Master Thesis: A standard exchange algorithm for polynomial approximation

Coralie Marchau

June 28, 2024

Internship January-July 2024

Supervised by :

Alexandre Goldsztejn and Christophe Jermann

Abstract

This report is about Chebyshev approximation problem seen like a LSIP. It's multivariate variant has received little attention those last decades. This report searches to study the problem under the vision of modern resolution techniques.

Stone-Weierstress theorem states the every continuous function can be approximated by a polynomial function, with its precision depending of the degree of approximation. Polynomial functions have numerous practical proprieties, two important ones being : cheapness of computations and ease to compute integrals or derivatives. There are functions for which it is interesting to keep an approximation instead of the original functions, with a precision adapted to the application. This is something that can be found in mathematical libraries for special functions.

While the univariate case is well studied, there are also multivariate functions that are interesting to approximate. Scaling the problem from univariate to multivariate functions, ask for adaptations and some proprieties cannot be verified as easily (like the propriety of equioscillation, a propriety indicating optimality).

An algorithm used to approximate functions is the Second Algorithm of Remez [Rem34] also presented in [Ree91] as it will be our primary source.

In this paper we will use some of the examples of [Ree91] to observe the comportment of the multivariate Second Remez Algorithm, define singular and regular functions and the functions that can be approximated by the algorithm. And finish by observing the convergence of those functions.

In a first section, we will talk precisely about univariate functions to see the basis more precisely and observe interesting proprieties. In the second section we will go more precisely on the changes necessary with using a multivariate functions and the results obtained with [Ree91]'s examples. After that we will conclude on our observations.

Thank you :

To my supervisors, Christophe Jermann and Alexandre Goldsztejn. For all the help they brought me and without who this wouldn't have been possible.
And to my family, for all their support during difficulties.

Introduction

Stone-Weierstress theorem indicate that for all functions continuous on an interval, there exists a polynomial function of degree n for which, on that interval, the error between this function and its polynomial approximation is at maximum e^*. With e^* solving the following :

$$\min_{a \in \mathbb{R}^{n+1}} \max_{x \in X} |P(a,x) - f(x)|. \tag{1}$$

This is something that is interesting in multiple branches of applied mathematics, be it in computer science (notably for mathematical library) or even in other domain (as in microbiology where it has been used to predict the growth of microbes). There exists multiple reasons why it can be interesting to replace a function by its approximation. The most common ones being to reduce computation time as certain function can be extremely long to calculate and to use the different polynomial proprieties. An approximation being sufficient for most needs.

Amongst the algorithms able to find the optimal polynomial a^* is the second Remez algorithm [Rem34]. This algorithm has known results of convergence in the univariate case [VEI60] and relatively well-known criterias to verify the optimality of a^*. Some of them no longer being valid in the multivariate case. The Equioscillation theorem of the univariate case being invalid for the multivariate case (notably due to the necessity of a strict increasing (or decreasing) order of the points).

In this report, we are going to study the second Remez Algorithm using the examples of Reemtsen [Ree91]. More precisely we will concentrate on the functions with two variables, at the degree of approximation 2. The choice of those functions was due to an ease of graphic representation. As before using the degree 3, representation stays practical for better comprehension.

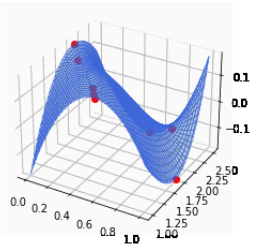


Figure 1: Example : multivariate 3D error

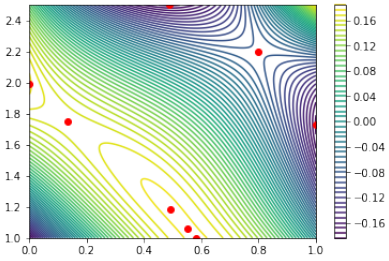


Figure 2: Example : multivariate level-set error

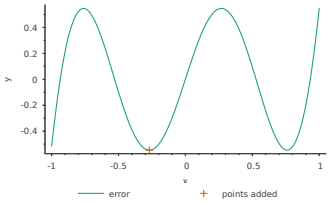


Figure 3: Example : univariate curve-line error

For each of those functions, we will verify their optimality, how they were classed be it regular or singular. Regular meaning more points reaching e^* that there are coefficients in the polynomial and singular meaning at most as many points reaching e^* that there are coefficients in the polynomial [Wat75]. We will also compare our results to Reemtsen.

Our implementation of the second algorithm of Remez will use modern strategies and tools of approximation, such as Soplex for the simplex or Ibex for the non-linear solving.

The goal is to confirm or contradict the results of Reemtsen and observe the convergence of those functions. Also to precisely see the conditions for optimality in the case of multivariate optimization.

In a first section, we will make a primer on polynomial approximation to present the problem, the equioscillation theorem, the second algorithm of Remez and its variants, before going over two examples and looking at its

convergence, all that in the univariate case.

In the second section, we will use this first section to redefine polynomials, extremas and the conditions on f to be able to approximate it correctly. We will then see the changes it brought on the way to implement the algorithm before seeing the obtained results and concluding this report.

Contents

1 Writing convention

Symbol	Meaning	Math
x	Variable(s)	$x \in X$
i	Index for variables	$x_i \in X_i, i \in [1,k]$
k	Number of variables	$k > 0$
X	Domain considered for the approximation.	$X \subset \mathbb{R}^k$
a	Polynomial coefficients	$a \in \mathbb{R}^{n+1}$
a^*	Optimal polynomial coefficients	
j	Index for coefficients	$a_j \in \mathbb{R}, j \in [0,m]$
n	Degree of approximation	$n >= 0$
m	Number of coefficients (in the univariate case $m = n+1$)	
$f(x)$	Function to approximate	
$f'_{x_i}(x)$	Derivative of f depending x_i	
$P(a,x)$	Polynomial approximation of f	
$E(a,x)$	Approximation error	$E(a,x) = f(x) - P(a,x)$
e^*	Maximum absolute error of the optimal approximation	
ϕ	monomial basis for P	
δ	Distance between max(E(a,x)) and P(a,x)	$\delta = \lvert \max(E(a,x)) - P(a,x) \rvert$

2 The univariate case

2.1 A Primer on polynomial approximation

Let's take a univariate function $f(x)$ continuous on a closed interval $X \subset \mathbb{R}$. Following Stone-Weierstrass theorem, there exist a polynomial function of the type $P(a,x) = \sum_{i=0}^{n} a_i * x^i$ that can approximate $f(x)$ to a precision depending of n.

Remark. *We assume that f is not a polynomial function of degree lower or equal to P as, in that case, $P(a,x) = f(x)$ is possible, meaning that there is no need for an approximation.*

As $P(a,x)$ is an approximation and following the previous remark, $\exists x$ for which $E(a,x) = P(a,x) - f(x) \neq 0$. Meaning that in the case of the best approximation, there should be an optimal error e^* that solves Equation 2 :

$$\min_{a \in \mathbb{R}^{n+1}} \max_{x \in X} \lvert P(a,x) - f(x) \rvert. \tag{2}$$

The algorithm studied here is Remez Second Algorithm. This exchange algorithm uses the Equioscillation Theorem and different results from De La Vallée Poussin to solve this problem [Rem34]. We will consider three variants of this algorithm that will be presented in the next section.

Theorem. ***Univariate Equioscillation Theorem*** *([tbANO64] page 20)* **:** *Let f be an univariate function on X approximated by a polynomial P of degree n with the coefficient a. This polynomial is the optimal polynomial approximating f if $\exists\ n+2$ points x_i for which $e^* = \lvert f(x_i) - P(a,x_i) \rvert$. Those $n+2$ points are also of alternating error sign.*

A way to see this is to use the following linear semi-infinite model :

$$\begin{aligned} \min_{a \in \mathbb{R}^{n+1}, z \in \mathbb{R}^+} \quad & z \\ s.t. \quad & -z \leq f(x) - P(a,x) \leq z \quad \forall x \in X. \end{aligned} \tag{3}$$

As we cannot work on an infinite amount of points, X is discretized in D. As such, the variables of the problem are the coefficient a and z. Meaning $n+2$ variables in a model with linear constraints (since $f(x)$ will always be valued since the x are constants). It can be seen by the fact that the modification of Equation(2) following is can be treated by a simplex. Equation(4

$$\begin{aligned} &\min_{a\in\mathbb{R}^{n+1}, z\in\mathbb{R}^+} z \\ &\quad s.t. \quad -z \leq f(x) - P(a,x) \leq z \quad \forall x \in D. \end{aligned} \tag{4}$$

Due to how the simplex algorithm is working, a variable is either in or out-of base (which means it either have or do not have a value different of 0). As such, if z is in base, its value will defer from 0. And if $\exists x \in D$ such as $P(a,x) \neq f(x)$, z is forced to not be null. As previously stated, for $x \in X$, $\exists P(a,x) \neq f(x)$. Meaning that $\exists D$ such as $\exists x \in D$ with $P(a,x) \neq f(x)$.

In the optimal case, we know that the value of z would be e^* and as such, there is need for at least an extrema to be in D, since otherwise there would not be a point where $P(a,x) - f(x) = \pm e^*$. But reducing it to this would be forgetting that a is also composed of variables.

For each a in base, there is need for one constraint to be active on a, meaning a point where $z = e^*$. Since there is n+1 a in our definition of P(a,x) for the univariate case, it means that there is at least n+1 extremas on top of the one previously stated.

With that theorem, it is important to remember that, from an implementation stand-point, reaching a point where there are n+2 x_i for which $e^* = |f(x_i) - P(a,x_i)|$ exactly is extremaly unlikely with the use of floats and the continuous nature of X. As such, our optimality with the second algorithm of Remez will be, in fact, to reach a state of near-equioscillation or : $\exists$ n+2 points x_i for which $e^* \pm \delta = max(E(a,x))$ with δ being close to 0.

For the sake of simplicity, we will generally consider those errors and this delta as negligible outside of the moments where we focus on the consequences of this near-equioscillation (for example mentioning a value of δ or when testing if we are at optimality).

2.2 Second Remez Algorithm

The second Remez Algorithm, also named Remez Exchange Algorithm, defined in [Rem34] is an iterative process created in 1934. Relying on different results and theorems of De La vallée Poussin and Chebyshev, it's goal is to compute a polynomial approximation of a function.

For this, it alternates the resolution of two problems. Equation(4) and Equation(5

$$\begin{aligned} &\max_{x\in X, z\in\mathbb{R}^+} z \\ &\quad s.t. \quad -z \leq f(x) - P(a,x) \leq z \quad \forall x \in B. \end{aligned} \tag{5}$$

Algorithm 1: Second Remez Algorithm

```
Begins :
D_0 ← first discretization of X
while a not optimal do
    â ← Equation(4
    x^+ ← Equation(5
    D_{i+1} ← D_i ∪ {x^+}
end
End
```

This algorithm can be summarized as a four-phase process.

The first phase is out of the loop and is a first discretization of the domain of approximation. There are multiple ways to make that first discretization but the most common ones are to get a chosen amount of equally-spaced and/or randomly selected points. To have the fastest computation, it is advised to begin with at least n+2 points so as to have the most efficient second phase.

The second phase is in the loop. Its goal is to use a linear solver to search for the best approximation polynomial on the discretized set. In the implementation, the choice has been made to use the solver Soplex to take care of

this part.

Remark. *So as to find the best polynomial for D, we use a minmax model (or the search of a as such to have the smallest greatest error). This found error is by definition a lower boundary of e*.*

Proof. Let's imagine e_i, the error found during the second phase which solves the minmax problem ($\hat{a} \leftarrow argmin e_i$ $-e_i \geq P(a,x) - f(x) \leq e_i \forall x \in D_i$). By definition, $|P(a,x) - f(x)| \leq y$. Let's suppose that $e_i > e*$. It means, $\forall x \notin D$, $|P(a,x) - f(x)| \leq e_i$ since otherwise we would need to add more constraints to our model as D will grow with x and adding constraint to a linear model will either degrade its solution as the field of possibility will be restrained or not change it bringing us back to the same situation as before. Since e* is the greatest error found on B and $D \in B$, for $e* \neq e_i$, it is necessary that there exist a x such as $|P(a,x) - f(x)| \geq e_i$ but as it is contrary to our supposition, we are proving by contradiction that e_i, the error of phase 2 is a lowerbound of e*. □

The third phase uses a non-linear solver to find the point(s) in which there are the greatest error on entirety of the domain of approximation. As long as we are out of the optimal case, this point will be out of the discretized set. In the implementation, this phase is taken care of using Ibex.

Remark. *In the third phase we are searching for $x \leftarrow argmax|P(a,x) - f(x)|; x \in B$. The error in x is a upperbound of e* as if the error for x is lower than e*, it means we have a polynomial more optimal than the optimal one which, by definition of optimal, is impossible.*

The fourth phase is the updating of D. From there, the loop which bring back to the second step until reaching a stopping criterion (in the implementation, we choose a number of turn and a distance between the errors of the phase 2 et 3).

Those last two phases are slightly changed to bring forth the three variants of the algorithm.

Exchange Algorithm : corresponding to the first version of the algorithm, in this variant D has its size fixed at n+2 and the extrema found during the third step replaces one of the member of D (the closest to it that is of same sign).

Cumulative Algorithm : In this version, instead of replacing the newfound extrema, it is simply added to D.

Remez+ : In this version, the third phase uses the first derivative test to find all extremas x for the given a and add them all in D.

The differences between those variants will mainly influence the number of turns and the rate of convergence to reach optimality.

2.2.1 Example of execution and comparison of variants

Example 1

Consider $f(x) = cos(x) + sin(5x) + x^2$ that we approximate with a polynomial of degree 3 $P(a,x) = a_0 + a_1 * x + a_2 * x^2 + a_3 * x^3$ on $X = [-1,1]$. Due to this degree, we will have five extremas on our error function and, if the polynomial is optimal, those five extremas will have the same error e^*.

Let's see a few turns of the algorithm for each variant.

Turn 1

With the discretized set D_0 of turn 1, with the use of the following linear model, we can find a the polynomial resolving the minmax model.

min z
$-z <= \quad f(x) - P(a,x) \quad <= \quad z \quad \forall x \in D_0$

z will give us our greatest error $\forall x \in D_0$. As to have a comparative experience with each variants, we decides to begin with D_0 of size n+2 (or, in this precise case, five).

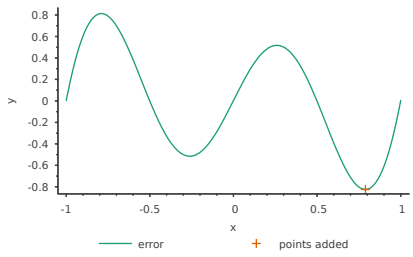


Figure 4: Turn 1 : Exchange

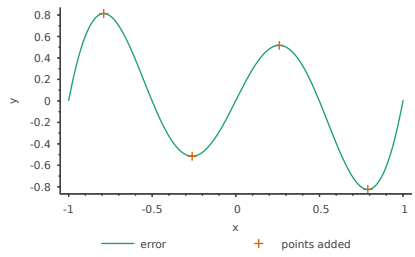


Figure 5: Turn 1 : Remez+

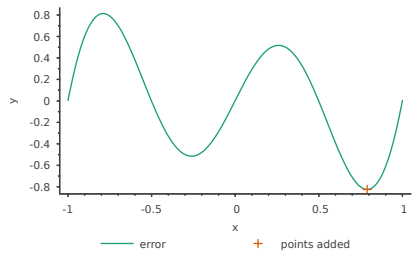


Figure 6: Turn 1 : Cumulative

As can be seen with the figures 1 to 3, in both the exchange and Cumulative variants, we find the same point at the end of the third phase whereas we find all extremas with the plus variant. It will, obviously have an effect on the following iteration as D will be of size 6 for the Cumulative variant, 5 for the exchange variant (as the newfound point is exchanged with the closest of same sign that was in D_0) and 9 for plus.

Turn 2

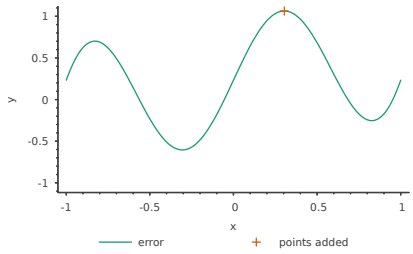


Figure 7: Turn 2 : Exchange

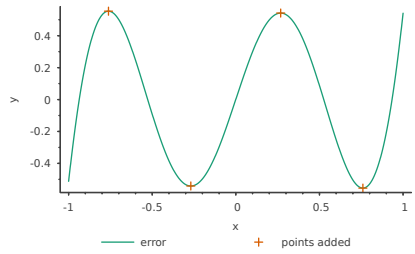


Figure 8: Turn 2 : Remez+

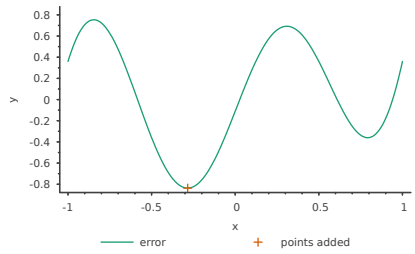


Figure 9: Turn 2 : Cumulative

The effect of that difference on the points added to D is visible during the second turn (figures 4 to 6). Plus finds itself very close to the optimal case, to the point where, to the naked eye on this graph it is hard to spot the difference in the different extremas (at this turn it is of $1e^{-2}$). For exchange and basic both, it can be seen that the point that as been added to D has gotten its error lowered. Exchange has corrected one of its extremas whereas Basic simply added the point found previously to its discretization. As there is not much to say at turn 3 compared to the others, let's jump at turn 4.

Turn 4

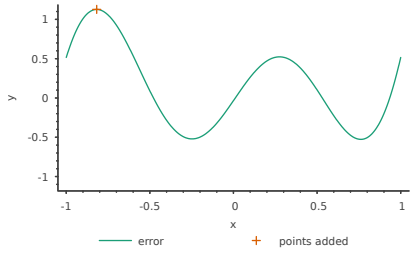


Figure 10: Turn 4 : exchange

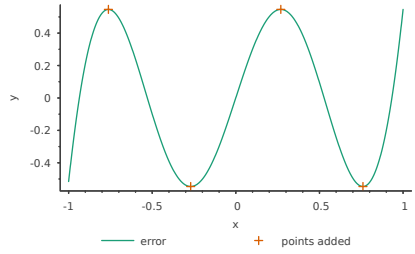


Figure 11: Turn 4 : plus

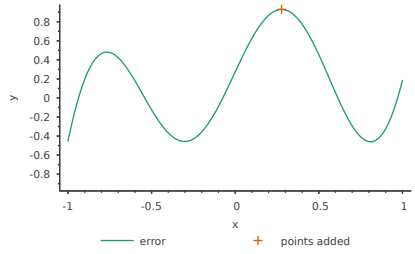


Figure 12: Turn 4 : basic

At this point we reach the end of the Plus variant as δ finds itself at around $1e^{-11}$. The attentive readers may have seen that between turn two and three, the maximal error rose before reducing. This is a normal phenomenon

that happen during the approximation process as D does not contain the ideal extremas. (Proof to find back in [tbANO64])

We will now pass a few turns to see the last graph for each algorithm with turn 14.

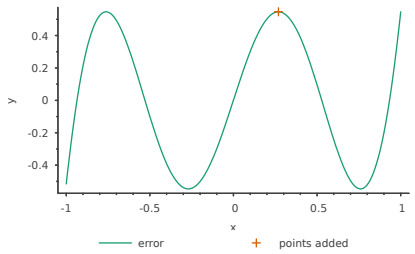


Figure 13: Final (turn 14) : exchange

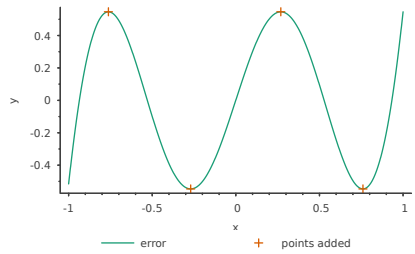


Figure 14: Final (turn 4) : plus

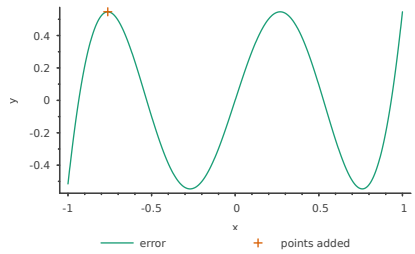


Figure 15: Final (turn 14) : basic

Example 2 (example Remez)

This example comes from [Rem34]. The function to approximate is abs(x) with$X = [-1, 1]$. As abs(x) cannot be derived at 0, we can use the propriety saying that the polynomial approximation of abs(x) at the degree 2*n is equal to the polynomial approximation of $\sqrt{x}$ with X=[0, 1] at the degree n. More precisely, we are going to approximate abs(x) to the degree 8.

Proof. It is due to the fact that $\sqrt{x^2} = abs(x)$. We will use y with $y = x^2$ for an easier comprehension. $abs([X]) = abs([-1, 1]) = [0, 1]$, explaining the change in domain. $P(a, y) = \sum_{i=0}^{i=n} a_i * y^i$. The equivalence between the square root and absolute function bringing to the following $P(a, x) = \sum_{i=0}^{i=n} a_i * x^{i*2}$. □

The problem that appears with this example is in the use of the Plus variant. To find every extremas, we searches for $f'(x) = 0$. But the derivative of $\sqrt{x}$ is not defined on 0 (as it is $\frac{1}{2\sqrt{x}}$).

Using the Cumulative and Exchange variants, we find back results close to those of [Rem34] with $a_{Cumulative} = [0.0346898, 3.809809403663, -10.36336245455, 13.71976028391, -6.235586689174]$ for the Cumulative variant and $a_{exchange} = [0.0346898, 3.80980940363, -10.36336245445, 13.71976028382, -6.235586689157]$. $a_{remez} = [0.034680, 3.809830, -10.36$ being from the article.

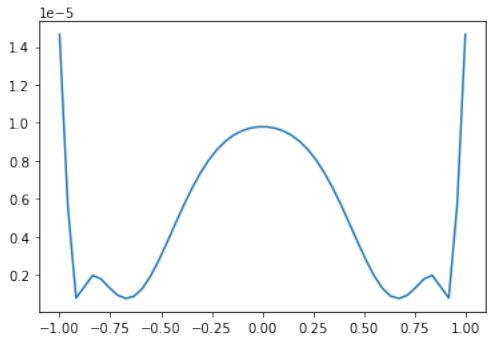


Figure 16: Absolute difference between polynomial Remez and Cumulative

As we can see, there is a distance of 10^{-5} between the results, which is not particularly shocking as no measures were taken to limitate floating-point error. The polynomial of Cumulative being slightly worse toward the extremities but a bit better toward the center. From this, we can consider that Cumulative is a bit further away from the optimal case that the polynomial from [Rem34].

2.3 Study of convergence

Convergence bottleneck

The two types of error that we can find for $E_a(x)$, the first is for $x \in D$. In this implementation, this error is obtained during the phase 2 by the linear solver as we resolve the minmax problem depending a with $x \in D$. We can also get the error with $x \in X$.

Those two errors converges slowly toward one another until reaching optimality as explained in [Rem34] by using a theorem from La Vallée Poussin. A quick summarized proof could be the following.

Proof. As $D \subset X$, the greater error for $x \in D$ is necessarily lower or equal to the greatest error for $y \in X$. In the case of optimality, all extremas will be in D forcing that the greatest error for $x \in D$ will be equal to the one for $y \in X$. As the points in D will grow (or shift) to contain the extremas, the distance between the greatest error for $x \in D_i$ and $y \in X$ will converge toward 0 forming a bottleneck. □

Remark. *As previously stated : the second phase is* $\hat{a} \leftarrow argmine1$ *with e1 being the greatest absolute error of* $f(x) - P(a, x)$ *for* $x \in D$*. The third phase is to search max e2* $x \in B$*. As such, we can conclude that a good indication of where we are at in the algorith is to check the difference between those two errors as they both converges toward* e^*

A way to measure the convergence toward optimality can then be the distance between those two errors toward each other.

Convergence rate

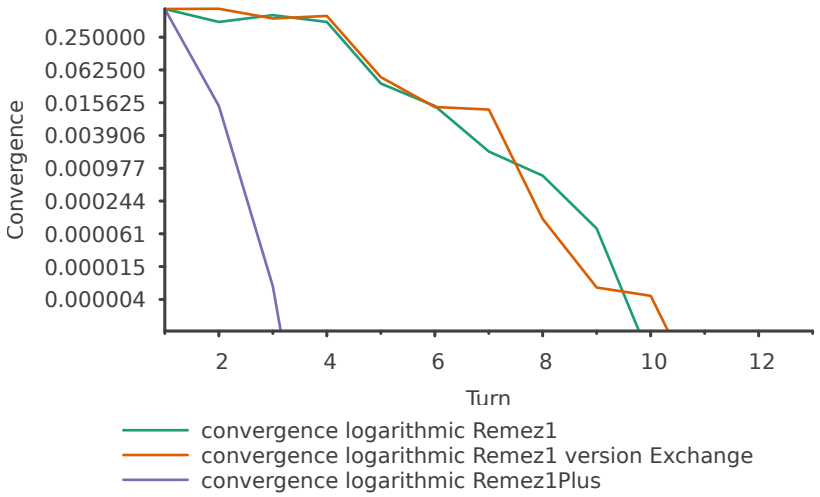


Figure 17: Convergence graph of$cos(x) + sin(5 * x) + x^2$

As the example showed, those three variants do not converge at the same speed. As seen in the example,it can be seen on the previous graph that the Basic and Exchange variants converges at a similar pace whereas the Plus variant converges far faster. Results by Verdinger ([VEI60]) indicates that the Plus variant should converge in a quadratic manner whereas basic and exchange should be linear.

3 Multivariate case

3.1 Primer

The Weierstrass-Stone theorem states that any continuous function has a polynomial such as its maximal error is inferior to an arbitrary $e > 0$. This covers multivariate functions, but such functions have different properties than the univariate case. In this first subsection, we will see more general changes, such as the types of polynomials or the way to search for extremas. In the next one, we will see the adaptations needed as we go from a problem with one variable to one with multiples. Then we will reach the experimental part using the examples from [Ree91].

3.1.1 Polynomials

There exist a wide variety of polynomials, be it in the univariate or multivariate case. But, with the presence of multiple variables, the possibilities immediately increases. For those interested, there is a lot of information and presentations of multiple kinds of multivariable polynomials in [Rei03]. Here we will just talk about two types of polynomial functions that are used in [Ree91].

The first kind, $P1$ is relatively close to what we had in the univariate case. With : $\sum_{i=0}^{i=n} \sum_{j=0}^{j=n-i} a_{ij} * x_1^i * x_2^j$, meaning that the number of coefficients a is $m = \binom{n+k}{k} + 1$. With it we consider that the degree correspond to the maximal power of variables considering the multiplication with one another.

The second kind, $P2$, is $\sum_{i=0}^{i=n} \sum_{j=0}^{j=n} a_{ij} * x_1^i * x_2^j$, meaning that $m = (n+1)^2$. With this type of polynomial, we can consider that the degree correspond to the maximal degree that each variable can reach.

By definition in [Rei03], it is to be noted that while $P1$ is of total degree n, $P2$ on the other hand would be simply of degree $2n$. Their respective monomial basis being ϕ_1 and ϕ_2

3.1.2 Graphs

For the multivariate case, some choices have to be done for graphic representation. For the case of functions with two variables, we have chosen to represent the error of the approximation with two different types of graphs. x1 and x2 always being the variables corresponding to the axes with the error serving of third variable (linked to the colors for the 2D graph and linked to height for the 3D graph).

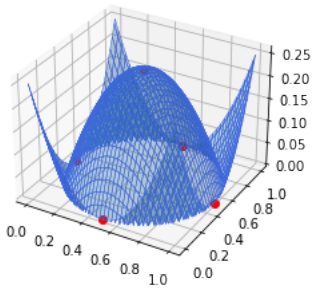


Figure 18: 3D-graph : $x_1^2 + x_2^2$, degree 2

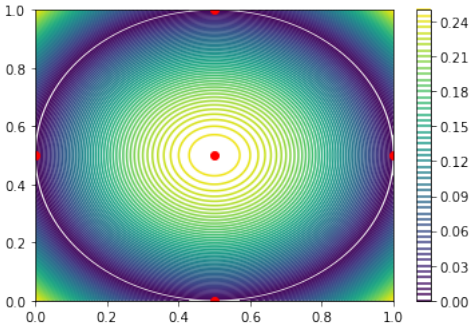


Figure 19: 2D-graph : $x_1^2 + x_2^2$, degree 2

Above are the two main types of graphs we will use for in this part. Both serves the purpose of seeing more precisely the profile of the error function and the extremas in different manners. The one of the left is a 3-dimensional representation of the error function, allowing to better see the general form of the error. But it is less precise to see where the extremas may be located. Particularly when in a static form as on here. The second makes the general form of the function more abstract but gives a better global view of where the extremas are located.

3.1.3 Extremas

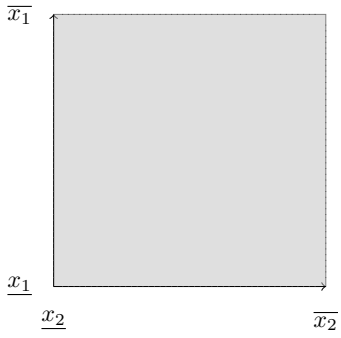
With the rise in dimension, finding the extremas of the error becomes a more difficult problem (5). As we can expect $m+1$ optimal extremas and an extrema may take multiple turns to be found, finding them one by one as done with the variant Exchange and Cumulative would be a tedious process. There is also the inherent issue of trying to find a maximum or minimum of a multivariate function efficiently.

A way to take care of this issue is to use the same method applied in the Plus variant but adapted for a multivariate approach with the second derivative test.

Theorem. ***First Derivative Test :***

For $f(x)$ continuous on an interval X, if at $f'(x) = 0$ it changes sign, $f(x)$ will be an extrema.

The Second Derivative Test derives from the first and serves to find the extremas inside of X. In the case of the search for extremas, we decided to simplify it as it is not problematic to get saddle points in D. Let's take the example for a function of two variables. This allows us to get the extremas easily as it is a known difficult problem to find the global extremas in a space.



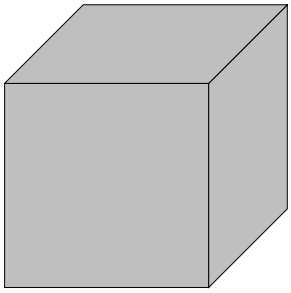
At the right, we have a graph of the space of x for a two variate function. It is the space that needs to be searched to find the extrema of the error.
If we fix one of the x (let's say x_1), we find ourselves with a univariate function, meaning that, we can find the extrema for that fixed x_1 by searching $f'_{x_2}(x) = 0$. This logic can be reverse by fixing x_2 instead.
At this point, it becomes logical that we can find the extremas in the gray box by searching $f'_{x_2}(x) = 0$ & $f'_{x_1}(x) = 0$. Though it will also give us the saddle points. The saddle points could be filtered out with a Second Derivative Test condition (), but as the saddle points do not bother us and the condition would an inequality, we decides to lose a bit of memory and catch them (this may not be a solution for some problems as we could imagine a function for which the error would bring many saddle points or in functions with a high number of variables). To ensure to have every extrema, you have to also have the extremas on the border of the space. For this, we use the first derivative on each border.

The issue of this method is linked to the use of derivative as we need to be able to derive depending on the different variables, it is important to also verify that we do not arrive to a point where $E'(a,x) = 0, \forall x \in X$. Though this situation should never happen in the general case, with no variable fixed, it can happen on the borders of the space.

As $E'(a,x) = P'(a,x) - f'(x)$, for it to be null $\forall x \in X$, it would be necessary for either all a to be equal to 0 or for them to be cancelled by $f'(x)$. Giving $f'(x) = P'(a,x)$. So, either $f'(x) = 0$, in which case it should not be approximated. Or $f'(x)$ is a polynomial of same degree as $P'(a,x)$. Meaning that $f(x) + c = P(a,x)$ with c being a constant. Meaning that we are trying to approximate a polynomial by another similar polynomial of same degree.

As for the borders, it means that we can have a situation where $f'_{x_a}(x) = 0 \forall x_b$. An example could a function with $g(x) * sin(x_a)$ on a border where $x_a = 0$ with the coefficient corresponding to x_b still being at 0. This situation is relatively improbable if we begin with m+1 correctly chosen points, but a verification can easily be implemented (in case of a derivative that falls at 0 on a border, we can simply verify that there exist a coefficient linked to an unfixed x that is not to 0). Another possibility could be a function of the form $g(x_b) * h(x)$ with h(x) being a polynomial function that could cancel $P(a,x)$ and $g(x_b)$ being a function depending only on x_b. Meaning that, if we fix x_b we could have the derivative that is worth 0 for all x_a.

This can be scaled up for more than two variables following the same logic. One verification in the "center" of the search space with all first degree derivative being equal to 0, then until only having the corners of the space, fixing variables to get every side. Another example, in three dimension could be the following.



For three variables, X corresponds to a cube. The inside of the cube would correspond to the space in which extremas would be searched with $f'_{x_1}(x) = f'_{x_2}(x) = f'_{x_3}(x) = 0$. Each face would then be searched by fixing one variable to find back on the same problems as for a two variable problem. Then the same thing for each border of each face. This means that for a number k of variables there would be $\sum_{i=1}^{i=k-1}(2^{k-i}\binom{k}{i}) + 1$ problems to resolve.

3.1.4 Function/Polynomial preconditions

As we have seen in the univariate case, it is important to put preconditions on the type of functions that can be approximated.

The first one being a derivation of the univariate case : a polynomial should not be approximated in a polynomial of degree equal or superior of same form.

Remark. *$Pi(a_1, x)$ should not approximate $Pj(a_2, x)$ if $\phi_i \subseteq \phi_j$.*

Another situation that should not happen is one where approximating f does not need the multivariate aspect, causing an infinite amount of extremas. For instance, imagine a function of form $f(x) = g(a^T x + b)$ with the polynomial of form $P_1(a, x) = \sum_{i=0}^{i=n} \sum_{j=0}^{j=n-i}(a_i j * x_i^i * x_j^j)$. Then the optimal multivariate approximation of f(x) can be deduced of the univariate for $g(y), \forall y \in Y$ with Y being the result interval of $(a^T x + b), \forall x \in X$.

Proof. Let's take $f(x) = g(a^T x + b)$. X being compact, we can define Y = $[\underline{a^T x + b}, \overline{a^T x + b}]$ as the interval on which g is acting depending on X. $P_m(a, x)$ is the optimal multivariate polynomial approximation of f(x) and $P_u(a, y)$ is the optimal univariate polynomial approximation of g(y) with $y \in Y$.

As such, we have :

$$M = \min_{p \in \mathbb{R}^m} \max_{x \in X} |f(x) - P_m(a, x)|, \tag{6}$$

$$m = \min_{p \in \mathbb{R}^n} \max_{t \in I} |g(t) - P_u(a, t)|, \tag{7}$$

$P_u(a, i)$ has n coefficients and $P_m(a, x)$ has m coefficients, with $m = \binom{n+k}{k} + 1$. Using the previous definitions of the multivariate polynomial P1 and the univariate polynomial of the first section we get :

$$M - m = \min_{p \in \mathbb{R}^m} \max_{x \in X} |f(x) - P_m(a, x)| - \min_{p \in \mathbb{R}^n} \max_{t \in I} |g(t) - P_u(a, t)|, \tag{8}$$

It can be seen that $M \leq m$ as $\forall t \in I$, $\exists x \in X$ such as $t = a^T x + b$. Meaning that any extrema of the univariate approximation is either an extrema of the multivariate or has a less important error than the extrema of the multivariate case.

As $\forall x \in X$, $\exists t \in I$ by definition of I, any extrema of the multivariate is present in the univariate. As such, $m \leq M$.

We can so conclude that by the linearity between x and t, $m = M$. □

This phenomenon causes issues for the algorithm during the phase 3, the search of the extremas, as we will find lines of extremas as seen in the following graph.

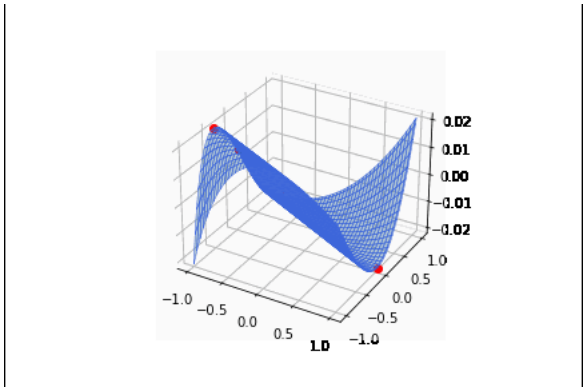


Figure 20: Example : $\sqrt{x_1 + 2 * x_2 + 4}$, degree 2, 3-dimensional

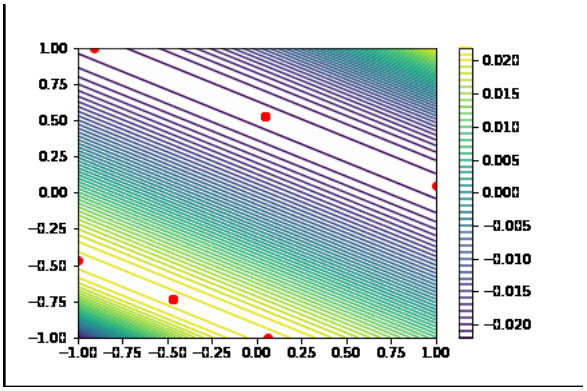


Figure 21: Example : $\sqrt{x_1 + 2 * x_2 + 4}$, degree 2, level sets

To prove that those line of errors are caused by the form of f(x), let's take $Y = a^T x + b$. And search for $e(x) = \hat{e}$.

$$\hat{e} = f(a^T x + b) - p^*(a^T x + b), \tag{9}$$

It can immediately be guessed that, if Y is constant, the result should also be constant.

With the use of an example in degree 2, we can see that it is possible to see the form of the function such as to have a line of extrema.

$$\begin{aligned} f(X) &= \hat{e} + q^*(X) && (10)\\ 0 &= -f(X) + \hat{e} + a_0 + a_1 * (X) + a_2 * (X)^2 && (11)\\ 0 &= -f(a^T x + b) + \hat{e} - a_0 + a_1 * (a^T x) && (12)\\ &\quad + a_1 * b + (a_2 * a^T x)^2 + a_2 * b^2 + 2 * (a_2 * a^T x) * b && (13) \end{aligned}$$

By searching the roots, it is then possible to use the last expression to find the function corresponding to the error line.

To conclude on this subsection, we can define the type of functions that we consider as to approximate.

- f must either not be a polynomial or be a polynomial not included in P.
- f must not be approximated by a P1-type polynomial if it is of the form $g(a^T x + b)$
- f must have defined derivative on X for its variables.

3.2 Algorithm

Using the previous section, here is Remez Plus Algorithm for multivariate functions.

Algorithm 2: Second Remez Algorithm

Begins :
$D_0 \leftarrow$ first discretization of X
while a not optimal **do**
  $\hat{a} \leftarrow$ Equation(4)
  $x^+ \leftarrow$ Equation(5)
  $D_{i+1} \leftarrow D_i \cup \{x^+\}$
end
End

As seen previously, there are a few updates in how those steps happen. So there will be a small subsection for each phase of the algorithm.

3.2.1 Phase 1 : First Discretization

To prepare the discretization, the first step is to calculate m depending of the type of polynomial. To have the most efficient Phase 2 and 3, it is better to have at least m+1 points in the first Discretized set. It can be advised though to put amongst those points the corners and the center of the space on top of either randomized or equally spaced points.

The points in the corners of the space are harder to reach for the solver during phase 3. As such, giving them from the beginning ensure their presence in D.

3.2.2 Phase 2 : Finding a

In the multivariate phase, we solve the same problem as in the univariate case. We are taking back the final model that was talked about in the corresponding part of section 1.

min z
$P(a,x) - z \quad >= \quad f(x) \quad \forall x \in D$
$P(a,x) + z \quad <= \quad f(x) \quad \forall x \in D$

The difference this time being in the amount of variables and minimal number of constraints. While we had n+2 variables in the univariate case, we now have m+1 variables. And just like with Remez Plus and Remez Cumulative, we have more and more constraints appearing at each turns.

As we are using a simplex with Soplex, it is not an issue for our case, but, it can be imagined that issues of slowness may appear in case of high degree or if f has numerous variables.

3.2.3 Phase 3 : Finding extremas

To find all the extremas, we uses a non-linear global optimisation solver. This is necessary to ensure finding all the points. For this we uses the tests described previously.

We begin by searching the stationary points for which all derivative of first order of $E(a,x)$ reach 0. Then, for each variable, we fix it to one of the extremity of its domain and search for the points where all derivative of first order of $E(a,x)$ for the non-fixed variables. Until there is only one variable to fix.

As the corners never changes, it is simpler to includes them in D from the beginning instead of checking the value of their error at every turn of the algorithm to see if they should be added.

The recursive way to do it for a multivariate function with k variables could be :

Algorithm 3: FIND-EXTREMAS

```
1 Input: F : Fixed x, X : Domain of f, a : coefficients .
2 Begins :
3 extrema ← {}
4 if ∃x ∈ X, x ∉ F then
5 |   Solves ∀x ∈ X, x ∉ F, E'(a, x) = 0
6 |   extrema ← results
7 |   ∀x ∈ X, x ∉ F, extrema ⋃ find-extremas(F ⋃ x, X, a).
8 |   Return extrema;
9 end
```

Conclusion

To finish this part we can see that there are not too many differences on the algorithm itself but the change in the function dimension does cause some adaptation to be necessary. This will be even more seen in the next section.

We have mentioned previously the three main phases of the algorithm but, we have not yet talked about the condition "a is optimal" of the loop. While in the univariate case it was relatively easy to see and conclude optimality

thanks to the Equioscillation theorem. In the multivariate case, new concepts need to be thought about. Notably those of Regularity and Singularity.

3.3 Optimality and convergence

Previously, in the univariate case, we said that we could use the Equioscillation Theorem to verify optimality. If we had n+2 extremas of the error function which, when ordered in increasing order $\forall i \in [1, n-1], e_i = E(a^*, x_i)$, $x_i < x_{i+1}$, would be of opposite sign two by two $\forall i \in [1, n-1], e_i = -e_{i+1}$.

In the multivariate case, this cannot be done in the same way as even finding the orders of the m+1 e_i would be problematic as our x are vectors of dimension n with $n > 1$.

As such, the equioscillation theorem is replaced by the following as presented by Cheney in [Che81].

Definition 1. Multivariate Equioscillation Condition :

For the condition to be satisfied at a^*, we need $xi, i \in [1, K]$ points where $E(a^*, xi) = \min_{a \in \mathbb{R}^m} \max_{x \in X} |f(x) - P(a^*, x)|$ with $K > 1$.

With those K points and ϕ correspondinf to the form of P. We will make the corresponding Haar matrix H. Let's use P_1 defined previously for the degree 2 as example. Each column will correspond to an xi giving :

$$\mathrm{H} = \{ \begin{matrix} 1 & ... & 1 \\ x1_1 & ... & xK_1 \\ x1_1^2 & ... & xK_1^2 \\ x1_2 & ... & xK_2 \\ x1_2x1_1 & ... & xK_2xK_1 \\ x1_2^2 & ... & xK_2^2 \end{matrix} \}$$

If the condition holds, the kernel of H, $u \neq 0$ and $u_i * E(a, xi) >= 0$.

The strong condition of equioscillation holds if and only if $K \geq m + 1$ and $u_i \neq 0, \forall i \in [1, K]$. This strongness allows to conclude that a^* is a strongly unique minimizer.

It can be seen that in the case where $K \leq m$, H is not full rank and u is a non-trivial kernel.

From this definition, we can see two different cases of equioscillation during multivariate approximation. In [Wat75] a function for which we find less than m+1 extremas for the error is called a singular function.

3.3.1 Regularity and Singularity

Definition 2. Regular Function

Let's take f(x) a multivariate function with x being a vector of dimension k. Due to the Stone-Weierstrass theorem, $\exists P(a, x)$ the polynomial function of degree n with m coefficients for which $e^* = \min_{p \in \mathbb{R}^m} \max_{x \in X} |f(x) - P(a^*, x)|$ with at least m+1 points for which $E(a, x) = e^*$.

Due to the Equioscillation theorem defined previously, the Haar matrix H is full rank and if $0 \notin u$, a^* is a strongly unique minimizer of the problem.

As there exist m+1 extremas, the number of extremas is equivalent to the number of variable of the simplex, making it more efficient to find solutions.

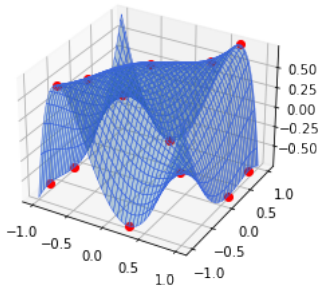


Figure 22: Example : $exp(x_1^2 + x_1 * x_2)$, degree 2, level sets

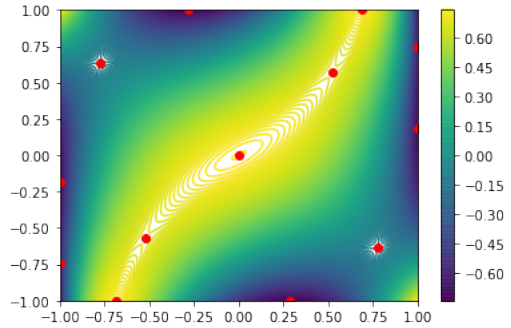


Figure 23: Example : $exp(x_1^2 + x_1 * x_2)$, degree 2, 3-dimensional

Definition 3. Singular Function Let's take f(x) a multivariate function with x being a vector of dimension k. Due to the Stone-Weierstrass theorem, $\exists P(a, x)$ the polynomial function of degree n with m coefficient for which $e^* = \min_{p \in \mathbb{R}^m} \max_{x \in X} |f(x) - P(a^*, x)|$ with at most m points for which $E(a, x) = e^*$.

The Haar matrix cannot reach full rank and as such, uniqueness cannot be proven.

Due to the presence of less variables than number of extremas, there is a potential difficulty to solving the problem [Wat75].

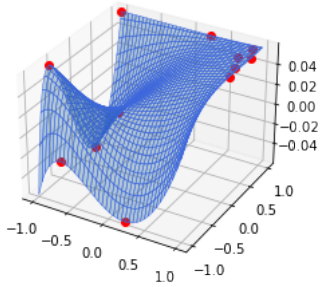


Figure 24: Example : $\frac{1}{x_1+2*x_2+4}$, degree 2, level sets

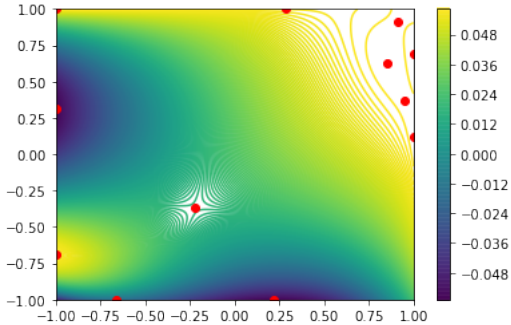


Figure 25: Example : $\frac{1}{x_1+2*x_2+4}$, degree 2, 3-dimensional

As described in [Wat75] from the results of [OW69], this phenomenon seems to come from extremas coalescing and becoming one. Making the optimal basis matrices of the simplex become more and more singular, deteriorating the solution.

3.3.2 Convergence

It is mentioned in [Wat75] that the convergence is not proven for the multivariate case but theorized that it may be quadratic (for regular cases) due to the method being analogous to the second algorithm of Remez.

In the following section, we will try to observe it using the examples from [Ree91].

3.4 Reemtsen Example

In the following table are the examples from [Ree91] that will we study in the experimental section :

Name	Function category	Error	Basis (degree)	f	X
f1 (4.1)	Regular ×	2.80626×10^{-2}	P1 (2)	$ln(x_1 + x_2) * sin(x_1)$	$[0, 1] \times [1, 2.5]$
f2 (4.2)	Regular ✓	1.77657×10^{-1}	P1 (2)	$(1 + x_1)^{x_2}$	$[0, 1] \times [1, 2.5]$
f4 (4.4)	Singular ✓	5.835897×10^{-2}	P2 (2)	$\frac{1}{x_1+2*x_2+4}$	$[-1, 1] \times [-1, 1]$
f5 (4.5)	Singular ×	7.354679×10^{-1}	P2 (2)	$exp(x_1^2 + x_1 * x_2)$	$[-1, 1] \times [-1, 1]$
f6 (4.6)	Singular ✓	1.140057×10^{-2}	P2 (2)	$\sqrt{x_1 + 2 * x_2 + 4}$	$[-1, 1] \times [-1, 1]$

This table contains the name we give the function (with its respective Reemtsen name in parenthesis), the category of the function depending as indicated by Reemtsen (either confirmed ✓ by our results or refuted ×). The error corresponds to the error indicated by Reemtsen (a table further down will give our results and how they compare), basis correspond to the type of polynomials using those described previously. f and X corresponds to the formula of the function and its space.

Using the equioscillation method and our implementation, we can confirm the category of three of the examples and refute two of them.

The protocol followed for each function was the following. Using the polynomial $\hat{a}$ found with our implementation, we searched for the extremas using the following model :

$$\begin{aligned} &\mathrm{x} \in X \\ &\hat{e_1} \qquad \leq |f(x) - P(\hat{a}, x)| \leq \quad \hat{e_2} \end{aligned}$$

e_1 being the error at the end of the phase 2 (being the maximal error in the discretized points) and e_2 being the error at the end of the phase 3 (or the real maximal error for $\hat{a}$). It is then important to remember that we are not necessarily at the optimal a^*.

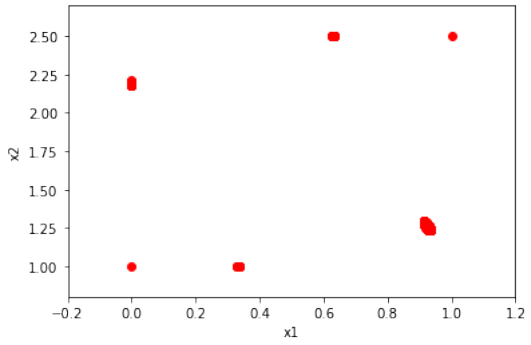
The fact that $\hat{a} \neq a^*$ means that $\hat{e_2} > e^*$ and $\hat{e_1} < e^*$. Since there are no points filtered by $|f(x) - P(\hat{a}, x)| \geq \hat{e_2}$, we can take off this part of the constraint, ending up with the following model.

$$\begin{aligned} &\mathrm{x} \in X \\ &\hat{e_1} \qquad \leq |f(x) - P(\hat{a}, x)| \end{aligned}$$

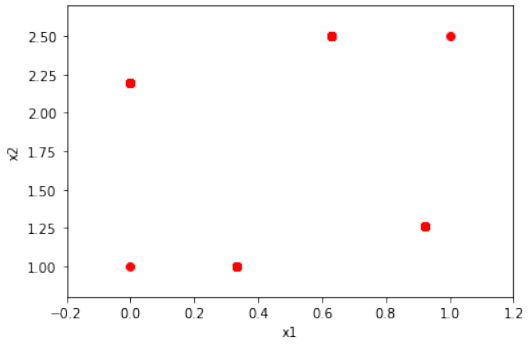
But, there is an issue with this method. When not at optimality, there may be some of the extremas which are below e_1. So we need to introduce a δ as the difference between $\hat{e}$ and e^*.This also allows to take into consideration float-related imprecision. But this δ must be small for precision and not too small as to not miss extremas. Giving the following model :

$$\begin{aligned} &\mathrm{x} \in X \\ &\hat{e_1} \qquad \leq |f(x) - P(\hat{a}, x)| \end{aligned}$$

To give an idea, here are two examples, using f1 and f2, showing the importance of delta. The first one, using f1 showing what happens when we use a delta too big.



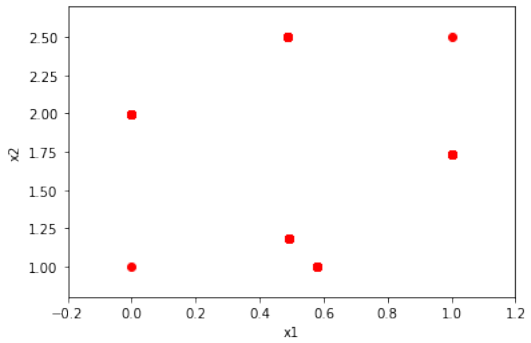
$f1, \delta = 1e-5$



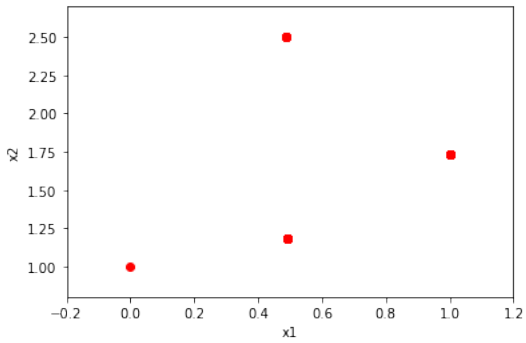
$f1, \delta = 1e-10$

As we can see, with the graph on the left, we have little pockets of extremas instead of points. Making it harder to find the precise point of the extrema to verify the equioscillation condition.

Now with f2, we will see the possible consequences of not putting a delta.



$f2, \delta = 1e-11$



$f2, \delta = 0$

As we can see, without delta we only find four of the seven extremas due to not being at optimality. And a delta of $1e-11$ suffice to find them all.

To find them as precisely as possible, we finally use both the results of this model and the points found during the phase 3. Finally to find precisely the points, the easiest solution is simply to take back all the points found during the last phase 3 and the corners of the space X and verify those that are close or bigger than e_1.

With those points, we can then verify the optimality using the equioscillation theorem presented previously. But, it is important to remember once more that we are not necessarily at the optimal point. So, in the singular case, we may find ourselves with u = 0. In that case, we will need to correct the core and the matrix.

A method that could be used to try and be closer to optimality, and thus avoid that type of issues, would be to use a Newton method on $\hat{a}$ and the points, as we should be relatively close to the optimal a^* and of its extremas. The goal would be to ensure that $\forall i \in [1, K], |f(x) - P(a^*, xi)| \approx e^*$.

3.5 Results

3.5.1 f1

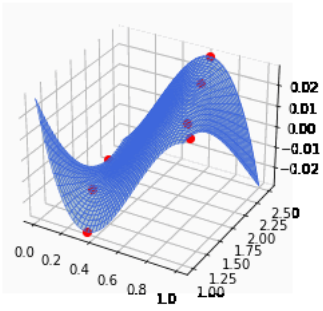


Figure 26: Example : f1, degree 2, 3-dimensional

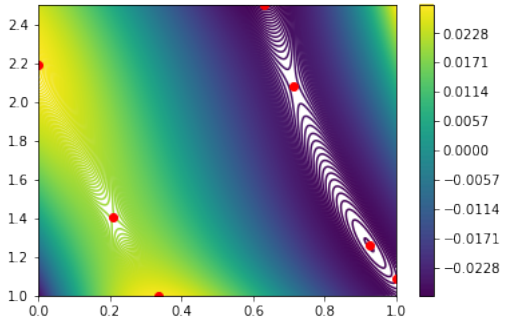


Figure 27: Example : f1, degree 2, level sets

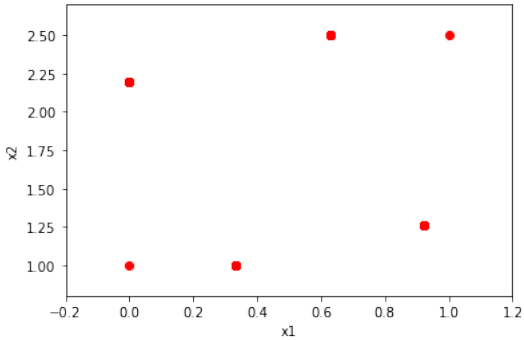


Figure 28: Example : f1, extremas

As we can see, for f1, there are only six extremas. As such we fill the Haar matrix with the corresponding points : (0, 1), (0.333614, 1) , (0, 2.19419) , (0.924601, 1.26511) , (0.630504, 2.5) and (1, 2.5). Using those, we find no kernel for H. Giving the impression that those points couldn't be the normal extremas or that there may be one missing at first view.

But by looking at the eigenvalue, we can realize that one of the value is far lower than the others, which could be the sign that this issue is due to not being at real optimality due to the different issues mentionned previously. By changing it to 0 and correcting the rest of the matrix, we find ourselves with a new Haar matrix. Matrix on which the equioscillation method is confirmed. Meaning that the weak condition is confirmable and would be probably be directly confirmed if post-treatments (like Newton) was applied to correct the coordinates of the extremas and values of a.

As we have the Equioscillation confirmed for less than $m+1$ extremas, we can conclude the optimality of f1.

Error	Basis	Distance Reemtsen	δ	m+1	nb extremas
2.8063150×10^{-2}	P1	1×10^{-7}	1×10^{-11}	7	6

This table is composed of the error found by our algorithm, type of polynomial, the difference between the errors found at phases 2 and 3, the difference between our error and the error of Reemtsen. And finally, the number of extremas we need for regularity and the number of extremas found.

As we can see, we are at the same error that [Ree91] until reaching 10^{-7}. As we don't have the polynomial from Reemtsen, it is unclear if the difference is due to the method (like our floating errors as we do not cushion them) or if our approximation is closer to optimality.

The final distance between the error of phase 2 and phase 3 is at 10^{-11}, as such we can consider that we were very clear to optimality as, at perfect optimality, those two errors should be identical.

3.5.2 f2

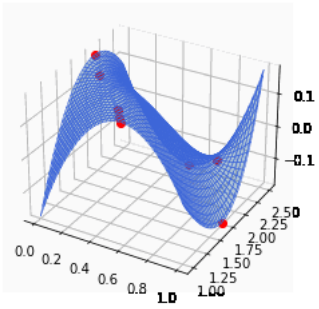


Figure 29: Example : f2, degree 2, 3-dimensional

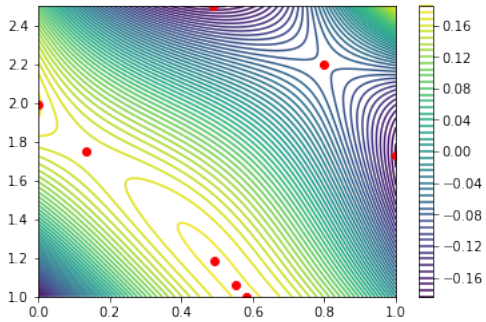


Figure 30: Example : f2, degree 2, level sets

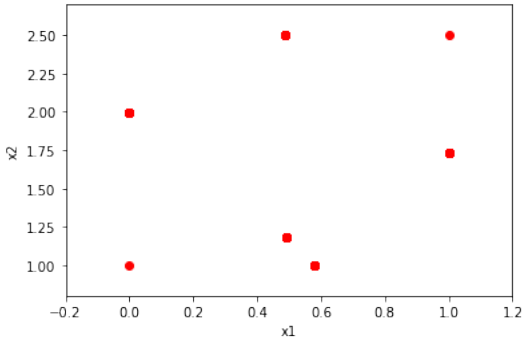


Figure 31: Example : f2, extremas

We find as many extremas as normal for a regular case with f2 with exactly m+1 extremas. Those beings : (0, 1), (0, 1.99277), (0.489717, 1.19192), (0.580058, 1), (0.488358, 2.5), (1, 1.73003) and (1, 2.5)

The Haar matrix is full rank and its kernel respect the equioscillation condition with no elements at 0. The strong condition seems to be acceptable. It is important to remember that we aren't on perfect optimality and our error isn't perfectly identical for all extremes.

Error	Basis	Distance Reemtsen	δ	m+1	nb extremas
$1.7765813769 \times 10^{-1}$	P1	1×10^{-6}	1×10^{-13}	7	7

We are at 10^{-6} from Reemtsen results. And the errors of phase 2 and 3 are at 10^{-13}. We have exactly the right number of extremes. As such we can be fairly certain of our proximity to real optimality.

3.5.3 f4

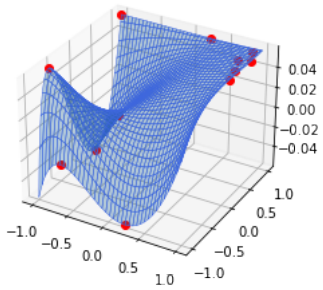


Figure 32: Example : f4, degree 2, 3-dimensional

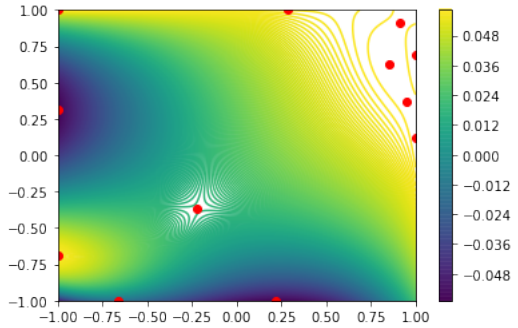


Figure 33: Example : f4, degree 2, level sets

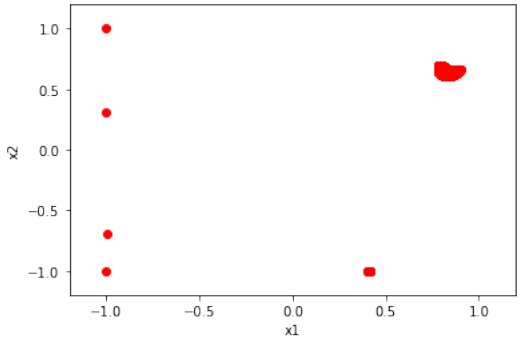


Figure 34: Example : f4, degree 2, extremas

For f4, the singularity bring us relatively rapidly to a stop as after the 34th turn with a basis matrix singularity of the simplex, which is unsurprising seeing what [Wat75] mentioned that as each application of the simplex identifies m+1 extremas of the error on the discretized set, if there are less than m+1 extrema, difficulty may occur and the simplex matrix may become more and more singular. It may be possible to perform more turns by adjusting the simplex method parameters.

We can get 6 extremas, their approximation localisations being : (-1, -1), (-1, -0.690564), (-1, 0.309047), (-1, 1), (0.40338, -1), (0.801971, 0.611246).

Error	Polynomial used	Distance results	δ	m+1	nb extremas
5.8×10^{-2}	P2	ϵ	1×10^{-5}	10	6

F4 is the function we manage the less to approximate. It is the only one where the error of Reemtsen is still between our error of phase 2 and phase 3. The difference between those phases still being at 10^{-5} when we are forced to stop.

3.5.4 f5

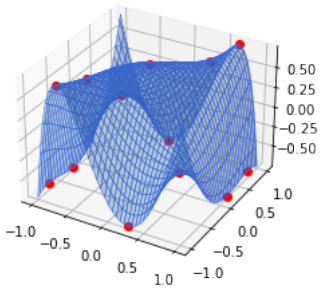


Figure 35: Example : f5, degree 2, 3-dimensional

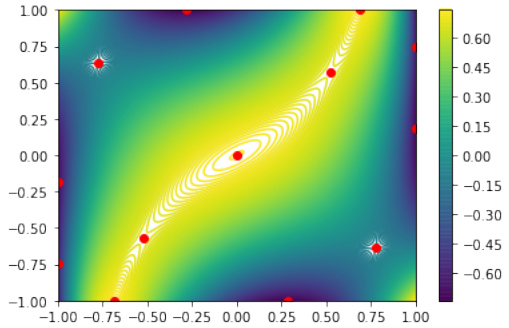


Figure 36: Example : f5, degree 2, level sets

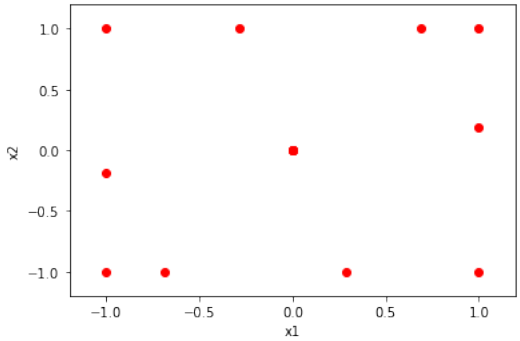


Figure 37: Example : f5, degree 2, extrema

For f5 we find 11 extremas or one more than would be strictly necessary for a regular case. With those 12 extremas : (-1, -1),(-0.686932, -1), (-1, -0.185914), (-1, 1), (-0.283355, 1), (0.283318, -1), (1, -1), (0,0) , (1, 0.186864), (0.686922, 1), (1, 1). When we do the Haar matrix, we find that we are extremely close to equioscillation with 11 points. The error of the matrix being of 10^{-15}, it can be easily guessed that this error is linked to floating error as, trying to take off some of the extrema by considering that they would disappear at optimality take us further away from equioscillation. The weak equioscillation holds.

Error	Polynomial used	Distance results	δ	m+1	nb extremas
$7.3546921623 \times 10^{-1}$	P2	1×10^{-6}	1×10^{-13}	10	11

3.5.5 f6

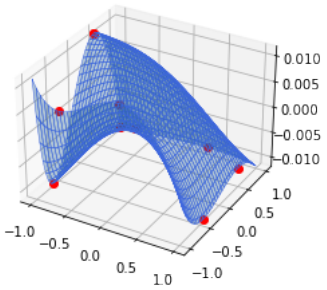


Figure 38: Example : f6, degree 2, 3-dimensional

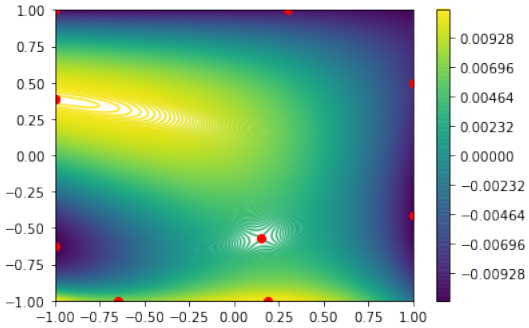


Figure 39: Example : f6, degree 2, level sets

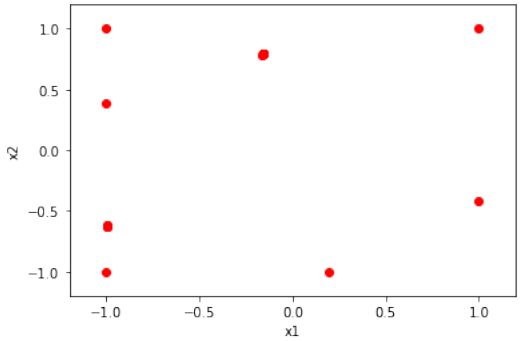


Figure 40: Example : f6, degree 2, extremas

For f6, we find the extremes : (-1, -1), (-1, -0.622294), (-0.999318, -0.622265), (-1, 0.383965), (-1, 1), (-0.158032, 0.793014), (0.190049, -1), (1, -0.415923), (1, 1). After doing the Haar matrix, we find a kernel that isn't full rank. Meaning than we only manage to reach the weak equioscillation condition.

As we only have 9 extremas, we can conclude singularity.

Error	Polynomial used	Distance results	δ	m+1	nb extremas
$1.140059945323 \times 10^{-2}$	P2	1×10^{-9}	1×10^{-11}	10	9

3.5.6 Results

To summarize, here are our conclusions for those examples.

Name	Function category	Error	Polynomial used	Distance results	δ
f1	Singular (Corrected)	2.8063150×10^{-2}	P1	1×10^{-7}	1×10^{-11}
f2	Regular	$1.7765813769 \times 10^{-1}$	P1	1×10^{-6}	1×10^{-13}
f4	Singular	5.8×10^{-2}	P2	ϵ	1×10^{-5}
f5	Regular (Corrected)	$7.3546921623 \times 10^{-1}$	P2	1×10^{-6}	1×10^{-13}
f6	Singular	$1.140059945323 \times 10^{-2}$	P2	1×10^{-9}	1×10^{-11}

As we can see, we have relarively close results to Reemtsen. We can also observe that we don't arrive to get our errors of phase 2 and 3 as close for the singular cases and regular cases.

With the corresponding approximating extremas :

Name	extremas				
f1	(0, 1) (1, 2.5)	(0.333614, 1)	(0, 2.19419)	(0.924601, 1.26511)	(0.630504, 2.5)
f2	(0, 1) (1, 1.73003)	(0, 1.99277) (1, 2.5)	(0.489717, 1.19192)	(0.580058, 1)	(0.488358, 2.5)
f4	(-1, -1) (0.801971, 0.611246)	(-1, -0.690564)	(-1, 0.309047)	(-1, 1)	(0.40338, -1)
f5	(-1, -1) (0.283318, -1) (0.686922, 1)	(-0.686932, -1) (1, -1) (1, 1)	(-1, -0.185914) (0, 0)	(-1, 1) (1, 0.186864)	(-0.283355, 1)
f6	(-1, -1) (0.19005, -1) (-0.15803, 0.79301)	(-1, -0.622294) (1, -0.415923)	(-0.999318, -0.622265) (1, 1)	(-1, 0.383965) (-1, 1)	

[-0.0115845,7.753579800958288604e-17,-2.799866126213438129e-17,-0.8020285384694316732,1.196226384676807531,-0.17838

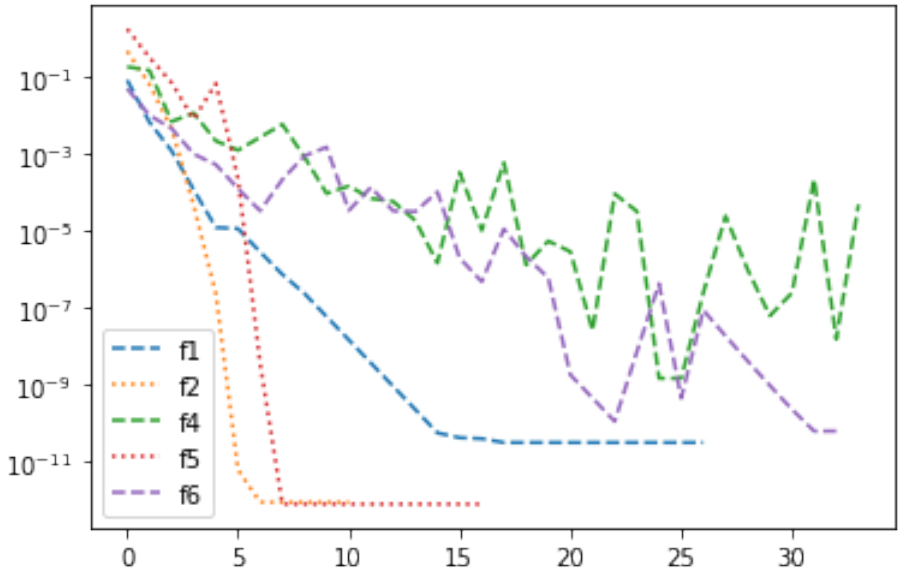


Figure 41: Convergence of the example by Reemtsen

The dotted lines corresponding to our regular functions and the dashed lines corresponding to the singular ones. We can see that the two dotted lines converge in a very similar manner, just as the dashed line of f4 and f6. f1 seems a bit intermediate between the singular and regular lines. We can also observe that the line of f4 and f6 are very saw-like.

A reason that could be thought would be, as remarked in [Wat75] that the irregularity causes issues for the simplex. As such it could potentially explain that lack of efficacy and those spikes. It is possible that the way f4 and f6 are singular make those more important than f1. As f1 behaves near regular with almost the right number of extremas (one missing).

4 Conclusion

In this report, we have talked about the polynomial approximation problem. As indicated by Stone-Weierstrass, there exists polynomial approximations for continuous functions, containing multivariate ones. Those approximation resolving the Equation 2. While the univariate part of the problem is well known, the multivariate aspect has greatly been left aside.

We have used the second algorithm of Remez on the examples of Reemtsen with two variables. To verify optimality, we used the equioscillation condition described in [Che81] and considered that our found extremas are close enough to real optimality to observe this equioscillation. If this happens with m points or less, it indicates singularity and if this happens with more than m points it means regulartity.

During our experimental phase, we confirmed Reemtsen results for f2, f4 and f6 but found contradictory results for f1 and f5. We have also a convergence graph that seem to agree with the guess from [Wat75] that non-singular problem seems to converge in a quadratic manner.

We have also seen some conditions to avoid another type of functions for which the second algorithm of Remez turns badly, the case where there is an infinite amounts of points for which $\|P(a,x) - f(x)\| = e^*$.

Multiples questions have appeared during the course of the internship. The most important one being on how to take care of the floating error. One such way that was thought about is the use of a Newton at the end of the algorithm using our last approximation as basis, since it should be very close to the optimal. To prove with certainty the optimality of the approximation, it could be good to either apply a Newton by interval or using Kantorovich theorem.

Another thing that could be studied is the possible link between the function of type $f(a^T x + b)$ and singularity for polynomials of type P2.

References

[Che81] Cheney. Introduction to APPROXIMATION THEORY. Second edition edition, 1981.

[Gon00] V. Goncharov. The theory of best approximation of functions. Journal of Approximation Theory - JAT, 106:2–57, 09 2000.

[Het76] R. Hettich. A newton-method for nonlinear chebyshev approximation. In Robert Schaback and Karl Scherer, editors, Approximation Theory, pages 222–236, Berlin, Heidelberg, 1976. Springer Berlin Heidelberg.

[HK93] R. Hettich and K. O. Kortanek. Semi-infinite programming: Theory, methods, and applications. SIAM Review, 35(3):380–429, 1993.

[OW69] M.R Osborne and G.A Watson. A note on singular minimax approximation problems. Journal of Mathematical Analysis and Applications, 25(3):692–700, 1969.

[Ree91] R. Reemtsen. A multiple exchange algorithm for multivariate chebyshev approximation. Journal of Optimization Theory and Applications, 71:85–103, 1991.

[Rei03] Manfred Reimer. Multivariate Polynomial Approximation. 01 2003.

[Rem34] E. Remez. Sur le calcul effectif des polynomes d'approximation de tchebychef. Compte rendus hebdomadaires des scéances de l'Académie des sciences, pages 337–340, 1934.

[tbANO64] I. P. NATANSON translated by ALEXIS N. OBOLENSKY. Constructive function theory. 1, 1964.

[VEI60] L. VEIDINGER. On the numerical determination of the best approximations in the chebyshev sense. Numerische Mathematik, 2:99–105, 1960.

[Wat75] G. A. Watson. A multiple exchange algorithm for multivariate chebyshev approximation. SIAM Journal on Numerical Analysis, 12(1):46–52, 1975.

[Wat00] G.A. Watson. Approximation in normed linear spaces. Journal of Computational and Applied Mathematics, 121(1):1–36, 2000.