DÉCOUVRIR LES LOIS DE KEPLER AVEC LE LANGAGE DE PROGRAMMATION PYTHON

Thomas Appéré, Enseignant agrégé de physique-chimie et d'informatique au lycée St-Paul (Vannes) et docteur en planétologie

Un exemple clair et bien détaillé de manipulations informatiques réalisables avec des élèves de Terminale pour étudier les trois lois de Kepler à l'aide du langage de programmation Python.

L'idée de s'appuyer sur la programmation et le codage dans l'enseignement de la physique-chimie remonte à plusieurs années, on trouve des propositions en ce sens dès 1968 [1]. C'est à partir des années 1990 que le numérique a trouvé sa place en lycée grâce à l'équipement à grande échelle en moyens informatiques. Les enseignants de physique-chimie ont mis en application les outils numériques dans deux domaines : l'acquisition et le traitement de données expérimentales d'une part, et la simulation de phénomènes et de lois physiques et chimiques d'autre part. Dans les deux cas, les élèves manipulent des logiciels dédiés dans lesquels les possibilités de programmation sont restreintes voire absentes. Grâce aux logiciels de modélisation d'un phénomène, les élèves observent les effets induits par la variation d'un ou plusieurs paramètres.

Des études ont montré que ce type d'« expérimentation numérique » améliore les apprentissages [2]. Cependant, ces logiciels dédiés peuvent apparaître comme des « boîtes noires » dont le principe de fonctionnement est inconnu. L'apport didactique est indubitablement renforcé si l'élève peut appréhender le principe de l'algorithme utilisé, les lois physiques ou chimiques mises en jeu dans celui-ci ou les hypothèses effectuées dans la modélisation. La programmation de la simulation par l'élève, en totalité ou en partie, peut ainsi contribuer à améliorer les apprentissages [3].

Avec la réforme du lycée en 2019, la programmation et le codage ont fait leur entrée en physique-chimie. Notre discipline est un terrain privilégié de contextualisation pour les mathématiques et l'informatique. Plusieurs notions des programmes de la seconde à la terminale font ainsi explicitement le lien avec des capacités numériques associées à l'utilisation d'un langage de programmation. Cela consiste généralement à adapter et compléter un code existant pour, par exemple, représenter des vecteurs vitesse d'un système, simuler la propagation d'une onde périodique, calculer le taux d'avancement final d'une transformation, tracer le diagramme de distribution des espèces d'un couple acide-base, etc.

Dans le programme de la spécialité physique-chimie de terminale, l'une de ces capacités numériques entre dans le chapitre associé au mouvement dans un champ de gravitation. Elle consiste à « exploiter, à l'aide d'un langage de programmation, des données astronomiques ou satellitaires pour tester les deuxième et troisième lois de Kepler. » [4] Je propose ici une activité répondant à cet objectif, mise en œuvre deux années de suite avec mes groupes-classes de terminale. Le langage de programmation utilisé est Python, comme préconisé dans le Bulletin officiel [4]. Les ressources liées à cette activité (fiche élève et sa correction, fichiers de données astronomiques au format CSV, programme Python élève et sa correction) sont à retrouver sur le site web du CLEA.

Présentation de l'activité

Cette activité traite des trois lois de Kepler. Elle consiste (i) à décrire le contenu de fichiers contenant des données astronomiques, (ii) à compléter plusieurs parties d'un programme Python et (iii) à exécuter ce programme pour afficher les orbites de plusieurs planètes et d'une comète, vérifier la loi des aires et la troisième loi de Kepler. Cette activité est généralement menée à son terme par les élèves en une séance de 2 heures.

L'idéal est que chaque élève dispose d'un poste informatique sur lequel est installé un environnement de développement (IDE) Python, tel que EduPython ou Pyzo (logiciels gratuits). Ce n'est cependant pas toujours possible. Lorsque l'activité est mise en œuvre avec un ordinateur par binôme ou trinôme, je conseille aux élèves de se relayer pour que chacun se confronte à l'écriture dans un langage de programmation. Certains élèves ont suivi la spécialité Numérique et sciences informatiques (NSI) en classe de première voire la suivent toujours en terminale. Ils sont plus à l'aise avec les notions d'algorithmique et de programmation et il est préférable de répartir ces élèves dans les différents binômes ou trinômes afin qu'ils aident leurs camarades.

Données astronomiques utilisées

J'ai choisi de faire travailler les élèves sur les orbites des quatre planètes telluriques et de la comète Encke. Pourquoi cette comète plutôt qu'une autre plus connue comme la comète de Halley ? Parce que son aphélie, point de sa trajectoire le plus éloigné du Soleil, est à seulement 4 unités astronomiques. Ainsi, il est possible de représenter sur un même graphique, de façon lisible, la totalité de l'orbite de la comète Encke et des quatre planètes telluriques.

Fig.1. **Formulaire de l'IMCCE pour calculer les éphémérides d'un corps céleste.**

Les données astronomiques (éphémérides) ont été téléchargées sur le site de l'Institut de mécanique céleste et de calcul des éphémérides (IMCCE), sur le portail Miriade dédié aux éphémérides des corps du Système solaire[1].

Les dates sélectionnées pour ces éphémérides, en UTC, couvrent une orbite avec un pas de temps d'un jour, le nombre de données est donc plus important pour la planète Mars que pour Mercure. Dans le formulaire (voir figure 1), on renseigne comme plan de référence l'écliptique et on demande des coordonnées rectangulaires. Les données générées, au format texte, ont été enregistrées au format CSV en choisissant le séparateur approprié.

Il est possible de demander aux élèves de générer eux-mêmes ces éphémérides à partir du site de l'IMCCE, en leur indiquant la date de début et le nombre de dates nécessaires pour chaque corps céleste. Cependant, cela demande du temps qui ne pourra pas être consacré à la suite de l'activité et l'opération est assez répétitive. J'ai fait le choix de leur projeter au tableau la page d'accueil du portail des éphémérides en leur présentant la démarche suivie pour la planète Mercure et le tableau de données obtenu. On pourra le cas échéant faire générer par chaque groupe les éphémérides d'un corps céleste différent, dont ils mettront le fichier de données astronomiques sur un espace commun.

Première loi de Kepler

Dans un premier temps, les élèves s'approprient le contenu des fichiers de données astronomiques. Grâce au logiciel Excel ou Bloc-notes, ils ouvrent l'un des fichiers CSV, par exemple Mercure.csv et indiquent à quoi correspond chacune des colonnes. La colonne «Date» est facile à interpréter ; on pourra leur faire remarquer que la date est donnée dans le système anglo-saxon, avec le mois indiqué avant le jour. Viennent ensuite les trois coordonnées spatiales (X ; Y ; Z) dans le système héliocentrique. On pourra leur demander ce que signifie le sigle «au» (astronomical unit) et de justifier le choix de cette unité de mesure. Les élèves ont généralement plus de difficulté à identifier la signification des trois coordonnées suivantes (Xp ; Yp ; Zp). Je les guide vers l'unité de ces coordonnées : unité astronomique par jour, ce qui correspond à l'unité d'une vitesse. Ce sont donc les coordonnées de la vitesse selon les trois axes du repère.

On demande ensuite aux élèves de lancer l'IDE Python installé sur leur ordinateur, d'ouvrir le fichier Python Kepler_ELEVE.py et de l'exécuter. Les orbites de Mercure, Vénus et la Terre s'affichent à l'écran. On invite les élèves à comparer les orbites de Mercure et de Vénus. On s'attend à ce qu'ils remarquent que leurs orbites semblent circulaires, celle de Mercure étant décentrée par rapport au Soleil. En effet, la notion d'ellipse n'a pas encore été abordée.

La première partie du programme informatique contient les lignes de code permettant de lire les données relatives à Mercure, Vénus et la Terre et d'afficher leur orbite. Par analogie, on demande aux élèves de compléter le programme pour qu'après exécution, il affiche les orbites de Mars et de la comète Encke. Les données astronomiques sont stockées dans des « tuples », qui sont des tableaux de données non modifiables. Dans le cas du tuple associé à Mercure, on accède à la première colonne du tableau en tapant `mercure[0]`, à la deuxième colonne en tapant `mercure[1]`, etc. Puis on affecte les données de la première colonne à une variable appelée `X_mercure` et ainsi de suite. Les élèves n'ont pas à savoir ce qu'est un tuple pour accomplir la tâche demandée. Il suffit de copier les lignes correspondant à la lecture des données relatives par exemple à la Terre, de les coller à la suite puis

1 http://vo.imcce.fr/webservices/miriade/?forms
Il existe depuis peu un nouveau portail plus convivial et en français https://ssp.imcce.fr/

de remplacer le mot `terre` par `mars` pour que le programme fonctionne. Ensuite, il s'agit de copier la ligne permettant de tracer l'orbite de la Terre, de la coller à la suite, de remplacer `terre` par `mars` et de choisir une couleur différente du bleu pour le tracé de l'orbite (par exemple rouge). Les élèves suivent le même cheminement pour tracer l'orbite de la comète Encke.

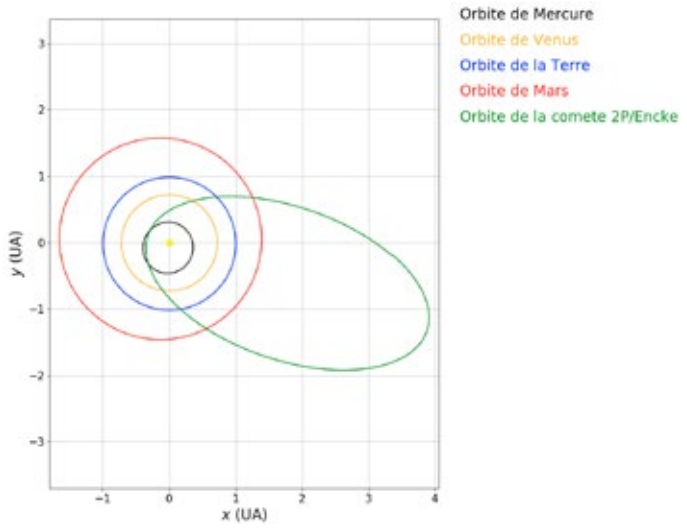

Fig.2. **Tracés obtenus à l'issue de la partie 1.**

Cette partie ne pose généralement pas de problème et les élèves sont satisfaits de voir s'afficher les orbites demandées. Certains personnalisent la couleur des orbites. Il arrive que les élèves ne parviennent pas à faire fonctionner leur programme du premier coup car l'orthographe est primordiale dans un langage de programmation comme Python. Ainsi, `X_terre` et `X_Terre` désignent deux variables différentes.

Comment qualifier l'orbite de la comète Encke ? C'est la question qu'on pose ensuite aux élèves. Certains connaissent la notion d'ellipse, on la prolonge en précisant ce que sont les foyers d'une ellipse. On conclut cette première partie en construisant avec les élèves la définition de la première loi de Kepler.

Deuxième loi de Kepler

Dans cette deuxième partie de l'activité, l'objectif est de comparer l'aire balayée par le segment reliant les centres de masse du Soleil et d'un astre (planète ou comète) pendant des durées égales. J'ai choisi deux points particuliers d'une orbite, le périhélie et l'aphélie, et j'ai écrit le code permettant de calculer l'aire parcourue par le rayon vecteur pendant une durée déterminée au voisinage du périhélie et de l'aphélie. C'est la fonction `calcul_aire()` qui se charge de ce calcul. Cependant, cette fonction prend entre autres paramètres d'entrée la distance séparant le Soleil et l'astre ainsi que sa vitesse orbitale. L'évolution de ces deux grandeurs au cours du temps est stockée dans deux listes, `distance_au_Soleil` et `vitesse`. C'est aux élèves de compléter les lignes de code pour calculer ces deux grandeurs et les stocker dans les listes appropriées. La position des lignes de code à compléter est indiquée en annexe de l'activité.

Il s'agit tout d'abord d'exprimer la distance au Soleil de l'astre en fonction des coordonnées spatiales de sa position :

$$\text{distance} = \sqrt{x^2 + y^2 + z^2}$$

De même, les élèves ont à trouver l'expression de la vitesse de l'astre :

$$\text{vitesse} = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

Ces expressions mathématiques sont à traduire en langage de programmation Python. Les traductions de la racine carrée, `np.sqrt()`, et de l'exposant, `nombre**exposant`, sont données dans le document. La difficulté réside surtout dans la manière de faire référence aux coordonnées spatiales de la position et de la vitesse de l'astre. En effet, ces données sont stockées sous forme de listes. Par ailleurs, les calculs de la distance au Soleil de l'astre et de sa vitesse se font dans une boucle *Pour*. Il convient d'expliquer aux élèves le fonctionnement de cette structure itérative, si cela n'a pas été fait dans une précédente activité numérique. Ici, la boucle incrémente sur l'indice i de 0 à (longueur de la liste X) - 1, par pas de 1. Cet indice i désigne en fait le jour pour lequel les coordonnées de la position et de la vitesse de l'astre ont été calculées dans les éphémérides. On accède au i-ème élément de la liste X en tapant `X[i]`. De plus, la fonction `append()` permet d'ajouter un élément (valeur numérique, chaîne de caractères...) à une liste, ici `distance_au_Soleil`

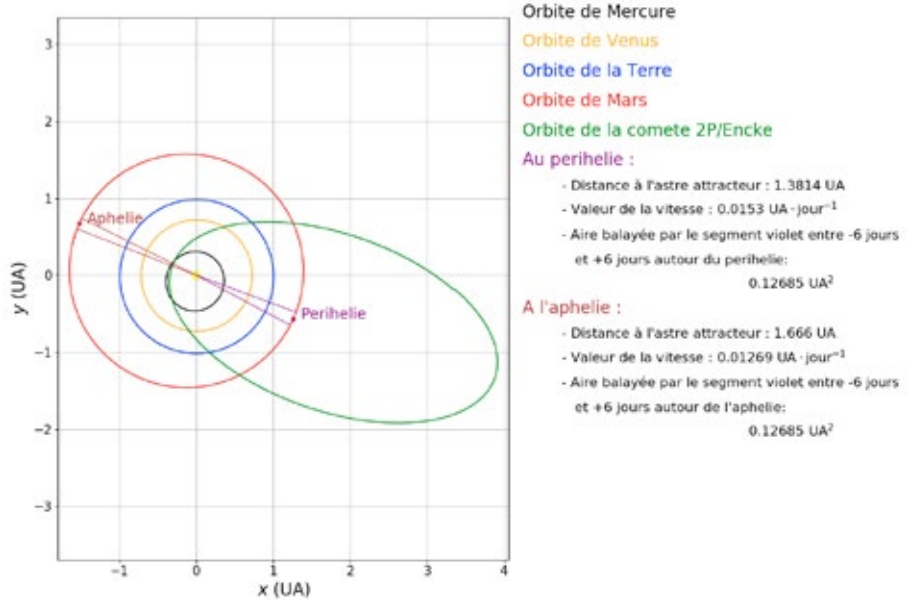

Fig.3. **Tracés obtenus dans la partie 2 pour Mars.**

et `vitesse`. Ces informations sont généralement suffisantes pour permettre aux élèves de compléter les deux boucles *Pour*.

Une fois le code complété, les élèves exécutent le programme et voient s'afficher la position du périhélie et de l'aphélie de Mars. On leur demande de définir ces deux positions en justifiant leur réponse à l'aide du code. Il s'agit ici de repérer dans le code que les variables `perihelie` et `aphelie` sont respectivement égales à `min(distance_au_Soleil)` et `max(distance_au_Soleil)`.

Ensuite, les élèves comparent les aires balayées par le segment de droite reliant le Soleil et la planète Mars au périhélie et à l'aphélie en des durées égales. Ils remarquent qu'elles sont égales. On pourra suggérer aux élèves de modifier cette durée pour vérifier que l'égalité des aires balayées est toujours valable.

De cette information, les élèves en déduisent que la vitesse au périhélie est maximale et qu'elle est minimale à l'aphélie. Enfin, ils modifient la valeur de la variable astre pour lui donner la valeur «`comete`», afin de vérifier que l'égalité des aires balayées est également valable dans le cas de l'orbite de la comète Encke.

Pour finir cette deuxième partie, on co-construit avec les élèves la définition de la deuxième loi de Kepler.

Troisième loi de Kepler

Dans cette dernière partie de l'activité, trois lignes du code sont à compléter pour :

1. Calculer la valeur de la variable `a`, demi-grand axe de l'orbite, en mètres.
2. Calculer la valeur de la période de révolution `T` de l'astre, en secondes.
3. Calculer la valeur du rapport $\frac{T^2}{a^3}$ et l'afficher.

Dans le cas du premier point, il s'agit de repérer que le calcul du demi-grand axe `aUA` est effectué à la ligne précédente, en unités astronomiques. Il faut donc convertir les unités astronomiques en mètres.
La période est quant à elle déjà calculée en jours et nommée `Tj`, il faut la convertir en secondes.
Enfin, le calcul du rapport $\frac{T^2}{a^3}$ se fait en traduisant l'expression en langage Python : `T**2/a**3`.

Les tâches à réaliser par les élèves dans cette partie consistent donc à interpréter un code existant pour le compléter, sans nécessairement avoir à comprendre l'intégralité des fonctions et structures itératives et conditionnelles utilisées.

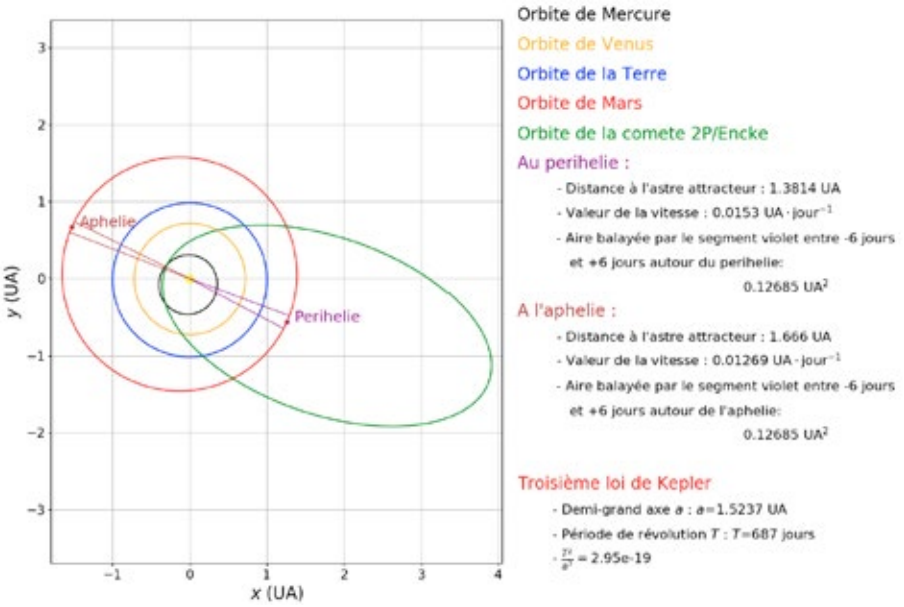

Fig.4. **Tracés obtenus dans la partie 3 pour Mars.**

Après exécution du programme, les élèves comparent la valeur du rapport $\frac{T^2}{a^3}$ pour la planète Mars et la comète Encke. Ils constatent que dans les deux cas, le rapport est égal à quelques décimales près.
Puis ils calculent le rapport $\frac{4\pi^2}{GM_S}$ et remarquent que sa valeur est égale au rapport . Ainsi ils peuvent donner la définition de la troisième loi de Kepler et conclure ainsi cette activité numérique.

De manière générale, cette activité n'a pas posé trop de difficultés aux élèves les deux années où ils l'ont réalisée. L'ayant menée au mois de novembre, j'ai pu traiter au préalable trois activités numériques sur trois autres chapitres permettant aux élèves de se familiariser avec le langage Python.

On peut envisager d'améliorer le programme Python en ajoutant un aspect dynamique, sous la forme de disques représentant les planètes et la comète et qui se déplaceraient sur leur orbite respective. Cela permettrait de mettre en évidence que la période de révolution d'un astre augmente lorsque le demi-grand axe de son orbite augmente. Une telle animation montrerait également que la vitesse orbitale d'un astre diminue lorsque le demi-grand axe de son orbite augmente.

Références bibliographiques

[1] *Computers in High School Physics*, A.M. Borl, The physics teacher **6**, 296 (1968).
[2] *The learning effects of computer simulations in science education*, N. Rutten, W.R. van Joolingen, J.T. van der Veen, Computers & Education **58**, 136-153 (2012).
[3] *Programmer en physique-chimie*, eduscol.education.fr, Ministère de l'Éducation nationale et de la Jeunesse (2018).
[4] *Physique-Chimie, classe terminale, enseignement de spécialité, voie générale*, BO spécial n°1 du 22 janvier 2019.

■