TD Langage C+
Séances 4, 5 et 6 /8

TD 4, 5 et 6 – Impédance

Dans le cadre de ces séances, en plus des algorithmes liés au projet, vous implémenterez une petite interface homme machine (IHM), des accès aux fichiers et des appels à un autre programme.

Attachez la plus grande rigueur à votre méthode de travail et à la validation de vos résultats.

Consignes générales :

- Ecrire peu de ligne, puis valider ces quelques lignes (debug, …)
- Ecrire des fonctions,
- Organisez votre développement

Etude de cas 1 : Calcul de l'impédance de circuit électrique et tracé du module et de la phase

Vous disposer de trois séances pour réaliser cette petite application. Celle-ci doit être écrite en C/C++ sous QtCreator. La visualisation des courbes se fera avec l'outil « *gnuplot* » disponible pour les OS : Linux, Windows et Mac.

Description de l'application

L'application doit permettre le calcul des impédances pour les 6 combinaisons de 2 composants R, L et C en série et parallèle. L'utilisateur doit dans un premier temps choisir le type de circuit dont il souhaite calculer l'impédance.

Les valeurs des deux composants sont ensuite renseignées par l'utilisateur. Il s'agira de demander uniquement les valeurs des composants correspondant au circuit choisi.

L'impédance se calcule à une fréquence donnée. L'utilisateur choisira le nombre de fréquences ***n*** pour lesquelles il souhaite calculer l'impédance du circuit. Si cette valeur ***n*** est supérieure à 1, il devra alors spécifier une fréquence minimale f_{min} et une fréquence maximale f_{max}. L'impédance du circuit sera alors calculée pour chaque fréquence $f_i \in [f_{\min}, f_{\max}]$, pour i allant de 1 à n (le pas pourra être linéaire ou logarithmique).

Les résultats peuvent être affichés à l'écran soit sous forme numérique (liste de valeurs à l'écran) soit par des courbes tracées par *gnuplot*.

RL série
RC parallèle
…

Gnuplot
écran

I- Un seul circuit… squelette de l'application

I-a Choisir un circuit (R//L, …). Déterminer la formule théorique de l'impédance pour ce circuit.

I-b Faire la fonction qui demande les valeurs des composants du circuit choisi. Tester cette fonction.

I-c Faire la fonction qui demande une fréquence. Tester cette fonction.

I-d Faire la fonction qui **retourne** l'impédance du circuit choisi pour les valeurs des composants et la fréquence choisies (passées en paramètre).

I-e Dans un *main* **afficher** les valeurs du module et de la phase de l'impédance du circuit calculé. Vérifier les valeurs obtenues.

II- Menu et fonctions d'interaction avec l'utilisateur

Penser à réutiliser les fonctions précédentes : il s'agit détendre leur fonctionnement.

Procédez par étape = compiler et tester fréquemment (un *main* compléter petit à petit).

II-a Type de circuit : Faire la fonction gérant l'affichage du menu et la gestion du choix de l'utilisateur.

II-b Valeur des composants : écrire la ou les fonctions permettant de demander à l'utilisateur la valeur de chaque composant du circuit.

II-c Fréquences : écrire la fonction qui demande le nombre de fréquences ainsi que les fréquences minimale et maximale.

Conseil : cette fonction retourne un tableau contenant les valeurs de toutes les fréquences à calculer ainsi que n.

II-d Affichage des résultats : écrire la fonction qui demande le type d'affichage (soit une liste des valeurs à l'écran, soit un affichage de courbes grâce à *gnuplot*) puis appelle la fonction appropriée. Implémenter ensuite la fonction d'affichage à l'écran.

III- Fonctions de calcul

Dans ces fonctions, aucune interaction avec l'utilisateur.

Quelques conseils, à prendre ou à laisser :

- **Nombre de fonctions :** Il y a 6 circuits, donc 6 impédances différentes, 6 fonctions.
- **Problème des complexes ... :** Dans le cas général les impédances seront complexes. Il faudra soit retourner des impédances complexes, soit retourner le module et l'argument de l'impédance.
- **…et des fréquences :** Retourner toujours des tableaux, et de même taille que le nombre de fréquence à calculer.

III-a Implémenter votre solution.

III-b Tester votre solution et gérer les problèmes arithmétiques.

IV – GNU Plot enregistrement de fichier

L'affichage des courbes se fait par le programme gnuplot. En C/C++ pour lancer un programme externe, il faut utiliser l'instruction « ***system*** ». Exemple :

```
#include <iostream>
#include <stdlib.h>
using namespace std;

int main(void)
{ system("gnuplot graph.p");
  cin.sync();
  cin.get();
  return 0;
}
```

Cet exemple lance le programme *gnuplot* avec le paramètre « graph.p » qui est un fichier de configuration.

Le rôle du fichier de configuration est de déterminer comment afficher les valeurs et dans quel fichier il faut aller lire les données. Voici un exemple de fichier de configuration de *gnuplot* :

```
# Lines
set style data lines
set size 1.0, 1.0
set origin 0.0, 0.0
set multiplot

# Magnitude response
# ============================================
set size 1.0, 0.5
set origin 0, 0.5
set grid
set logscale x 10
set logscale y 10
set title "Module"
set xlabel "Frequence (Hz)"
set ylabel "Impedance |Z|"

plot "Impedance.dat" using 1:2 title "Module"

#Phase response
# ============================================
set size 1.0, 0.5
set origin 0, 0
# Axis labels
set title "Phase"
set ylabel "Phase (degrees)"
set xlabel "Frequence (Hz)"
unset logscale y
plot "Impedance.dat" using 1:3  title "Phase"

pause -1 "Hit return"
exit
```

La majeure partie de ce fichier décrit comment tracer le gain et la phase (couleur, échelle log, légende, points liés, …). L'instruction pour tracer une courbe est « ***plot*** ». Dans l'exemple, le fichier « Impedance.dat » contient les données à afficher. Il s'agit d'un fichier texte. Reportez-vous au document ***NoteCpp.pdf***, section ***3.10 Fichier***.

IV-a Ecrire la fonction permettant d'enregistrer vos données dans un fichier texte : une colonne fréquence, une colonne module et une colonne phase. Chaque colonne étant séparée par un espace. On crée une nouvelle ligne pour chaque fréquence.

IV-b Ecrire la fonction permettant d'exécuter le programme *gnuplot* avec un fichier de configuration. Penser à créer ce fichier de configuration…

IV-c Tester votre solution et améliorer votre application.

V – Pour aller plus loin

Uniquement si le reste fonctionne, pour les motivés, voici quelques challenges.

V-a Structures et classes : une méthode pour éviter le passage de nombreux paramètres à une fonction est le regroupement de ces variables au sein d'une structure de données. En C++ on dispose des structures (***struc***) et des classes (***class***). Les informations de l'utilisateur peuvent être regroupées dans une structure, de même pour les informations d'impédance (trinôme : fréquence, amplitude et phase), de même pour les nombre complexes.

V-b Création d'une interface graphique : Remplacement de *gnuplot* par des widgets. Pour des tracés type Bode, la bibliothèque QWT permet de faire très rapidement de jolies choses… Après avoir téléchargé cette bibliothèque, compiler la (suivre les instructions du fichier INSTALL, puis regarder l'exemple « bode »).

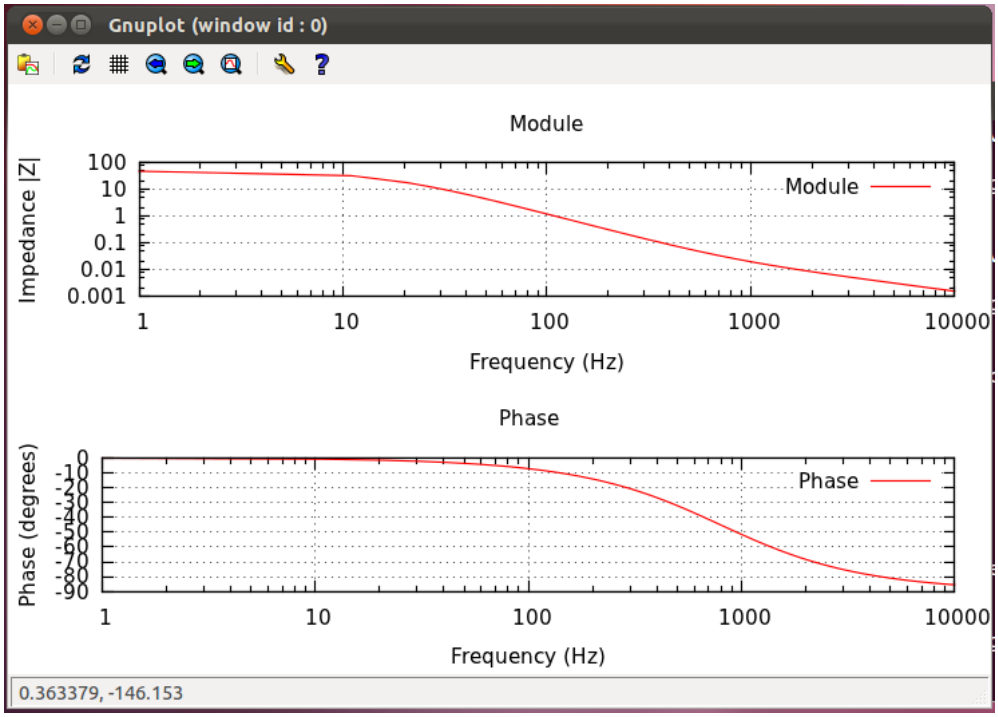

Figure 1: Exemple de tracé avec GNU Plot