Exercice 1 : Série E1 — Freinage d'une voiture avec traînée aérodynamique

(a)

0.0.1 Modèle physique

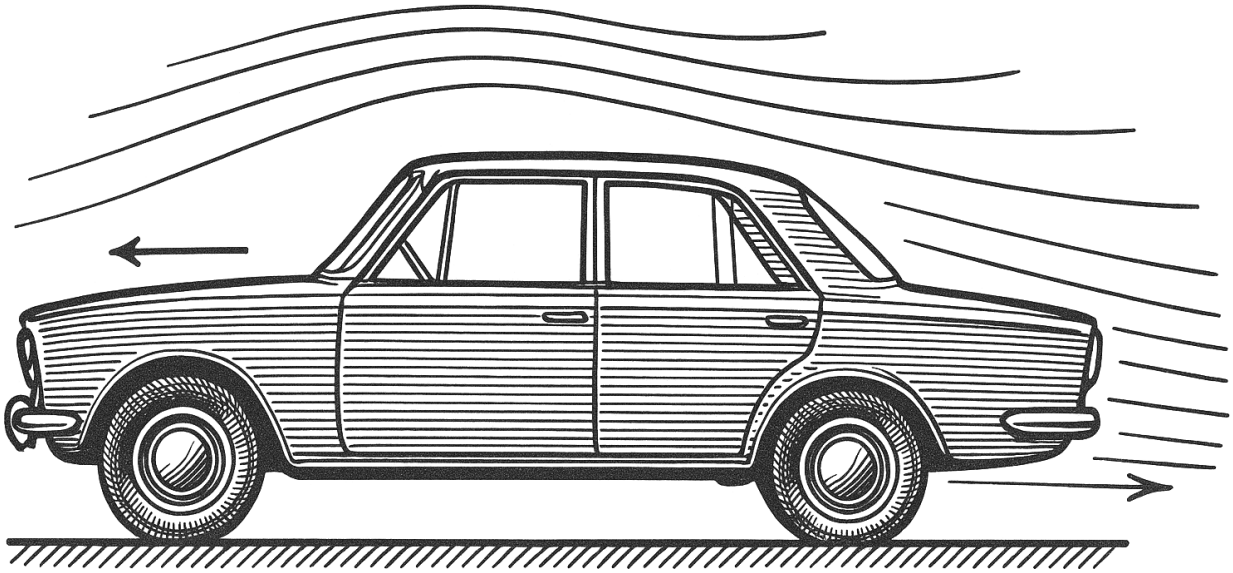

Figure 1 – Voiture

On modélise le freinage d'une voiture par une décélération constante $a_0 = \mu g$ et une force de traînée aérodynamique quadratique $F_D = \frac{1}{2}\rho C_d A v^2$. On montre que la vitesse initiale v_0 mène aux formes fermées suivantes, avec :

$$k = \frac{\rho C_d A}{2m}.$$

Le temps d'arrêt vaut :

$$t_{\text{stop}} = \frac{1}{\sqrt{a_0 k}} \arctan\left(v_0 \sqrt{\tfrac{k}{a_0}}\right)$$

et la distance d'arrêt vaut :

$$x_{\text{stop}} = \frac{1}{2k} \ln\left(1 + \tfrac{k}{a_0} v_0^2\right).$$

Toutes les grandeurs doivent être exprimées en unités SI cohérentes, soit :

- m en kilogrammes (kg),
- A en mètres carrés (m^2),
- ρ en kilogrammes par mètre cube ($\text{kg} \cdot \text{m}^{-3}$),
- g en mètres par seconde carrée ($\text{m} \cdot \text{s}^{-2}$),
- v_0 en mètres par seconde ($\text{m} \cdot \text{s}^{-1}$).

0.0.2 Cahier des charges

Écrire un programme `brake.c` qui lit **sept arguments** en `int` ou `double` depuis la ligne de commande, donnés dans des unités volontairement peu pratiques. On suggère d'utiliser un suffixe dans le nom des variables pour rappeler l'unité, par exemple `v0_kmh` pour la vitesse initiale en km/h. Les arguments sont :

1. `v0_kmh` : vitesse initiale en km/h ;
2. `m_t` : masse du véhicule en tonnes ;
3. `mu_permille` : coeff. de frottement en ‰ ;
4. `A_dm2` : surface frontale en dm^2 ;
5. `Cd` : coefficient de traînée (sans dimension) ;
6. `rho_gL` : densité de l'air en g/L (indice : 1 g/L = 1 $\mathrm{kg/m}^3$) ;
7. `g_Gal` : accélération gravitationnelle en Gal (1 Gal = 0.01 $\mathrm{m/s}^2$).

Le programme doit :

1. convertir **toutes** les grandeurs en unités SI ;
2. calculer successivement a_0, k, t_{stop} et x_{stop} ;
3. afficher les résultats avec trois décimales, en secondes pour le temps et en mètres pour la distance.

0.0.3 Exemple d'exécution

```
$ ./freinage 130 1.4 800 22 0.30 1.225 981
Temps d'arrêt : 4.594 s
Distance d'arrêt : 82.881 m
```

0.0.4 Déroulement conseillé

1. Réfléchir au problème, aux conversions et aux formules.
2. Écrire un squelette de programme avec les inclusions nécessaires et la fonction `main`.
3. Lire les arguments et vérifier le nombre attendu ; quitter proprement sinon.
4. Implémenter les conversions et vérifier les valeurs intermédiaires au besoin.
5. Calculer les grandeurs physiques et valider avec l'exemple fourni.
6. Formater l'affichage des résultats avec trois décimales.

```c
#include <math.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>

static bool is_valid(double v0, double m, double mu,
                     double A, double Cd, double rho, double g) {
    return v0 >= 0 && m > 0 && mu >= 0 && A > 0 && Cd >= 0 && rho > 0 && g > 0;
}

int main(int argc, char *argv[]) {
    if (argc != 8) {
        fprintf(stderr,
                "Usage: %s v0_kmh m_t mu_permille A_dm2 Cd rho_gL g_Gal\n",
                argv[0]);
        return EXIT_FAILURE;
    }

    const double v0_kmh = atof(argv[1]);
    const double m_t = atof(argv[2]);
    const double mu_permille = atof(argv[3]);
    const double A_dm2 = atof(argv[4]);
    const double Cd = atof(argv[5]);
    const double rho_gL = atof(argv[6]);
    const double g_Gal = atof(argv[7]);

    const double v0 = v0_kmh * (1000.0 / 3600.0);
    const double m = m_t * 1000.0;
    const double mu = mu_permille / 1000.0;
    const double A = A_dm2 * 0.01;
    const double rho = rho_gL;
    const double g = g_Gal * 0.01;

    if (!is_valid(v0, m, mu, A, Cd, rho, g)) {
        fputs("Arguments physiques invalides.\n", stderr);
        return EXIT_FAILURE;
    }

    const double a0 = mu * g;
    const double k = (rho * Cd * A) / (2.0 * m);
    const double t_stop = (1.0 / sqrt(a0 * k)) * atan(v0 * sqrt(k / a0));
    const double x_stop = (1.0 / (2.0 * k)) * log(1.0 + (k / a0) * v0 * v0);

    printf("Temps d'arrêt : %.3f s\n", t_stop);
    printf("Distance d'arrêt : %.3f m\n", x_stop);
    return EXIT_SUCCESS;
}
```