

## Matlab 11-12 for dummies : exercice 6

### Méthode du tir

On vous demande d'utiliser la méthode de bisection pour trouver la norme de la vitesse initiale d'un obus pour atteindre une cible à la même altitude située à une distance de  $\alpha$  mètres. Le tube du canon est orienté à 45 degrés et la masse de l'obus en kg est donné par  $M$ , tandis que  $g = 9.81 \text{ m/s}^2$ . Les forces agissant sur l'obus sont la gravité et la force de trainée due au frottement de l'air sur l'obus. La position horizontale et verticale  $(x(t), y(t))$  de l'obus en fonction du temps sont donc solutions du système suivant :

$$\begin{cases} Mx''(t) &= -C(t) x'(t) \\ My''(t) &= -Mg - C(t) y'(t) \end{cases}$$

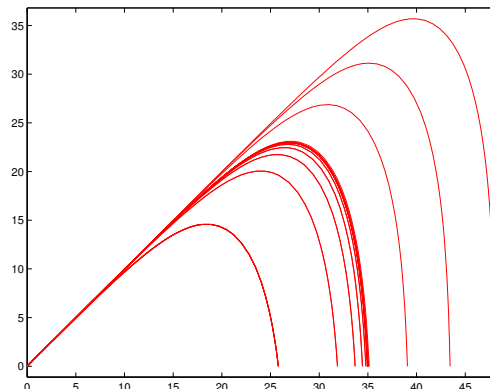
avec  $C(t)$  égal au dixième de la norme de la vitesse à l'instant  $t$ . En d'autres mots<sup>1</sup> cela revient à écrire :

$$C(t) = \frac{1}{10} \sqrt{(x'(t))^2 + (y'(t))^2}$$

Vous disposez d'un programme calculant la solution d'un problème non-linéaire avec la méthode de bisection. Il s'agit de modifier ce programme pour obtenir la norme de la vitesse initiale de l'obus.

Maintenant, on va utiliser les fonctions de la toolbox de MATLAB. En particulier, on tirera profit des fonctions `ode45` et `events` afin d'effectuer l'intégration du mouvement et d'obtenir le point et l'instant de l'impact. Plus précisément, on vous demande de :

1. Ecrire une fonction `[tf, v0,n] = shoot(alpha,M)` qui fournit le temps nécessaire `tf` pour atteindre la cible, ainsi que la norme de la vitesse initiale `v0`. On utilisera `ode45` avec la technique de la bisection. Pendant le calcul, on représentera graphiquement toutes les trajectoires calculées sur une unique figure avec une échelle horizontale et verticale identique. La fonction retournera aussi le nombre `n` de tirs requis dans la procédure d'ajustage.
2. Pour entamer la technique de bisection, il faut évidemment commencer par encadrer l'objectif. Définir une stratégie efficace et robuste pour réaliser ce premier encadrement est le second objectif de ce problème. Il n'est pas interdit de proposer une amélioration de la technique de bisection pour améliorer la rapidité de la procédure d'ajustage. L'évaluation se fera sur l'exactitude de votre calcul, mais aussi sur la robustesse et la rapidité de votre algorithme.
3. Votre fonction (avec toutes les éventuelles sous-fonctions que vous auriez créées) sera incluse dans un unique fichier `shoot.m`, sans y adjoindre le programme de test fourni ! Ce fichier devra être rendu via le web avant le mardi 29 novembre à 23h59.



<sup>1</sup> Pour ceux qui n'avaient pas compris :-). Certains tuteurs taquins se reconnaîtront !

## Implémentation de la méthode de bisection

Voici, à toutes fins utiles, l'implémentation de la méthode de bisection vue au dernier cours :-)

```
function bissec(a,b,tol,nmax);
n = 0; delta = tol + 1;
if (f(a)*f(b) > 0)
    error('The sign of the functions at the extrema must be different');
end
while abs(delta) >= tol && n <= nmax
    delta = (b-a)/2;
    x = a + delta;
    n = n + 1;
    if (f(x)*f(a) > 0)
        a = x;
    else
        b = x;
    end
    fprintf('\n x = %21.14e (Estimated error %21.14e at %i iteration ) ',x,abs(delta),n);
end
if (n == nmax)
    error('Aie !');
end
end

function y = f(x)
y = x .* sin(x) - 1;
end
```