Teoría de Circuitos 2018
Trabajo Práctico de Laboratorio Nº4

Diseño de Filtros Analógicos

Instrucciones

Se pide a cada grupo que escriba un programa que permita diseñar filtros analógicos a partir de las funciones aproximación estudiadas en clase. Se plantea a continuación un esquema básico de 3 etapas de diseño, el cual podrá ser modificado en caso de que el grupo lo considere apropiado, siempre y cuando se conserve la funcionalidad prevista del programa.

Consideraciones generales:

- Se les recuerda a los alumnos que la política de Fraude y Plagio del Instituto rige sobre este trabajo.

Pautas para la evaluación (en orden de importancia):

Para cumplir condición de aprobación (nota = 4):

- **Cumplimiento de toda la funcionalidad pedida.**
- **Manejo e incorporación de los conceptos teóricos.**
- **Buena organización, comentarios, limpieza de variables y figuras.**
- **La visualización de todos los gráficos debe tener sus adecuadas referencias, escalas, grillas y la posibilidad de realizar zoom, paneo y tracking.**
- **Todas las funciones de cálculo de transferencias en forma normalizada deben estar escritas en su totalidad por el grupo.**

Los siguientes puntos sumarán a la nota:

- Buen manejo de errores numéricos.
- Posibilidad de agregar aproximaciones y tipos de filtros sin necesidad de cambiar todo el código.
- Diseño intuitivo y prolijo de la interfaz gráfica.
- Originalidad e inventiva.
- Aportes no obligatorios.

*El mejor programa presentado recibirá **doble** calificación, cada una de ellas equivalente a un TP de laboratorio.*

Entregas:

El desarrollo de la aplicación consistirá de una entrega y dos rondas de correcciones, en las cuales cada grupo presentará la funcionalidad o las modificaciones realizadas. Se aceptarán entregas del código fuente hasta el día correspondiente a la entrega a las 13:00 Hs. Todo cambio posterior pasará a la entrega siguiente.

- Jueves 17 de Octubre de 2019 a las 10:00 Hs (Presentación de plan de trabajo a la cátedra y consultas - Sin nota).
- Jueves 31 de Octubre de 2019 a las 10:00 Hs (Entrega final y presentación grupal).

Etapa 1

La primera etapa del programa deberá permitir al usuario elegir la aproximación apropiada para la plantilla que el mismo ingresa.

Datos provistos por el usuario

Tipo de Filtro

Pasa-bajos	Pasa-banda
Pasa-altos	Rechaza-banda
Retardo de grupo	

Aproximaciones obligatorias para cumplir condición de aprobación (TODAS)

Grupo 1	Grupo 2	Grupo 3	Grupo 4
Butterworth	Chebycheff	Legendre	Cauer
Bessel	Chebycheff Inverso	Gauss	

Opcionales

Transicionales(Opcional)	Custom (Opcional)

Para cada aproximación se debe poder elegir entre un rango de desnormalización de 0 % ($A(\omega_p) = A_p$) y 100 % ($A(\omega_a) = A_a$).

Alternativas de cálculo

- Mínimo y máximo orden para cada aproximación. (aún si no cumple plantilla) .
- Orden definido por el usuario (aún si no cumple plantilla).
- Especificando el Q máximo permitido (aún si no cumple plantilla).

Gráficos

Se deben mostrar al menos los siguientes gráficos.

- Curvas de atenuación (original y normalizada) en escala apropiada y posibilidad de superponer la plantilla, indicando el orden utilizado.
- Fase.
- Retardo de grupo.
- Máximo Q correspondiente a cada curva.
- Respuesta al impulso.
- Respuesta al escalón.
- Diagrama de polos y ceros (la escala del eje de absisas debe coincidir con la escala del eje de ordenadas por defecto).

La visualización debe tener de forma preseteada las bandas de interés correspondientes a cada tipo de desnormalización.

Etapa 2

Una vez elegida una única función aproximación a ser implementada, la segunda etapa debe permitir la división en etapas de segundo y primer orden a partir de la transferencia total.

Funcionalidad prevista

- Separación de H(s) en etapas de segundo orden en forma automática, poniendo sus parámetros de interés en evidencia.
- Cálculo de rango dinámico total.
- La ganancia de cada etapa y el orden de las mismas deberá poder ser variable por el usuario.
- Crear, borrar y cambiar la composición de etapas completas, de forma gráfica.
- Gráfica de las transferencias de las etapas en cascada o individualmente, en escala apropiada.
- **Funcionalidad de división y ordenamiento automático de etapas y determinación de la ganancia de cada una para maximización del rango dinámico. (OPCIONAL)**

Etapa 3 (OPCIONAL)

Una vez definidas todas las etapas de la transferencia y sus respectivas ganancias, el programa debe definir la implementación en circuitos para cada etapa, en función de sus parámetros.

Funcionalidad prevista

- Sugerencia de las celdas apropiadas para cada etapa, con los valores de sus componentes (valores nominales).
- Indicación de todas las sensibilidades involucradas y su efecto en los parámetros de la celda.
- Posibilidad de elección de la celda por parte del usuario (sólo se deben permitir aquellas celdas adecuadas para implementar cada etapa).
- Posibilidad de cambiar el valor de los componentes de cada celda.
- Generación y exportación de un informe en HTML con todos los parámetros del diseño del filtro y sus respectivos gráficos, presentados de forma adecuada.

Aspectos adicionales

- Se debe tener en cuenta que el usuario puede querer visualizar más de un orden de una misma aproximación al mismo tiempo.
- Debe existir la posibilidad de comenzar el proceso desde su inicio, descartando todo lo realizado hasta el momento.
- Debe existir la posibilidad de volver a etapas anteriores del diseño.
- El programa debe poder guardar el estado de la sesión en un archivo para ser reabierto luego.
- Incorporar un instructivo de uso.
- Se debe poder exportar la función transferencia (numerador y denominador) final y de cada etapa de forma tal de poder ser utilizada.
- Se recomienda **fuertemente** el uso de herramientas de control de versiones como Git o SVN, a fin de evitar sorpresas inesperadas tanto durante el desarrollo como entregas.
- Si bien se recomienda realizar el trabajo práctico utilizando Python, se encuentra abierta la posibilidad a aquellos grupos que lo deseen de realizarlo en Matlab, Java o C++ (consultar por otros lenguajes), con la asistencia de librerías matemáticas o científicas que se encuentren disponibles y asegurando compatibilidad con Windows, Linux y Mac. En caso de ser así, el grupo deberá informarlo a la cátedra lo antes posible para recibir toda la asistencia necesaria.

Páginas web de interés

Control de versiones

- https://bitbucket.org/
- https://github.com/
- https://riouxsvn.com/
- http://tortoisesvn.net/

Python

- https://wiki.python.org/moin/GuiProgramming
- https://python-para-impacientes.blogspot.com/p/tutorial-de-tkinter.html
- http://www.tutorialspoint.com/python/python_gui_programming.htm
- http://docs.python-guide.org/en/latest/scenarios/gui/
- http://www.scipy.org/
- https://wiki.python.org/moin/TkInter

Java

- http://www.ee.ucl.ac.uk/~mflanaga/java/
- http://commons.apache.org/proper/commons-math/userguide/index.html
- http://dst.lbl.gov/ACSSoftware/colt/
- http://www.scijava.org/
- https://github.com/yannrichet/jmathplot
- http://jzy3d.org/
- http://www.jfree.org/

C++

- http://www.qt.io/developers/
- http://www.gnu.org/software/gsl/
- https://en.wikipedia.org/wiki/List_of_numerical_libraries#C.2B.2B
- http://arma.sourceforge.net/
- http://itpp.sourceforge.net/4.3.1/
- http://www.boost.org/