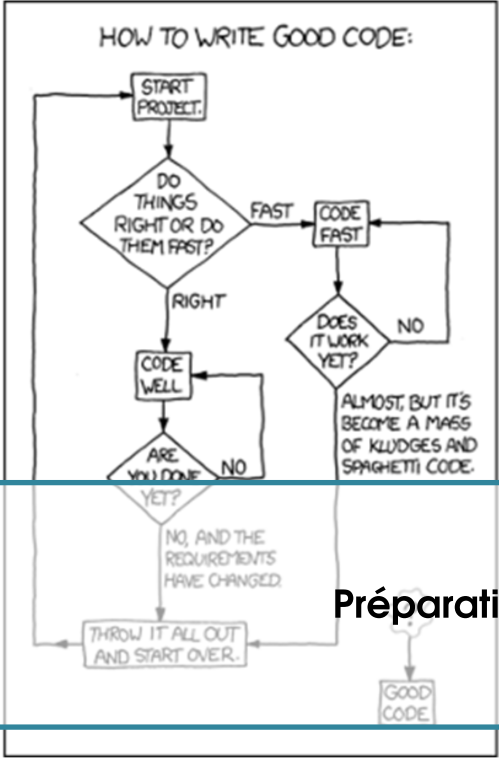
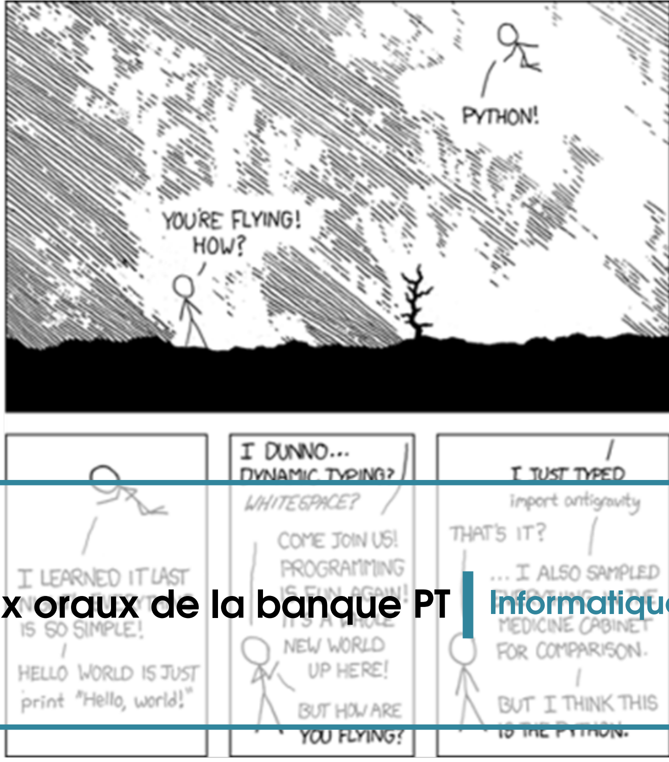



http://xkcd.com/353/
http://xkcd.com/844/

PT – PT ⋆

**Oral**

## Préparation aux oraux de la banque PT | Informatique

# Exercices | Préparation aux oraux de la banque PT
# Épreuve de TP de SII

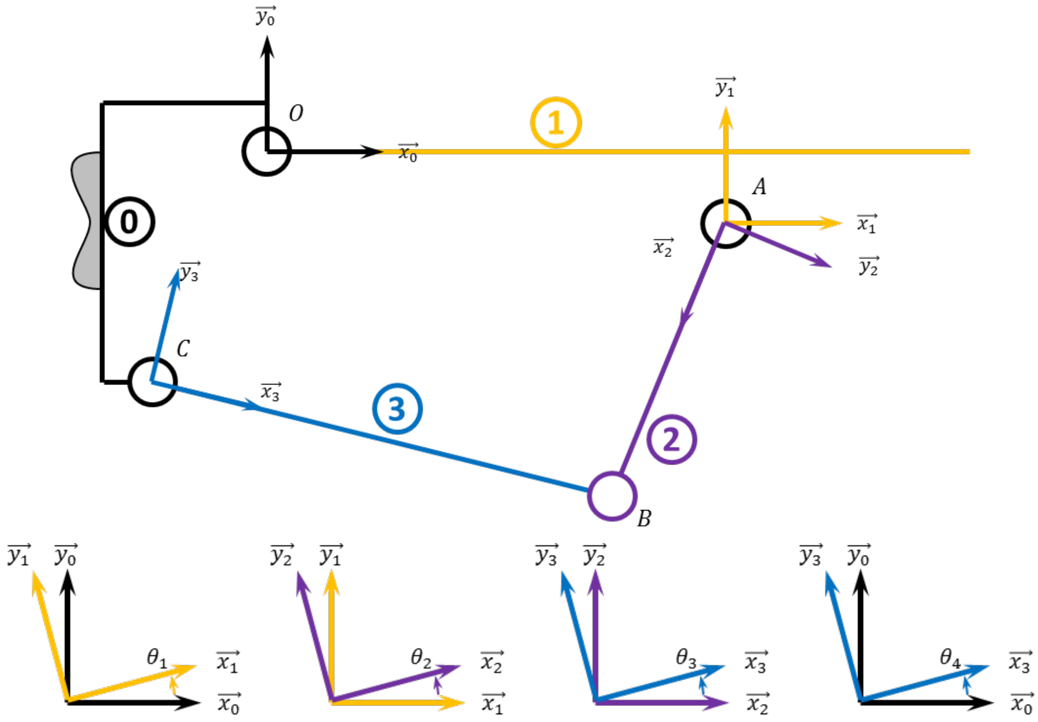

| | | |
|---|---|---|
| 1 | Loi entrée-sortie du portail ABB | 2 |
| 2 | Traitement du signal | 3 |
| 3 | Détermination des paramètres de rugosité | 4 |
| 4 | Mesure au marbre d'un défaut de planéité | 5 |



*Xavier Pessoles*
*PT – PT ⋆*

# 1 Loi entrée-sortie du portail ABB

On donne le schéma cinématique et le paramétrage de la loi Entrée-Sortie du portail ABB.



Un moteur à courant continu commande la liaison pivot entre **0** et **3**.
On a :

- $\overrightarrow{OA} = a\overrightarrow{x_1} - f\overrightarrow{y_1}$ ;
- $\overrightarrow{AB} = b\overrightarrow{x_2}$ ;
- $\overrightarrow{BC} = -c\overrightarrow{x_3}$ ;
- $\overrightarrow{OC} = -d\overrightarrow{x_0} - e\overrightarrow{y_0}$ ;

Les équations régissant la loi entre l'angle d'entrée $\theta_4$ et l'angle de sortie $\theta_1$ sont donnés par les équations suivantes :

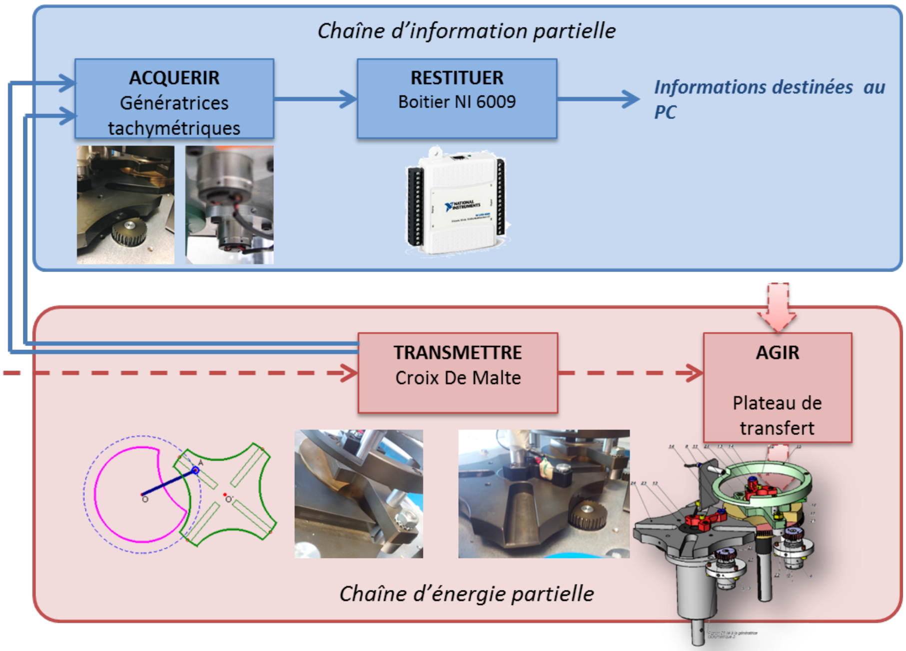
$$\left\{\begin{array}{l} a\cos\theta_1 + b\cos\theta_2\cos\theta_1 - b\sin\theta_2\sin\theta_1 - c\cos\theta_4 + d + f\sin\theta_1 = 0 \\ a\sin\theta_1 + b\cos\theta_2\sin\theta_1 + b\sin\theta_2\cos\theta_1 - c\sin\theta_4 + e - f\cos\theta_1 = 0 \\ \theta_1 + \theta_2 + \theta_3 - \theta_4 = 0 \end{array}\right.$$

On donne le programme `01_LoisESABB.py` permettant de tracer $\theta_2$, $\theta_3$ et $\theta_4$ en fonction de $\theta_1$.

1. Exécuter le programme. Quelle phase de vie du portail est simulée ?
2. Quel est le domaine de variation angulaire du moteur pour un cycle d'ouverture ou de fermeture ?
3. Modifier le programme pour avoir une plus grande précision angulaire sur les différentes positions.
4. Expliquer comment a été rédigée la fonction `systeme`. Vous préciserez la nature et le type des données d'entrées et de sorties.
5. Quel est l'objectif de la fonction `resoudre` ? Commenter les différents blocs d'instruction. Vous préciserez en particulier le rôle de la fonction `fsolve` et vous justifierez le rôle de la variable `sol_ini`.
6. Le fichier `mesure.txt` situé dans le répertoire `data` contient les mesures des angles $\theta_1$ et $\theta_4$ en fonction du temps. Afficher $\theta_4$ en fonction de $\theta_1$ (angles mesurés). (Dans le fichier texte, la première colonne correspond au temps, la seconde à $\theta_4$ et la dernière à $\theta_1$.)
7. Afficher sur un même graphe la courbe mesurée et la courbe simulée.
8. On souhaite que la vitesse du portail soit de $6,59°.\text{min}^{-1}$. On néglige le temps d'accélération. Modifier le programme en conséquence et afficher $\theta_1$ et $\theta_4$ en fonction du temps.
9. Tracer $\dot{\theta}_4(t)$ en fonction du temps.
10. Tracer $\dot{\theta}_4(t)$ pour les courbes mesurées et simulées.

## 2 Traitement du signal

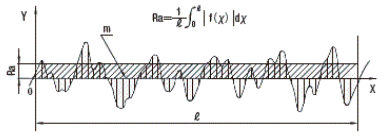
La vitesse du maneton et de la croix Malte de la capsuleuse est mesurée grâce à deux génératrices tachymétriques.




Le fichier `mesure.txt` contenu dans le dossier `data` contient les mesures réalisées par chacune des deux génératrices.

1. Ouvrir le fichier `02_FiltrageCapsuleuse.py` et l'exécuter. Commenter l'allure des courbes.
2. Déterminer la fréquence d'échantillonnage du signal mesuré.
3. Commenter les 3 fonctions `lireFichierMesure`, `filtrageMoyenne` et `filtrageMoyenneGlissante`. Pour cela vous préciserez la nature des variables d'entrée et de sortie. Vous commenterez ensuite les blocs d'instruction.
4. Modifier les valeurs du filtre et observer leur effet. Quelles valeurs permettent d'avoir une courbe lissée.
5. On souhaite filtrer le signal grâce à un filtre passe-bas. L'équation différentielle est la suivante : $s(t)+\tau\dfrac{\mathrm{d}s(t)}{\mathrm{d}t}=e(t)$. $e(t)$ désigne le signal non filtré et $s(t)$ désigne le signal filtré. Exprimer la suite $s_k$ correspondant au signal filtré en fonction du pas de calcul $k$, de $\tau$ et des échantillons $s_k$ et $e_k$ nécessaires.
6. Implémenter la fonction `filtrePB` prenant comme arguments la liste des échantillons $e_k$, le pas de calcul $p$ et la pulsation de coupure du filtre $1/\tau$.
7. Tracer le signal non filtré ainsi que le signal filtré avec différentes pulsations de coupure.

## 3 Détermination des paramètres de rugosité




Un rugosimètre est un appareil permettant de mesurer une le profil d'une ligne sur une surface fabriquée. On peut donc mesurer une altitude $(y)$ en fonction d'une position longitudinale $(x)$. Une fois le profil mesuré, il est possible de calculer un certain nombre de paramètres sur le profil mesuré :

$$R_y = y_{\max} - y_{\min} \quad Ra = \frac{1}{l}\int_0^l |f(x)|\mathrm{d}x.$$

1. Exécuter le programme et identifier les différents profils.
2. Analyser l'influence de `tau` sur les profils.
3. Commenter les programmes `filtrePB`, `calculAire` et `rechercheh`. Vous expliciterez quelles sont les données d'entrées et de sorties de ces programmes et vous expliquerez quelles sont les méthodes numériques utilisées.
4. Déterminer le paramètre `tau` qui permet de déterminer la porteuse du profil.
5. Calculer le paramètre $R_y$.
6. Calculer le paramètre $R_a$.

## 4 Mesure au marbre d'un défaut de planéité

Le bras de mesure est utilisé en métrologie afin de contrôler les spécifications inscrites sur le dessin de définition d'une pièce fabriquée. Nous allons nous intéresser ici à l'algorithme de calcul du défaut de planéité à partir d'un nuage de points mesurés à l'aide du palpeur du bras. Les mesures sont réalisées dans un repère machine R(O,x,y z). Les coordonnées des points sont stockées dans un fichier `plan.csv` sous forme de tableau à trois colonnes x, y et z (contient 350 points).

Le défaut de planéité est défini comme étant la distance $d$ la plus petite entre deux plans parallèles $P_1$ et $P_2$ tels que tous les points mesurés soient positionnés entre ces deux plans.

En pratique, on associe aux nuages de points $M_i$ un plan $\Pi$. Ce plan $\Pi$ d'équation $ax+by+cz+d=0$ est construit suivant le critère des moindres carrés :

$$S=\sum_{i=0}^{n-1}(e_i)^2 \text{ doit être minimale}$$

Les distances $e_i$ sont définies comme les distances des points $M_i$ au plan $\Pi$.

Pour $c \neq 0$, l'équation du plan devient $Ax+By+z+D=0$. On rappelle que si une fonction admet un extremum en un point alors toutes ses dérivées partielles seront forcément nulles en ce point. On va donc calculer les trois dérivées partielles de $S$ : $S'(A)$, $S'(B)$ et $S'(D)$ et définir $A$, $B$, $D$ pour :

$$\frac{\partial S}{\partial A}=0, \frac{\partial S}{\partial B}=0, \frac{\partial S}{\partial D}=0$$

L'équation du plan $\Pi$ obtenue, le défaut de planéité calculé est $d$ la somme entre la distance du point $M_i$ le plus éloigné au dessus du plan et la distance du point $M_j$ le plus éloigné en dessous du plan.

1. A partir du fichier `plan.csv` fourni, créer la liste des listes de coordonnées des points mesurées. On appellera cette liste `pointMesure`.
2. Montrer que la minimalisation de $S$ permet de définir 3 équations faisant intervenir les paramètres $(A,B,D)$ du plan. Mettre ce système sous forme matricielle.

$$\underbrace{\begin{bmatrix} \quad & \quad & \quad \\ & & \\ & & \end{bmatrix}}_{M}\underbrace{\begin{bmatrix} A \\ B \\ D \end{bmatrix}}_{P}=\underbrace{\begin{bmatrix} \quad \\ \\ \end{bmatrix}}_{V}$$

3. Executer le programme `planMoindresCarres`. Analyser le résultat affiché.
4. Commenter les étapes de la fonction `vecteurP` en vous appuyant sur le système d'équations obtenu à la question 2.
   Pour la résolution du système linéaire, on précise les méthodes de numpy :
   - np.linalg.inv(M) pour définir la matrice inverse de M
   - np.dot(M,V) pour le produit de deux matrices
5. Définir la fonction `defautPlaneite` qui a pour argument un vecteur et une liste de listes des coordonnées de points et renvoie la valeur calculée du défaut de planéité. Afficher à l'écran cette valeur.