WARSAW UNIVERSITY OF TECHNOLOGY

Numerical Methods 1

Faculty of Mathematics and Information Science

Project 2

- Paper -

By **Elie SAAD**

June 15, 2020

Contents

1 Problem Statement

Write a computer program to implement the Jarratt method for finding a root of the polynomial

$$p(x) = c_0T_0(x) + c_1T_1(x) + ... + c_nT_n(x),$$

where $T_k(x)$ denotes the Chebyshev polynomial of the first order.

2 Definitions

Definition 2.1 (Chebyshev Polynomial)**.** The **Chebyshev Polynomial** of the degree n is defined as

$$T_n(x) = \cos(n \cos^{-1} x), \text{ for } x \in [-1, 1].$$

An alternative definition for the first kind where the polynomials are more evident, is given by the recursion

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x), \text{ for } n = 1, 2, ... \end{aligned}$$

Definition 2.2 (Newton's Method)**.** The iteration given by

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \text{ for } i = 0, 1, ..., m,$$

is called the **Newton's Method**, starting with an initial guess x_0, and m determined by some stopping criterion. Meaning the iteration process termination occurs when the accuracy is considered good enough. The function $f(x)$ is an arbitrary function.

Definition 2.3 (Jarratt's Method)**.** The iteration given by

$$x_{i+1} = x_i - \frac{f(x_i)}{f'[x_i - \frac{1}{2}u(x_i)]} \text{ for } i = 0, 1, ..., m$$

is called the **Jarratt's Method**, starting with an initial guess x_0, $u(x) = \frac{f(x)}{f'(x)}$, and m determined by some stopping criterion. Meaning the iteration process termination occurs when the accuracy is considered good enough. The function $f(x)$ is an arbitrary function

3 Notation

The following technical notation is used:

- $T_i(x)$ is a Chebyshev polynomial of degree i where $T_i(x) = \cos(i \arccos(x))$.
- $t(x) \in \mathbb{R}^n$ is denoted as the Chebyshev vector, where $t(x) = \begin{pmatrix} T_0(x) \\ \vdots \\ T_n(x) \end{pmatrix}$ and $|t(x)| = n$.
- $c \in \mathbb{R}^n$ is denoted as the coefficients vector, where $c = \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix}$, $|c| = n$, and $c_i \in \mathbb{R}$ is an arbitrary number for $i = 0, ..., n$.
- $p_n(x) \in C^\infty$ is the polynomial defined as $p_n(x) = c^T t(x) = c_0 T_0(x) + ... + c_n T_n(x)$.

4 Proposed Solution

In what follows we propose a solution to tackle the problem that has been given by first reducing it into multiple smaller sub-problems. We have decided to start off by tackling the derivation problem of what seems to be a computationally heavy task for large $p_n(x)$ since, the larger the n is, the more computationally heavy the calculation of the Chebyshev polynomials gets. Then we build up an algorithm for computing the approximation for the iterative Jarratt's Method. After that we will prove the correctness of our algorithm, and finally we will prove the time complexity.

4.1 Simplifying the Derivation Problem

The Jarratt's method requires the evaluation of one function $p_n(x)$ and two derivative evaluations, $p_n'(x)$ and $p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right)$, per step. And we know that $p_n(x) = c^T t(x)$. Thus evaluating the derivatives becomes a problem of evaluating the derivatives of the Chebyshev vector. Thus finding the derivative for $p_n'(x)$ is equivalent for finding the derivative of the Chebyshev vector $t'(x)$. Also, finding the derivative for $p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right)$ is equivalent for finding the derivative of the Chebyshev vector $t'\left(x - \frac{1}{2}\frac{t(x)}{t'(x)}\right)$.

We will start off by calculating $t'(x)$. We know that $t(x) = \begin{pmatrix} T_0(x) \\ \vdots \\ T_n(x) \end{pmatrix}$ where $T_i(x) = \cos(i \arccos(x))$ for $i = 0, ..., n$. Thus, evaluating the derivative we get $t'(x) = \begin{pmatrix} T_0'(x) \\ \vdots \\ T_n'(x) \end{pmatrix}$. So, for every $T_i'(x)$ we have

$$\begin{aligned} T_i'(x) &= (cos(i \arccos(x)))' \\ &= (-sin(i \arccos(x))(i \arccos(x)) \\ &= -i(\arccos(x))' \sin(i \arccos(x)) \\ &= -\left(-\frac{1}{\sqrt{1-x^2}}\right) i \sin(i \arccos(x)) \\ &= \frac{i \sin(i \arccos(x))}{\sqrt{1-x^2}}. \end{aligned}$$

Now, we take

$$t\left(x - \frac{1}{2}\frac{t(x)}{t'(x)}\right) = \begin{pmatrix} T_0\left(x - \frac{1}{2}\frac{t(x)}{t'(x)}\right) \\ \vdots \\ T_n\left(x - \frac{1}{2}\frac{t(x)}{t'(x)}\right) \end{pmatrix},$$

then taking the derivative of $t'\left(x - \frac{1}{2}\frac{t(x)}{t'(x)}\right)$ is equivalent of finding the derivative of every element $T_i'\left(x - \frac{1}{2}\frac{T_i(x)}{T_i'(x)}\right)$ where $i = 0, ..., n$. Thus, for every $T_i'\left(x - \frac{1}{2}\frac{T_i(x)}{T_i'(x)}\right)$ we have

$$\begin{aligned} T_i'\left(x - \frac{1}{2}\frac{T_i(x)}{T_i'(x)}\right) &= \left(\cos\left(i \arccos\left(x - \frac{1}{2}\frac{\cos(i\arccos(x))}{\frac{i\sin(i\arccos(X))}{\sqrt{1-x^2}}}\right)\right)\right)' \\ &= \left(\cos\left(i\left(\pi - \arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))} - x\right)\right)\right)\right)' \\ &= \left(-\sin\left(i\left(\pi - \arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))} - x\right)\right)\right)\right) \\ &\quad .\left(i\left(\pi - \arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))} - x\right)\right)\right)' \end{aligned}$$

$$= -\left(-\left(\arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(i\arccos(x))}-x\right)\right)'\right)$$

$$i\sin\left(i\left(\pi-\arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))}-x\right)\right)\right)$$

$$= -\frac{\left(\frac{1}{2i}\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{\sin(i\arccos(x))}\right)'-1\right)i\sin\left(i\left(\pi-\arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))}-x\right)\right)\right)}{\sqrt{1-\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(i\arccos(x))}-x\right)^2}}$$

$$= -\frac{i\left(-\frac{x\cos(i\arccos(x))}{2i\sqrt{1-x^2}\sin(i\arccos(x))}+\frac{\cos^2(i\arccos(x))}{2\sin^2(i\arccos(x))}-\frac{1}{2}\right)\sin\left(i\left(\pi-\arccos\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(n\arccos(x))}-x\right)\right)\right)}{\sqrt{1-\left(\frac{\sqrt{1-x^2}\cos(i\arccos(x))}{2i\sin(i\arccos(x))}-x\right)^2}}$$

$$= -\frac{i\left(-\frac{xT_i(x)}{2i\sqrt{1-x^2}\sin(i\arccos(x))}+\frac{T_i^2(x)}{2\sin^2(i\arccos(x))}-\frac{1}{2}\right)\sin\left(i\left(\pi-\arccos\left(\frac{1}{2}\frac{T_i(x)}{T_i'(x)}-x\right)\right)\right)}{\sqrt{1-\left(\frac{1}{2}\frac{T_i(x)}{T_i'(x)}-x\right)^2}}$$

Thus, calculating $T_i(x)$ and $T_i'(x)$ is required for the calculation of $T_i'\left(x-\frac{1}{2}\frac{T_i(x)}{T_i'(x)}\right)$ to avoid repetition and reduce computation.

4.2 Building the Algorithm

Before writing the main algorithm that will approximate $p_n(x)$ using Jarratt's method, we will first write the algorithm that calculates the Chebyshev vector $t(x)$, its derivative $t'(x)$, and the derivative of the modified Chebyshev vector $t'\left(x-\frac{1}{2}\frac{t(x)}{t'(x)}\right)$. The algorithm returns the Chebyshev vector along with the derivative of the modified Chebyshev vector. The algorithm is represented by the following snippet

$c \leftarrow$ coefficients vector
$x \leftarrow$ initial value
$t \leftarrow \emptyset$
$t' \leftarrow \emptyset$
$mt' \leftarrow \emptyset$
$n \leftarrow |c|$
for $i \leftarrow 1$ to n **do**
  $T_i(x) \leftarrow \cos(i\arccos(x))$
  $t \leftarrow t\bigcup\{T_i(x)\}$
  $T_i'(x) \leftarrow \frac{i\sin(i\arccos(x))}{\sqrt{1-x^2}}$

$t' \leftarrow t' \bigcup \{T_i'(x)\}$

$T_i'(x - \frac{1}{2}\frac{T_i(x)}{T_i'(x)}) \leftarrow -\frac{i\left(-\frac{xT_i(x)}{2i\sqrt{1-x^2}\sin(i\arccos(x))} + \frac{T_i^2(x)}{2\sin^2(i\arccos(x))} - \frac{1}{2}\right)\sin\left(i\left(\pi - \arccos\left(\frac{1}{2}\frac{T_i(x)}{T_i'(x)} - x\right)\right)\right)}{\sqrt{1 - \left(\frac{1}{2}\frac{T_i(x)}{T_i'(x)} - x\right)^2}}$

$mt' \leftarrow mt' \bigcup \left\{T_i'(x - \frac{1}{2}\frac{T_i(x)}{T_i'(x)})\right\}$

end for

$p_n(x) \leftarrow c^T t$

$p_n'(x) \leftarrow c^T t'$

$p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right) \leftarrow c^T mt'$

return $p_n(x)$, $p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right)$

The main algorithm takes the coefficients vector c, the initial value x_0, the tolerance value tol, and the maximum number of iterations $maxi$. It returns the final solution x and the number of iterations i. If the algorithm returns $i = maxi$, it means that most probably the final solution is wrong, and therefore, the initial guess was too far from the correct solution. The main algorithm consists of the Jarratt's Method is described as follows

if $x_0 = 0$ **then**

$x_0 \leftarrow tol$

end if

$x \leftarrow x_0$

$n \leftarrow |c|$

$i \leftarrow 0$

$flag \leftarrow FALSE$

while flag=FALSE **do**

$p_n(x), p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right) \leftarrow$ fill with values from the previous algorithm

$x \leftarrow x - \frac{p_n(x)}{p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right)}$

$corr \leftarrow \frac{p_n(x)}{p_n'\left(x - \frac{1}{2}\frac{p_n(x)}{p_n'(x)}\right)}$

if $|corr| \leq tol \vee i > maxi$ **then**

$flag \leftarrow TRUE$

end if

$i \leftarrow i + 1$

end while

return x, i

This concludes the whole algorithm used.

4.3 Proof of Correctness

Proof. We will prove the correctness of the algorithm by induction over n.

Let $P(n)$ be the predicate "The n-th iteration of the Jarratt's Method is returned to $p_n(x)$".

We will begin with the initial step of $n = 1$:

We have

$$\begin{aligned} p_1(n) &= c_0T_0(x) + c_1T_1(x) \\ &= c_0 + c_1x \end{aligned}$$

Noting that we have

$$\begin{aligned} t &= \begin{pmatrix} T_0(x) \\ T_1(x) \end{pmatrix} = \begin{pmatrix} 1 \\ x \end{pmatrix} \\ t' &= \begin{pmatrix} T_0'(x) \\ T_1'(x) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ mt' &= \begin{pmatrix} T_0'(x - \frac{1}{2}\frac{T_0(x)}{T_0'(x)}) \\ T_1'(x - \frac{1}{2}\frac{T_1(x)}{T_1'(x)}) \end{pmatrix} = \begin{pmatrix} 0 \\ -\frac{\left(-\frac{x^2}{2i\sqrt{1-x^2}\sin(i\arccos(x))} + \frac{x^2}{2\sin^2(i\arccos(x))} - \frac{1}{2}\right)\sin\left(i\left(\pi - \arccos\left(-\frac{1}{2}x\right)\right)\right)}{\sqrt{1+\frac{1}{4}x^2}} \end{pmatrix}. \end{aligned}$$

So we get

$$\begin{aligned} p_1(x) &= c^Tt = \begin{pmatrix} c_0 & c_1 \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix} = c_0 + c_1x \\ p_1'(x) &= c^Tt' = \begin{pmatrix} c_0 & c_1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = c_1 \end{aligned}$$

$$\begin{aligned} p_1'(x - \frac{p_1(x)}{p_1'(x)}) &= c^Tmt' = \begin{pmatrix} c_0 & c_1 \end{pmatrix} \begin{pmatrix} 0 \\ -\frac{\left(-\frac{x^2}{2i\sqrt{1-x^2}\sin(\arccos(x))} + \frac{x^2}{2\sin^2(\arccos(x))} - \frac{1}{2}\right)\sin\left(\left(\pi - \arccos\left(-\frac{1}{2}x\right)\right)\right)}{\sqrt{1+\frac{1}{4}x^2}} \end{pmatrix} \\ &= c_1 \left(-\frac{\left(-\frac{x^2}{2\sqrt{1-x^2}\sin(\arccos(x))} + \frac{x^2}{2\sin^2(\arccos(x))} - \frac{1}{2}\right)\sin\left(\left(\pi - \arccos\left(-\frac{1}{2}x\right)\right)\right)}{\sqrt{1+\frac{1}{4}x^2}} \right) \end{aligned}$$

Thus the Jarratt's Method approximation of the base case is

$$x_1 = x_0 - \frac{p_1(x_0)}{p_1'\left(x_0 - \frac{1}{2}\frac{p_1(x_0)}{p_1'(x_0)}\right)}$$

Therefore the predicate $P(1)$ is true.

For the inductive step we shall prove $P(k) \implies P(k+1)$. That is, we need to prove that if the Jarratt's Method works on $p_k(x)$ then it works on $p_{k+1}(x)$.

Let us assume that for an arbitrary k, the Jarratt's Method works for $p_k(x)$ with

$$t_k = \begin{pmatrix} T_0(x) \\ \vdots \\ T_k(x) \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ \cos(k\arccos(x)) \end{pmatrix}$$

$$t'_k = \begin{pmatrix} T'_0(x) \\ \vdots \\ T'_k(x) \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ \frac{k\sin(k\arccos(x))}{\sqrt{1-x^2}} \end{pmatrix}$$

$$mt'_k = \begin{pmatrix} T'_0(x - \frac{T_0(x)}{T'_0(x)}) \\ \vdots \\ T_k(x - \frac{1}{2}\frac{T_k(x)}{T'_k(x)}) \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ \vdots \\ -\frac{k\left(-\frac{xT_k(x)}{2k\sqrt{1-x^2}\sin(k\arccos(x))} + \frac{T_k^2(x)}{2\sin^2(k\arccos(x))} - \frac{1}{2}\right)\sin\left(k\left(\pi - \arccos\left(\frac{1}{2}\frac{T_k(x)}{T'_k(x)} - x\right)\right)\right)}{\sqrt{1-\left(\frac{1}{2}\frac{T_k(x)}{T'_k(x)} - x\right)^2}} \end{pmatrix}$$

$$p_k(x) = c_0 + ... + c_k\cos(k\arccos(x))$$

$$p'_k(x) = c_1 + ... + c_k\frac{k\sin(k\arccos(x))}{\sqrt{1-x^2}}$$

$$p'_k\left(x - \frac{1}{2}\frac{p_k(x)}{p'_k(x)}\right) = c_1\left(-\frac{\left(-\frac{x^2}{2\sqrt{1-x^2}\sin(\arccos(x))} + \frac{x^2}{2\sin^2(\arccos(x))} - \frac{1}{2}\right)\sin\left(\left(\pi - \arccos\left(-\frac{1}{2}x\right)\right)\right)}{\sqrt{1+\frac{1}{4}x^2}}\right)$$

$$+...+c_k\left(-\frac{k\left(-\frac{xT_k(x)}{2k\sqrt{1-x^2}\sin(k\arccos(x))} + \frac{T_k^2(x)}{2\sin^2(k\arccos(x))} - \frac{1}{2}\right)\sin\left(k\left(\pi - \arccos\left(\frac{1}{2}\frac{T_k(x)}{T'_k(x)} - x\right)\right)\right)}{\sqrt{1-\left(\frac{1}{2}\frac{T_k(x)}{T'_k(x)} - x\right)^2}}\right)$$

And thus, for the case of $n = k+1$, it trivially follows from the assumptions for the case $n = k$ cause it encompasses the same elements of the vectors

t_k, t'_k, and mt'_k, but with an extra elements that is trivially proved by the Jarratt's Method. Therefore $P(k) \implies P(k+1)$. And this concludes our proof. $\square$

4.4 Proof of Time Complexity

Let T be the function that is equal to the running time.
First let us consider the running time of the first part of the algorithm. Let c_i be the cost of the command execution, where $i = 1, ..., 17$ (since we have 17 command lines in this algorithm).
We denote by t_i the time for each command to execute, where $i = 1, ..., 17$.
Going through the algorithm by command lines calculating the time of each command in relations with its cost we get

$$
\begin{aligned}
T(n) = &\sum_{i=1}^{n} c_i t_i \\
&= c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7(n+1) + c_8 n + c_9 n + c_{10} n + c_{11} n + c_{12} n \\
&\quad + c_{13} n + c_{14} n + c_{15} n + c_{16} n + c_{17} \\
&= c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 n + c_7 + c_8 n + c_9 n + c_{10} n + c_{11} n + c_{12} n \\
&\quad + c_{13} n + c_{14} n + c_{15} n + c_{16} n + c_{17} \\
&= n(c_7 + c_8 + c_9 + c_{10} + c_{11} + c_{12} + c_{13} + c_{14} + c_{15} + c_{16} + c_{17}) + (c_1 + c_2 \\
&\quad + c_3 + c_4 + c_6 + c_7) \\
&= n(c^{(1)}) + c^{(2)},
\end{aligned}
$$

where $c^{(i)}$ is a constant with $i = 1, 2$. Thus the time is linear for this algorithm with $O(n)$.

Now let us consider the running time of the second part of the algorithm where we evaluate the Jarratt's Method. Let c_j be the cost of the command execution, where $j = 1, ..., 14$ (since we have 14 command lines in this algorithm).
We denote by t_j the time for each command to execute, where $j = 1, ..., 14$.
Going through the algorithm by command lines calculating the time of each

command in relations with its cost we get

$$
\begin{aligned}
T(n) &= \sum_{i=1}^{n} c_i t_i \\
&= c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7(i+1) + c_8 in + c_9 i + c_{10} i + c_{11} i + c_{12} i \\
&\quad + c_{13} i + c_{14} \\
&= c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 i + c_7 + c_8 in + c_9 i + c_{10} i + c_{11} i + c_{12} i \\
&\quad + c_{13} i + c_{14} \\
&= in(c_8) + i(c_7 + c_8 + c_9 + c_{10} + c_{11} + c_{12} + c_{13}) + (c_1 + c_2 + c_3 + c_4 + c_5 \\
&\quad + c_6 + c_7 + c_{14}) \\
&= in(c^{(1)}) + n(c^{(2)}) + c^{(3)},
\end{aligned}
$$

where $c^{(1)}$, $c^{(2)}$, and $c^{(3)}$ are constants. Thus the time is exponential for this part of the algorithm with $O(in)$ where i is the given number of iterations.

Therefore we have a quadratic time overall with a $O(in)$.

5 Experimentation

In what follows, we will begin by testing the correctness of both the Jarratt's Method and the Chebyshev parts of the algorithm discussed in this paper. We will test the correctness of both parts of the algorithm in a concrete example, first solved by hand, and then applied to both parts of the algorithm. Then finally we will test the time of the algorithm as a whole discussed in this paper by running it on multiple polynomials with increasing norm value, then plotting the time of the algorithm as a whole on a graph. All of the testing is done using the matlab code provided with this paper.

5.1 Experimentation on the Correctness

Let $c = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, $x_0 = 0.2$, $tol = 0.01$, and $i = 5$. We know from using the definition of the Jarratt's Method that we have

$$
t = \begin{pmatrix} 1 \\ 0.1 \\ -0.98 \end{pmatrix}, t' = \begin{pmatrix} 0 \\ 1 \\ 0.4 \end{pmatrix}, mt' = \begin{pmatrix} 0 \\ 0.5 \\ -62.275 \end{pmatrix},
$$

thus we get

$$p_2(x_0) = 0.12, p_2'(x_0) = 1.4, p_2'\left(x_0 - \frac{1}{2}\frac{p_2(x_0)}{p_2'(x_0)}\right) = -61.775,$$

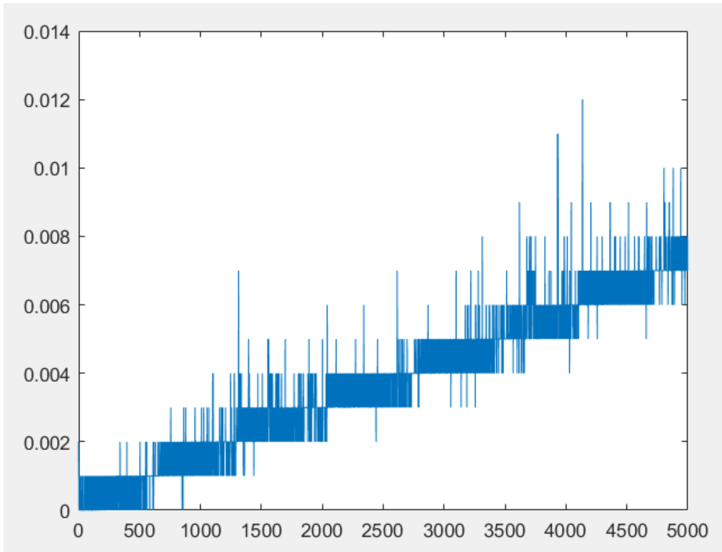
with $|curr| = 0.002$ and $i = 1$. Therefore, $x_1 = 0.102$ is the answer.

We get almost the exact same approximation using the algorithm applied in matlab

```
>> c=[1,1,1];x0=0.1;tol=0.01;maxi=5;[x,i]=jarratt(x0,c,tol,maxi)
    "t:1"    "t:0.1"    "t:-0.98"

    "dt:0"    "dt:1"    "dt:0.4"

    "mdt:0"    "mdt:0.5"    "mdt:-62.275"

p: 0.12
dp: 1.4
mdp: -61.775
corr: -0.0019425

x =

      0.10194


i =

     1
```

5.2 Experimentation on the Time

We have run the algorithm consecutively five thousand times, each time running them on a polynomial that is one degree higher than the one before, starting with the degree one, going all the way up to five thousand. We have obtained the following graph where the blue line is the time for the algorithm described in this paper



The graph shows that the time is quadratic, just how we have proven it to be for our proposed Jarratt's Method algorithm.

6 Conclusion

In this paper we have devised a way of approximating the zero value of a polynomial $p_n(x)$, made out of Chebyshev polynomials $T_i(x)$ where $i = 0, ..., , n$, using the Jerratt's Method algorithm that we've devised as a solution to the given statement problem. We have proven that the algorithm provided is a valid method of approximating the zero of the polynomial $p_n(x)$. We have also proven that the proposed algorithm have the time complexity of time $O(in)$ where n is the degree of the polynomial $p_n(x)$, and i is the number of iterations it took to reach the tolerance *tol* provided to the algorithm. And finally we have concluded with experimentation that shows that the algorithm works in practice as well. This paper has been accompanied with three files, being

1. "main.m" is the file that we have used to display the time complexity,
2. "chebyshev.m" is the file that creates the polynomial along with its Chebyshev components for the main calculations for our tests,
3. "jarratt.m" is the file that uses the Jarratt's Method algorithm described in this document to approximate $p_n(x)$.

References

[1] Jarratt, P. (1966) *Multipoint iterative methods for solving certain equations.* The Computer Journal, Volume 8, Issue 4, January 1966, Pages 398–400.

[2] McNamee, M.J. (2007) *Numerical methods for roots of polynomials, Part 1.* Amsterdam, First Edition, Pages 141–151.