Time-over-threshold calibration methods for ALPIDE MAPS

Leah Albrow
Supervisors: Andrea Triolo, Ivan Ravasenga

Abstract
The ALPIDE sensors in ALICE's ITS2 are digital pixel sensors, therefore do not collect information about the charge deposited by incoming particles in the standard run configuration. A new sensing mode, the color run, allows for measurements to be taken of the duration that pixels remain 'on'. This quantity is known as the Time-over-Threshold (ToT). If the pixels are properly calibrated, ToT can be converted into deposited charge and used for particle identification. It is impractical to calibrate each pixel in depth, due to the time-consuming nature of calibration runs. In this study, four methods of calibrating the pixels based on two measured data points, and the threshold for the pixel to switch on are compared. The methods are primarily ranked on the accuracy of the fits, and ease of implementation and time costs are considered as secondary factors.

1 Introduction

ALICE (A Large Ion Collider Experiment) is one of the primary experiments operating at the Large Hadron Collider (LHC). During the LHC's Long Shutdown 2 (2019-2022), ALICE underwent significant upgrades, (ALICE Collaboration, 2024), including major enhancements to the Inner Tracking System (ITS), now known as ITS2. The ITS2 uses the ALPIDE sensor (ALICE ITS ALPIDE development team, 2016), and is the largest deployment of Monolithic Active Pixel Sensors (MAPS) in high-energy physics. Each ALPIDE chip has 1024×512 sensing pixels, and are arranged in seven layers, with 192 staves forming a cylindrical structure around the collider beam pipe (ITS Collaboration, 2024). Overall, there are 12.5×10^9 pixels, each with individually tunable discrimination thresholds. These pixels have digital readouts, so they are either on or off, and do not measure the deposited charge in the default configuration.

When accurately calibrated, the system can determine the position of incoming particles, which is crucial for reconstructing the trajectories of particles produced by LHC collisions. The ALPIDE chip can be configured in a special reading configuration, known as the color run, where an additional parameter, the Time-over-Threshold (ToT), becomes accessible. ToT represents the duration a pixel remains in the 'on' state, which depends on the charge deposited by an incoming particle. This information can be used for particle identification, expanding the capabilities of the ITS2. To convert the ToT to the deposited charge, precise calibration at the individual pixel level is necessary for the innermost layers of ALPIDE chips.

The complete ToT-charge curve can be obtained from a ToT calibration run, however these runs are time-consuming and do not produce other useful scientific data. Therefore, a method is needed to calibrate the pixels using a minimal number of measurements while still maintaining accuracy. Two ToT measurements with injected charge 300 and 600 electrons are known, as well as the discriminant threshold for each pixel.

2 Methods

A method for characterising the pixels had already been developed before this study. This method used a parametric model, which applied different functions to different sections of the ToT-charge curve. This parametric model was compared to several techniques applied to fitting Equation 6.1 from Bolzonella et al. (2024). Although this equation was orginally used for calibrating Timepix4 chips, the resultant curve looked similar to the ToT-charge curves from the ALPIDE MAPS. First, a basic least squares fit using SciPy's curvefit function was performed. To improve the goodness-of-fit to the complete datasets, two machine learning (ML) algorithms were used to account for the additional knowledge we have about the distributions. The techniques used were k-nearest neighbours regression algorithm and a neural network.

All methods used three specific points in the ToT-charge plane for fitting. The first point's charge coordinate is the threshold, which is the minimum charge required to activate the pixel. Next, measurements were taken for the Time-over-Threshold (ToT) with 300 e- and 600 e- of injected charge. A line was then constructed from these two measured points and extrapolated to determine the intersection with the threshold. The first point's ToT coordinate is chosen halfway between the intersection and the vertical (charge) axis. The three points used for fitting can be seen as the large black points in Figure 1. These points are referred to as (T_1, Q_1), (T_2, Q_2), and (T_3, Q_3) in this report.

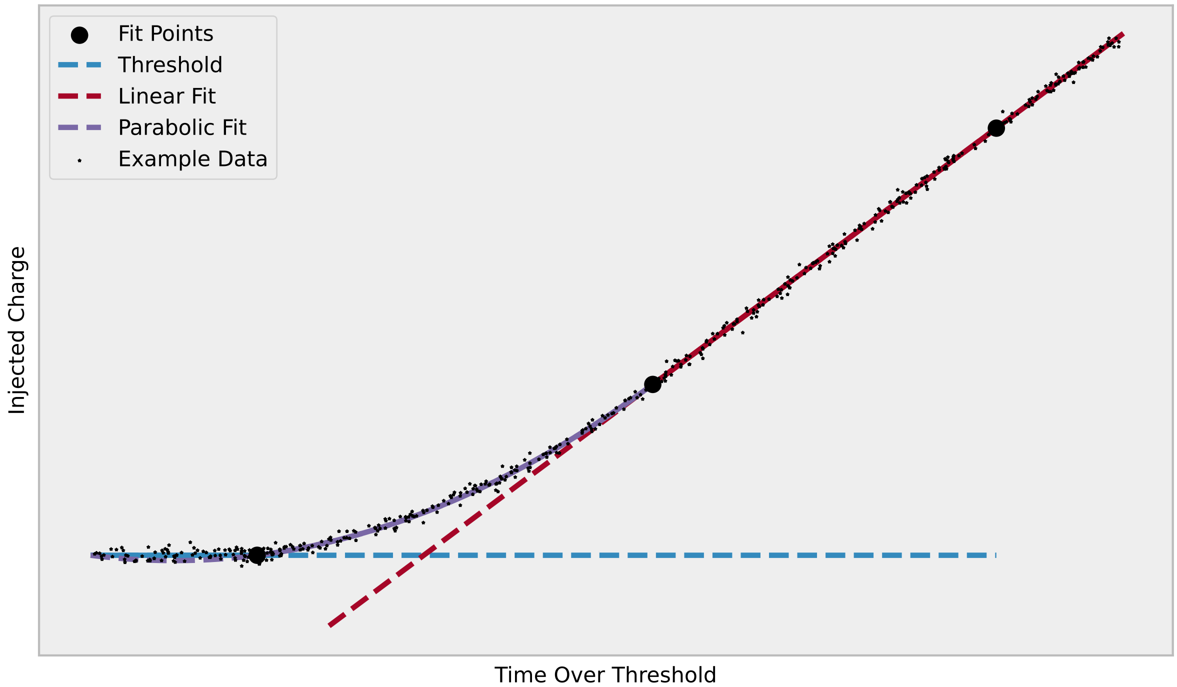

Figure 1: Parametric method of ToT-charge curves, demonstrated with simulated data. The solid regions show where each function is used, and the dashed regions show the continuation of each function outside the region. The two rightmost large fit points are directly measured, and the third fit point is from the pixel's threshold and halfway from the intersection of the threshold and linear fits.

A significant limitation is the amount of available complete data for the pixels. Row 100 of 9 chips has been measured (full column of 1024 pixels) with the complete charge-ToT curves. This means we have a maximal fully known data set of 9216 pixels to both train and validate the quality of fits from. There are many other pixels with the measured 300 and 600 e- points taken, but without full knowledge of the curve, the χ^2 cannot be calculated.

2.1 Parametric Fitting

The parametric fit is constructed in three segments. First, if the injected charge is less than Q_1 (the threshold charge), the pixel remains off. Between Q_1 and Q_2, the ToT is modelled parabolically. The parabola is defined to pass through points 1 and 2, and a third point where the charge equals Q_1 and the ToT is zero. This ensures continuity of the fit at the junctions. For charges above Q_2, a linear function defined by the points Q_2 and Q_3 is applied. These three fits are illustrated using simulated data in Figure 1.

In order to completely describe this model, the coefficients a, b, and c of the parabola, and the slope and offset of the linear function must be determined. Although these values can be fully calculated from the measured points, requiring so many parameters is not ideal, and actually extracting information from the model is inconvenient. This method is also susceptible to over-fitting, and small changes in either of the measured points can greatly change the overall model, particularly in the linear regions.

2.2 Functional Form

As in Bolzonella et al. (2024), the function

$$Q = p_0 + p_1 T + \frac{p_2}{T + p_3} \tag{1}$$

was used to fit the ToT-charge data. The parameter p_3 controls the location of the vertical asymptote of the function, which will always be to the left of the region of interest for pixel calibration. In order to use a non-linear least squares regression fit, such as SciPy's curvefit function, there must be at least as many data points as fitting parameters. This is because the fitting is done by minimising the sum of squares of the residuals of a system of equations, which will be indeterminate if there are more parameters than data points. As the effect of p_3 is minimal, to conduct a curvefit with the constraint of only having three measured points, p_3 was set to zero for all fitting. The remaining parameters adjust the steepness and location of the slope, so the effects of removing this parameter are likely insignificant and able to be compensated for with other parameters.

The functional form used for fitting is thus

$$Q = p_0 + p_1 T + \frac{p_2}{T}. \tag{2}$$

This function was fit to the data, and is labelled the least-squares (LS) fit. I then investigated whether there were ways to improve the fit using the additional knowledge we have about the shape of the ToT-charge curve. I considered two methods of approaching this, either by creating synthetic points with uncertainties from the measured points (e.g. adding a fourth point along a linear extrapolation of the measured data), or by using machine learning methods to reduce the sensitivity of the LS fit, and provide information from the complete curve.

2.3 Machine Learning

2.3.1 k-Nearest Neighbours

Two different machine learning methods were attempted to improve the quality of fit of Equation 2. The first method was the k-nearest neighbours algorithm (k-NN), a supervised learning technique used for both regression and classification tasks. For regression, k-NN uses a training set of vectors. When predicting the output for a new vector, it computes a weighted average of the vectors in the training set, based on the distances to the k nearest neighbours. k is a specified value, and must be chosen carefully to avoid over-fitting or under-fitting the data.

For this data, the training input vectors were (T_1, Q_1, T_2, T_3, p_{0i}, p_{1i}, p_{2i}), where the p_i parameters are from the LS fit to the three points. The output vectors were (p_{0f}, p_{1f}, p_{2f}), and are trained on the true best fit parameters. These were found from fitting Equation 2 to complete measured pixel ToT-charge curves. The regression was performed using the Python package SCIKIT-LEARN. For later discussion of results these fits are called KNN.

Due to time constraints for this study, only $k = 3$ was explored, as it provided a good fit, and was an improvement on the LS fit. However, it may be valuable to investigate other values of k in the future if this method is chosen to be implemented for pixel calibration. Additionally, including the initial parameter guesses may be unnecessary, as these parameters are determined through T_1, Q_1, T_2, and T_3. This means the information may be degenerate and have no effect (or even a negative effect) on the performance of the k-NN algorithm. The impacts of this have not been explored in this study, but would be valuable in understanding the effectiveness of this algorithm for the task.

2.3.2 Neural Network

The second machine learning method employed was a neural network-based model constructed using TENSORFLOW and KERAS. A simple architecture was chosen. Initially, the training and testing data were standardised using a StandardScaler. The standardised data was then passed through a neural network with three dense layers, as depicted in Figure 2. The model was trained for 100 epochs, as the loss did not improve beyond this point.

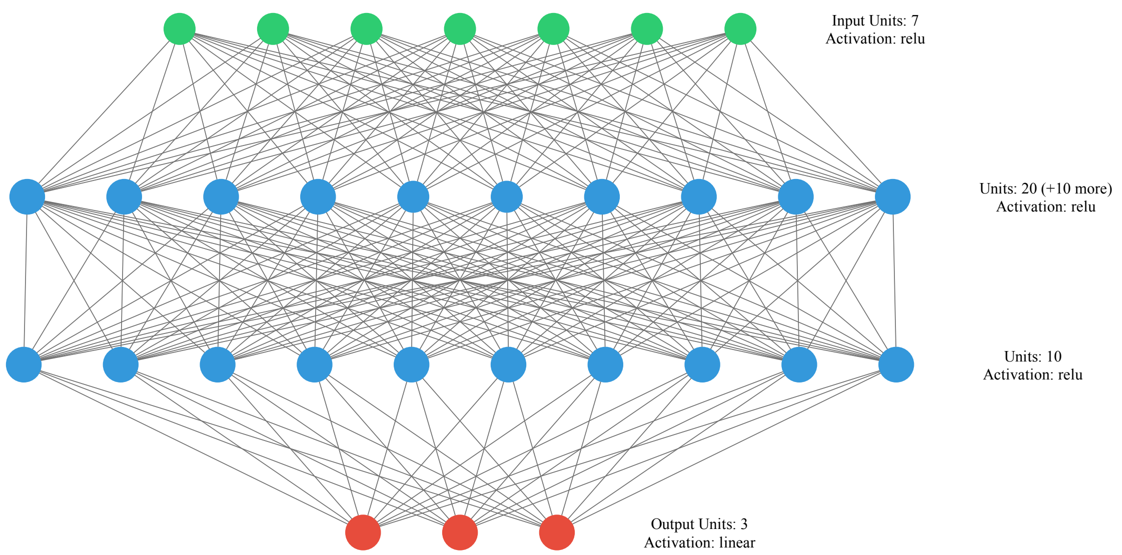

Figure 2: Neural network taking a seven parameter vector, and predicting the best fit parameters for the model.

The performance was evaluated based on the loss function, shown in Figure 3, which was defined using mean squared error (MSE). The Adam optimiser, an implementation of the Adaptive Moment Estimation, and the Rectified Linear Unit (ReLU) activation function were used in the training of the neural network. The output of this model is referred to as NN for subsequent discussion and comparison. Chips 0-6 were used for training, while chips 7 and 8 served as the validation set for examining the performance of the NN method.

For the later work where this is applied to pixels where the complete curve is unknown,

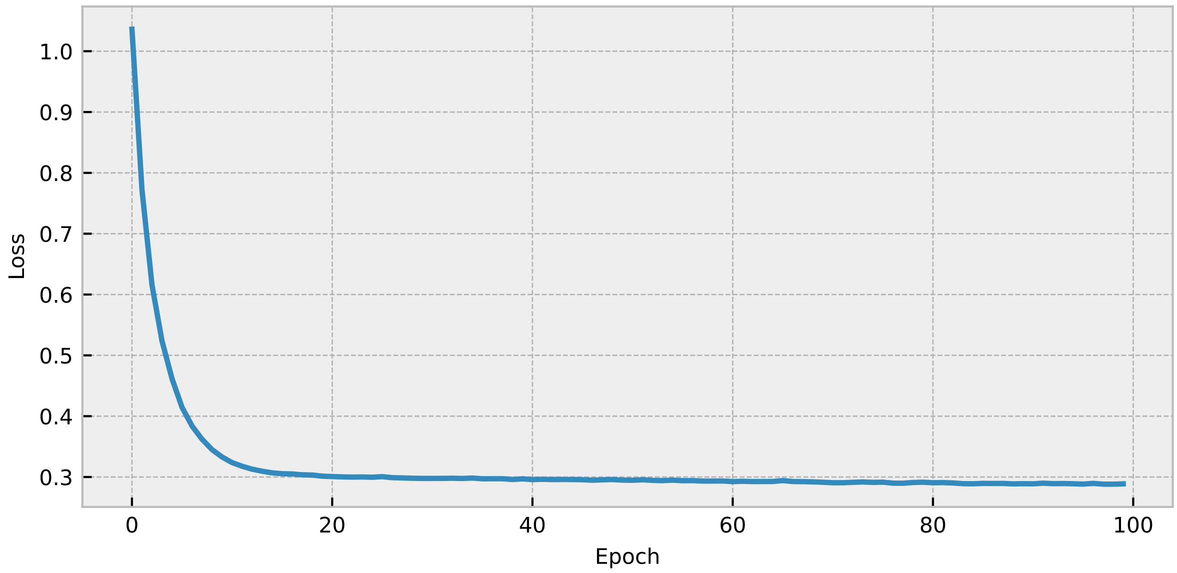

Figure 3: Mean squared error throughout the learning period for the neural network.

the neural network is trained on all nine chips, as more data will improve the quality of the fits. The NN and KNN models were then used to predict the best fits for other pixels/chips, but as the complete curves were not known, the quality of these fits could not be verified.

3 Results and Discussion

The LS, KNN and NN fits were compared against the parametric fits, using the χ^2 as a measure of quality of fit. The definition

$$\chi^2 = \sum \frac{(x_i - X_i)^2}{X_i} \tag{3}$$

was used, with x_i being the observed data (the measured data points), and X_i being the expected data (the fitting curves). The data from chips 7 and 8 was evaluated using this metric, so 2048 fits were completed in total, as the other chips with the full available measurements were used to train the models. An example pixel with the different fits completed is shown in Figure 4.

Individual pixels may be better fit by different methods, but we are interested in the overall performance across a large sample of chips. To compare performance of different methods, histograms were plotted of the χ^2 results for all of the fitted pixels. The results of this can be seen in Figure 5. All of the histograms have a significant amount of overlap, with the KNN and NN fits seeming to have a lower mean χ^2. This was tested using Welch's t-test, which tests the null hypothesis that two samples have equal means. This test was conducted on the logarithmically transformed data, and the results of the tests are shown in Table 1.

The results of the t-tests show all compared fits have statistically significant differences, with the closest being LS to Parametric and KNN to NN. In general, the closest fit to the complete curves were achieved through the NN and KNN methods. These outperform the naïve LS fitting approach, which is also generally better than the parametric method. This also shows that the KNN and NN approaches yield the most similar results, as they have the smallest t-statistic between them, but still have statistically significant differences.

The difference between the χ^2 values for each pixel were also plotted in a grid of histograms, shown in Figure 6. This was created in order to see whether the methods which had the lowest χ^2 value were always better performing, or if there was a dependence on the data given for

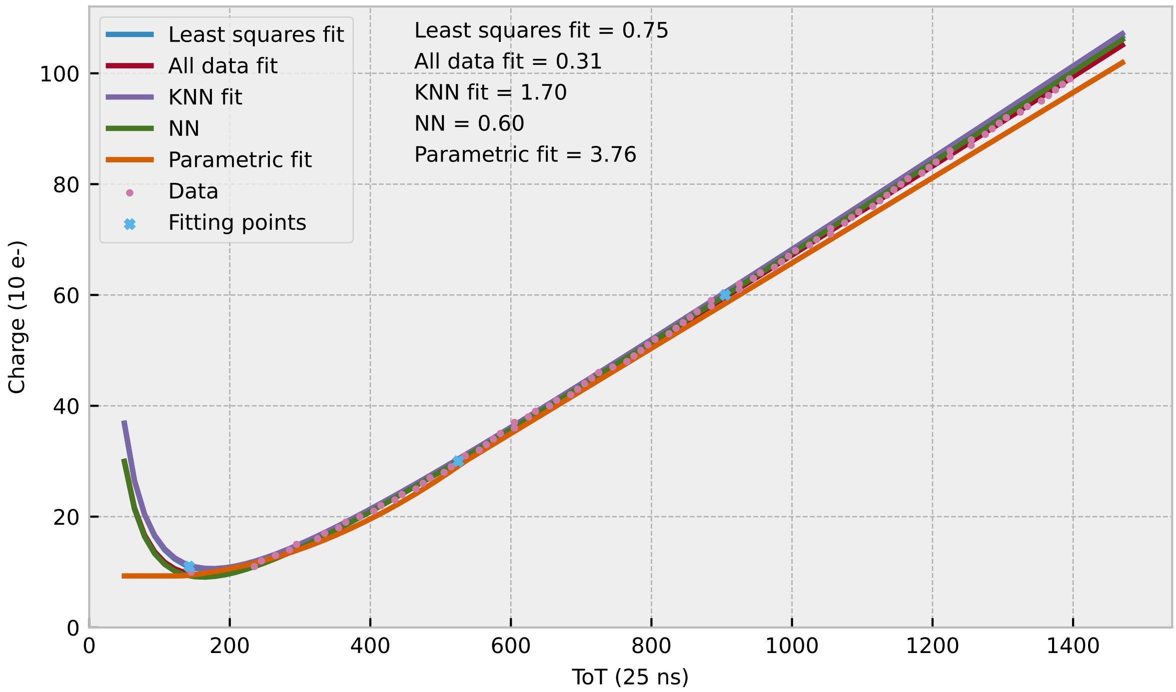

Figure 4: Models for pixel ID 959 on row 100 chip 8. Note the blue fitting points do not match those used to define the parametric fit, as there has been a new calibration run since the parametric fit was determined.

Table 1: Welch's t-test absolute t-statistic results for χ^2 distributions of pixels on chip 7 and 8. P-values are not reported as the largest p-value was 1.7×10^{-16}. Cells are colour coded according to their magnitude.

Parametric	138.2447			
LS	75.6604	23.4611		
KNN	64.4660	39.9671	12.6521	
NN	59.9901	52.6710	21.1832	8.2760
	All	Parametric	LS	KNN

quality of fit. Note that the difference in χ^2 is on a linear scale, and the previous histograms were on a logarithmic scale. Here we see that the complete all data fit is always the best, and there are no immediately obvious unusual structures in the distribution, however this was not investigated in detail.

Another factor to consider is the resolution of the charge in ToT bins achieved by different methods. To assess this, the number of counts across all the fitted pixels was plotted in a 2D histogram, revealing the spread of the data across various fitting regions. Figure 7 highlights which fits had a narrower range and illustrates the variance between pixels. The true resolution of the data is shown in the final panel of this figure. The resolution plots show the major issue with the least squares fit, which can be seen in the streaky features in the top left panel. This shows that the least squares fit is very sensitive to the given data points, meaning the dispersion at the highest ToT gets large. This is representative of the data, but given that we only have data points at 30 and 60, we cannot constrain this accurately. This may be one of the reasons for the improved fit from the machine learning approaches. By averaging multiple possible solutions, or seeing patterns in true fit and the given data points, the sensitivity to variation in

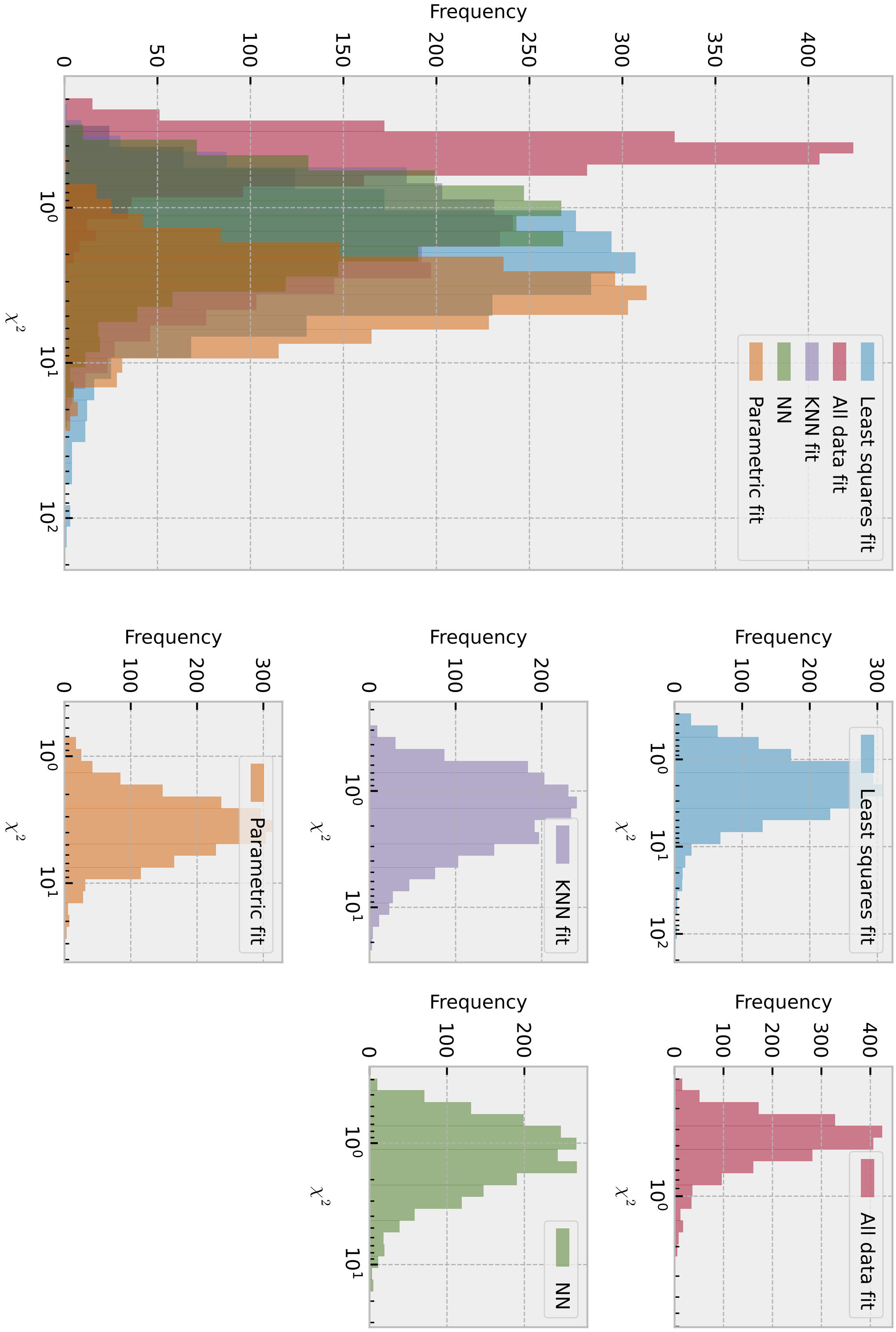

Figure 5: χ^2 for different models, data from row 100 of chips 7 and 8.

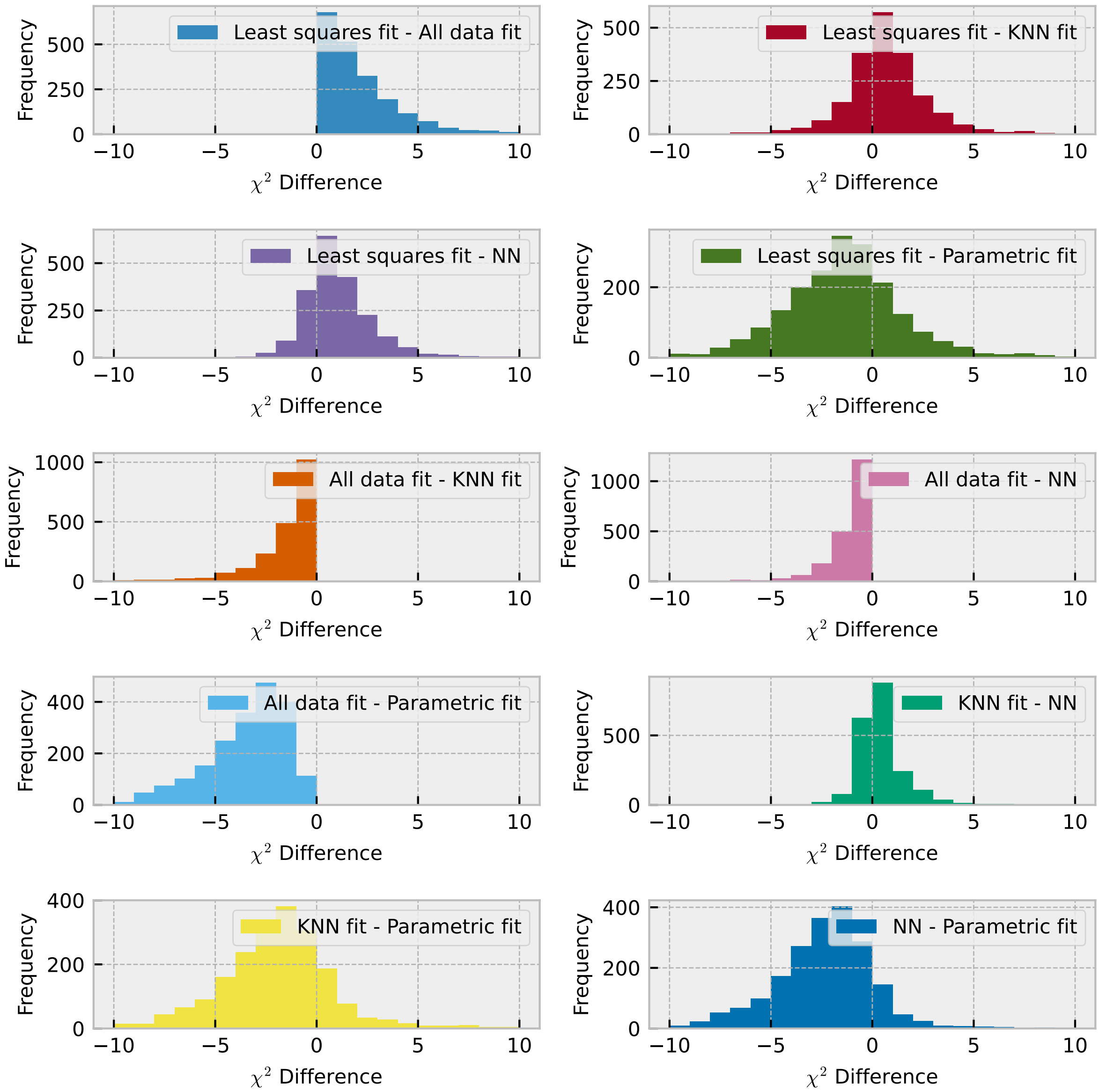

Figure 6: Differences in χ^2 for different methods of modelling across the studied pixels. Whichever fit comes first in the label is favoured if the data is negative.

the data is able to be reduced.

The one-dimensional projections of the histograms for each slice of Figure 7 along the x-axis were taken, and the standard deviation for each slice was calculated, as shown in Figure 8. This standard deviation represents the resolution for a specific Time-over-Threshold, which indicates the maximum accuracy in measuring charge from ToT. When connecting the measured ToT to charge, it is necessary to consider the dispersion in ToT and how it relates to the dispersion in charge. This relationship defines the possible range of charge values corresponding to the measurement. Therefore, a lower σ charge allows for more accurate predictions of the deposited charge from ToT measurements.

Figure 8 shows the streakiness of the fits for LS and the Best fits as a high σ charge in the high ToT region of the plot. Note that the resolution plot does not indicate accuracy to the true value. As the parametric method is quite rigid with it's linear trend for high ToT it results

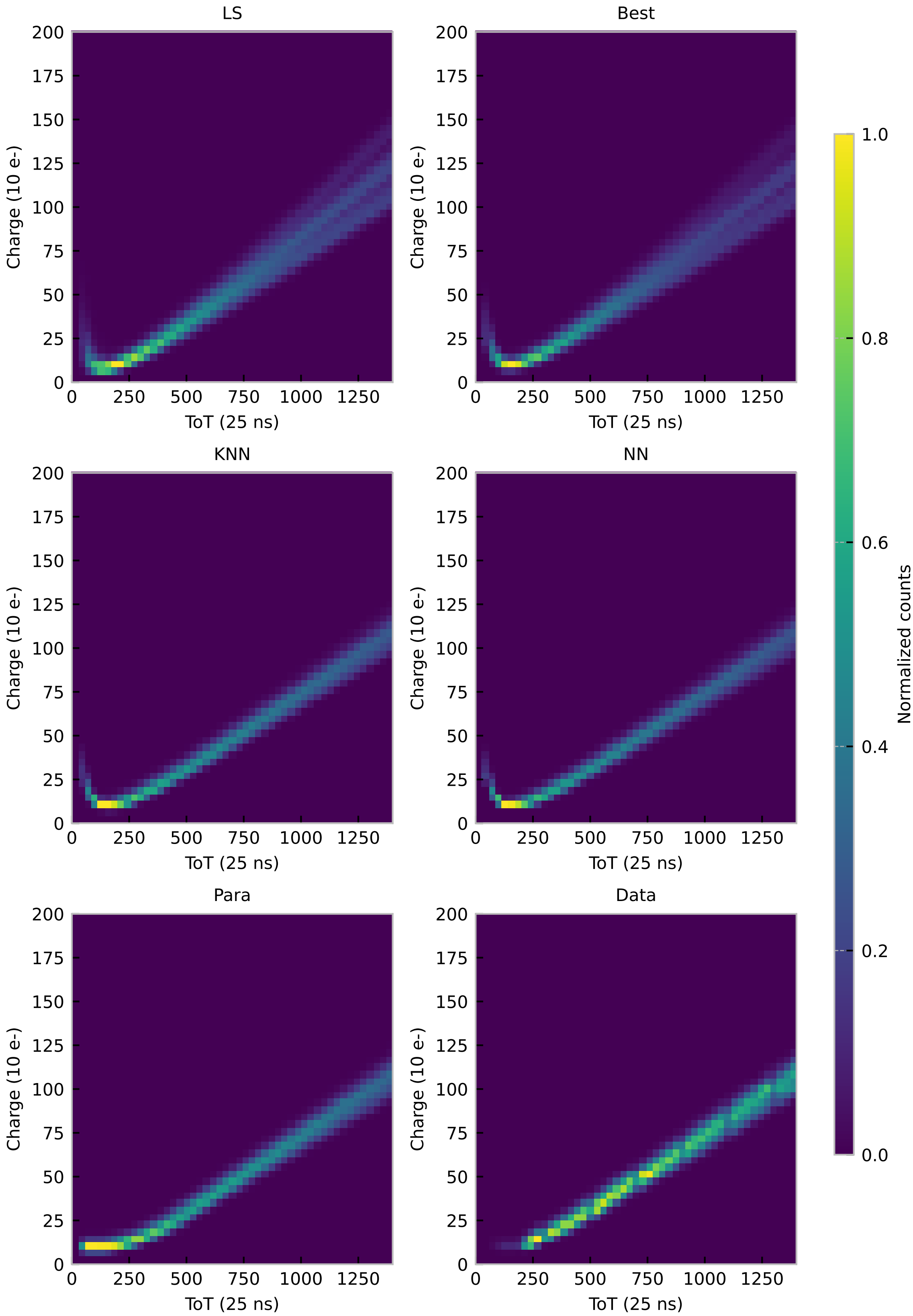

Figure 7: Histogram of different models across chip 7 and 8, showing the spread of different modelling methods compared with the data.

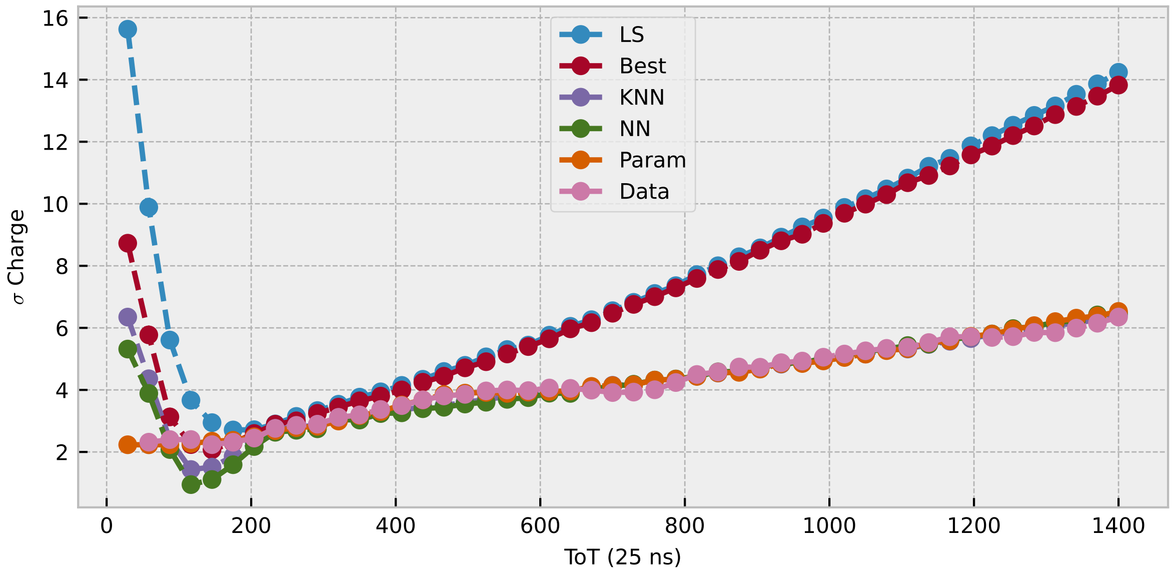

Figure 8: Dispersion plot for different modelling methods.

in a low charge resolution, but the χ^2 results show the parametric fit is less accurate than the more dispersed LS fit.

I believe a reason that the ML-based methods perform well for this task is that by taking in multiple datasets, they are trained on a reasonable range of possibilities for a 'good' fit to the data. This will also mostly constrain the ML fits to a range defined by the dispersion of the best fits. Additionally, the quality of the ML fits will be improved by increasing the amount of data the models are trained on. There are 1024×9 full curves for the training to be performed with currently, so if there is any archival data or future data to be added into the training set this could improve the fitting. The fits have also been performed using a very simple neural network architecture, which could potentially be improved upon. When I tried more complex versions the fit quality became a lot worse, but I did not investigate alternative architectures in depth.

Overall, although using a neural network seems the most optimal method for vector prediction, given the limited amount of data and ability to take further data, using a least squares fit and a KNN regression seems to give a medium - when these two fits are very similar there is a high degree of confidence the fit is good.

Finally, I wanted to look at how the fits of different pixels changed over time. Chip 0 row 100 was analysed for runs taken at three different points over a two year period, and the charge density and resolution plots were created of the KNN and NN fits, and histograms of the parameters for the three different data collections. Due to time constraints I was only able to produce the plots for this question, which are shown in Figure 9, Figure 10 and Figure 11.

Software

The plotting and analysis in this project made extensive use of the Python packages SciPy, matplotlib, NumPy, TensorFlow, scikit-learn, and pandas. Both ROOT and uproot were used for analysis of the calibration run root files.

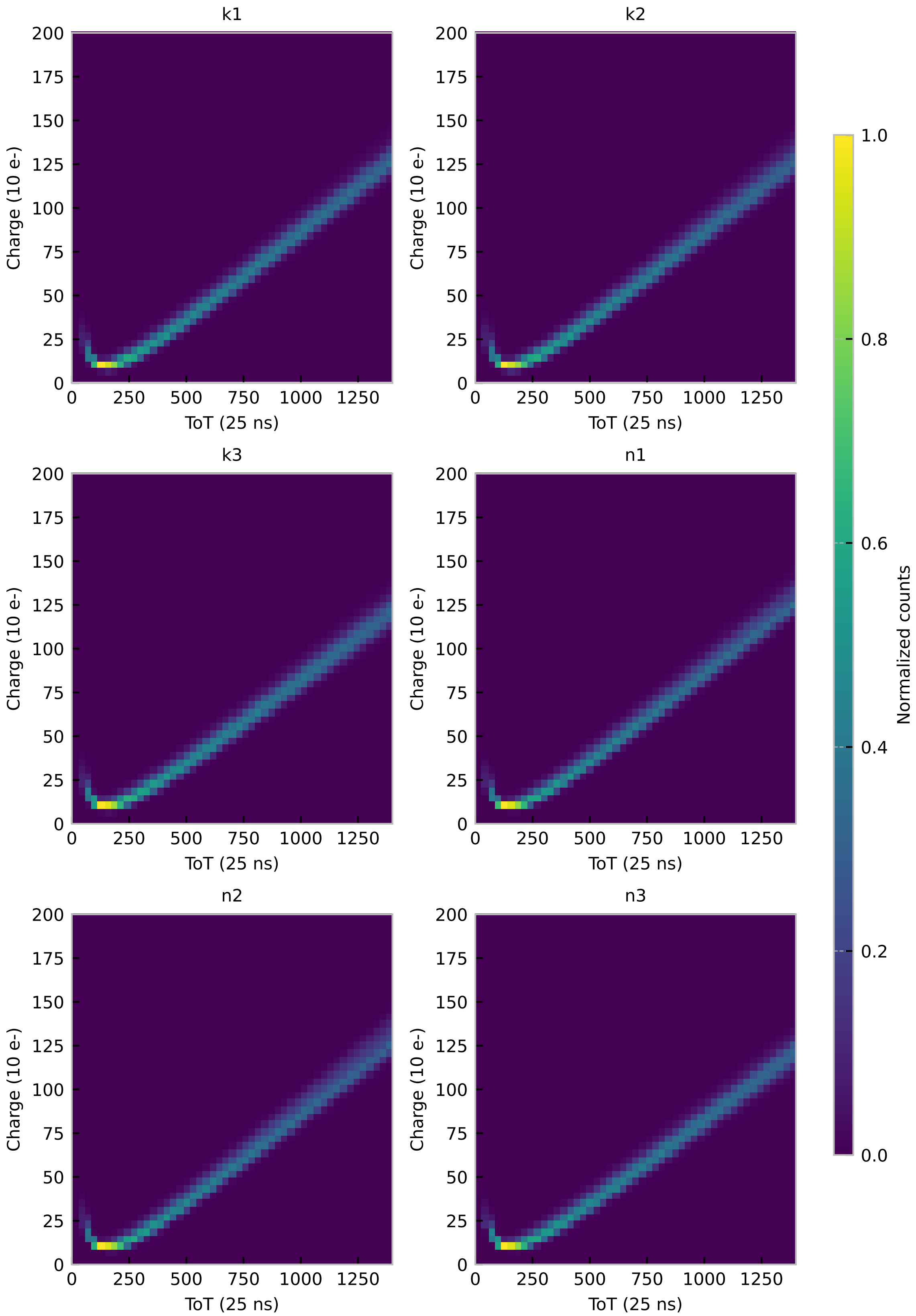

Figure 9: Charge density plot for data taken at three separate points in time. The n labels refer to the fitting completed using a neural network, and the k labels refer to the fitting completed by the k-NN algorithm.

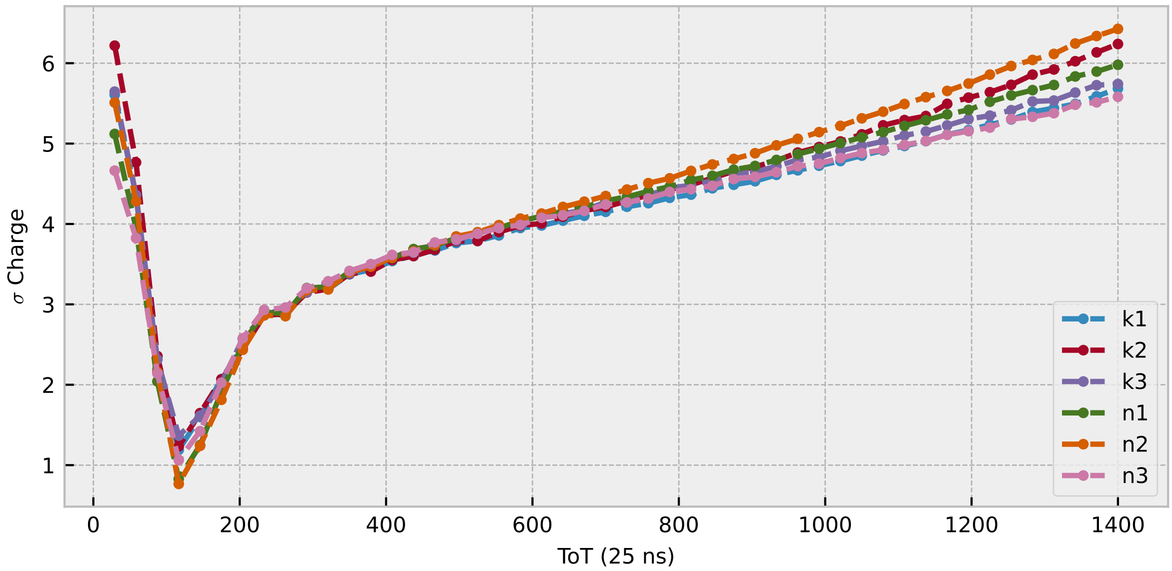

Figure 10: Charge dispersion plot for the three datasets fit by neural networks and k-NN methods.

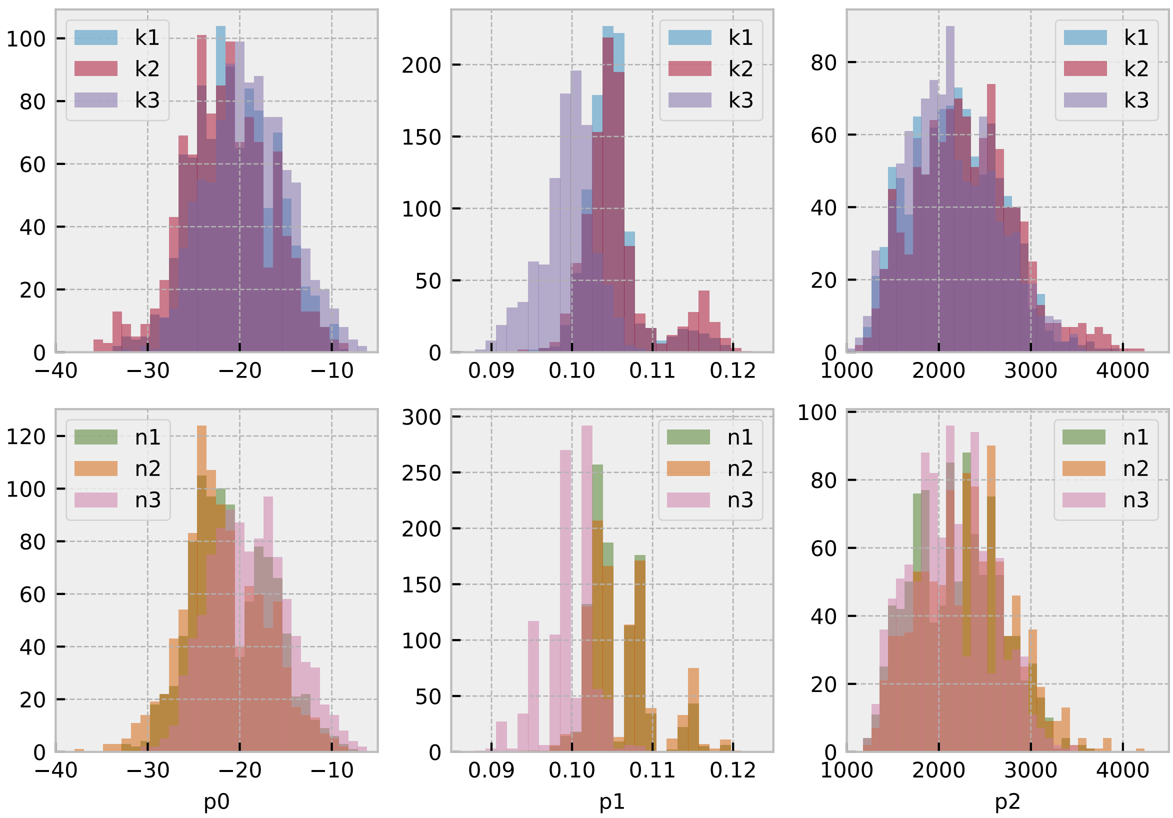

Figure 11: Distribution of fitting parameters plotted using different fitting methods at different points in time. The first dataset shows the most significant different from the first two datasets, and has differences regardless of modelling method used.

Acknowledgements

Thank you to my supervisors Andrea and Ivan for the support and interesting project, and for making a significant amount of time in their schedules to answer my many questions. I would also like to acknowledge the support of the non-member state summer student programme, as this has allowed me to participate in science which is not accessible back at home.

Bibliography

ALICE Collaboration 2024, Journal of Instrumentation, 19, P05062

ALICE ITS ALPIDE development team 2016, ALPIDE Operations Manual

Bolzonella R., et al., 2024, Timing Resolution Performance of Timepix4 Bump-Bonded Assemblies (arXiv:2404.15499), http://arxiv.org/abs/2404.15499

ITS Collaboration 2024, The Calibration of the Upgraded ALICE Inner Tracking System for the LHC Run 3