university of groningen / faculty of science and engineering

Bachelor Project Mathematics

How fast can we run?

T.M. van Belle
s2766582

Abstract. In this paper, different models for the yearly best times on the marathon are derived. This was done using both maximum likelihood estimation and cubic spline models. In the models based on maximum likelihood estimation, different trends throughout the dataset are considered. Namely linear, quadratic, and exponential. It was found that the exponential model has the best fit. According to this model, the mean yearly best time will converge to a time of 2:03:53. Furthermore three different cubic spline models are derived. These models were found to be quite vulnerable in predicting future times.

supervised by
Prof. dr. E.C. Wit

Contents

1 Introduction **2**

2 Maximum Likelihood Estimation **5**
2.1 Linear model 5
2.2 Quadratic Model 7
2.3 Exponential-decay Model 9

3 Confidence Intervals **11**
3.1 Cramér Theorem 11
3.2 The Delta Method 12
3.3 Computing Confidence Intervals 13

4 Cubic Splines **13**
4.1 Piecewise Polynomials 14
4.2 Non-penalised Cubic Spline Model 17
4.3 Penalised Cubic Spline 19
4.4 Choosing the Smoothness Parameter 20
4.5 Monotonic Cubic Splines 21

5 Residual Analysis **22**

6 Conclusion **24**

A Dataset **26**

B Programms **28**
B.1 Dataverwerking 28
B.2 Log-likelihood Functions 30
B.3 Maximum Likelihood Estimation 31
B.4 (Penalised) Cubic Spline 34
B.5 Monotonic Cubic Spline 39

C Derivation of the matrix S **40**

1 Introduction

On Saturday the sixth of May 2017, reigning world champion Eliud Kipchoge ran a stunning 2:00:25 on a marathon in Monza. This race was organized by Nike to try and break the two-hour barrier. The organisers made sure that the conditions were optimal for long distance running. For instance, they hired some world class pacers, who took turns in pacing Kipchoge around the 2.4 kilometers long circuit in Monza. Also, drinks were available at any given moment. As a consequence, the International Association of Athletics Federations (IAAF) did not recognise this event as an official marathon. Hence, the time Eliud Kipchoge ran does not count as an official world record. According to Nike, it does however show that a sub-two hour marathon can be possible in the near future.

Athletics records in general are a subject of great interest for both statisticians and physiologists. An interesting question to both of them is: What is the maximum possible performance for an athlete? The physiologist will mainly focus on human physiology, trying to find out what is still within reach of human body. Statisticians will focus on the available record data, to try and find the statistical lower, and in some cases upper, limit. A lot of research already has been done in this field.

For instance, one could consider short term forecasting future records. This can be done using the yearly best times from previous years. To account for differences in race conditions, it is assumed that:

$$Y_n = X_n + c_n,$$

where Y_n is the yearly best time in year n, X_n's are independent and identically distributed and c_n is a non-random trend. In this model, different distributions and trends for X_n and c_n respectively can be considered. Richard L. Smith[8] considered three distributions:

- Normal distribution
- Gumbel distribution
- Generalized Extreme Value distribution

Furthermore, three trends were considered:

- Linear drift
- Quadratic drift
- Exponential decay

Hence, a total of nine different models were considered. The parameters were estimated by applying numerical maximum likelihood. Of course, the models were analysed using residual analysis. It was found that none of the models were particularly better than the linear model with normal distribution (i.e. the sum of the residuals did not decrease as other models were fitted).

The next step would be to verify whether these models could also predict future times well. Since predicting times can be quite unstable, predictive distributions were made. By looking at different quantiles of this distribution, one could get an indication of times that are likely to be ran in the far future. A natural way to develop predictive distribution is using Bayesian statistics, as done by R. L. Smith and J. E. Miller [9]. As expected, the results for this method were, on the short term prediction, quite similar to the maximum likelihood methods as discussed before.

Another point of interest could be verifying whether a new record is consistent with previous record data. If the newly set time is not consistent with previous data, this could be an indication of performance-enhancing drug usage. One particular example that has been studied repeatedly is Wang Junxia's time on the women's 3000m. In 1993, she broke the world record by more than 16 seconds, causing a lot of suspicion amongst other athletes. To verify the validity of Junxia's time, models for an ultimate record were derived.

Modelling the so called ultimate record can be done in many different ways. M.E. Robinson and J.A. Tawn[7] proposed to use the Generalized Extremal Value distribution, with an exponential decay of the mean parameter μ_t:

$$\mu_t = \alpha - \beta \left(1 - \exp(-\gamma t)\right).$$

Defining $x_{p,t}$ to be the time which is beaten with probability p in year t, then

$$x_{p,t} = \mu_t + \sigma \frac{\left(1 - \log\left(\frac{1}{1-p}\right)\right)^{-\xi}}{\xi},$$

where ξ, μ, and σ are parameters of the generalized extreme value distribution. Then, we can obtain an expression for the ultimate record by letting $p = 0$ and $t \to \infty$:

$$x_{ult} = \left\{ \begin{array}{ll} \alpha - \beta + \sigma/\xi & \text{for } \xi < 0 \\ -\infty & \text{for } \xi \geq 0. \end{array} \right\}$$

Hence, by obtaining estimates for all the parameters involved, an estimate for the ultimate record can be derived. This method gives an estimate that

is a statistical asymptote. This gives a nice marker, but it does not necessarily say a lot about a performance by some athlete at the present time.

Another interesting question is finding the "ultimate record" that could be ran today, i.e. what is possible for current athletes to run under perfect conditions. As proposed by J.H.J. Einmahl and J.R. Magnus [2], instead of using the yearly best times as a data set, one could consider using all top performances to have an estimate for given the current regulations and materials available. The personal bests of the best m athletes are considered as iid observations from some distribution $F(x)$. Let $X_{1,m} \leq X_{2,m} \leq \ldots \leq X_{m,m}$ be the associated ordered statistics, where $X_{m,m}$ denotes the world record. The aim would then be to estimate the right endpoint of the distribution $F(x)$,

$$x^* = \sup\{x|F(x) < 1\}.$$

This endpoint really represents the statistical asymptote of the best times ran as of today. Hence, this can be seen as the limit time that could be ran today.

Furthermore, it could be interesting to compare performances on different athletics events. In this way, a measure for the quality of the current world record can be found. H.J. Grubb[5] proposed to use the average speed on certain running events as a measure to compare performances. This seems like a natural measure, since it takes you at least twice as long to run double the distance. Hence, the average speed is lower as the distance increases. However, it was found that this is not completely true. For instance the average speed of the 100m world record is slower than the speed of the 200m world record. Hence, these two events can be neglected, and further computations were done on the events of at least 400m. Since the speed decreases with distance, it is assumed that

$$\log(v_i) = -a + (1 - b)\log(d_i),$$

where v_i is the speed on distance d_i. Since this model was found to be not flexible enough, another model is proposed,

$$v_i = \frac{A}{\log(d_i) - B} + C,$$

where A measures the decrease in speed, $\exp(B)$ is an asymptote which can be interpreted as the distance at which the maximum speed is attained, and C is the asymptote for speed at very long distances.

Throughout this paper, we will focus on the prediction of records, both long term and short term. The main questions we would like to answer are:

- Is it possible for humans to run a marathon within two hours?
- If so, when can we expect to see such an event?

To be able to create a better model, cubic spline models will be investigated. The main idea of cubic spline regression is to divide the data into multiple parts, and fit a third degree polynomial in each part. These polynomials should satisfy the following properties:

- The composition of the polynomials together is twice differentiable,
- The first and second derivative of the composition are continuous.

Since a polynomial fit is made in each part of the data, these models allow for great flexibility. Not only is a third degree polynomial quite flexible, but there are also a lot of options in choosing the division of the data. A major disadvantage however, is that these models are only capable of short term prediction. Due to the great flexibility, a lot of uncertainty is introduced in the model.

Thoughout this paper, a dataset of the yearly best times on the marathon will be used, the dataset can be found in appendix A. In this dataset, for each year between 1940 and 2016, the best time on an official IAAF event in that particular year is given. Furthermore there is some information on runner that ran this particular time, and the location. In this thesis, only the year and the yearly best times will be used. For practical purposes the year will be normalised to 1940. (i.e. For year k, the normalised year will be $(k - 1940)$).

2 Maximum Likelihood Estimation

In this section, three different models for the yearly best time on the marathon will be derived. It is assumed that the yearly best time for year n, Y_n where $n \in \{0, \ldots, N\}$, has a normal distribution with mean μ_n depending on the year n, and with standard deviation σ. Note that in these models, it is assumed there is only one trend throughout the whole data set. This does not necessarily have to be the case, but it is assumed for simplicity of the models. In section 4, models with multiple trends will be derived. Also, one could consider different distributions for Y_n, however it was found by Smith[8] that the normal distribution was the most suitable.

2.1 Linear model

First of all, an easy model is considered, where Y_n's are normally distributed with a linear function for the mean. In other words,

$$Y_n \sim \mathcal{N}(\mu_n, \sigma^2),$$

with,

$$\mu_n(\boldsymbol{\beta}) = \beta_0 - \beta_1 n,$$

where $\beta_1 > 0$.
From here, the goal is to estimate the parameters β_0, β_1, and σ using maximum likelihood estimation. To achieve this, the logarithm of the likelihood function is considered and the maximum of this function given the dataset y_n will be found. The likelihood function of Y_n is given by:

$$L(\sigma, \beta_0, \beta_1 | y_n) = \prod_{n=0}^{N} f(y_n | \sigma, \beta_0, \beta_1).$$

Hence, the log-likelihood function is given by:

$$\begin{aligned} l(\sigma, \beta_0, \beta_1 | y_n) =& \sum_{n=0}^{N} \log\left(f(y_n - \mu_n(\boldsymbol{\beta}))|\sigma, \beta_0, \beta_1\right) \\ =& \sum_{n=0}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left[-\frac{(y_n - \beta_0 + \beta_1 n)^2}{2\sigma^2}\right]\right) \\ =& (N+1) \cdot \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{1}{2\sigma^2} \cdot \sum_{n=0}^{N} (y_n - \beta_0 + \beta_1 n)^2. \end{aligned}$$

Then, using maximum likelihood estimation, the optimal solution can be derived by setting the derivatives with respect to β_0 and β_1 equal to zero. Then, the following system of equations is obtained:

$$\frac{\mathrm{d}}{\mathrm{d}\,\beta_0} l(\sigma, \beta_0, \beta_1 | y_n) = \frac{1}{\sigma^2} \cdot \sum_{n=0}^{N} (y_n - \beta_0 + \beta_1 n) = 0$$

$$\frac{\mathrm{d}}{\mathrm{d}\,\beta_1} l(\sigma, \beta_0, \beta_1 | y_n) = -\frac{1}{\sigma^2} \cdot \sum_{n=0}^{N} ((y_n - \beta_0 + \beta_1 n) n) = 0$$

This system can be solved by hand, but since the data is available digitally, a numerical solution is more practical. Therefore, the log-likelihood function was programmed in R[6], and the numerical estimations were found using the optim function. The programs used can be seen in Appendices B.2 and B.3(lines 5-14). The estimates were found to be:

$$\hat{\beta}_0 = 8852.49606, \quad \hat{\beta}_1 = 22.68473, \quad \hat{\sigma} = 271.12109.$$

As can be seen in Figure 1, the resulting model not a very good fit. Hence, taking another trend in the mean μ_n is needed to obtain a better model.

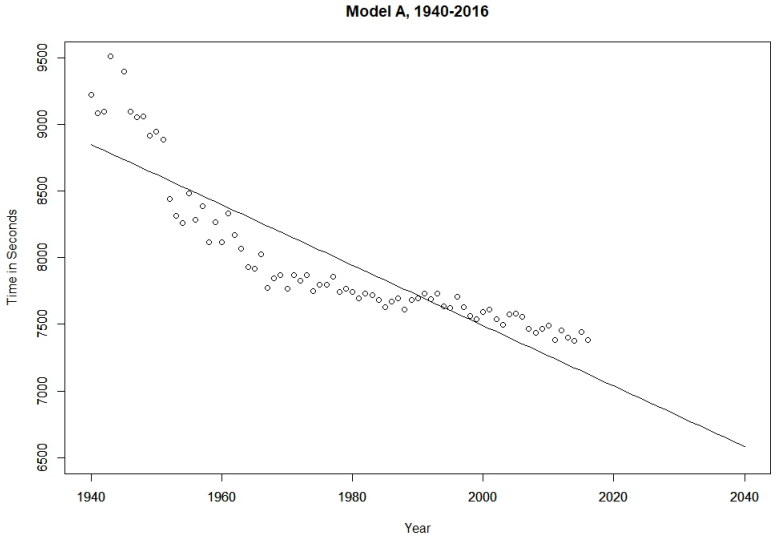

Figure 1: Linear Model fitted to the data

2.2 Quadratic Model

Since the linear model is not satisfactory, a model with a quadratic drift in the mean μ_n is derived. Note that we still assume that Y_n has a normal distribution with mean μ_n and variance σ^2. The quadratic-drift in μ_n can be expressed as follows:

$$\mu_n(\beta) = \beta_0 - \beta_1 n + \beta_2 \frac{n^2}{2},$$

where $\beta_1 > 0$.
As before, the likelihood function can be derived using:

$$L(\sigma, \beta_0, \beta_1, \beta_2 | y_n) = \prod_{n=0}^{N} f(y_n | \sigma, \beta_0, \beta_1, \beta_2).$$

Hence, the log-likelihood function is given by:

$$
\begin{aligned}
&l(\sigma, \beta_0, \beta_1, \beta_2 | y_n) \\
&= \sum_{n=0}^{N} \log\left(f\left(y_n - \mu_n(\boldsymbol{\beta}) | \sigma, \beta_0, \beta_1, \beta_2\right)\right) \\
&= \sum_{n=0}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left[-\frac{\left(y_n - \beta_0 + \beta_1 n - \beta_2 \frac{n^2}{2}\right)^2}{2\sigma^2}\right]\right) \\
&= (N+1) \cdot \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{1}{2\sigma^2} \cdot \sum_{n=0}^{N} \left(y_n - \beta_0 + \beta_1 n - \beta_2 \frac{n^2}{2}\right)^2.
\end{aligned}
$$

Maximizing this expression will give the estimates for the parameters. This was again done using the optim function in R (B.3(lines 32-41)). The estimates were found to be:

$$\hat{\beta}_0 = 9321.0995642, \quad \hat{\beta}_1 = 60.1797866, \quad \hat{\beta}_2 = 0.9867799, \quad \hat{\sigma} = 161.3466542.$$

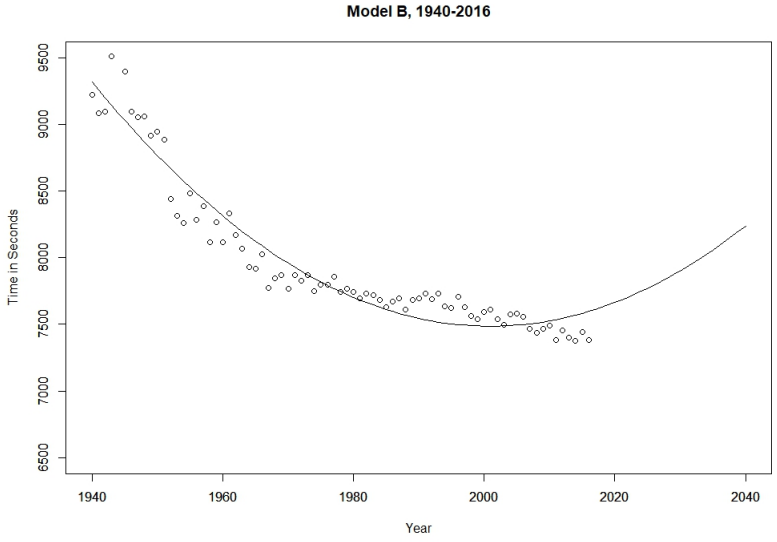

Figure 2: The Quadratic Model fitted to the data

As can be seen in Figure 2, this model seems to be a better fit. However, due to the quadratic shape, the model does not seem to give a great prediction for the future. Hence we would like to have a model that has a similar fit, but is also able to predict future times (at least on short term).

2.3 Exponential-decay Model

By looking at the data, some exponential decay model seems to be the best possible fit. We define the following function for the mean μ_n:

$$\mu_n(\boldsymbol{\beta}) = \beta_0 - \frac{\beta_1}{\beta_2} \cdot (1 - (1 - \beta_2)^n),$$

where $\beta_1 > 0,$ and $0 < \beta_2 < 1$.
One of the advantages of this model is the fact that it has a limit as n tends to infinity, namely,

$$\lim_{n\to\infty} \beta_0 - \frac{\beta_1}{\beta_2} \cdot (1 - (1 - \beta_2)^n) = \beta_0 - \frac{\beta_1}{\beta_2}.$$

Note that this model gives an estimate for the yearly best time in a particular year. This does not mean it is a prediction for the world record itself. Hence when using this model, we can find an eventual limit for the mean of the yearly best time Y_n. Finding the parameter estimates for this particular model is quite difficult, since the likelihood function is more complicated than the ones used in both the linear and quadratic models. The likelihood function in this case is given by:

$$L(\sigma, \beta_0, \beta_1, \beta_2 | y_n) = \prod_{n=0}^{N} f(y_n | \sigma, \beta_0, \beta_1, \beta_2).$$

Hence, the log-likelihood function is given by:

$$\begin{aligned}
&l(\sigma, \beta_0, \beta_1, \beta_2 | y_n) \\
=&\sum_{n=0}^{N} \log\left(f\left(y_n - \mu_n(\boldsymbol{\beta}) | \sigma, \beta_0, \beta_1, \beta_2\right)\right) \\
=&\sum_{n=0}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left[-\frac{\left(y_n - \beta_0 + \frac{\beta_1}{\beta_2} \cdot (1 - (1 - \beta_2)^n)\right)^2}{2\sigma^2}\right]\right) \\
=&(N+1) \cdot \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{1}{2\sigma^2} \cdot \sum_{n=0}^{N} \left(y_n - \beta_0 + \frac{\beta_1}{\beta_2} \cdot (1 - (1 - \beta_2)^n)\right)^2.
\end{aligned}$$

Maximizing this expression will give the estimates for the parameters. This was also done using the optim function in R(B.3(lines 59-68)). The estimates were found to be:

$$\hat{\beta_0} = 9558.579, \quad \hat{\beta_1} = 103.0248, \quad \hat{\beta_2} = 0.04848448, \quad \hat{\sigma} = 134.9606.$$

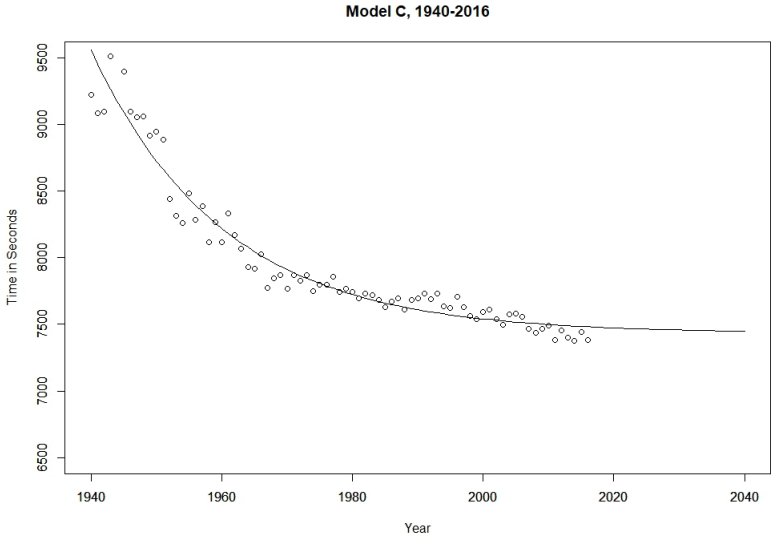

Figure 3: The Exponential Model fitted to the data

In Figure 3, it can be observed that the fitness of this model seems to be better than the fitness of both the linear and the quadratic model. However, since we only assumed one trend though-out the whole dataset, the model is over/under estimating some parts of the data. Hence, it might be convenient to assume multiple trends on different parts of the dataset. Such models will be derived in section 4.
It can also be seen that this model actually has a statistical asymptote. This asymptote can be found by taking the limit as n tends to infinity of the mean μ_n.

$$\lim_{n\to\infty} \hat{\beta}_0 - \frac{\hat{\beta}_1}{\hat{\beta}_2} \cdot \left(1 - \left(1 - \hat{\beta}_2\right)^n\right) = \hat{\beta}_0 - \frac{\hat{\beta}_1}{\hat{\beta}_2} \approx 7433.474.$$

Hence, we can conclude that the mean of the yearly best time will tend towards a time of $2:03:53$. Please note that the estimated standard deviation of the yearly best time is $\hat{\sigma} = 134.9606$. Eventually, the yearly best time will be normally distributed with mean $\mu = 7433.474$ and standard deviation $\sigma = 134.9606$. Hence we can derive a 95% confidence interval for the yearly best time on the long run. The 95% confidence interval was found to be:

$$[7168.935, 7698.013].$$

Hence we can conclude that, according to the derived model, in each year, with at least 97.5% certainty, a time under $1:59:28$ will not be ran. However, there is also some uncertainty in the model itself. Hence we would

like to find some confidence interval for the model as well. This will be done in the next section.

3 Confidence Intervals

The models derived in section 2 give a prediction for the yearly best time in the future, however we would like to know how certain the derived models are. Hence 95% confidence intervals for the models will be derived. To find these intervals, the variance of the models has to be derived. This can be done using the asymptotic normality of the maximum-likelihood estimate.

3.1 Cramér Theorem

The following theorem can be used to show the asymptotic normality of the maximum likelihood estimate.

Theorem 3.1. *(Cramér) Let $X_1, X_2, \ldots$ be i.i.d. with density $f(x|\boldsymbol{\theta})$, and let $\boldsymbol{\theta}_0$ denote the true value of the parameter. If*

1. *$\boldsymbol{\Theta}$ is an open subset of $\mathbb{R}^k$,*
2. *Second partial derivatives of $f(x|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ exist and are continuous for all x, and may be passed under the integral sign,*
3. *There exists a function $K(X)$ such that $E_{\boldsymbol{\theta}}(K(x)) < \infty$ and each component of $\dot{\boldsymbol{\Psi}}(x, \boldsymbol{\theta})$ is bounded in absolute value by $K(x)$ uniformly in some neighbourhood of $\boldsymbol{\theta}$,*
4. *$\mathcal{F}(\boldsymbol{\theta}) = E_{\boldsymbol{\theta}}\dot{\boldsymbol{\Psi}}(X, \boldsymbol{\theta}_0)$ is positive definite,*
5. *$f(x|\boldsymbol{\theta}) = f(x|\boldsymbol{\theta}_0)$ a.e. implies that $\boldsymbol{\theta} = \boldsymbol{\theta}_0$,*

Then, there exists a strongly consistent sequence $\hat{\boldsymbol{\theta}}_n$ of roots of the likelihood equation such that

$$\sqrt{n}(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_0) \xrightarrow{\mathcal{L}} \mathcal{N}(\mathbf{0}, \mathcal{F}(\boldsymbol{\theta}_0)^{-1}).$$

Proof. The proof for this theorem is rather complicated and can be found on page 121-122 of [3]. □

It is often said that this theorem states that the maximum likelihood estimate is asymptotically normal. However, this is not true. If there is a unique root of the likelihood equation for every n, this sequence will be consistent and asymptotically normal. Note that the parameter estimates for the models derived in section 2 are unique roots of the likelihood equation, hence we

can conclude that the estimates are strongly consistent and asymptotically normal. That is:

$$\hat{\boldsymbol{\beta}} \xrightarrow{\mathcal{L}} \mathcal{N}\left(\boldsymbol{\beta}, \frac{\mathcal{F}(\boldsymbol{\beta})^{-1}}{n}\right).$$

Hence,

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}) \approx \frac{\mathcal{F}(\boldsymbol{\beta})^{-1}}{n},$$

where $\boldsymbol{\beta}$ is the true parameter. Hence we can find the variance-covariance matrix of the estimator $\hat{\boldsymbol{\beta}}$, by finding the inverse of the Fisher information matrix. The Fischer information matrix is defined by:

$$\mathcal{F}(\boldsymbol{\beta}) = -\mathbb{E}\left(\frac{\partial^2}{\partial \boldsymbol{\beta}^2} \log\left(f(\boldsymbol{Y}|\boldsymbol{\beta})\right)\right).$$

The optim function in R gives the Hessian of the maximized function, which is equal to the Fischer information matrix.

3.2 The Delta Method

To be able to compute the variances of the different models, the variance has to be derived first. In the case of the linear model this can be done quite easily:

$$\mathrm{Var}\left(\beta_0 - \beta_1 x\right) = \mathrm{Var}(\beta_0) + x^2\mathrm{Var}(\beta_1) - 2x\mathrm{Cov}(\beta_0, \beta_1).$$

When working with the quadratic model, things get somewhat more difficult however it can be found that:

$$\begin{aligned}
&\mathrm{Var}\left(\beta_0 - \beta_1 x + \beta_2 \frac{x^2}{2}\right) \\
=&\mathrm{Var}(\beta_0) + x^2\mathrm{Var}\left(-\beta_1 + \beta_2\frac{x}{2}\right) + 2x\mathrm{Cov}\left(\beta_0, -\beta_1 + \beta_2\frac{x}{2}\right) \\
=&\mathrm{Var}(\beta_0) + x^2\left(\mathrm{Var}(\beta_1) + \frac{x^2}{4}\mathrm{Var}(\beta_2) - x\mathrm{Cov}(\beta_1, \beta_2)\right) \\
&+ 2x\mathrm{Cov}\left(\beta_0, -\beta_1 + \beta_2\frac{x}{2}\right) \\
=&\mathrm{Var}(\beta_0) + x^2\mathrm{Var}(\beta_1) + \frac{x^4}{4}\mathrm{Var}(\beta_2) - x^3\mathrm{Cov}(\beta_1, \beta_2) - 2x\mathrm{Cov}\left(\beta_0, \beta_1\right) \\
&+ x^2\mathrm{Cov}\left(\beta_0, \beta_2\right).
\end{aligned}$$

Although this expression is rather complicated, it allows for computation of confidence intervals. For the exponential-decay model however this does not work. Hence some other method needs to be found in order to compute the variance.

In general, we would like to find the variance of some function $h(\hat{\boldsymbol{\beta}})$. Note that using a first order Taylor expansion, it can be found that:

$$h(\hat{\boldsymbol{\beta}}) \approx h(\boldsymbol{\beta}) + \nabla h(\boldsymbol{\beta})^T \left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right).$$

Hence,

$$\begin{aligned}
\mathrm{Var}(h(\hat{\boldsymbol{\beta}})) &\approx \mathrm{Var}\left(h(\boldsymbol{\beta}) + \nabla h(\boldsymbol{\beta})^T \left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right)\right) \\
&= \mathrm{Var}\left(\nabla h(\boldsymbol{\beta})^T \hat{\boldsymbol{\beta}}\right) \\
&= \nabla h(\boldsymbol{\beta})^T \mathrm{Cov}\left(\hat{\boldsymbol{\beta}}\right) \nabla h(\boldsymbol{\beta}) \\
&\approx \nabla h(\hat{\boldsymbol{\beta}})^T \mathrm{Cov}\left(\hat{\boldsymbol{\beta}}\right) \nabla h(\hat{\boldsymbol{\beta}}).
\end{aligned}$$

Now, we can apply this with

$$h(\hat{\boldsymbol{\beta}}) = \hat{\beta}_0 - \frac{\hat{\beta}_1}{\hat{\beta}_2} \cdot \left(1 - \left(1 - \hat{\beta}_2\right)^x\right).$$

In order to compute the variance of the exponential-decay model, the first order partial derivatives of $h(\boldsymbol{\beta})$ with respect to β_0, β_1, and β_3 have to be found. It can be verified that:

$$\nabla h(\boldsymbol{\beta}) = \begin{pmatrix} 1 \\ -\frac{1-(1-\beta_2)^x}{\beta_2} \\ \beta_1 \frac{1-(1-\beta_2)^x - \beta_2 x(1-\beta_2)^{x-1}}{\beta_2^2} \end{pmatrix}.$$

3.3 Computing Confidence Intervals

Now, it is possible to compute 95% confidence intervals for the three different models. Hence the programs used in section 2 were extended to compute confidence intervals B.3(lines 16-24; 43-51; 72-86). The method used to find the confidence interval was adding and subtracting 1.96 times the standard deviation of the model from the model. The models including 95% confidence intervals can be found in Figures 4,5, and 6.

4 Cubic Splines

In previous sections, we have focussed on making models to predict future yearly best times. However, in these models it was assumed that there was only one random trend thoughout the whole dataset. One can imagine that in general, this is not the case. For instance, the non-random trend could be linear in some parts of the data, and quadratic or exponential in other parts.

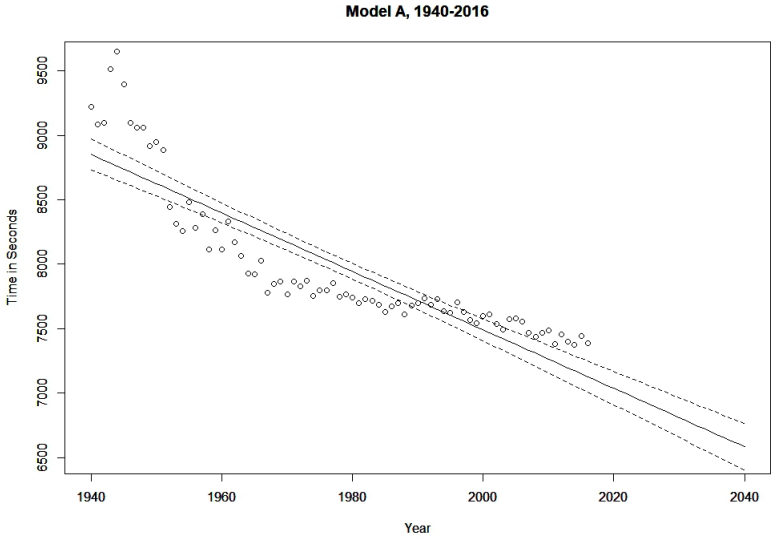

Figure 4: The Linear Model plotted with the data and a 95% confidence interval

To compensate for this, a more flexible model will be derived, in which the data can be split up into different parts. This can be done using cubic splines, a method in which we estimate a third degree polynomial through each of the data parts, with some continuity conditions:

- The composition of the polynomials together is twice differentiable,
- The first and second derivative of the composition are continuous.

The main idea of a cubic spline is to define basis functions. Then, it is assumed that the true function $f(x)$ can be written as a linear combination of basis functions:

$$f(x) = \sum_{m=1}^{M} \beta_m \cdot h_m(x),$$

where $h_m(X)$ denote the basis functions and M the total number of basis functions. In the next section, the basis functions for the cubic spline model will be derived.

4.1 Piecewise Polynomials

First of all, the data has to be divided into different parts. It seems reasonable to divide the dataset into three different parts, since dividing into more parts leads to too small parts. Hence, two nodes have to be chosen. These nodes will be referred to as ξ_1 and ξ_2. It can be seen that a piecewise

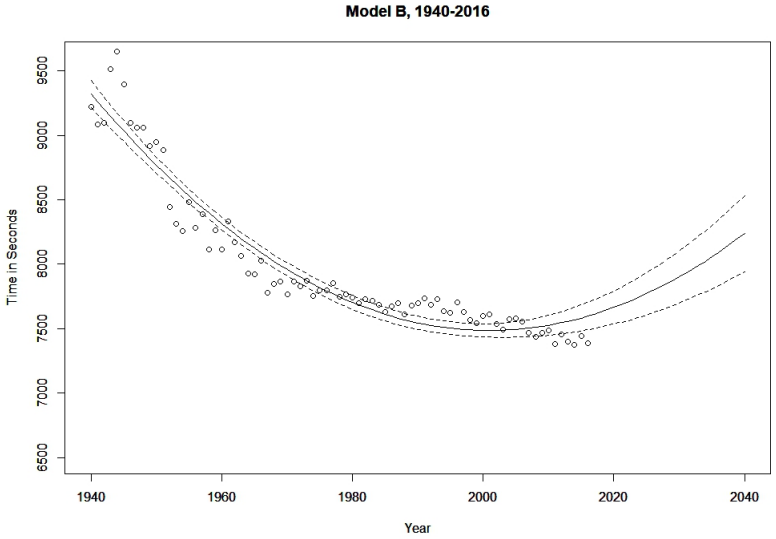

Figure 5: The Quadratic Model plotted with the data and a 95% confidence interval

constant function on the whole domain can be defined using the following basis functions:

$$h_1(x) = I(x < \xi_1), \quad h_2(x) = I(\xi_1 \leq x < \xi_2), \quad h_3(x) = I(\xi_2 \leq x).$$

Estimation of the function $f(x) = \sum_{m=1}^{3} \beta_m \cdot h_m(x)$ then amounts to $\hat{\beta}_m = \bar{y}_m$, the mean of y_n's in the mth region.
To estimate a piecewise linear function, three additional basis functions are needed, namely:

$$h_{m+3}(x) = h_m(x) \cdot x \quad \text{for } m = 1, 2, 3.$$

However, a continuous fit is preferred, hence some constraints on the parameters have to be imposed. For example, $\lim_{x \to \xi_1^+} f(x) = \lim_{x \to \xi_1^-} f(x)$ implies that $\beta_1 + \xi_1\beta_4 = \beta_2 + \xi_1\beta_5$. We can redefine the basis functions in such a way that these properties are automatically satisfied:

$$h_1(x) = 1, \quad h_2(x) = x, \quad h_3(x) = (x - \xi_1)_+, \quad h_4(x) = (x - \xi_2)_+,$$

where $(t)_+ = I(t > 0) \cdot t$.
The order of the polynomials can be increased further. The name cubic spline already implies that we would like to have third order polynomials. However, when increasing the order of the polynomials, it is preferred to also increase the continuity constraints. In figure 7 one can see cubic spline

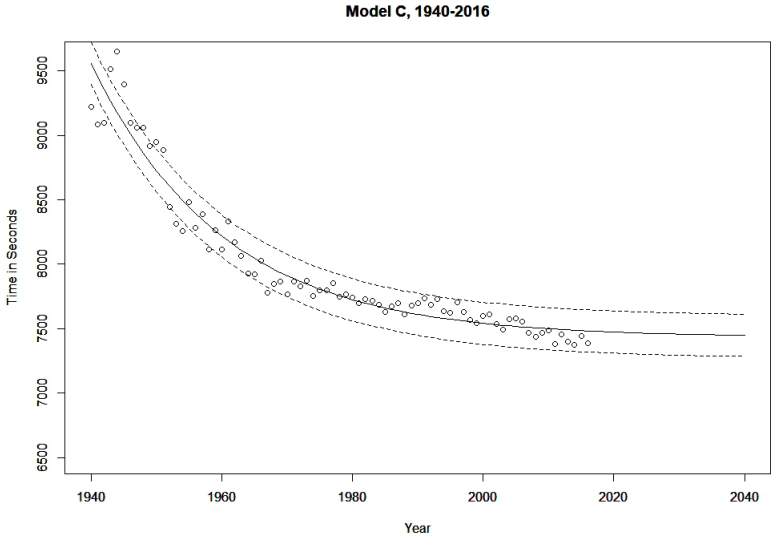

Figure 6: The Exponential Model plotted with the data and a 95% confidence interval

fits with increasing order of continuity. Also, the (in this case known) real data function is plotted.
One can see, that with increasing the order of continuity, a better estimate for the real data function is obtained. Hence, second order continuity constraints are imposed on the model. It can be seen that there are now 6 basis functions needed to be able to create any cubic polynomial on the whole interval.

$$\begin{aligned} h_1(x) &= 1, \quad h_3(x) = x^2, \quad h_5(x) = (x-\xi_1)_+^3, \\ h_2(x) &= x, \quad h_4(x) = x^3, \quad h_6(x) = (x-\xi_2)_+^3. \end{aligned} \tag{1}$$

Using the following basis functions given by (1), any cubic polynomial can be created. Furthermore, it can be checked that these basis functions satisfy the second order continuity properties:

$$\begin{aligned} \frac{\mathrm{d}^2}{\mathrm{d}\,x^2}\left[\sum_{m=1}^{6}\beta_m \cdot h_m(x)\right] &= \sum_{m=1}^{6}\beta_m \cdot \frac{\mathrm{d}^2}{\mathrm{d}\,x^2}\left[h_m(x)\right] \\ &= 2\beta_3 + 6\beta_4 x + 6\beta_5(x-\xi_1)_+ + 6\beta_6(x-\xi_2)_+, \end{aligned}$$

which is the sum of continuous functions, hence is continuous. Hence, these basis functions can be used to create a cubic spline model. This will be done in the next section.

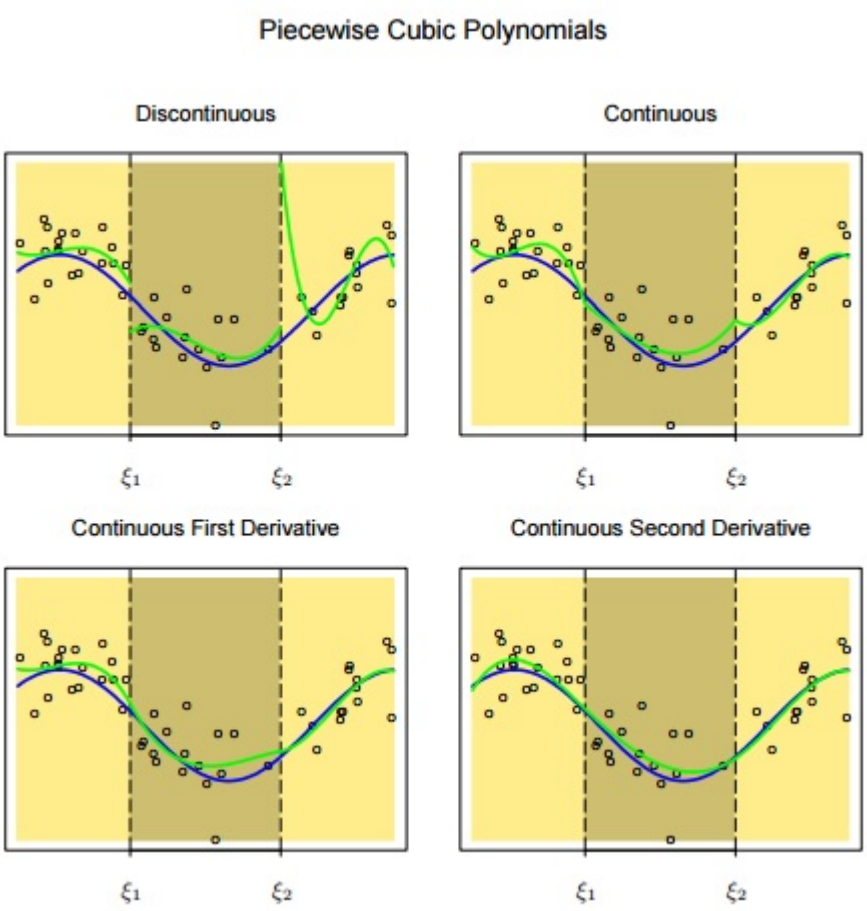

Figure 7: The cubic spline model with increasing order of continuity[4]

4.2 Non-penalised Cubic Spline Model

Given the basis functions (1), we would like to find optimal parameters $\hat{\boldsymbol{\beta}}$ such that the difference between the regression function $\hat{f}(x) = \sum_{m=1}^{6} \hat{\boldsymbol{\beta}}_m \cdot h_m(x)$ and the data is minimized. It is known that:

$$y_n = f(x_n) + \epsilon_n,$$

where ϵ_n is the difference between the true function $f(x)$ and the dataset $\boldsymbol{y}$. For convenience, the problem is rescaled such that the normalised year is in the closed interval between 0 and 1. Now, we construct the design matrix $\boldsymbol{X}$, where

$$\boldsymbol{x}_{n,m} = h_m(y_n)$$

Then, the problem can be written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}.$$

Hence, the problem reduces to a regular least squares problem, where minimizing

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2$$

yields the solution. It is known that the solution to this problem is given by:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{2}$$

A non-penalised cubic spline model was fitted to the data, with two equidistant nodes (i.e 1966 and 1992). The program used can be seen in B.4. It was found that:

$$\hat{\boldsymbol{\beta}} = \begin{pmatrix} 9419.317 \\ -3882.663 \\ -7985.853 \\ 19163.793 \\ -35331.930 \\ 27993.607 \end{pmatrix}.$$

Then, the estimate $\hat{f}(x)$ of the true function $f(x)$ can be expressed as follows:

$$\hat{f}(x) = \sum_{m=1}^{6} \hat{\beta}_m \cdot h_m(x),$$

The resulting cubic spline was plotted in Figure 8. As for the maximum

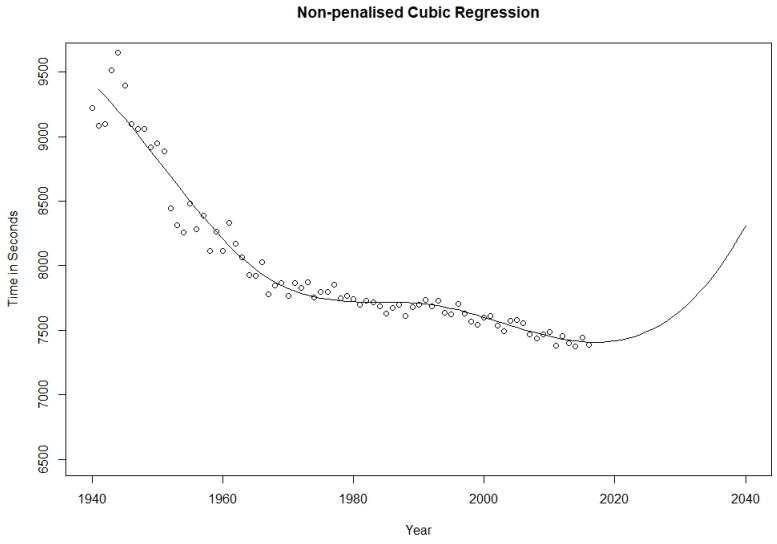

Figure 8: The estimated cubic spline function $\hat{f}(x)$ plotted with the data

likelihood models, we would like to find a confidence interval for the model. This can be done using the delta method. However, the variance-covariance

matrix for the estimator $\hat{\boldsymbol{\beta}}$ cannot be derived from the Fischer information matrix. Hence the variance of the least squares estimator was computed:

$$\begin{aligned}
\mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right) &= \mathrm{Var}\left(\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}\right) \\
&= \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\mathrm{Var}\left(\boldsymbol{y}\right)\left[\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\right]^T \\
&= \mathrm{Var}\left(\boldsymbol{y}\right)\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1} \\
&= \mathrm{Var}\left(\boldsymbol{y}\right)\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}.
\end{aligned}$$

A 95% confidence interval was derived using the delta method. The model with confidence interval was plotted in figure 9. As can be seen, the model has a good fit, however when predicting the future, it is very uncertain.

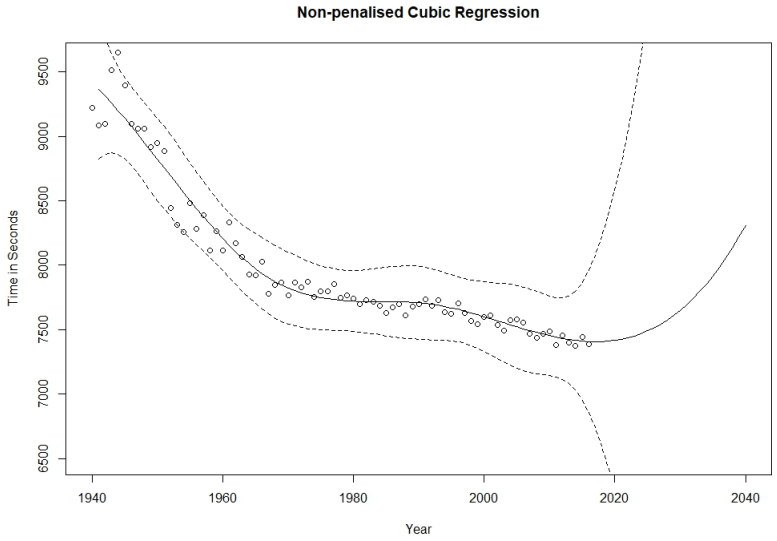

Figure 9: The estimated cubic spline function $\hat{f}(x)$ plotted with the data and 95% confidence interval

4.3 Penalised Cubic Spline

Since overestimation of the data can be a problem when using cubic spline, it is preferable to be able to control the smoothness of the resulting regression function. Therefore, a wiggliness penalty can be included in the cubic spline model. Rather than fitting the model by minimizing

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2$$

it could be done by minimizing,

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2 + \lambda \int_0^1 \left(f''(x)\right)^2 dx, \quad \lambda \in \mathbb{R}. \tag{3}$$

By controlling the smoothness coefficient λ, it is possible to control the trade off between the model fit and the model smoothness. Note that $\lambda = 0$ results in a non-penalised cubic spline, and $\lambda \to \infty$ leads to a straight line estimate. Since $f(x)$ is linear in β_i, the penalty can be written as a quadratic form of $\boldsymbol{\beta}$

$$\int_0^1 \left(f''(x)\right)^2 dx = \boldsymbol{\beta}^T \boldsymbol{S} \boldsymbol{\beta},$$

where $\boldsymbol{S}$ is some coefficient matrix. The computation of this matrix can be found in appendix C. Note that (3) can now be rewritten as

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2 + \lambda \boldsymbol{\beta}^T \boldsymbol{S} \boldsymbol{\beta}.$$

Using a slight extension of (2), the solution is given by:

$$\hat{\boldsymbol{\beta}}_\lambda = (\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{S})^{-1} \boldsymbol{X}^T \boldsymbol{y}.$$

4.4 Choosing the Smoothness Parameter

Since $\lambda \in \mathbb{R}$, an infinite amount of models has been created in the previous section. If λ is chosen too high, the data will be over smoothed. On the other hand, if λ is chosen too low, the data will be under smoothed. In both cases, the estimation $\hat{f}(x)$ will not be close to the true function $f(x)$. Hence, some criterion for λ has to be derived. The most natural choice would be to minimize

$$M(\lambda) = \frac{1}{N} \sum_{n=0}^{N} \left(\hat{f}_\lambda(x_n) - f(x_n)\right)^2.$$

However, the true function $f(x)$ is not known, hence this cannot be used directly.
We define the ordinary cross validation score,

$$\nu_o(\lambda) = \frac{1}{N} \sum_{n=0}^{N} \left(\hat{f}_\lambda^{[-n]}(x_n) - y_n\right)^2,$$

where $\hat{f}_\lambda^{[-n]}(x_n)$ denotes the model fitted to all data except for y_n.
Then, since $y_n = f(x_n) + \epsilon_n$:

$$\begin{aligned}
\nu_o(\lambda) &= \frac{1}{N} \sum_{n=0}^{N} \left(\hat{f}_\lambda^{[-n]}(x_n) - f(x_n) - \epsilon_n\right)^2 \\
&= \frac{1}{N} \sum_{n=0}^{N} \left(\hat{f}_\lambda^{[-n]}(x_n) - f(x_n)\right)^2 - \left(\hat{f}_\lambda^{[-n]}(x_n) - f(x_n)\right) \epsilon_n + \epsilon_n^2.
\end{aligned}$$

Since $\hat{f}_\lambda^{[-n]}(x_n)$ is independent of ϵ_n, and $\mathbb{E}(\epsilon_n) = 0$, it can be seen that

$$\mathbb{E}(\nu_o(\lambda)) = \frac{1}{N}\mathbb{E}\left(\sum_{n=0}^{N}\left(\hat{f}_\lambda^{[-n]}(x_n) - f(x_n)\right)^2\right) + \sigma^2.$$

Since $\hat{f}_\lambda^{[-n]}(x_n) \approx \hat{f}_\lambda(x_n)$, we have that

$$\mathbb{E}(\nu_o(\lambda)) \approx \mathbb{E}(M(\lambda)) + \sigma^2.$$

Hence, choosing λ minimizing $\nu_o(\lambda)$ is a reasonable approach. However, calculating $\nu_o(\lambda)$ by leaving out one data point at a time, is very inefficient. It can be shown that

$$\nu_o(\lambda) = \frac{1}{N}\sum_{n=0}^{N}\frac{\left(y_n - \hat{f}_\lambda(x_n)\right)^2}{(1 - A_{n,n})^2},$$

where A is the influence matrix. In our particular case:

$$A = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S})^{-1}\boldsymbol{X}^T$$

. In practice, the weights $1 - A_{n,n}$ are replaced by the mean weight, $\mathrm{tr}(I - A)/N$. Then the generalised cross validation score can be derived

$$\nu_g(\lambda) = \frac{N\sum_{n=0}^{N}\left(y_n - \hat{f}_\lambda(x_n)\right)^2}{(\mathrm{tr}(I - A))^2}.$$

Now, it is possible to plot the GCV score versus different values of λ. This plot can be found in Figure 10. It was found that the most optimal value is $\lambda = 0.0001341543$. The resulting model was plotted in figure 11. Similar to the non-penalised model, we would like to find the 95% confidence interval for the model. Using similar computations, it can be shown that:

$$\mathrm{Var}\left(\hat{\boldsymbol{\beta}}_\lambda\right) = \mathrm{Var}\left(\boldsymbol{y}\right)\left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S}\right)^{-1}\boldsymbol{X}^T\boldsymbol{X}\left[\left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S}\right)^{-1}\right]^T.$$

By applying the delta method, a 95% confidence interval was derived. The penalised cubic spline model with 95% confidence interval was plotted in Figure 12. It can be seen that the penalised model is already a bit more certain than the regular cubic spline model.

4.5 Monotonic Cubic Splines

Another restriction one can make, is for the regression function to be monotonic. Due to time constraints, instead of deriving new basis functions that satisfy the new constraints, the R library mgcv[6] was used. The program

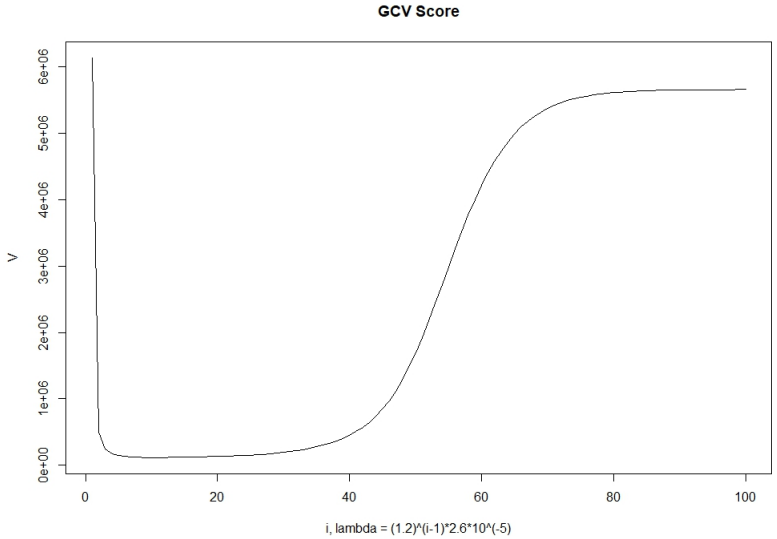

Figure 10: The Generalised Cross Validation score plotted versus i where $\lambda = 1.2^{i-1} \cdot 2.6 \cdot 10^{-5}$

used is heavily based on an example given in the documentation of the pcls function. The program can be found in B.5. The resulting monotonic cubic spline model can be seen in Figure 13. Unfortunately, this program does not give the variance of the model, and hence no confidence interval could be derived. What can be observed however, is that the model actually nicely follows the dataset, and is not too wiggly.

5 Residual Analysis

In order to analyse the models, residuals of the models defined by

$$r_n = (y_n - f(x_n))^2$$

were computed and plotted. It was found that particularly the time set in 1943 is a statistical outlier. In all of the models this data point has the biggest residual. Furthermore, the total residuals defined by

$$R = \sum_{n=0}^{N} r_n$$

were compared for the different models.

As can be seen, the cubic spline model has the least residual. However, we have seen that it is very vulnerable. Also, the attempt to make the model

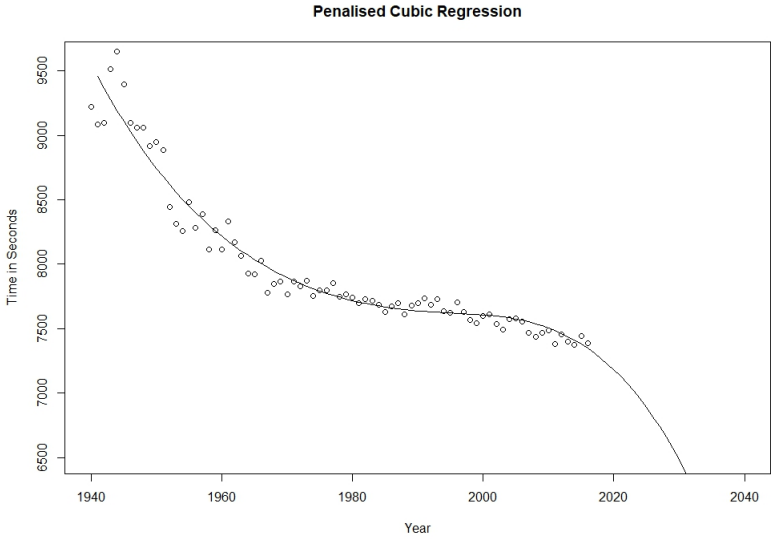

Figure 11: The estimated penalised cubic spline function $\hat{f}_\lambda(x)$ plotted with the data for $\lambda = 0.0001341543$

Model	Residual value
Cubic Spline	1101344
Monotonic Cubic Spline	1154319
Penalised Cubic Spline	1325142
Exponential	1402679
Quadratic	2006032
Linear	5663148

Table 1: Residuals for the different models

less vulnerable did not really succeed. The resulting penalised cubic spline was still quite unreliable, and also the residual has increased by quite a margin. From the maximum likelihood models, the exponential has by far the least residual. Also, it gave the best prediction for the future. The plots of the residuals can be found in Figures 14, 15, 16, 17, 18, and 19 .

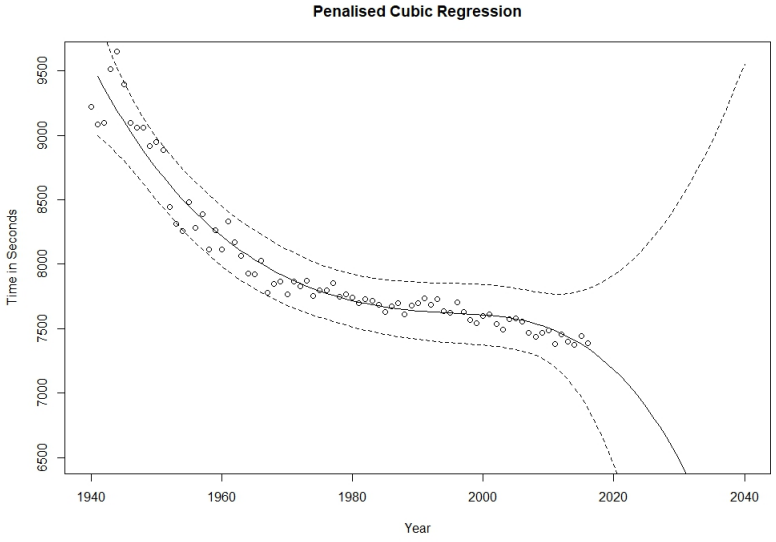

Figure 12: The estimated penalised cubic spline function $\hat{f}_\lambda(x)$ plotted with the data and 95% confidence interval for $\lambda = 0.0001341543$

6 Conclusion

Six different models for the yearly best times in past and future years were derived. It was found that some models had a better fit than others. The best fitted model was the non-penalised cubic spline model, however, this model showed signs of an over fit. The second best was the monotonic cubic spline model, which also showed to follow the linear trend of the last couple of years nicely.

From the maximum likelihood models derived in section 2, the exponential model was found to be the best fit. However, it was heavily restricted due to the assumption of only one trend through-out the whole dataset. Still an eventual limit for the mean yearly best time was derived and found to be 2:03:53. A time of 1:58:28 would still fall within the eventual 95% confidence interval for the data. If we also consider the deviation in the model itself, it can be found that the mean yearly best time will converge to a time between 1:58:07 and 2:09:39 with 95% certainty. Considering both lower bounds of the 95% confidence intervals, we can find a time of 1:53:43. Both the linear and quadratic models were found to be rather useless for prediction of future times.

From the cubic spline models, the regular and penalised models were found to be very uncertain, especially when predicting future times. The monotonic model seemed to be a lot more stable, however this cannot be checked since the variance of the model was not available.

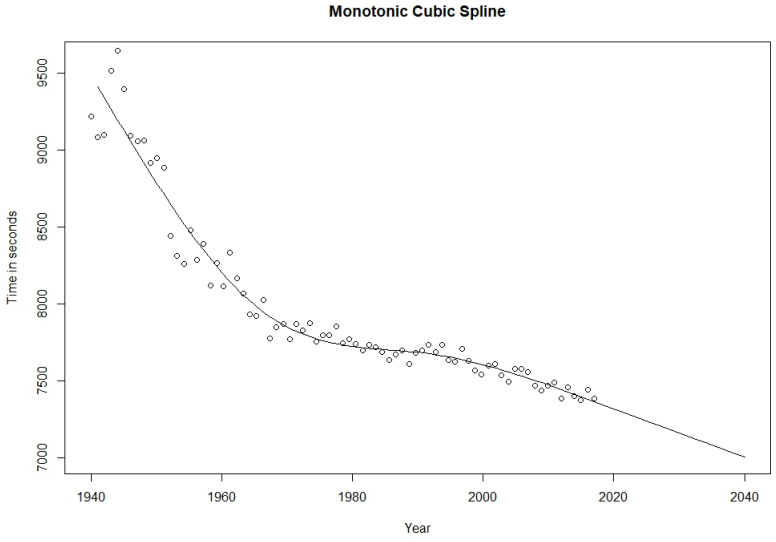

Figure 13: The estimated penalised cubic spline function $\hat{f}_\lambda(x)$ plotted with the data

References

[1] David C Blest. Lower bounds for athletic performance. *The Statistician*, 45:243–253, 1996.

[2] John HJ Einmahl and Jan R Magnus. Records in athletics through extreme-value theory. *Journal of the American Statistical Association*, 103(484):1382–1391, 2008.

[3] Thomas Shelburne Ferguson. *A course in large sample theory*, volume 49. Chapman & Hall London, 1996.

[4] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[5] HJ Grubb. Models for comparing athletic performances. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(3):509–521, 1998.

[6] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[7] Michael E Robinson and Jonathan A Tawn. Statistics for exceptional athletics records. *Applied Statistics*, pages 499–511, 1995.

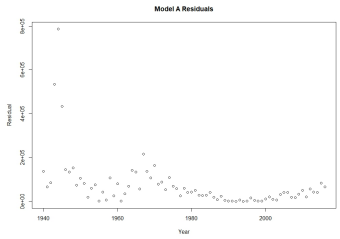

Figure 14: Linear Model Residuals

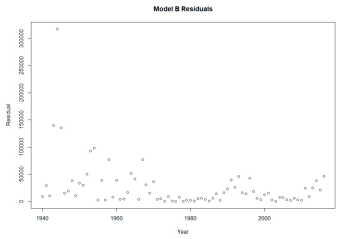

Figure 15: Quadratic Model Residuals

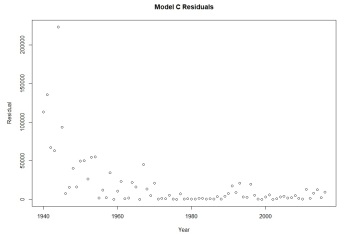

Figure 16: Exponential Model Residuals

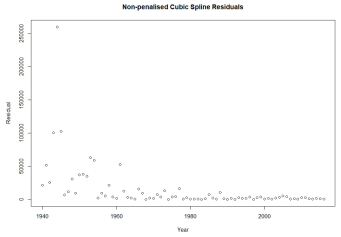

Figure 17: Cubic Spline Model Residuals

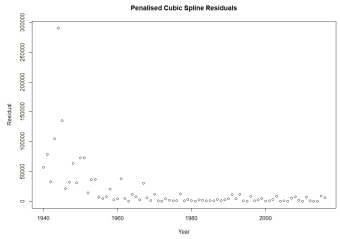

Figure 18: Penalised Cubic Spline Model Residuals

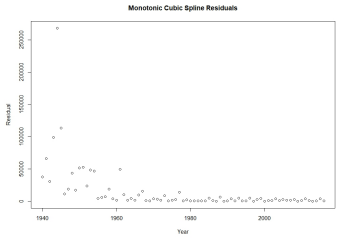

Figure 19: Monotonic Cubic Spline Model Residuals

[8] Richard L Smith. Forecasting records by maximum likelihood. *Journal of the American Statistical Association*, 83(402):331–338, 1988.

[9] RL Smith and JE Miller. A non-gaussian state space model and application to prediction of records. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 79–88, 1986.

A Dataset

Year	**Time**	**Name**	**Date**	**Place**
2016	2:03:05	Eliud Kipchoge (KEN)	24 Apr	London ENG
2015	2:04:00	Eliud Kipchoge (KEN)	27 Sep	Berlin GER
2014	2:02:56	Dennis Kipruto Kimetto (KEN)	28 Sep	Berlin GER

2013	2:03:22	Wilson Kipsang Kiprotich (KEN)	29 Sep	Berlin GER
2012	2:04:15	Geoffrey Mutai Kiprono (KEN)	30 Sep	Berlin GER
2011	2:03:02	Geoffrey Mutai Kiprono (KEN)	19 Apr	Boston USA
2010	2:04:48	Patrick Makau Musyoki (KEN)	11 Apr	Rotterdam NED
2009	2:04:26	Duncan Kibet (KEN)	05 Apr	Rotterdam NED
2008	2:03:58	Haile Gebreselasie (ETH)	28 Sep	Berlin GER
2007	2:04:26	Haile Gebreselasie (ETH)	30 Sep	Berlin GER
2006	2:05:56	Haile Gebreselasie (ETH)	24 Sep	Berlin GER
2005	2:06:19	Haile Gebreselasie (ETH)	16 Oct	Amsterdam NED
2004	2:06:16	Evans Rutto (KEN)	10 Oct	Chicago USA
2003	2:04:55	Paul Tergat (KEN)	28 Sep	Berlin GER
2002	2:05:37	Khalid Khannouchi (MAR)	14 Apr	London ENG
2001	2:06:50	Josephat Kiprono (KEN)	22 Apr	Rotterdam NED
2000	2:06:36	Antonio Pinto (POR)	16 Apr	London ENG
1999	2:05:42	Khalid Khannouchi (MAR)	24 Oct	Chicago USA
1998	2:06:05	Ronaldo daCosta (BRA)	20 Sep	Berlin GER
1997	2:07:10	Khalid Khannouchi (MAR)	19 Oct	Chicago USA
1996	2:08:25	Martin Fiz (ESP)	24 Mar	Kyong-Ju KOR
1995	2:07:02	Sammy Lelei (KEN)	24 Sep	Berlin GER
1994	2:07:15	Cosmas Ndeti (KEN)	18 Apr	Boston USA
1993	2:08:51	Dionicio Ceron (MEX)	05 Dec	Fukuoka JPN
1992	2:08:07	David Tsebe (RSA)	27 Sep	Berlin GER
1991	2:08:53	Koichi Morishita (JPN)	03 Feb	Beppu JPN
1990	2:08:16	Steve Moneghetti (AUS)	30 Sep	Berlin GER
1989	2:08:01	Juma Ikangaa (TAN)	05 Nov	New York USA
1988	2:06:50	Belayneh Dinsamo (ETH)	17 Apr	Rotterdam NED
1987	2:08:18	Takeyuki Nakayama (JPN)	06 Dec	Fukuoka JPN
1986	2:07:51	Rob deCastella (AUS)	21 Apr	Boston USA
1985	2:07:12	Carlos Lopes (POR)	20 Apr	Rotterdam NED
1984	2:08:05	Steve Jones (WAL)	21 Oct	Chicago USA
1983	2:08:37	Rob deCastella (AUS)	09 Apr	Rotterdam NED
1982	2:08:52	Alberto Salazar (USA)	19 Apr	Boston USA
1981	2:08:18	Rob deCastella (AUS)	06 Dec	Fukuoka JPN
1980	2:09:01	Gerard Nijboer (NED)	26 Apr	Amsterdam NED
1979	2:09:28	Bill Rodgers (USA)	16 Apr	Boston USA
1978	2:09:05	Shigeru So (JPN)	05 Feb	Beppu JPN
1977	2:10:55	Bill Rodgers (USA)	04 Dec	Fukuoka JPN
1976	2:09:55	Waldemar Cierpinski (GER)	31 Jul	Montreal CAN
1975	2:09:56	Bill Rodgers (USA)	21 Apr	Boston USA
1974	2:09:12	Ian Thompson (ENG)	31 Jan	Christchurch NZL
1973	2:11:12	John Farrington (AUS)	14 Jul	Sydney AUS

1972	2:10:30	Frank Shorter (USA)	03 Dec	Fukuoka JPN
1971	2:11:08	Derek Clayton (AUS)	25 Sep	Hobart AUS
1970	2:09:28	Ron Hill (ENG)	23 Jul	Edinburgh SCO
1969	2:11:07	William Adcocks (ENG)	06 Apr	Athens GRE
1968	2:10:47	William Adcocks (ENG)	08 Dec	Fukuoka JPN
1967	2:09:36	Derek Clayton (AUS)	03 Dec	Fukuoka JPN
1966	2:13:45	Alastair Wood (SCO)	09 Jul	Forres SCO
1965	2:12:00	Morio Shigematsu (JPN)	12 Jun	Chiswick ENG
1964	2:12:11	Abebe Bikila (ETH)	21 Oct	Tokyo JPN
1963	2:14:28	Leonard Edelen (USA)	15 Jun	Chiswick ENG
1962	2:16:09	Mang-Hyang Yu (PRK)	24 Oct	Pyongyang PRK
1961	2:18:54	Takayuki Nakao (JPN)	21 Mar	Nagoya JPN
1960	2:15:16	Abebe Bikila (ETH)	10 Sep	Rome ITA
1959	2:17:45	Sergey Popov (RUS)	11 Oct	Kosice SVK
1958	2:15:17	Sergey Popov (RUS)	24 Aug	Stockholm SWE
1957	2:19:50	Sergey Popov (RUS)	01 Sep	Moscow RUS
1956	2:18:04	Paavo Kotila (FIN)	12 Aug	Pieksamaki FIN
1955	2:21:21	Veikko Karvonen (FIN)	04 Sep	Copenhagen DEN
1954	2:17:39	James Henry Peters (ENG)	26 Jun	Chiswick ENG
1953	2:18:34	James Henry Peters (ENG)	04 Oct	Turku FIN
1952	2:20:42	James Henry Peters (ENG)	14 Jun	Windsor ENG
1951	2:28:07	Veikko Karvonen (FIN)	03 Aug	Tampere FIN
1950	2:29:09	Fyeodosiy Vanin (SOV)	15 Jul	Moscow RUS
1949	2:28:39	Salomon Knnen (FIN)	02 Oct	Turku FIN
1948	2:31:02	Mikko Hietanen (FIN)	07 Sep	Stockholm SWE
1947	2:30:58	Mikko Hietanen (FIN)	23 Aug	Loughborough ENG
1946	2:31:37	Mikko Hietanen (FIN)	11 Jul	Vuoksenniska FIN
1945	2:36:37	Sven Hakansson (SWE)	28 Oct	Gteborg SWE
1944	2:40:48	Charles Robbins (USA)	12 Nov	Yonkers USA
1943	2:38:35	Grard Cot (CAN)	07 Nov	Yonkers USA
1942	2:31:38	Zaiten Kimoto (JPN)	02 Nov	Tokyo JPN
1941	2:31:27	Leslie Pawson (USA)	30 May	Salisbury Beach USA
1940	2:33:42	Shoichiro Takenaka (JPN)	01 Nov	Tokyo JPN

B Programms

B.1 Dataverwerking

```
1 library(readr)
2 YBMT <- read_csv("C:/Users/thobe/Documents/
    Universiteit/Bachelor_Thesis/Thesis/
```

```
    YearlyBestMarathonTimes1940_2016.csv",
3 col_types = cols(Time = col_character()))
4
5 #Function to convert the time in hh:mm:ss to time in
    seconds
6 toSeconds <- function(x){
7 if (!is.character(x)) stop("x must be a character
    string of the form H:M:S")
8 if (length(x)<=0)return(x)
9
10 unlist(
11 lapply(x,
12 function(i){
13 i <- as.numeric(strsplit(i,':',fixed=TRUE)[[1]])
14 if (length(i) == 3)
15 i[1]*3600 + i[2]*60 + i[3]
16 else if (length(i) == 2)
17 i[1]*60 + i[2]
18 else if (length(i) == 1)
19 i[1]
20 }
21 )
22 )
23 }
24
25 #Create new columns: Time in seconds and speed in kmph
26 YBMT$Seconds <- toSeconds(YBMT$Time)
27 YBMT$Speed <- (42195/YBMT$Seconds)*3.6
28
29 #Invert dataset to have the first year on the first
    row
30 YBMT <- YBMT[ nrow(YBMT):1, ]
31
32 #Create new column with the worldrecord time in
    seconds per year
33 YBMT$WRSeconds <- YBMT$Seconds
34 y <- length(YBMT$WRSeconds)
35 YBMT$WR <- integer(length = y)
36 YBMT$WR[1] <- 1
37 for (i in 1:(y-1)){
38 YBMT$WR[i+1] <- if (YBMT$Seconds[i+1] > YBMT$WRSeconds
    [i]) 0 else 1
39 YBMT$WRSeconds[i+1] <- if (YBMT$WR[i+1] == 0) YBMT$
    WRSeconds[i] else YBMT$WRSeconds[i+1]
```

```
}

#Create column with normalized year (1940 taken to be the zeroth year)
YBMT$NormYear <- YBMT$Year - 1940
```

B.2 Log-likelihood Functions

```
#Define the different drift functions, Linear, Quadratic, and Exponential
Lineardrift <- function(beta_0, beta_1, n){
beta_0 - n*beta_1
}
Quadraticdrift <- function(beta_0, beta_1, beta_2, n){
beta_0 - n*beta_1 + beta_2*n*n/2
}
Exponentialdecay <- function(beta_0, beta_1, beta_2, n){
beta_0 - beta_1*(1-(1-beta_2)^n)/beta_2
}

#Create loglikelihood functions for the three different models
NormalLinearLoglikelihood <- function(par, y){
term <- 0
z <- length(y)
for (i in 1:z){
term <- term + (((y[i] - Lineardrift(par[1], par[2], (i-1)))^2)/(2*par[3]^2))
}
-(-z*log((sqrt(2*pi*par[3]^2))) - term)
}

NormalQuadraticLoglikelihood <- function(par, y){
term <- 0
z <- length(y)
for (i in 1:z){
term <- term + (((y[i] - Quadraticdrift(par[1], par[2], par[3], (i-1)))^2)/(2*par[4]^2))
}
-(-z*log(sqrt(2*pi*par[4]^2)) - term)
}

NormalExponentialLoglikelihood <- function(par, sigma,
```

```
    y){
32 term <- 0
33 z <- length(y)
34 for (i in 1:z){
35 term <- term + (((y[i] - Exponentialdecay(par[1], par
       [2], par[3], (i-1)))^2)/(2*sigma^2))
36 }
37 -(-z*log(sqrt(2*pi*sigma^2)) - term)
38 }
```

B.3 Maximum Likelihood Estimation

```
1  #Define window limits for plots
2  x <- c(1940,2040)
3  y <- c(6500,9600)
4
5  #Find the maximum likelihood estimates for Linear
       model
6  NormalLinear <- optim(par = c(8800, 22, 180),
       NormalLinearLoglikelihood, y = YBMT$Seconds,
       hessian = T)
7  #Define the function of the fitted model
8  NormalLinearModel <- function(x){
9  NormalLinear$par[1] - NormalLinear$par[2]*(x-1940)
10 }
11 #Plot the fitted model with the data
12 plot(function(x) NormalLinearModel(x), xlim = x, ylim
       = y, col = "Black", xlab = "Year", ylab = "Time in
       Seconds", main = "Model A, 1940-2016")
13 par(new=TRUE)
14 plot(YBMT$Year, YBMT$Seconds, xlim = x,  ylim = y, col
       = "Black", xlab = "", ylab = "")
15
16 #Compute variance of the Linear model
17 NormalLinearErrors <- (solve(NormalLinear$hessian))
18 NormalLinearVariance <- function(x){
19 NormalLinearErrors[1,1] + NormalLinearErrors[2,2]*(x
       -1940)^2 - 2*(x-1940)*NormalLinearErrors[1,2]}
20 par(new=TRUE)
21 #Plot the 95% confidence interval with the model and
       the data
22 plot(function(x) NormalLinearModel(x) + 1.96*sqrt(
       NormalLinearVariance(x)), xlim = x, ylim = y, col =
       "Black", xlab = "Year", ylab = "Time in Seconds",
```

```
    main = "Model A, 1940-2016", lty = 2)
23 par(new=TRUE)
24 plot(function(x) NormalLinearModel(x) - 1.96*sqrt(
    NormalLinearVariance(x)), xlim = x, ylim = y, col =
    "Black", xlab = "Year", ylab = "Time in Seconds",
    main = "Model A, 1940-2016", lty = 2)
25
26 #Compute residuals
27 NormalLinearResiduals <- (YBMT$Seconds -
    NormalLinearModel(YBMT$Year))^2
28 NormalLinearResidual <- sum(NormalLinearResiduals)
29 #Plot residuals versus the Year
30 plot(YBMT$Year, NormalLinearResiduals, xlab = "Year",
    ylab = "Residual", main = "Model A Residuals")
31
32 #Find maximum likelihood estimates for the Quadratic
    model
33 NormalQuadratic <- optim(par = c(7500, 22, 22, 180),
    NormalQuadraticLoglikelihood, y = YBMT$Seconds,
    method = "L-BFGS-B", hessian = T)
34 #Define function for the fitted model
35 NormalQuadraticModel <- function(x){
36 NormalQuadratic$par[1] - NormalQuadratic$par[2]*(x
    -1940) + NormalQuadratic$par[3]*(x-1940)^2/2
37 }
38 #Plot fitted model with the data
39 plot(function(x) NormalQuadraticModel(x), xlim = x,
    ylim = y, col = "Black", xlab = "Year", ylab = "
    Time in Seconds", main = "Model B, 1940-2016")
40 par(new=TRUE)
41 plot(YBMT$Year, YBMT$Seconds, xlim = x,  ylim = y, col
    = "Black", xlab = "", ylab = "")
42
43 #Compute the variance of the Quadratic model
44 NormalQuadraticErrors <- (solve(NormalQuadratic$
    hessian))
45 NormalQuadraticVariance <- function(x){
46 NormalQuadraticErrors[1,1] + NormalQuadraticErrors
    [2,2]*(x-1940)^2 + ((x-1940)^4)/4*
    NormalQuadraticErrors[3,3]-2*(x-1940)*
    NormalQuadraticErrors[1,2]+(x-1940)^2*
    NormalQuadraticErrors[1,3]-((x-1940)^3)*
    NormalQuadraticErrors[2,3]}
47 #Plot 95% Confidence intervals with the data and the
```

```
    fitted model
par(new=TRUE)
plot(function(x) NormalQuadraticModel(x) + 1.96*sqrt(
    NormalQuadraticVariance(x)), xlim = x, ylim = y,
    col = "Black", xlab = "Year", ylab = "Time in
    Seconds", main = "Model B, 1940-2016", lty = 2)
par(new=TRUE)
plot(function(x) NormalQuadraticModel(x) - 1.96*sqrt(
    NormalQuadraticVariance(x)), xlim = x, ylim = y,
    col = "Black", xlab = "Year", ylab = "Time in
    Seconds", main = "Model B, 1940-2016", lty = 2)

#Compute residuals for the Quadratic model
NormalQuadraticResiduals <- (YBMT$Seconds -
    NormalQuadraticModel(YBMT$Year))^2
NormalQuadraticResidual <- sum(
    NormalQuadraticResiduals)
#Plot residual versus the year
plot(YBMT$Year, NormalQuadraticResiduals, xlab = "Year
    ", ylab = "Residual", main = "Model B Residuals")

#Find maximum likelihood estimates for the Exponential
     model
NormalExponential <- optim(par = c(8000, 1, 1),
    NormalExponentialLoglikelihood, sigma = 180, y =
    YBMT$Seconds, control = list(maxit=20000), hessian
    = T)
#Define function of the fitted model
NormalExponentialModel <- function(x) {
NormalExponential$par[1] - NormalExponential$par[2]*
    (1-(1-NormalExponential$par[3])^(x-1940))/
    NormalExponential$par[3]
}
#Plot fitted model with the data
plot(function(x) NormalExponentialModel(x), xlim = x,
    ylim = y, col = "Black", xlab = "Year", ylab = "
    Time in Seconds", main = "Model C, 1940-2016")
par(new=TRUE)
plot(YBMT$Year, YBMT$Seconds, xlim = x,  ylim = y, col
     = "Black", xlab = "", ylab = "")
#Find maximum likelihood estimate for the standard
    deviation
Sigma <- optimize(NormalExponentialLoglikelihood, par
    = c(NormalExponential$par[1], NormalExponential$par
```

```
    [2], NormalExponential$par[3]), y = YBMT$Seconds,
    interval = c(0, 10000))

#Compute variance for the Exponential model
beta <- NormalExponential$par
NormalExponentialErrors <- function(x){
if (length(x) != 1){
x <- x[-c(2:length(x))]
}
sol <- c(1,(-(1-(1-beta[3])^(x-1940))/beta[3]),(beta
    [2]*((1-(1-beta[3])^(x-1940))-beta[3]*(x-1940)*(1-
    beta[3])^(x-1940-1))/beta[3]^2))
as.vector(sqrt(sol%*%(solve(NormalExponential$hessian)
    )%*%sol))
}

#Plot 95% Confidence interval with the data and the
    model
par(new=TRUE)
plot(function(x) NormalExponentialModel(x) + 1.96*
    NormalExponentialErrors(x), xlim = x, ylim = y, col
     = "Black", xlab = "Year", ylab = "Time in Seconds"
    , main = "Model C, 1940-2016", lty = 2)
par(new=TRUE)
plot(function(x) NormalExponentialModel(x) - 1.96*
    NormalExponentialErrors(x), xlim = x, ylim = y, col
     = "Black", xlab = "Year", ylab = "Time in Seconds"
    , main = "Model C, 1940-2016", lty = 2)

#Compute residuals for the Exponential model
NormalExponentialResiduals <- (YBMT$Seconds -
    NormalExponentialModel(YBMT$Year))^2
NormalExponentialResidual <- sum(
    NormalExponentialResiduals)
#plot residuals versus the year
plot(YBMT$Year, NormalExponentialResiduals, xlab = "
    Year", ylab = "Residual", main = "Model C Residuals
    ")
```

B.4 (Penalised) Cubic Spline

```
#Define window limits for plots
x <- c(1940,2040)
y <- c(6500,9600)
```

```
4
5  #Redefine the problem to the interval [0,1]
6  xdata <- YBMT$NormYear/max(YBMT$NormYear)
7  p <- length(xdata)
8
9  #Define the six basis functions for cubic spline
       regression
10 h1 <- function(x,z){
11 x^0
12 }
13 h2 <- function(x,z){
14 x
15 }
16 h3 <- function(x,z){
17 x^2
18 }
19 h4 <- function(x,z){
20 x^3
21 }
22 h5 <- function(x,z){
23 sol <- (x-z[1])^3
24 if (sol < 0){
25 sol <- 0
26 }
27 sol
28 }
29 h6 <- function(x,z){
30 sol <- (x-z[2])^3
31 if (sol < 0){
32 sol <- 0
33 }
34 sol
35 }
36
37 #Function to define the matrix X using the basis
       functions
38 spl.X <- function(x,xk){
39 q <- length(xk)+4
40 n <- length(x)
41 X <- matrix(0,n,q)
42 for (i in 1:n){
43 X[i,1] <- h1(x[i],xk)
44 X[i,2] <- h2(x[i],xk)
45 X[i,3] <- h3(x[i],xk)
```

```
46 X[i,4] <- h4(x[i],xk)
47 X[i,5] <- h5(x[i],xk)
48 X[i,6] <- h6(x[i],xk)
49 }
50 X
51 }
52
53 #Choose knots
54 xk <- c(26/p,52/p)
55 #Define matrix X
56 X <- spl.X(xdata, xk)
57 #Compute beta
58 SplineBeta <- as.vector((solve(t(X)%*%X)%*%t(X)%*%YBMT
      $Seconds))
59 #Define array and matrix Xp for prediction
60 xp <- (1:100)/p
61 Xp <- spl.X(xp,xk)
62 #Plot data and cubic spline model
63 plot(YBMT$Year, YBMT$Seconds, xlim = x,  ylim = y, col
       = "Black", xlab = "Year", ylab = "Time in Seconds"
      , main = "Non-penalised Cubic Regression")
64 lines(xp*77+1940, Xp%*%SplineBeta, xlim = x, ylim = y)
65
66 #Compute Cov matrix for estimator
67 SplineError <- var(YBMT$Seconds)*solve(t(X)%*%X)
68 #apply delta method
69 SplineVariance <- function(x){
70 hoi <- c(h1(x, xk), h2(x, xk),h3(x, xk),h4(x, xk),h5(x
      , xk),h6(x, xk))
71 as.vector(hoi%*%SplineError%*%hoi)
72 }
73 #Find pointwise errors
74 SplineErrors <- 1:length(xp)
75 for (j in 1:length(xp)){
76 SplineErrors[j] <- sqrt(SplineVariance(xp[j]))
77 }
78 #Plot 95% Confidence interval
79 lines(xp*77+1940, Xp%*%SplineBeta + 1.96*SplineErrors,
       xlim = x, ylim = y, lty = 2)
80 lines(xp*77+1940, Xp%*%SplineBeta - 1.96*SplineErrors,
       xlim = x, ylim = y, lty = 2)
81
82 #Compute Residuals of the model
83 SplineResiduals <- (YBMT$Seconds - (Xp%*%SplineBeta)
```

```
    [1:77])^2
84 #Plot residuals versus the year
85 plot(YBMT$Year, SplineResiduals, xlab = "Year", ylab =
     "Residual", main = "Non-penalised Cubic Spline
    Residuals")
86 SplineResidual <- sum(SplineResiduals)
87
88 #Create function for the penalty matrix S
89 spl.S <- function(xk){
90 q <- length(xk)+4
91 S <- matrix(0,q,q)
92 S[3,3] <- 4
93 S[3,4] <- 6
94 S[3,5] <- 6*(1-xk[1])^2
95 S[3,6] <- 6*(1-xk[2])^2
96 S[4,4] <- 12
97 S[4,5] <- 6*(2-3*xk[1]+xk[1]^3)
98 S[4,6] <- 6*(2-3*xk[2]+xk[2]^3)
99 S[5,5] <- 12*(1-3*xk[1]+3*xk[1]^2-xk[1]^3)
100 S[5,6] <- 6*(2-xk[1]-3*xk[2]+6*xk[1]*xk[2]+xk[2]^3-3*
    xk[1]*xk[2]^2)
101 S[6,6] <- 12*(1-3*xk[2]+3*xk[2]^2-xk[2]^3)
102 S[4,3] <- S[3,4]
103 S[5,3] <- S[3,5]
104 S[6,3] <- S[3,6]
105 S[5,4] <- S[4,5]
106 S[6,4] <- S[4,6]
107 S[6,5] <- S[5,6]
108 S
109 }
110
111 #Create matrices X and S
112 X <- spl.X(xdata,xk)
113 S <- spl.S(xk)
114
115 #Find GCV Score for different values of lambda
116 lambda <- 2.6e-05
117 n <- length(data)
118 V <- 0
119 for (i in 1:100){
120 beta <- (solve(t(X)%*%X+lambda*S)%*%t(X)%*%YBMT$
    Seconds)
121 trA <- sum(diag(X%*%solve(t(X)%*%X+lambda*S)%*%t(X)))
122 rss <- sum((YBMT$Seconds - X%*%beta)^2)
```

```
123 V[i] <- n*rss/(n-trA)^2
124 lambda <- 1.2*lambda
125 }
126 #Plot GCV score versus lambda
127 plot(1:100, V, type = "l", main = "GCV Score", xlab =
      "i, lambda = (1.2)^(i-1)*2.6*10^(-5)")
128
129 #Find ideal lambda
130 i <- (1:100)[V==min(V)]
131 lambda <- 1.2^(i-1)*2.6e-05
132 PSplineBeta <- (solve(t(X)%*%X+lambda*S)%*%t(X)%*%YBMT
      $Seconds)
133 #plot fitted model for ideal lambda
134 plot(YBMT$Year, YBMT$Seconds, xlim = x, ylim = y, col
      = "Black", xlab = "Year", ylab = "Time in Seconds"
      , main = "Penalised Cubic Regression")
135 lines(xp*p+1940, Xp%*%PSplineBeta, xlim = x, ylim = y)
136
137 #Find cov matrix for estimator
138 PSplineError <- var(YBMT$Seconds)*solve(t(X)%*%X+
      lambda*S)%*%t(X)%*%X%*%t(solve(t(X)%*%X+lambda*S))
139 #Apply delta method
140 PSplineVariance <- function(x){
141 hoi <- c(h1(x, xk), h2(x, xk),h3(x, xk),h4(x, xk),h5(x
      , xk),h6(x, xk))
142 as.vector(hoi%*%PSplineError%*%hoi)
143 }
144 #Find pointwise errors
145 PSplineErrors <- 1:length(xp)
146 for (j in 1:length(xp)){
147 PSplineErrors[j] <- sqrt(PSplineVariance(xp[j]))
148 }
149 #Plot 95% Confidence interval
150 lines(xp*77+1940, Xp%*%PSplineBeta + 1.96*
      PSplineErrors, xlim = x, ylim = y, lty = 2)
151 lines(xp*77+1940, Xp%*%PSplineBeta - 1.96*
      PSplineErrors, xlim = x, ylim = y, lty = 2)
152
153 #Compute residuals for fitted model
154 PSplineResiduals <- (YBMT$Seconds - (Xp%*%PSplineBeta)
      [1:77])^2
155 #Plot Residuals
156 plot(YBMT$Year, PSplineResiduals, xlab = "Year", ylab
      = "Residual", main = "Penalised Cubic Spline
```

```
    Residuals")
157 PSplineResidual <- sum(PSplineResiduals)
```

B.5 Monotonic Cubic Spline

```
1  library(mgcv)
2  d <- YBMT$NormYear/max(YBMT$NormYear)
3  y <- -YBMT$Seconds
4  xp <- (1:100)/length(d)
5  plot(d*77+1940,-y, xlim = c(1940,2040), ylim = c(7000,
       9600), xlab = "Year", ylab = "Time in seconds",
     main = "Monotonic Cubic Spline")
6  dat <- data.frame(d=d,y=y)
7
8  f.ug <- gam(y~s(d,k=4,bs="cr"))
9  # Create Design matrix, constraints etc. for monotonic
       spline....
10 sm <- smoothCon(s(d,k=4,bs="cr"),dat,knots=NULL)[[1]]
11 Q <- mono.con(sm$xp);   # get constraints
12 G <- list(X=sm$X,C=matrix(0,0,0),sp=f.ug$sp,p=sm$xp,y=
     y,w=y*0+1)
13 G$Ain <- Q$A;G$bin <- Q$b;G$S <- sm$S;G$off <- 0
14
15 p <- pcls(G);  # fit spline (using s.p. from
     unconstrained fit)
16
17 #Get prediction vector
18 fv<- as.vector(-Predict.matrix(sm,data.frame(d=xp))%*%
     p)
19 #Plot predicted model
20 lines(xp*77 + 1940,fv,col=1)
21
22 #Compute Residuals
23 MSplineResiduals <- (YBMT$Seconds - fv[1:77])^2
24 #Plot Residuals
25 plot(YBMT$Year, MSplineResiduals, xlab = "Year", ylab
     = "Residual", main = "Monotonic Cubic Spline
     Residuals")
26 #Obtain total residual
27 MSplineResidual <- sum(MSplineResiduals)
```

C Derivation of the matrix $\boldsymbol{S}$

The matrix $\boldsymbol{S}$ is defined by:

$$\int_0^1 \left(f''(x)\right)^2 dx = \boldsymbol{\beta}^T \boldsymbol{S} \boldsymbol{\beta}.$$

Note that

$$\frac{\mathrm{d}^2}{\mathrm{d}\, x^2}\left[f(x)\right] = 2\beta_3 + 6\beta_4 x + 6\beta_5 (x-\xi_1)_+ + 6\beta_6 (x-\xi_2)_+.$$

Since there are no terms in β_1 and β_2, the first two rows and columns of the matrix $\boldsymbol{S}$ have all zero terms. For the other terms, it can be seen that some integrals have to be solved.

$$\boldsymbol{s}_{3,3} = \int_0^1 4dx = 4$$

$$\boldsymbol{s}_{3,4} = \int_0^1 12x dx = 6$$

$$\begin{aligned}
\boldsymbol{s}_{3,5} =& \int_0^1 12\left(x-\xi_1\right)_+ dx \\
=& 12 \int_{\xi_1}^1 (x-\xi_1) dx \\
=& 12 \int_0^{1-\xi_1} x dx = 6(1-\xi_1)^2
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{s}_{3,6} =& \int_0^1 12\left(x-\xi_2\right)_+ dx \\
=& 12 \int_{\xi_2}^1 (x-\xi_2) dx \\
=& 12 \int_0^{1-\xi_2} x dx = 6(1-\xi_2)^2
\end{aligned}$$

$$\boldsymbol{s}_{4,4} = \int_0^1 36x^2 dx = 12$$

$$\begin{aligned}
\boldsymbol{s}_{4,5} =& \int_0^1 36x(x-\xi_1)_+ dx \\
=& 36 \int_{\xi_1}^1 x(x-\xi_1) dx \\
=& 6(2 - 3\xi_1 + \xi_1^3)
\end{aligned}$$

$$
\begin{aligned}
\boldsymbol{s}_{4,6} &= \int_0^1 36x(x-\xi_2)_+ dx \\
&= 36 \int_{\xi_2}^1 x(x-\xi_2) dx \\
&= 6(2 - 3\xi_2 + \xi_2^3)
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{s}_{5,5} &= \int_0^1 36 \left((x-\xi_1)_+\right)^2 dx \\
&= 36 \int_{\xi_1}^1 (x-\xi_1)^2 dx \\
&= 12(1 - 3\xi_1 + 3\xi_1^2 - \xi_1^3)
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{s}_{5,6} &= \int_0^1 36 \left((x-\xi_1)_+\right) \left((x-\xi_2)_+\right) dx \\
&= 36 \int_{\xi_2}^1 (x-\xi_1)(x-\xi_2) dx \\
&= 6(2 - 3\xi_1 - 3\xi_2 + 6\xi_1\xi_2 + \xi_2^2 - 3\xi_1\xi_2^2)
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{s}_{6,6} &= \int_0^1 36 \left((x-\xi_2)_+\right)^2 dx \\
&= 36 \int_{\xi_2}^1 (x-\xi_2)^2 dx \\
&= 12(1 - 3\xi_2 + 3\xi_2^2 - \xi_2^3)
\end{aligned}
$$

Note that this matrix $\boldsymbol{S}$ is symmetric, hence the missing coefficients can be found. Filling in the two chosen nodes, the following matrix $\boldsymbol{S}$ is obtained:

$$
\boldsymbol{S} = \begin{bmatrix}
0 & 0 & 0.0000000 & 0.000000 & 0.000000 & 0.0000000 \\
0 & 0 & 0.0000000 & 0.000000 & 0.000000 & 0.0000000 \\
0 & 0 & 4.0000000 & 6.000000 & 2.632147 & 0.6324844 \\
0 & 0 & 6.0000000 & 12.000000 & 6.153071 & 1.6921011 \\
0 & 0 & 2.6321471 & 6.153071 & 3.486740 & 5.1033507 \\
0 & 0 & 0.6324844 & 1.692101 & 5.103351 & 0.4107042
\end{bmatrix}.
$$