teaching sciences for innovation

Proposition de corrigé du sujet de concours

Banque PT

Modélisation - Informatique

de l'année *2016*

Ceci est une proposition de corrigé des concours de CPGE, réalisée bénévolement par des enseignants de Sciences Industrielles pour l'Ingénieur et d'Informatique, membres de l'UPSTI (Union des Professeurs de Sciences et Techniques Industrielles), et publiée sur le site de l'UPSTI : www.upsti.fr

Ce document vous apportera des éléments de corrections pour le sujet traité, mais n'est pas un corrigé officiel du concours, ni un corrigé exhaustif très détaillé relatif au problème posé.

L'UPSTI ne répondra pas directement aux questions que peuvent soulever ces corrigés : nous vous invitons à voir avec vos enseignants si vous souhaitez des compléments d'information, et à passer par eux pour nous faire remonter des éventuels problèmes.

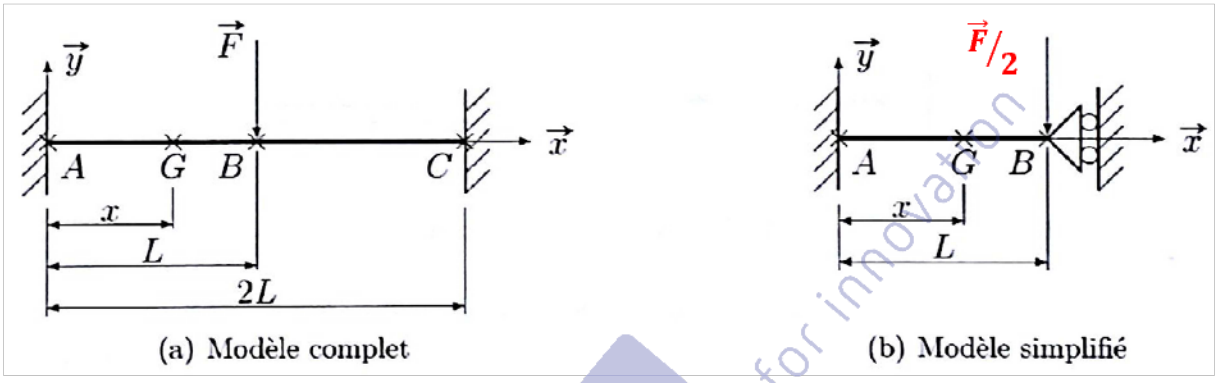
L'équipe UPSTI

ETUDE D'UN TENSIOMETRE ELECTRIQUE

PARTIE MODELISATION

Question 1

Erreur dans le sujet : la force doit valoir F/2 dans le modèle simplifié, pour conserver la même action du bâti en A. Les questions seront traitées en corrigeant cette erreur.



(a) Modèle complet (b) Modèle simplifié

Équivalence des deux modèles :

- Le modèle complet est symétrique, on sait alors que la déformée de la poutre possède une tangente horizontale en son milieu ($x = L$). La liaison en B sur le modèle simplifié bloque la rotation suivant $\vec{z}$, ce qui assure également une pente nulle.
- Le blocage de la translation suivant $\vec{x}$ est sans conséquence, puisqu'il n'y aura aucune déformation suivant $\vec{x}$ (aucun effort normal).
- Au niveau des actions mécaniques, on doit conserver le même torseur de cohésion et les mêmes actions aux appuis (en A), ce qui est obtenu en appliquant une force de $-\frac{F}{2}\,\vec{y}$ en B.

Intérêt de cette simplification :

En utilisant la méthode statique, le degré d'hyperstatisme pour un problème plan peut se calculer ainsi :

$h = m + Ns - 3\,(p-1)$ avec :

- h : degré d'hyperstatisme recherché;
- m : nombre de mobilités : $\boldsymbol{m = 0}$;
- N_S : nombre d'inconnues statiques (dépend du modèle retenu);
- p : nombre de pièces (bâti compris) : $\boldsymbol{p = 2}$.

Degré d'hyperstatisme du modèle complet (en problème plan) :

- $\boldsymbol{N_S = 6}$ car il y a 3 inconnues de liaisons par liaison encastrement ;
- $h = m + Ns - 3\,(p-1) = 0 + 3 + 3 - 3 \cdot 1 = 3.$
- Ce résultat peut sembler évident du fait de l'encastrement en trop.

Degré d'hyperstatisme du modèle simplifié (en problème plan) :

- $\boldsymbol{N_S = 5}$:
  - ⇨ 3 inconnues de liaisons pour la liaison encastrement ;
  - ⇨ 2 inconnues de liaison pour la liaison appui plan.
- $h = 0 + 3 + 2 - 3 \cdot 1 = 2$

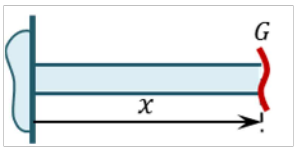
Si on ne tient pas compte des forces selon $\vec{x}$, qui resteront nulles, on se retrouve avec $h = 2$ (modèle complet), et $h = 1$ (modèle simplifié). La simplification permet de n'étudier qu'un seul tronçon, et de réduire les calculs.

Question 2

On isole le tronçon de gauche $[AG]$, qui est soumis aux actions mécaniques suivantes :

$$\{\tau_{(bati\ A\to poutre)}\} = \begin{Bmatrix} X_A & - \\ Y_A & - \\ - & M_A \end{Bmatrix}_{A,R}$$

$$\{\tau_{(coh)}\} = \begin{Bmatrix} T_X & - \\ T_Y & - \\ - & M_Z \end{Bmatrix}_{G,R} = \begin{Bmatrix} T_X & - \\ T_Y & - \\ - & M_Z + xT_Y \end{Bmatrix}_{A,R}$$



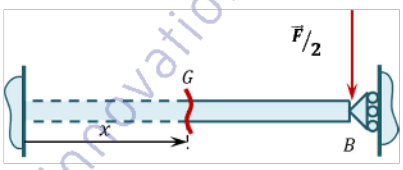
Principe fondamental de la statique, équation des moments en A en projection sur $\vec{z}$:

$$\boldsymbol{M_A + M_z + xT_Y = 0}$$

On isole maintenant le tronçon de droite $[GB]$, qui est soumis aux actions mécaniques suivantes :

$$\{\tau_{(bati\ B\to poutre)}\} = \begin{Bmatrix} X_B & - \\ 0 & - \\ - & M_B \end{Bmatrix}_{A,R} \quad -\{\tau_{(coh)}\} = -\begin{Bmatrix} T_X & - \\ T_Y & - \\ - & M_Z \end{Bmatrix}_{G,R}$$

$$\{\tau_{(F\to poutre)}\} = \begin{Bmatrix} 0 & - \\ -\frac{F}{2} & - \\ - & 0 \end{Bmatrix}_{B,R}$$



Le principe fondamental de la statique (somme des résultantes en projection sur $\vec{y}$) amène à :

$$T_Y = -\frac{F}{2}$$

Au final, on a donc :

$$\boldsymbol{M_A + M_z - x\frac{F}{2} = 0}$$

Question 3

On a :

$$EI\frac{d^2u_y(x)}{dx^2} = M_z \Leftrightarrow EI\frac{d^2u_y(x)}{dx^2} = x.\frac{F}{2} - M_A$$

En intégrant deux fois successivement on obtient :

$$EI\frac{du_y(x)}{dx} = \frac{x^2F}{4} - M_A.x + C_1$$

$$EIu_y(x) = \frac{x^3F}{12} - \frac{M_A.x^2}{2} + C_1.x + C_2$$

Conditions aux limites :

- $u_{y\,(x=0)} = 0 \Rightarrow C_2 = 0$
- $\frac{du_y}{dx}_{(x=0)} = 0 \Rightarrow C_1 = 0$
- $\frac{du_y}{dx}_{(x=L)} = 0 \Rightarrow M_A = \frac{LF}{4}$

Finalement :

$u_y(x) = \frac{F}{12EI}x^3 - \frac{LF}{8EI}x^2$ soit : $A_1 = \frac{F}{12EI}$ et $A_2 = -\frac{LF}{8EI}$

Question 4

La déformation sur la peau supérieure vaut : $\varepsilon(x) = -y\frac{d^2u_y(x)}{dx^2} = -\frac{e}{4}.\left(\frac{F}{EI}x - \frac{LF}{2EI}\right)$

$\varepsilon(x) = \frac{eLF}{8EI} - \frac{eF}{4EI}x$ soit : $A_3 = \frac{eLF}{8EI}$ et $A_4 = -\frac{eF}{4EI}$

Question 5

Pour une meilleure précision de mesure, il faut placer les jauges aux endroits où la déformation est maximale. Puisque $\varepsilon(x) = \frac{eF}{4EI}\left(\frac{L}{2} - x\right)$, les jauges devraient être placées en $x = 0$ et $x = L$, mais ceci n'est pas faisable (on est gêné par l'encastrement et la tige du piston).

$$\varepsilon_1 = \varepsilon\left(x = \frac{L}{4}\right) = \frac{eF}{4EI}\left(\frac{L}{2} - \frac{L}{4}\right) = \frac{eF}{16EI}L \qquad \varepsilon_2 = \varepsilon\left(x = \frac{3L}{4}\right) = \frac{eF}{4EI}\left(\frac{L}{2} - \frac{3L}{4}\right) = -\frac{eL}{16EI}F$$

Et par symétrie $\varepsilon_1 = \varepsilon_4$ et $\varepsilon_2 = \varepsilon_3$.

$$\varepsilon_1 = -\varepsilon_2 = -\varepsilon_3 = \varepsilon_4 = kF \quad \text{avec :} \quad k = \frac{eL}{16EI}$$

Question 6

Question qui n'est peut-être plus au programme de physique…

Pour calculer l'incertitude relative, on applique le logarithme népérien à la relation $R = \rho\frac{l}{S}$:

$$\ln R = \ln\rho + \ln l - \ln S$$

Puis on prend la différentielle de cette équation :

$$\frac{\delta R}{R} = \frac{\delta\rho}{\rho} + \frac{\delta l}{l} - \frac{\delta S}{S}$$

Question 7

La section vaut $= a \cdot b$, d'où : $\frac{\delta S}{S} = \frac{\delta a}{a} + \frac{\delta b}{b} = -2\nu\frac{\delta l}{l}$

$$\frac{\delta R}{R} = \frac{\delta\rho}{\rho} + (1 + 2\nu)\frac{\delta l}{l}$$

Question 8

$$\frac{\delta\rho}{\rho} = C\frac{\delta V}{V} = C\left(\frac{\delta l}{l} + \frac{\delta S}{S}\right) = C(1 - 2\nu)\frac{\delta l}{l}$$

$$\frac{\delta R}{R} = [1 + 2\nu + C(1 - 2\nu)]\frac{\delta l}{l} \qquad K = 1 + 2\nu + C(1 - 2\nu)$$

Question 9

$\frac{\delta R}{R} = \Pi\sigma + (1 + 2\nu)\frac{\delta l}{l}$ et $\sigma = E\frac{\delta l}{l}$

$$\frac{\delta R}{R} = [\,\Pi E + 1 + 2\nu\,]\frac{\delta l}{l} \qquad K = \Pi E + 1 + 2\nu$$

Question 10

$$K_{cuivre} = 1 + 2\nu + 1(1 - 2\nu) = 2 \qquad K_{silicium} = 10^{-9} \cdot 10^{11} + 1 + 2 * 0.4 = 101{,}8$$

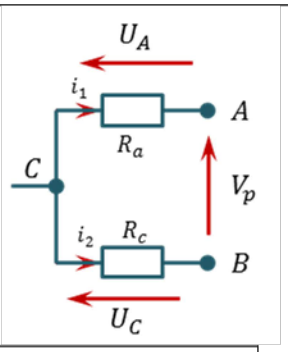
Pour une même déformation $\frac{\delta l}{l}$ à mesurer, le silicium produira une plus grande variation de résistance (environ 50 fois supérieure). La jauge en silicium est donc beaucoup plus sensible.

Question 11

Le phénomène principal qui peut produire une erreur sur la mesure est la température : dérive due à la dilatation thermique, dilatation différentielle entre jauge et support.

Question 12

En appliquant la loi des mailles dans la maille ABC ci-contre, on a :



$$V_p = U_c - U_a = R_c.i_2 - R_a.i_1$$

avec : $i_1 = \frac{V_{alim}}{R_a+R_b}$ et $i_2 = \frac{V_{alim}}{R_c+R_d}$

$$V_p = \left(\frac{R_c}{R_c+R_d} - \frac{R_a}{R_a+R_b}\right)V_{alim} = \frac{R_bR_c - R_aR_d}{(R_a+R_b)(R_c+R_d)}V_{alim}$$

Question 13

$$V_p = \frac{(R_0+\Delta R_b)(R_0+\Delta R_c) - (R_0+\Delta R_a)(R_0+\Delta R_d)}{(2R_0+\Delta R_c+\Delta R_d)(2R_0+\Delta R_a+\Delta R_b)}V_{alim} = V_p = \frac{R_0(\Delta R_b+\Delta R_c) - R_0(\Delta R_a+\Delta R_d)}{4R_0^2+o(R_0^2)}V_{alim}$$

(En négligeant les $\Delta R_i \Delta R_j$)

$$V_p = \frac{\Delta R_b+\Delta R_c - \Delta R_a - \Delta R_d}{4R_0}V_{alim} = \frac{V_{alim}}{4}\left(\frac{\Delta R_b}{R_0} + \frac{\Delta R_c}{R_0} - \frac{\Delta R_a}{R_0} - \frac{\Delta R_d}{R_0}\right)$$

Question 14

Si on considère la question 5 : $\varepsilon_1 = -\varepsilon_2 = -\varepsilon_3 = \varepsilon_4$. Puisque ΔR_b et ΔR_c s'additionnent, il faut choisir R_c associé à la jauge 4, et R_a et R_d associés aux jauges 2 et 3 indifféremment. Ainsi toutes les variations de résistance vont se cumuler (en choisissant mal, on peut obtenir une tension V_p toujours nulle).

Question 15

$$V_p = \frac{V_{alim}}{4}\left(\frac{\Delta R_b}{R_0} + \frac{\Delta R_c}{R_0} - \frac{\Delta R_a}{R_0} - \frac{\Delta R_d}{R_0}\right) = \frac{V_{alim}}{4}K(\varepsilon_1 + \varepsilon_4 - \varepsilon_2 - \varepsilon_3)$$

$$V_p = K\,\varepsilon_1\,V_{alim}$$

Question 16

$$V_p = K\,\varepsilon_1\,V_{alim} = K.\frac{eL}{16EI}F.V_{alim} = K\,\frac{eL}{16EI}PS_mV_{alim}$$

La sensibilité de ce capteur linéaire correspond au rapport grandeur de sortie (tension) sur grandeur mesurée (pression), c'est-à-dire au gain du capteur.

$$\text{Sensibilité} = \frac{V_p}{P} = K.\frac{eL}{16EI}.S_m.V_{alim}$$

Question 17

La tension mesurée pour une pression de 50 kPa vaut :

$$V_p = 101{,}8 * \frac{0{,}1 * 2}{16 * 10^{11} * 2.10^{-4}} * 50.10^3 * 4 * 16 = 0{,}2\ V$$

PARTIE INFORMATIQUE

Question 18

L'intervalle de mesure est $[0\,,1350\ \text{HPa}] = [\,0\,,1000\ \text{mmHg}\,]$ car $\frac{1350}{1013}\cdot 750 \approx\ 1000$.

Avec une précision de $0{,}02\ \text{mmHg}$, il faut $\frac{1000}{0{,}02} = 50000$ points.

Les CAN sont capables de coder $2^{10} = 1024$, $2^{12} = 4096$, ou $2^{16} = 65536$ valeurs différentes.

Seul le CAN avec une résolution de 16 bits permet de coder 50000 valeurs.

Question 19

Si on choisit de stocker en mémoire la valeur de la pression (réel, toujours positif), il faut prendre un type de stockage **réel codé sur 32 bits**. (16 bits suffirait, mais ce type n'est pas proposé).

Ainsi, pour une mesure de 60 secondes échantillonnées à 1000 Hz, il faut donc coder 60 000 valeurs de 32 bits (soit 4 octets). Quantité de mémoire nécessaire pour stocker le tableau : 60000*4 = **240 ko**.

Remarque : une autre solution est de stocker les valeurs « brut de sortie de CAN » avec des entiers sur 16 bits, mais il faudrait ensuite traiter les informations pour retrouver la valeur en mmHg. Cette option est moitié moins gourmande en mémoire.

Question 20

On est en présence d'un filtre passe-bas, avec une coupure au niveau de $\omega =\ 20\ rad/s$.

Les pulsations cardiaques sont au maximum de pulsation $\omega = 2\pi f = 2\pi\frac{190}{60} \approx\ 20\ rad/s$. Les pulsations inférieures à la pulsation cardiaque maximale ne seront donc pas affectées par le filtre (ce qui est préférable si on veut pouvoir les mesurer !).

Remarque : on est limite, on aurait pu choisir une pulsation de coupure un peu plus élevée

L'amortissement $z = 0{,}7$ choisi, est la valeur la plus faible permettant de ne pas avoir de résonance (qui fausserait les mesures aux alentours de $=\ 20\ rad/s$). Cette valeur $z = 0{,}7$ permet également, dans un diagramme de Bode, d'avoir la courbe réelle de gain au plus proche des asymptotes. La coupure est donc nette et brutale, c'est intéressant car la pulsation cardiaque maximale à ne pas filtrer est proche de la cassure.

Question 21

En discrétisant l'équation différentielle, on a :

$$\frac{1}{\omega^2}\ddot{U}_f(t) + \frac{2z}{\omega_0}\dot{U}_f(t) + U_f(t) = U_e(t) \Rightarrow \frac{1}{\omega^2}\ddot{U}_{f,i+1} + \frac{2z}{\omega}\dot{U}_{f,i+1} + U_{f,i+1} = U_{e,i+1}$$

En introduisant les relations du schéma de Newmark dans l'équation différentielle, on a :

$$\frac{1}{\omega^2}\ddot{U}_{f,i+1} + \frac{2z}{\omega_0}\left(\dot{U}_{f,i} + (1-\gamma)T_e\ddot{U}_{f,i} + \gamma T_e\ddot{U}_{f,i+1}\right) + U_{f,i} + T_e\dot{U}_{f,i} + T_e^2\left(\frac{1}{2}-\beta\right)\ddot{U}_{f,i} + T_e^2\beta\ddot{U}_{f,i+1} = U_{e,i+1}$$

On a donc :

$$\ddot{U}_{f,i+1}\left(\frac{1}{\omega^2} + \beta T_e^2 + \frac{2z}{\omega}\gamma T_e\right) + U_{f,i} + \left(T_e + \frac{2z}{\omega}\right)\dot{U}_{f,i} + \left(T_e^2\left(\frac{1}{2}-\beta\right) + \frac{2z}{\omega}(1-\gamma)T_e\right)\ddot{U}_{f,i} = U_{e,i+1}$$

$$\ddot{U}_{f,i+1} = \frac{1}{\left(\frac{1}{\omega^2} + \frac{2z}{\omega}\gamma T_e + \beta T_e^2\right)}\left(U_{e,i+1} - \left(T_e^2\left(\frac{1}{2}-\beta\right) + \frac{2z}{\omega}(1-\gamma)T_e\right)\ddot{U}_{f,i} - \left(T_e + \frac{2z}{\omega}\right)\dot{U}_{f,i} - U_{f,i}\right)$$

On obtient la relation demandée, avec :

$$B_2 = -\left[\frac{2z}{\omega}(1-\gamma)T_e + {T_e}^2\left(\frac{1}{2}-\beta\right)\right] \qquad B_3 = -\left[\frac{2z}{\omega} + T_e\right] \qquad B_4 = -1$$

Question 22

Le schéma d'intégration n'est **stable** que **si** $T_e < 0.175\ s$ (on est en limite de stabilité pour $T_e = 0{,}175s$)

Question 23

L'erreur du schéma d'intégration est proportionnelle au pas de temps ($\eta = 5T_e$), la convergence est donc d'**ordre 1**, c'est une **convergence linéaire.**

Remarque : cette conclusion est valable pour les choix de β = 1/6 et γ= 1/2 qui ont été faits (valeurs classiques confirmées par une simulation Python (fichier joint), permettant de retrouver les courbes de la figure 7).

Question 24

Une fréquence d'échantillonnage de $1000Hz$ correspond à un pas de temps $T_e = 0{,}001s$. Ce choix permet d'obtenir une très bonne stabilité ($0{,}001 < 0{,}175$) et une erreur très raisonnable ($\eta = 0{,}005$, soit $0{,}5\%$).

Question 25

```
def newmark(gamma,beta,omega,z,e,Te):
    """retourne le signal filtré de la liste e, par le schéma de Newmark"""
    n=len(e)
    uf,ufp,ufpp =[0],[0],[0]      #valeurs initiales de uf, uf' et uf''

    for i in range(n-1):
        ufpp.append(B1*(e[i+1]+B2*ufpp[i]+B3*ufp[i]+B4*uf[i]))
        ufp.append(ufp[i]+(1-gamma)*Te*ufpp[i]+gamma*Te*ufpp[i+1])
        uf.append(uf[i]+Te*ufp[i]+Te*Te*(0.5-beta)*ufpp[i]+Te*Te*beta*ufpp[i+1])
    return uf
```

Question 26

Dans la fonction précédente, on occupe de l'espace mémoire en stockant dans des listes toutes les valeurs de $\dot{U}_{f,i}$ et $\dot{U}_{f,i}$ pour i ∈ [0 , N-1], alors que seules les valeurs aux instants iT_e et $(i+1)T_e$ sont utiles.

```
def newmark_optimise(gamma,beta,omega,z,e,Te):
    """retourne le signal filtre de la liste e, par le schéma de Newmark
    Minimise la taille des variables intermédiaires,
    on ne stocke pas les valeurs des dérivées dans des listes"""
    n=len(e)
    uf=[0]              #valeurs initiales
    ufp0,ufpp0=0,0      #valeurs initiales

    for i in range(n-1):
        ufpp1=(B1*(e[i+1]+B2*ufpp0+B3*ufp0+B4*uf[i]))
        ufp1=ufp0+(1-gamma)*Te*ufpp0+gamma*Te*ufpp1
        uf.append(uf[i]+Te*ufp0+Te*Te*(0.5-beta)*ufpp0+Te*Te*beta*ufpp1)
        ufpp0=ufpp1
        ufp0=ufp1
    return uf
```

Question 27

La méthode choisie est de détecter les changements de signes de la dérivée. Le programme fonctionne avec les hypothèses du sujet : signal parfaitement lissé, pression systolique toujours possible à trouver.
Remarque : un minimum est toujours détecté avant un maximum, on peut donc tester le critère P > 4.10^{-5} lors de la mesure du maximum.

```
def pression_systolique(data):
    """retourne la pression systolique en recherchant le minimum
    et le maximum local"""
    N=len(data)
    derivee_avant=data[1]-data[0]
    for i in range(1,N):
        derivee_apres=data[i]-data[i-1]
        if derivee_apres>=0 and derivee_avant<0:
            min=data[i]
        if derivee_apres<=0 and derivee_avant>0:
            max=data[i]
            if (max-min)/min>4e-5:
               return max

        derivee_avant=derivee_apres
```

Question 28

Cette fonction est une simple adaptation de la précédente, on parcourt les données par temps croissant, jusqu'à ce que le critère ne soit plus vérifié. Une alternative aurait été de parcourir les données à l'envers.

```
def pression_diastolique(data):
    """retourne la pression diastolique en recherchant le minimum
    et le maximum local"""
    N=len(data)
    derivee_avant=data[1]-data[0]
    pdiastol=0
    for i in range(1,N):
        derivee_apres=data[i]-data[i-1]
        if derivee_apres>=0 and derivee_avant<0:
            min=data[i]
        if derivee_apres<=0 and derivee_avant>0:
            max=data[i]
            if (max-min)/min>4e-3 and max>40:     # le critère est vérifié
               pdiastol=max
            if (max-min)/min<4e-3 and pdiastol!=0:  # le critère n'est plus vérifié
               return pdiastol                      # dernière valeur valide

        derivee_avant=derivee_apres
```

Question 29

Une clé primaire permet d'identifier de manière unique un enregistrement dans une table. Une clé primaire est unique et peut être composée d'un ou de plusieurs champs de la table (deux lignes distinctes ne peuvent pas avoir la même valeur pour les champs définis au niveau de la clé primaire). On choisit souvent d'ajouter un champ *identifiant* entier indépendant et auto incrémenté comme clé primaire.
Pour la table « *patients* », le champ « *numero_secu* » est le meilleur candidat pour une autre clé primaire (mais les étrangers et les enfants peuvent ne pas avoir de numéro de sécurité sociale). Pour la table « *mesures* », c'est le champ « *datetime* » qui pourrait convenir, il est peu probable qu'il y ait deux mesures simultanées. (Dans ce cas, on pourrait éventuellement prendre le couple « *datetime,pid* ».)

Question 30

```
SELECT datetime, pdias, psyst, pouls
        FROM mesures
        WHERE datetime>=time1 AND datetime<= time2
```

Question 31

```
SELECT datetime, pdias, psyst, pouls
        FROM mesures
        WHERE datetime>=time1 AND datetime<= time2 AND pid=id_patient AND type='tension'
```

Question 32

```
clf()
# Pression systolique, trait bleu plein, marqueur cercle
plot(resultat_requete[0], resultat_requete[1],'bo-')
# Pression diastolique, trait pointillé rouge, marqueur étoile
plot(resultat_requete[0], resultat_requete[2],'r*--')
# Pouls, trait mixte vert, marqueur diamant
plot(resultat_requete[0], resultat_requete[3],'gD-.')

xlabel('Temps (s)')
legend(['P systolique','P diastolique','Pouls'])
show()
```

Question 33

```
def analyse(valeurs):
    """ retourne les min, max, et moyenne d'une liste"""
    min=valeurs[0]
    max=valeurs[0]
    somme=0
    for k in valeurs:
        if k>max:
            max=k
        if k<min:
            min=k
        somme+=k
    moyenne=somme/len(valeurs)

    return min,max,moyenne
```

Question 34

En comptant un coût unitaire pour les affectations, comparaisons et les opérations, la complexité vaut :

$$C_{pire} = 3_{\text{affectation valeurs initiales}} + n \cdot (2_{\text{comparaisons}} + 1_{\text{affectation}} + 2_{\text{calcul de la somme}}) + 2_{\text{calcul de la moyenne}}$$

$$\boldsymbol{C_{pire} = 5 + 5n}$$

Au meilleur des cas, on a trouvé dès le début le max et le min, cela enlève l'affectation dans la boucle.

$$\boldsymbol{C_{meilleur} = 5 + 4n}$$

On est évidemment sur une complexité linéaire en $\mathcal{O}(n)$.

Question 35

On peut choisir de faire un tri par insertion (une première boucle parcourt l'ensemble des éléments de la liste, en commençant par le second élément. Une seconde boucle descend dans la liste pour remonter les prédécesseurs tant que leur valeur est supérieure à la valeur de l'élément à mettre à sa place).

```
def tri_insertion(liste):
    """ tri une liste par insertion"""
    n=len(liste)
    for i in range (1,n):
        valeur=liste[i] # on mémorise l'élément
        j=i
        while j>0 and liste[j-1]>valeur:
            liste[j]=liste[j-1]
            j=j-1
        liste[j]=valeur
    return liste

def mediane(valeurs):
    """calcule la valeur médiane d'une liste non triée"""
    tri_insertion(valeurs)
    if len(valeurs)%2 == 0 :
        i=len(valeurs)//2
        return (valeurs[i-1]+valeurs[i])/2
    else :
        i = len(valeurs)//2
    return valeurs[i]
```

Question 36

Dans la fonction « *médiane* », le cout vient exclusivement de la fonction « *tri_insertion* ».

Complexité du tri par insertion :

- **Meilleur des cas** : le tableau est déjà trié. Il y a donc $n-1$ comparaisons à effectuer. La complexité est donc de classe linéaire : $C(n) = \mathcal{O}(n)$.
- **Pire des cas** : le tableau est trié à l'envers. Il y a alors une comparaison à effectuer à la première étape, puis deux, … puis $n-1$. On en déduit donc un nombre total de $\frac{n\cdot(n-1)}{2}$ comparaisons. La complexité est donc de classe quadratique : $C(n) = \mathcal{O}(n^2)$.

La méthode de tri choisie est donc très efficace si la liste est presque déjà triée, mais moins performante sur une liste quelconque qu'un tri rapide ($\mathcal{O}(n\log(n)$ en moyenne et $\mathcal{O}(n^2)$ dans le pire des cas) ou un tri fusion ($\mathcal{O}(n\log(n)$ dans tous les cas).

Question 37

```
SELECT nom, prenom, telephone
    FROM patients
    WHERE id = (
        SELECT pid FROM mesures
            WHERE psyst>160 AND pdias>110 AND pouls>100 AND pouls<150)
```

Ou bien :

```
SELECT nom, prenom, telephone FROM patients
    JOIN mesures
    ON id = pid
        WHERE psyst>160 AND pdias>110 AND pouls>100 AND pouls<150
```