**COMPUTER PROGRAMME FOR THE GENERATION OF COEFFICIENTS OF CHEBYSHEV APPROXIMATION FUNCTIONS USING VISUAL C++.**

**AKINGBOGUN OLUWOLE. O.**

140403548

DEPARTMENT OF ELECTRICAL/ELECTRONICS ENGINEERING

UNIVERSITY OF LAGOS, LAGOS NIGERIA.

April 2016.

Abstract

This report concentrates on one of the important stages of synthesis of filters. It deals with the coefficient of the Chebyshev which is an intermediate step in filter development. The program developed for calculation of this coefficients is developed along with the output. The program is very advantageous and also comes with its limitations.

# Introduction to Filter Approximations

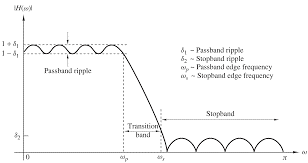
Approximations are mathematical methods used in translating the filter specifications into realizable transfer functions. With these established approximations, a filter designer can choose from it and design a filter with ease. (Dimopoulos)

Filter approximations deal with normalized low pass specifications since they are not easily denormalized but can be transformed to any filter type, such as high pass, band pass, band reject filters.

The Chebyshev approximation is a commonly used approximation.

# Chebyshev Approximation

The Chebyshev approximation is an approximation whose attenuation at the stop band is very attainable. Large attenuation at the stop band is achieved by changing the approximation conditions the pass band. The criterion used is to minimize the maximum deviation from the ideal flat characteristic. The Chebyshev polynomials are ideal for this purpose. A main feature of the Chebyshev approximation is the equiripple characteristic of the passband as shown below:



# The Chebyshev Polynomial

The nth order Chebyshev polynomial is stated as follows:

Knowledge of the fact that and makes computation of the other values easy by recursion using the formula:

# Roots of Chebyshev Function

The Chebyshev filter has magnitude of:

With n being the filter order, being a parameter which controls the amount of ripple on the pass band and being the upper pass band edge.

The roots of this equation is given by:

Where,

With,

The transfer function can be expressed as:

K is the denominator constant given as:

# The Coefficients

With the formulas given above, the coefficients of the Chebyshev pole expansion is analyzed below:

The coefficients of and the constant are seen to be and respectively.

# Generating the coefficients using C++ language

The coefficients given in the previous section are more conveniently calculated using a computer program. The program of choice for this project is the C++ programming language. The integrated development environment(compiler) used is the Microsoft Visual Studio. The components of the program consist of:

* The epsilon function given as (function of Amax).
* The sigma function(σ) as a function of n, k and Amax.
* The omega function(ω) as a function of n, k and Amax
* The denominator constant(K) as a function of n and Amax.
* The main function that outputs the coefficients calculated.

# Program Listing

#include <iostream>

#include <cmath>

#include <iomanip>

#define PI 3.14159265

using namespace std;

struct roots

{

double real;

double complex;

};

//function prototypes

void studentInfo();

void titleOutput(double ApMax);

double parameterCalc(double ApMax);

roots rootCalc(double order, double k, double parameter);

void output(roots root);

double denomCalc(double ApMax, double order);

int main()

{

//variable declaration

double ApMax[] = {0.25, 0.50, 1.00};

double orders[] = {2, 3, 4, 5};

double parameter, temp, k, den;

roots root;

//program begins

studentInfo();

//loop for the passband ripples, ApMax

for(int x = 0; x < 3; x++)

{

titleOutput(ApMax[x]);

parameter = parameterCalc(ApMax[x]);

//loop for the set of orders

for(int y = 0; y < 4; y++)

{

cout << int(orders[y]) << "\t";

temp = orders[y];

k = 0;

while (temp > 0)

{

root = rootCalc(orders[y], k, parameter);

if (root.complex < 1e-3)

{

cout << "(s + " << root.real << ")";

break;

}

output(root);

k += 1;

temp -= 2;

}

cout << setw((5-y)\* 13);

cout << denomCalc(ApMax[x], orders[y]);

cout << endl;

}

cout << endl;

}

return 0;

}

//This function prints the students and program information

void studentInfo()

{

cout << "NAME: AKINGBOGUN OLUWOLE. O." << endl;

cout << "MATRIC NO: 140403548" << endl;

cout << "DEPARTMENT: ELECTRICAL/ELECTRONIC ENGINEERING" << endl;

cout << "COURSE CODE: EEG 407" << endl;

cout << "COURSE TITLE: ACTIVE NETWORKS AND SYNTHESIS" << endl;

cout << "PROGRAM: GENERATION OF CHEBYSHEV COEFFICIENTS" << endl;

cout << "PACKAGE USED: C++" << endl;

cout << endl;

}

//This function prints the title head for the coefficients

void titleOutput(double ApMax)

{

cout << "Maximum loss, ApMax = " << ApMax << "dB" << endl;

cout << "Order\tNumerator of H(s)" << setw(80) << "Denominator Constant" << endl;

}

//This function calculates the value for the parameter, Epsilon

double parameterCalc(double ApMax)

{

return sqrt(pow(10, 0.1\*ApMax) - 1);

}

//This function calculates the roots for each order

roots rootCalc(double order, double k, double parameter)

{

roots root;

double A = ((1 + (2.0\*k)) / order) \* (PI / 2.0);

double B = (1.0 / order) \* (asinh(1.0/parameter));

root.real = sin(A) \* sinh(B);

root.complex = cos(A) \* cosh(B);

return root;

}

//This function prints the coefficients in a polynomial form

void output(roots root)

{

cout << showpoint << fixed << setprecision(5);

double coeff1 = 2.0 \* root.real;

double coeff2 = pow(root.real, 2) + pow(root.complex, 2);

cout << "(s^2 + " << coeff1 << "s + " << coeff2 << ")";

}

//This function evaluates the denominator constant

double denomCalc(double ApMax, double order)

{

double den = 0.0;

double temp = 0.0;

temp = pow(2, order - 1);

den = 1 / (temp \* parameterCalc(ApMax));

return den;

}

# Output

NAME: AKINGBOGUN OLUWOLE. O.

MATRIC NO: 140403548

DEPARTMENT: ELECTRICAL/ELECTRONIC ENGINEERING

COURSE CODE: EEG 407

COURSE TITLE: ACTIVE NETWORKS AND SYNTHESIS

PROGRAM: GENERATION OF CHEBYSHEV COEFFICIENTS

PACKAGE USED: C++

Maximum loss, ApMax = 0.25dB

Order Numerator of H(s) Denominator Constant

2 (s^2 + 1.79668s + 2.11404) 2.05406

3 (s^2 + 0.76722s + 1.33863)(s + 0.76722) 1.02703

4 (s^2 + 0.42504s + 1.16195)(s^2 + 1.02613s + 0.45485) 0.51351

5 (s^2 + 0.27005s + 1.09543)(s^2 + 0.70700s + 0.53642)(s + 0.43695) 0.25676

Maximum loss, ApMax = 0.50000dB

Order Numerator of H(s) Denominator Constant

2 (s^2 + 1.42562s + 1.51620) 1.43139

3 (s^2 + 0.62646s + 1.14245)(s + 0.62646) 0.71569

4 (s^2 + 0.35071s + 1.06352)(s^2 + 0.84668s + 0.35641) 0.35785

5 (s^2 + 0.22393s + 1.03578)(s^2 + 0.58625s + 0.47677)(s + 0.36232) 0.17892

Maximum loss, ApMax = 1.00000dB

Order Numerator of H(s) Denominator Constant

2 (s^2 + 1.09773s + 1.10251) 0.98261

3 (s^2 + 0.49417s + 0.99420)(s + 0.49417) 0.49131

4 (s^2 + 0.27907s + 0.98650)(s^2 + 0.67374s + 0.27940) 0.24565

5 (s^2 + 0.17892s + 0.98831)(s^2 + 0.46841s + 0.42930)(s + 0.28949) 0.12283

Process returned 0 (0x0) execution time : 0.043 s

Press any key to continue.

# Conclusion

The program output show high accuracy. It responded fast in calculating the coefficients. The limitation of using this program is the accessibility to the proper ide to run the program. The project points out an advantage the computer has in filter analysis and design.

Java, C#, Matlab, Python are also good substitutes as programming languages used in generating Chebyshev coefficients.

# Reference

Dimopoulos, H. G. (n.d.). *Analog Electronic Filters: Theory, Design and Synthesis.*