
- $\beta \sim Normal(0, 5)$
    - Similarly, β, the linear coefficient, could be either positive or negative. We are quite sure that it is positive, since the trend line in Figure 1 is going upwards. But just to be sure, we'll also consider the negative possibility. Thus, a Normal would be a good choice.
    - β should be quite small, since the trend line is not very steep. We could set the mean to a neutral 0, and give it a wide standard deviation of 5, since there is a possibility (although small) that it could reach 10 or more.
- $\gamma \sim Normal(0, 5)$
    - Using the same logic as above, we can use a Normal distribution for γ.
    - γ is the quadratic coefficient and should be very small, since, as we can observe in Figure 1, the quadratic curve upwards is not very strong at all. We can again give it a neutral mean of 0, and a smaller standard deviation of 2, since there isn't much chance that it will be a big number.
- $A \sim Normal(0, 10)$

  - Same logic as above, a Normal distribution is a good choice.
  - A is the coefficient for the cosine function. It could be quite large, since, as observable in Figure 1, the tips and valleys of the data line are quite far apart. A mean of 0 and a standard deviation of 10 should do a good job as a prior.

- $\phi \sim Normal(0, \frac{\pi}{2})$
  - To prevent sampling symmetry (see section 3.2), we set the bounds for ϕ to be $[-\pi, \pi]$. A Normal with mean 0 (right in the middle of the two bounds) and a standard deviation of $\frac{\pi}{2}$ (so that 95% of the mass is inside the bounds) would work well.
- $\sigma \sim Gamma(1, 0.3)$
  - σ is the only parameter that must have a positive support. A Gamma distribution is a good option for such a support.
  - Setting α at 1, and β at 0.3 would result in a distribution with mean ~3.3, and a broad range that could account for many different values of σ.

3.4. STAN MODEL

All of the choices made above make up a complete model to feed into Stan. The code for this is in section 3 in the Jupyter notebook.

3.5. RESULTS

We get the following results from Stan:

```
Inference for Stan model: anon_model_fa04a25ae6cd71ecb67e1c9725a0161c.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

        mean se_mean     sd   2.5%    25%    50%    75%  97.5%  n_eff   Rhat
alpha 314.57  1.5e-3   0.06 314.46 314.53 314.57 314.61 314.67   1376    1.0
beta    3.08  4.4e-4   0.02   3.05   3.07   3.08   3.09   3.11   1371    1.0
gamma   0.21  2.6e-5 1.0e-3   0.21   0.21   0.21   0.21   0.21   1533    1.0
A       2.86  4.5e-4   0.03   2.81   2.84   2.86   2.88   2.91   3035    1.0
phi    -0.42  1.6e-4 8.8e-3  -0.43  -0.42  -0.42  -0.41   -0.4   2885    1.0
sigma   0.99  2.4e-4   0.01   0.96   0.98   0.99   0.99   1.01   2519    1.0
lp__   -1521    0.04   1.71  -1526  -1522  -1521  -1520  -1519   1567    1.0

Samples were drawn using NUTS at Mon Dec 16 21:48:11 2019.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

We see very good convergences (all Rhat values are 1.0) and indications of non-collinearity (all n_eff values are above 1300).

The standard deviations for all parameters are very low. We can also see this in the histograms in the pair plots (Jupyter notebook, section 3). All the posterior distributions are relatively thin and span very small ranges in the x-axis. This is expected. We have established model stationariness (with the ADF test in section 2.4), so these values should look practically the same through all time-steps. Furthermore, since every time-step is considered a new sample by Stan, we would have a total sample size of 3139 (at the time of this report), which corresponds to 3139 likelihoods. These likelihoods, because they point towards the very similar values, repeatedly narrow down the width of our posteriors, until they have very extremely standard deviations.

All parameter values fall within the expected range. It is important to note that these parameters are relative to the chosen unit for t, as we made 1 unit = 4 years (see section 3.2 for more information). α is around 314, which is where the offset of the original data line is (Figure 1). β and γ are relatively small, since the trend line is not very steep. A fits well with the shape of the seasonal variations (the peaks and valleys are ~6 units apart). ϕ matches with the start of the data line (it is shifted a little

towards the right compared to a normal cosine function). Finally, σ looks decent for the amount of noise we see in the data.

4. PREDICTIONS

4.1. ASSUMPTIONS

We use the same assumptions for future data predictions as the inference model (quadratic trend and cosine seasonal variations, homoscedasticity, and normality). However, we will remove the last assumption: stationariness.

For data prediction, it is easy and computationally feasible to generate new parameter values at each new time-step. This was not feasible with Stan inference.

The removal of this assumption makes for a much more interesting model: we can account for future changes in CO_2 trends. These may come from future policies that either curb or exacerbate the yearly increase in CO_2 emissions, or a general increase/decrease in awareness about climate change (that may not necessarily come from any policy changes).

The result is a much more complex time-based model (shown below), where the parameter values at each time-step depend on those at the time-step before.

Unfortunately, since we don't have much information about the future (the Paris agreement was a great milestone towards lowering CO_2 emissions, but the US has since withdrawn from it, and UN climate change talks have been sadly unfruitful). To account for such uncertainty, we will need to introduce yet another assumption: normality and homoscedasticity in the behavior of parameter values. This means that each parameter value follows a Normal distribution whose mean is the value at the time-step before it, with a fixed standard deviation:

$$p_t \sim Normal(p_{t-1}, \sigma_p)$$

The normality is a reflection of our uncertainty and ambivalence: it is symmetrical, so there is equal probability of the next parameter value being either higher or lower than the current one.

4.2. GRAPHICAL MODEL

The full model that we'll use for predictions is as follows:

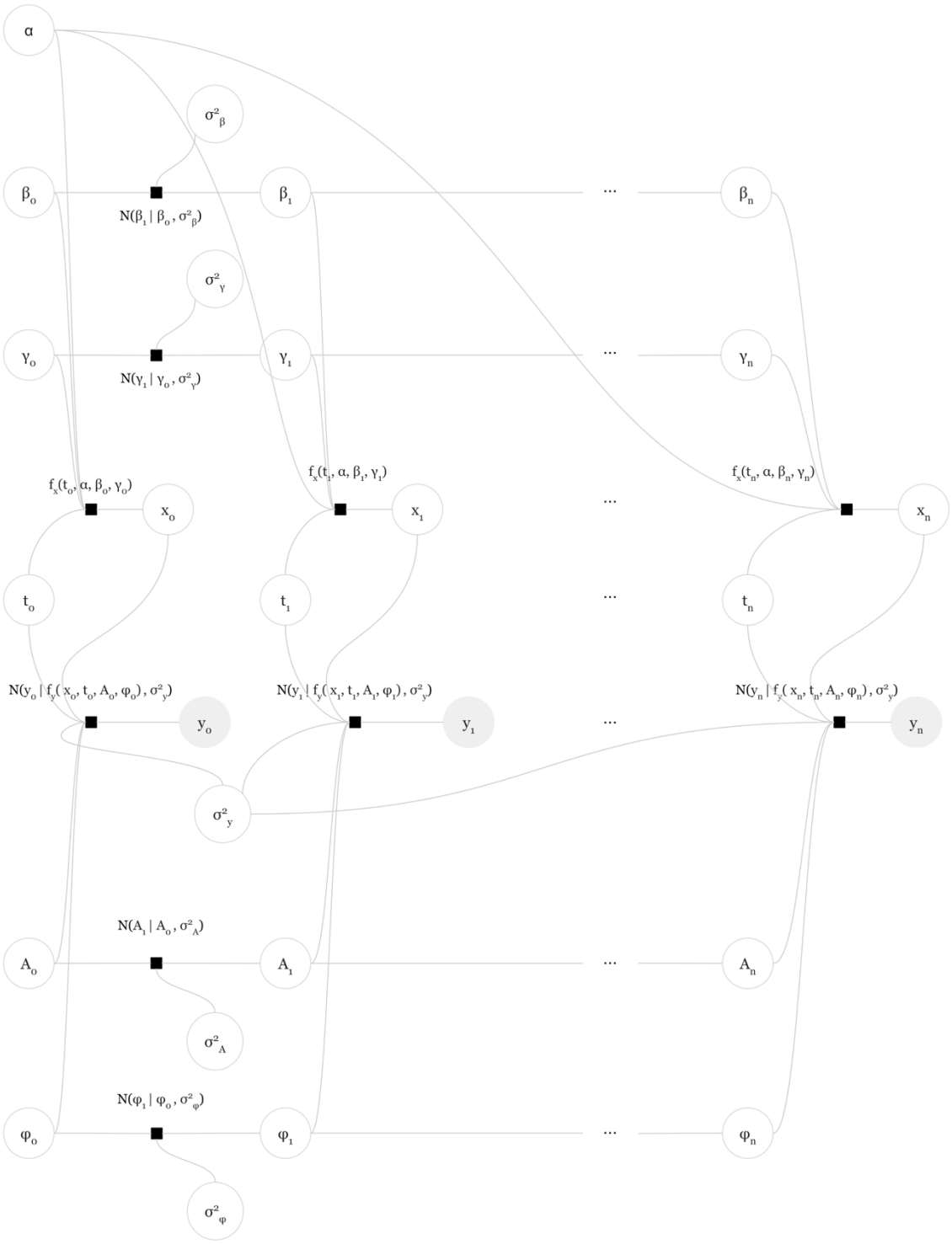


Figure 5. Factor graph for prediction model

Note:

$$f_x(t, \alpha, \beta, \gamma) = \alpha + \beta t + \gamma t^2$$

$$f_y(x, t, A, \phi) = x + A \cdot cos(2\pi t + \phi)$$

This is a rather complex graph. To simplify things a bit, we can group parameters together into vector groups: $v_{at} = (\alpha_t, \beta_t, \gamma_t)$, and $v_{bt} = (A_t, \phi_t)$. This leads to the following graph:

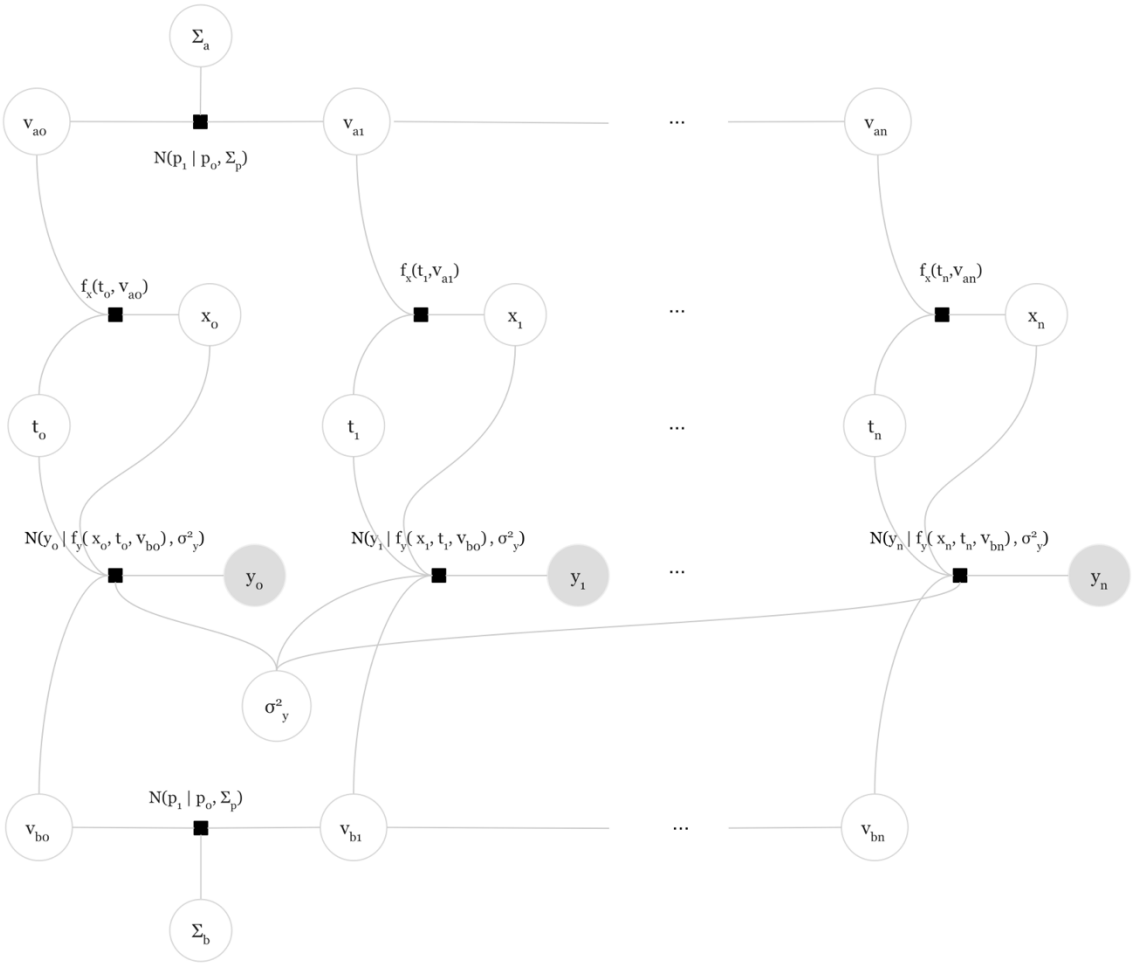


Figure 6. Simplified factor graph for prediction model

Moving from the left side to the right side corresponds with moving through time from the present (with y_0) to the future (with y_n).

Every parameter value (α_1, α_2,...) follows a Normal distribution whose mean is that parameter's previous value, and whose standard deviation is a fixed constant (e.g. σ_α). We can interpret this as the evolution of parameter values: at each time step, the parameter takes on a new value that is very close to its previous value.

Then, we calculate the underlying trend value - x at time t using $x_t = f_x(t, \alpha_t, \beta_t, \gamma_t) = \alpha_t + \beta_t t + \gamma_t t^2$. Note that we have not added the seasonal variation just yet.

After that, we can get the CO_2 level with $y_t = f_x(x_t, t, A_t, \phi_t) = x_t + A \cdot \cos(2\pi t + \phi_t)$. This would serve as the mean value in $level \sim Normal(f_x(x_t, t, A_t, \phi_t), \sigma_y)$. $level$ is our main prediction value.

Note that only the parameter values are functions of their previous values. x_t and y_t are not dependent on x_{t-1} and y_{t-1}. If they were, then we would essentially have a random walk situation, whose results are not very useful in this case.

4.3. RESULTS

We can now generate future predictions with this model. The code for this is in the Jupyter notebook, section 4.

We can generate a plot showing the predictions and 95% confidence intervals:

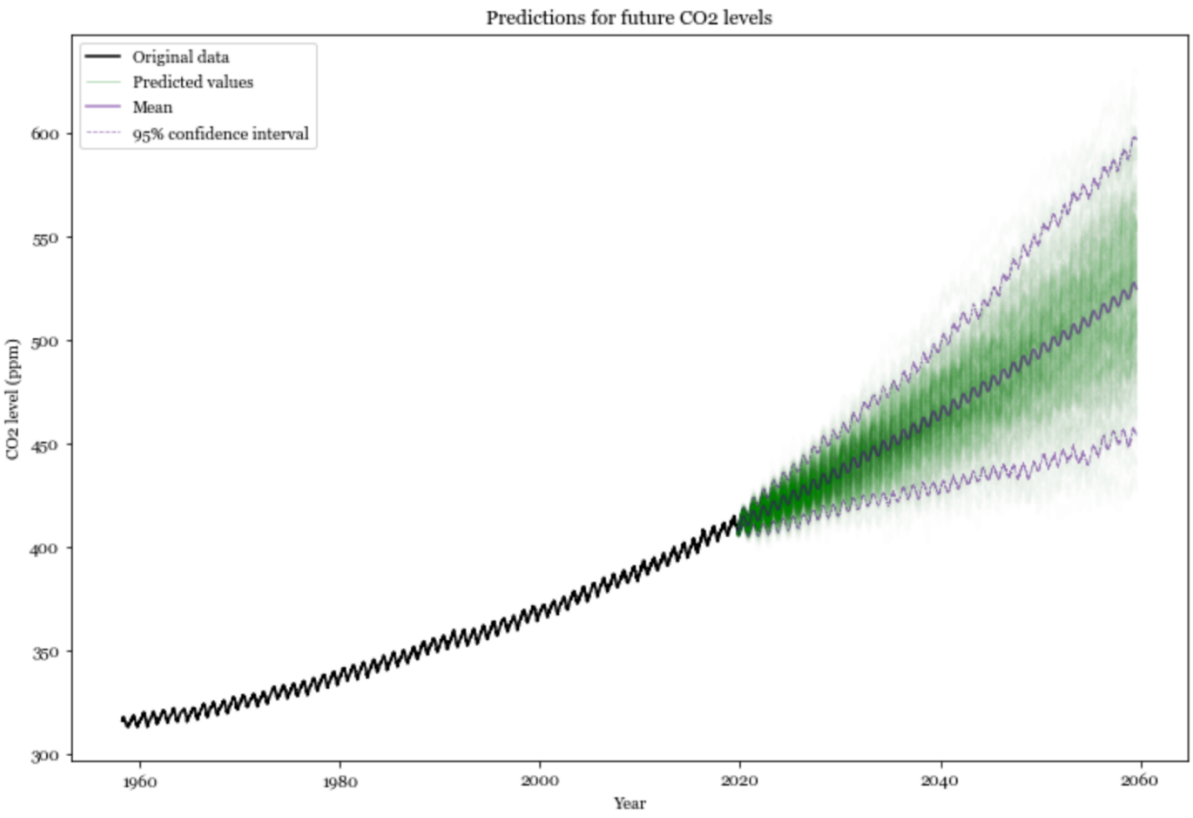


Figure 7. Predicted CO_2 levels until 2060

The CO_2 levels will likely keep increasing with a quadratic manner. By 2040, there is a decent chance that we will have passed the 450 ppm mark. By 2058, there is also a good chance that we'll have passed 500 ppm.

However, the further we go into the future, the more uncertainty there is (see how the 95% confidence intervals widen from left to right). By 2058, the 95% confidence interval is about 150 ppm wide.

We can plot out the probability of reaching 450 ppm through the years:

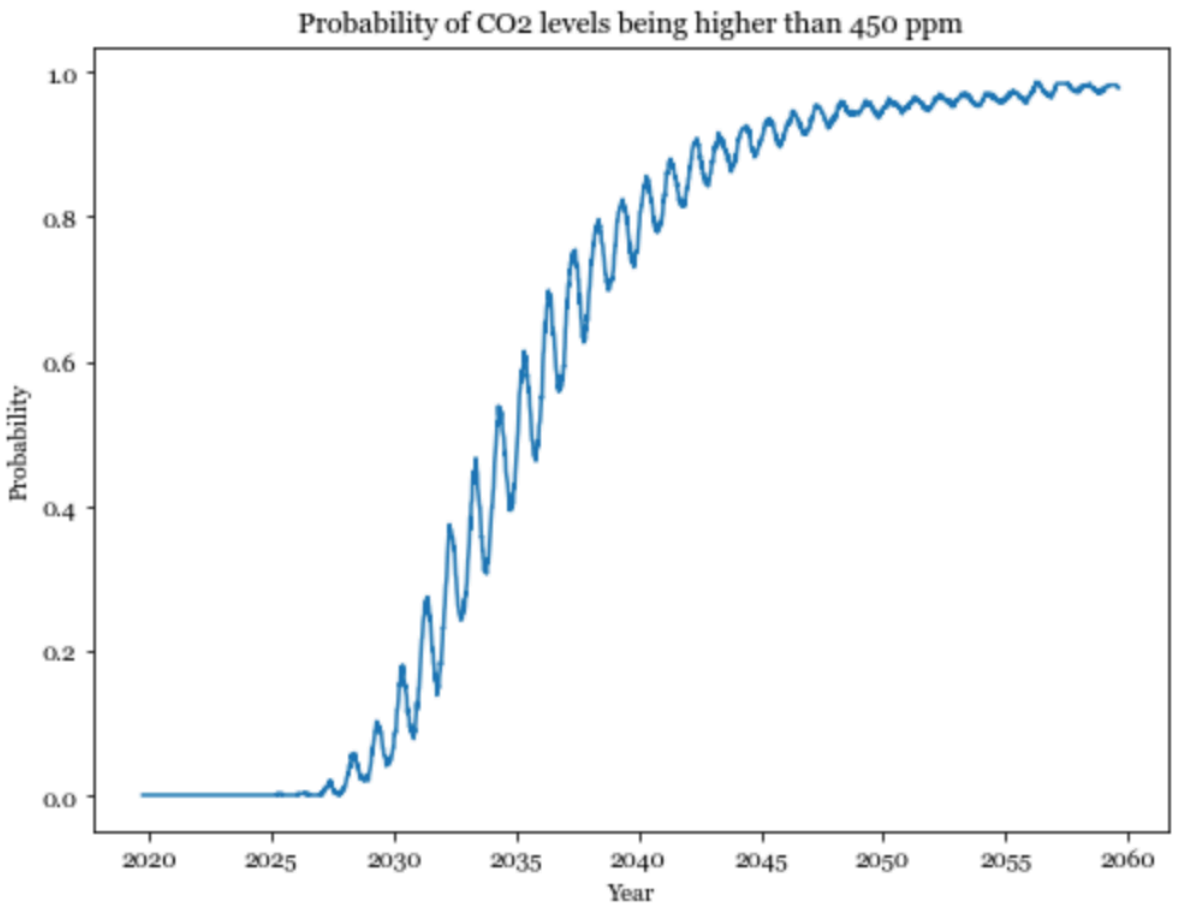


Figure 8. Probability of CO_2 levels reaching 450 ppm

The cosine movement is due to seasonal fluctuations in CO_2 levels.

Inspecting the generated probability array, we can find the first time that there is a high (defined as larger than 90%) chance that the CO_2 levels will be higher than 450 ppm:

```
There is a 90% chance that the CO2 levels will reach 450 ppm in 2042
```

Intuitive explanation of the graph and results:

> *Since the CO_2 level is predicted to increase every year, we can expect the chances of it reaching 450 ppm and above also increasing every year. By 2031 there is already a 20% chance that it will have reached that number. By 2042, there is a 90% chance that this will have already happened!*

Since 450 ppm is considered high risk for climate change, and since there is a high chance that we'll have reached that number within 20 years' time, new policy needs to be made to curb the rapid increase in CO_2 levels. The fast pace at which the CO_2 levels are increasing warrant urgent action from both policy makers and normal consumers.

5. POSSIBLE IMPROVEMENTS

5.1. PREDICTION BIAS

I used a rather ambivalent Normal distribution to model the evolution of parameter values. This ambivalence comes from the fact that Normal distributions have the same means and modes, and here we set them to be 0, so there is an equal chance that the new parameter values will be either higher or lower than the previous ones. However, we could still collect more information about our variable (e.g. expert opinion on future policy changes) to bias our model towards a more realistic future. For example, if we find that CO_2 levels are likely to drop thanks to the success of the Paris agreement (despite the US having withdrawn from it), then we could use a skew-normal distribution with mode 0 but one that is skewed towards the negative direction (so the distribution parameter α should be negative), so that our predictions are more likely to show deceleration in the yearly increase of CO_2 level.

5.2. "RUGGEDNESS"

The generated data, when plotted in a scatter plot, looks similar to the original data:

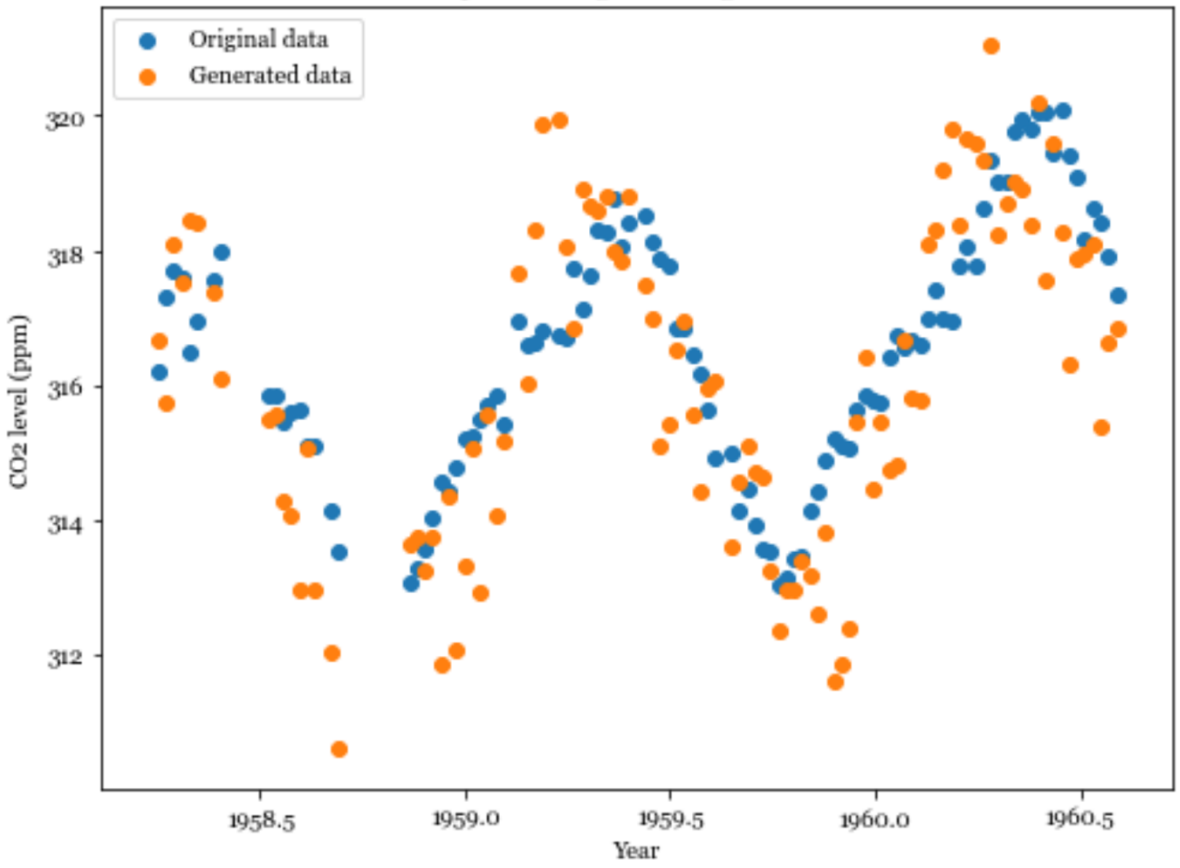


Figure 9. Scatter plot of original and generated data

However, when we plot the data as connected lines, we can see a clear difference:

Figure 10. Line plot of original and generated data

The generated data has a more rugged, "zig-zaggy" pattern than the original data.

We can devise a test statistic to see if this is a significant difference. I call it the "ruggedness" test. We start with three values: y_t, y_{t+1}, and y_{t+2}. We calculate the differences $d_1 = y_{t+1} - y_t$ and $d_2 = y_{t+2} - y_{t+1}$. If d_1 is positive, then the CO_2 levels increased from y_t to y_{t+1}. If it is negative, then there was a decrease. The same goes for d_2. We then compare the signs of d_1 and d_2. If both have the same signs, then there was an increase following an increase, or a decrease following a decrease. What we're interested in is when they have different signs (which means increase → decrease, or decrease → increase), since this is what gives the data its "ruggedness". We keep doing this for y_{t+3}, y_{t+4}, ... The final test statistic is just the number of times there were

consecutive different signs for d_i and d_{i+1} (i.e. increase → decrease, or decrease → increase).

This test statistic is implemented in the Jupyter notebook, section 5.2. Generating 4000 data sets from our 4000 posterior samples, and applying the test to those data sets gives us the following result:

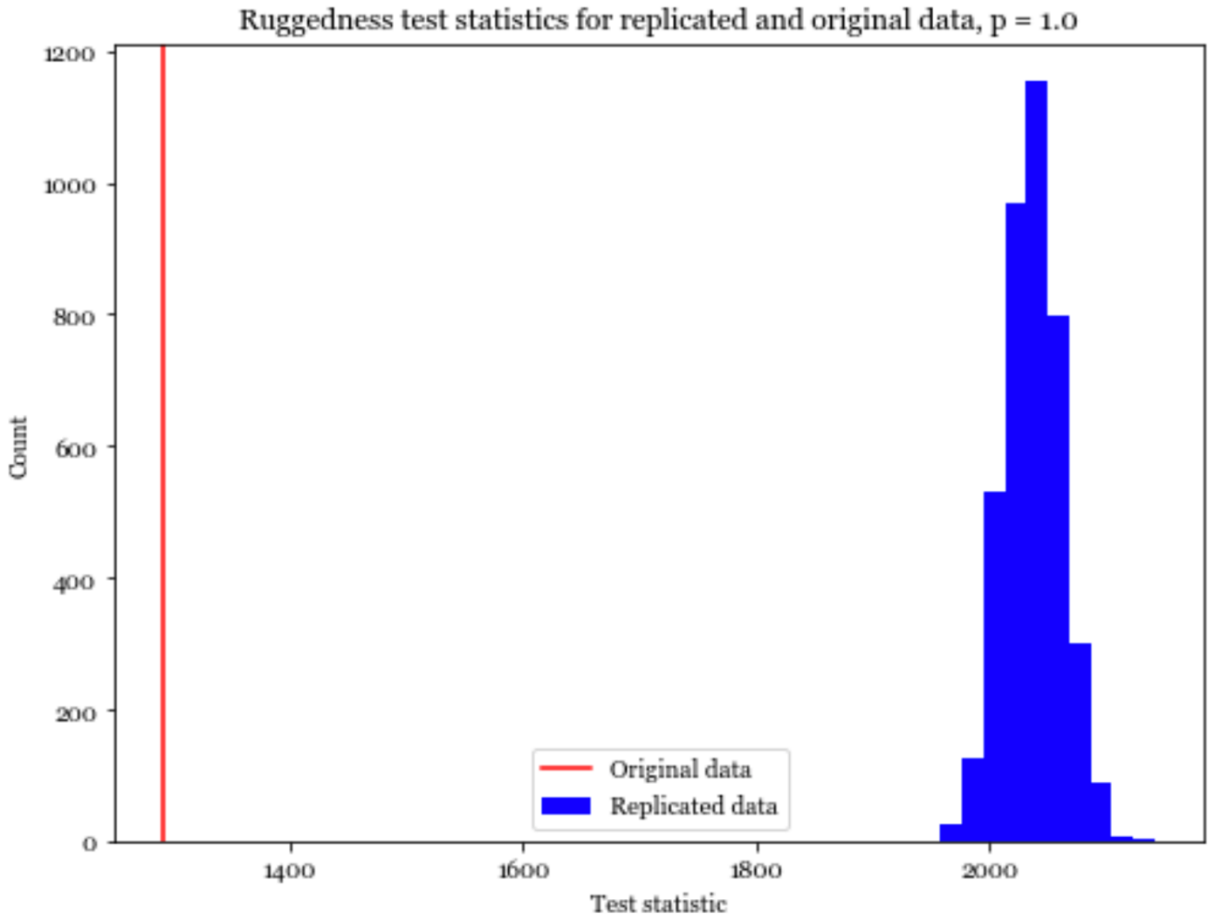


Figure 11. "Ruggedness" test statistics for replicated and original data

The p-value is 1. All generated data sets have a higher test statistic than the original data set. In other words, they are all more "rugged" than the original data set.

This is a statistically significant result. There is something about the model that we haven't accounted for. There may be a violation of the normality assumption: perhaps

y_t doesn't follow a Normal distribution with the given parameters. Or, it could be the case that y_{t+2} is in fact dependent on y_{t+1} and y_t, and so on.

However, this is not a fatal problem for the model. What we have may not be a good reflection of for the noisy behavior of the original data, but its fit for the general trend (quadratic) and seasonal variations (cosine) is still very good. As for the noise, we still see the same standard deviation and variance in the generated data sets as in the original data set. It is only the time-based behavior (the zig-zaggy behavior) that is different.

We might be able to fix this by keeping a history of the signs of the noise component ($Normal(0, \sigma)$). Every time we generate a new noise component, we'll make it more likely for that component to have the same sign as the previous one. For example, if the previous component was -1, then we'll use a distribution that is biased towards the negative space. If it was +1, then we'll use one that is biased towards the positive space. This fix does further complicate the model. Since we only want to predict the future trends and seasonal variations, it is rather unnecessary.