COMMENT RÉALISER DES ANALYSES STATISTIQUES AVEC MATLAB ?

Ce document constitue une courte introduction à la boîte à outils Statistique du logiciel MATLAB.

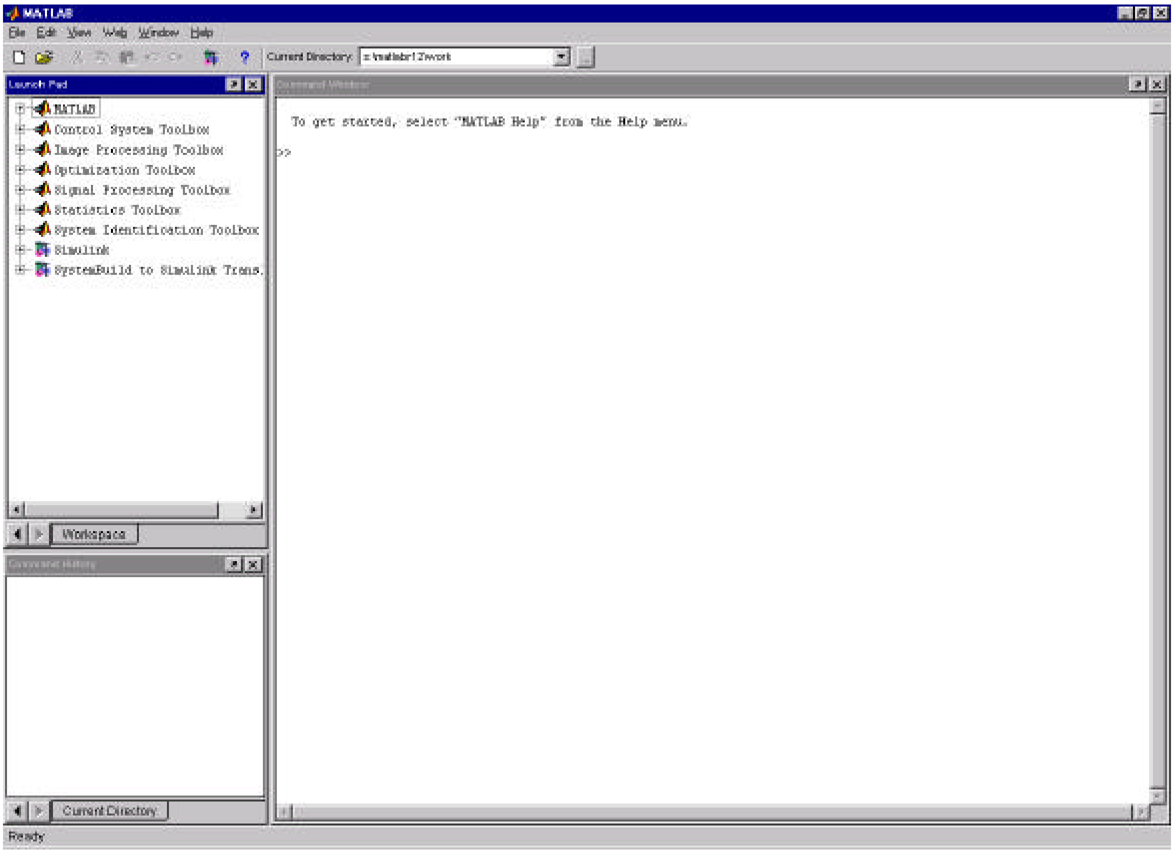
Vous trouvez le code MATLAB permettant d'analyser les données de l'hélicoptère, comme cela est fait tout au long de votre syllabus.
Ces données peuvent être téléchargées sur le site du cours : *http://www.stat.ucl.ac.be/cours/fsat2*

Vous trouvez également à la fin de ce document les fonctions de la toolbox STAT qui vous seront utiles pour vos analyses statistiques en T2, T5 et T6.

1. Généralités

· **3 fenêtres dans Matlab**

Launch Pad - accès aux démos et documentations des différentes toolbox
Command History - historique des lignes de commandes exécutées
Command Window - fenêtres où l'on tape les lignes de commandes



· **Aide : 2 commandes utiles**

help – aide on-line sur une fonction Matlab (ce que fait cette fonction et quels sont ses paramètres)

```
>> help mean

 MEAN   Average or mean value.
   For vectors, MEAN(X) is the mean value of the elements in X. For matrices, MEAN(X) is a row vector
containing the mean value of each column.  For N-D arrays, MEAN(X) is the mean value of the elements
along the first non-singleton dimension of X.
    MEAN(X,DIM) takes the mean along the dimension DIM of X.

   Example: If X = [0 1 2
                    3 4 5]
    then mean(X,1) is [1.5 2.5 3.5] and mean(X,2) is [1
                                                       4]

   See also MEDIAN, STD, MIN, MAX, COV.
```

lookfor – aide on-line par mots-clés

```
>> lookfor histogram
HIST  Histogram.
HISTC Histogram count.
ROSE   Angle histogram plot.
HISTEQ Enhance contrast using histogram equalization.
IMHIST Display histogram of image data.
IMADJDEMO Intensity adjustment and histogram equalization demo.
HISTFIT Histogram with superimposed fitted normal density.
IMHISTC Image histogram.
```

· **Vérifier / modifier le répertoire par défaut** (passer par exemple du répertoire Z: à   Z:\T2)
pwd – afficher le répertoire courant

```
>> pwd
ans =
z:\
```

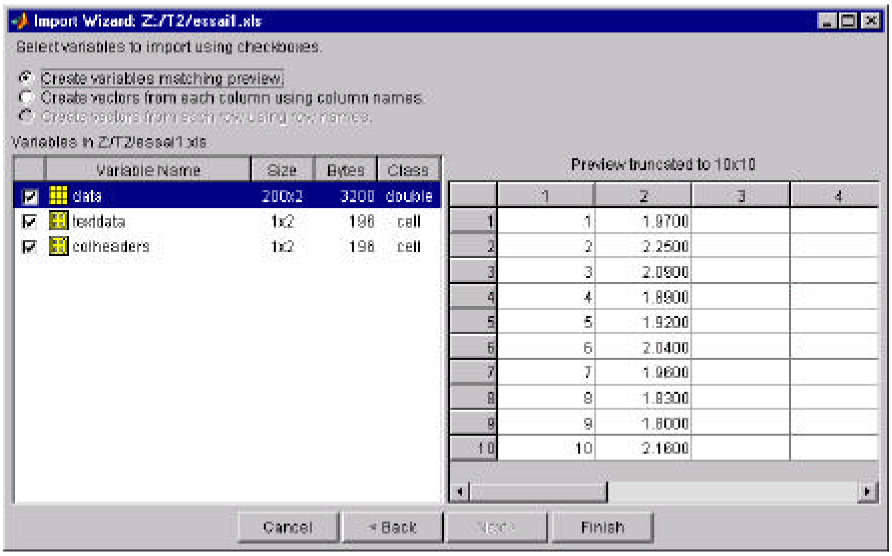
cd – changer le répertoire courant

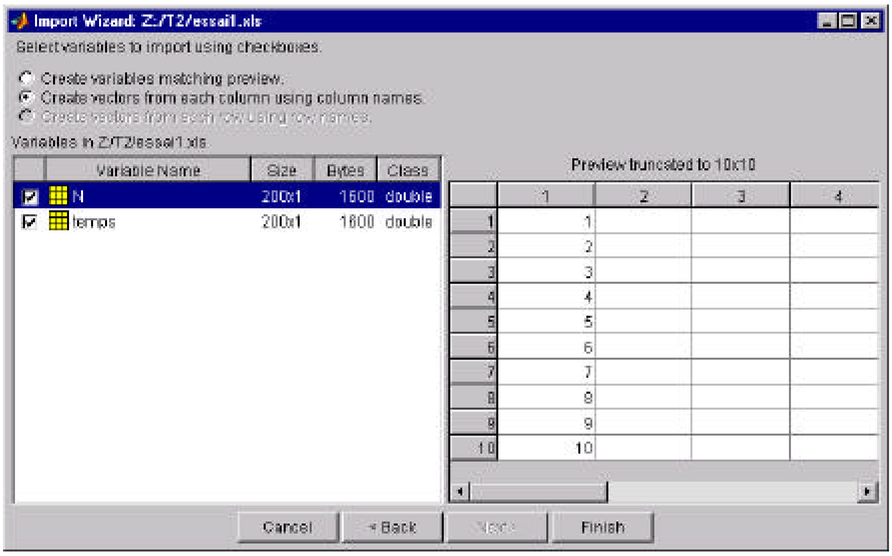
```
>> cd z:\T2
>> pwd
ans =
z:\T2
```

→ Pensez à modifier le répertoire par défaut juste après ouverture de Matlab

· **Importer des données (essai1.xls par exemple)**

- Sauvez dans votre répertoire les fichiers qui se trouvent sur le site *http://www.stat.ucl.ac.be/cours/fsat2*
- Utilisez le menu File / Import data
- Sélectionnez le fichier à importer (.csv, .xls, .txt, .dat, .tab ...) : Spreadsheet (*.csv, *.xls, *.wk1)
- Choisissez si vous voulez importer les données sous forme d'une matrice ou sous la forme de vecteurs pour chaque colonne du fichier Excel
- Terminez en cliquant sur Finish
- Changez le nom par défaut (*data*) des données importées en effectuant la commande ***essai1 = data***
- Supprimez l'ancien fichier *data* en effectuant la commande ***clear data***





· Lister les variables de votre répertoire

who – liste les variables de votre répertoire
whos - liste les variables de votre répertoire ainsi que des renseignements sur leur taille et leur forme

• Sauvegarder l'espace de travail

Afin de conserver votre espace de travail càd toutes les variables de votre répertoire, il faut le sauvegarder dans un fichier .mat avant de fermer Matlab. Quand vous ouvrez à nouveau matlab, il vous suffit alors d'ouvrir ce fichier .mat pour retrouver toutes vos données. (utilisez le menu ***File – Save Workspace as***...)

• Principales opérations sur les vecteurs et sur les matrices

- Créer une matrice explicitement

```
>> A = [1 2 3 ; 4 5 6]
A =
   1    2    3
   4    5    6
```

- Créer un vecteur ligne

```
>> B = [1 2 3]
B =
   1    2    3
```

- Créer un vecteur colonne

```
>> C = [1 ; 2 ; 3]
C =
   1
   2
   3
```

ou en transposant

```
>> C = [1  2  3]'
C =
   1
   2
   3
```

- Accéder à l'élément de la matrice A se trouvant à l'intersection de la 2°ligne et de la 3° colonne

```
>> A(2,3)
ans =
   6
```

- Sélectionner une ligne ou une colonne d'une matrice

```
>> A(2,:)
ans =
   4    5    6
```

```
>> A(:,2)
ans =
   2
   5
```

- Modifier un élément d'une matrice

```
>> A(2,3) = 99
A =
   1    2    3
   4    5   99
```

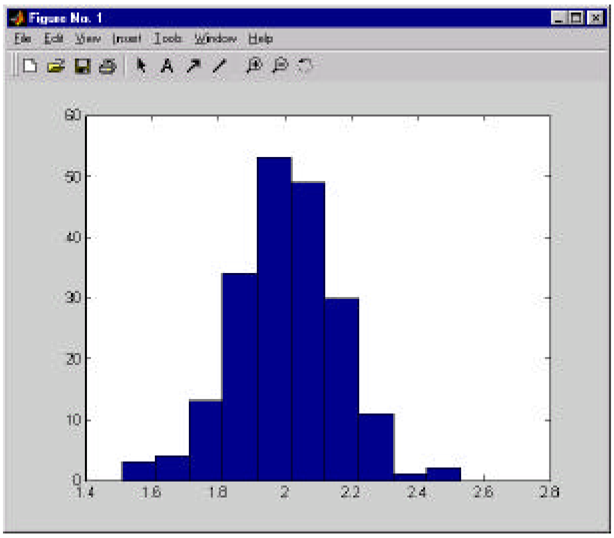
2. Essai1 – 200 lancés du prototype A

Le fichier Essai1 reprend les temps de vol d'un hélicoptère lancé 200 fois. Nous allons voir comment obtenir avec MATLAB l'histogramme et le graphique temporel qui permettent de visualiser respectivement la distribution d'une variable quantitative et l'évolution temporelle d'une série de données (*cfr* paragraphes 3.1 et 3.3).

- **Histogramme**

Comme le temps de vol est une variable quantitative continue, on peut résumer les temps des 200 lancés par un histogramme.

>> hist(essai1(:,2))



→ Cet histogramme sur la deuxième colonne de la matrice *essai1* est celui obtenu par défaut avec la fonction ***hist***. Par défaut, les données sont divisées en 10 classes de même longueur et le nombre d'observations contenues dans chacune est représenté. Mais avec les données, cela donne des classes très artificielles et qui n'ont pas beaucoup de sens… C'est pourquoi, pour faire l'histogramme, on peut utiliser une autre méthode qui permet de déterminer les bornes des classes.
Si l'on calcule les temps de vol minimum et maximum, on constate que les temps de vol sont entre 1.51 et 2.53 :

```
>> min(essai1(:,2))
ans =
   1.5100
```

```
>> max(essai1(:,2))
ans =
   2.5300
```

On peut alors utiliser la fonction ***histc*** pour compter le nombre d'observations qui tombent dans les classes [1.5,1.6), [1.6,1.7), [1.6,1.7), [1.7,1.8), [1.8,1.9), [1.9,2), [2,2.1), [2.1,2.2), [2.2,2.3), [2.3,2.4), [2.4,2.5) et [2.5,2.6) :

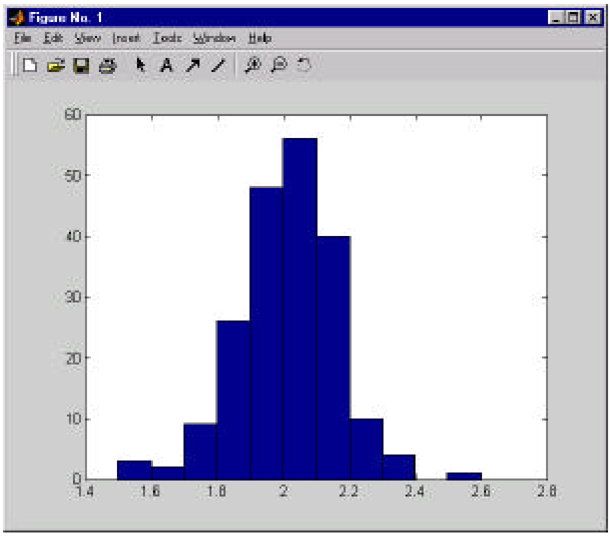
```
>> N = histc(essai1(:,2),[1.5 1.6 1.7 1.8 1.9 2 2.1 2.2 2.3 2.4 2.5])

N =

    3
    2
    9
   26
   48
   56
   40
   10
    4
    0
    1
```

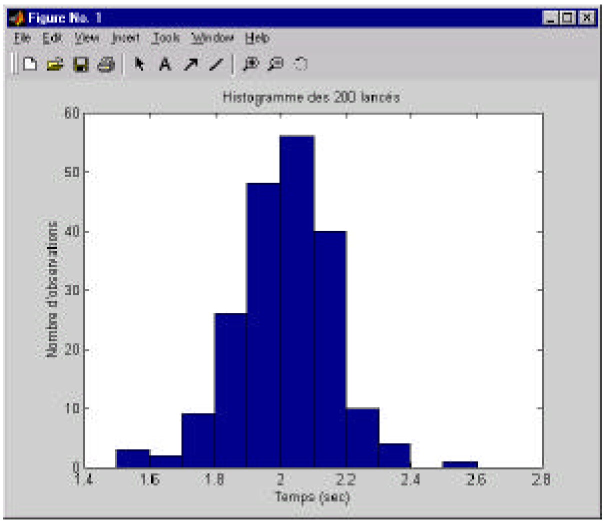
Finalement, on fait l'histogramme avec la commande ***bar*** :

>> bar([1.5 1.6 1.7 1.8 1.9 2 2.1 2.2 2.3 2.4 2.5],N,'histc')

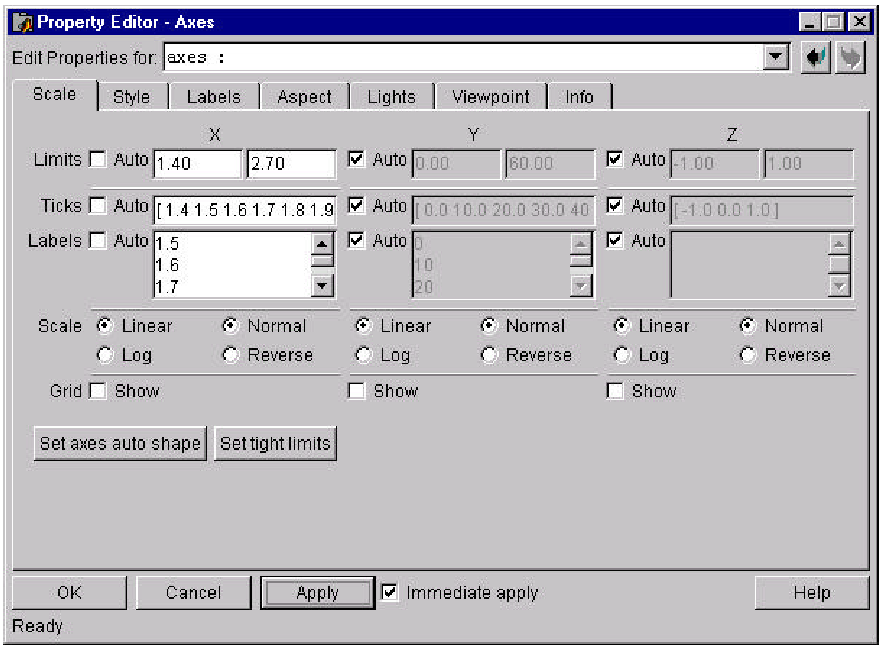


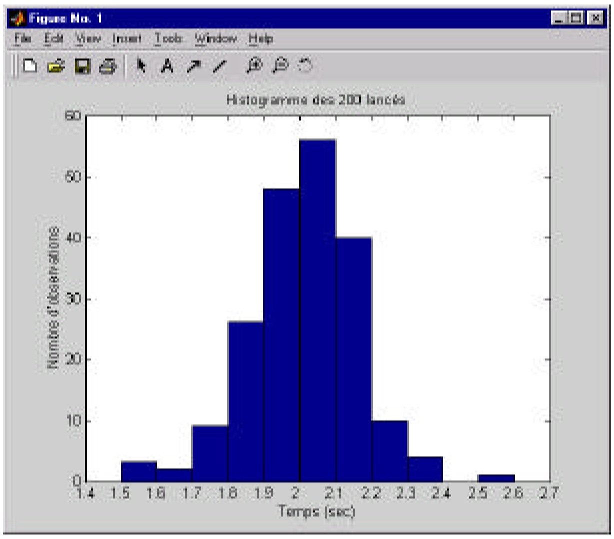
On peut rajouter un titre au graphique et des labels aux axes en utilisant le menu *Insert* de la fenêtre graphique ou en utilisant les commandes suivantes:

>> title('Histogramme des 200 lancés')
>> xlabel('Temps (sec)')
>> ylabel('Nombre d''observations')



On peut également modifier les chiffres sur l'axe horizontal en utilisant le menu *Edit – Axes properties*, s'il l'on voulait par exemple rajouter toutes les bornes des classes.

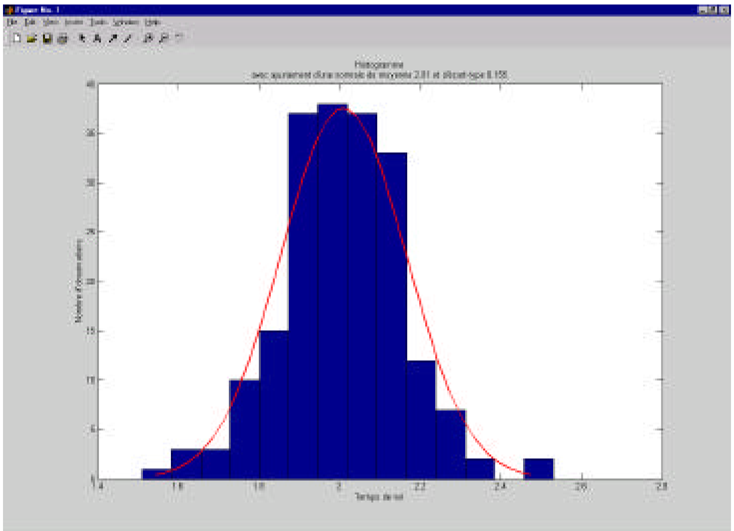




· Histogramme + courbe Normale

On peut superposer une courbe Normale sur un histogramme pour voir si la distribution Normale ajuste bien les données. On utilise pour cela la fonction ***histfit*** qui a comme premier argument les données et comme second argument le nombre de classe (ex : $\sqrt{200} \approx 14$).

>> histfit(essai1(:,2),14)



On peut à nouveau rajouter un titre au graphique et des labels aux axes en utilisant le menu *Insert* de la fenêtre graphique ou en tapant les commandes ***title***, ***xlabel*** et ***ylabel***.

· Diagramme temporel

Le diagramme temporel permet de visualiser l'évolution temporelle des durées de vol des 200 lancés d'hélicoptère.

- On utilise la fonction ***plot***. Les 3 premiers arguments servent à tracer les lignes reliant les points d'abscisses ***essai1(:,1)*** et d'ordonnées ***essai1(:,2)*** par une ligne mauve continue ***'m-'*** . Les 3 derniers arguments servent à dessiner les points d'abscisses ***essai1(:,1)*** et d'ordonnées ***essai1(:,2)*** en utilisant des points bleus ***'b.'***.

>> plot(essai1(:,1),essai1(:,2),'m-',essai1(:,1),essai1(:,2),'b.')

- Ensuite, on peut rajouter un quadrillage pour faciliter la comparaison visuelle

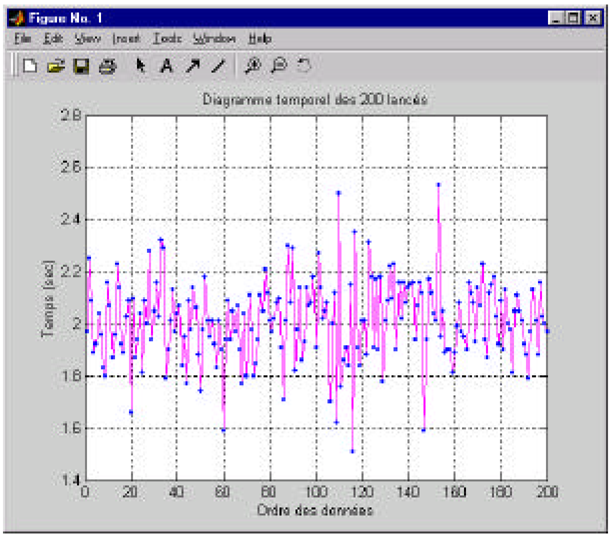
>> grid

- Et finalement, on peut taper les commandes suivantes pour obtenir le titre et les labels aux axes

>> title('Diagramme temporel des 200 lancés')
>> xlabel('Ordre des données')
>> ylabel('Te mps (sec)')
Pour les couleurs des points et des lignes, vous avez les codes suivants :
b = blue, ***g*** = green, ***r*** = red, ***c*** = cyan, ***m*** = magenta, ***y*** = yellow, ***k*** = black

Pour les types de points, vous avez les codes suivants :
. = point, ***o*** = circle, ***x*** = x-mark, ***+*** = plus, ***** = star, ***s*** = square, ***d*** = diamond, ***v*** = triangle (down), ***^*** = triangle (up), ***<*** = triangle (left), ***>*** = triangle (right), ***p*** = pentagram, ***h*** = hexagram

Pour les types de lignes, vous avez les codes suivants :
- = solid, ***:*** = dotted, ***-.*** = dashdot, ***--*** = dashed



3. Essai3 – Comparaison de 4 hélicoptères de dimensions différentes

Le fichier Essai3 reprend les temps de vol de 4 hélicoptères différents lancés chacun 8 fois. Nous allons voir comment obtenir avec MATLAB des graphiques (diagrammes en points et boxplots, *cfr* paragraphe 3.1) et des résumés numériques pour comparer les temps de vols de ces 4 prototypes (Moyenne, écart-type, minimum, maximum et médiane, *cfr* paragraphe 3.3).

• Diagramme en points (bon pour n < 15)

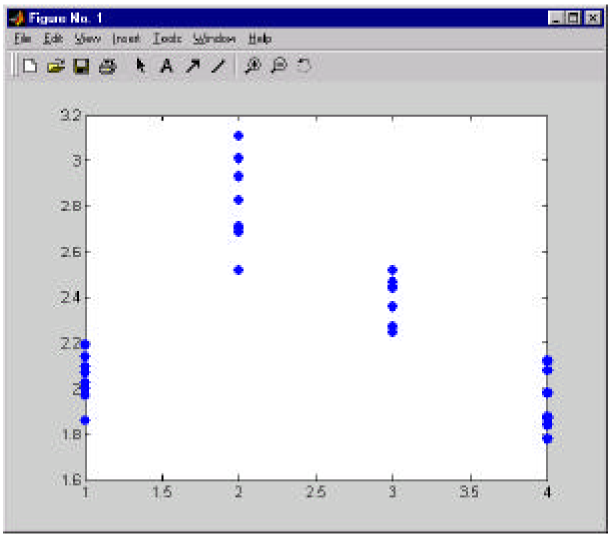
Le diagramme en points ci-dessous est très utile pour comparer les distributions des 4 types d'hélicoptères.

- On crée le vecteur des abscisses du graphe X-Y à partir d'un vecteur unitaire de longueur 8 (***ones(1,8)***) :

```
>> [ones(1,8) ones(1,8)*2 ones(1,8)*3 ones(1,8)*4]
ans =
   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3   4   4   4   4   4   4   4   4
```

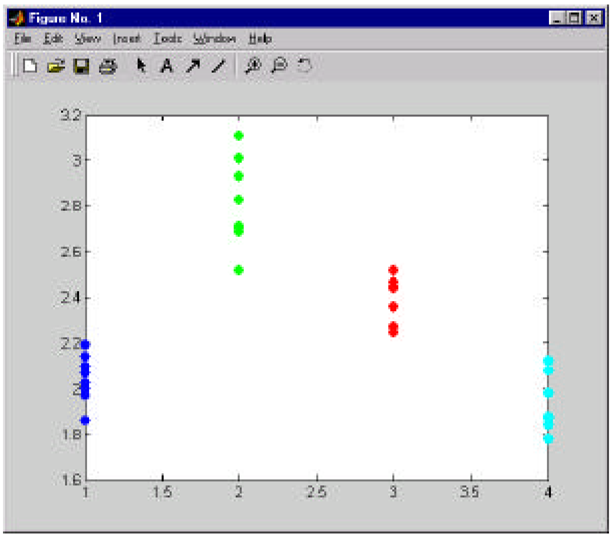
- On utilise la commande plot pour dessiner un graphe X-Y où les abscisses sont dans le premier argument et les ordonnées dans le deuxième argument. On peut en plus spécifier le type et la couleur des points dans les troisième argument.

```
>> plot([ones(1,8) ones(1,8)*2 ones(1,8)*3 ones(1,8)*4],essai3,'b.')
```



On peut également utiliser la commande suivante pour dessiner des points différents selon le type d'hélicoptère.

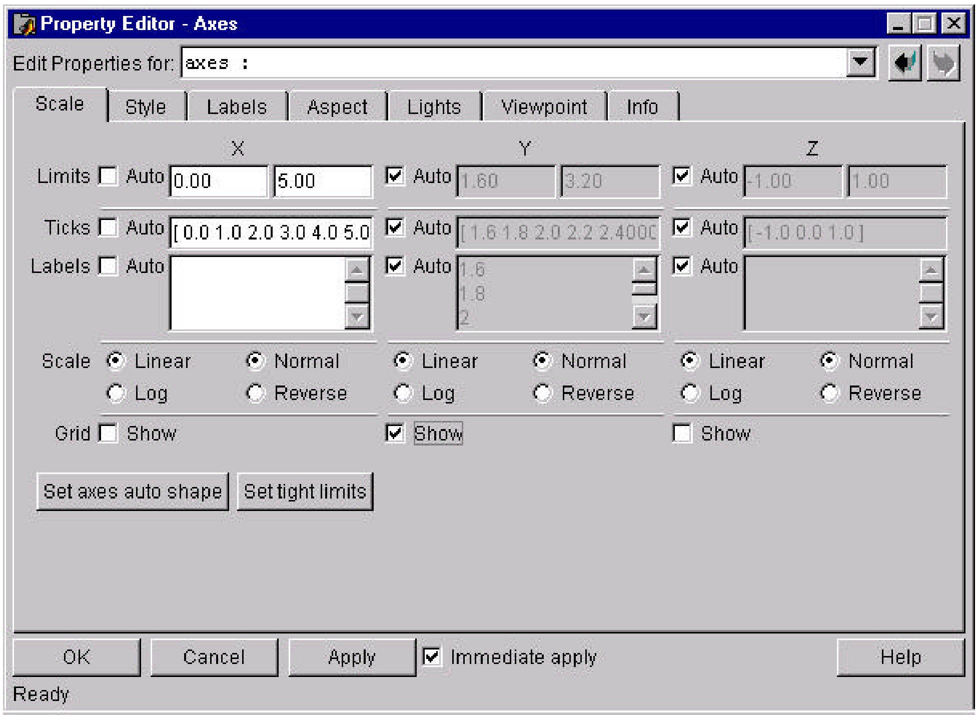
```
>> plot( ones(1,8),essai3(1:8),'b.',
         ones(1,8)*2,essai3(9:16),'g.',
         ones(1,8)*3,essai3(17:24),'r.',
         ones(1,8)*4,essai3(25:32),'c.')
```

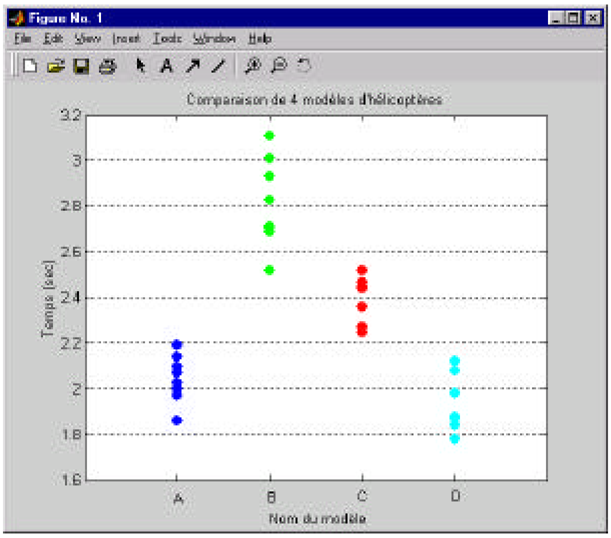


- On rajoute un titre au graphique et des labels aux axes en utilisant les commandes ***title***, ***xlabel*** et ***ylabel***.

>> title('Comparaison de 4 modèles d''hélicoptères')
>> xlabel('Nom du modèle')
>> ylabel('Temps (sec)')

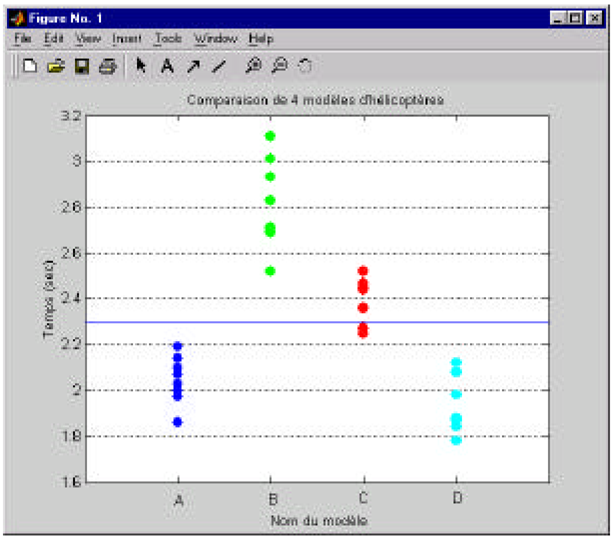
- On modifie l'échelle de l'axe horizontal en utilisant le menu EDIT – AXES PROPERTIES. On modifie d'abord les limites pour mieux voir les 2 séries de points aux extrémités, ensuite on retire les labels. On peut également cocher le quadrillage horizontal pour faciliter la comparaison (Grid - Show). Finalement, en utilisant le menu INSERT – TEXT de la fenêtre graphique, on peut ajouter les lettres A, B, C et D sous les séries de points correspondants. On obtient le graphique ci-dessous.





- On peut également rajouter une ligne horizontale au niveau de la moyenne globale de ces 32 observations (2.2997) pour comparer les 4 groupes d'observations par rapport à cette valeur. On utilise pour cela la commande ***line*** avec comme premier argument les abscisses des 2 points à relier et comme second argument les ordonnées des points à relier.

>> line([0 5],[mean(essai3) mean(essai3)])



- ## Boxplot (presque toujours OK)

Le boxplot par groupe est un autre moyen de comparer visuellement les distributions des temps de vol pour les 4 types d'hélicoptères.

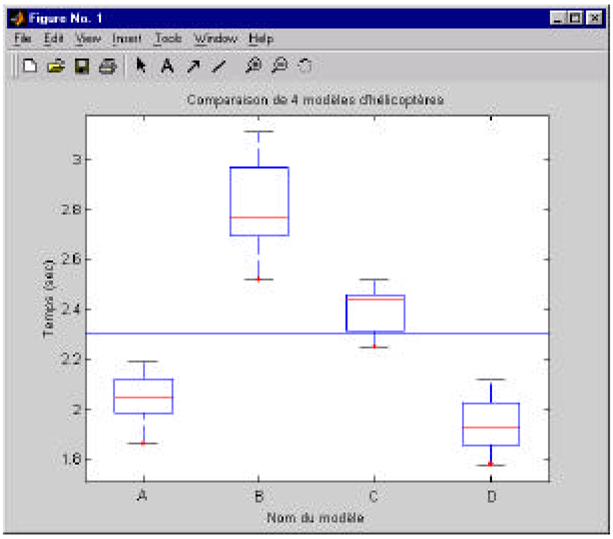
>> boxplot(essai3,[ones(1,8) ones(1,8)*2 ones(1,8)*3 ones(1,8)*4])

Le premier argument de la fonction boxplot est le vecteur de toutes les observations. Le second argument est le vecteur qui définit les groupes de 8 données.
On peut rajouter également une ligne horizontale au niveau de la moyenne comme niveau de référence, ainsi qu'un titre et des labels aux axes.

>> line([0 5],[mean(essai3) mean(essai3)])

>> title('Comparaison de 4 modèles d''hélicoptères')
>> xlabel('Nom du modèle')
>> ylabel('Temps (sec)')

Finalement, de la même manière que pour le diagramme en points, on peut modifier les propriétés de l'axe horizontal et rajouter les lettres A, B, C et D.



- **Statistiques descriptives**

On peut comparer les temps de vol des 4 types d'hélicoptères en résumant chaque série d'observations par des valeurs comme la moyenne et l'écart-type, le nombre d'observations, l'observation la plus petite et celle la plus grande, ainsi que la médiane. Toutes ces valeurs peuvent être regroupées dans une matrice :

```
>> A = essai3(1:8)
A =
   1.9700
   2.0300
   2.1000
   1.8600
   2.0000
   2.1400
   2.0700
   2.1900

>> B = essai3(9:16)
B =
   2.5200
   3.0100
   2.6900
   2.7100
   2.9300
   2.8300
   2.7000
   3.1100

>> C = essai3(17:24)
C =
   2.5200
   2.2500
   2.4700
   2.4400
   2.4400
   2.4500
   2.3600
   2.2700

>> D = essai3(25:32)
D =
   1.8700
   2.1200
   1.8400
   2.0800
   1.9800
   1.9800
   1.7800
   1.8800

>> [ mean(A) std(A) length(A) min(A) max(A) median(A); mean(B) std(B) length(B) min(B) max(B) median(B); mean(C) std(C) length(C) min(C) max(C) median(C); mean(D) std(D) length(D) min(D) max(D) median(D)]
ans =
   2.0450   0.1041   8.0000   1.8600   2.1900   2.0500
   2.8125   0.1947   8.0000   2.5200   3.1100   2.7700
   2.4000   0.0971   8.0000   2.2500   2.5200   2.4400
   1.9413   0.1191   8.0000   1.7800   2.1200   1.9300
```

On obtient donc la table suivante après l'avoir transformé en Word :

	Moyenne	Ecart-type	N	Minimum	Maximum	Médiane
A	2.0450	0.1041	8.0000	1.8600	2.1900	2.0500
B	2.8125	0.1947	8.0000	2.5200	3.1100	2.7700
C	2.4000	0.0971	8.0000	2.2500	2.5200	2.4400
D	1.9413	0.1191	8.0000	1.7800	2.1200	1.9300

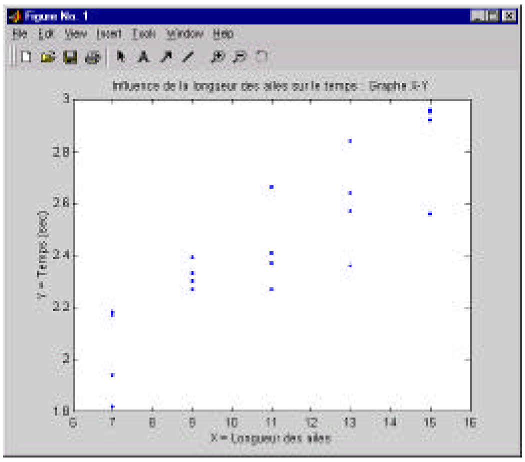
4. Essai4 – Etude de l'effet de la longueur des ailes sur le temps de vol

Cinq variantes de l'hélicoptère sont testées avec des longueurs d'ailes différentes (7, 9, 11, 13 et 15). Chacun est lancé 4 fois. Nous allons voir comment obtenir avec MATLAB un graphe X-Y pour visualiser le lien entre 2 variables et le coefficient de corrélation de Pearson pour mesurer l'intensité d'un lien linéaire entre 2 variables (*cfr* paragraphe 3.4).

· Graphe X-Y

Le graphe X-Y permet de visualiser le lien entre les 2 variables X et Y. On l'obtient avec la commande ***plot*** (le premier argument est le vecteur des abscisses, le second argument est le vecteur des ordonnées et le troisième argument le type de points):

```
>> plot(essai4(:,2),essai4(:,3),'b.')
>> title('Influence de la longueur des ailes sur le temps : Graphe X-Y')
>> xlabel('X = Longueur des ailes')
>> ylabel('Y = Temps (sec)')
```



· Coefficient de corrélation linéaire de Pearson

Le coefficient de corrélation linéaire de Pearson mesure la force du lien linéaire entre 2 variables quantitatives. On l'obtient dans Matlab en utilisant la commande ***corrcoef***. Le résultat est une matrice des corrélations : 1 = corrélation de X avec X = corrélation de Y avec Y et 0.8766 = corrélation de X avec Y (ou Y avec X).

```
>> corrcoef(essai4(:,2),essai4(:,3))
ans =
   1.0000    0.8766
   0.8766    1.0000
```

5. Essai3 - Evaluer et comparer les performances de plusieurs produits

Le fichier Essai3 a pour but de comparer les temps de vols de 4 modèles d'hélicoptères (A, B, C et D). Nous allons voir comment les comparer graphiquement par le calcul d'intervalles de confiance sur la moyenne et sur l'écart-type.

• Diagramme en points (§ 6.3)

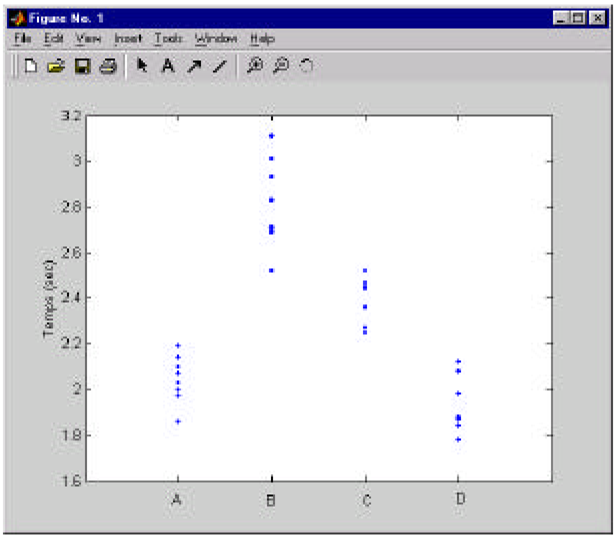
On peut faire un diagramme en points pour comparer les temps de vols des 4 hélicos. Ce type de graphique est bien adapté puisque pour chaque modèle d'hélico il n'y a que 8 mesures.

- On crée le vecteur des abscisses du graphe X-Y à partir d'un vecteur unitaire de longueur 8 (***ones(1,8)***) :

```
>> X = [ones(1,8) ones(1,8)*2 ones(1,8)*3 ones(1,8)*4]
ans =
   1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  3  3  3  3  3  3  3  3  4  4  4  4  4  4  4  4
```

- On utilise la commande plot pour dessiner un graphe X-Y où les abscisses sont dans le premier argument et les ordonnées dans le deuxième argument. On peut en plus spécifier le type et la couleur des points dans le troisième argument.

```
>> plot(X,essai3,'b.')
```



• Calcul des intervalles de confiance sur la moyenne et l'écart-type (§ 6.2.4)

On peut plus rigoureusement comparer les temps de vols des 4 hélicos en calculant un intervalle de confiance pour le temps de vol moyen de chacun. On pourra alors voir pour quels modèles les moyennes sont significativement différentes. On peut également calculer des intervalles de confiance sur les écarts-types pour pouvoir comparer la variabilité des temps de vols des 4 modèles.

- On calcule tout d'abord les moyennes $\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$ avec la fonction ***mean***

```
>> MEAN3 = [mean(essai3(1 :8)); mean(essai3(9 :16)); mean(essai3(17 :24)); mean(essai3(25 :32))]
MEAN3 =
   2.0450
   2.8125
   2.4000
   1.9413
```

- On calcule ensuite les écarts-types $S = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_i - \overline{X})^2}$ avec la fonction ***std***

```
>> STD3 = [std(essai3(1 :8)); std(essai3(9 :16)); std(essai3(17 :24)); std(essai3(25 :32))]
STD3 =
   0.1041
   0.1947
   0.0971
   0.1191
```

- On en déduit les intervalles de confiance sur la moyenne pour chaque modèle en utilisant la fonction ***tinv*** pour obtenir les valeurs tabulées de la t : $\left[\overline{X} - t_{N-1;0.975} \frac{S}{\sqrt{N}}, \overline{X} + t_{N-1;0.975} \frac{S}{\sqrt{N}} \right]$

```
>> LongueurIC = [ tinv(0.975,8-1) * std(essai3(1:8))/sqrt(8) ;  tinv(0.975,8-1) * std(essai3(9:16)) / sqrt(8) ;
                  tinv(0.975,8-1) * std(essai3(17:24))/sqrt(8) ; tinv(0.975,8-1) * std(essai3(25:32)) / sqrt(8) ]
Longueur IC =
   0.0870
   0.1628
   0.0812
   0.0996

>> ICMEAN3 = [ mean(essai3(1:8)) - LongueurIC(1) , mean(essai3(1:8)) + LongueurIC(1)   ;
               mean(essai3(9:16)) - LongueurIC(2) , mean(essai3(9:16)) + LongueurIC(2)   ;
               mean(essai3(17:24)) - LongueurIC(3) , mean(essai3(17:24)) + LongueurIC(3)   ;
               mean(essai3(25:32)) - LongueurIC(4) , mean(essai3(25:32)) + LongueurIC(4) ]
ICMEAN3 =
   1.9580    2.1320
   2.6497    2.9753
   2.3188    2.4812
   1.8417    2.0408
```

- On en déduit également les intervalles de confiance sur l'écart-type pour chaque modèle en utilisant la fonction ***chi2inv*** pour obtenir les valeurs tabulées de la Chi² : $\left[\sqrt{\frac{(N-1)S^2}{c^2_{N-1;0.975}}}, \sqrt{\frac{(N-1)S^2}{c^2_{N-1;0.025}}} \right]$

```
>> Denominateur = [chi2inv(0.975,8-1) ; chi2inv(0.025,8-1)]
Denominateur =
  16.0128
   1.6899

>> Numerateur = [ (8-1) * var(essai3(1:8))   ;   (8-1) * var(essai3(9:16))   ;
                  (8-1) * var(essai3(17:24))   ;   (8-1) * var(essai3(25:32)) ]
Numerateur =
   0.0758
   0.2653
   0.0660
   0.0993

>> ICSTD3 = sqrt( [ Numerateur(1) / Denominateur(1) , Numerateur(1)/Denominateur(2)   ;
                    Numerateur(2) / Denominateur(1) , Numerateur(2)/Denominateur(2)   ;
                    Numerateur(3) / Denominateur(1) , Numerateur(3)/Denominateur(2)   ;
                    Numerateur(4) / Denominateur(1) , Numerateur(4)/Denominateur(2)   ] )
ICSTD3 =
   0.0688    0.2118
   0.1287    0.3963
   0.0642    0.1976
   0.0787    0.2424
```

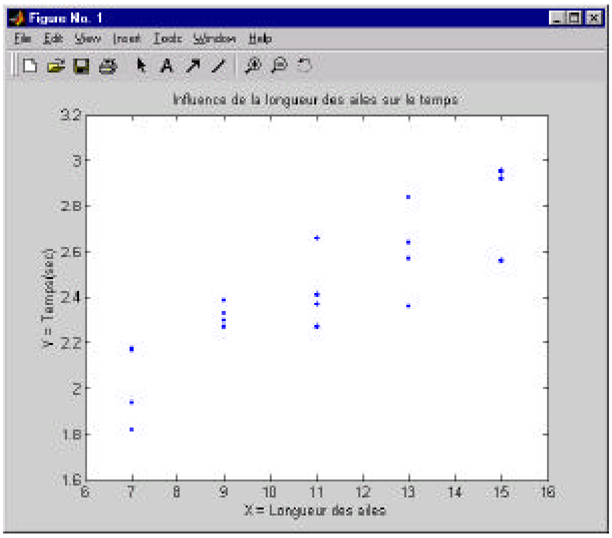
6. Essai4 - Expliquer une variable par une autre : le modèle linéaire simple

On veut voir et modéliser le lien existant entre le temps de vol de l'hélicoptère et la longueur des ailes. C'est le but de l'Essai4 où 5 hélicoptères de longueur d'ailes 7, 9, 11, 13 et 15 ont été lancés 4 fois chacun.

- ## Graphe X-Y (Intro Section 7)

On peut faire un graphe X-Y pour visualiser s'il y a une relation entre les 2 variables et de quelle type elle est. Ici, la relation entre le temps de vol et la longueur des ailes semble simplement linéaire.

```
>> plot(essai4(:,2),essai4(:,3),'b.')
>> xlabel('X = Longueur des ailes')
>> ylabel('Y = Temps(sec)')
>> title('Influence de la longueur des ailes sur le temps')
```



· ## Estimation des paramètres du modèle de régression linéaire simple et intervalles de confiance et de prédiction (§ 7.3)

On utilise la fonction regress de MATLAB pour estimer les paramètres du modèle $Temps = b_0 + b_1 \cdot Ha + e$. Ce sera la même fonction utilisée pour estimer les paramètres d'un modèle polynomial ou d'un modèle de régression multiple.

En utilisant la commande ***help regress***, on peut voir comment utiliser cette fonction dans MATLAB :

```
>> help regress

 REGRESS Multiple linear regression using least squares.
    b = REGRESS(y,X) returns the vector of regression coefficients, b, in the linear model  y = Xb, (X is an nxp matrix, y is the nx1
    vector of observations).
    [B,BINT,R,RINT,STATS] = REGRESS(y,X,alpha) uses the input, ALPHA to calculate 100(1 - ALPHA) confidence intervals for B and
    the  residual vector, R, in BINT and RINT respectively.  The vector STATS contains the R-square statistic along with the F and p
   values for the regression.
    The X matrix should include a column of ones so that the model contains a constant term.  The F and p values are computed under
    the assumption that the model contains a constant term, and they are not correct for models without a constant.  The R-square
    value is the ratio of the regression sum of squares to the total sum of squares.
```

- Il faut donc commencer par définir le vecteur y (variable expliquée = temps de vol = 3° colonne de essai4) et la matrice X (variables explicatives en colonne). Attention, la matrice X contiendra d'abord une colonne de *1* correspondant à l'intercept b_0 et ensuite la colonne des longueurs des ailes (2° colonne de essai4) correspondant à b_1.

```
>> y = essai4(:,3)
y =
   1.9400
   2.3000
   2.6600
   2.3600
   2.9600
   2.1800
   2.3300
   2.3700
   2.6400
   2.5600
   1.8200
   2.3900
   2.4100
   2.5700
   2.9200
   2.1700
   2.2700
   2.2700
   2.8400
   2.9500
```

```
>> X = [ones(20,1), essai4(:,2)]
X =
   1    7
   1    9
   1   11
   1   13
   1   15
   1    7
   1    9
   1   11
   1   13
   1   15
   1    7
   1    9
   1   11
   1   13
   1   15
   1    7
   1    9
   1   11
   1   13
   1   15
```

- En appliquant la fonction ***regress*** à y et X, nous obtenons les estimations des paramètres b_0 et b_1 , ainsi qu'un intervalle de confiance pour chacun $IC_{95\%}(b_0)$ et $IC_{95\%}(b_1)$. Nous pouvons également obtenir une estimation de la variance du terme d'erreur σ² en appliquant la formule $S^2_{Y.X} = \frac{1}{N-2}\sum_{i=1}^{N} e_i^{\ 2}$ puisque les résidus e_i sont également calculés par le logiciel.

```
>> [B,BINT,R,RINT,STATS] = regress(y,X,0.05)
B =
   1.3895
   0.0960
```

⇒ Les paramètres estimés sont donc b_0 = *1.3895* et b_1 = *0.0960*.

```
BINT =
   1.0930   1.6860
   0.0699   0.1221
```

⇒ Pour chaque paramètre, un intervalle de confiance à 95% est donné:
$IC_{95\%}(b_0)$ = [1.0930 ; 1.6860]
$IC_{95\%}(b_1)$ = [0.0699 ; 0.1221]

```
R =
  -0.1215
   0.0465
   0.2145
  -0.2775
   0.1305
   0.1185
   0.0765
  -0.0755
   0.0025
  -0.2695
  -0.2415
   0.1365
  -0.0355
  -0.0675
   0.0905
   0.1085
   0.0165
  -0.1755
   0.2025
   0.1205
```

⇒ Le vecteur R est le vecteur contenant les résidus $e_i = Y_i - b_0 - b_1X_i$.
On peut en déduire une estimation de la variance du terme d'erreur $S^2_{Y.X}$:

```
>> S2residus = sum(R.*R)/(20-2)
S2residus =
   0.0247
```

On peut en déduire une estimation de l'écart-type du terme d'erreur $S_{Y.X}$:

```
>> Sresidus = sqrt(sum(R.*R)/(20-2))
Sresidus =
   0.1572
```

```
RINT =
  -0.4285   0.1855
  -0.2794   0.3724
  -0.0981   0.5271
         . . .
  -0.4943   0.1433
  -0.1075   0.5125
  -0.1866   0.4276
```

⇒ RINT est un vecteur d'intervalles de confiance pour chaque résidus. Ce n'est pas un élément très intéressant...

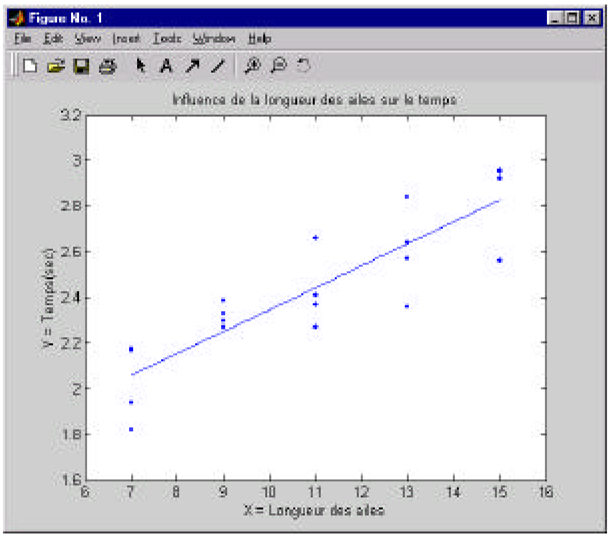
```
STATS =
   0.7684   59.7075    0.0000
```

⇒ Il s'agit du coefficient R², de la valeur de la statistique F et de la p-valeur, 3 éléments **non vus dans ce cours**.

Ajout de la droite de régression sur le graphique X-Y.

- On peut ajouter sur le graphe X-Y la droite de régression grâce à la fonction ***refline*** qui prend comme premier argument la pente de la droite et comme second argument l'intercept.

```
>> plot(essai4(:,2),essai4(:,3),'b.')
>> xlabel('X = Longueur des ailes')
>> ylabel('Y = Temps(sec)')
>> title('Influence de la longueur des ailes sur le temps')
>> refline(B(2),B(1))
```



Prédiction.

Pour faire une prédiction, on utilise le modèle estimé. On peut également calculer un intervalle de confiance sur la réponse moyenne ou un intevalle de prédiction selon ce à quoi on s'intéresse.

Je prédis ce temps de vol à l'aide du modèle estimé à 2.3495 sec:

```
>> Ypredit = B(1)+B(2)*10
Ypredit =
   2.3495
```

Ensuite, par exemple, je m'intéresse tout d'abord au temps de vol moyen d'un hélicoptère dont les ailes mesurent 10 cm. Je calcule donc un **intervalle de confiance** sur la réponse moyenne en appliquant la formule $(b_0 + b_1 X_0) \pm t_{N-2,0.975} S_{\hat{Y}_0}$ où $S_{\hat{Y}_0} = S_{Y.X}\sqrt{\frac{1}{N} + \frac{(X_0 - \overline{X})^2}{(N-1)S_X^2}}$: [2.2712 ; 2.4278]

```
>> LongueurIC = tinv(0.975,20-2) * Sresidus
                    * sqrt( 1/20 + ((10-mean(essai4(:,2)))^2 ) / ( (20-1) * var(essai4(:,2)) ) )
LongueurIC =
   0.0783

>> ICmoyenne = Ypredit + [-LongueurIC  LongueurIC]
ICmoyenne =
   2.2712   2.4278
```

Si je m'intéresse ensuite au temps de vol lors d'un lancé en particulier d'un hélicoptère ayant des ailes de 10 cm, je calcule un **intervalle de prédiction** à 95% pour cette valeur en appliquant la formule

$(b_0 + b_1 X_0) \pm t_{N-2;0.975} s_{Y_P}$ où $s_{Y_p} = s_{Y.X}\sqrt{1 + \frac{1}{N} + \frac{(X_0 - \overline{X})^2}{(N-1)s_X^2}}$: [2.0101 ; 2.6889]

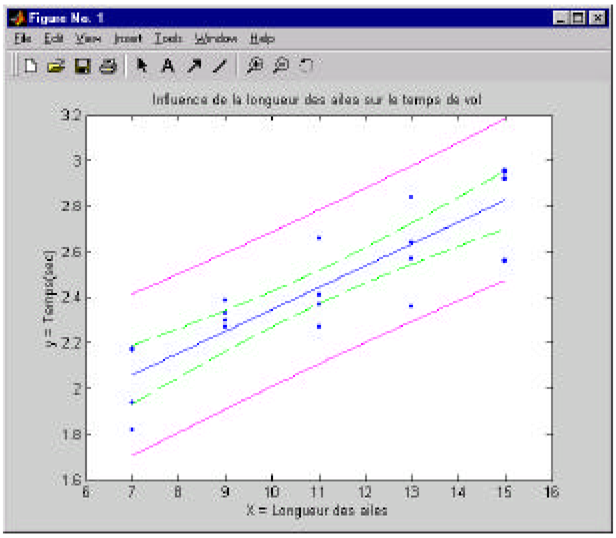
```
>> LongueurIP = tinv(0.975,20-2) * Sresidus
                    * sqrt( 1 + 1/20 + ((10-mean(essai4(:,2)))^2) / ( (20-1)*var(essai4(:,2)) ) )
LongueurIP =
   0.3394

>> IP = Ypredit + [-LongueurIP  LongueurIP]
IP =
   2.0101   2.6889
```

- ## Ajout des intervalles de confiance et de prédiction sur le graphe X-Y.

On peut ajouter sur le graphe X-Y la droite de régression grâce à la fonction ***refline*** mais aussi les intevalles de confiance (tirets verts) et les intervalles de prédiction (lignes mauves) tout le long de la droite en appliquant les formules de ces intervalles de manière vectorielle à une fine grille de points entre 7 et 15, ***vecX***. La commande ***hold on*** permet de dessiner sur un même graphique plusieurs ***plot***s.

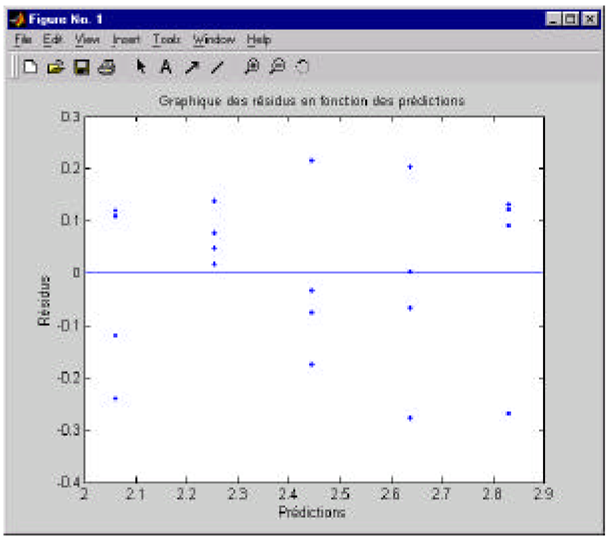
```
>> vecX = (7:0.05:15)
>> Ypredit = B(1) + B(2)*vecX
>> LongueurIC = tinv(0.975,20-2) * Sresidus
                    * sqrt( 1/20 + (( vecX - mean(essai4(:,2)) ).^2) / ((20-1) * var(essai4(:,2))) )
>> LongueurIP = tinv(0.975,20-2) * Sresidus
                    * sqrt( 1+ 1/20 + (( vecX - mean(essai4(:,2)) ).^2) / ((20-1) * var(essai4(:,2))) )
>> hold on
>> plot( essai4(:,2),essai4(:,3),'b.')
>> plot(vecX , Ypredit - LongueurIC , 'g--' )
>> plot(vecX , Ypredit + LongueurIC , 'g--' )
>> plot(vecX , Ypredit - LongueurIP , 'm-' )
>> plot(vecX , Ypredit + LongueurIP , 'm-' )
>> refline(B(2),B(1))
>> xlabel('X = Longueur des ailes')
>> ylabel('y = Temps(sec)')
>> title('Influence de la longueur des ailes sur le temps de vol')
```



- ## Graphique des résidus.

Finalement il est judicieux de faire un graphique des résidus afin de valider le modèle. Il s'agit d'un simple graphique X-Y où les X sont les prédictions $b_0 + b_1.Ha$ et les Y les résidus e_i .

```
>> plot(B(1)+B(2)*essai4(:,2),R,'b.')
>> refline(0,0)
>> xlabel('Prédictions')
>> ylabel('Résidus')
>> title('Graphique des résidus en fonction des prédictions')
```



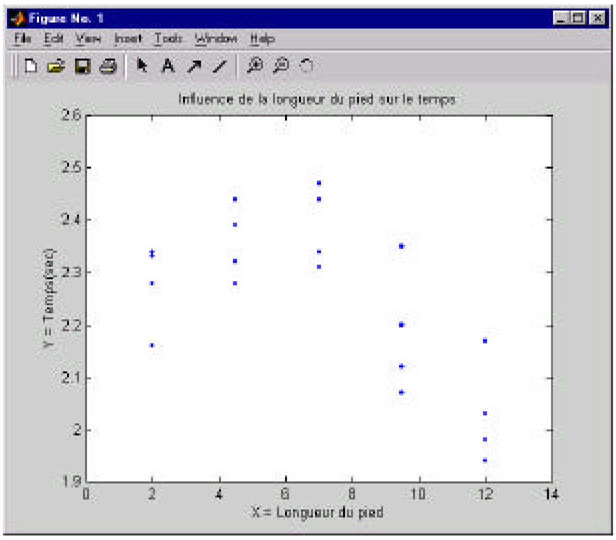
7. Essai5 - Expliquer une variable par une autre : la régression polynomiale

On veut voir et modéliser le lien existant entre le temps de vol de l'hélicoptère et la longueur du pied. C'est le but de l'Essai5 où 5 hélicoptères longueur de pied 2, 4.5, 7, 9.5 et 12 ont été lancés 4 fois chacun.

- **Graphe X-Y (Intro Section 7)**

On peut faire un graphe X-Y pour visualiser s'il y a une relation entre les 2 variables et de quelle type elle est. Ici, la relation semble quadratique.

```
>> plot(essai5(:,2),essai5(:,3),'b.')
>> xlabel('X = Longueur du pied')
>> ylabel('Y = Temps(sec)')
>> title('Influence de la longueur du pied sur le temps')
```



- **Estimation des paramètres du modèle de régression polynomiale (§ 7.3)**

On utilise à nouveau la fonction regress de MATLAB pour estimer les différents paramètres du modèle $Temps = b_0 + b_1 \cdot H_p + b_2 \cdot H_p^2 + e$.

- Il faut commencer par définir le vecteur *y* (variable expliquée = temps de vol = 3° colonne de essai5) et la matrice *X* (variables explicatives en colonne). Attention, la matrice *X* contiendra d'abord une colonne de *1* correspondant au terme constant b_0. Ensuite la colonne des longueurs des pieds (2° colonne de essai5) correspondant à b_1 et finalement la colonne des carrés des longueurs des pieds (carrés de la 2° colonne de essai5).

```
>> y = essai5(:,3)
y =
   2.3400
   2.2800
   2.3400
   2.2000
   1.9800
   2.1600
   2.3900
   2.4400
   2.0700
   2.0300
   2.2800
   2.4400
   2.3100
   2.3500
   2.1700
   2.3300
   2.3200
   2.4700
   2.1200
   1.9400
```

```
>> X =[ones(20,1) essai5(:,2) essai5(:,2).*essai5(:,2)]
X =
   1.0000    2.0000    4.0000
   1.0000    4.5000   20.2500
   1.0000    7.0000   49.0000
   1.0000    9.5000   90.2500
   1.0000   12.0000  144.0000
   1.0000    2.0000    4.0000
   1.0000    4.5000   20.2500
   1.0000    7.0000   49.0000
   1.0000    9.5000   90.2500
   1.0000   12.0000  144.0000
   1.0000    2.0000    4.0000
   1.0000    4.5000   20.2500
   1.0000    7.0000   49.0000
   1.0000    9.5000   90.2500
   1.0000   12.0000  144.0000
   1.0000    2.0000    4.0000
   1.0000    4.5000   20.2500
   1.0000    7.0000   49.0000
   1.0000    9.5000   90.2500
   1.0000   12.0000  144.0000
```

- En appliquant la fonction ***regress*** à y et X, nous obtenons les estimations des paramètres b_0 , b_1 et b_2 , ainsi qu'un intervalle de confiance pour chacun $IC_{95\%}(b_0)$, $IC_{95\%}(b_1)$ et $IC_{95\%}(b_2)$. Nous pouvons également obtenir une estimation de la variance du terme d'erreur σ² en appliquant la formule $S_{Y.X}^2 = \frac{1}{N-p}\sum_{i=1}^{N} e_i^2$ puisque les résidus e_i sont également calculés par le logiciel.

```
>> [B,BINT,R] = regress(y,X,0.05)

B =
   2.1398
   0.0865
  -0.0081
BINT =
   1.9593    2.3202
   0.0268    0.1462
  -0.0123   -0.0039

R =
   0.0596
  -0.0853
  -0.0091
  -0.0318
  -0.0334
  -0.1204
   0.0247
   0.0909
  -0.1618
   0.0166
  -0.0004
   0.0747
  -0.0391
   0.1182
   0.1566
   0.0496
  -0.0453
   0.1209
  -0.1118
  -0.0734
```

⇒ Les paramètres estimés sont donc b_0 = *2.1398* , b_1 = *0.0865* et b_2 = *-0.0081*.

⇒ Pour chaque paramètre, un intervalle de confiance à 95% est donné:
$IC_{95\%}(b_0)$ = [1.9593 ; 2.3202]
$IC_{95\%}(b_1)$ = [0.0268 ; 0.1462]
$IC_{95\%}(b_2)$ = [-0.0123 ; -0.0039]

⇒ Le vecteur R est le vecteur contenant les résidus $e_i = Y_i - b_0 - b_1X_i$.
On peut en déduire une estimation de la variance du terme d'erreur $S_{Y.X}^2$:

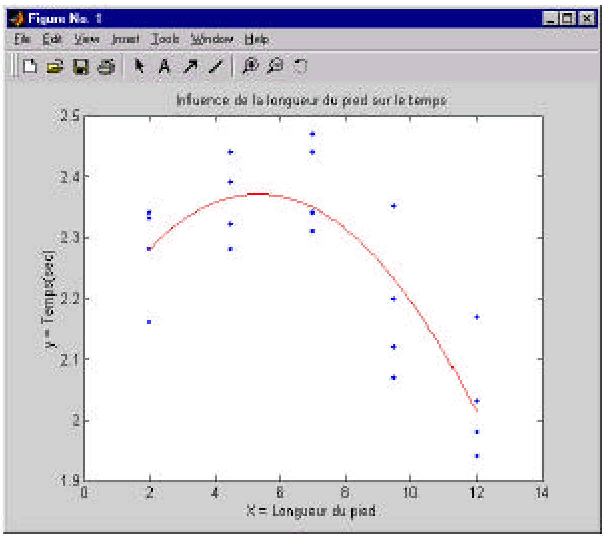
```
>> S2residus = sum(R.*R)/(20-3)
S2residus =
   0.0086
```

On peut en déduire une estimation de l'écart-type du terme d'erreur $S_{Y.X}$:

```
>> Sresidus = sqrt(sum(R.*R)/(20-3))
Sresidus =
   0.0925
```

- On peut ajouter sur le graphe X-Y la courbe correspondant au modèle estimé. On utilise à nouveau une grille de points entre 2 et 13, ***vecX***. On trace d'abord les points bleus sur le graphe et les 3 derniers arguments tracent en rouge la courbe du modèle estimé.

```
>> vecX = 2:0.01:12
>> hold on
>> plot( essai5(:,2) , essai5(:,3) , 'b.' )
>> plot( vecX , B(1) + B(2)* vecX + B(3)* vecX.* vecX , 'r-')
>> xlabel('X = Longueur du pied')
>> ylabel('y = Temps(sec)')
>> title('Influence de la longueur du pied sur le temps')
```



8. Essai7 - Ajuster une surface de réponse à des données expérimentales

On veut voir et modéliser le lien existant entre le temps de vol de l'hélicoptère et la longueur du pied et la longueur des ailes pour des hélicoptères aux coins coupés. C'est le but de l'Essai7 qui reprends les temps de vols du plan factoriel 3² répété 4 fois.

• Estimation des paramètres du modèle de régression multiple (§ 10.2)

On utilise à nouveau la fonction regress de MATLAB pour estimer les différents paramètres du modèle $Temps = b_0 + b_1 \cdot H_{pSTD} + b_2 \cdot H_{aSTD} + b_3 \cdot H_{pSTD}^2 + b_4 \cdot H_{aSTD}^2 + b_5 \cdot H_{aSTD} \cdot H_{pSTD} + e$.

- Il faut commencer par définir le vecteur *y* (variable expliquée = temps de vol = 6° colonne de essai7) et la matrice *X* (variables explicatives STANDARDISEES en colonne).

```
>> HpSTD = (  essai7(:,1) - (12+2 ) / 2  )  /  (  (12-2) / 2  )
>> HaSTD = (  essai7(:,2) - ( 15+7 ) / 2)  /  (  (15-7) / 2  )
```

```
>> y = essai7(:,6)       >> X = [ ones(36,1)   HpSTD   HaSTD   HpSTD.^2   HaSTD.^2   HpSTD.*HaSTD ]
y =                      X =
   2.6600                    1   -1   -1    1    1    1
   3.3000                    1   -1    0    1    0    0
   4.0300                    1   -1    1    1    1   -1
   2.7700                    1    0   -1    0    1    0
   3.1500                    1    0    0    0    0    0
   4.0500                    1    0    1    0    1    0
   ...                                 . . .
   2.3400                    1    0   -1    0    1    0
   3.0100                    1    0    0    0    0    0
   3.9200                    1    0    1    0    1    0
   1.9700                    1    1   -1    1    1   -1
   2.5000                    1    1    0    1    0    0
   2.8300                    1    1    1    1    1    1
```

- Ensuite on applique la fonction ***regress*** à y et X.

```
>> [B,BINT,R] = regress(y,X,0.05)
B =
   3.2428
  -0.4808
   0.6296
  -0.4167
  -0.0079
  -0.1431
```

⇒ Les paramètres estimés sont donc
$b_0 = 0.7570$, $b_1 = 0.2159$, $b_2 = 0.2184$, $b_3 = -0.0167$, $b_4 = -0.0005$ et $b_5 = -0.0072$.

```
BINT =
   3.1125    3.3730
  -0.5522   -0.4095
   0.5583    0.7009
  -0.5402   -0.2931
  -0.1315    0.1156
  -0.2305   -0.0558
```

⇒ Pour chaque paramètre, un intervalle de confiance à 95% est donné.

```
R =
   0.1337
  -0.0069
  -0.0417
   . . .
   0.1191
   0.1547
   0.0062
```

⇒ Le vecteur R contient les résidus. On peut en déduire une estimation de la variance du terme d'erreur $S_{Y.X}^2$:

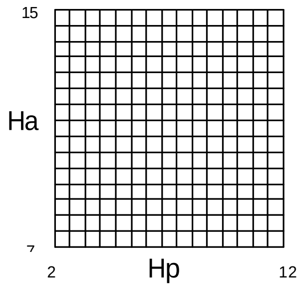
```
>> S2residus = sum(R.*R)/(36-6)
S2residus =
   0.0293
```

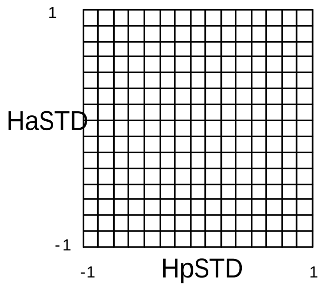
On peut en déduire une estimation de l'écart-type du terme d'erreur $S_{Y.X}$:

```
>> Sresidus = sqrt(sum(R.*R)/(36-6))
Sresidus =
   0.1711
```

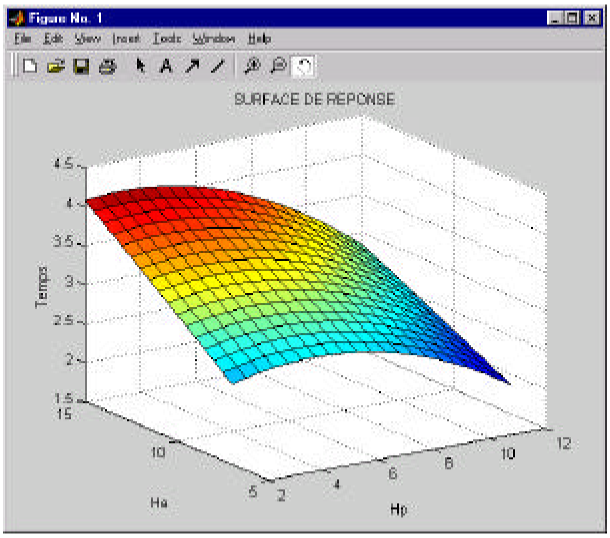
- **Représentation graphique du modèle (§ 10.3)**

- On peut commencer par représenter la *surface de réponse* c-à-d le temps de vol estimé en fonction des longueurs des ailes et du pied. Pour cela, on crée d'abord une grille ***[Hp,Ha]*** de points sur le domaine expérimentale [2;12]×[7;15] ainsi qu'une grille ***[HpSTD,HaSTD]*** sur le domaine standardisé[-1;1] ×[-1;1] avec la fonction ***meshgrid***. Ensuite en chaque point du quadrillage ***[HpSTD,HaSTD]*** on utilise le modèle estimé pour prédire le temps T. Finalement, on effectue le graphe en 3 dimensions Hp-Ha-T en utilisant la fonction ***surf***. Ce graphe peut tourner et être modifié dans la fenêtre graphique.



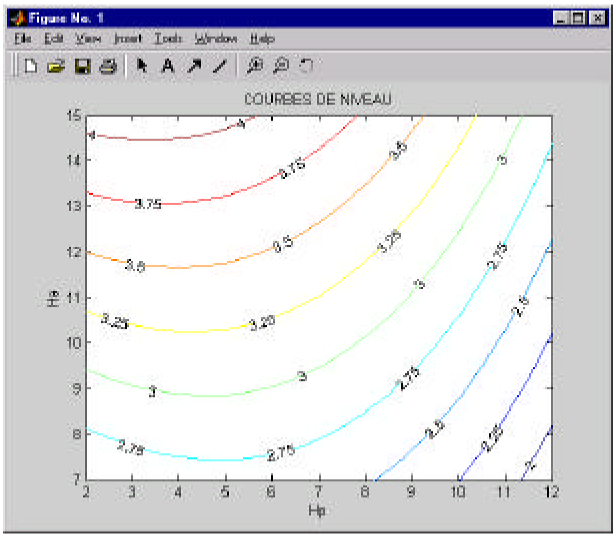



```
>> [Hp,Ha] = meshgrid( 2:.5:12 , 7:.5:15 )
>> [HpSTD,HaSTD] = meshgrid( ((2:.5:12)-7)/5 , ((7:.5:15)-11)/4 )
>> T = B(1) + B(2)*HpSTD + B(3)*HaSTD + B(4)*HpSTD.^2 + B(5)*HaSTD.^2 + B(6)*HpSTD.*HaSTD
>> surf(Hp,Ha,T)
```



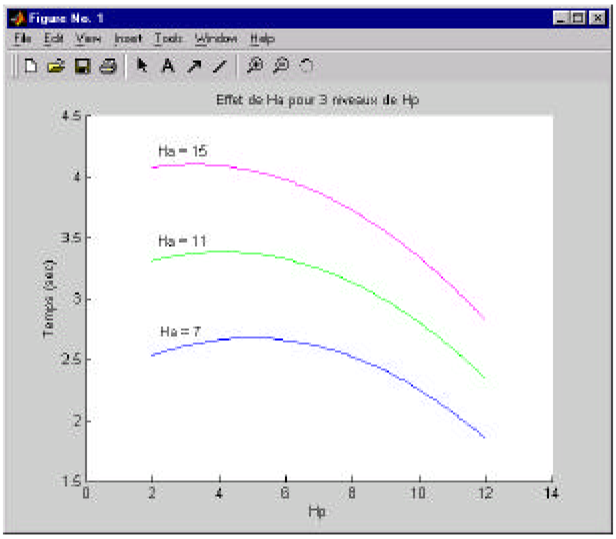
- On peut également représenter le temps de vol en fonction des longueurs des ailes et du pied par des *courbes de niveau*. On obtient alors un graphique en 2 dimensions facilement interprétable et utile pour trouver un optimum. Pour cela, on utilise le quadrillage déjà crée précédemment HpSTD-HaSTD et le temps T estimé en chaque point du quadrillage. Mais cette fois-ci on applique la fonction ***contour*** en utilisant comme arguments Hp, Ha et T, ainsi qu'un vecteur reprenant les niveaux des courbes qui nous intéressent : 2, 2.25, 2.5, 2.75, 3, 3.25, 3.5, 3.75 et 4. Finalement, on ajoute les valeurs du temps de vol sur les courbes en utilisant la fonction ***clabel***.

```
>> [c,h]=contour(Hp,Ha,T,[2 2.25 2.5 2.75 3 3.25 3.5 3.75 4]);
>> clabel(c,h)
```



- On peut finalement visualiser les effets d'interaction par des *graphes d'interaction*.
  Ce graphe ci-dessous représente l'évolution du temps de vol estimé en fonction de la longueur du pied pour 3 niveaux de longueur des ailes Ha=7, 11 ou 15. Inverser le rôle de Ha et Hp est biensur équivalent.

```
>> Hp = (2:0.05:12)
>> HpSTD = (Hp-7)/5
>> hold on
>> plot( Hp , B(1) + B(2)*HpSTD + B(3)*(-1) + B(4)*HpSTD.^2 + B(5)*(-1)^2 + B(6)*(-1)*HpSTD , 'b-' )
>> plot( Hp , B(1) + B(2)*HpSTD + B(3)*0 + B(4)*HpSTD.^2 + B(5)*0^2 + B(6)*0*HpSTD , 'g-' )
>> plot( Hp , B(1) + B(2)*HpSTD + B(3)*1 + B(4)*HpSTD.^2 + B(5)*1^2 + B(6)*1*HpSTD , 'm-' )
>> xlabel('Hp')
>> ylabel('Temps')
>> title('Effet de Ha pour 3 niveaux de Hp')
```



- **Recherche de conditions optimales et d'un intervalle de prédiction (§ 10.4)**

- On conclut en analysant le modèle et les graphiques ci-dessus que le temps de vol sera maximal pour un hélicoptère au pied coupé quand Ha = 15 (HaSTD = 1) et Hp = 3.3 (HpSTD = -0.74). Pour ces conditions optimales, le modèle prédit un temps de vol de 4.1 secondes:

```
>> Tpredit = B(1) + B(2)*(-0.74) + B(3)*1 + B(4)*(-0.74)^2 + B(5)*1^2 + B(6)*1*(-0.74)
Tpredit =
   4.0980
```

- On peut finalement donner un intevalle de confiance autour de cette valeur en appliquant la formule $\hat{Y} \pm t_{N-p,0.975} S_P$ avec $S_P = S_{Y.X}\sqrt{1+\mathbf{X_0}'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X_0}}$: [3.72 , 4.47]

```
>> X0 = [1; -0.74; 1; (-0.74)^2; 1^2; 1*(-0.74)]
X0 =
   1.0000
  -0.7400
   1.0000
   0.5476
   1.0000
  -0.7400
>> Longueur = tinv(0.975,36-6) * Sresidu s* sqrt(  + X0'*((X'*X)^(-1))*X0)
Longueur =
   0.3735
>> IP = Tpredit +[-Longueur,Longueur]
IP =
   3.7245    4.4715
```

9. Principales fonctions de la toolbox STAT…

1. Distributions

Parameter estimation

betafit - Beta parameter estimation
binofit - Binomial parameter estimation
expfit - Exponential parameter estimation
gamfit - Gamma parameter estimation
mle - Maximum likelihood estimation (MLE)
nbinfit - Negative binomial parameter estimation
normfit - Normal parameter estimation
poissfit - Poisson parameter estimation
unifit - Uniform parameter estimation
weibfit - Weibull parameter estimation

Probability density functions (pdf)

betapdf - Beta density
binopdf - Binomial density
chi2pdf - Chi square density
exppdf - Exponential density
fpdf - F density
gampdf - Gamma density
geopdf - Geometric density
hygepdf - Hypergeometric density
lognpdf - Lognormal density
nbinpdf - Negative binomial density
normpdf - Normal (Gaussian) density
pdf - Density function for a specified distribution
poisspdf - Poisson density
tpdf - T density
unidpdf - Discrete uniform density
unifpdf - Uniform density
weibpdf - Weibull density

Cumulative Distribution functions (cdf)

betacdf - Beta cdf
binocdf - Binomial cdf
cdf - Specified cumulative distribution function
chi2cdf - Chi square cdf
ecdf - Empirical cdf (Kaplan-Meier estimate)
expcdf - Exponential cdf. fcdf - F cdf
gamcdf - Gamma cdf
geocdf - Geometric cdf
hygecdf - Hypergeometric cdf
logncdf - Lognormal cdf
nbincdf - Negative binomial cdf
normcdf - Normal (Gaussian) cdf
poisscdf - Poisson cdf. raylcdf - Rayleigh cdf
tcdf - T cdf. unidcdf - Discrete uniform cdf
unifcdf - Uniform cdf
weibcdf - Weibull cdf.

Critical Values of Distribution functions (Percentiles)

betainv - Beta inverse cumulative distribution function
binoinv - Binomial inverse cumulative distribution function
chi2inv - Chi square inverse cumulative distribution function
expinv - Exponential inverse cumulative distribution function
finv - F inverse cumulative distribution function
gaminv - Gamma inverse cumulative distribution function
geoinv - Geometric inverse cumulative distribution function
hygeinv - Hypergeometric inverse cumulative distribution function
icdf - Specified inverse cdf
logninv - Lognormal inverse cumulative distribution function
nbininv - Negative binomial inverse distribution function
norminv - Normal (Gaussian) inverse cumulative distribution function
poissinv - Poisson inverse cumulative distribution function
tinv - T inverse cumulative distribution function
unidinv - Discrete uniform inverse cumulative distribution function
unifinv - Uniform inverse cumulative distribution function
weibinv - Weibull inverse cumulative distribution function

Random Number Generators

betarnd - Beta random numbers
binornd - Binomial random numbers
chi2rnd - Chi square random numbers
exprnd - Exponential random numbers
frnd - F random numbers
gamrnd - Gamma random numbers
geornd - Geometric random numbers
hygernd - Hypergeometric random numbers
lognrnd - Lognormal random numbers
mvnrnd - Multivariate normal random numbers
mvtrnd - Multivariate t random numbers
nbinrnd - Negative binomial random numbers
normrnd - Normal (Gaussian) random numbers
poissrnd - Poisson random numbers
random - Random numbers from specified distribution
trnd - T random numbers
unidrnd - Discrete uniform random numbers
unifrnd - Uniform random numbers
weibrnd - Weibull random numbers
wishrnd - Wishart random matrix

Statistics

betastat - Beta mean and variance
binostat - Binomial mean and variance
chi2stat - Chi square mean and variance
expstat - Exponential mean and variance
fstat - F mean and variance
gamstat - Gamma mean and variance
geostat - Geometric mean and variance
hygestat - Hypergeometric mean and variance
lognstat - Lognormal mean and variance
nbinstat - Negative binomial mean and variance
normstat - Normal (Gaussian) mean and variance
poisstat - Poisson mean and variance
tstat - T mean and variance
unidstat - Discrete uniform mean and variance

unifstat - Uniform mean and variance
weibstat - Weibull mean and variance

Likelihood functions

betalike - Negative beta log-likelihood
gamlike - Negative gamma log-likelihood
nbinlike - Negative likelihood for negative binomial distribution
normlike - Negative normal likelihood
weiblike - Negative Weibull log-likelihood

2. Descriptive Statistics

corrcoef - Correlation coefficient
cov - Covariance crosstab - Cross tabulation
geomean - Geometric mean
grpstats - Summary statistics by group
harmmean - Harmonic mean
iqr - Interquartile range
kurtosis - Kurtosis
mean - Sample average (in matlab toolbox)
median - 50th percentile of a sample
moment - Moments of a sample
prctile - Percentiles
range - Range
skewness - Skewness
std - Standard deviation (in matlab toolbox)
tabulate - Frequency table
var - Variance (in matlab toolbox)

3. Linear Models

anova1 - One-way analysis of variance
anova2 - Two-way analysis of variance
dummyvar - Dummy-variable coding
friedman - Friedman's test (nonparametric two-way anova)
glmfit - Generalized linear model fitting
glmval - Evaluate fitted values for generalized linear model
kruskalwallis - Kruskal-Wallis test (nonparametric one-way anova)
leverage - Regression diagnostic
multcompare - Multiple comparisons of means and other estimates
polyfit - Least-squares polynomial fitting
polyval - Predicted values for polynomial functions
rcoplot - Residuals case order plot
regress - Multivariate linear regression
rstool - Multidimensional response surface visualization (RSM)

4. Hypothesis Tests

ranksum - Wilcoxon rank sum test (independent samples)
signrank - Wilcoxon sign rank test (paired samples)
signtest - Sign test (paired samples)
ztest - Z test
ttest - One sample t test
ttest2 - Two sample t test

Distribution Testing

kstest - Kolmogorov-Smirnov test for one sample
kstest2 - Kolmogorov-Smirnov test for two samples
lillietest - Lilliefors test of normality

Nonparametric Functions

friedman - Friedman's test (nonparametric two-way anova)
kruskalwallis - Kruskal-Wallis test (nonparametric one-way anova)
ksdensity - Kernel smoothing density estimation
ranksum - Wilcoxon rank sum test (independent samples)
signrank - Wilcoxon sign rank test (paired samples)
signtest - Sign test (paired samples)

5. Statistical Plotting

boxplot - Boxplots of a data matrix (one per column)
cdfplot - Plot of empirical cumulative distribution function
fsurfht - Interactive contour plot of a function
gplotmatrix - Matrix of scatter plots grouped by a common variable
gscatter - Scatter plot of two variables grouped by a third
lsline - Add least-square fit line to scatter plot
normplot - Normal probability plot
qqplot - Quantile-Quantile plot
refline - Reference line
weibplot - Weibull probability plot

6. Utility Functions

combnk - Enumeration of all combinations of n objects k at a time
tiedrank - Compute ranks of sample, adjusting for ties
zscore - Normalize matrix columns to mean 0, variance 1