UNIVERSITY OF COPENHAGEN
NIELS BOHR INSTITUTE

Exam 2022

Advanced Methods in Applied Statistics

Kimi Cardoso Kreilgaard (TWN176)

Submission Date
02.03.2022

I, Kimi Kreilgaard, expressly vow to uphold my scientific, academic and moral integrity by working individually on this exam and soliciting no direct external help or assistance.

Contents

1 Problem 1: Distributions and Best Fit Functions **1**
1.1 (1A)+(1B) . . . 1
1.2 Fitting the First Column . . . 3
1.3 Fitting the Second Column . . . 4
1.4 Fitting the Third Column . . . 5

2 Problem 2: Spherical Isotropy **6**
2.1 (2A) . . . 6
2.2 (2B) . . . 8

3 Problem 3: Best Fit with Given Function **9**

4 Problem 4: Classification of No-Shows **11**
4.1 Exploring the Data . . . 11
4.2 Hyperparameter Optimization . . . 12
4.2.1 Constructing the Classifier . . . 13
4.3 (4A) . . . 13
4.4 (4B) . . . 15
4.5 (4C) . . . 15

5 Problem 5: Mars Data and Splines **16**
5.1 Problem (5A) . . . 16
5.2 Problem (5B) . . . 16
5.3 Problem (5C) . . . 17

1 Problem 1: Distributions and Best Fit Functions

1.1 (1A)+(1B)

We need to identify the function from which the entries in the first, second and third columns of the data set was sampled. Before we can do this, however, we quickly visualize the distribution of the physical observable in all of the columns. These are displayed in Fig. 1. Visually inspecting the distribution from the desired columns, we can argue which functions that would able to fit them, from the list of functions provided in the assignment. First, I will provide arguments for which function I have chosen to fit with, afterwards the fitting method and results are shown for each column.

First Column:
The first thing we notice, is that the samples have periodic behaviour and that the number of entries in each bin decreases as x grows larger. This leads me to believe that the samples are drawn from one of the two functions:

$$\frac{1}{x+5}\sin(ax) \qquad \text{and} \qquad \sin(ax) + c\exp(bx) + 1. \tag{1}$$

There doesn't seem to be dampening of the amplitude of the sine component, and thus I choose to fit with the latter function.

Second Column:
It appears that there is a minimum around $x = 0$, from where the function is increasing on both sides. This looks like a parabola, and likely one that has been shifted upwards with a constant since it is zero nowhere within the truncated range. We will therefor fit it with the function:

$$1 + ax + bx^2 \tag{2}$$

Third Column:
Here it is clear that there are bins with no entries throughout the data. Inspecting the values closer, we find that they are all integers and I therefor look for a discrete function $f(k)$. We can discard the logarithmic distribution as a candidate, since the most likely value is significantly larger than 1. The Poisson distribution is a limiting case of the binomial distribution which arises when the number of trials n increases indefinitely whilst the product $\lambda = np$, which is the expected value of the number of successes from the trials, remains constant. For large number of trials the two distributions will therefor look similar. I will attempt a fit with a Poisson distribution, in case the p-value comes out low, I can follow up with a fit with a binomial distribution. The binomial and

Poisson distributions are respectively given by:

$$\binom{n}{k} p^k (1-p)^{n-k} \qquad \text{and} \qquad \frac{\lambda^k \exp(-\lambda)}{k!} \tag{3}$$

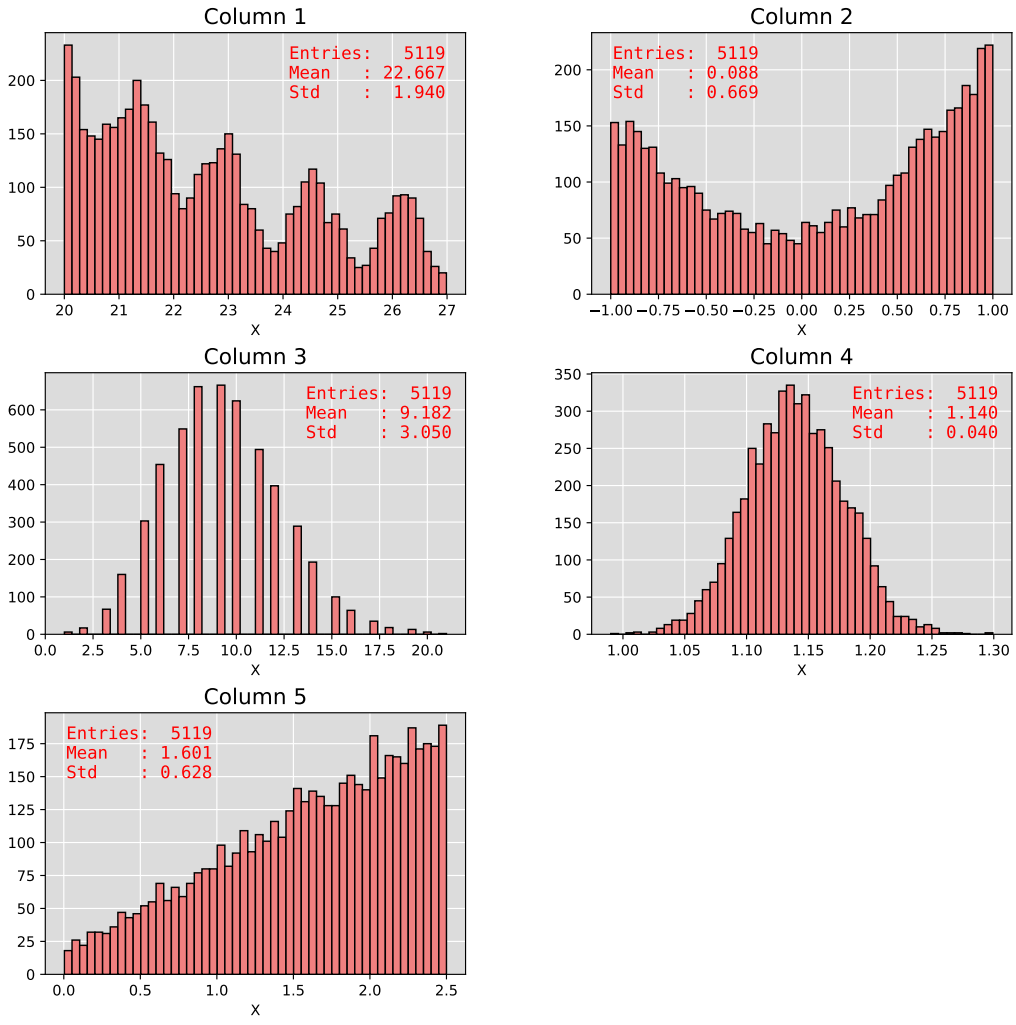

Figure 1. 1D histograms of the entries in each of the columns.

1.2 Fitting the First Column

Since the data is periodical, there are multiple local minima, which can be difficult to fit for minimizers. I therefor choose to utilize Bayesian nested sampling, more specifically the implementation in `UltraNest`. I define a normalised function with the use of numerical integrations through `np.trapz`. I then define a `prior_transform` function, which is the prior we will use to sample the parameters in the fit. The prior is assumed to be flat within the bounds provided in the problem formulation. Furthermore it transforms it back to physical scales, since the sampling actually occurs between 0 and 1. The only thing left to define is a function which take the parameters a, b and c as input and computes the log likelihood for the data in the column. Notice that we don't use the negative likelihood here, since we don't use a minimizer but are instead trying to maximize the Bayesian evidence and thus also the likelihood. The `UltraNest.ReactiveNestedSampler` can then be used to estimate the parameters. We find that $a = 3.899 \pm 0.002$, $b = -0.390 \pm 0.009$ and $c = 6426 \pm 1002$. The nested sampling result can be visualized in a cornerplot, which is displayed in Fig. 2. Notice, how the distributions for the parameters b and c are actually slightly skewed, and asymmetrical errors could therefor be assigned.

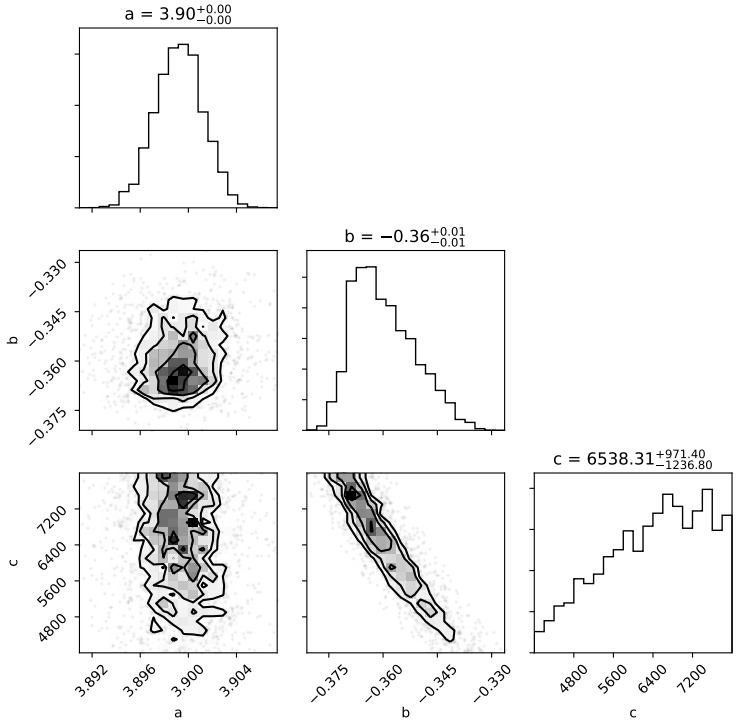

Figure 2. Cornerplot from Bayesian nested sampling with UltraNest.

The fit with these parameter values can be seen in Fig. 3. I have also performed a binned chi2 testbetween the data and the fit. I use 50 bins, since this ensured that there were still 5 entries in each bin. The test is however dependent on the number of bins, so it is important that there

is more than at least 1 entry in each bin, since the Poisson error on the counts otherwise will be huge. I find the chi-square to be $\chi^2 = 35.398$ with associated probability $p = 0.8929$. I can therefor not reject the hypothesis that the data was in fact sampled from a function proportional to: $\sin(ax) + c\exp(bx) + 1$.

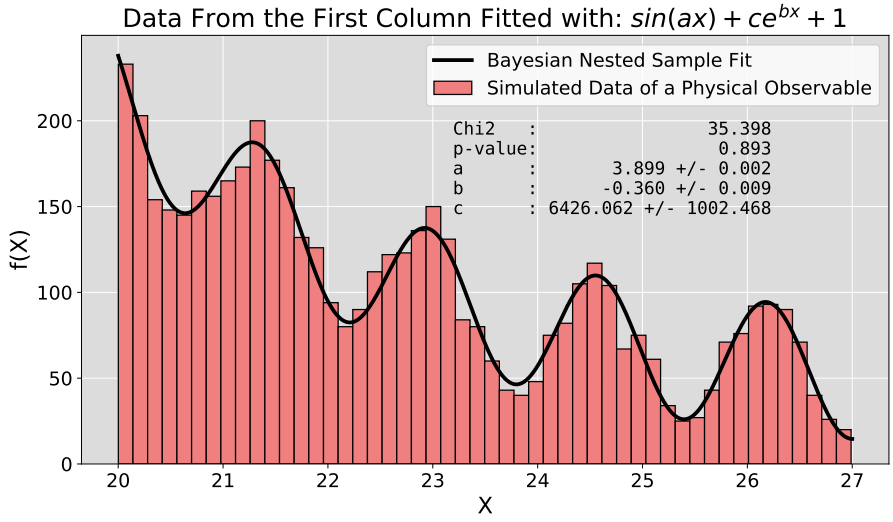

Figure 3. Histogram of the entries in the first column with the best-fit function over-plotted. A binned chi2 test was also performed, with the results displayed on the plot.

1.3 Fitting the Second Column

Since there is only one minimum here, we can just use a standard minimization of the negative log likelihood approach. I use `Minuit.migrad()` as my minimizer, and define the target function myself as the negative likelihood. To make sure we get appropriate error estimations for a likelihood fit we need to set `errordef=0.5`[1]. I impose the bounds from the problem formulation here as well, and as initial guess for both parameters I choose the center value of the bound. Similar to the previous column, when I have obtained the parameters, I perform a binned Chi2-test to get a statistical measure of how good the fit is. I use 50 bins again. A histogram of the data in column 2 is plotted in Fig. 4 along with the best fit. The best fit values and the results are also displayed in the figure. I find the chi-square to be $\chi^2 = 33.7181$ with associated probability $p = 0.9411$. I can therefor not reject the hypothesis that the data was in fact sampled from a function proportional to: $1 + ax + bx^2$.

[1] p.39,https://root.cern.ch/download/minuit.pdf [Accessed 31-03-2022]

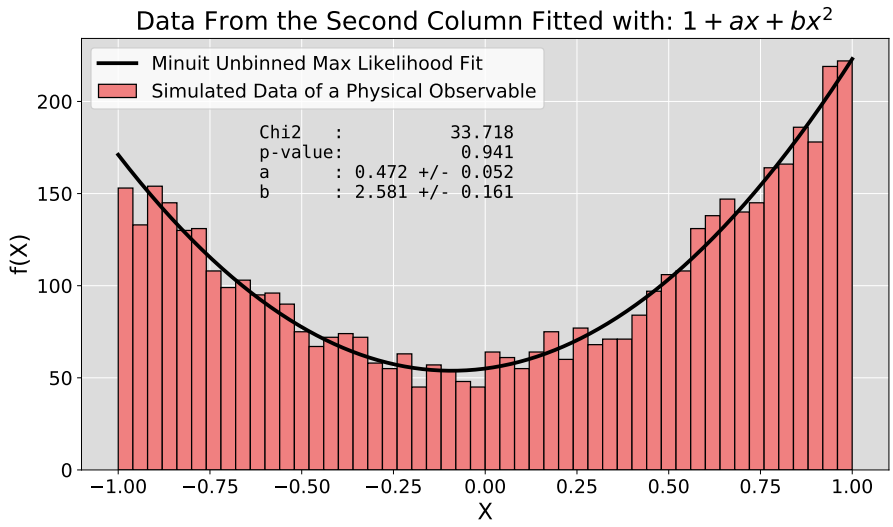

Figure 4. Histogram of the entries in the second column with the best-fit function over-plotted. A binned chi2 test was also performed, with the results displayed on the plot.

1.4 Fitting the Third Column

The procedure for fitting the third column is very similar to what I explained for the second column. When evaluating the fitted function, to use in the chi2 test, one however has to be careful to only evaluate it for integers since it will otherwise be zero due to the nature of the discrete functions. A histogram of the data in column 3 is plotted in Fig. 5 along with the best fit. The best fit parameter values and the results are also displayed in the figure. I find the chi-square to be $\chi^2 = 11.0599$ with associated probability $p = 0.9447$. I can therefor not reject the hypothesis that the data was in fact sampled from a Poisson distribution.

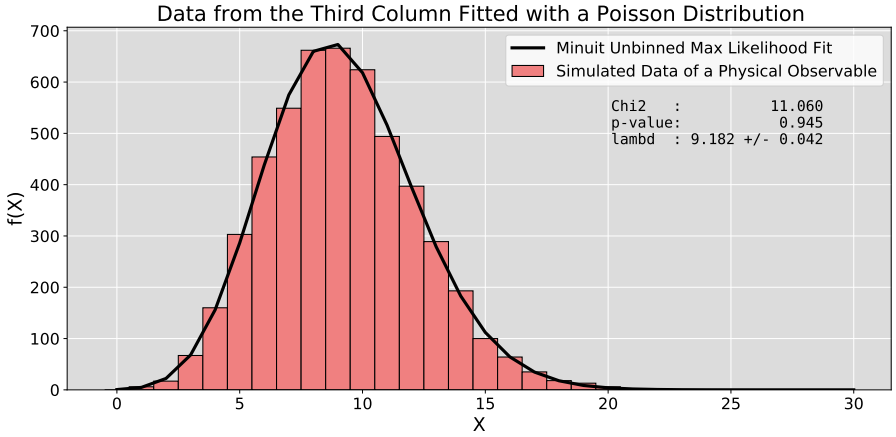

Figure 5. Histogram of the entries in the third column with the best-fit function over-plotted. A binned chi2 test was also performed, with the results displayed on the plot.

2 Problem 2: Spherical Isotropy

We load the data, pairs of coordinates containing the azimuth and the zenith angle of the data points.

2.1 (2A)

We need to quantify whether the provided data is isotropically distributed. A spherically isotropic distribution is uniform in the azimuth angle from 0 to 2π, and uniform in cos(zenith angle) from -1 to 1. We utilize this to sample 139 isotropically distributed data points which are plotted in Fig. 6 together with the actual data. By eye it is hard to judge if the red points, the data, is also isotropically distributed. There do seem to be a little clustering, but this can also occur in random samples of isotropic data.

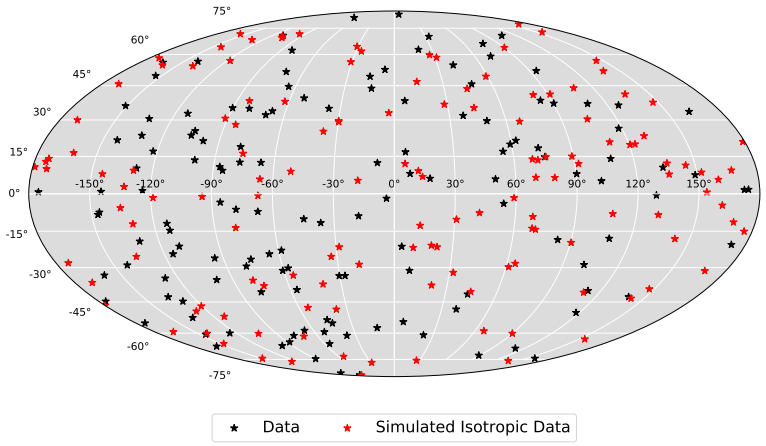

Figure 6. Mollweide projection plot of the provided data and simulated isotropic data. Both samples contain 139 events.

To get a measure of whether or not the data is isotropic, we therefore compute the cumulative two point auto-correlation function, which is defined as:

$$\mathcal{C}(\vec{n_i}, \phi) = \frac{2}{N_{tot}(N_{tot}-1)} \sum_{i=1}^{N_tot]} \sum_{j=1}^{i-1} \Theta(\cos(\phi_{ij}) - cos(\phi))), \tag{4}$$

where $\cos(\phi_{ij})$ is a measure of an angular distance, which is easily computed by the dot product: $\cos(\phi_{ij}) = \vec{n_i} \cdot \vec{n_j}$. The function eseentially counts the pairs of events that are within a angular distance ϕ, it is however normalised to be within the range -1 to 1, since this is the values of cos that spans the entire sphere. This cumulative two point auto-correlation function for the data can then be compared to the predicted function for perfect isotropic data, which is given by:

$$\mathcal{C}_{iso}(\vec{n_i}, \phi) = \frac{1}{2}(1 - \cos(\phi)) \tag{5}$$

These are the building blocks for defining a quantity that is a statistical describes the deviation from an isotropic background distribution. Since both are cumulative functions, we can now compare the them with a Kolmogorov Smirnov (KS) test, defined as:

$$KS(\vec{n_i}) = \sup_{\phi} |\mathcal{C}_{iso}(\vec{n_i}, \phi) - \mathcal{C}(\vec{n_i}, \phi)|. \tag{6}$$

The KS test statistic is found to be $KS = 0.0226$, and the procedure is illustrated in Fig. 7 below.

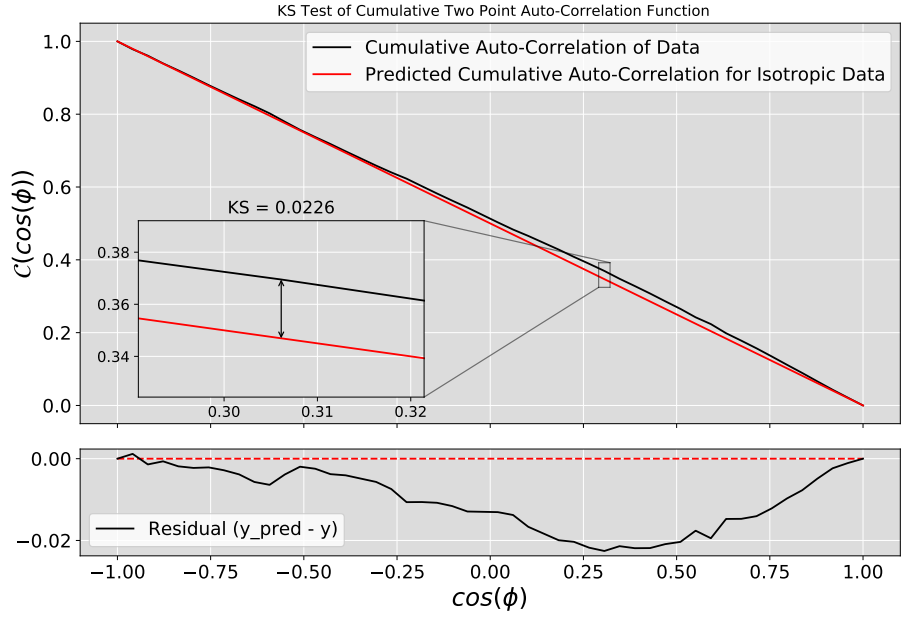

Figure 7. The upper panel show the two cummulative two-point auto-correlation functions, respectively for the provided data and for isotropic data. The KS distance is marked with an arrow, and the KS value is found to 0.0226. The lower panel shows the residual between the two function.

To test the null hypothesis, $H0$, that the data is isotropically distributed, we need to obtain a probability. We do this by performing 5000 pseudo-experiments, where we sample isotropic data. To be able to compare it to the data we keep the number of points in each experiment at 139. We can now calculate the KS value for each of these experiments with the same approach as already described, which will give us a distribution of KS test statistics for isotropic data. We then obtain the p-value with a one tail test: we integrate the function above the KS value obtained for the provided data, 0.0226, and divide it by the full integral. Since this is a distribution with discrete values, and not an actual function, we do this by dividing the number of events above the cut with the total number of events, 5000. This gives us a probability of 0.0042, meaning that we can reject $H0$ with 99.58% certainty. So most likely, the data is not sampled from a isotropic distribution. The distribution of KS values for the 5000 pseudo-experiments is displayed in Fig. 8, where the

cut at 0.0226 is also marked.

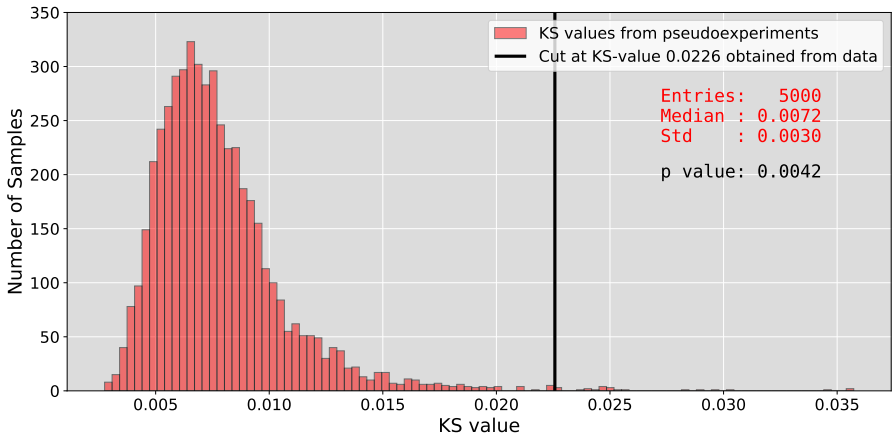

Figure 8. The distribution of KS test statistics for 5000 samples of isotropic data is plotted in red. The black vertical line marks the KS test statistic obtained for the provided data. The probability of the provided data being isotropic is 0.0042.

2.2 (2B)

We now need to test if two other hypothesis fit the data better than the null hypothesis. We work with the hypotheses HA and HB. To test these alternative hypotheses, we repeat the procedure from before for each hypothesis. We produce 5000 pseudo-experiments for each, with each sample still containing 139 points. The method of sampling has been changed, to sample according to the hypotheses. This provided us with a distribution of KS test statistics for both HA and HB. These are plotted in Fig. 9 along with the KS distribution for the isotropic data. I was not confident on which probability we should report for each of the alternative hypotheses, so I have computed two for each of them. One tests whether the median of the KS distribution for HX is compatible with the KS distribution of the isotropic data. The other tests whether KS value of the provided data is compatible with the KS distribution obtained from the hypothesis HX. Here X stands for both A and B. These are reported in Tab. 1.

	Probability
Data compatible with HA	0.3274
Data compatible with HB	0.1144
H0 compatible with HA	0.0122
H0 compatible with HB	0.0846

Table 1. Probabilities of different hypotheseses.

From this it is evident that the most likely match with the provided data is hypothesis HA, but neither HA nor HB can be rejected. We also see that HB is more compatible with $H0$, which

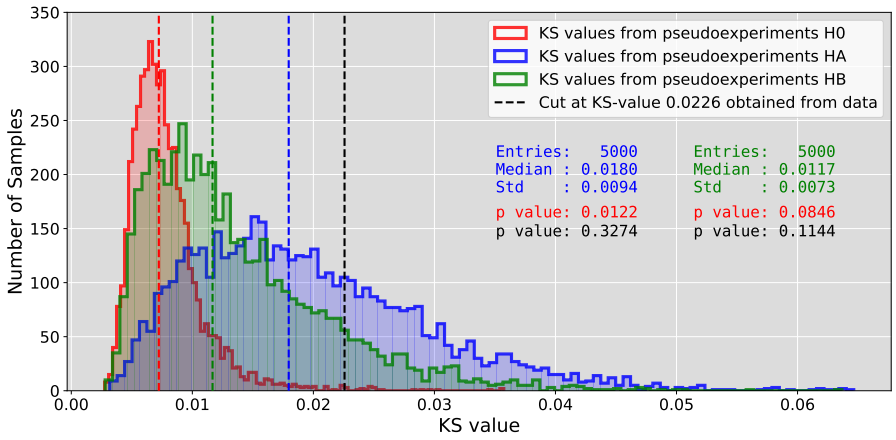

Figure 9

makes since a larger percentage of the data points are actually isotropically distributed.

3 Problem 3: Best Fit with Given Function

We are provided some data, and need to fit the function:

$$f(x, a, b) = \frac{\cos(ax)\cos(bx)}{x^2} + 2 \tag{7}$$

To normalize the function I again use a numerical approach with `np.trapz`. I start by using a standard minimizer approach with Minuit and negative log likelihood as the target function. I enforce the bounds states in the assignment, and use the center values as the initial guess. The best fit values, according to this approach is $a = 4.971 \pm 0.212$ and $b = 17.307 \pm 0.134$. I however run into some problems due to the many local minima present. When performing a binned chi2 test on the resulting best fit (with 50 bins), I obtain a chi2 of $\chi^2 = 99.7347$ with a probability of $p = 0.0000$. So clearly this is not a good fit. Nonetheless, the distribution of the data with the Minuit "best"-fit over plotted is displayed in Fig. 10.

I therefore try the Bayesian nested sampling method that we used for Column 1 in the first problem, which generally performs better with many minima. I enforce the boudns by defining flat priors within these. The log likelihood function is then passed to the `ReactiveNestedSampler`, which finds the best fit parameters to be $a = 5.125 \pm 0.086$ and $b = 22.457 \pm 0.076$. Performing the binned chi2 test (again with 50 bins), I obtain a chi2 of $\chi^2 = 31.641$ with a probability of $p = 0.967$, which is much better. The cornerplot is displayed in Fig. 11, where we can see that the distributions are symmetrical. The histogram of the data over-plotted with the best fit obtained with `UltraNest` is displayed in Fig. 12.

We now make a 2D raster scan of the test statistic used for the fitting routine, in this case the log

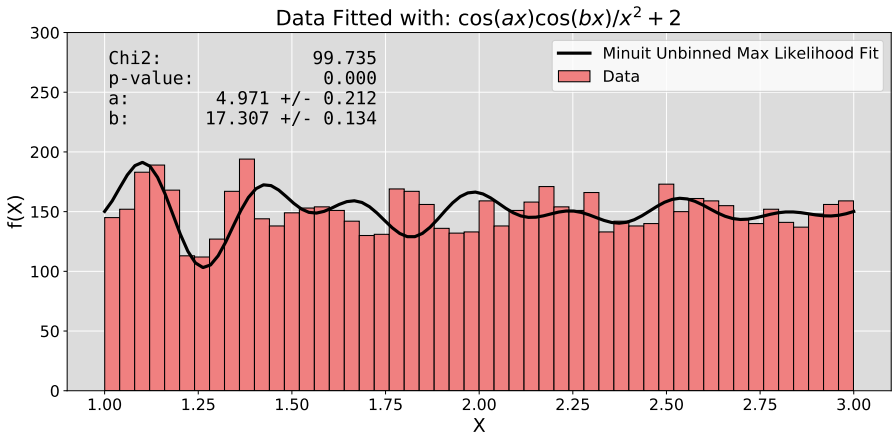

Figure 10. Minuit Fit with negative log likelihood as target function.

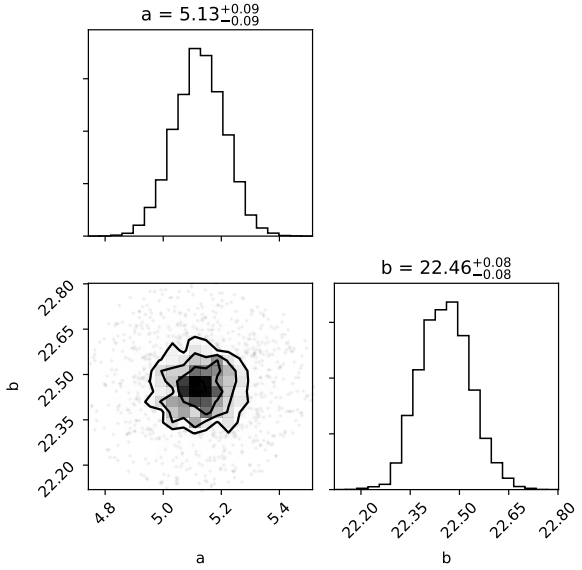

Figure 11. Cornerplot of the Bayesian Nested Sampling with UltraNest.

likelihood, around the best-fit parameters $\hat{a}$ and $\hat{b}$ from the `UltraNest` approach. This is done by making a `np.meshgrid` of the defined range in both a and b, and evaluating the log likelihood with steps of 0.1 in both a and b. The raster scan is seen in Fig. 13. The best fit from both the nested sampling and the raster scan is marked, they lie directly on top of each other, however, so only one cross can be seen.

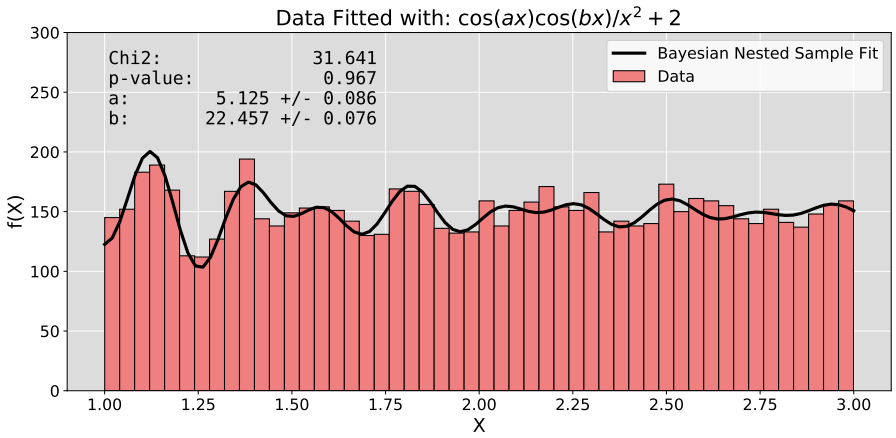

Figure 12. UltraNest Fit with log likelihood as target function.

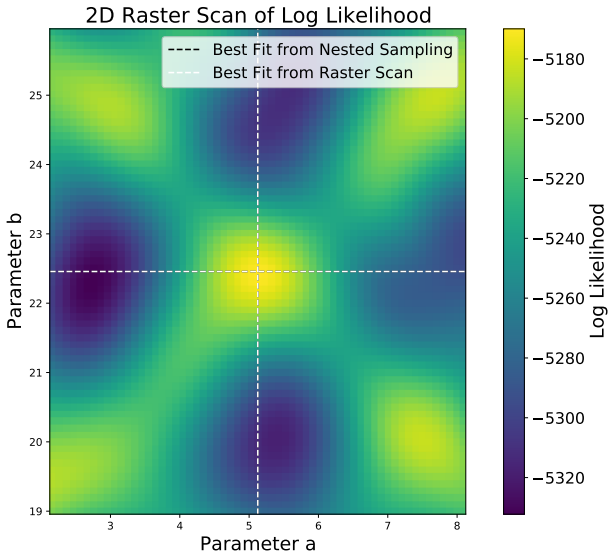

Figure 13. 2D raster scan of the log likelihood. The best fit from both the nested sampling and the raster scan is marked.

4 Problem 4: Classification of No-Shows

4.1 Exploring the Data

We import all of the data provided, but before we start building our model, it is a good check to see if the train and test data share similar properties. If not the model will not perform well. We start by getting some counts of the data. We find that there are 20000 patients in all of the data sets. In the train set there are 15951 patients who showed and 4049 who didn't. In the test set there are 15987 patients who showed and 4013 who didn't. Calculating the ratio between the classes

in both samples, we find that the ratio is 0.25 in both cases, which is promising. The classes are unbalanced, but as long as they are unbalanced in a similar manner, this should be alright. One thing we also need to ensure to produce a good model, is that the train and test sample represent the parameter space in a similar manor. To ensure this is true, I plot histograms of each parameters for both the train and test set. This can be seen in Fig. 14 below. We see that the data sets have similar patterns, which is good.

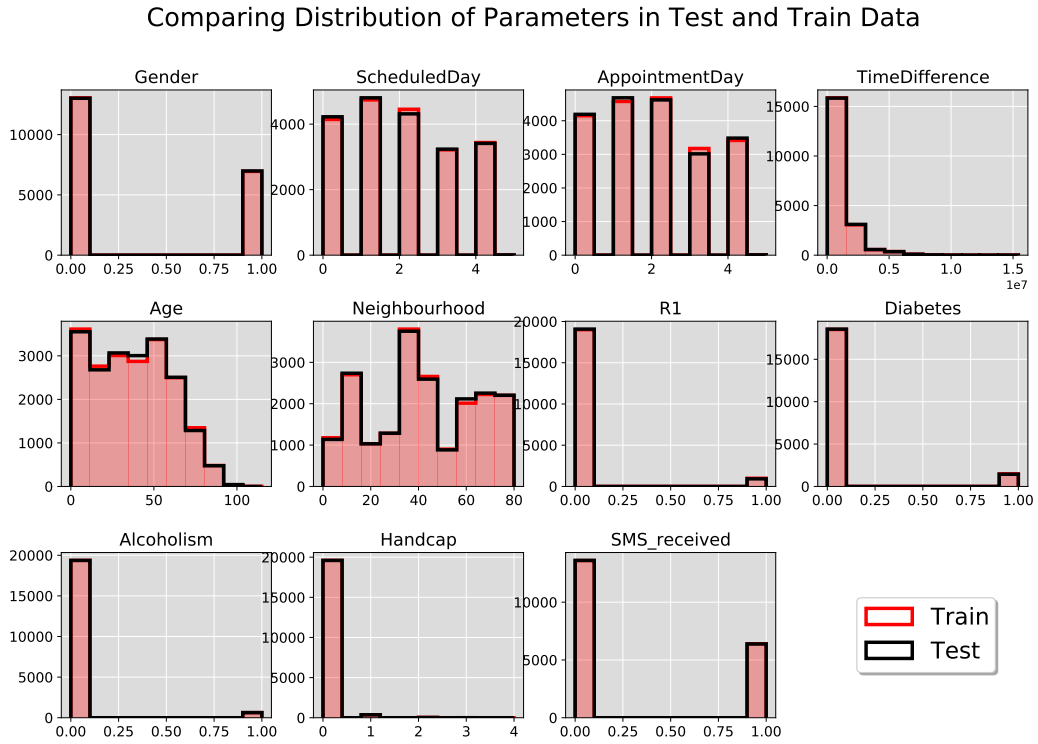

Figure 14. Histograms for each parameter in the data set plotted for both the train and test sample.

4.2 Hyperparameter Optimization

I choose to use `XGBoost`'s `XGBClassifier` for this problem. There are many different models that could be used and some may perform better than others. It has two main hyperparameters that we can change for optimal precision, namely the `max_depth` and the `n_estimators`. There are different ways to choose the parameters, when we are tweaking more than one at a time, examples are a gridsearch, bayesian optimization and a randomized search. For this problem I have chosen a grid search with cross validation implemented with `sklearn`'s `GridSearchCV`. I make an initial search for `max_depth` in the interval 2 to 10, and `n_estimators` in the interval 1 to 150. I use 3 folds for the cross validation. In this run the best accuracy was 0.818 with `max_depth`=4 and `n_estimators`=75.

From this I make a new more constrained grid search in the interval 2 to 6 and 50 to 100 for the two hyperparameters respectively. This is also run with 3 folds cross validation. The best accuracy obtained also 0.818 with `max_depth`=4 and `n_estimators`=80, so it is not meaningful to do more constrained grid searches. The results from these two searches are illustrated in Fig. 15. The final best configuration of `max_depth`=4 and `n_estimators`=80 is chosen for the model.

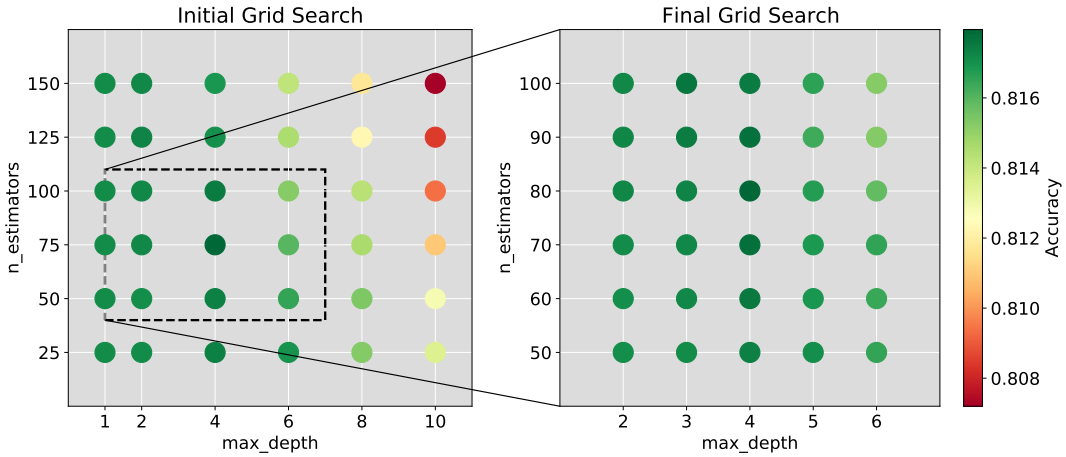

Figure 15. Accuracy plotted as a function of `max_depth` and `n_estimators` where the color indicates the accuracy. This is plotted for the two grid searches performed.

4.2.1 Constructing the Classifier

We use the hyperparameter settings found to be optimal in the last grid search. I choose to set the learning rate to 0.1 and the metric that is evaluated is a logloss function. When the model is set up, it is fitted to the train set with `.fit()`. When the model is fitted is has to methods we can use. `.predict()` will give us a list of labels it has predicted while `.predict_proba()` will return the test statistic. First however we can use `.predict()` and produce a ROC curve to see if how the trade off between true positive rate (TPR) and false positive rate (FPR) is. The ROC curve is computed with `roc_curve` from `sklearn.metrics`, and is displayed in Fig. 16. It is definately better than a random classifier, which would look like a plot going from the bottom left corner to the top right, but it is not the best classifier. A perfect classifier would go straight up to the left corner, misclassifying no events. This is likely an effect of the data we use, and the fact that it is hard to classify which patients are no shows.

4.3 (4A)

Now that we have constructed and trained our model, we can construct the plot that is asked for in the problem. In Fig. 17 I have plottet overlaid histograms using all events from the test file versus the test statistic; separated into 'No-show==1' and 'No-show==0', in the right panel. In

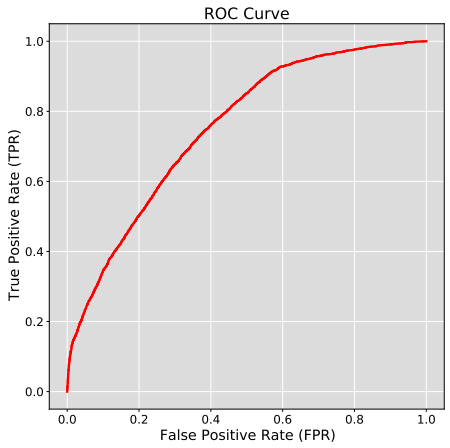

Figure 16. The receiver operating characteristic curve for the predictions on the test data.

the left panel I have done the same for the train data to be able to see if there would be a large discrepancy between the two (which is undesired). We see that there is no significant discrepancy between the two ,the slight difference in accuracy is likely due to statistical fluctuations. The accuracy displayed, is the accuracy obtained from using the `.predict()` function, which implicitly uses a cut on the test statistic of 0.5 to seperate the classes. In an attempt to increase the accuracy, I wanted to investigate whether we could shift the cut upwards or downwards and obtain better classification. To do this I computed the resulting accuracy for both the train and test sample for different values of the cut in the range 0 to 1. The accuracy is plotted as a function of the value of the cut in Fig. 18. Unfortunately there was no real peak in the function, and the accuracy was not improved, so I will keep the cut at the default 0.5 for the classification of the blind sample.

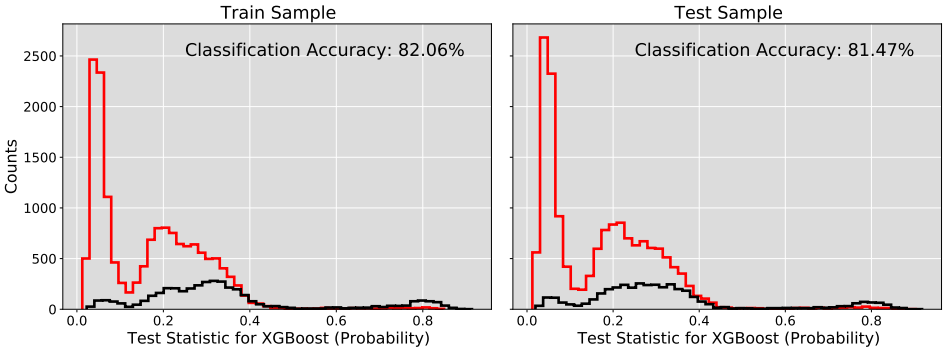

Figure 17. Left panel: Overlaid histograms using all events from the train file versus the test statistic; separated into 'No-show==1' and 'No-show==0'. Right panel: the same using the test file.

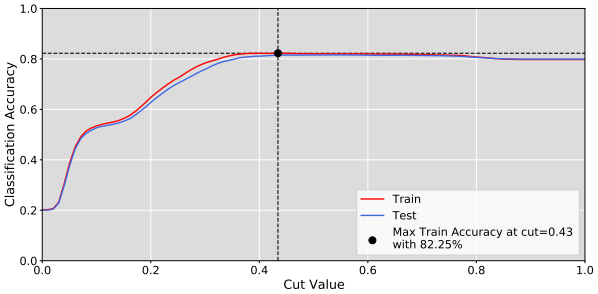

Figure 18. Accuracy as a function of the cut chosen for the test statistic in classifying the classes.

	Feature Importance
R1	0.48386505
TimeDifference	0.17927186
Age	0.05610883
ScheduledDay	0.04755081
SMS received	0.04445668
Handcap	0.0400703
Alcoholism	0.03573309
AppointmentDay	0.0346735
Neighbourhood	0.03245685
Gender	0.02748342
Diabetes	0.01832965

Table 2. Feature Importance

4.4 (4B)

The `sklearn` wrapper for `XGBoost` has an inbuilt `feature_importance_` function that we can use to rank the variables. The results are displayed in Tab. 2.

4.5 (4C)

We now make predictions of the "real data" where we don't have the labels but have the ID's instead. This is done using `model.predict(X_blind)` where model is the defined model that has been trained on the training set and `X_blind` are the parameters (not the ID) from the real data set. The ID's of the true no-shows can be found in the file `Data/Kreilgaard.AMAS_Exam.Problem4.NoShowTrue.txt` and the false no-shows can be found in the file `Data/Kreilgaard.AMAS_Exam.Problem4.NoShowFalse.txt`. There are 20000 individuals in this data set. The model classified 798 as true no-shows and 19202 as false no-shows. This gives a ratio of 0.04, which is a lot less than what we saw before (0.25). This could of course just be a tricky sample, but I suspect that the model has a hard time classifying them correctly.

5 Problem 5: Mars Data and Splines

First we visualize the data, as can be seen in Fig. 19

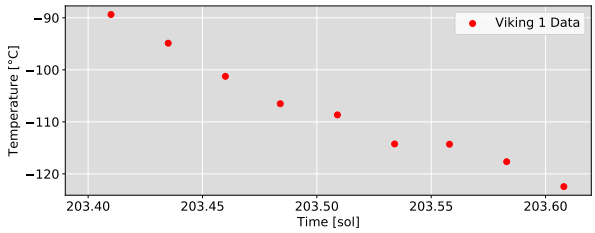

Figure 19. Scatterplot of the provided data

5.1 Problem (5A)

I use the data provided to create 2 splines: a linear and a cubic one, both implemented with `scipy.integrate.interp1d`. Evaluating the temperature on sol 203.570, I find the temperature to be $T_{lin} = -115.9128$ and $T_{lin} = --115.3253$ degrees celsius respectively for the linear and the cubic spline. The two values are slightly different, but this is expected since we can expect the linear spline to undershoot some regions while the cubic spline has a tendency to over-interpret the features of the data (overshooting) especially when there are curves in the data.

5.2 Problem (5B)

I now make a scatter plot of the interpolated temperatures from both splines, using 250 points ranged between 203.410 and 203.608 sol. The scatter plot is displayed in Fig. 20.

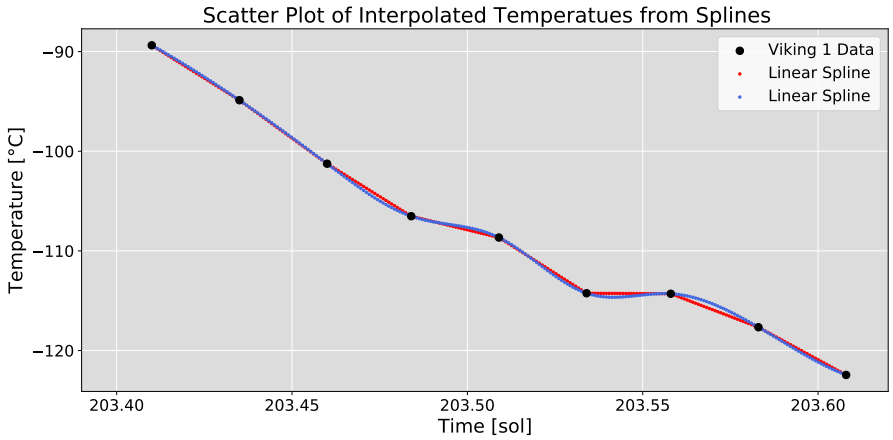

Figure 20. Scatter plot of the interpolated temperatures from a linear and a cubic spline.

We check whether the temperature is continously dropping, by taking the difference between points: temp[i] - temp[i-1]. If this value is ever positive, the first value (i-1) is largest, and the temperature has not dropped. The difference in temperature between two points is plotted as a function of the time in Fig. 21. We see that the cubic spline violates the condition in the time interval 203.5428 sol to 203.5563 sol. We should therefore look more closely at this interval to make sure that the interpolated temperature is monotonically decreasing.

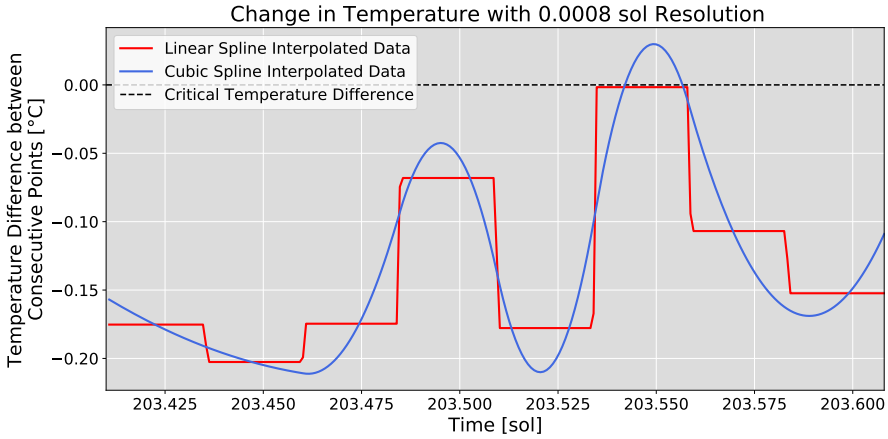

Figure 21. The difference in temperature between two points is plotted as a function of the time. The resolution for 250 points as is used here is 0.008 sols.

5.3 Problem (5C)

We now imagine that proposed electronics components for a Mars rover, or other Mars planetary-surface exploration vehicle, are unable to sustain temperature changes of more than 0.09 C within 0.0004 sol. To check this I have plotted the absolute temperature change with a resolution of 0.004 sol as a function of the time in Fig. 22. This means I had to reevaluate the spline functions in new points that corresponded to this resolution. In black I have marked the maximum temperature change, and we see that this is violated, therefore the proposed electronics components are not able to sustain the temperature changes. New electronics is therefore required for a successful mission.

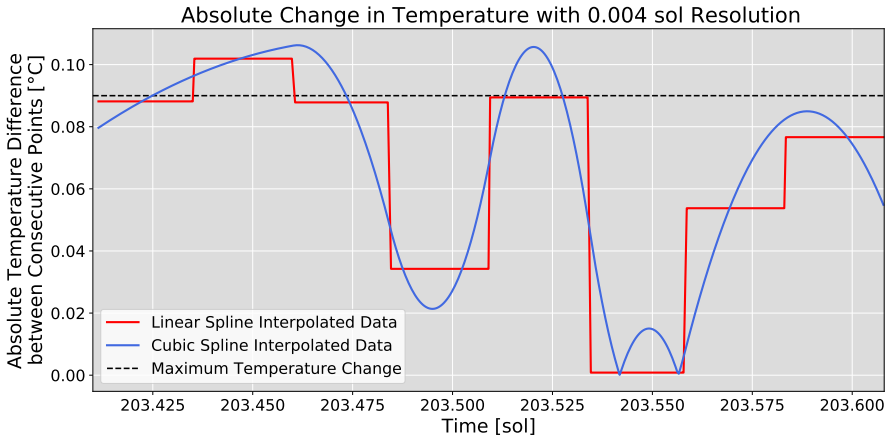

Figure 22. Absolute change in temperature as a function of time, with a time resolution of 0.0004 sol.