AMSI VACATIONRESEARCH SCHOLARSHIPS 2020–21

*Get a Thirst for Research this Summer*

# Piecewise Rational Approximation

Duc Minh (Bob) La

Supervised by Dr. Julien Ugon and Dr. Reinier Díaz Millán

Deakin University

22/02/2021

Vacation Research Scholarships are funded jointly by the Department of Education, Skills and Employment and the Australian Mathematical Sciences Institute.

# Contents

# 1 Abstract

The report will reflect the work on alternative projection method for rational approximation. The method was investigated for further development to be integrated with more complicated method. The method introduced great potential for the work of piece-wise rational approximation as it has fast processing time and an somewhat easy implementation and theories behind. Nevertheless, there are still certain drawbacks needed to be addressed and fixed with further research to improve its accuracy.

## 2 Introduction

Approximation is a developing field in mathematics that has quite a range of application especially in computer calculation (Powell et al. 1981). It focuses on approximate complex function with simpler functions to achieve better run-time while still can maintain the accuracy to a certain degree (Petrushev and Popov 1987). Therefore, the problem can be expressed as

$$minimise\ sup_{t\in[a,b]}\ |f(t)-g(t)|$$

with $f(t)$ as the goal function (the function that need to be approximated) and $g(t)$ as the result function (the approximated function). $g(t)$ belong to a class of an specific functions, such as, polynomials, rational functions, trigonometric functions, etc.

There are many ways to approach the problem. The one this paper focuses on would be alternative projection. The method was firstly introduced by Neumann (1949). The reason for this choice was because the method is simple, fast but still can be very applicable in many cases such as economic analysis (Judd 1996) or computerized tomography (Bauschke and Borwein 1996) or linear prediction theory (Badea and Seifert 2016).

Though the original was focused on creating a polynomial for the result, the paper would take a different turn to a rational function. The reason for this change was because rational approximation is proved to be able to offer high flexibility and more suitable with extremely complex function (Blair, Edwards, and Johnson 1976). The problem then can be transformed as: for $p(t)=\sum_{n=0}^{n}\alpha_n t^n$ (so so p belongs to the set of all polynomials with degree less or equal to n with $n\geqslant 0$) and $q(t)=\sum_{m=0}^{m}\beta_m t^m$ (so q belongs to the set of all polynomials with degree less or equal to m with $m\geqslant 0$), the problem would be minimising on the set of polynomials

$$minimise\ sup_{t\in[a,b]}\ \left|f(t)-\frac{p(t)}{q(t)}\right|.$$

Thus, the goal of the approximation process would be determining the coefficients of the polynomials: $A=(\alpha_0,\alpha_1,...,\alpha_n)$ and $B=(\beta_0,\beta_1,...,\beta_n)$. This has turn the problem into a linear combinations of basic functions which help it become easier to approach and solved (Millán, Sukhorukova, and Ugon 2020).

## Authorship

The work and data presented in this report was done by Duc Minh La (Bob) with the support and advice of Dr. Julien Ugon and Dr. Reinier Diaz Millan.

## 3 Alternative Projection

The idea of the algorithm was first presented by Neumann (1949). The idea grounded the base for finding the intersection of hyper-planes. The work in this report take the algorithm and test on the rational approximation.

To explain, the algorithm has approached the problem in a linear way. To be more specific, for a selected point named $x_0$, with a perfect approximation result, the difference between the goal function and the approximated function would be 0:

$$f(x_0) - \frac{p(x_0)}{q(x_0)} = 0.$$

This then can be transformed into:

$$p(x_0) - f(x_0)q(x_0) = 0$$

or turning this into vector-liked form, we can have:

$$(A, X_n) - f(x_0)(B, X_m) = 0$$

with $X_n = (x_0^0, x_0^1, ..., x_0^n)$ and $X_m = (x_0^0, x_0^1, ..., x_0^m)$. As $x_0$ and $n, m$ are known or predefined values, the above equation is actually a formula for a hyper-plane in $\mathcal{R}^{n+m+2}$.

Therefore, with this showed, for the approximation problem, for each input there will be a corresponding hyper-plane. All of these hyper-planes will exist in the same space, which in this case of rational approximation would be $\mathcal{R}^{n+m+2}$. As any point on a hyper-plane would satisfy a perfect approximation at the corresponding input, the intersecting point of all the hyper-planes would hold the needed result for the approximation (Badea and Seifert 2016). The way to find that point, which is also the algorithm, is to use projection. The method of alternating orthogonal projections is well-known and has many researches about it. The solutions presented in this report was based on Bauschke and Borwein (1996) and Deutsch (1992). In details, the algorithm starts at a random point then project that point on the first hyper-plane to find the projected point on the first hyper-plane, and from that point, the algorithm would continue with the second hyper-plane until it goes through all the hyper-planes (the order of the hyper-planes would be predefined), and then repeat the whole process again from the last point of the last iteration. The number of iterations depends on a predefined value named error rate which is the value of the distance between a new projected point on hyper-plane with the prior projected point on that same hyper-plane from the most recent iteration (Deutsch 1984). The reason this value was chosen to be the stopping criteria was because that it reflects how near the algorithm to the convergent point, in other words, the intersecting point; hence, the smaller the error rate, the more accurate the final result. An example of the algorithm can be seen below from figure 1 for the case of 3 hyper-planes.

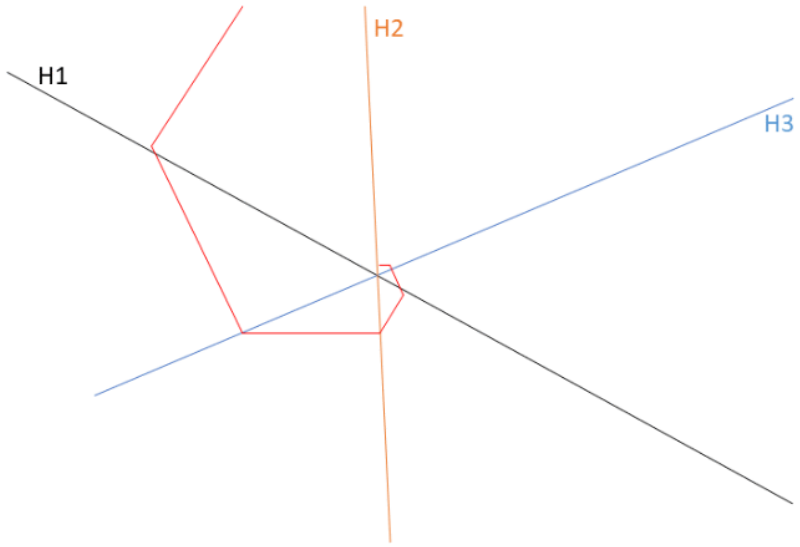



Figure 1: The algorithm run when there is an intersecting point.

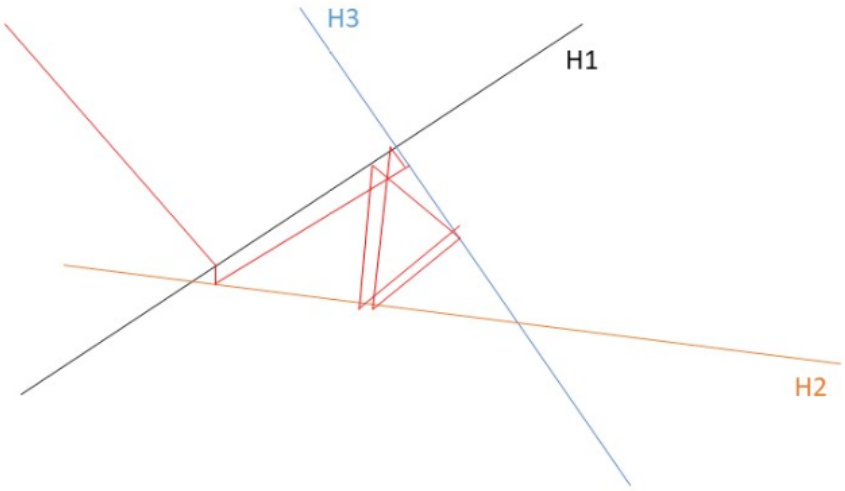



Figure 2: The algorithm run when there is no intersecting point.

However, the case of having an intersecting point is very rare unless the goal function itself is a rational function made of 2 polynomials. In most of the cases, the hyper-planes would not have any intersection. With this case, instead of having a solid result, the algorithm will return a list of convergent points on each hyper-plane as can be seen from figure 2. From this list of convergent points, the final result can be calculated which is the center point of all the points (the point that has the shortest distance to all of the point). The formula would be:

$$final\ point = \frac{\sum resulted\ points}{number\ of\ hyperplanes}.$$

From the description of the algorithm, it can be seen that the algorithm would take 4 inputs:

- h: the number of hyper-planes (which will be determined by how we selecting the values)
- n: the degree of the numerator of the resulted function
- m: the degree of the denominator of the resulted function
- r: the error rate

Thus, the run-time of the algorithm for each iteration is expected to be: $O(h(n+m+2))$ and the number of iteration would be based on $r$. However the actual run-time can be affected by other factors such as the processing power of the computer.

To test and evaluate the application of the alternative projection algorithm for rational approximation, 3 experiments were done. Firstly, testing the algorithm with a simple rational function to check the accuracy of the algorithm as well finding any interesting traits as for this case, there should be an intersection. Secondly, testing with a complex function to see how the algorithm will work in a more real-case scenario. Lastly, testing with when the function has 2 inputs to see how the algorithm will work in a more complex case with many dimensions.

# 4 Experiments

## 4.1 With a simple function

For this experiments, the function used was:

$$f(t) = \frac{t^3+3}{t^2+1}.$$

The experiment was done by testing with different groups of inputs to see how they affect the final results. $(n, m)$ is the value of the aforementioned pair $n$ and $m$, the same as error rate for $r$. For the step from 0 to 3, it means that in the range of $[0, 3]$, how many points will be selected to create the hyper-planes. The keys to evaluate them were the means of the differences between the goal function and approximated function on each selected points for the accuracy and also the computation time. In addition, as this is a rational function

created by 2 polynomials, it is expected to have an intersection when $n \geqslant 3$ and $m \geqslant 2$. The results of the experiment can be seen from the figure 3 below.

From the results of figure 3 it can be seen that, higher value of n and m would create better results. The same case when we increase the number of hyper-planes or decrease the error rate. However, combining with the computation time, there is trade back as better results means slower computing. Based on that, increasing n and m would be considered the best as the computing time is not much difference but the results increased moderately which also can be seen from 4 when the differences are quite visible. However, it is expected that all the solutions are the same when $n \geqslant 3$ and $m \geqslant 2$; hence, increasing n and m would make longer run-time but not better accuracy. Regarding the number of hyper-planes, increasing it is quite time-consuming yet it is not quite efficient as can be seen from 5 when there are almost no differences between step 0.01 and and step 0.001. Interestingly, for the case of error rate, it is the most time-consuming one when it can reach to more than 600 seconds when the error is $10^{-8}$, yet, it is the attribute that introduce the most significant change in the results. As can be seen from 6, when the error rate is $10^{-5}$ and $10^{-8}$ the 2 lines are almost identical and the max difference also goes from 0.2 to around 0.004 then $-2 \times 10^{-5}$ correspondingly.

To explain the result, as this is a rational function, higher number of hyper-planes would not make any difference as only few number of hyper-planes at certain points can tell the direction of the function. Regarding the pair n and m, it is easy to understand as higher numbers mean they will offer more flexibility for approximation but when it goes over the certain point (in this case is when $n \geqslant 3$ and $m \geqslant 2$) the accuracy will stay unchanged. In terms of the error rate, as this case is expected to have a intersecting point, higher error rate means that the final point will be more extremely closed to the intersecting point, hence, presenting significantly better results.

### 4.2 With a complex function

For this experiment, the used function was a complex *Sine* function which will introduce a more real-life application of this method. The methodology would be the same as the case of simple function. The final results can be seen below from figure 7.

In general, the method did not work well with an complex function. As can be seen from the image of figure 8, 9 and 10 the approximated cannot follow the extreme fluctuation of the goal function but only more like a line to run through. However, based on the means column of figure 7 it is still somewhat acceptable and also the max differences are not too extreme (around 5 only).

In details, in this case, the results revealed a fascinating fact that higher m and n actually does not mean better result such as the case of the $(m, n) = (3, 3)$ in figure 7. The same results also can be seen from figure 8 when there are not much difference between each cases. Furthermore, regarding the error rate, the result does not change much between $10^{-5}$ and $10^{-8}$; hence, unlike in the previous case, the the error rate does not have

| no | (n, m) | step from 0 to 3 | error rate | computation time (s) | iteration times | the means of f(t) - g(t) |
|---|---|---|---|---|---|---|
| 1 | ***(2,2)*** | 0.1 | 1.00E-03 | 0.12 | 411 | 3.00E-02 |
| 2 | ***(3,3)*** | 0.1 | 1.00E-03 | 0.54 | 194 | 1.98E-03 |
| 3 | ***(4,4)*** | 0.1 | 1.00E-03 | 0.59 | 365 | -8.70E-04 |
| 4 | (3,3) | ***0.01*** | 1.00E-03 | 0.7 | 201 | 2.32E-03 |
| 5 | (3,3) | ***0.001*** | 1.00E-03 | 5.83 | 201 | 2.41E-03 |
| 6 | (3,3) | 0.1 | ***1.00E-05*** | 4.29 | 13976 | -1.08E-05 |
| 7 | (3,3) | 0.1 | ***1.00E-08*** | 663.67 | 1983536 | 1.37E-06 |

Figure 3: Results for simple function.

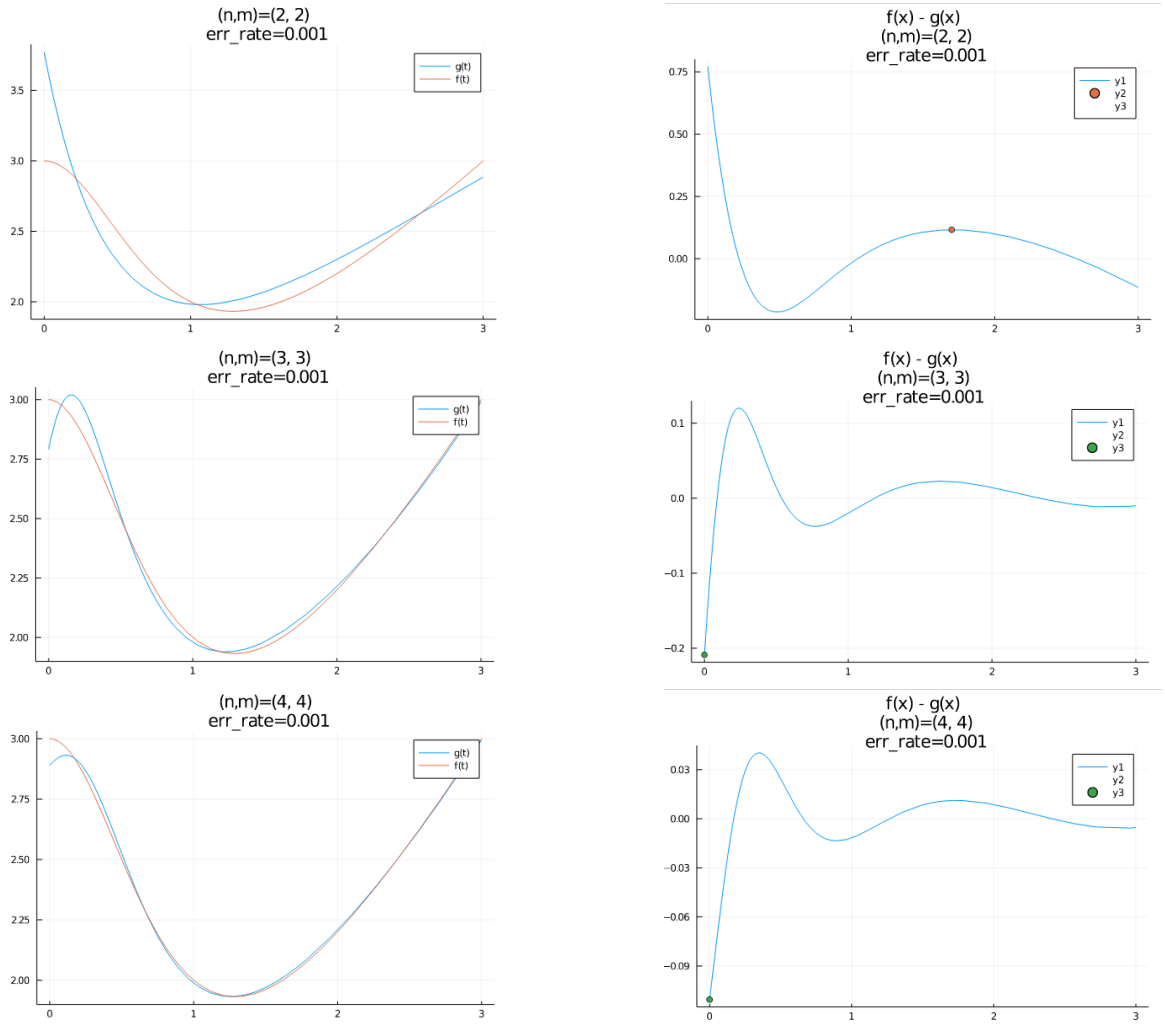
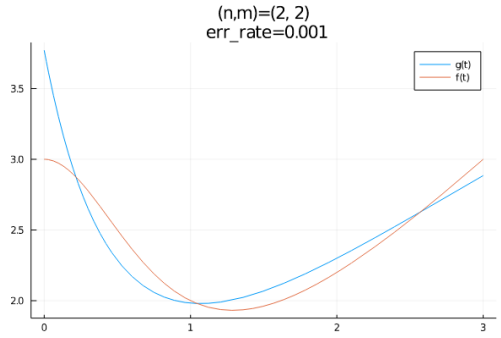
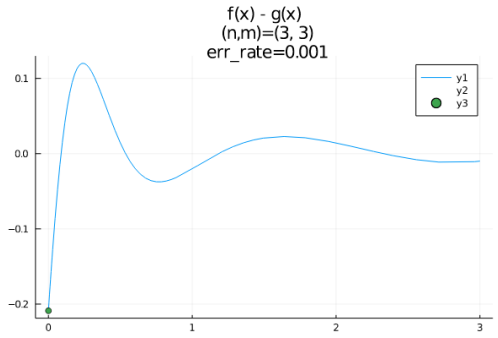


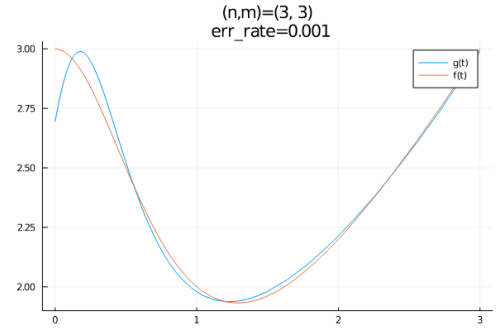

Figure 4: Comparing 2 functions based on (n,m).



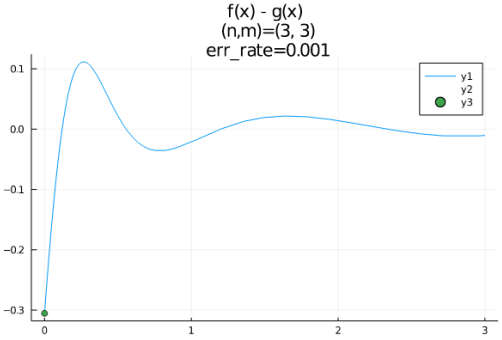

(a) step: 0.1



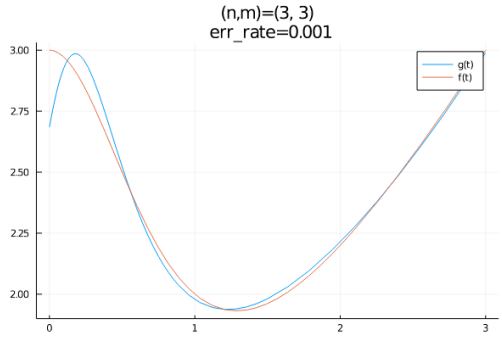

(b) step: 0.1



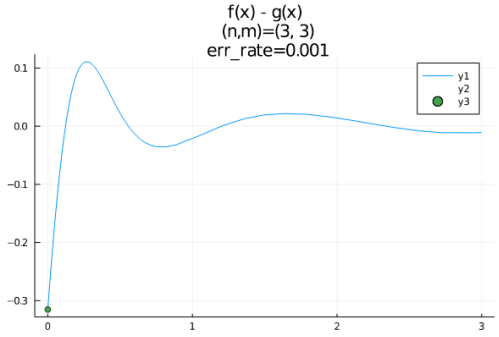

(c) step: 0.01




(d) step: 0.01




(e) step: 0.001




(f) step: 0.001

Figure 5: Comparing 2 functions based on number of hyper-planes.

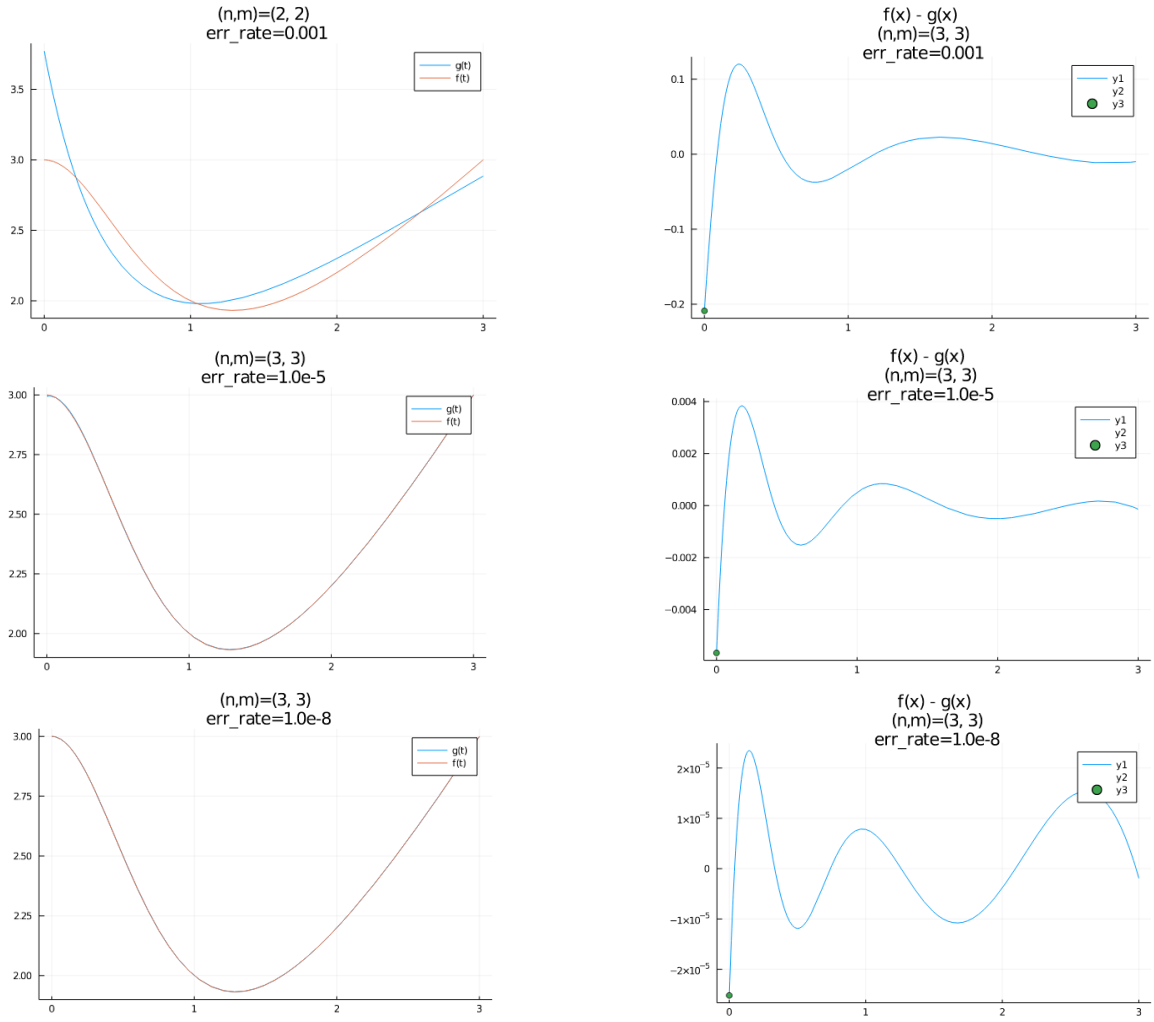



Figure 6: Comparing 2 functions based on the error rate.

| no | (n, m) | step from 0 to 3 | error rate | computation time (s) | iteration times | the means of f(t) - g(t) |
|---|---|---|---|---|---|---|
| 1 | ***(2,2)*** | 0.1 | 1.00E-03 | 0.36 | 36 | 0.41 |
| 2 | ***(3,3)*** | 0.1 | 1.00E-03 | 0.44 | 28 | 1.42 |
| 3 | ***(4,4)*** | 0.1 | 1.00E-03 | 0.45 | 62 | 0.83 |
| 4 | (3,3) | ***0.01*** | 1.00E-03 | 0.53 | 48 | 0.62 |
| 5 | (3,3) | ***0.001*** | 1.00E-03 | 4.26 | 144 | 0.09 |
| 6 | (3,3) | 0.1 | ***1.00E-05*** | 0.21 | 608 | 0.75 |
| 7 | (3,3) | 0.1 | ***1.00E-08*** | 0.49 | 1682 | 0.72 |

Figure 7: Results for complex function.

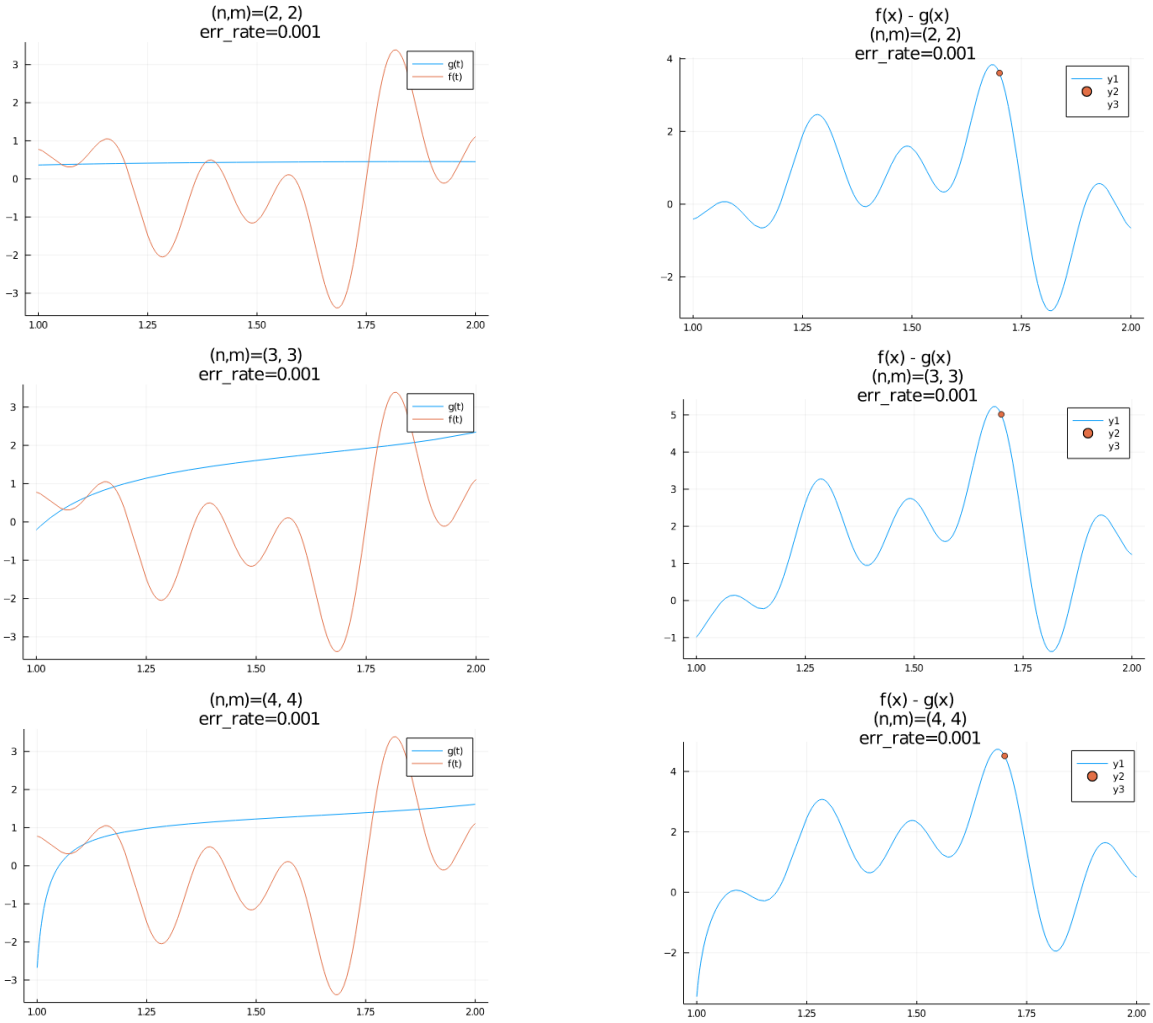
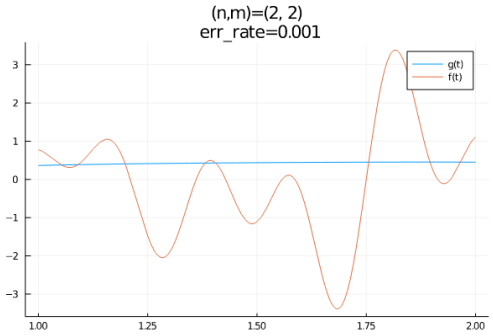
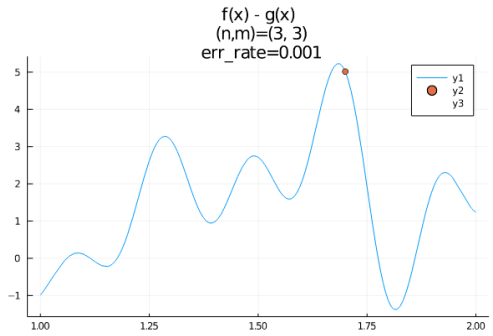


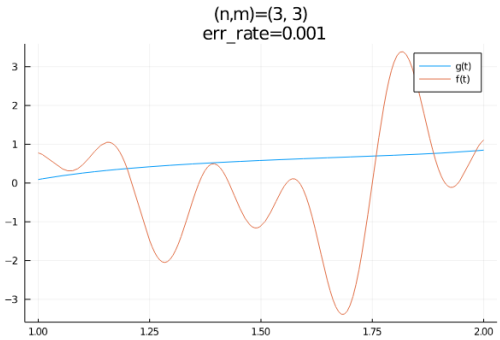

Figure 8: Comparing 2 functions based on (n,m).



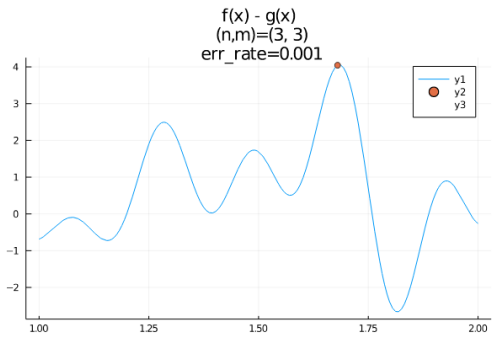

(a) step: 0.1



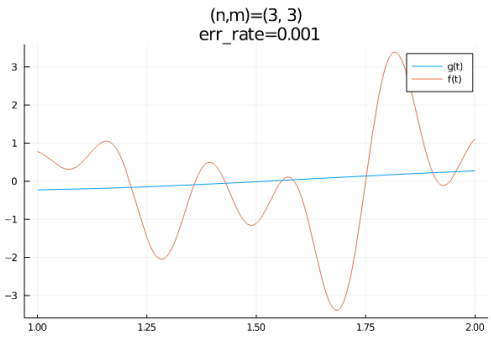

(b) step: 0.1



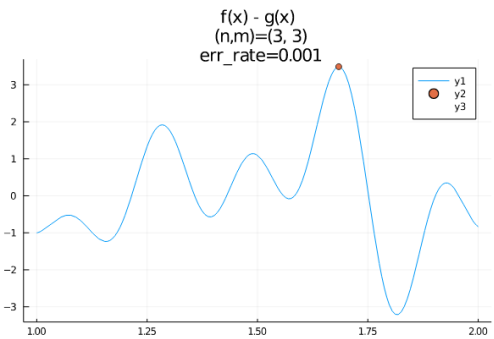

(c) step: 0.01




(d) step: 0.01




(e) step: 0.001




(f) step: 0.001

Figure 9: Comparing 2 functions based on number of hyper-planes.

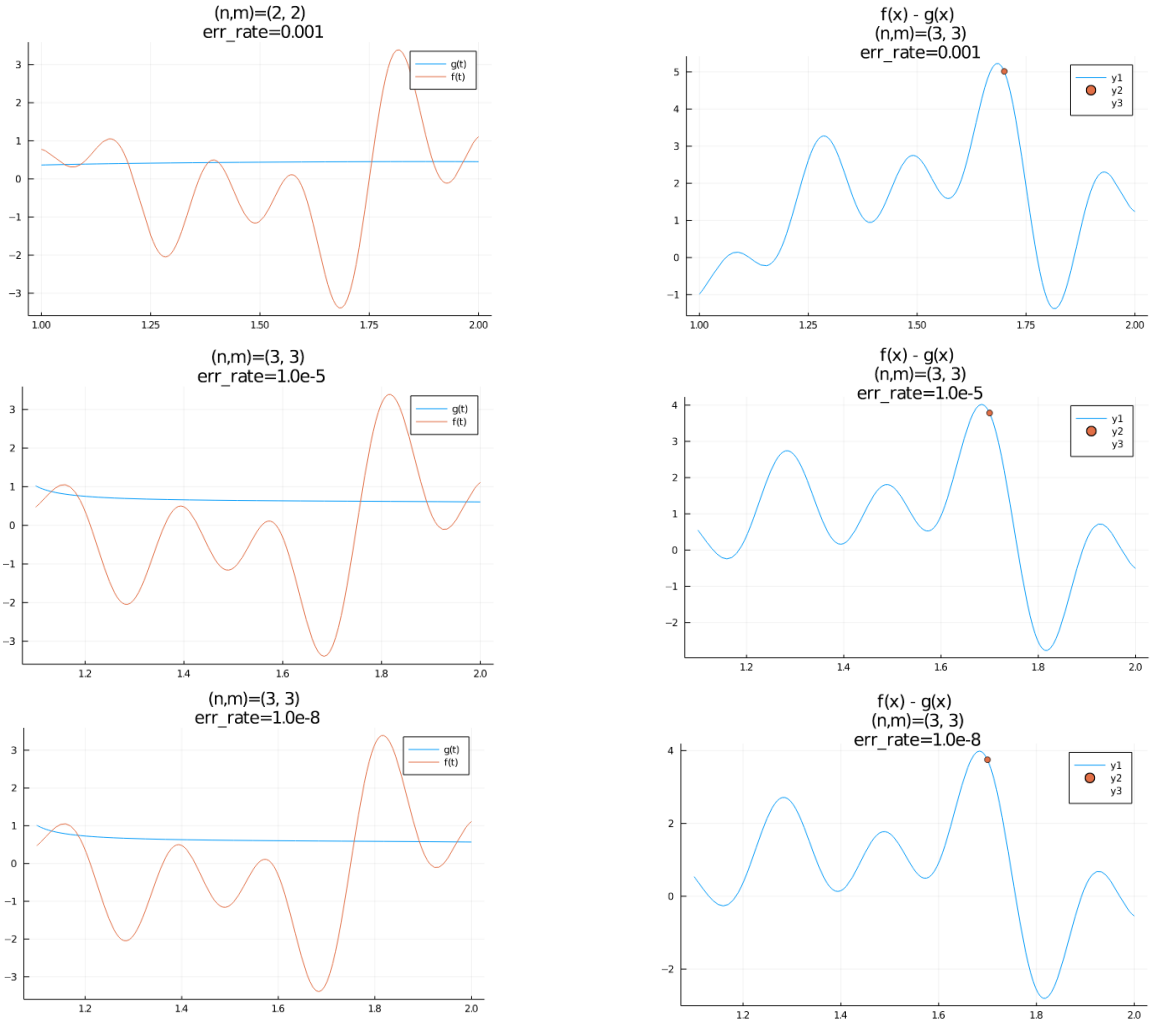



Figure 10: Comparing 2 functions based on the error rate.

| no | (n, m) | step from 0 to 3 | error rate | computation time (s) | iteration times |
|---|---|---|---|---|---|
| 1 | ***(2,2)*** | 0.1 | 1.00E-03 | 3.64 | 236 |
| 2 | ***(3,3)*** | 0.1 | 1.00E-03 | 1.93 | 50 |
| 3 | ***(4,4)*** | 0.1 | 1.00E-03 | 2.97 | 161 |
| 4 | (3,3) | ***0.01*** | 1.00E-03 | 99.23 | 230 |
| 6 | (3,3) | 0.1 | ***1.00E-05*** | 191.32 | 16108 |
| 7 | (3,3) | 0.1 | ***1.00E-08*** | 548.96 | 51960 |

Figure 11: Results for multivariate.

much effect. Yet, increasing the number of hyper-planes has huge positive effect in this case as it is the one that has the lowest means among others in figure 7. It also can be seen from results on 9. Thus, in this case, the effect of each input are almost contrary to the case of simple function.

To explain, firstly, the problem this time would be the case of no intersection. Because of that, more iteration (determined by the error rate) would not have much effect as the list of convergent points would not change much as can be seen from figure 2. Secondly, as this function has too many fluctuations, the number of planes would play a crucial role as it is needed to tell the direction of function. In other words, more planes means that the algorithm can know more about the fluctuation. Lastly, for the case of n,m, it is due to the problem of having extreme points which will be explained later.

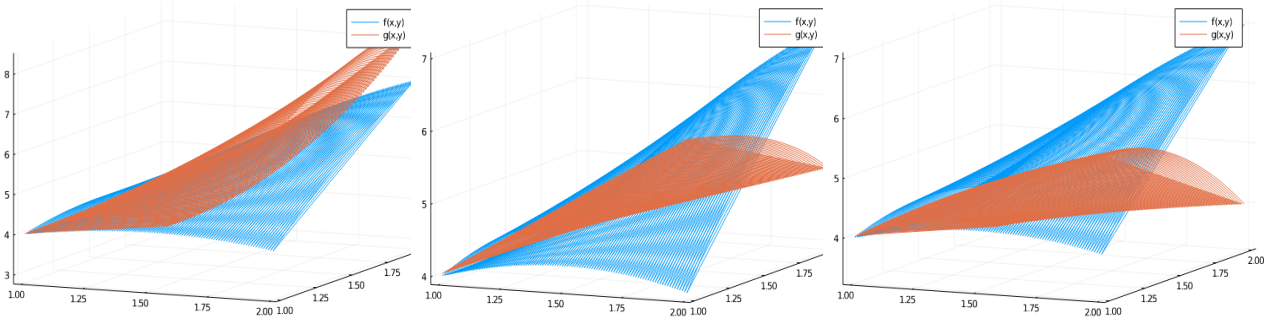
### 4.3 With 2 variables

For the experiment with 2 variables, we used a polynomial to see how the algorithm will solve a multivariate problem. The function is

$$f(x, y) = 3xy - x^2 - y^2 + 3.$$

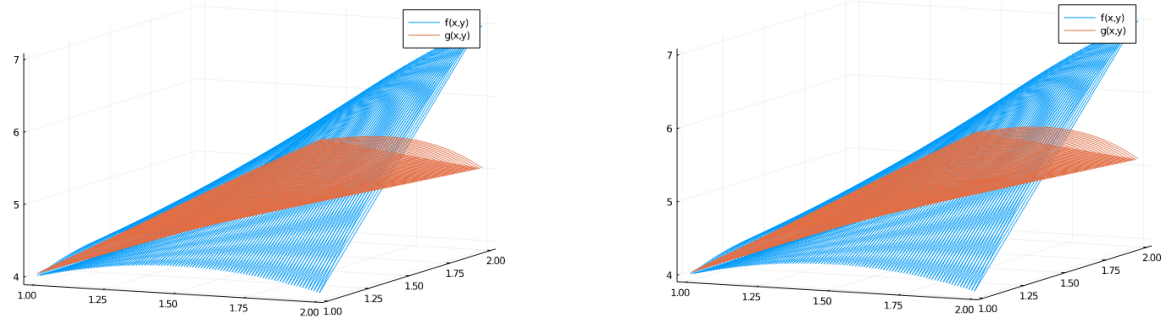
The applied method is similar to the other two experiments however we instead of using the means, we will take comment by direct observation of the 2 functions on a 3D coordination. The results of this experiment can be viewed from figure 11.

In general, the results are not too positive as for all the cases, the approximated function has quite a different shape from the goal function. Higher n and m in this case does not show better result from figure 12. This maybe because of the goal function is closer to $(n, m) = (2, 0)$, hence, it produce the best result. For the number of hyper-planes, it does not have much difference but it does show a slight better as can be seen from figure 13. However, combining with the run-time factor, it maybe not worth it. Regarding the error rate from figure 14, the results when error rate is $10^{-5}$ and when it is $10^{-8}$ are almost identical. The reason is similar to the explanation of the case of complex function as the chance of having an intersecting point in this case is quite



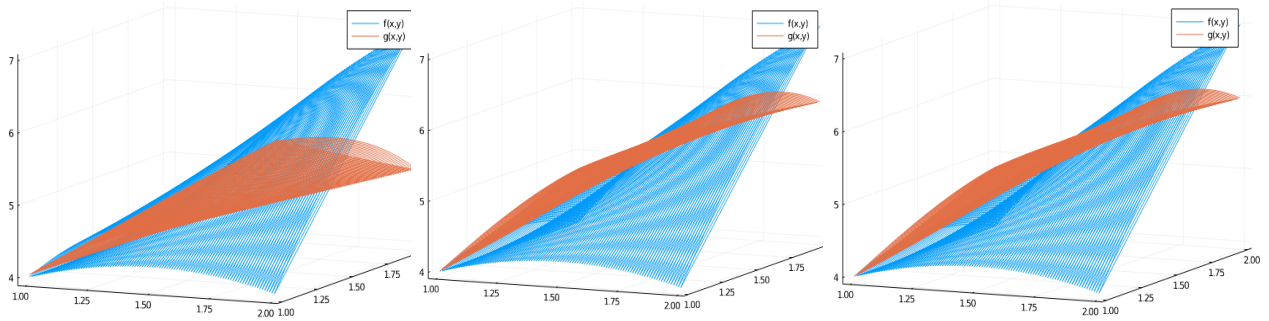

(a) (n,m) = (2,2) (b) (n,m) = (3,3) (c) (n,m) = (4,4)

Figure 12: Comparing 2 functions based on (n,m).




(a) step: 0.1 (b) step: 0.001

Figure 13: Comparing 2 functions based on the number of planes.




(a) error rate = $10^{-3}$ (b) error rate = $10^{-5}$ (c) error rate = $10^{-8}$

Figure 14: Comparing 2 functions based on the error rate.

rare as well. However, it also showed a better results then when error rate $10^{-3}$ based on the moving trend of the function. Nevertheless, the long run-time is quite concerning for this factor based on the results in figure 11

# 5 Discussion

## 5.1 Advantages

From the experiments, it is undeniable that the algorithm has the charm of speed and robustness. It can compute and has the approximation result extremely fast (most of the time less in a second) and it is reliable with the clear back-end. Furthermore, it is very easy to implement as it was based on alternating orthogonal projections (Prasad 1980). Plus, for a simple problem in a range, the accuracy is somewhat acceptable, especially when it is rational approximation. Therefore, it is a great method to be combined with other methods such as uniform approximation in piece-wise approximation.

## 5.2 Limitations

### 5.2.1 Low accuracy

As can be seen from the experiment with complex function, the algorithm does not work well with many fluctuations.

### 5.2.2 Having extreme points

When implementing the algorithm, a problem appeared which we called "extreme points". An example can be seen from figure 15. It is when the approximated goes to a value that is extremely different from goal function at certain points. There are 2 possible explanation for this issue. Firstly, it can come from the calculation machine (such as the laptop). Computer itself also has to use approximation as the it has limited resources so it cannot store the absolute value of number that has infinite digits such as $\pi$ or $\sqrt{2}$. Undoubtedly, the stored value is rounded to a certain degree that the difference is extremely small, however, when it goes exponentially, it will become a big difference. This also explains why higher n and m can actually result in worse results. Secondly, the problem can come from the algorithm itself. Figure 16 showed a simple case of 3 planes with it resulted point $x$ and the convergent points on each hyper-plane. As it can be seen, the x point is much nearer to the point $x1$ and $x2$ but far from the point $x3$. So putting in more extreme cases, there will be high chances that the resulted point can be far away from the some hyper-planes and this will create the problem of extreme points.

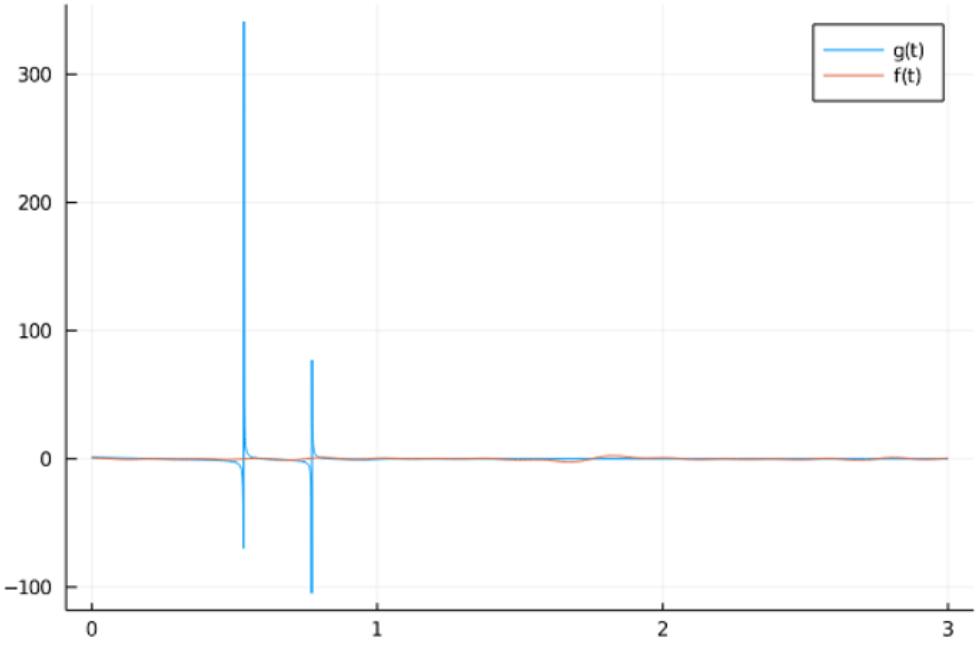



Figure 15: Example of having extreme points

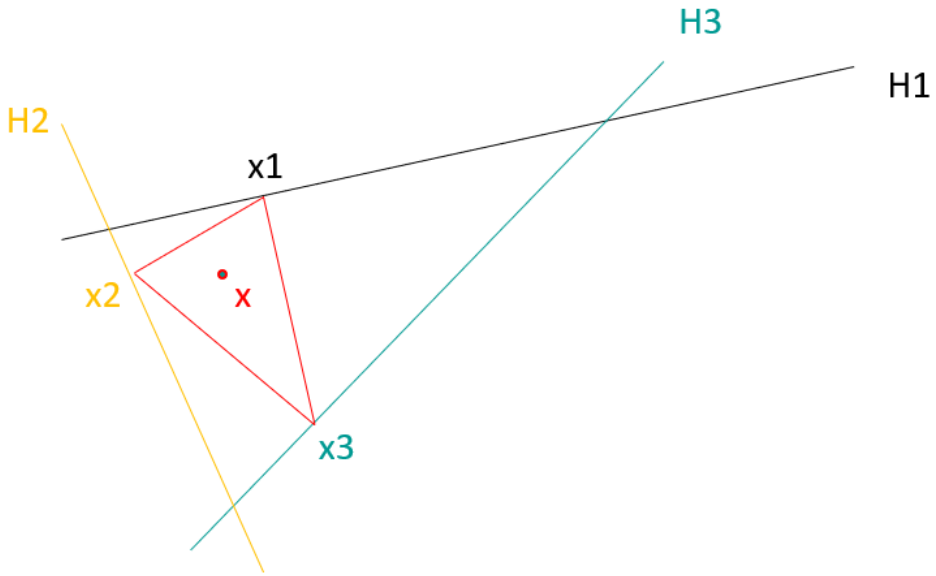



Figure 16: Example of distance among planes

# 6 Further research

## 6.1 Infinite number of hyper-planes

One way to fix all the aforementioned problems, as discussed, is increasing the number of hyper-planes. However, determine which points to create the hyper-planes is also problem, as well as deciding how many hyper-planes is sufficient as we can have an infinite number of hyper-planes in the interval. Therefore, we want to have a method that works for infinitely many hyper-planes. However, this approach is still in the early stage.

## 6.2 Calculating the resulted point

At the moment, the way to calculate the final point is still quite simple by taking the average of all the points. This can be effective at certain cases, yet also can be the reason for the extreme points problem as discussed. Changing the way to calculate the last point can be the key to better the algorithm. Some recommendations can be: finding some hyper-planes that has intersection first and find those intersecting points or deciding which hyper-planes should be prioritized.

## 6.3 Combining with others methods

Combining with other methods is an obvious application of this method as discussed. This still needs to do more research on how to determine which part is the best to use this method to increase the speed and which part will use a more complicated method such as uniform approximation to achieve better accuracy. This is also the fundamental part of piece-wise approximation. The work of Spline approximation and interpolation presented by Powell et al. (1981) is a great candidate for the solution.

# 7 Acknowledgement

At the end of this report, I wish to acknowledge the help and support of Dr. Julien Ugon and Dr. Reinier Diaz Millan. They have helped a lot on picking the problem and research topic, going through all the mathematics problem and helping with the implementation on Julia Lang. My journey would never be completed without their tremendous help.

## References

Badea, Catalin and David Seifert (2016). "Ritt operators and convergence in the method of alternating projections." In: *Journal of Approximation Theory* 205, pp. 133–148. ISSN: 0021-9045. URL: https://ezproxy.deakin.edu.au/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0021904516000174&site=eds-live&scope=site.

Bauschke, Heinz H. and Jonathan M. Borwein (1996). "On Projection Algorithms for Solving Convex Feasibility Problems". In: *SIAM Review* 38.3, pp. 367–426. DOI: 10.1137/S0036144593251710. eprint: https://doi.org/10.1137/S0036144593251710. URL: https://doi.org/10.1137/S0036144593251710.

Blair, JM, CA Edwards, and JH Johnson (1976). "Rational Chebyshev approximations for the inverse of the error function". In: *Mathematics of Computation* 30.136, pp. 827–830.

Deutsch, Frank (1984). "Rate of convergence of the method of alternating projections". In: *Parametric optimization and approximation*. Springer, pp. 96–107.

— (1992). "The method of alternating orthogonal projections". In: *Approximation theory, spline functions and applications*. Springer, pp. 105–121.

Judd, Kenneth L (1996). "Approximation, perturbation, and projection methods in economic analysis". In: *Handbook of computational economics* 1, pp. 509–585.

Millán, R Díaz, Nadezda Sukhorukova, and Julien Ugon (2020). "An algorithm for best generalised rational approximation of continuous functions". In: *arXiv preprint arXiv:2011.02721*.

Neumann, John Von (1949). "On Rings of Operators. Reduction Theory". In: *Annals of Mathematics* 50.2, pp. 401–485. ISSN: 0003486X. URL: http://www.jstor.org/stable/1969463.

Petrushev, P. P. and Vasil A. Popov (1987). *Rational Approximation of Real Functions*. Encyclopedia of Mathematics and Its Applications volume 28. Cambridge University Press. ISBN: 9780521331074. URL: https://ezproxy.deakin.edu.au/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=616938&site=eds-live&scope=site.

Powell, Michael James David et al. (1981). *Approximation theory and methods*. Cambridge university press.

Prasad, S. (1980). "Generalized array pattern synthesis by the method of alternating orthogonal projections". In: *IEEE Transactions on Antennas and Propagation* 28.3, pp. 328–332. DOI: 10.1109/TAP.1980.1142332.