A 02417 Times Series Analysis - Assignment 1

Authors:

- Jeppe Klitgaard <s250250@dtu.dk>
- Yunis Wirkus <s250700@dtu.dk>

Date: 2025-03-02

A.1 Plot Data

We are given a tabular data set "`BIL54`" describing the total number of registered motor driven vehicles in Denmark. The data given as a time series with monthly observations starting in January 2018.

We divide the data set into a *training set* containing data from January 2018 to December 2023, and a *test set* containing data from January 2024 to December 2024.

A.1.1 Construct time variable

The data set is indexed by an `ISO8601` date-time column, which we transform into a floating point time variable, x as follows:

$$x = \text{Year} + \frac{\text{Month}}{12} \qquad \begin{aligned} \text{Month} &\in \{0, 1, \ldots, 11\} \\ \text{Year} &\in \{2018, 2019, \ldots, 2024\} \end{aligned} \tag{1}$$

When referring to a vector of time values, we will follow the vector notation $\boldsymbol{x}$. Other vectors are denoted similarly, while matrices are denoted with a double underline, $\mathbf{X}$.

It is important to distinguish between the time-like vector-valued variable $\boldsymbol{x}$ and the design matrix $\mathbf{X}$, which will be introduced later.

A.1.2 Plotting observations

Using only the training data, we plot the total number of registered vehicles in Denmark as a function of time in Figure 1.

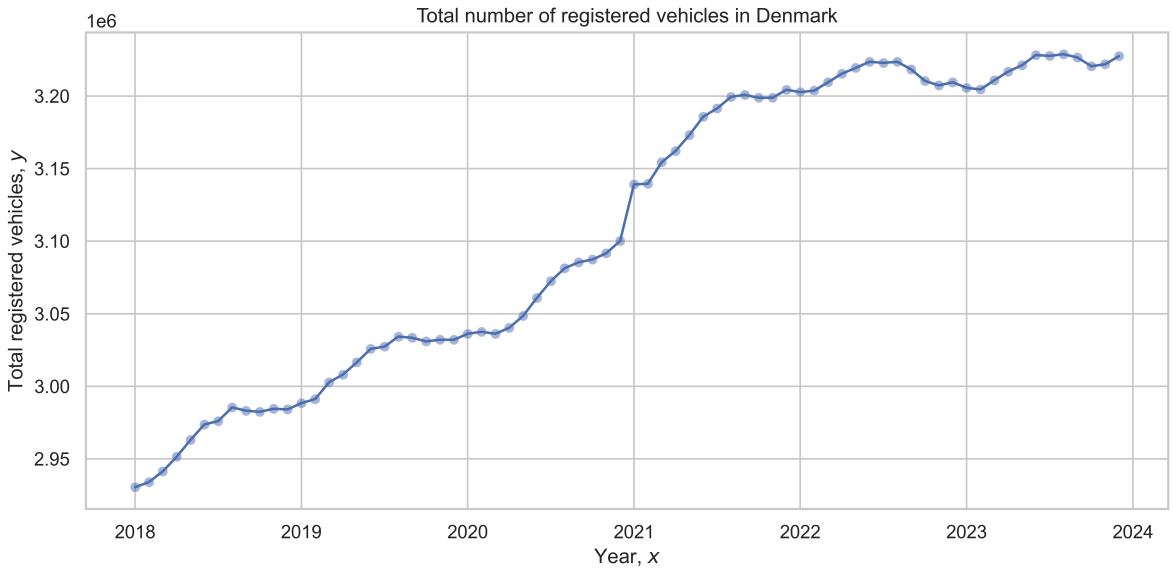

Figure 1: Training data describing the total number of registered vehicles in Denmark as a function of time.

Apart from a significant jump between 2021 and 2022, the data follows roughly a linear trendline, especially between 2018 and the beginning of 2021. After the mentioned jump, the linear trend continues for approximately 6 months into 2021. Then the dataset shows signs of stagnating growth for the total number of cars. From end 2021 to 2024, the data roughly follows linearly trend again, though notably with a smaller slope than previously. This can be interpreted as a saturation in car registration across Denmark.

We pose that reasons for this could be:

1. Most of the population already has a car and no further need to register a new car
2. Due to some legislation or other factor it has become less popular of viable to register a new car, which causes weaker demand for new car registrations

A.2 Linear Trend Model

We are given the General Linear Model (GLM):

$$Y_t = \theta_i + \theta_2 \cdot x_t + \epsilon_t \tag{2}$$

A.2.1 Matrix Form

We immediately rewrite Eq. 2 as a matrix, observing the tensor notation outlined previously:

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon} \tag{3}$$

Where $\mathbf{X}$ denotes the *design matrix*, $\boldsymbol{\theta}$ the *parameter vector*, and $\boldsymbol{\epsilon}$ represents a stochastic noise term, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2)$.

In our case, the *design matrix* is constructed with an intercept θ_1 and a single trend parameter θ_2:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \tag{4}$$

Using the first 3 data points of the given data, we can construct the following model:

$$\begin{aligned} \boldsymbol{y} &= \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon} \\ \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} &= \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} \\ \begin{bmatrix} 2930483 \\ 2934044 \\ 2941422 \end{bmatrix} &= \begin{bmatrix} 1 & 2018.000 \\ 1 & 2018.083 \\ 1 & 2018.167 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} \end{aligned} \tag{5}$$

A.2.2 Parameter Estimation

We can estimate the parameters $\boldsymbol{\theta}$ leveraging the *normal equations*.

This is done by minimizing the *residual sum of squares* (RSS) between the observations and the model predictions, that is:

$$\hat{\boldsymbol{\theta}} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\boldsymbol{\theta}\|^2 \tag{6}$$

We state without proof that the solution to the above optimization problem is given by the *normal equations*:

$$(6) \quad \Rightarrow \quad \mathbf{X}^T\boldsymbol{y} = \mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\theta}} \quad \Leftrightarrow \quad \hat{\boldsymbol{\theta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\boldsymbol{y} \tag{7}$$

Where $\mathbf{X}$ is assumed to be invertible.

We additionally consider the standard errors of the parameter estimates, $\hat{\boldsymbol{\theta}}$.

Under the assumption that the observations are described by Eq. 2, that is, the data is drawn from a *Simple Linear Model* overlaid with a stochastic (i.i.d) noise term, we find that the residuals of the mdoel are exactly the noise term, $\boldsymbol{\epsilon}$.

$$\epsilon = \boldsymbol{y} - \hat{\boldsymbol{y}} \tag{8}$$

Where

$$\hat{\boldsymbol{y}} = \mathbf{X}\hat{\theta} \tag{9}$$

For the benefit of the reader, we reproduce the relationship between the residuals and the covariance matrix, $\boldsymbol{\Sigma}$ [1, p. 36-37]:

From Eq. 7, we have:

$$\begin{aligned}
\mathbb{E}[\hat{\theta}] &= \mathbb{E}\Big[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}\Big] \\
&= \mathbb{E}\Big[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\theta + \epsilon)\Big] && (3) \\
&= \mathbb{E}\Big[(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{X})\theta\Big] && \epsilon \sim \mathcal{N}(0, \sigma^2) \\
&= \theta
\end{aligned} \tag{10}$$

We consider the following:

$$\begin{aligned}
\hat{\theta} - \mathbb{E}[\hat{\theta}] &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y} - \theta \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\theta + \epsilon) - \theta \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon
\end{aligned} \tag{11}$$

We can now evaluate the covariance of the predicted parameters, $\hat{\theta}$:

$$\begin{aligned}
\mathrm{Cov}[\hat{\theta}] &= \mathbb{E}\Big[(\hat{\theta} - \mathbb{E}[\hat{\theta}])(\hat{\theta} - \mathbb{E}[\hat{\theta}])^T\Big] \\
&= \mathbb{E}\Big[\Big((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon\Big)\Big((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon\Big)^T\Big] \\
&= \mathbb{E}\Big[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon\epsilon^T\mathbf{X}\Big((\mathbf{X}^T\mathbf{X})^{-1}\Big)^T\Big] \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbb{E}[\epsilon\epsilon^T]\mathbf{X}\Big((\mathbf{X}^T\mathbf{X})^T\Big)^{-1} \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\,\mathrm{Var}[\epsilon\epsilon^T]\mathbf{X}\Big((\mathbf{X}^T\mathbf{X})^T\Big)^{-1} \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\sigma^2\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} && \epsilon \sim \mathcal{N}(0, \sigma^2) \\
&= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}
\end{aligned} \tag{12}$$

Where we understand the variances of the predicted variables to be the diagonal elements of the covariance matrix:

$$\mathrm{Var}[\hat{\theta}] = \mathrm{diag}(\mathrm{Cov}[\hat{\theta}]) \tag{13}$$

Which gives the following standard errors for the parameter estimates:

$$\sigma_{\hat{\theta}_i} = \sqrt{\mathrm{Var}[\hat{\theta}_i]} = \sqrt{\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}_i} \qquad i \in \{1, 2\} \tag{14}$$

Where σ is calculated using the *residual sum of squares* (RSS) with N and p denoting the number of rows and parameters in the design matrix $\mathbf{X}$ respectively:

$$\sigma = \frac{\left\| \boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{\theta}} \right\|^2}{\sqrt{N-p}} \tag{15}$$

Computing these for the entire training dataset gives:

$$\begin{aligned} \hat{\theta}_1 &= (-110 \pm 4) \times 10^6 \\ \hat{\theta}_2 &= (56.1 \pm 1.8) \times 10^3 \end{aligned} \tag{16}$$

Using these predicted parameters, we are able to produce an estimate of the vehicle registrations for the training dataset using the General Linear Model as seen in Figure 2.

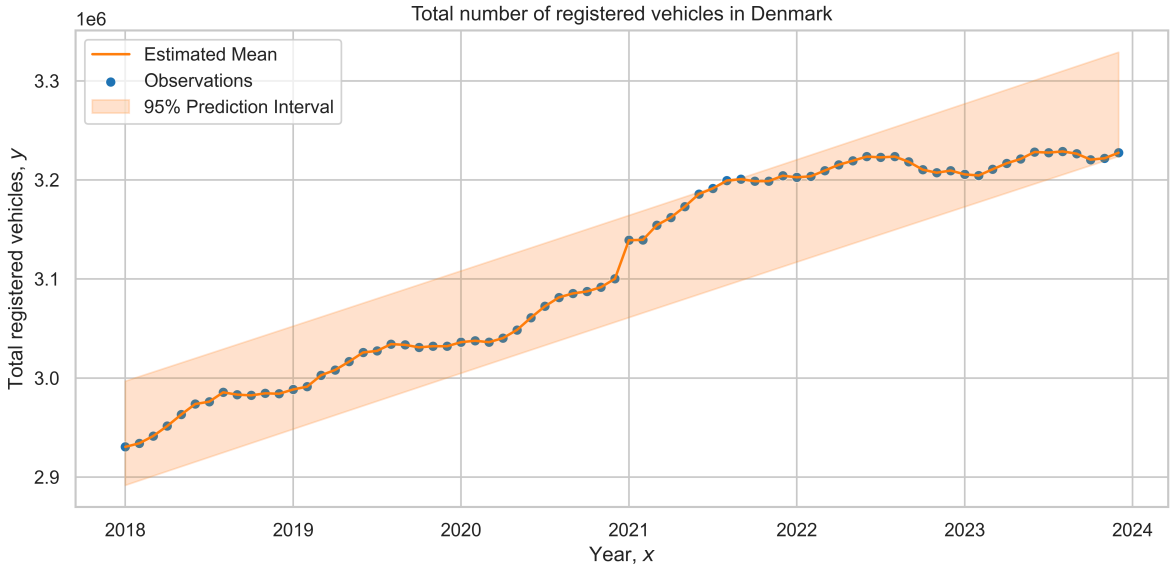

Figure 2: Estimation of vehicle registrations using a Linear Trend Model.

A.2.3 Prediction

We now wish to predict future vehicle registrations using our simple model. From Eq. 9, we see that doing so simply requires the construction of an appropriate design matrix, $\hat{\mathbf{X}}$. This may be constructed simply as:

$$\hat{\mathbf{X}}_i = \begin{bmatrix} 1 & 2024 + \frac{i-1}{12} \end{bmatrix} \qquad i \in \{1, 2, ..., 12\} \tag{17}$$

Where i indexes the rows of the design matrix.

Carrying out the forecasting as described by Eq. 9 along with appropriate estimation of the confidence interval of our prediction, we obtain Table 1. It should be noted that the estimation of the confidence interval is valid only in the case where the observations are described by the General Linear Model.

Time	Total Vehicles Registered	95% Confidence Interval, lower bound	95% Confidence Interval, upper bound
2024.000	3223801	3333802	3228505
2024.083	3223805	3338540	3233124
2024.167	3231177	3343279	3237742
2024.250	3237629	3348020	3242359
2024.333	3244611	3352763	3246974
2024.417	3255073	3357507	3251587
2024.500	3254569	3362252	3256199
2024.583	3258060	3366999	3260809
2024.667	3256515	3371748	3265418
2024.750	3252275	3376498	3270025
2024.833	3253059	3381249	3274631
2024.917	3258025	3386002	3279236

Table 1: 12 month forecast of vehicle registrations in Denmark.

A.2.4 Plot of Forecast

We can now present the forecasted data in Table 1 along with the training, test, and prediction data sets

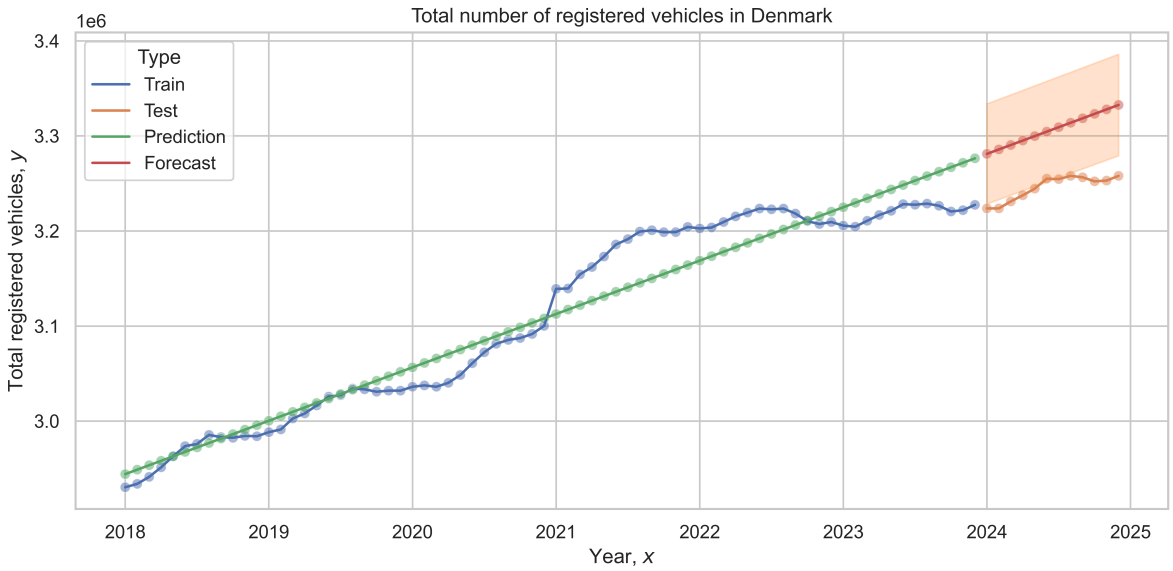

Figure 3: 12 month forecast of vehicle registrations in Denmark.

A.2.5 Commentary on Forecast

We find that the prediction is a relatively poor match against the test data, from which we understand that a Simple Linear Model is likely not an appropriate model for the data.

In particular, we observe significant local deviations from the model, which can be understood by considering the modelling domain. It is reasonable to assume that the number of registered vehicles will be influenced by market conditions, such as government subsidies, registration fees, and taxation schemes. Additionally, we would expect supply chain disruptions to have strongly influence the contemporary pricing and availability of vehicles.

A better model would likely incorporate these factors. A model that incorporates locality without apriori domain knowledge could be a *Weighted Least Squares* model with local weights.

A.2.6 Residual Analysis

In order to substantiate Section A.2.5, we perform a residual analysis on the prediction and forecasting data.

Recalling the assumptions of our model, we expect the residuals to be normally distributed around zero:

$$\boldsymbol{r} = \boldsymbol{y} - \hat{\boldsymbol{y}} \sim \mathcal{N}(0, \sigma^2) \tag{18}$$

It is readily apparent in Figure 4 that the residuals are not normally distributed, nor do they average to zero:

$$\mathbb{E}[\boldsymbol{r}] = -8739 \tag{19}$$

It should be noted, that the sum of the residuals is relatively close to zero, which is by construction as can be seen in Section A.2.2.

We conclude that the model is not appropriate to describe the data.

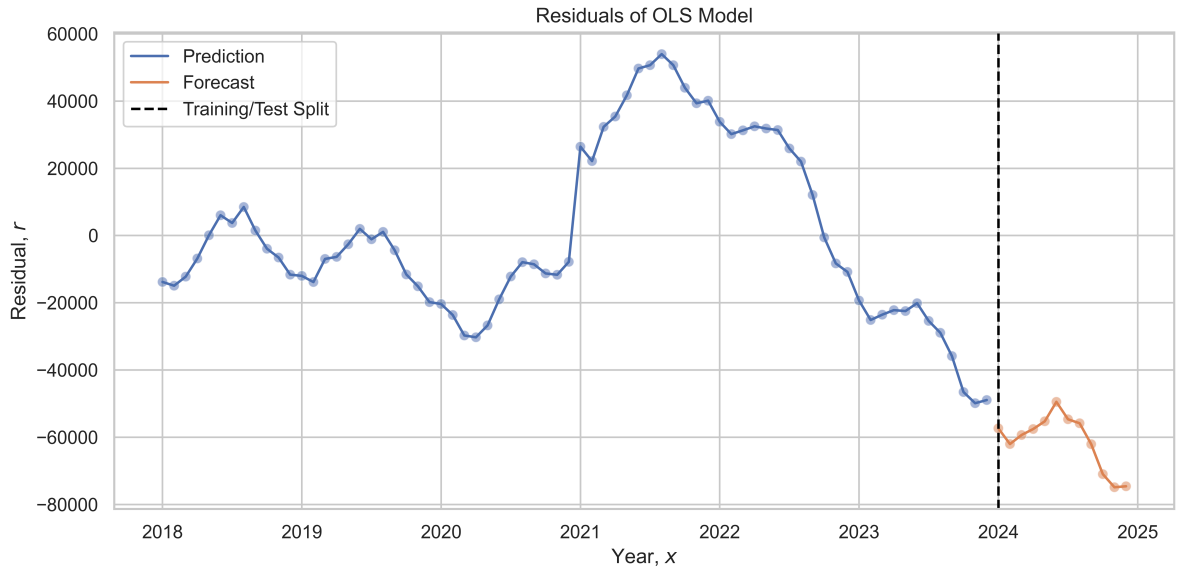

Figure 4: Residual analysis on prediction and forecasting of total vehicle registrations in Denmark. Dashed black line delineates the transition from prediction to forecasting.

A.3 WLS - Local Linear Trend Model

In order to mitigate some of the issues observed in the Linear Trend Model of Section A.2, we propose a *Weighted Least Squares* model with local weights.

That is, more recent observations are weighted higher than older observations.

This is facilitated by repurposing the variance-covariance matrix of the residuals. For Ordinary Least Squares (OLS), we assumed the residuals to be modelled as $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2)$. That is, we have imposed an assumption of *homoscedasticity* on the residuals $\boldsymbol{\epsilon}$. In Weighted Least Squares (WLS) modelling, we relax this assumption and instead consider the variances of the residuals to be *heterogeneous* – that is, the residuals are *heteroscedastic*:

$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\sigma}^2) \tag{20}$$

For Global Weighted Least Squares, one would ideally know the variances of the residuals a priori and simply weigh the residuals by the inverse of the variances.

We instead choose to *exploit* the Weighted Least Squares model to introduce *locality* in the estimation by letting the covariances of the residuals be modelled by $\sigma^2\mathbf{W}$. We recall that the solution to the OLS problem (Eq. 7) was obtained by optimization of the *residual sum of squares* (RSS) between the observations and the model predictions. If we choose to weigh the residuals by their recency, we able to obtain an estimator that values recent information more highly than older information.

A concrete example of a $\mathbf{W}$ that would induce locality would be one given by

$$\mathbf{W} = \begin{bmatrix} \lambda_N & 0 & 0 & \cdots & 0 \\ 0 & \lambda_{N-1} & 0 & \cdots & 0 \\ 0 & 0 & \lambda_{N-2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_0 \end{bmatrix} \tag{21}$$

If we set $\lambda_N, \lambda_{N-1} = 0$, we can appreciate that the model is now more local in that the two oldest observations are not considered in the estimation. Typically one would choose $\mathbf{W}$ to be a diagnoal matrix of exponentially decaying weights, as we will see in Section A.3.1.

Referencing [1, p. 38-39], we state without proof that the *normal equations* for the WLS model are given by:

$$(\mathbf{X}^T\mathbf{W}\mathbf{X})\hat{\boldsymbol{\theta}} = \mathbf{X}^T\mathbf{W}\boldsymbol{y} \tag{22}$$

Note that we have changed the notation slightly, denoting the weight matrix as $\mathbf{W}$ rather than perverting the sigma as is done in [1]. That is, we have made the substitution $\mathbf{W} = \boldsymbol{\Sigma}^{-1}$.

Assuming invertibility of $\mathbf{X}^T\mathbf{W}\mathbf{X}$, we obtain the parameter estimator:

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}\boldsymbol{y} \tag{23}$$

We again follow the proof given in Section A.2.2, though noting that the variances of the residuals are now given by $\mathbb{E}\left[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T\right] = \sigma^2\mathbf{W}$. As such, Eq. 12 becomes:

$$\mathrm{Cov}\left[\hat{\boldsymbol{\theta}}\right] = \sigma^2\left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \tag{24}$$

A typical choice of the weight matrix would be to have exponentially decaying weights:

$$\mathbf{W} = \sum_{i=1}^{N}\sum_{j=1}^{N}\delta_{ij}\lambda^{N-i} \qquad\qquad \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{25}$$

Where δ_{ij} is *Kroneckers delta*, λ is the *forgetting factor*, and N denotes the dimension of $\boldsymbol{y}$.

To recap, we obtaing *locality* in our model by gradually *forgetting* older observations.

A.3.1 Variance-Covariance and Weight Matrices

Encouraging the reader to grit their teeth we summarise the 'variance-covariance' matrices for the OLS and WLS models as follows:

$$\begin{aligned} \boldsymbol{\Sigma} &= \sigma^2\mathbb{I} && && \text{OLS} \\ \mathbf{W} &= \sigma^2\mathrm{diag}(\boldsymbol{\lambda}) && \boldsymbol{\lambda} = \begin{bmatrix} \lambda^{N-1} & \lambda^{N-2} & \dots & \lambda^0 \end{bmatrix} && \text{WLS} \end{aligned} \tag{26}$$

Stating these more explicitly, they become:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma^2 & & \\ & \ddots & \\ & & \sigma^2 \end{bmatrix} \qquad \text{OLS}$$

$$\mathbf{W} = \begin{bmatrix} \sigma^2\lambda^{N-1} & & & & \\ & \sigma^2\lambda^{N-2} & & & \\ & & \ddots & & \\ & & & \sigma^2\lambda^1 & \\ & & & & \sigma^2 \end{bmatrix} \qquad \text{WLS} \tag{27}$$

For the rest of the analysis, we will consider the *forgetting factor* $\lambda \equiv 0.9$.

A.3.2 Weighting Regimen

Considering 72 time steps, we visualise the decay of the weights in Figure 5

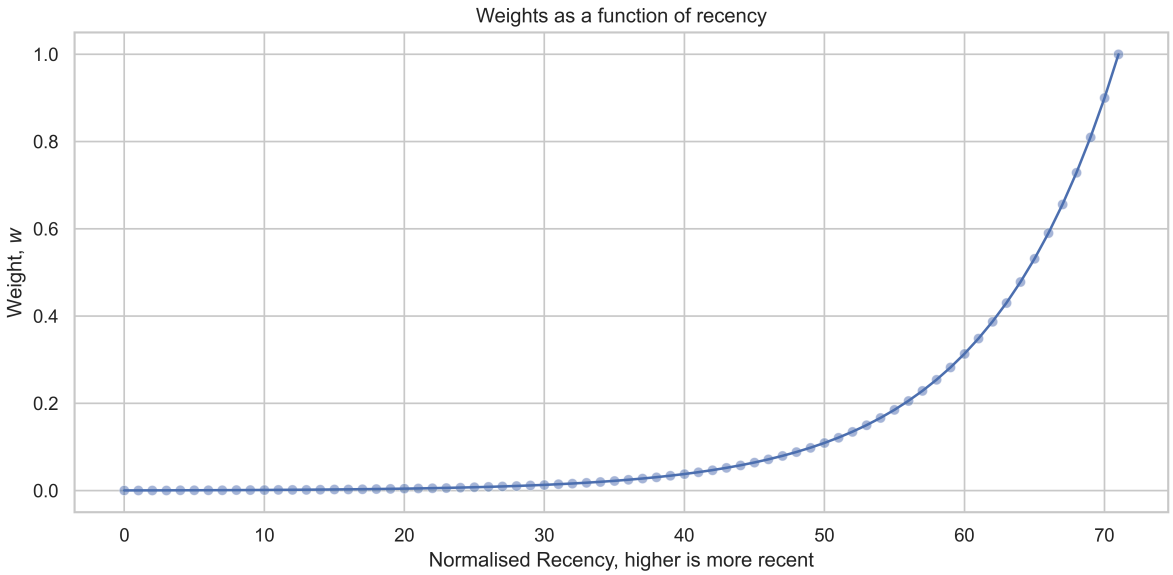

Figure 5: Decay of weights in the Weighted Least Squares model for $\lambda \equiv 0.9$.

To better visualise the weights, we plot the training data where the opacity of the individual observations is given by their weight in the WLS model parameterised by $\lambda = 0.9$:

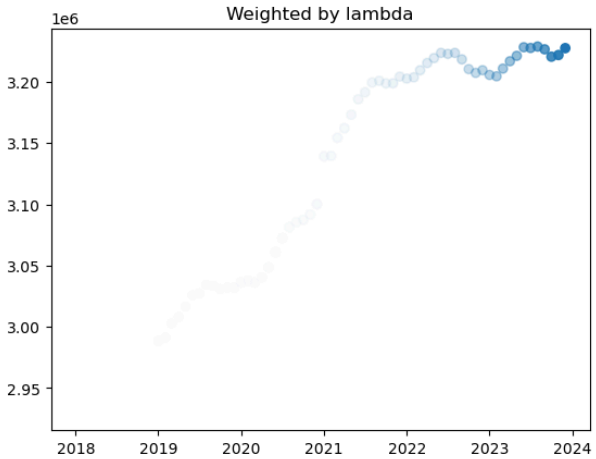

Figure 6: Forgetting in WLS Model visualised using training data set for $\lambda \equiv 0.9$.

A.3.3 Sums of weights

We find that the sum of the weights is given by a geometric sequence:

$$\sum_{i=1}^{N} \lambda^{N-i} = \frac{1-\lambda^N}{1-\lambda} \tag{28}$$

For the limit $N \to \infty$ and $\lambda = 0.9$, we find:

$$\lim_{N\to\infty} \sum_{i=1}^{N} \lambda^{N-i} = 10 \tag{29}$$

We consider $N = 72$ and find the truncation gives:

$$\sum_{i=1}^{72} \lambda^{72-i} \approx 9.995 \qquad \text{WLS} \tag{30}$$

The corresponding sum for the OLS model is clearly not bounded and just becomes:

$$\sum_{i=1}^{N} 1 = N \qquad \text{OLS} \tag{31}$$

For $N = 72$, this simply becomes 72.

A.3.4 Parameter Estimation

We can now estimate the parameters of the WLS model using Eq. 23:

$$\begin{aligned} \hat{\theta}_1 &= (-1159200 \pm 600) \times 10^2 \\ \hat{\theta}_2 &= (5173.5 \pm 2.6) \times 10^2 \end{aligned} \tag{32}$$

Where the standard error is computed using Equation (3.43) in [1, p. 39].

A.3.5 Forecasting

We now perform a 12 month forecast using the WLS model in similar to fashion to the one carried out in Section A.2.3, as seen in Figure 7.

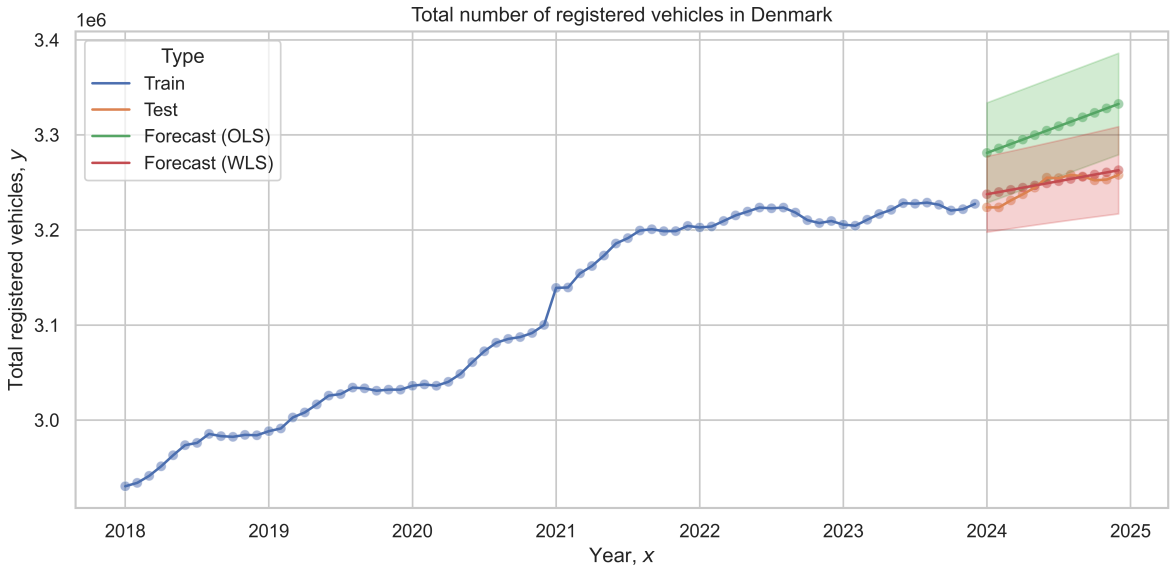

Figure 7: Comparison of the OLS and WLS ($\lambda = 0.9$) forecasts of total number of registered vehicles in Denmark.

It should be noted that the standard error estimate used in the WLS model has been corrected with respect to Eq. 15 for the truncating effect of the weights by calculating the degrees of freedom using the *total memory*, T of the model instead:

$$\sigma = \frac{\left\| \boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{\theta}} \right\|^2}{\sqrt{T - p}} \tag{33}$$

Where the *total memory* is given by the sum of the weights:

$$T = \sum_{i=1}^{N} \lambda^{N-i} = \frac{1 - \lambda^N}{1 - \lambda} \tag{34}$$

For $\lambda < 1$ and $N \in \mathbb{Z}_+$ we observe that $T < N$, which agrees with intuition – a model with less memory *should* be less *confident* in its predictions, all other things being equal.

Even with this correction reducing the model confidence, we find an improved prediction and associated prediction interval as seen in Figure 7. This can be understood as the sum of squared residuals being reduced by the weighting scheme.

While the prediction does exhibit an improved fit to the test data when compared to the prediction carried out with an OLS model, it remains important to note that the model may not generally be expected to forecast well, particularly in the event of drastic changes in legislation or governance of motor driven vehicles.

Given a choice between the two models, short-term forecasts would be better served by the WLS model. It is not possible to confidently state which model would be preferable for long-term forecasting.

A.3.6 Different forgetting factors

Our choice of $\lambda = 0.9$ was somewhat arbitrary and wholly unjustified. In order to gauge which choice of forgetting factor may best fit out test data, we run the WLS analysis repeatedly. It should be noted that the choice of forgetting factor will inevitably be a qualitative matter and there is no

guarantee that the λ that minimises the sum of squared residuals is necessarily a good or sensible choice for prediction for data sets that are not the current test set.

The outcome of the analysis can be seen in Figure 8, where we have omitted the confidence interval estimates for clarity.

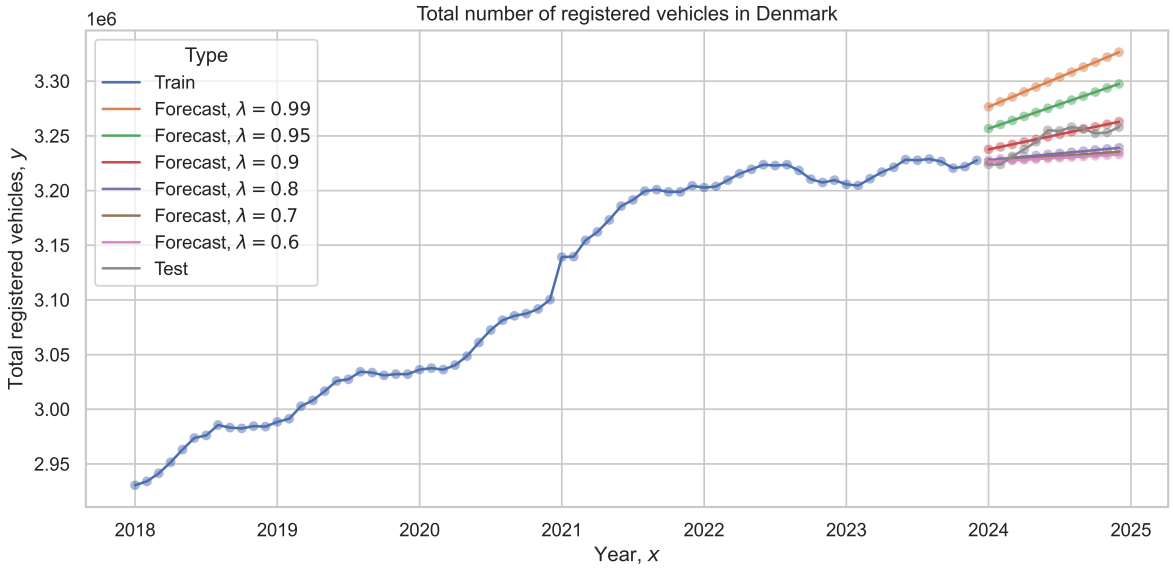

Figure 8: 12 month forecasts of total number of registered vehicles in Denmark using Local Weighted Least Squares with a variety of forgetting factors.

We find that $\lambda = 0.9$ is a reasonable choice for the test data set, though it is not necessarily the best choice for other data sets.

A.4 Recursive Estimation and Optimization of λ

The OLS problem whose parameters may be obtained as given by Eq. 7 can be reformulated as an *iterative* problem:

For a timestep t we find from Eq. 7:

$$\hat{\theta}_t = \left(\mathbf{X}_t^T \mathbf{X}_t\right)^{-1} \mathbf{X}_t^T \boldsymbol{y}_t = \mathbf{R}_t^{-1} \boldsymbol{h}_t \tag{35}$$

Having used the definitions:

$$\begin{aligned} \mathbf{R}_t &\equiv \mathbf{X}_t^T \mathbf{X}_t = \sum_{i=1}^{t} \boldsymbol{x}_i \boldsymbol{x}_i^T \\ \boldsymbol{h}_t &\equiv \mathbf{X}_t^T \boldsymbol{y}_t = \sum_{i=1}^{t} \boldsymbol{x}_i y_i \end{aligned} \tag{36}$$

A.4.1 Update Equations

Inspection of Eq. 36 readily reveals the objects at timestep t as a function of their values at timestep $t-1$:

$$\begin{aligned} \mathbf{R}_t &= \mathbf{R}_{t-1} + \boldsymbol{x}_t \boldsymbol{x}_t^T = \sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i^T + \boldsymbol{x}_t \boldsymbol{x}_t^T \\ \boldsymbol{h}_t &= \boldsymbol{h}_{t-1} + \boldsymbol{x}_t y_t \ = \sum_{i=1}^{t-1} \boldsymbol{x}_i y_i + \boldsymbol{x}_t y_t \end{aligned} \tag{37}$$

We can then rewrite Eq. 35 as:

$$\hat{\theta}_t = \hat{\theta}_{t-1} + \mathbf{R}_t^{-1}\boldsymbol{x}_t\left(y_t - \boldsymbol{x}_t^T\hat{\theta}_{t-1}\right) \tag{38}$$

Somewhat less neatly typeset, we also include the update equations as a pen-and-paper scan to complete the assignment objective.

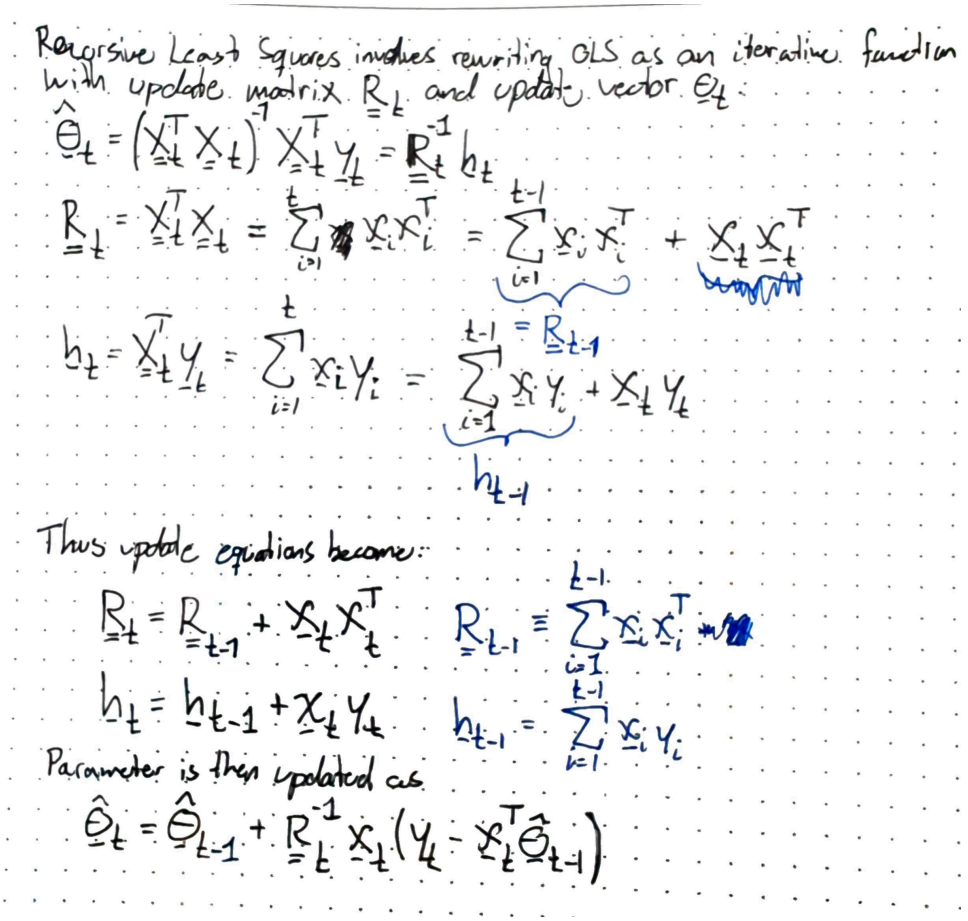

Figure 9: Update equations on pen and paper by Jeppe Klitgaard (s250250).

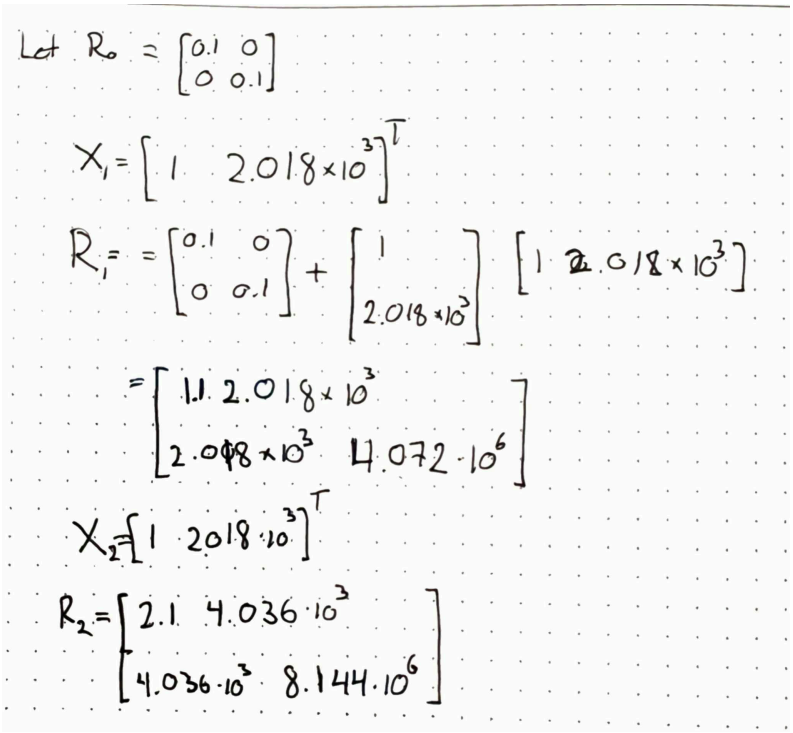

Figure 10: Calculations of R_1 and R_2 on pen and paper by Jeppe Klitgaard (s250250).

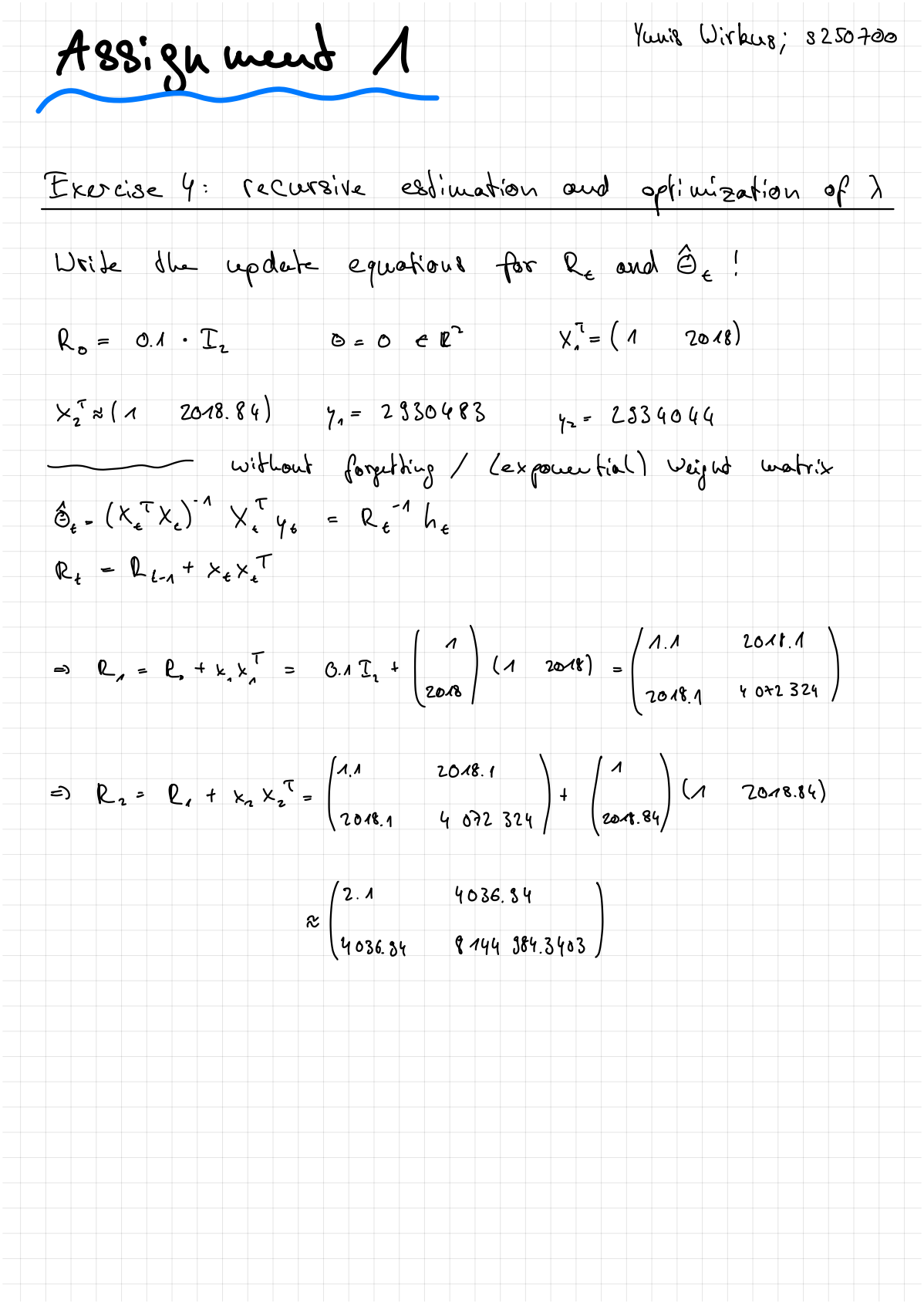

Figure 11: Update equations on pen and paper by Yunis Wirkus (s250700).

A.4.2 Computational Implementation

We implement the Recursive Least Squares as given in Eq. 37 and Eq. 38 in Python as follows:

```python
def iterate(R, theta, x_t, y_t):
    R_t = R + x_t @ x_t.T
    theta_t = theta + (inv(R_t) @ x_t) * (y_t - x_t.T @ theta)

    return R_t, theta_t
```

Using initial conditions $\mathbf{R}_0 = 0.1 \cdot \mathbb{I}$ and $\theta_0 = 0$, we iterate over the data set to $t' = t - 3$ (not inclusive) and obtain the parameter estimates $\hat{\theta}$. These are presented in Figure 12.

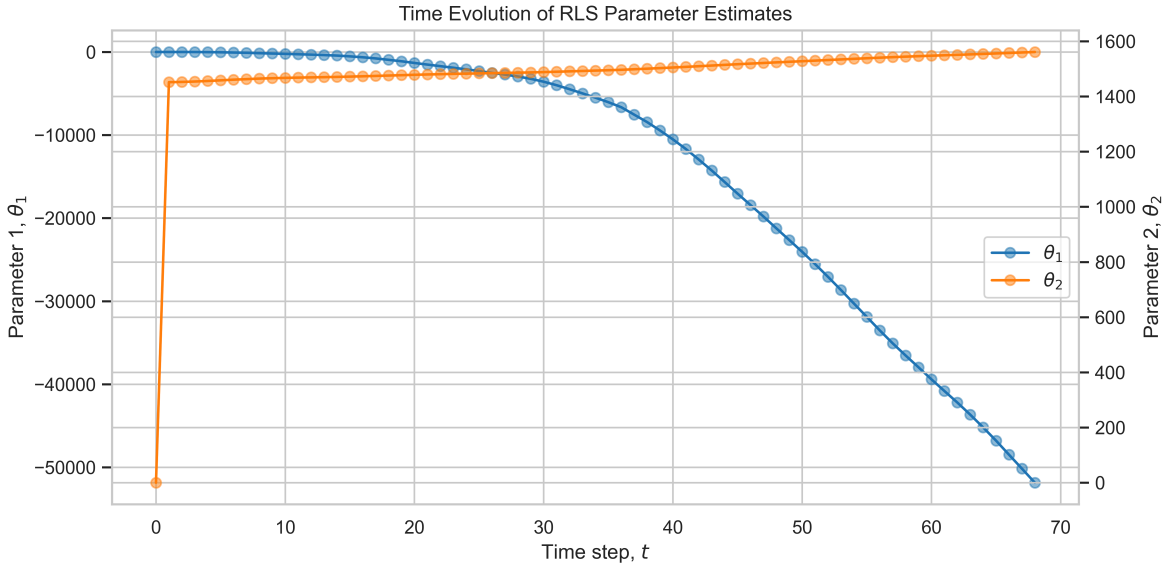

Figure 12: Parameter estimates obtained using Recursive Least Squares.

We find that the initial parameter estimates are very poor, but gradually improve as more data is included in the estimation. Notably, we can see that as the slope, θ_2, increases as part of the iterative correction, the intercept, θ_1, shifts down.

While the matrix operations in Eq. 38 are somewhat opaque, careful inspection of the parameter update equation reveals a residual term $y_t - \boldsymbol{x}_t^T \hat{\theta}_{t-1}$ where $\boldsymbol{x}_t^T \hat{\theta}$ gives a prediction $\hat{y}_t$ using the model found in the previous timestep.

The term $\mathbf{R}_t^{-1} \boldsymbol{x}_t$ then scales the residual appropriately, where we understand $\mathbf{R}$ to take the role of $\mathbf{X}^T\mathbf{X}$ in the OLS model.

Now we will draw a comparison to the regular, non-recursive OLS model. Imagine setting

$$\varepsilon_t = y_t - \boldsymbol{x}_t^T \hat{\theta}_{t-1} \tag{39}$$

Which gives:

$$\hat{\theta}_t = \hat{\theta}_{t-1} + \mathbf{R}_t^{-1} \boldsymbol{x}_t \varepsilon_t \tag{40}$$

It becomes clear upon comparison with the regular OLS model that the term $\boldsymbol{h}_t \equiv \boldsymbol{x}_t \varepsilon_t$ may be interpreted as height of the parameter projection. Given $\mathbf{R}_t = \mathbf{X}_t^T \mathbf{X}_t$ is a projection of the design matrix $\mathbf{X}_t$ into a symmetric, invertible matrix, $\boldsymbol{h}_t$ can be thought of as a scaled residual term.

Then overall the term becomes

$$
\begin{aligned}
&\mathbf{R}_t^{-1}\boldsymbol{x}_t(y_t - \boldsymbol{x}_t^T\hat{\theta}_{t-1}) \\
&= \mathbf{R}_t^{-1}\boldsymbol{x}_t\varepsilon \\
&= \mathbf{R}_t^{-1}\boldsymbol{h}_t \\
&= \hat{\theta}_t^* \qquad\qquad \text{from OLS} \\
\Rightarrow \hat{\theta}_t &= \hat{\theta}_{t-1} + \hat{\theta}_t^*
\end{aligned}
\tag{41}
$$

So we can think of the new $\hat{\theta}_t$ as an updated parameter estimate, based on the previous estimate *and* a correction term arising from the residual of the current observation compared against a prediction of this observation using the previous model.

A.4.3 Calculation and Comparison of Parameter Estimates

We now calculate the estimates $\hat{\theta}_N$. The meaning of $\hat{\theta}_N$ is ambiguous in the task. We interpret it to mean $\hat{\theta}_1$ and $\hat{\theta}_2$ at time $t' = t$.

This gives us:

Model	$\hat{\theta}_1$	$\hat{\theta}_2$
OLS	-110.365428×10^6	56.14×10^3
RLS	-58.32×10^3	1.568×10^3

Table 2: Comparison of parameter estimates obtained using Ordinary Least Squares (OLS) and Recursive Least Squares (RLS).

From Table 2, we see that the RLS model has produced a significantly different parameter estimate than the OLS model. In order to gauge the performance of the two models, we compare the predictions in Figure 13.

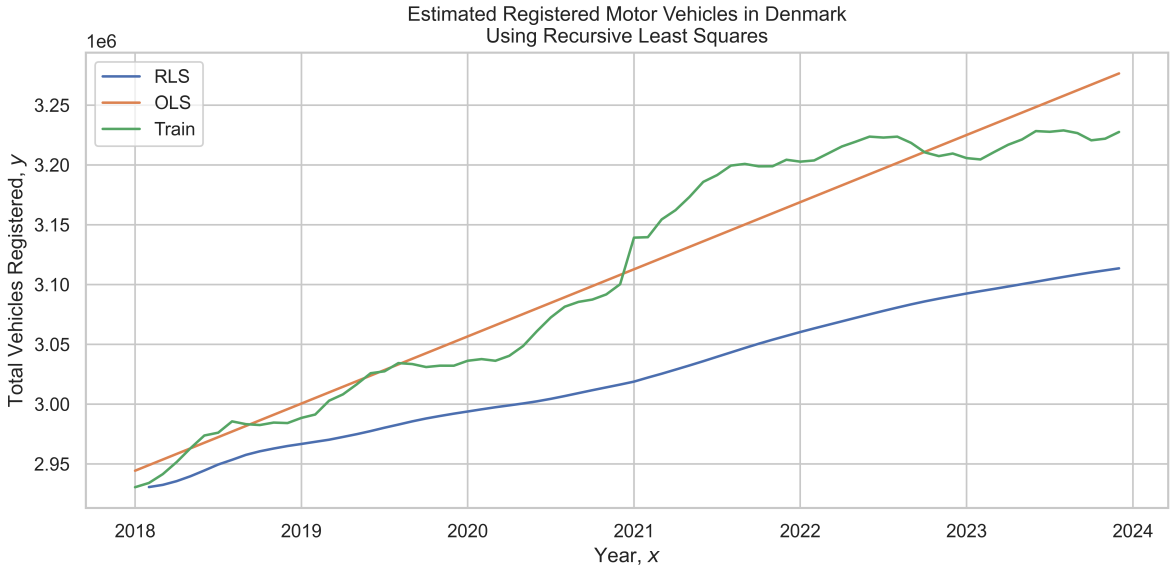

Figure 13: Comparison of the OLS and RLS forecasts of total number of registered vehicles in Denmark.

It is readily apparent that the RLS model is a poor fit to the data, which we attribute to a very poor initial guess of the parameters θ.

From the intuition we gathered in Section A.4.2, we understand that $\mathbf{R}_t^{-1}$ determines the scale of updates to parameters $\hat{\theta}_t$. We can allow for larger updates by reducing the initial information matrix $\mathbf{R}_t$, though notably it cannot be $\mathbf{0}$ as this would not be invertible. Additionally, we can improve the performance of our RLS model by having a better estimate for $\hat{\theta}_0$.

Based on this, we use the OLS model to come up with a reasonable $\hat{\theta}_0$ and set a smaller initial information matrix $\mathbf{R}_0$:

$$\begin{aligned} \mathbf{R}_0 &= \mathbb{I} \cdot 10^{-8} \\ \hat{\theta}_0 &= \begin{bmatrix} -100 \times 10^6 \\ 50 \times 10^3 \end{bmatrix} \end{aligned} \tag{42}$$

This increases the initial 'eagerness' of the model to update the parameters when fed new information, while also giving a decent guess for the initial parameters.

In order to get an intuition for the RLS model, we now reproduce Figure 12 with the updated initial conditions:

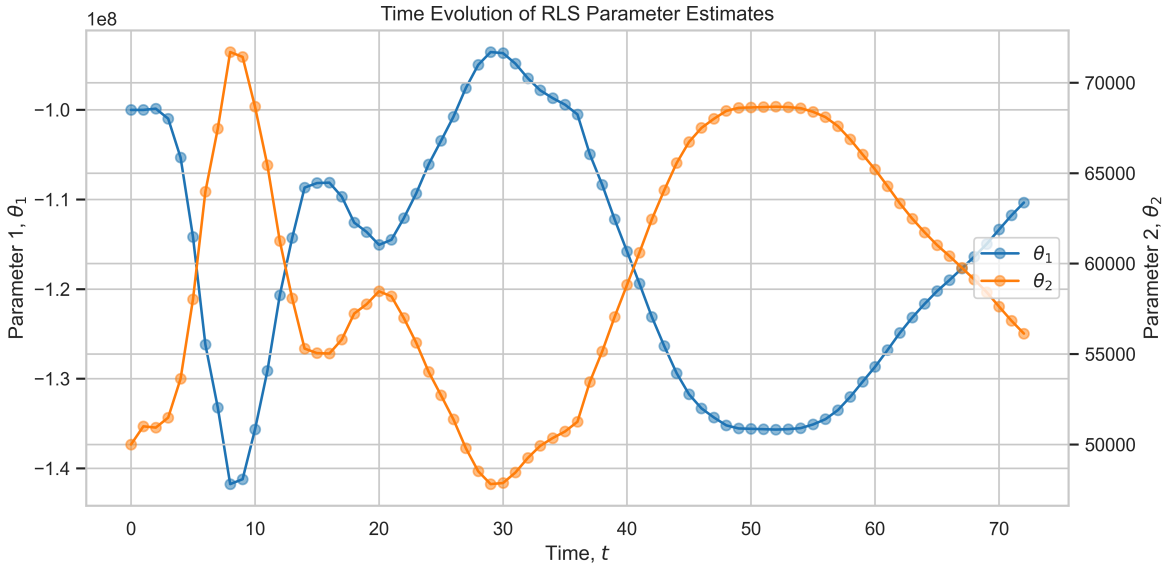

Figure 14: Parameter estimates obtained using Recursive Least Squares with updated initial conditions.

Reproducing Figure 13 with the updated initial conditions, we observe a significantly improved fit to the data:

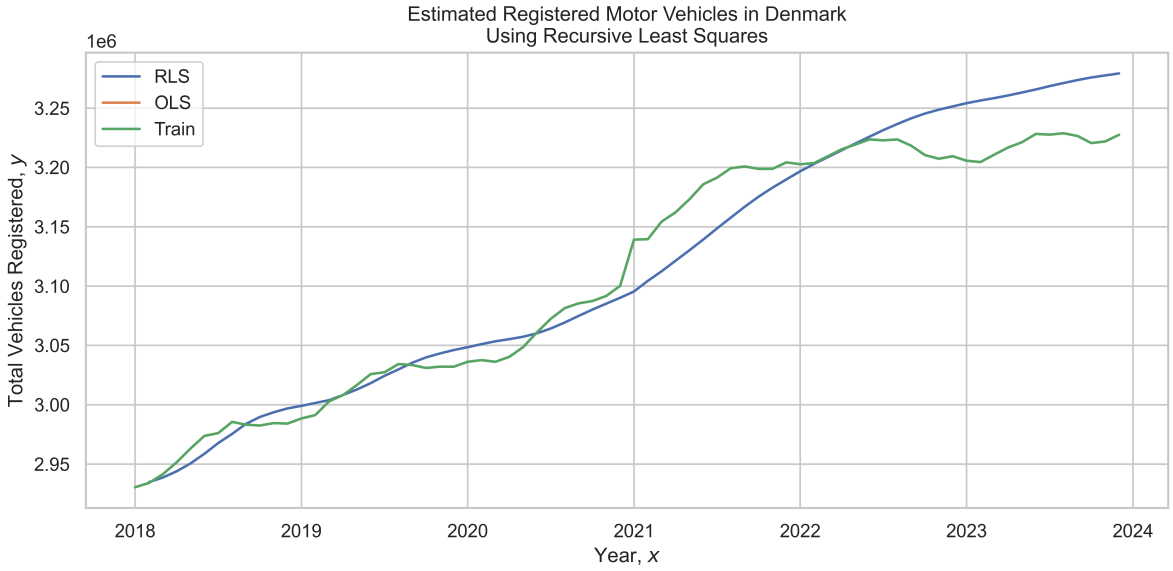

Figure 15: Comparison of the OLS and RLS forecasts of total number of registered vehicles in Denmark with updated initial conditions.

From Figure 15, we can realise that the recursive model could be improved with forgetting as the parameters of a linear model are not constant with respect to time for the given data set.

A.4.4 Recursive Estimation of Forgetting Factor

We first seek to get some intuition for the recursive model with *forgetting*. Having realised previously that the $\mathbf{R}_t$ matrix represents some *knowledge* of the observations accumulated over past iterations, we can understand the *forgetting factor* as a way to *decay* this knowledge.

As such, we introduce the *forgetting factor*, λ_t:

$$\mathbf{R}_t = \lambda_t \mathbf{R}_{t-1} + \boldsymbol{x}_t \boldsymbol{x}_t^T \tag{43}$$

It should be noted that one usually choses λ_t to be a constant and it should not be confused with the power series that appears along the diagonal in Weighted Least Squares with exponential forgetting. The exponential nature of the decay of knowledge is encoded into the iterative algorithm, as can be realised by inspection when expanding $\mathbf{R}_{t-1}$ in Eq. 43.

Using the RLS model as given by the update equations in Section A.4.1 with the forgetting factor modification in Eq. 43, we can now estimate the parameters θ_1, θ_2 as a function of time t for forgetting factors $\lambda \in \{0.6, 0.99\}$.

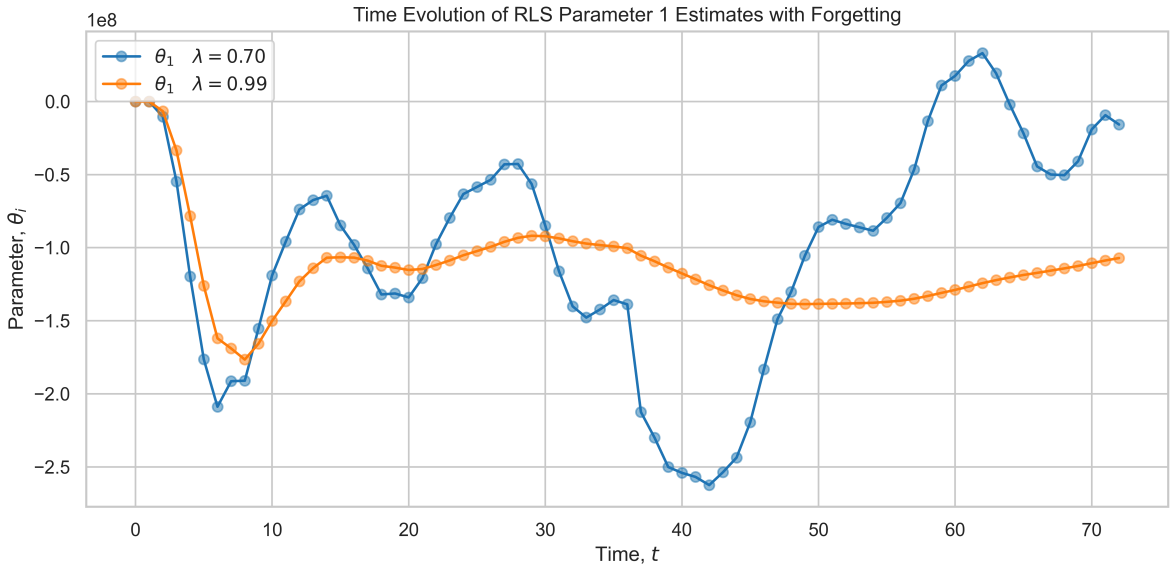

Figure 16: Estimates of parameter θ_1 obtained using Recursive Least Squares with forgetting factors $\lambda \in \{0.6, 0.99\}$.

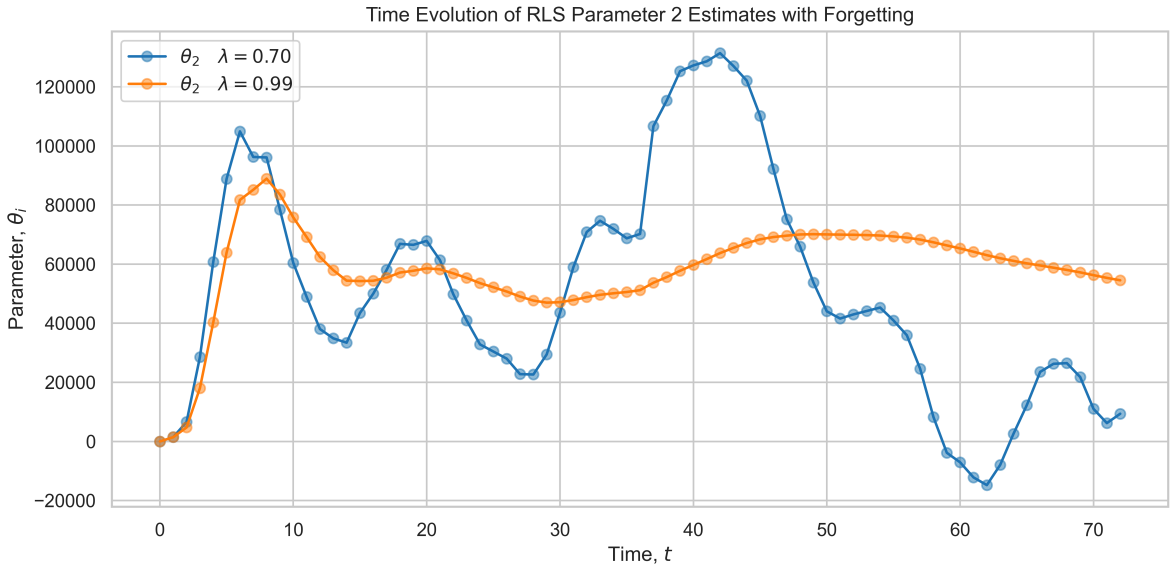

Figure 17: Estimates of parameter θ_2 obtained using Recursive Least Squares with forgetting factors $\lambda \in \{0.6, 0.99\}$.

Inspecting Figure 16 and Figure 17, we observe higher variation in the parameter estimates for $\lambda = 0.6$ than for $\lambda = 0.99$. We can understand this by appreciating the increase in *locality* that arises from a lower forgetting factor. As expected, the more local model varies its parameter estimates more rapidly than a less local model. Both models have been initialised with $\mathbf{R}_0 = \mathbb{I} \cdot 10^{-8}$ and $\boldsymbol{\theta}_0 = \mathbf{0}$.

Model	λ	θ_1	θ_2
WLS	0.7	−15739978.67	9370.98
WLS	0.99	−107098187.37	54532.97
RLS	0.7	−15739977.87	9370.98
RLS	0.99	−107084089.08	54526.0

Table 3: Parameter estimates using WLS and RLS models with different forgetting factors. Note that the parameters for the RLS model are for $t = N$.

From Table 3, we observe that the RLS model with $\lambda = 0.7$ produces final parameter estimates that are very close to the WLS model.

A.4.5 One-Step Predictions with RLS

We introduce the concept of a prediction horizon, k, which denotes the number of steps ahead of given data set we wish to predict. To ease notation, we employ $t + k|t$ to mean a prediction for time $t + k$ given data available at time t. As such:

$$\hat{y}_{t+k|t} = \boldsymbol{x}_{t+k|t}^T \hat{\boldsymbol{\theta}}_t \tag{44}$$

We may then define the residual as:

$$\hat{\varepsilon}_{t+k|t} = y_{t+k} - \hat{y}_{t+k|t} \tag{45}$$

We employ the RLS model with forgetting factor $\lambda \in$ to produce residuals against the training data set for one-step predictions ($k \equiv 1$) and present the results in Figure 18.

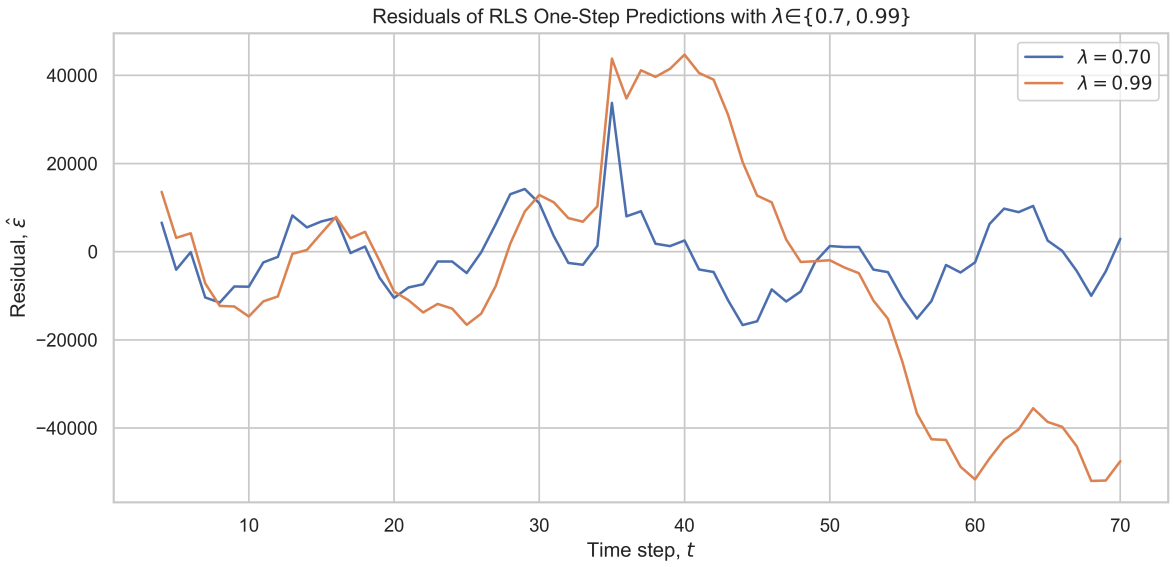

Figure 18: Residuals obtained using Recursive Least Squares with forgetting factors $\lambda \in \{0.6, 0.99\}$ for one-step predictions.

We observe that the model with shorter memory ($\lambda = 0.7$) is generally better at adaption to the changes in trend found in data set. Again, this matches with our expectations that a more local model will be able to better match the local trends in the data. As a result, the magnitude of the residuals is generally lower for the $\lambda = 0.7$ model.

A.4.6 Optimal Forgetting Factor for a Prediction Horizon

We now seek to understand how the optimal forgetting factor λ^* arises for a given prediction horizon characterised by a k-step prediction.

We consider the instructive cases of the extrema, $\lambda \to 0$ and $\lambda \to 1$. For $\lambda \to 0$, we have a model that is *very* local and forgets past information almost immediately. We would expect this to perform relatively poorly, even for a somewhat non-linear model. For $\lambda \to 1$, we have a model that is *very* global and retains past information almost indefinitely, will perform well for a truly linear model, but for coloured noise, it will perform poorly.

Appreciating these two logical extrema, we can begin to see how our forgetting factor should be matched to the horizon we are trying to predict. If we just want to predict the next timestep, it is intuitive that a more local model would in generally perform better since longer-term non-linear trends will not have a significant impact on the prediction. On the other hand, if we are trying to predict further into the future, an overly local model would not have sufficient understanding of longer-term trends that may dominate the residual for a prediction horizon of this width.

To investigate this, we run the RLS model with forgetting factors $\lambda \in \{0.50, 0.51, \ldots, 0.99\}$ for prediction horizons $k \in \{1, 2, \ldots, 12\}$. and collect the Root Mean Square Error as a function of k and λ:

$$\text{RMSE}(k, \lambda) = \sqrt{\frac{1}{N-k} \sum_{t=1}^{N-k} \hat{\varepsilon}^2_{t+k|t}(\lambda)} \tag{46}$$

Where $\hat{\varepsilon}_{t+k|t}(\lambda)$ denotes the residuals for an RLS model with forgetting factor λ and prediction horizon k.

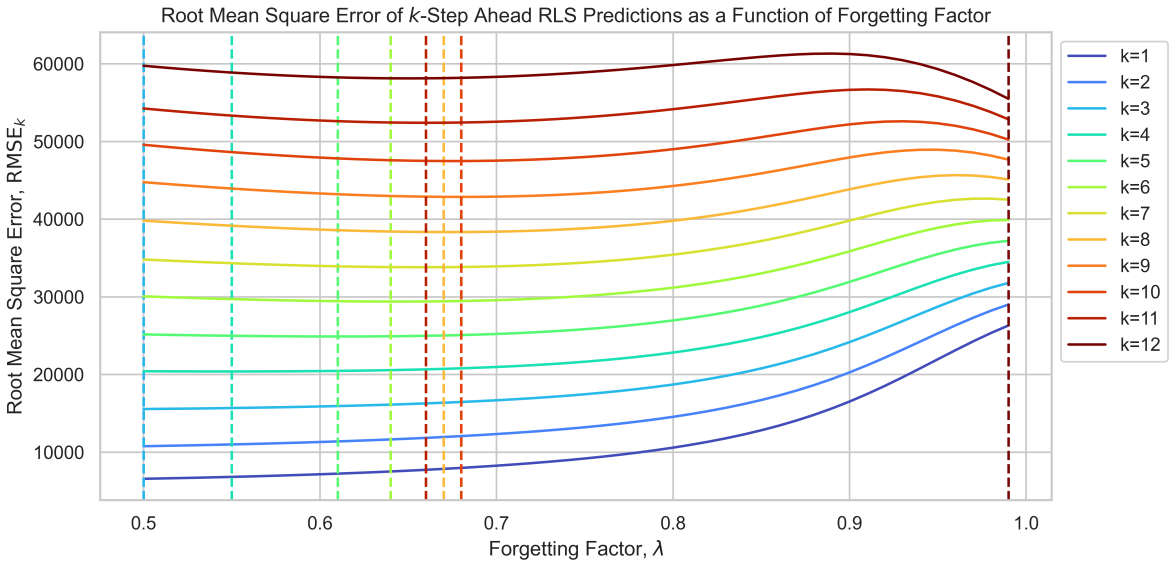

Figure 19: Root Mean Square Error as a function of prediction horizon k and forgetting factor λ. The minimal RMSE for each k is marked with a dashed vertical line of the corresponding colour.

From Figure 19, we observe that the optimal forgetting factor for the very highest value of $k = 12$ is $\lambda = 0.99$, while shorter prediction horizons are better served by a more local models. A choice of around $\lambda = 0.65$ is generally a good choice across different horizons.

Taking note of the shape of the curves in Figure 19, we can validate our speculation above. For short prediction horizons, the optimal forgetting factor is lower, while for longer prediction horizons, the optimal forgetting factor is higher.

A.4.7 Predictions Using RLS and Other Models

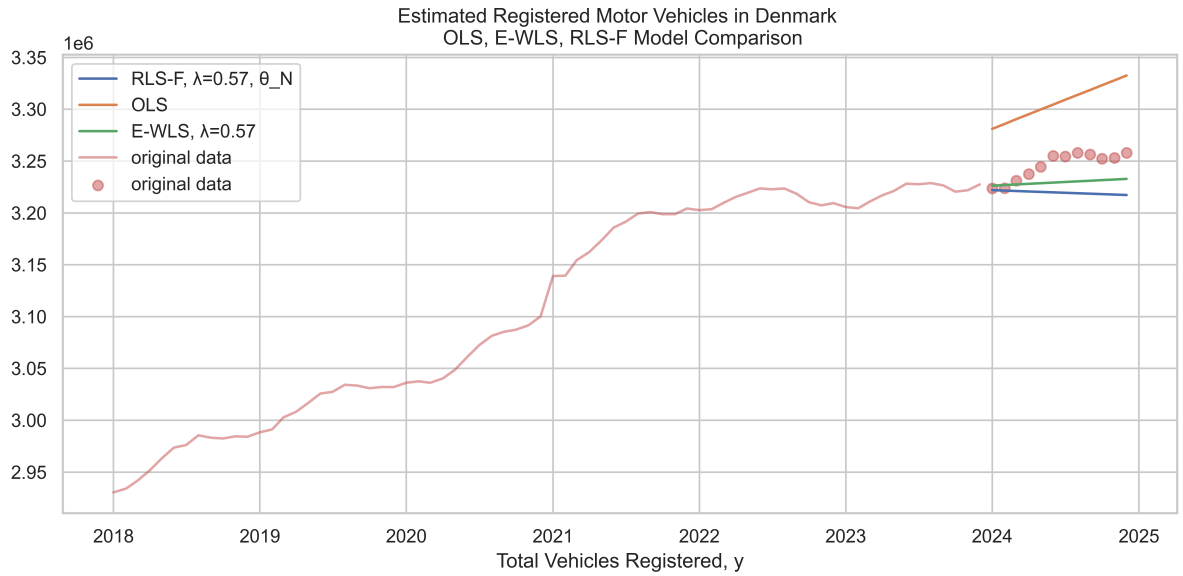

Figure 20: Prediction of total car sales, compared by model (OLS, WLS, RLS) for horizon $k = 12$ and forgetting factor $\lambda = 0.57$.

Here the weight factor $\lambda = 0.57$ is chosen loosely based on the previous Figure 19, as an estimated good value (in the middle of optimal values for $k \in \{1, \ldots, 11\}$). Therefore, it is not surprising that for $k = 12$ the RLS-F (RLS with forgetting) does perform very poorly. For the chosen time horizon, the RLS-F model would perform best for $\lambda \to 1$, at which point it basically becomes an OLS model, better at predicting more global trends.

In this szenario the E-WLS (exponential/local WLS) model performs best, simply by visual analysis. Yet, all models perform quite poorly, not finding a good compromise between local trends and global trend. This could be overcome by different choices of λ. However, the general notion of "forgetting" models (such as E-WLS and RLS-F) being better at predicting short term horizons (e.g. $k \in \{1, 2, 3, 4\}$) remains. Nonetheless, this is highly dependent on noise and variance in the given time-series.

In the short following exploration, we investigate superficially, whether the choice of λ fundamentally changes that notion or just marginally improves the models predictive power. This time, we consider a range for λ, where it does not have its absolute minimum, but given Figure 19 for $k = 12$, the RMSE is close to minimal values.

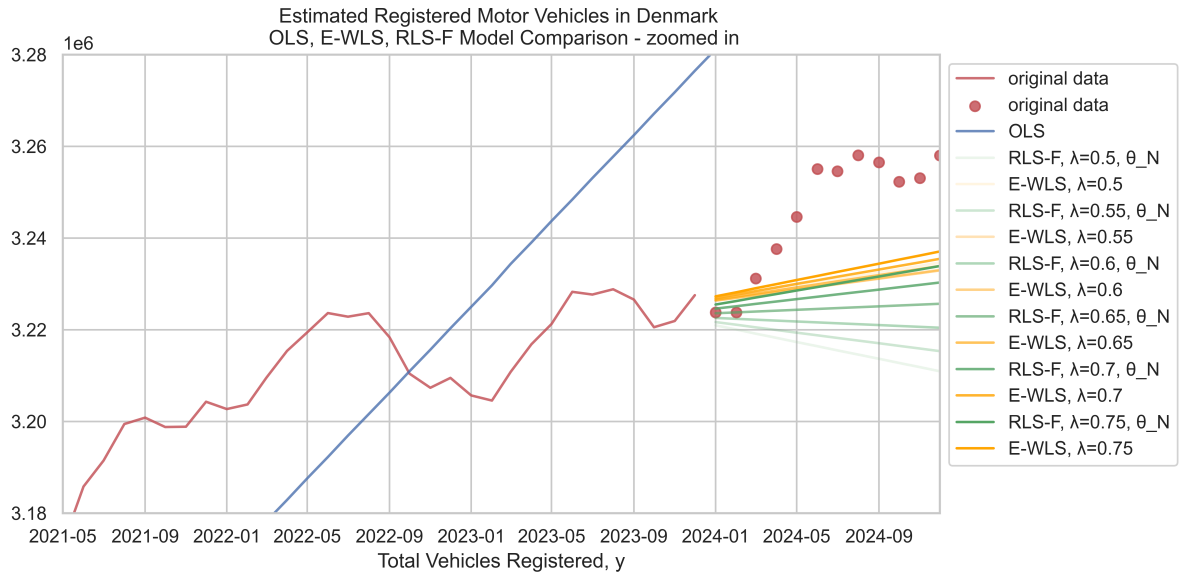

Figure 21: Zoomed in prediction graph of total car sales, compared by model (OLS, WLS, RLS) for horizon $k = 12$ and forgetting factor $\lambda \in [0.5, 0.75]$.

As we can see in Figure 21, the RLS-F with high "forgetting" (low λ) does capture extremely short-term trends; in fact, goes opposite the global trend. For higher λ it approximates the WLS model, for $\lambda > 0.87$ (according to Figure 19) it will approximate the OLS solution. Overall, the predictive power does improve for certain parameter λ, but only for select time horizons $k \in \{1, 2\}$.

A.4.8 Reflections on time adaptive models

A.4.8.1 Overfitting VS Underfitting

The problem with these linear models arises from their simplicity - they do not have the flexibility to capture the dynamics of the underlying process, and as a result they generally performed somewhat poorly in explaining the data and predicting it.

Possible mitigations to consider would be:

- Increasing the amount of parameters for the model, thus making it a multivariate linear regression with hyperplanes as predictors - the dataset has more exogenous variables to offer
- Increasing the model complexity to for example polynomial models - however, these come with different problems, especially for time-series where they can exhibit asymptotic behaviour towards boundries
- Tuning the λ hyperparameer or even the structure of the entire weight matrix $\mathbf{W}$

We generally observe underfitting in the models here, given the model is unable to even describe the training data well. In this instance, the underfitting arises not from a lack of data, but rather a lack of model complexity.

A.4.8.2 Test Set Challenges

There are challenges in time-series due to covariance and independence assumptions. So far, each observation is treated as an independent random variable (with the exception of an exponential weight matrix, which somewhat introduces the notion of more recent samples being more important than older ones).

However, time series can present trend-changes (as is the example here) or even seasonality behaviours. This is different from a dataset, that simply maps one variable or event to another, where order of the events is not relevant.

Thus, splitting the dataset can introduce difficulties when the underlying process is not stationary or in cases where the model does not sufficiently capture the dynamics of the underlying process. This can severely impact the models predictive ability. The chosen timeframe may not describe the trend well of this has changed across the training/test boundary.

One way to handle this would be by cross-validation. A common method, that would also be applicable to time-series is an adjusted K-Fold cross-validation. Basically, creating different models by sequentially leaving out some parts of the training data and then weighting by the confidence of each model. Bootstrapping is another famous method but due to its nature it is not applicable here.

A.4.8.3 Recursive Estimation as a Saviour?

Actually the step-wise prediction (for only the test dataset though) was superior to global model parameters.

Yet, it was not applicable or helpful for *actual* prediction, as the time horizon to predict into the future may vary and the last parameter estimation cuts off at the end of the training dataset. Hence it is does not have the advantages of continually updating parameters any longer.

However, in general the RLS models suffer from 'burn-in' where the initial predictions may have a large uncertainty or bias. Additionally, they are more computationally expensive. These properties make then not a good fit so far, but it might depend on the origin of the time series.

A.4.8.4 Additional adaptive estimation techniques

Some ideas for adaptive (maybe recursive) estimation on time series:

- Bayesian models, taking new information to update likelihood and posterior; would also allow for a good initial estimate depending on the origin of the data for the prior distribution
- momentum models; taking sudden large jumps in time-series as a penalty to a weight, not to update much -> would be great for noisy time series
- averaging different weighted models to capture notions of local trend changes and reversion to a global trend
- dynamic weight models; adjust the size of a WLS dynamically/recursively, based on the variance, noise or momentum of data
- recurrent neural-network based models -> most often a good predictor but hard to interpret or modify without continuous re-training (very computationally expensive)
- regularized models: Ridge-regression (L2) or ISTA (L1), the latter does not work well on most time-series because the L1 norm penalizes mostly "edges"

A.4.8.5 Additional Comments

- RLS models can be labourious to get right, especially choosing initial values for θ, R_t and λ
- a huge pitfall is to think that: modelling a time series, that is (visually) very obviously not showing a linear behavior, with a simple linear model…is a good idea
- as for the initial description and visual analysis, it became apparent that there are significant trend changes present; so much in fact, that one could possibly also just take the last few data points to predict and save oneself all the hustle - a wiser choise of train/test split would significantly improve the models

Bibliography

[1] H. Madsen, *Time Series analysis*. Chapman & Hall/CRC, 2008.