Principe

Soit deux grandeurs x et y. On suppose que y dépend de x sous la forme d'une fonction $y = f(x)$, le plus souvent :

— une fonction linéaire $f(x) = a.x$;

— ou une fonction affine $f(x) = a.x + b$.

a et b sont les paramètres de la régression que l'on cherche à déterminer. On dispose pour ce faire d'un ensemble de mesures de couples de valeurs $\{(x_i, y_i)\}$ pour les grandeurs x et y.

En faisant l'hypothèse que les écarts données-modèle $\varepsilon_i = y_i - f(x_i)$ (nommés résidus) ont une distribution normale, on montre que le meilleur estimateur des paramètres est obtenue par la **méthode des moindres carrés** : les paramètres sont tels que $\sum_i \varepsilon_i^2$ est minimale.

On obtient les résultats suivants :

— pour une fonction linéaire $a = \dfrac{\sum_i x_i y_i}{\sum_i x_i^2}$;

— pour une fonction affine $a = \dfrac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2}$ et $b = \bar{y} - a.\bar{x}$ où $\bar{\cdot}$ désigne la moyenne.

Remarque : dans le cas d'une fonction affine la qualité de la modélisation est évaluée par le ***coefficient de détermination*** *r^2 qui est compris entre 0 et 1. Une valeur proche de 1 correspond à un très bon accord entre données et modèle, une valeur proche de 0 à un très mauvais accord. Cette évaluation n'est cependant pas parfaite, on préférera une approche utilisant les incertitudes-types si on dispose de leur valeur (voir au verso).*

Mise en œuvre pratique

Sous Python

On entre les données dans des arrays *numpy* afin d'utiliser les fonctionnalités de la bibliothèque.

— pour une fonction linéaire $f(x) = a.x$:

```
a=sum(x*y)/sum(x^2)
```

— pour une fonction affine $f(x) = a.x + b$:

```
p = np.polyfit(x,y,1)
a = p[0]
b = p[1]
```

Calculatrice Numworks

❏ Dans le menu « Régressions », onglet « Données », entrer les couples de données dans les colonnes X1 et Y1 ;

❏ Le graphe des y_i en fonction des x_i apparaît sous l'onglet « Graphique ». Cliquer sur « Régression » puis choisir le type de modèle.

❏ Le graphe du modèle calculé est affiché. Pour connaître son équation ainsi que le coefficient de détermination, cliquer à nouveau sur « Régression ». On retrouve les valeurs sous l'onglet « Stats ».

Calculatrice TI (fonction affine)

❏ Accéder à l'éditeur avec la touche « stats » puis le menu « ÉDIT » ; entrer les couples de valeurs dans les colonnes L_1 et L_2.

❏ Appuyer sur la touche « Stats » à nouveau, dans le menu « CALC » choisir l'option 4 « RégLin (ax+b) ».

❏ Après avoir validé plusieurs fois avec « Entrée », les paramètres de la régression linéaire s'affichent, ainsi que le coefficient de détermination.

❏ Apputer sur « GRAPH » pour faire tracer le graphe du modèle.

Calculatrice Casio (fonction affine)

❏ Accéder au menu « STAT » et entrer les couples de données dans les listes 1 et 2.

❏ Cliquer « CALC » (F1), puis « X » (F2), puis « ax+b » (F1) : les paramètres de la régression linéaire s'affichent, ainsi que le coefficient de détermination.

❏ Apputer sur F6 pour faire tracer le graphe du modèle.

Incertitudes

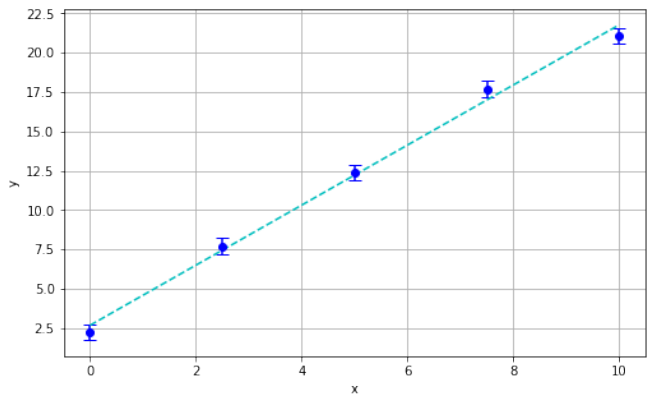
La méthode des moindres carrés ne tient pas compte des incertitudes portant sur les mesures. Il existe d'autres méthodes, comme celle dite du χ^2, qui les prennent en compte : *Regressi* l'utilise pour ses modélisations.

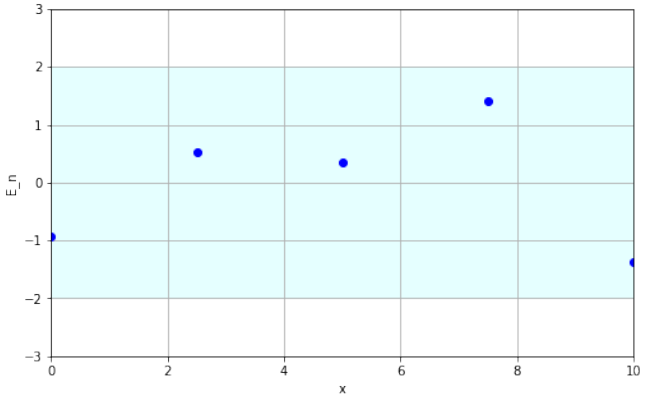
On peut néanmoins tout de même utiliser la régression linéaire, les incertitudes permettant de valider le modèle obtenu à la place du coefficient de corrélation.

Visualisation graphique

Afin de vérifier la compatibilité entre donnés et modèle, on trace un graphe comportant les couples de mesures avec des **barres d'erreur** de demi-longueur égale à l'incertitude-type. Puis on trace la droite de modélisation.Il y a compatibilité si la droite passe à moins de deux fois la barre d'erreur de chaque point (graphe de gauche).

Remarque : Si les barres d'erreurs sont très petites, on peut aussi calculer les écarts normalisés entre $ax_i + b$ et y_i pour chaque couple de points. Ils doivent tous être inférieurs à 2 pour qu'il y ait compatibilité (graphe de droite).





Utilisation d'une simulation Monte-Carlo

Les données utilisées étant incertaines, les paramètres du modèle (linéaire ou affine) sont eux-mêmes incertains.

Pour déterminer la variabilité de ces paramètres, on simule un grand nombre d'expériences par un algorithme *Monte-Carlo* (voir fiche méthode). Dans chaque expérience :

- on simule une série de donnés $\{x_i, y_i\}$ tirées aléatoirement selon des lois de distribution d'écart-type les incertitudes-types des données ;
- on calcule les paramètres de la modélisation pour ces données simulées.

On obtient ainsi un ensemble de valeurs pour chaque paramètre de modélisation, dont on extrait la valeur moyenne et l'écart-type, qui représente l'incertitude-type pour ce paramètre.

Voici un exemple de code à compléter :

```
# Initialisation
Nsim =    # Nombre de simulations souhaitées
a_MC = np.zeros(N)    #Tableau vide pour les valeurs de la pente
b_MC = np.zeros(N)    #Tableau vide pour les valeurs de l'ordonnée à l'origine
Nmes =      #Nombre de points de mesure

#Création d'une boucle for permettant de générer les N expériences virtuelles
for i in range(Nsim):
    #Tirage aléatoire des Nmes points de mesure simulés
    x_MC =    #Loi de probabilité à indiquer
    y_MC =    #Loi de probabilité à indiquer
    #Régression linéaire pour l'expérience virtuelle en cours
    p_MC = np.polyfit(x_MC, y_MC, 1)
    a_MC[i] = p_MC[0]
    b_MC[i] = p_MC[1]
```

Il ne reste plus qu'à extraire des tableaux a_{MC} et b_{MC} les valeurs moyennes et les écarts-types.