JAVA PRÜFUNG I

Bedingungen:

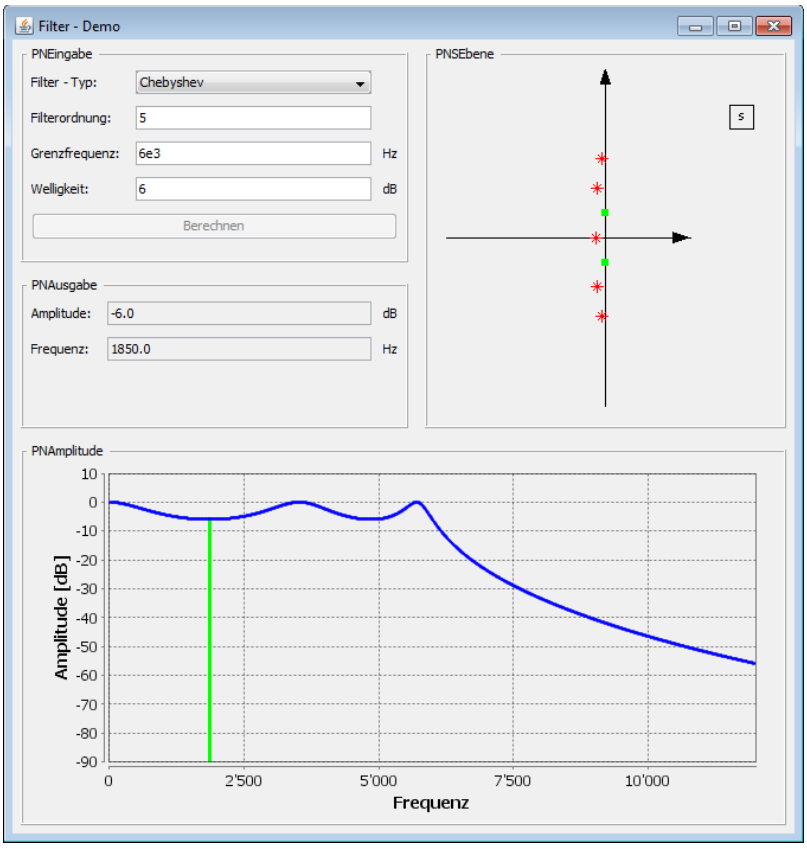
- Erlaubte Hilfsmittel: Unterrichtsunterlagen, Java Buch und Übungen.
- **Die Prüfung ist schrittweise, gemäss Aufgabenstellung lokal auf Ihrem Computer zu lösen. Kopieren sie zu diesem Zwecke den gesamten Ordner *P1_FilterDemo_FS2015_Vorlage* auf Ihre lokale Harddisk und importieren sie das Projekt in Eclipse. Am Ende der Prüfung ist der Ordner src umbenannt in *NameVorname* abzugeben.**
- **Setzen sie als erstes Ihren Namen und Vornamen in die Dateien.**
- **Gegenseitiges Abschreiben in irgendeiner Form führt zur Note 1!**
- **Folgend sie bei der Wahl von Variablen den Angaben in der Aufgabenstellung.**
- **Die Beilage muss abgegeben werden!**

Beschreibung:

Ziel der Applikation ist die Berechnung und Visualisierung von Butterworth und Chebyshev - 1 - Tiefpassfiltern. Lineare, zeitinvariante Filter lassen sich durch eine rational gebrochene Funktion im Laplace - Bereich beschreiben:

$$H(s) = \frac{B(s)}{A(s)}$$

Das Zählerpolynom $B(s)$ beschreibt die Lage der endlichen Nullstellen, das Nennerpolynom $A(s)$ beschreibt die Lage der Polstellen der Übertragungsfunktion. Beim Butterworth - Filter liegen die Pole auf einem Halbkreis in der LHE der Laplace - Ebene, beim Chebyshev - 1 - Filter liegen sie auf einer halben Ellipse in der LHE. Die Nullstellen liegen alle bei unendlich.



Struktur:

Die Applikation ist im bekannten Model-View-Controller Entwurfsmuster gehalten. Die View implementiert die graphische Benutzerschnittstelle und erlaubt die Eingabe der Filterspezifikation und repräsentiert die berechneten Pole, den Amplitudengang des Filters sowie Frequenz und Amplitude des Testtons. Dem Controller kommt die Steuerung der Applikation zu. Er delegiert Berechnungsaufgaben an das Model. Das Model ist für die Berechnung der entsprechenden Filter, unter Verwendung der Klassen *MicroMatlab* und *FilterFactory* verantwortlich und kapselt die zugehörigen Daten. Zum Aufdatieren der Benutzerschnittstelle wird das Observable - Observer Entwurfsmuster verwendet: Das Model agiert dabei als Observable, die View als Observer. Allfällige Updates werden von der View an die entsprechenden Panel weiter geleitet.

In der Folge sind die Hauptaspekte von View, Controller und Model erläutert. Zum besseren Verständnis, ergänzen das Layout und das Klassendiagramm in der Beilage die Beschreibung.

View:

Die View ist als *GridBagLayout* organisiert und in 4 *JPanel* unterteilt:

PNEingabe beheimatet die Elemente zur Definition des Filters. Mittels *JComboBox* kann zwischen Butterworth- und Chebyshev-Approximation gewählt werden. Die darunter folgenden drei *NumberTextField* erlauben die Eingabe von Filterordnung (Ganzzahl), Grenzfrequenz (Double) und Welligkeit (Double). Der *JButton jbBerechne* dient dazu, die Berechnung auszulösen. Die darunter folgende *RigidArea* wird hinzugefügt, um eine fixe, minimale Breite von 300 zu erzwingen. Sämtliche *JLabel* sind ***LINE_START*** ausgerichtet und skalieren nicht. Da alle relevanten Eingaben auf demselben *JPanel* geschehen, übernimmt die innere Klasse *PNEController* die Steuerung: Der *JButton* erscheint nur *enabled*, wenn die Eingabedaten korrekt sind oder in der Folge Daten korrekte geändert werden.

PNAusgabe beheimatet die Elemente zur Ausgabe von Amplitude und Frequenz des Markers in *PNAmplitude*. Das *PNAusgabe* ist bereits ausprogrammiert.

PNSEbene dient der Darstellung der Pole in der s-Ebene. Mittels Mausrad lässt sich zoomen. Das *PNSEbene* ist bereits ausprogrammiert.

PNAmplitude bringt den Amplitudengang des Filters zur Darstellung. Mittels Mausrad lässt sich grüne Marker bewegen, wodurch entsprechend der zugehörigen Amplitude der Ton ertönt. Das *PNAmplitude* ist bereits ausprogrammiert.

***Controller*:**

Der Controller ist in seiner Funktionalität sehr bescheiden. Er erzeugt das entsprechende Filter und setzt es mittels *setFilter*() ins Model.

***Model*:**

Das *Model* beheimatet das Filter, erzeugt die Frequenzachse und berechnet den zugehörigen Amplitudengang. Das *Model* hat weiter den *WaveTablePlayer* um den Ton zu erzeugen. Die eigentliche Berechnung der Filter geschieht mittels einer *FilterFactory* im *Controller*: Die statischen Methoden *createButter*() und *createCheby1*() erlauben die entsprechenden Filter - Objekte zu erzeugen. Die dazu verwendeten Algorithmen sind in der Aufgabenstellung beschrieben.

Aufgabe 1: User - Interface (40 ~Pte.)

Die Klassen View und PNEingabe enthalten teils Pseudo-Code und sind gemäss Layout-Beschreibung zu programmieren.

a) Implementieren Sie die Klasse View.

b) Implementieren Sie den Konstruktor und die Methode actionPerformed() der Klasse PNEingabe.

Die korrekte Freigabe des JButton und des Textfeldes für die Welligkeit wird erst in Aufgabe 4 ausprogrammiert. kreieren

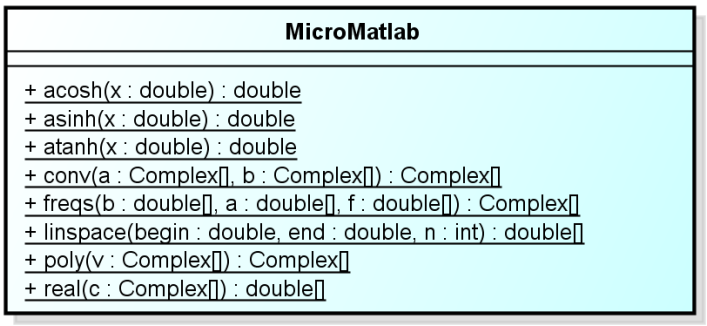
Aufgabe 2: Klasse *Controller* (~8 Pte.)

Der Controller erzeugt das entsprechende Filter und setzt es mittels *setFilter*() ins Model. Das Model.

a) Implementieren Sie die Klasse Controller gemäss Pseudo-Code.

Aufgabe 3: Klasse *Model* (~18 Pte.)

Das *Model* beheimatet das Filter, erzeugt die Frequenzachse und berechnet den zugehörigen Amplitudengang. Das *Model* hat weiter den *WaveTablePlayer* um den Ton zu erzeugen. Zur Berechnung der Filter und des Amplitudenganges stehen auch die nachfolgenden Methoden in MicroMatlab zur Verfügung:



a) Implementieren Sie die Klasse Model gemäss Angaben im Code.

Aufgabe 4: Klasse *PNEController* (~22 Pte.)

Die Klasse PNEController steuert die Freigabe der GUI - Elemente: Der JButton erscheint nur *enabled*, wenn die Eingabedaten korrekt sind oder in der Folge Daten korrekte geändert werden. Da es sich um eine innere Klasse handelt, kann sie direkt auf die Elemente der äusseren Klasse zugreifen.

a) Ergänzen Sie den Konstruktor der Klasse *PNEingabe* entsprechend der Beschreibung im Code.

b) Implementieren Sie die Methoden der Klasse *PNEController*.

c) Testen Sie ob die Freigabe der GUI - Elemente richtig funktioniert.

Aufgabe 5: Klasse *FilterFactory* (~21 Pte.)

Der FilterFactory kommt die Berechnung der beiden Filtertypen zu. Zur Berechnung stehen wiederum die Methoden in MicroMatlab mit zur Verfügung.

a) Implementieren Sie die beiden Methoden gemäss nachfolgender Beschreibung.

Berechnung der Filter

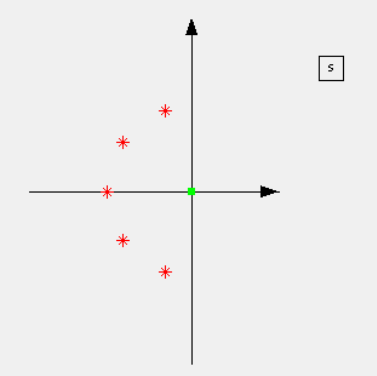
Lineare, zeitinvariante Filter lassen sich durch eine rational gebrochene Funktion im Laplace - Bereich beschreiben:

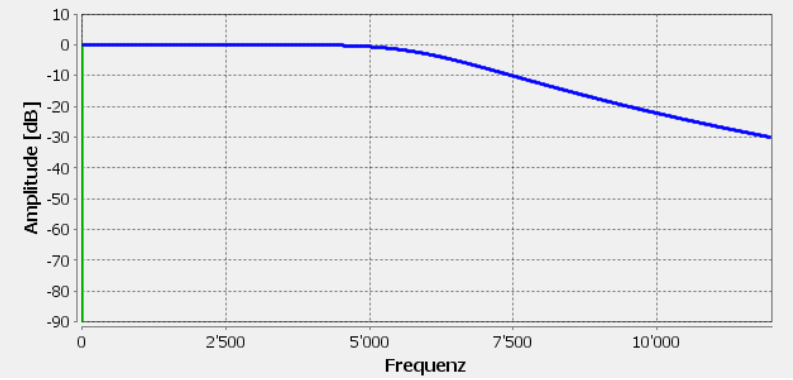
$$H(s) = \frac{B(s)}{A(s)}$$

Das Zählerpolynom $B(s)$ beschreibt die Lage der endlichen Nullstellen, das Nennerpolynom $A(s)$ beschreibt die Lage der Polstellen der Übertragungsfunktion. Die Differenz der Ordnung von $A(s)$ und $B(s)$ definiert die Zahl der Nullstellen, die bei unendlich liegen. Sind alle Nullstellen bei unendlich, zerfällt $B(s)$ in die Konstante b_0.

Butterworth - Filter

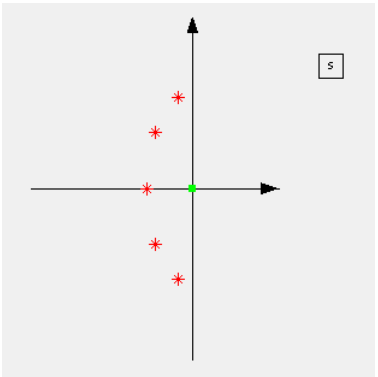
Bei einem Butterworth - Filter der Ordnung N und Grenzkreisfrequenz ω_g liegen sämtliche Nullstellen bei unendlich. Die Pole liegen auf einem Halbkreis in der linken Halbebene in s:

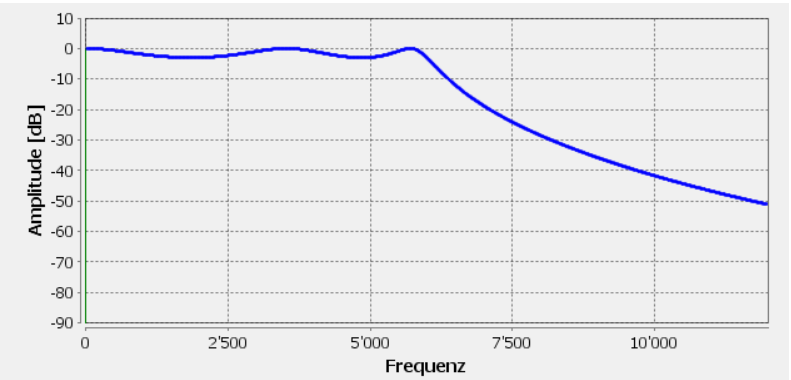




Chebyshev - 1 - Filter

Bei einem Chebyshev - 1 - Filter der Ordnung N mit Grenzkreisfrequenz ω_g und Welligkeit R liegen sämtliche Nullstellen bei unendlich. Die Pole liegen auf einer Ellipse in der linken Halbebene in s. Sie gehen direkt durch Stauchung aus den Polen des Butterworth - Filters hervor.





Aufgrund der Pole kann die Übertragungsfunktion nun wie folgt gebildet werden:

$$H(s) = \frac{b_0}{(s-p_0)\cdot \ldots \cdot (s-p_{N-1})} = \frac{b_0}{A(s)}$$

Die Konstante b_0 ist durch den Konstanten Term im Polynom $A(s)$ gegeben.

Algorithmen zur Berechnung der Filter

Butterworth - Filter

Input:

N	Ordnung des Filters
Wg	Grenzkreisfrequenz des Filters

Output:

Filter - Objekt	Objekt mit Zähler- und Nennerpolynom und zugehörigen Pol- und Null stellen.

Algorithmus:

1. Die Lage der N Pole für n gleich Null bis N-1 berechnen:

$$p_n = e^{j\left(\frac{n\cdot\pi}{N}+\frac{\pi}{2}\left(1+\frac{1}{N}\right)\right)} \cdot \omega_g$$

2. Aufgrund der Pole das Nennerpolynom berechnen:

$$A(s) = (s - p_0) \cdot \ldots \cdot (s - p_{N-1}) = a_0 \cdot s^N + \cdots + a_N$$

3. Aufgrund des konstanten Terms a_{N-1} in $A(s)$ das Polynom 0-ter Ordnung $B(s)$ festlegen:

$$B(s) = a_N$$

4. Filter - Objekt mit Zählerpolynom $B(s)$, Nennerpolynom $A(s)$ und den Polstellen p_n erzeugen.

Matlab - Code:

```
n = 0:N-1;
p = exp(j*(n*pi/N + pi/2*(1+1/N)))*Wg;
A = real(poly(p));
B = A(N+1);
```

wobei p die Pole beschreibt, A und B die Koeffizienten von $A(s)$ und $B(s)$ sind und Wg die Grenzkreisfrequenz des Filters ist.

Chebyshev - 1 - Filter

Input:

N	Ordnung des Filters
R	Welligkeit des Filter in dB
Wg	Grenzkreisfrequenz des Filters

Output:

Filter - Objekt	Objekt mit Zähler- und Nennerpolynom und zugehörigen Pol- und Null stellen.

Algorithmus:

1. Aufgrund der Welligkeit R den Wert ε berechnen

$$\varepsilon = \sqrt{10^{\frac{R}{10}} - 1}$$

2. Faktor zur Stauchung des Realteils von p_n berechnen

$$s_{re} = \sinh\left(\frac{1}{N} \cdot \operatorname{asinh}\left(\frac{1}{\varepsilon}\right)\right)$$

3. Faktor zur Stauchung des Imaginärteil von p_n berechnen

$$s_{im} = \cosh\left(\frac{1}{N} \cdot \operatorname{asinh}\left(\frac{1}{\varepsilon}\right)\right)$$

4. Die Lage der N Pole für n gleich Null bis N-1 berechnen:

$$p_n = e^{j\left(\frac{n\cdot\pi}{N} + \frac{\pi}{2}\left(1+\frac{1}{N}\right)\right)} \cdot \omega_g$$

   und den Realteil mit s_{re} und den Imaginärteil mit s_{im} multiplizieren um die Lage der Pole $\tilde{p}_n$ zu erhalten.

5. Aufgrund der Pole das Nennerpolynom berechnen:

$$A(s) = (s - \tilde{p}_0) \cdot \ldots \cdot (s - \tilde{p}_{N-1}) = a_0 \cdot s^N + \cdots + a_N$$

6. Aufgrund des konstanten Terms a_{N-1} in $A(s)$ das Polynom 0-ter Ordnung $B(s)$ festlegen:

$$B(s) = a_N$$

7. Filter - Objekt mit Zählerpolynom $B(s)$, Nennerpolynom $A(s)$ und den Polstellen p_n erzeugen.

Matlab - Code:

```
eps = sqrt(10^(R/10)-1.0);
sre = sinh((1.0/N)*asinh(1.0/eps));
sim = cosh((1.0/N)*asinh(1.0/eps));

n = 0:N-1;
p = exp(j*(n*pi/N + pi/2*(1+1/N)))*Wg;
p = (sre*real(p) + j*sim*imag(p));

A = real(poly(p));
B = A(N+1);
```

wobei `p` die Pole beschreibt, `A` und `B` die Koeffizienten von $A(s)$ und $B(s)$ sind, `R` die Welligkeit und dB und `Wg` die Grenzkreisfrequenz des Filters ist.