TP 15

Physique numérique

Extrait du programme officiel

Capacités numériques exigibles
Simuler, à l'aide d'un langage de programmation ou d'un tableur, un processus aléatoire permettant de caractériser la variabilité de la valeur d'une grandeur composée. Simuler, à l'aide d'un langage de programmation ou d'un tableur, un processus aléatoire de variation des valeurs expérimentales de l'une des grandeurs — simulation Monte–Carlo — pour évaluer l'incertitude sur les paramètres du modèle.
Tester, à l'aide d'un langage de programmation, le stigmatisme approché d'une lentille demi–boule pour les rayons proches de l'axe optique.
Mettre en œuvre la méthode d'Euler à l'aide d'un langage de programmation pour simuler la réponse d'un système linéaire du premier ordre à une excitation de forme quelconque. À l'aide d'un langage de programmation, simuler la réponse d'un système linéaire du deuxième ordre à une excitation de forme quelconque. À l'aide d'un langage de programmation, résoudre numériquement une équation différentielle du deuxième ordre non–linéaire et faire apparaître l'effet des termes non–linéaires.
Simuler, à l'aide d'un langage de programmation, l'action d'un filtre d'ordre 1 ou 2 sur un signal périodique dont le spectre est fourni. Mettre en évidence l'influence des caractéristiques du filtre sur l'opération de filtrage.
Exploiter, à l'aide d'un langage de programmation, des données astronomiques ou satellitaires pour tester les deuxième et troisième lois de Kepler.

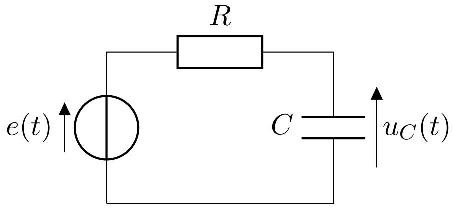
L'objectif des séances de TP liées à ce sujet est d'approfondir les compétences de physique numérique. Les quatre premières compétences ont déjà été mobilisées en TP ou en DM cette année.

1 Simulations en électrocinétique et électronique

1.1 Circuits du premier ordre

1.1.1 Mise en place théorique

On considère le circuit ci–dessous avec $R = 1\,\mathrm{k\Omega}$ et $C = 4\,\mathrm{\mu F}$:



C'est à vous!

1. Déterminer l'équation différentielle vérifiée par la tension u_C aux bornes du condensateur.

1.1.2 Réponse en régime permanent

Il est recommandé de n'utiliser qu'un seul programme pour les parties 1.1.2 et 1.1.3 et de l'augmenter graduellement de façon à ce qu'il puisse répondre à toutes les questions en un seul run.
La qualité de graphiques proposés en sortie de programme sont évidemment un enjeu de taille !

C'est à vous!

2. Écrire un programme Python permettant de déterminer l'évolution de $u_C(t)$ dans le cas d'un échelon de tension (c'est–à–dire $e(t=0^-)=0$ et $e(t=0^+)=E_0$).
3. Modifier ce programme afin de déterminer l'évolution de $u_C(t)$ dans le cas du régime libre (c'est–à–dire $e(t=0^-)=E_0$ et $e(t=0^+)=0$).

1.1.3 Réponse en régime variable à l'aide de la méthode d'Euler

C'est à vous!

4. Déterminer l'évolution de $u_C(t)$ dans le cas d'un signal en créneau d'amplitude E_0 et de période T : Que peut–on dire de la tension $u_C(t)$ en fonction du rapport $\dfrac{T}{RC}$?
5. Déterminer l'évolution de $u_C(t)$ dans le cas d'un signal en triangle d'amplitude E_0 et de période T. Que peut–on dire de la tension $u_C(t)$ en fonction du rapport $\dfrac{T}{RC}$?
6. Déterminer l'évolution de $u_C(t)$ dans le cas d'un signal sinusoïdal d'amplitude E_0 et de période T. Que peut–on dire de la tension $u_C(t)$ en fonction du rapport $\dfrac{T}{RC}$?

1.1.4 Réponse en régime variable à l'aide de la réponse fréquentielle d'un filtre passe–bas

On va maintenant utiliser le fait qu'un circuit RC série est un filtre passe–bas lorsque l'on considère la tension aux bornes du condensateur comme la sortie du filtre.
On utilisera la même méthode quant au développement du programme (on peut utiliser des fonctionnalités interactives si on le souhaite).
La valeur de T pour le signal d'entrée est laissée au choix du programmateur.

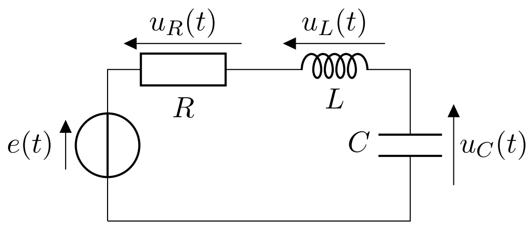
C'est à vous!

7. Écrire la fonction de transfert associée à ce filtre passe–bas.
8. Écrire un programme python permettant de déterminer la fonction `H(f,f0)` avec `f` et `f0` les fréquences du signal et propre du filtre respectivement.
9. À l'aide de cette fonction, tracer le diagramme de Bode en amplitude et en phase de ce filtre.
10. Modifier le programme de façon à de nouveau répondre aux questions 4 à 6.

1.2 Circuits du deuxième ordre

1.2.1 Mise en place théorique

On considère le circuit ci–après avec $R = 1\,\mathrm{k\Omega}$, $L = 30\,\mathrm{mH}$ et $C = 40\,\mathrm{nF}$:



C'est à vous!

11. Déterminer l'équation différentielle vérifiée par la tension u_C aux bornes du condensateur.
12. Déterminer l'équation différentielle vérifiée par la tension u_R aux bornes de la résistance.
13. Déterminer l'équation différentielle vérifiée par la tension u_L aux bornes de la bobine.

1.2.2 Réponse en régime permanent

Il est recommandé de n'utiliser qu'un seul programme pour les parties 1.2.2 et 1.2.3 et de l'augmenter graduellement de façon à ce qu'il puisse répondre à toutes les questions en un seul run.
La qualité de graphiques proposés en sortie de programme sont évidemment un enjeu de taille !

C'est à vous!

14. Écrire un programme Python permettant de déterminer l'évolution de $u_C(t)$, $u_R(t)$ et $u_L(t)$ dans le cas d'un échelon de tension (c'est–à–dire $e(t=0^-)=0$ et $e(t=0^+)=E_0$).
    Comment évoluent les différentes réponse lorsque R varie ?
15. Modifier ce programme afin de déterminer l'évolution de $u_C(t)$ dans le cas du régime libre (c'est–à–dire $e(t=0^-)=E_0$ et $e(t=0^+)=0$).

1.2.3 Réponse en régime variable à l'aide de la méthode d'Euler

C'est à vous!

16. Déterminer l'évolution de u_C, u_R et u_L dans le cas d'un signal en créneau d'amplitude E_0 et de période T. Que peut–on dire des tensions u_C, u_R et u_L lorsque R varie ? Lorsque le rapport $\dfrac{T}{\sqrt{LC}}$ varie ?
17. Déterminer l'évolution de u_C, u_R et u_L dans le cas d'un signal en triangle d'amplitude E_0 et de période T.
    Que peut–on dire des tensions u_C, u_R et u_L lorsque R varie ? Lorsque le rapport $\dfrac{T}{\sqrt{LC}}$ varie ?
18. Déterminer l'évolution de u_C, u_R et u_L dans le cas d'un signal sinusoïdal d'amplitude E_0 et de période T.
    Que peut–on dire des tensions u_C, u_R et u_L lorsque R varie ? Lorsque le rapport $\dfrac{T}{\sqrt{LC}}$ varie ?

1.2.4 Réponse en régime variable à l'aide de la réponse fréquentielle d'un filtre d'ordre 2

On va maintenant utiliser le fait qu'un circuit RLC série se comporte comme un filtre du second ordre (passe–bas, passe–bande ou passe–haut selon le composant aux bornes duquel on mesure la tension de sortie).
On utilisera la même méthode quant au développement du programme (on peut utiliser des fonctionnalités interactives si on le souhaite).

C'est à vous!

19. Écrire les fonctions de transfert des filtres passe–bas, passe–bande et passe–haut d'ordre 2.
20. Écrire un programme python permettant de déterminer les fonctions `H_bas(f,f0,Q)`, `H_bande(f,f0,Q)` et `H_haut(f,f0,Q)` avec `f` et `f0` les fréquences du signal et propre du filtre respectivement et `Q` le facteur de qualité.
21. À l'aide de ces fonctions, tracer les diagrammes de Bode en amplitude et en phase de ces filtres.
22. Modifier le programme de façon à de nouveau répondre aux questions 16 à 18.

2 Simulations en mécanique : Pendule simple

C'est à vous!

23. Établir l'équation différentielle vérifiée par l'angle $\theta(t)$, fait par le fil maintenant le pendule avec la verticale, sans faire l'approximation des petits angles.
24. Écrire un programme python permettant de tracer l'évolution de $\theta(t)$ pour plusieurs valeurs de $\theta_0 = \theta(t=0)$ en choisissant $\dot{\theta}(t=0)) = 0$.
25. Comparer les résultats obtenus avec la formule de Borda.

Annexe 1 — Résolution d'équations différentielles d'ordre 2

La méthode d'Euler explicite, comme toutes les méthodes de résolution numérique d'équations différentielles, s'applique aux équations différentielles d'ordre 1.
Pour discuter les équations différentielles d'ordre plus élevé, il faut utiliser la méthode de réduction de l'ordre d'une équation différentielle, qui consiste à transformer l'équation différentielle scalaire en équation différentielle vectorielle.
Prenons par exemple l'équation différentielle du second ordre :

$$\ddot{x}(t) + \frac{\kappa}{m}(\dot{x}(t))^2 + \omega_0^2 x(t) = K\sin(\omega t). \tag{1}$$

Posons le vecteur $X(t) = (x(t), v(t))$ avec $v(t) = \dot{x}(t)$. On constate que la dérivée première de ce vecteur s'écrit $\dot{X}(t) = (\dot{x}(t), \dot{v}(t)) = (v(t), \ddot{x}(t))$.
Posons alors la fonction vectorielle F, qui prend le temps t ainsi qu'un vecteur de deux dimensions x et v en paramètres et qui renvoie un vecteur à deux dimensions, telle que :

$$F\left(X, t\right) = F\left(\begin{pmatrix} x \\ v \end{pmatrix}, t\right) = \begin{pmatrix} v \\ -\frac{\kappa}{m}v^2 - \omega_0^2 x + K\sin(\omega t) \end{pmatrix}.$$

On constate alors que $\dot{X}(t) = F\left(X, t\right)$ est équivalent à l'équation différentielle (1). Mais cette nouvelle équation est une équation différentielle du premier ordre vectorielle. Le schéma numérique de résolution s'applique à cette équation. La différence est qu'au lieu de manipuler uniquement y_i à chaque étape, il est maintenant nécessaire de manipuler un couple de valeur (x_i, v_i) qui varient selon le schéma numérique à chaque étape.

Pour résoudre l'équation $\dot{y}(t) = f(y, t)$ sur l'intervalle $[t_0, t_f]$, il existe une fonction prédéfinie dans la bibliothèque scipy.integrate dénommée odeint. L'instruction permettant d'utiliser cette fonction a la structure suivante :

odeint(f, liste_y0, liste_t)

avec

- f est une fonction ayant pour paramètre (dans l'ordre) liste_y et liste_t qui sont des *numpy.array* (liste_y est crée par odeint, liste_t est défini par l'utilisateur) ;
- liste_t est un numpy.array défini préalablement et contenant les bornes t_0 et t_f ;
- liste_y0 est un numpy.array contenant les conditions initiales $[y_0, (y_0', ...)]$ correspondant à la première valeur de liste_t.

L'objet liste_y retourné par la fonction odeint est un numpy.array de nombre de lignes égal à liste_t et de nombre de colonnes égal à len(liste_y0) contenant la liste des valeurs du vecteur y (y',...) pour chaque valeur t du tableau liste_t.

Annexe 2 — Décomposition de Fourier de signaux périodiques

Signal carré

Un signal carré de fréquence f et d'amplitude E_0 se décompose comme suit :

$$e(t) = \frac{4E_0}{\pi} \sum_{n=0}^{\infty} \frac{\sin(2\pi(2n+1)ft)}{2n+1}$$

Signal triangulaire

Un signal carré de fréquence f et d'amplitude E_0 se décompose comme suit :

$$e(t) = \frac{8E_0}{\pi^2} \sum_{n=0}^{\infty} \frac{\sin(2\pi(2n+1)ft)}{(2n+1)^2}$$

Annexe 3 — Formule de Borda

En prenant en compte l'effet des non linéarités à l'ordre le plus bas dans l'équation du mouvement, Il est possible de montrer que la période des oscillations T devient dépendante de l'amplitude des oscillations du pendule θ_0. Cette effet est décrit par la formule de Borda :

$$\frac{T}{T_0} = 1 + \frac{1}{16}\theta_0^2 + \frac{11}{3072}\theta_0^4 \ .$$