6 September 2021

CERN Summer Student Report
Accelerating Schottky Analysis II

Lewis Kennedy

Keywords: Schottky noise analysis, chromaticity, tune, longitudal profile, parallel computing, optimization

Abstract

The parallelized L-BFGS-B minimization routine developed in stage 1 was extended to be used in the calculation of beam parameters of real, experimentally obtained spectra. An average evaluation time of **17.2s** was achieved when calculated sequentially using L-BFGS-B. By visualising the function to be minimized and by developing a robust parameter tuning method, further improvements to the evaluation time were made using the Differential Evolution algorithm. A final average evaluation time of **7.9s** per spectrum was achieved.

Contents

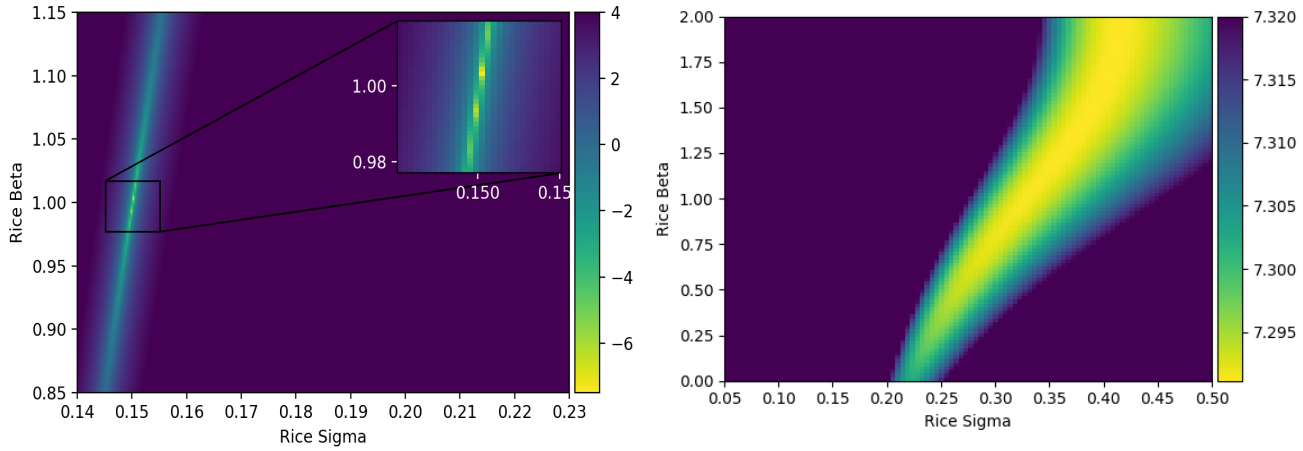
1 Introduction

In the initial stages of the project, we focused exclusively on the optimization and parallelization of the algorithm developed on simulated spectra [1] with the goal of improving the computational efficiency as measured through the evaluation time of beam parameter calculations. In this stage, we build on this further by applying the newly developed algorithm to a real spectra dataset obtained through experiment and measure the performance and validity of our methods in both accuracy and evaluation time. Moreover, we explore the shape of our cost function to determine whether the L-BFGS-B [2] algorithm is sufficient as a minimization routine for our function or if a more vigorous method should be used. Since the L-BFGS-B method uses a simple gradient stepping method, it shows good performance in local minimization problems but is prone to poor performance in problems involving the minimization of a function with multiple minima.

One such method we investigate is the Differential Evolution algorithm [3] which is a population-based metaheuristic search algorithm and while it is not guaranteed to find an optimal solution, the probability of doing so is greatly increased relative to using a local minimization routine on a functional with multiple minima. The Differential Evolution algorithm first carries out an exploration stage of the solution space featuring a large stepping routine before beginning the exploitation stage where candidate solutions are selected and iteratively improved on by using smaller perturbations. A robust parameter tuning process inspired by the methods in [4] is developed to explore the effect of the various Differential Evolution control parameters; Population Number (NP), Differential Weight (F) and Crossover Probability (CR) along with the effect of different mutation strategies and initial population sampling methods on our solution accuracy and evaluation time.

2 Cost Function

To determine the best minimization algorithm for our use case, we created a visualization showing the shape of the function to be minimized. Fig. 1a shows the cost function shape using the algorithm in [1] for a noiseless simulated spectrum at the corresponding expected chromaticity while Fig. 1b shows the cost function shape for a single real spectrum (FIll7435_B1_SPEC_75).



(a) Cost Function for simulated spectrum 1.

(b) Cost Function for Fill7435_B1_SPEC_75.

Figure 1: Multiple minima for both simulated and real spectra cost functions at their respective expected chromaticities.

Since the simulated function appears to feature a number of local minima, local minimization routines may not be sufficient for finding the true global minimum. Conversely, the cost function of the real spectrum as shown in Fig. 1b shows a relatively flat, wide surface without any clear or visible global minima, making it difficult to determine at least visually where the true minimum lies. Numerically, it may also be very difficult to determine the rice pairing giving the true minima since there are a great number of pairs minimizing the function relatively well. Since at this stage it is not clear which minimization method is the most optimal, both methods were compared and tested on the Fill7435 spectra using sequential methods with their performance and validity measured.

3 Differential Evolution

3.1 Parameter Tuning

The performance of the `scipy.optimize.differential_evolution` algorithm was tested against the simulated spectra and had an initial evaluation time of around 220s per spectrum which was significantly slower than the L-BFGS-B algorithm developed in stage 1 at 83s per spectrum. We then began exploring the various Differential Evolution parameters such as Differential Weight (F), Crossover Probability (CR) and Population Size (NP) and found several studies suggesting rough estimates for optimal values of these parameters [3] [5]. The problem however is that the parameter values are very case specific and are heavily dependent on the function being minimized. We therefore developed a parameter tuning method using visualization techniques to determine the ideal parameters with computational efficiency and accuracy in mind.

As shown in Fig. 2, maps of the cost function and computational cost were generated as a function of CR and F for the given population size. The function cost is measured as the sum of the absolute differences for chromaticity of both transverse planes against their relative expected values and the computational cost is measured as the evaluation time. Finally, the logarithm of both function cost and computational cost are taken to better highlight changes in the heatmaps.

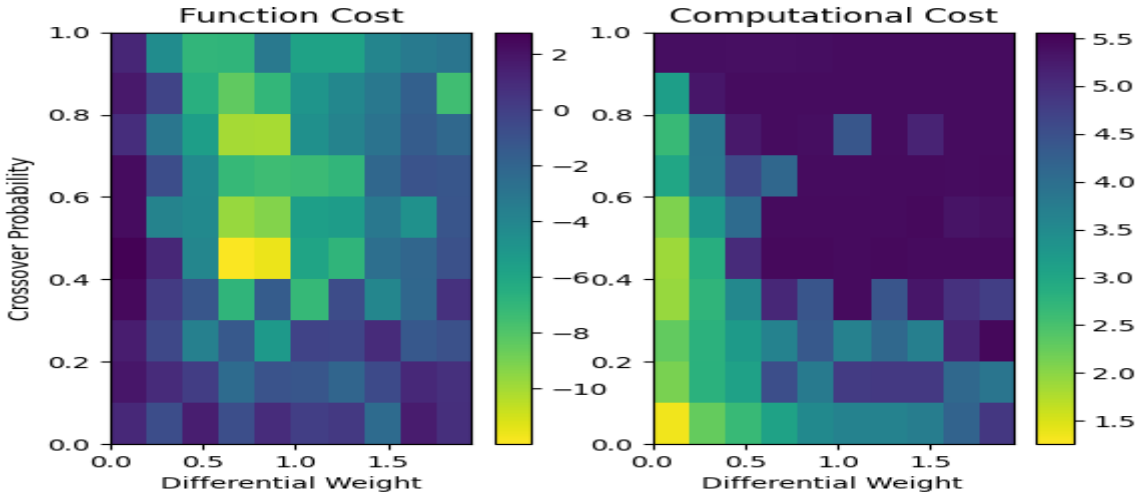


Figure 2: Visualisation of costs for Differential Evolution algorithm parameters.

An optimal population size of NP=8 was determined which allows for the parallel distribution of each individual population member across the 8 CPU cores used in testing. Other population sizes were also tested and as expected for all NP>8 the evaluation time was slower but more accurate. For cases where NP<8 we see similar but still slower evaluation times along with reduced accuracy. This is mostly due to having a reduced search space due having a lower number of initial guess population members. It is therefore important to always utilize the maximum number of CPU cores available to maximize the initial guess population and search space without evaluation time penalties.

A crossover probability of around CR=0.3 and a differential weight F $\in$ [0.05, 0.2] were obtained through inspection of Fig. 2. It is sufficient at this point to pass this differential weight interval as a tuple to our mutation parameter in `differential_evolution` and, in doing so, making use of dithering which uses adaptive altering of the mutation parameter in the given range for faster convergence [6]. Due to time constraints, a callback function was created and passed into the algorithm used to create Fig. 2 in order to implement an upper time boundary. This is characterised by the flat, dark sections of the computational cost heatmap where any solutions with an evaluation time greater than the time limit were discarded. A more robust parameter tuning process would have been carried out had there been sufficient time and would have allowed for the testing of slower but more accurate solutions.

4 Application of Methods to Real Spectra

4.1 Differential Evolution

The values obtained in the parameter tuning stage were then used in the calculation of chromaticities and Rice values for the real spectra as obtained through experimental methods. The algorithm was then modified to take the solution array from the previous spectrum as the initial guess for the next spectrum, with the initial population passed as a matrix of uniformly distributed initial guess arrays centered around the previous spectrum's solution.

As shown in Fig. 3, the results for the Fill7435 spectra converge to a relatively stable solution after an initial period of slight instability when using the default DE/best/1 mutation strategy with uniformly distributed initial population noise.

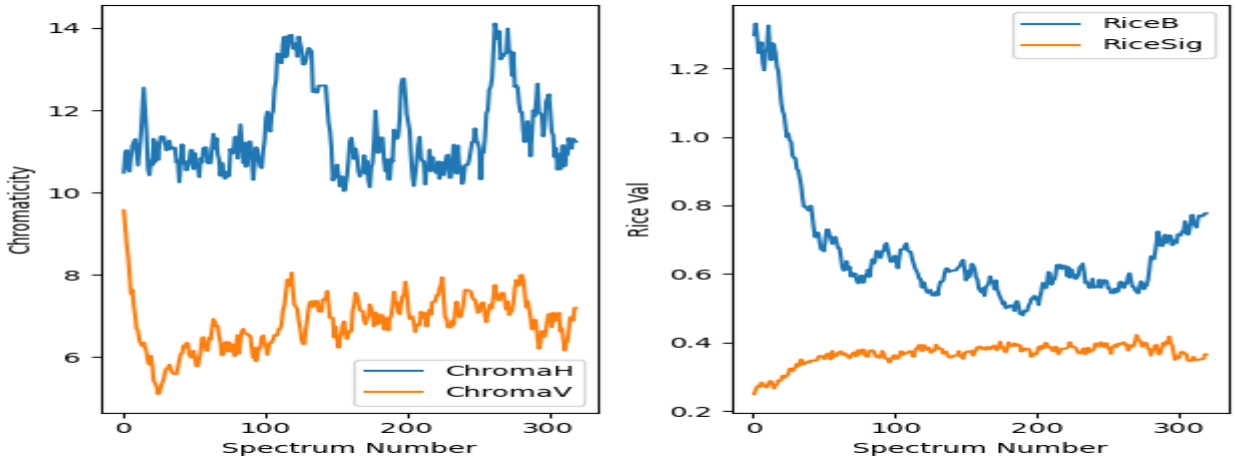
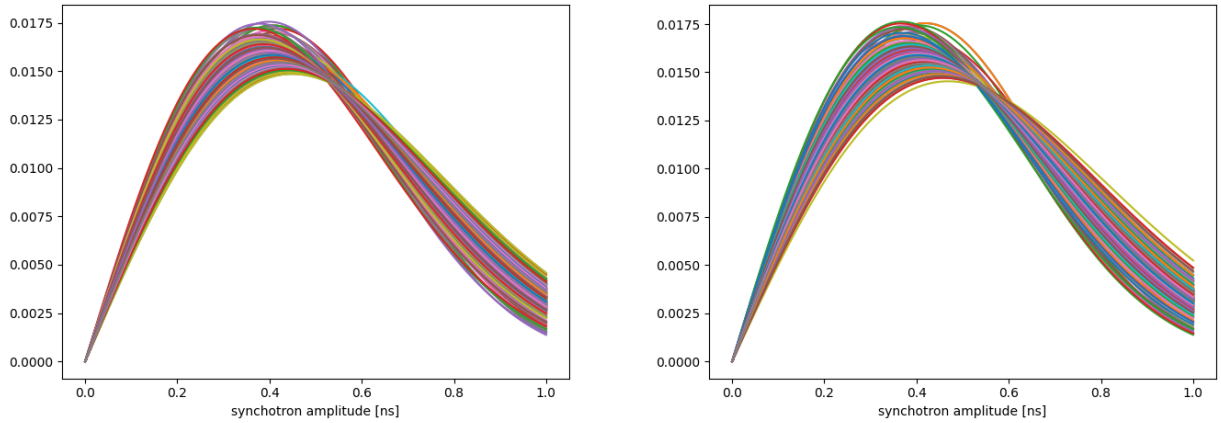


Figure 3: Chromaticity and Rice solutions for Fill7435 spectra using the default mutation strategy, DE/best/1bin where initial spectrum is Fill7435_B1_SPEC_0.

4.1.1 Mutation Strategies

Using the default DE/best/1 method there is some probability that the routine fails to find the global minimum as a result of choosing only the very best solution from the parent population, effectively reducing the search space. We therefore investigated the performance of other mutation strategies and checked their validity through inspection of the resulting synchotron amplitude distributions given by the rice parameter solutions for each strategy.

Changing from the default method as shown in Fig. 4a to DE/best/2, which increases the search radius of the solution space by selecting and perturbing the 2 best parent population members instead of just 1, we see only very minor changes in the synchotron amplitude distribution as shown in Fig. 4b.

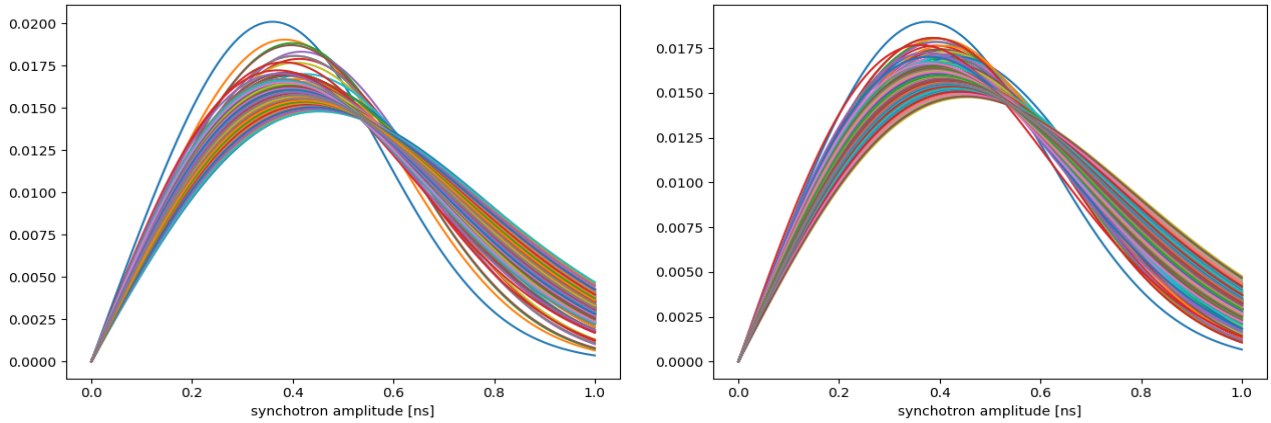


(a) DE/best/1bin.

(b) DE/best/2bin.

Figure 4: Synchotron amplitude distributions based on fitted rice parameters. Each line corresponds to one spectrum.

Other mutation strategies such as DE/rand/1 and DE/rand/2 additionally randomize candidate solution selection from the parent population, effectively increasing the search space and the probability of finding the true global minima. Typically they suffer from slower convergence rates as a consequence and in our case, both methods in Fig. 5a and Fig. 5b performed notably poorer than other mutation strategies.

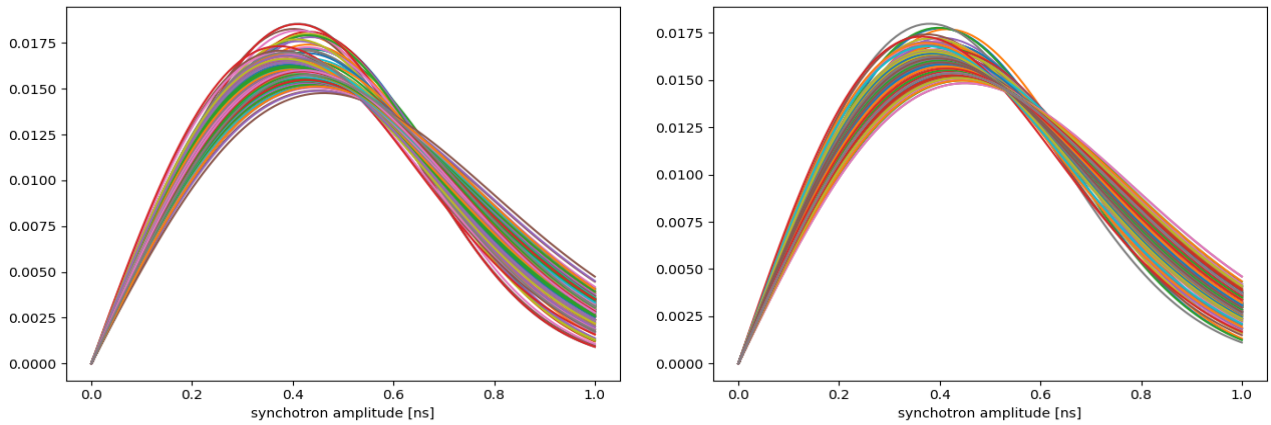


(a) DE/rand/1.

(b) DE/rand/2.

Figure 5: Synchotron amplitude distributions based on fitted rice parameters. Each line corresponds to one spectrum.

Finally, the DE/current-to-best/1 and DE/rand-to-best/1 strategies are somewhat of a middle ground between the DE/best/x and DE/rand/x strategies, providing a compromise between exploitation of the best solution and exploration of the solution space.



(a) DE/current-to-best/1.

(b) DE/rand-to-best/1.

Figure 6: Synchotron amplitude distributions based on fitted rice parameters. Each line corresponds to one spectrum.

Taking into account the synchotron amplitude distribution shapes, the graphs shown in Appendix A and runtimes, it was determined that the DE/best/1bin mutation strategy shown in Fig. 4a was the most optimal method for Differential Evolution with an average evaluation time of 7.9s per spectra calculated sequentially.

4.2 L-BFGS-B

The algorithm developed in the previous stage of the project was also similarly applied to the FIll7435 data for comparison of minimization methods. Shown in Fig. 7 are the results using the parallel L-BFGS-B method.

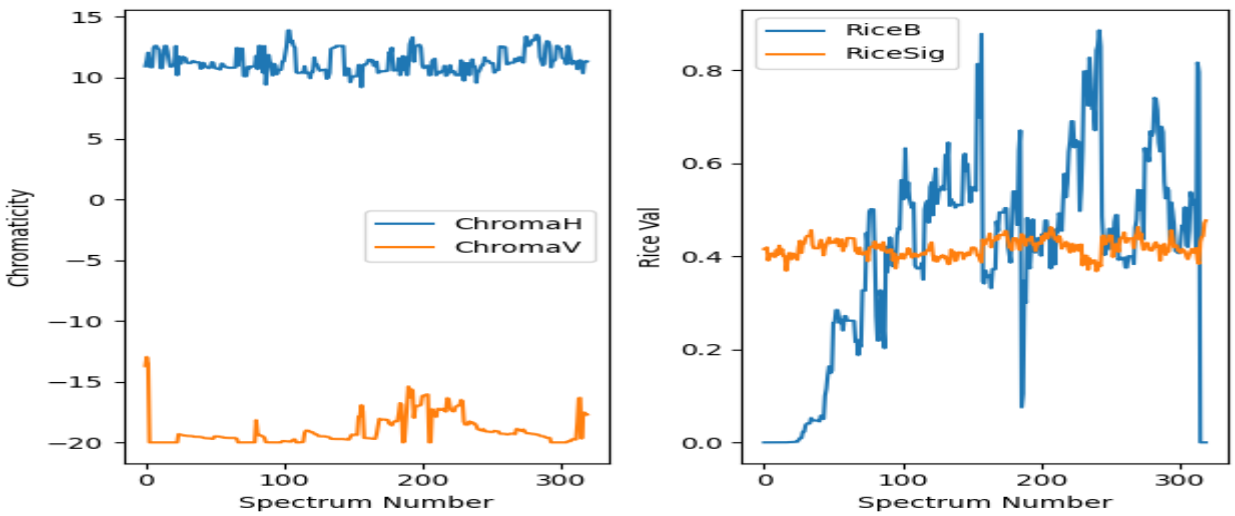


Figure 7: Chromaticity and Rice solutions for Fill7435 spectra using OptimParallel(L-BFGS-B) and Fill7435_B1_SPEC_0 initial guess.

In Fig. 8, we begin the sequential process on the 2nd spectrum of Fill7435 instead, providing a change of initial guess, and observe the sensitivity of passing a wrong initial guess using this method.

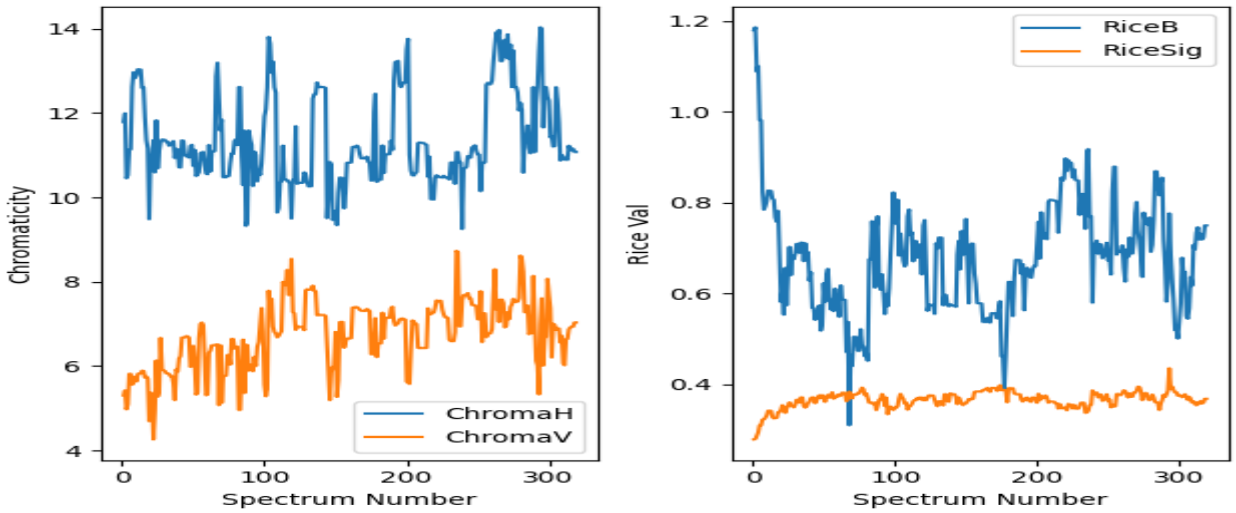
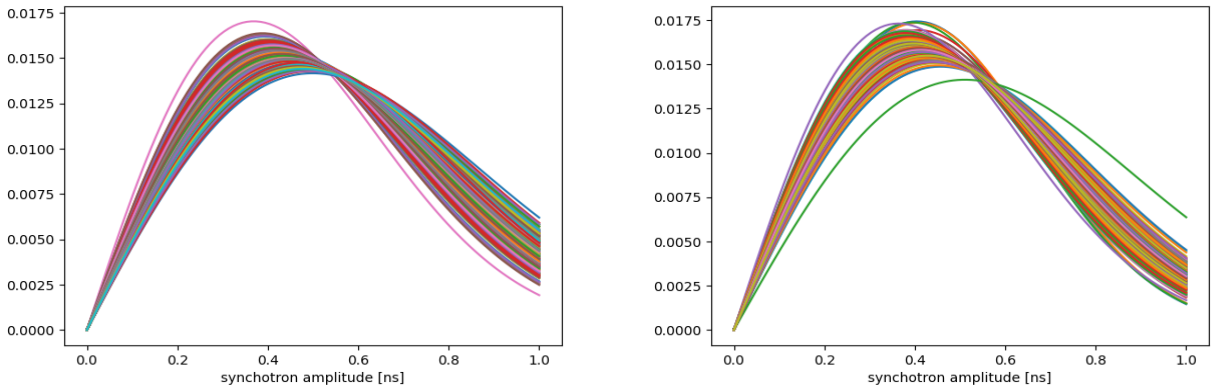


Figure 8: Chromaticity and Rice solutions for Fill7435 spectra using OptimParallel(L-BFGS-B) and FILL7435_B1_SPEC_1 initial guess.

For the ChromaV solutions, we see that the L-BFGS-B returns wrong results which is likely due to the sensitivity of the method to the initial guess. If this initial guess is wrong, then the routine appears to get trapped in a local minima around -19 for the duration of the evaluation for all 320 spectra as in Fig. 7. However, regardless of the initial spectrum guess the synchotron amplitude distribution doesn't change much as shown in Fig. 9a and Fig. 9b.



(a) Synchotron Amplitude Distribution with Fill7435_B1_SPEC_0 as initial guess.

(b) Synchotron Amplitude Distribution with Fill7435_B1_SPEC_1 as initial guess.

Figure 9: Synchotron amplitude distributions based on fitted rice parameters. Each line corresponds to one spectrum.

The final evaluation time of the L-BFGS-B algorithm on the sequential calculation of parameters was 17.2s. Hence, the Differential Evolution algorithm appears to be a much faster and more reliable method for computing the solution parameters sequentially as by changing the initial guess we still see accurate solutions as shown by the final results in Fig. 10.

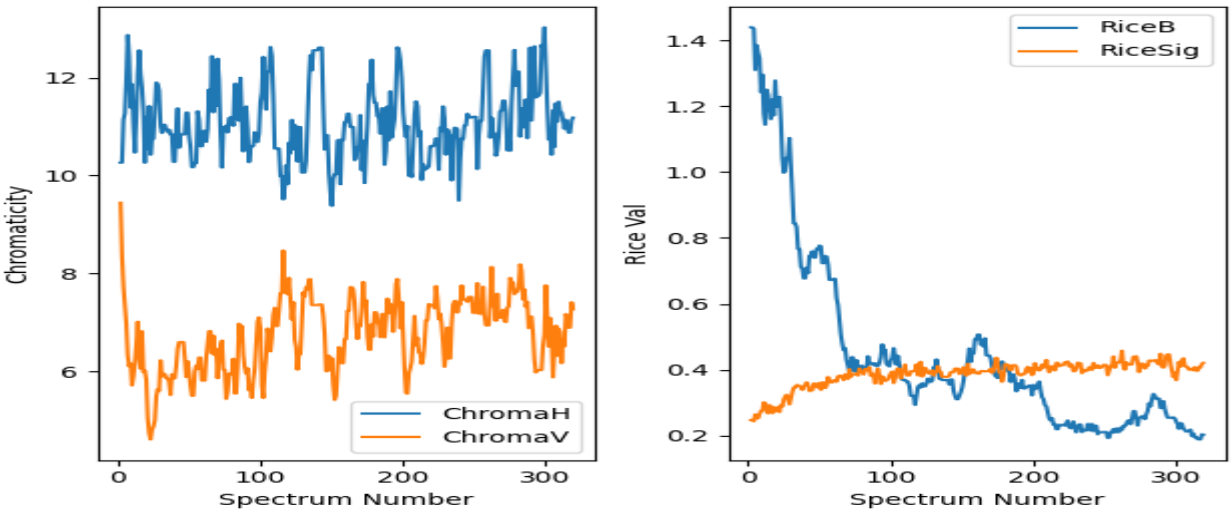


Figure 10: Chromaticity and Rice solutions for Fill7435 spectra using DE/best/1bin with Fill7435_B1_SPEC_1 initial guess.

5 Conclusions

Visualising the cost function allowed us to identify the presence of a number of local minima and determine that the most optimal minimization routine for our noiseless, simulated function is the Differential Evolution algorithm. When applied to real spectra, this method showed significantly improved evaluation time at an average of **7.9s** per spectrum vs **17.2s** with L-BFGS-B, when spectra are analysed sequentially, with the result of the previous one being used in the analysis of the following. Moreover, when using the L-BFGS-B method, the solutions were inaccurate due to being stuck in local minima early in the sequential process while the Differential Evolution algorithm managed to find minima closer to the true global value. Hence, the Differential Algorithm also outperformed the algorithm developed in stage 1 when applied to real spectra.

Through investigating the effect of various parameters on computational cost and solution accuracy it was noted that by increasing population size (NP) we see improved solution accuracy but for any NP greater than the available CPU count we see reduced evaluation time performance due to the surplus population queue. However, as this method is very easily scalable to systems with a larger number of cores and is not limited by the dimensionality of the problem as is the case with OptimParallel, we would see increased solution accuracy and likely reduced computation time on a system with more CPU cores.

References

[1] A. S. Hassan and L. C. Kennedy, "Accelerating Schottky Analysis," Aug 2021.

[2] R. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," SIAM Journal of Scientific Computing, vol. 16, pp. 1190–1208, Sept. 1995.

[3] R. Storn, "On the usage of differential evolution for function optimization," in Proceedings of North American Fuzzy Information Processing, pp. 519–523, 1996.

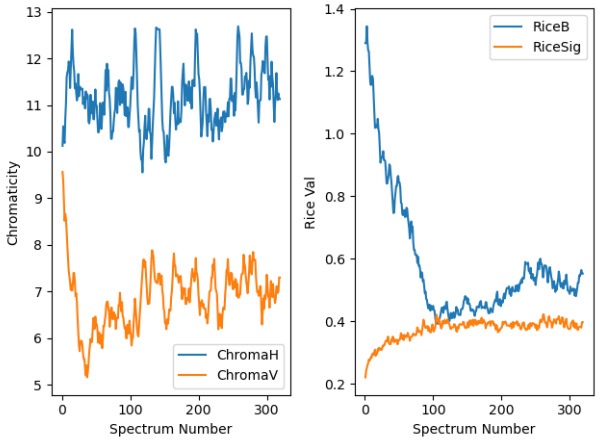
[4] K. Y. Kok and P. Rajendran, "Differential-evolution control parameter optimization for unmanned aerial vehicle path planning," PLOS ONE, vol. 11, pp. 1–12, 03 2016.

[5] R. Gämperle, S. D. Müller, and P. Koumoutsakos, "A parameter study for differential evolution," in WSEAS Int. Conf. on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, pp. 293–298, Press, 2002.

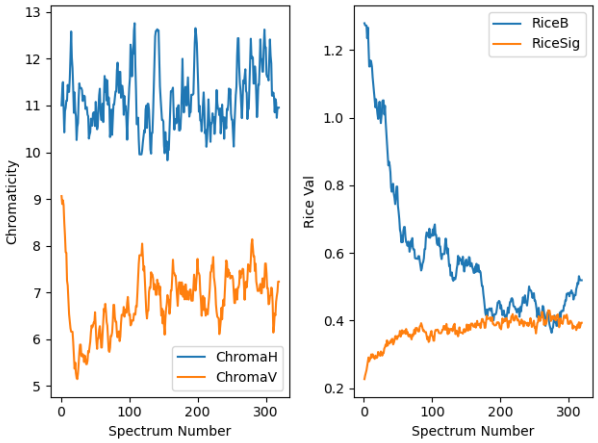
[6] D. Dawar, "Differential evolution with dither and annealed scale factor," 12 2014.

Appendices

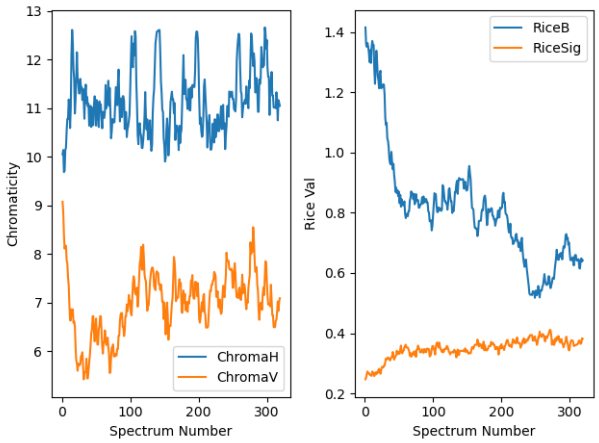
A Solutions



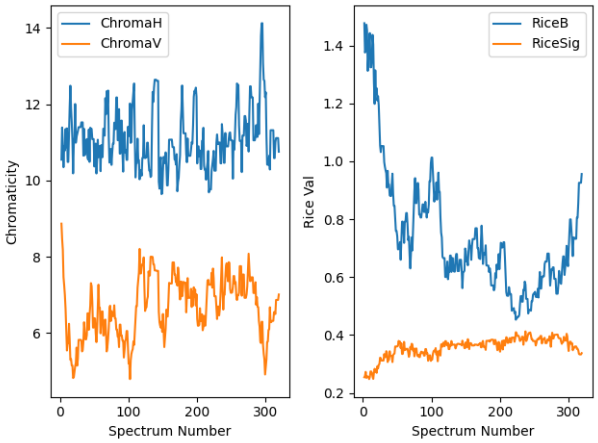
(a) DE/rand/1bin



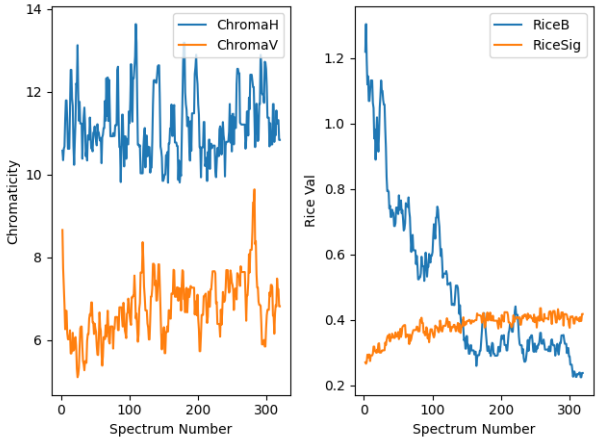
(b) DE/rand/1bin



(c) DE/best/2bin



(d) DE/current-to-best/1bin



(e) DE/rand-to-best/1bin

Figure 11: Solutions per mutation strategy.