

# **MS BGD**

## **MDI 720 : Statistiques**

**Joseph Salmon**

<http://josephsalmon.eu>

Télécom Paristech, Institut Mines-Télécom

# Plan

Introduction : visualisation / Python

Moindres carrés uni-dimensionnels

- Modélisation

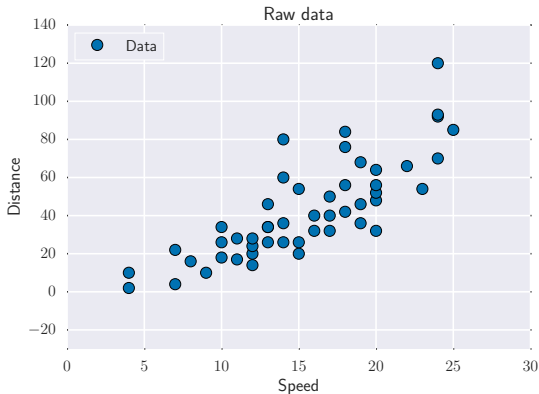
- Formulation mathématique

- Centrer - Réduire

- Vraisemblance

# Point de départ en dimension deux

Exemple : distance de freinage d'une voiture en fonction de la vitesse ( $n = 50$  mesures)

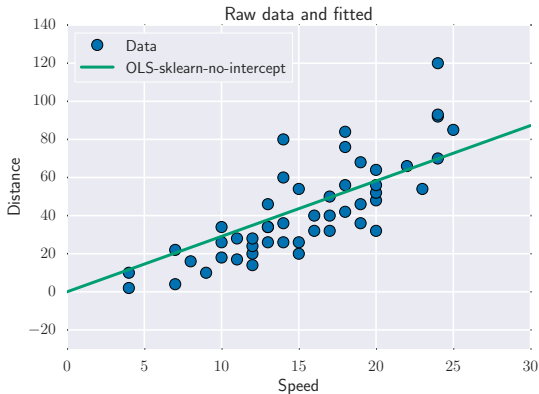


Dataset *cars* :

<https://forge.scilab.org/index.php/p/rdataset/source/file/master/csv/datasets/cars.csv>

# Point de départ en dimension deux

Exemple : distance de freinage d'une voiture en fonction de la vitesse ( $n = 50$  mesures)



Dataset *cars* :

<https://forge.scilab.org/index.php/p/rdataset/source/file/master/csv/datasets/cars.csv>

## Commandes sous python

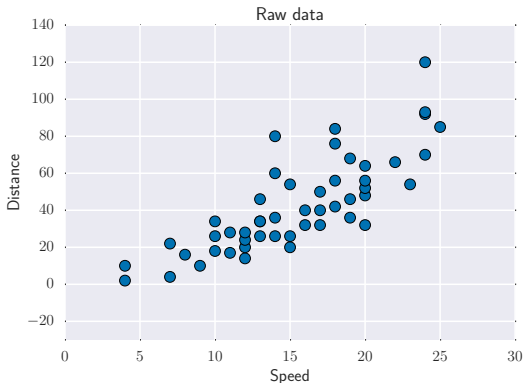
```
import pandas as pd
import matplotlib.pyplot as plt
import sklearn.linear_model as lm
# Load data
url = 'https://forge.scilab.org/index.php/p/rdataset/
      source/file/master/csv/datasets/cars.csv'
dat = pd.read_csv(url)
y = dat['dist']
X = dat[['speed']] # sklearn needs X to have 2 dim.

skl_linmod = lm.LinearRegression(fit_intercept=False)
skl_linmod.fit(X, y) # Fit regression model

fig = plt.figure(figsize=(8, 6))
plt.plot(X, y, 'o', label="Data")
plt.plot(X, skl_linmod.predict(X),
         label="OLS-sklearn-no-intercept")
plt.legend(loc='upper left')
plt.show()
```

# Point de départ en dimension deux : avec constante à l'origine

Exemple : distance de freinage d'une voiture en fonction de la vitesse ( $n = 50$  mesures)

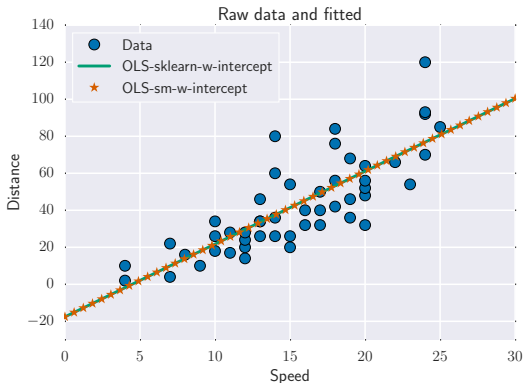


Dataset *cars* :

<https://forge.scilab.org/index.php/p/rdataset/source/file/master/csv/datasets/cars.csv>

# Point de départ en dimension deux : avec constante à l'origine

Exemple : distance de freinage d'une voiture en fonction de la vitesse ( $n = 50$  mesures)



Dataset *cars* :

<https://forge.scilab.org/index.php/p/rdataset/source/file/master/csv/datasets/cars.csv>

## Commandes sous python : avec constantes

```
import statsmodels.api as sm

# data, fitted, etc
y = dat['dist']
X = dat[['speed']]
X = sm.add_constant(X)
results = sm.OLS(y,X).fit()

# plot
fig, ax = plt.subplots(figsize=(8,6))
ax.plot(X['speed'], y, 'o', label="data")
ax.plot(X['speed'], results.fittedvalues,
        linewidth=3, label="OLS-sklearn-no-intercept")
ax.legend(loc='best')
```



# Sommaire

Introduction : visualisation / Python

Moindres carrés uni-dimensionnels

- Modélisation

- Formulation mathématique

- Centrer - Réduire

- Vraisemblance

# Modélisation I

Observations :  $(y_i, x_i)$ , pour  $i = 1, \dots, n$


Hypothèse de modèle linéaire ou de régression linéaire :

$$y_i \approx \theta_0^* + \theta_1^* x_i$$

- ▶  $\theta_0^*$  : ordonnée à l'origine (inconnue)
- ▶  $\theta_1^*$  : coefficient directeur (inconnu)

Rem: les deux paramètres sont inconnus du statisticien

## Définition

- ▶  $y$  est une **observation** ou une variable à expliquer
- ▶  $x$  est une **variable explicative** ou covariable ( : *feature*)

# Interprétation des notations

## Exemple : dataset *cars*

- ▶  $n = 50$
- ▶  $y_i$  : temps de freinage de la voiture  $i$
- ▶  $x_i$  : vitesse de la voiture  $i$
- ▶  $y$  : l'observation est le temps de freinage
- ▶  $x$  : la variable explicative est la vitesse

L'hypothèse de régression linéaire/modèle linéaire revient à postuler que le temps de freinage d'une voiture est proportionnel à sa vitesse

---

**Exo:** utiliser `describe()` de Pandas pour obtenir quelques informations basiques.

---

## Modélisation II

On donne un sens au symbole  $\approx$  de la manière suivante :

### Modèle probabiliste

$$y_i = \theta_0^* + \theta_1^* x_i + \varepsilon_i,$$

$$\varepsilon_i \stackrel{i.i.d.}{\sim} \varepsilon, \text{ pour } i = 1, \dots, n$$

$$\mathbb{E}(\varepsilon) = 0$$

où i.i.d. signifie « indépendants et identiquement distribués »

### Interprétation



$\varepsilon_i = y_i - \theta_0^* - \theta_1^* x_i$  : erreurs entre le modèle théorique et les observations, représentées par des variables aléatoires  $\varepsilon_i$  centrées (on parle aussi de **bruit blanc**).

Rem: l'aspect aléatoire peut avoir diverses causes : bruit de mesure, bruit de transmission, variabilité dans une population, etc.

# Modélisation III

## Définition

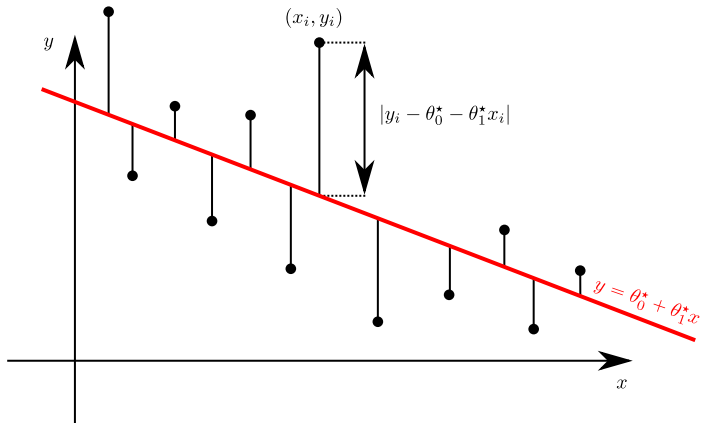
On appelle

- **ordonnée à l'origine** la quantité  $\theta_0^*$  ( : *intercept*)
- **pente** la quantité  $\theta_1^*$  ( : *slope*)

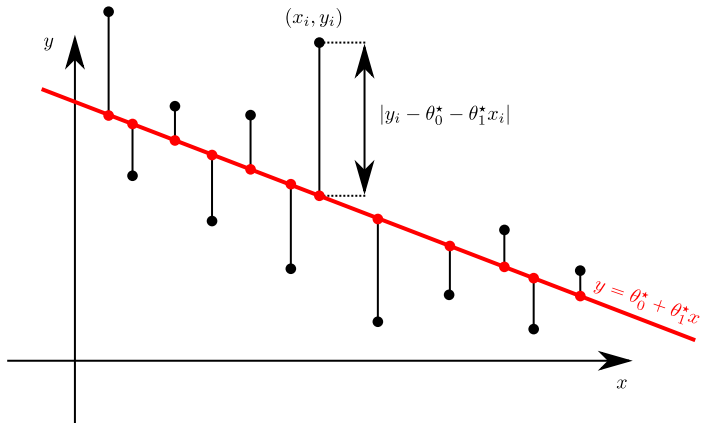
## Objectif

Estimer  $\theta_0^*$  et  $\theta_1^*$  (inconnus) par des quantités  $\hat{\theta}_0$  et  $\hat{\theta}_1$  dépendant des observations  $(y_i, x_i)$  pour  $i = 1, \dots, n$

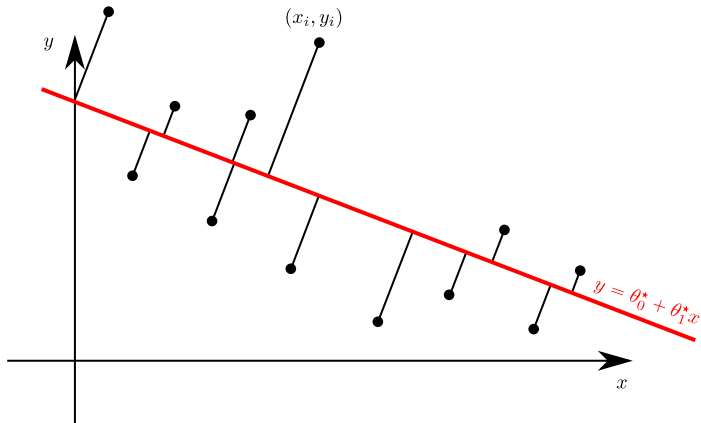
# Estimateur des moindres carrés : visualisation



# Estimateur des moindres carrés : visualisation

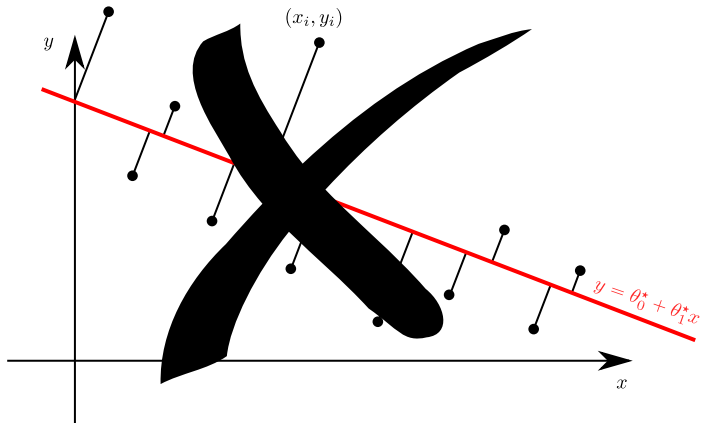


# Estimateur des moindres carrés : visualisation





# Estimateur des moindres carrés : visualisation




# Estimateur des moindres carrés : formulation

Pour des raisons mathématiques on peut choisir de minimiser la somme des carrés des “erreurs”

## Définition

L'estimateur des **moindres carrés** est défini comme suit :

$$(\hat{\theta}_0, \hat{\theta}_1) \in \arg \min_{(\theta_0, \theta_1) \in \mathbb{R}^2} \sum_{i=1}^n (y_i - \theta_0 - \theta_1 x_i)^2$$

- ▶ on l'appelle aussi l'estimateur des **moindres carrés ordinaires**, MCO ( : *ordinary least-square, OLS*)
- ▶ l'intérêt original vient de ce que les conditions du premier ordre sont équivalentes à résoudre un système linéaire

Rem: la notation «  $\in \arg \min$  » ne présage en rien de l'unicité...

# Paternité des moindres carrés



(a) **Adrien-Marie Legendre** : "Nouvelles méthodes pour la détermination des orbites des comètes", 1805



(b) **Carl Friedrich Gauss** : "Theoria Motus Corporum Coelestium in sectionibus conicis solem ambientium" 1809

# Aparté

## Définition

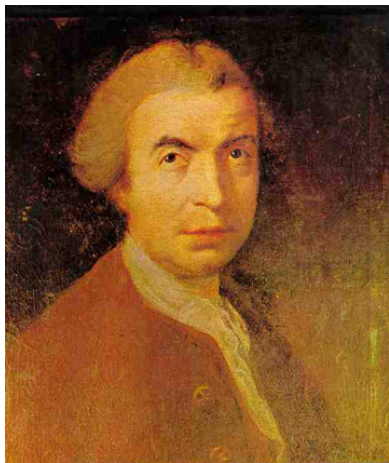
On définit l'estimateur des **moindres déviations absolues** ( : *Least Absolute Deviation (LAD)*) comme suit :

$$(\hat{\theta}_0, \hat{\theta}_1) \in \arg \min_{(\theta_0, \theta_1) \in \mathbb{R}^2} \sum_{i=1}^n |y_i - \theta_0 - \theta_1 x_i|$$

Rem: difficile à calculer sans ordinateur ; nécessite un algorithme itératif d'optimisation non-lisse (*i.e.*, fonctions non différentiables)

Rem: on verra plus tard qu'il est revanche plus robuste (aux *outliers*) que l'estimateur MCO

# Paternité des moindres déviations absolues



(c) **Ruđer Josip Bošković** :“???",  
1757



(d) **Pierre-Simon de Laplace** :  
“Traité de mécanique céleste”, 1799

# Sommaire

Introduction : visualisation / Python

Moindres carrés uni-dimensionnels

Modélisation

Formulation mathématique

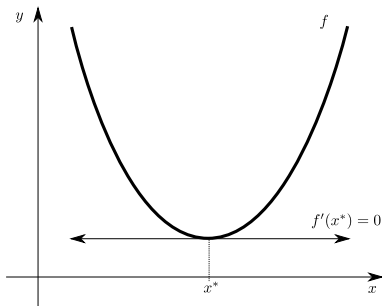
Centrer - Réduire

Vraisemblance

# Condition du premier ordre pour un minimum local (CNO)

## Théorème : règle de Fermat

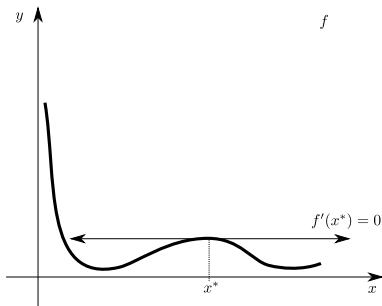
Si  $f$  est différentiable en un minimum local  $x^*$  alors le gradient de  $f$  est nul en  $x^*$ , i.e.,  $\nabla f(x^*) = 0$ .



# Condition du premier ordre pour un minimum local (CNO)

## Théorème : règle de Fermat

Si  $f$  est différentiable en un minimum local  $x^*$  alors le gradient de  $f$  est nul en  $x^*$ , i.e.,  $\nabla f(x^*) = 0$ .



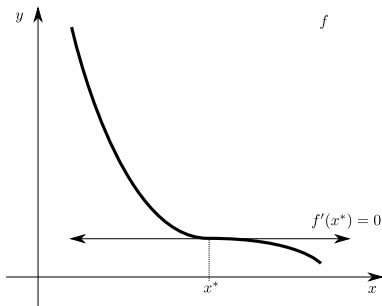
Rem: Ce n'est une condition suffisante que si  $f$  est convexe !



# Condition du premier ordre pour un minimum local (CNO)

## Théorème : règle de Fermat

Si  $f$  est différentiable en un minimum local  $x^*$  alors le gradient de  $f$  est nul en  $x^*$ , i.e.,  $\nabla f(x^*) = 0$ .



Rem: Ce n'est une condition suffisante que si  $f$  est convexe !

## Retour aux moindres carrés

$$\hat{\boldsymbol{\theta}} = (\hat{\theta}_0, \hat{\theta}_1) \in \arg \min_{(\theta_0, \theta_1) \in \mathbb{R}^2} \frac{1}{2} \sum_{i=1}^n (y_i - \theta_0 - \theta_1 x_i)^2$$

On cherche donc à minimiser une fonction de deux variables :

$$f(\theta_0, \theta_1) = f(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^n (y_i - \theta_0 - \theta_1 x_i)^2$$

Conditions nécessaires du premier ordre (CNO) :

$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\hat{\boldsymbol{\theta}}) = \sum_{i=1}^n (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\hat{\boldsymbol{\theta}}) = \sum_{i=1}^n (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) x_i = 0 \end{cases}$$

---

**Exo:**  $f$  est elle convexe ? Aide : calculer sa Hessienne et déterminer si elle est semi-définie positive ou non.

---

## Suite du calcul

Notation usuelle de la moyenne :  $\bar{x}_n = \frac{1}{n} \sum_{i=1}^n x_i$  et  $\bar{y}_n = \frac{1}{n} \sum_{i=1}^n y_i$

Avec ces notations, les CNO s'écrivent (en divisant par  $n$ ) :

$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\hat{\theta}) = \sum_{i=1}^n (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\hat{\theta}) = \sum_{i=1}^n (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i) x_i = 0 \end{cases}$$

$\Leftrightarrow$

$$\begin{cases} \hat{\theta}_0 = \bar{y}_n - \hat{\theta}_1 \bar{x}_n & \text{(CNO1)} \\ \hat{\theta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sum_{i=1}^n (x_i - \bar{x}_n)^2} & \text{(CNO2)} \end{cases}$$

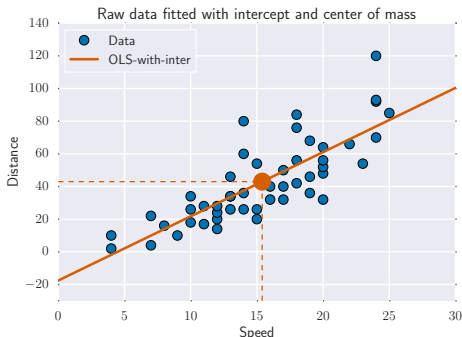
---

**Exo:** Prouver que la formule est vraie si et seulement si  $\mathbf{x} = (x_1, \dots, x_n)^\top$  est non constant, *i.e.*,  $\mathbf{x}$  non proportionnel à  $\mathbf{1}_n$

---

# Centre de gravité et interprétation

$$(\text{CNO1}) \Leftrightarrow (\bar{x}_n, \bar{y}_n) \in \{(x, y) \in \mathbb{R}^2 : y = \hat{\theta}_0 + \hat{\theta}_1 x\}$$



- ▶  $\overline{speed} = 15.4$
- ▶  $\overline{dist} = 42.98$
- ▶  $\hat{\theta}_0 = -17.579095$  l'ordonnée à l'origine (négatif!!!)
- ▶  $\hat{\theta}_1 = 3.932409$  pente de la droite

Interprétation physique : le centre de gravité du nuage de points est sur la droite de régression (estimée)

# Reformulation vectorielle

Notation :  $\mathbf{x} = (x_1, \dots, x_n)^\top$  et  $\mathbf{y} = (y_1, \dots, y_n)^\top$

$$(\text{CNO2}) \Leftrightarrow \hat{\theta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sum_{i=1}^n (x_i - \bar{x}_n)^2}$$

$$(\text{CNO2}) \Leftrightarrow \hat{\theta}_1 = \text{corr}_n(\mathbf{x}, \mathbf{y}) \cdot \frac{\sqrt{\text{var}_n(\mathbf{y})}}{\sqrt{\text{var}_n(\mathbf{x})}}$$

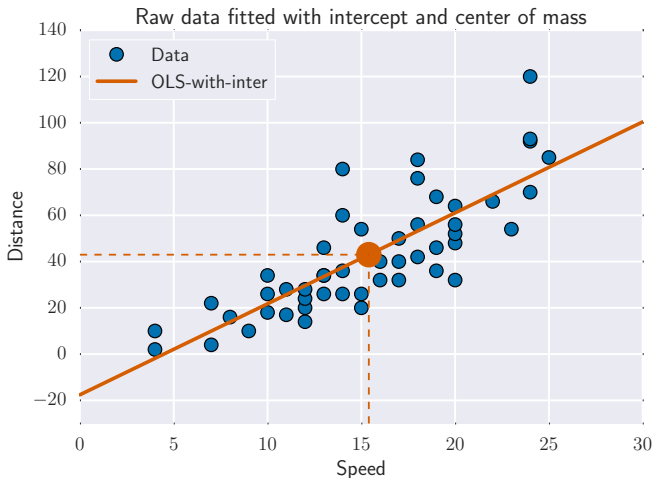
où 
$$\text{corr}_n(\mathbf{x}, \mathbf{y}) = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sqrt{\text{var}_n(\mathbf{x})} \sqrt{\text{var}_n(\mathbf{y})}}$$

et 
$$\text{var}_n(\mathbf{z}) = \frac{1}{n} \sum_{i=1}^n (z_i - \bar{z}_n)^2 \text{ (pour tout } \mathbf{z} = (z_1, \dots, z_n)^\top \text{)}$$

respectivement **corrélations empiriques** et **variances empiriques**

## Retour sur l'exemple du dataset *cars*

Pente de la droite tracée :  $\text{corr}_n(\mathbf{x}, \mathbf{y}) \cdot \frac{\sqrt{\text{var}_n(\mathbf{y})}}{\sqrt{\text{var}_n(\mathbf{x})}} = 3.932409$ .



# Sommaire

Introduction : visualisation / Python

Moindres carrés uni-dimensionnels

Modélisation

Formulation mathématique

Centrer - Réduire

Vraisemblance

# Recentrage

Nouveau modèle d'observation, dit **(re)centré** :

$$\text{Si pour tout } i = 1, \dots, n : \begin{cases} x'_i = x_i - \bar{x}_n \\ y'_i = y_i - \bar{y}_n \end{cases} \Leftrightarrow \begin{cases} \mathbf{x}' = \mathbf{x} - \bar{x}_n \mathbf{1}_n \\ \mathbf{y}' = \mathbf{y} - \bar{y}_n \mathbf{1}_n \end{cases}$$

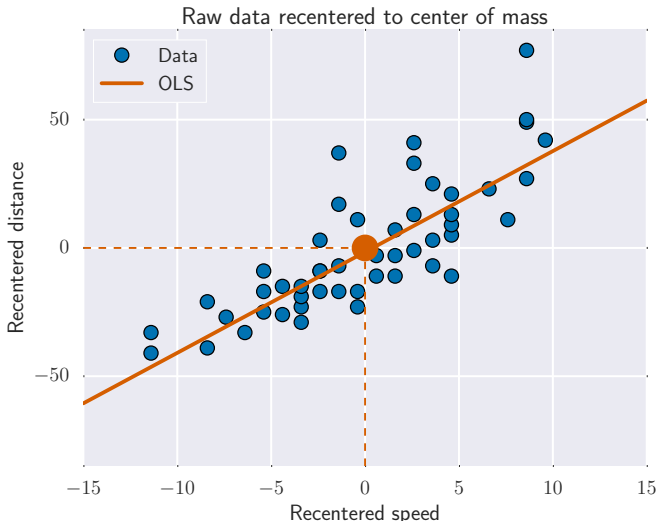
si l'on note  $\mathbf{1}_n = (1, \dots, 1)^\top \in \mathbb{R}^n$  et que l'on résout le programme des moindres carrés pour les  $(\mathbf{x}', \mathbf{y}')$  alors

$$\begin{cases} \hat{\theta}'_0 = 0 \\ \hat{\theta}'_1 = \frac{\frac{1}{n} \sum_{i=1}^n x'_i y'_i}{\frac{1}{n} \sum_{i=1}^n x'^2_i} \end{cases}$$

Rem: équivalent à choisir pour origine le centre de gravité du nuage de points, *i.e.*,  $(\bar{x}'_n, \bar{y}'_n) = (0, 0)$



## Recentrage (II)



## Recentrage et réinterprétation

Considérons le coefficient  $\hat{\theta}'_1$  ( $\hat{\theta}'_0 = 0$ ) des données centrées  $\mathbf{y}'$  et  $\mathbf{x}'$   
L'objectif se réécrit

$$\hat{\theta}'_1 = \arg \min_{\theta_1} \sum_{i=1}^n (y'_i - \theta_1 x'_i)^2 = \arg \min_{\theta'_1} \sum_{i=1}^n x'^2_i \left( \frac{y'_i}{x'_i} - \theta_1 \right)^2$$

Interprétation :  $\hat{\theta}'_1$  est une moyenne pondérée des “pentes”  $\frac{y'_i}{x'_i}$

$$\hat{\theta}'_1 = \frac{\sum_{i=1}^n x'^2_i \frac{y'_i}{x'_i}}{\sum_{i=1}^n x'^2_i}$$

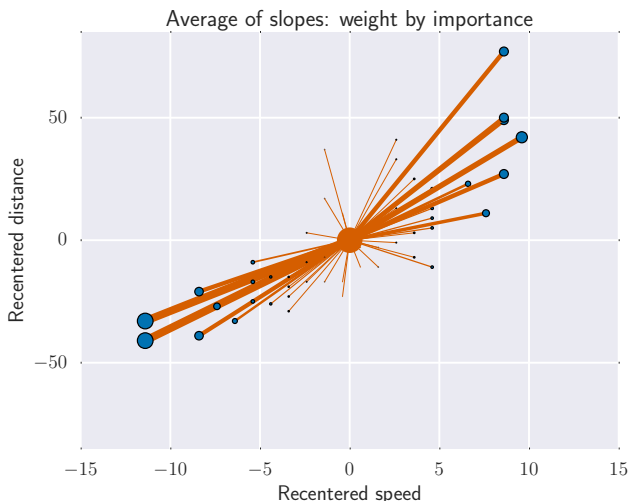
Influence des points extrêmes : poids proportionnels au carré des  $x'_i$

---

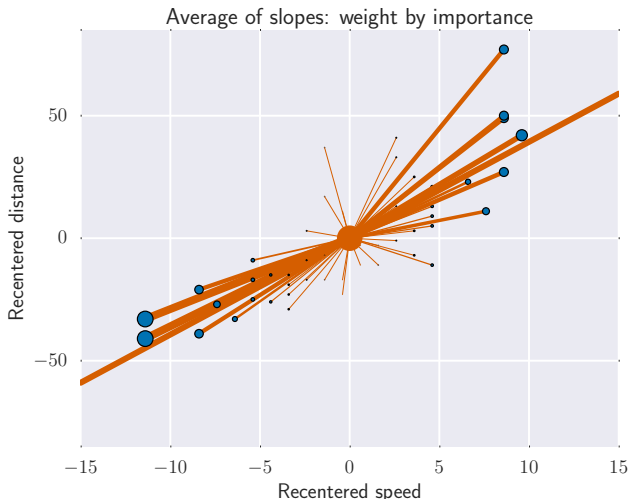
**Exo:** Montrer que cette formule donne bien la solution du problème d'optimisation précédent.

---

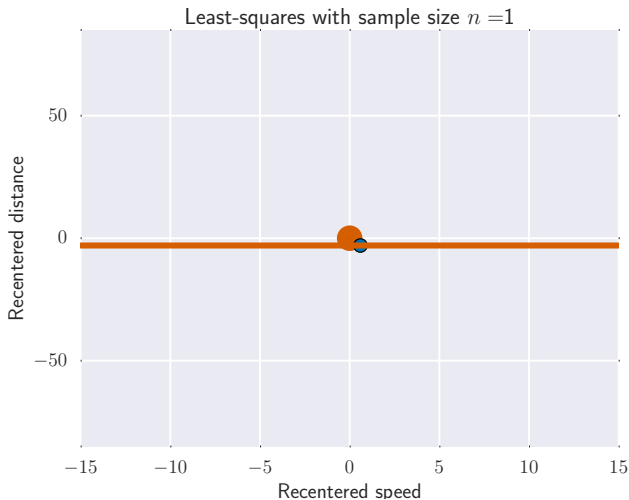
# Illustration de l'influence des points extrêmes



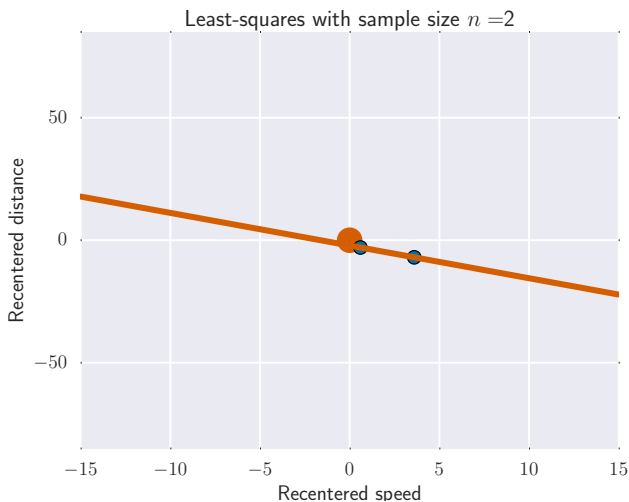
## Illustration de l'influence des points extrêmes



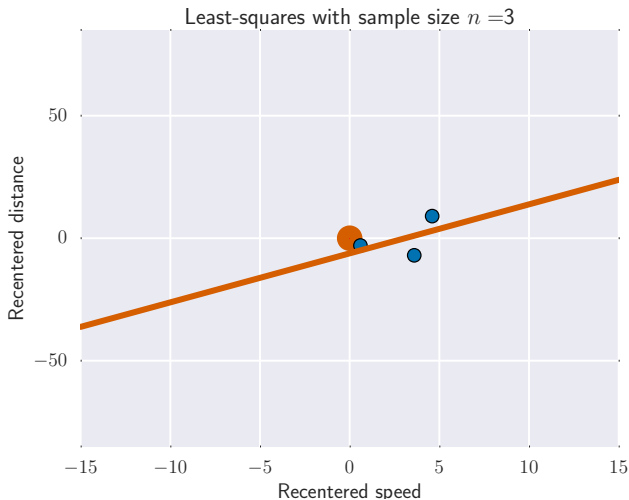
# Illustration de l'influence des points extrêmes



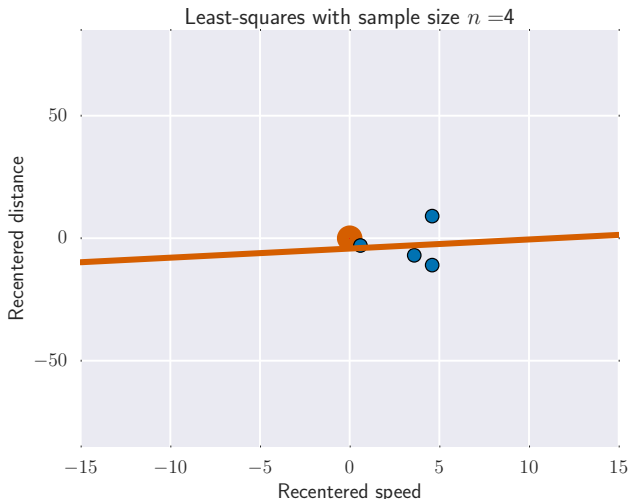
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

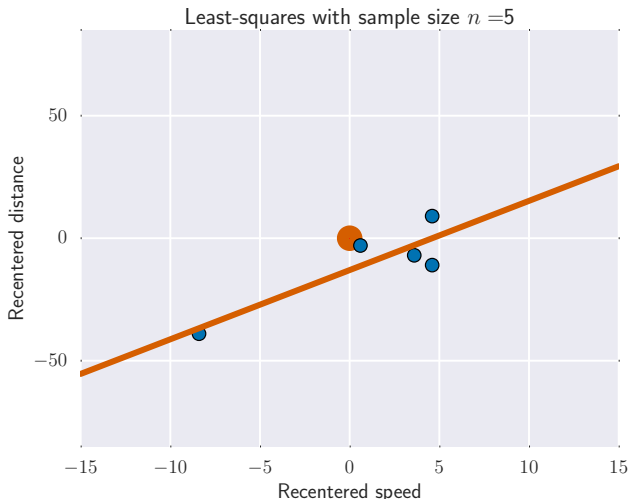


# Illustration de l'influence des points extrêmes

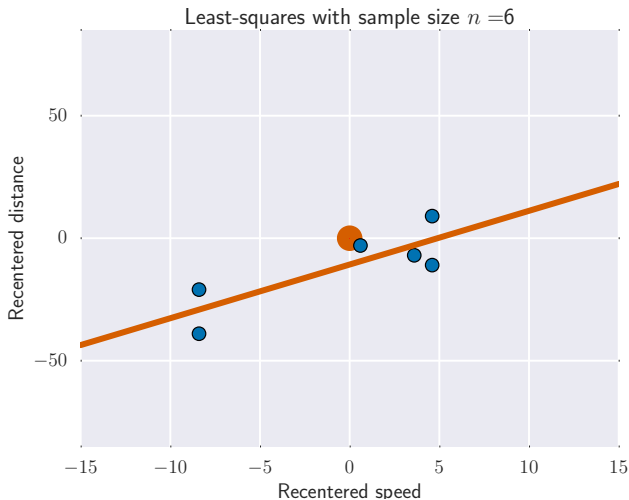




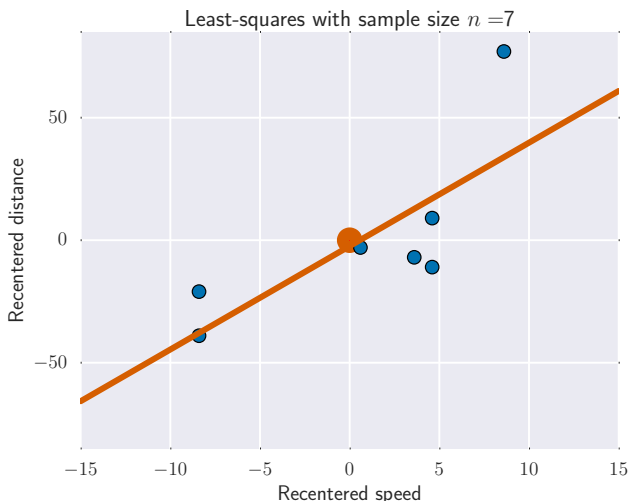
# Illustration de l'influence des points extrêmes



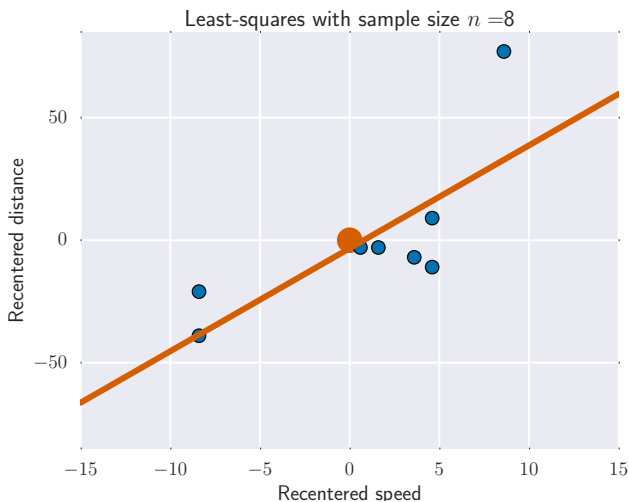
# Illustration de l'influence des points extrêmes



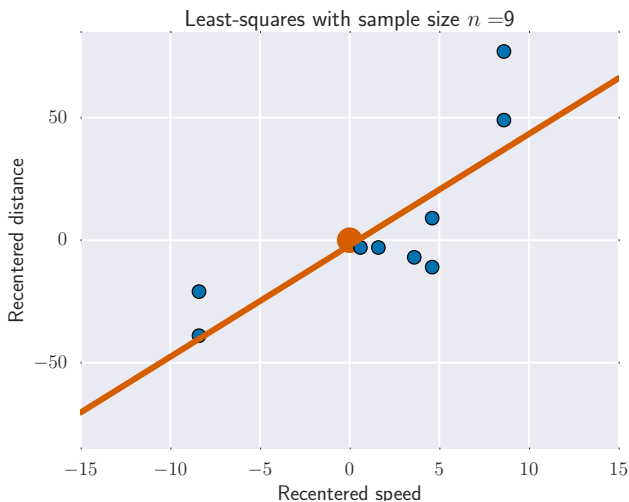
# Illustration de l'influence des points extrêmes



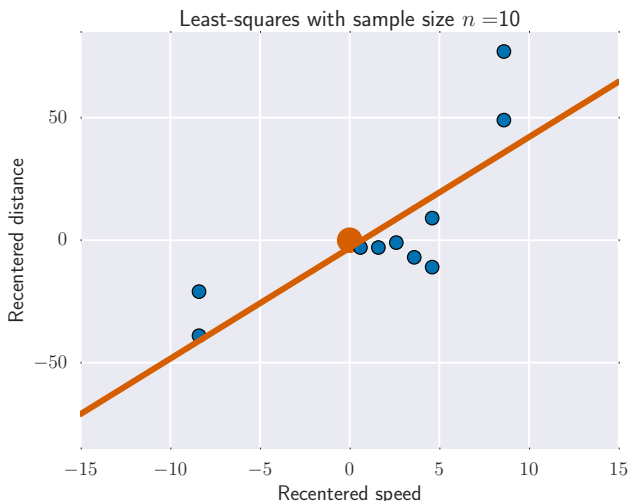
# Illustration de l'influence des points extrêmes



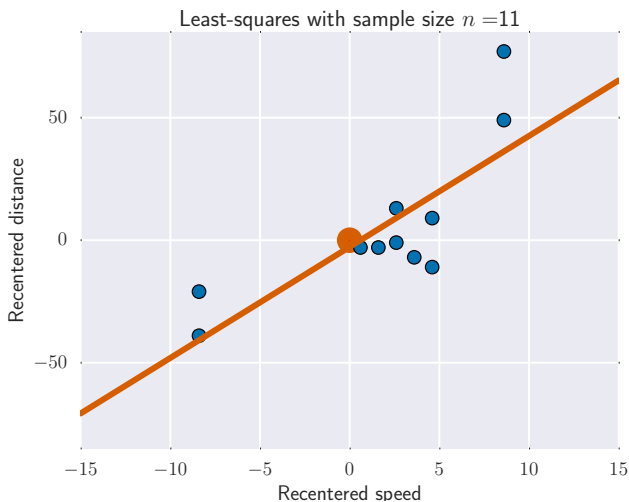
# Illustration de l'influence des points extrêmes



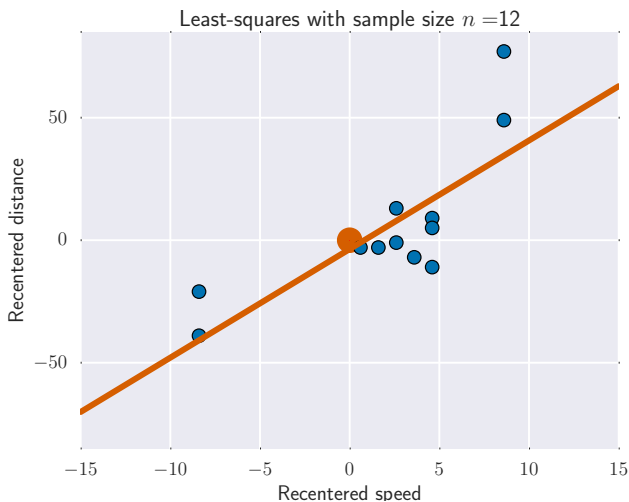
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

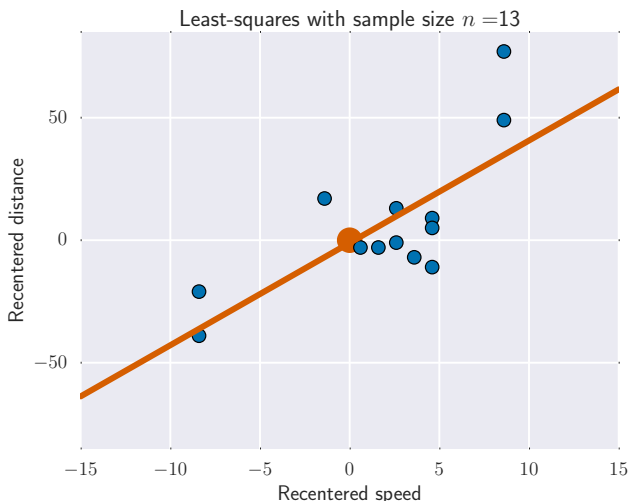


# Illustration de l'influence des points extrêmes





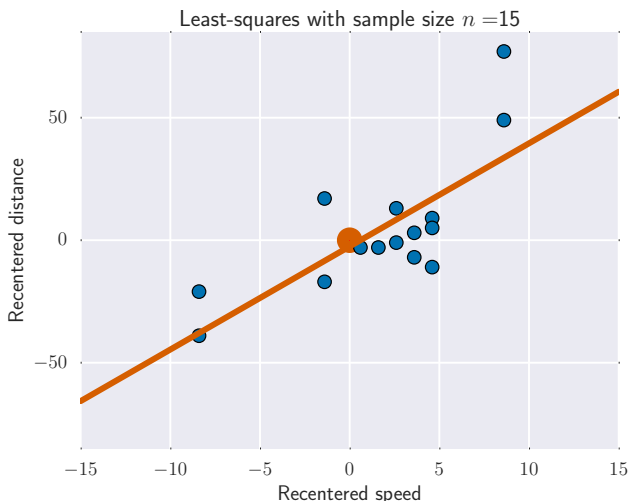
# Illustration de l'influence des points extrêmes



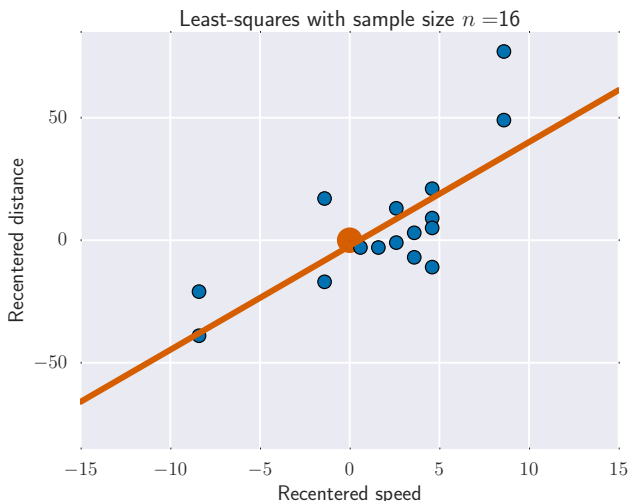
# Illustration de l'influence des points extrêmes



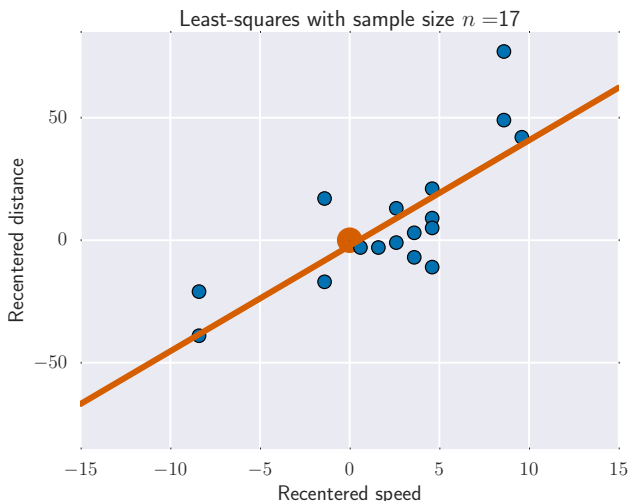
# Illustration de l'influence des points extrêmes



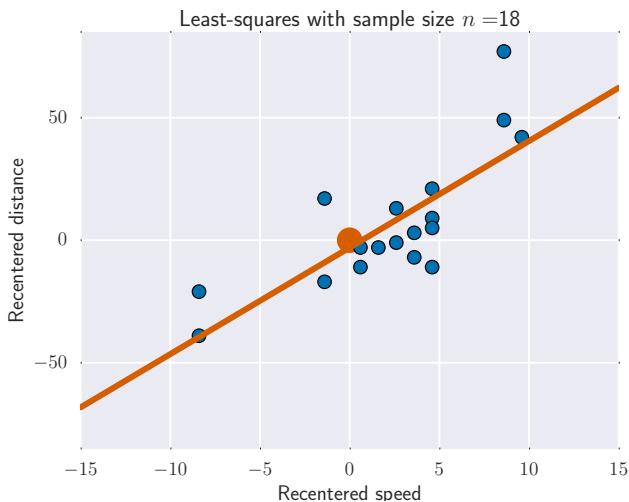
# Illustration de l'influence des points extrêmes



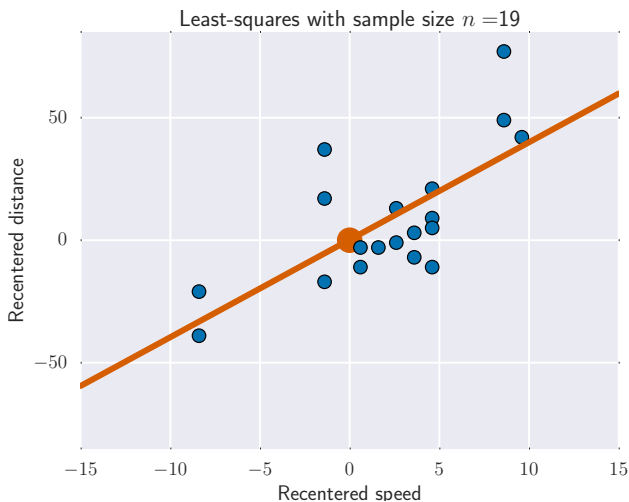
# Illustration de l'influence des points extrêmes



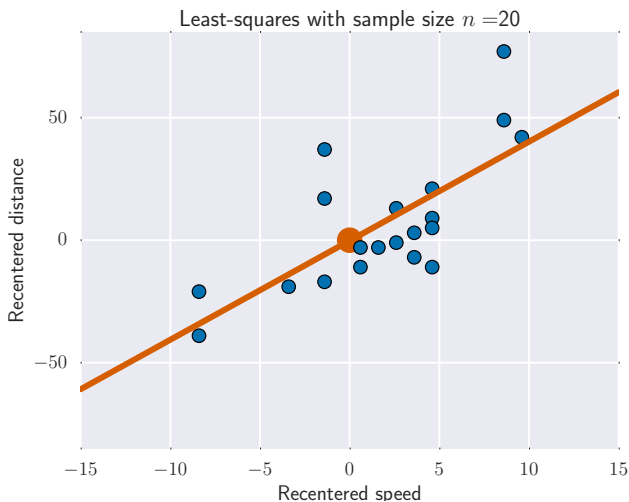
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

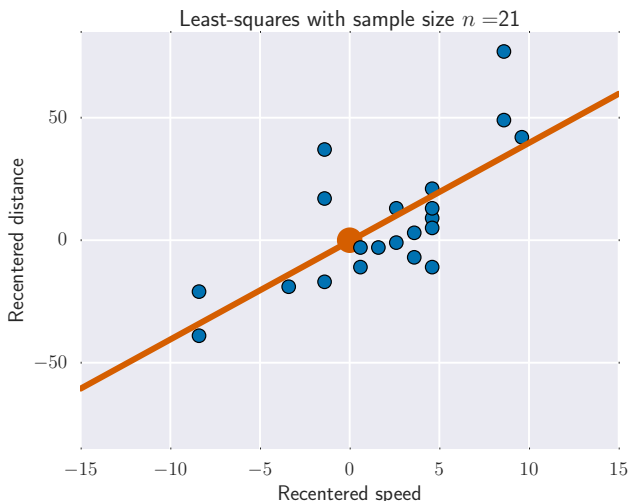


# Illustration de l'influence des points extrêmes

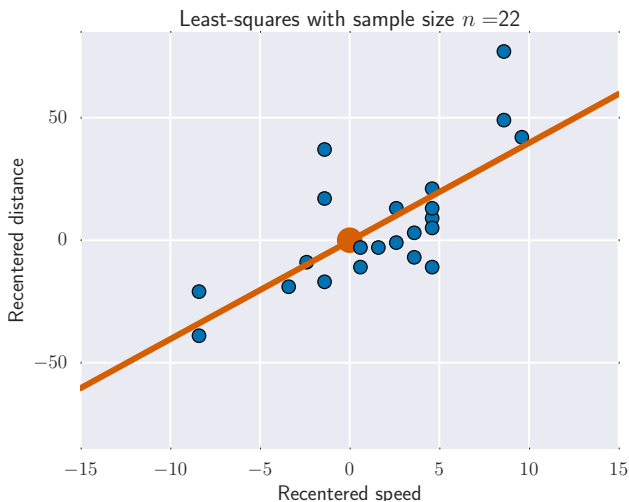




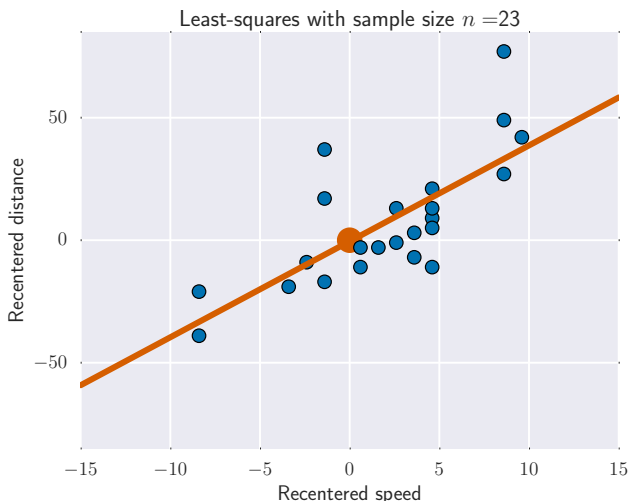
# Illustration de l'influence des points extrêmes



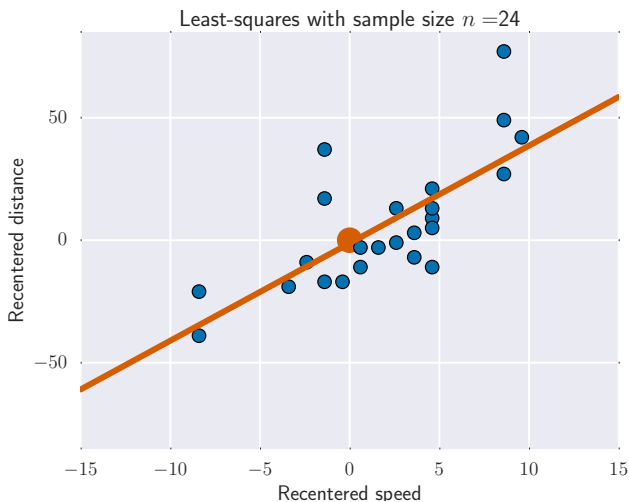
# Illustration de l'influence des points extrêmes



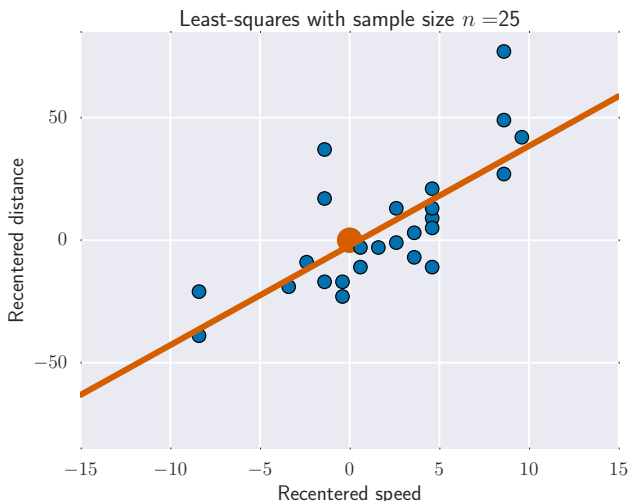
# Illustration de l'influence des points extrêmes



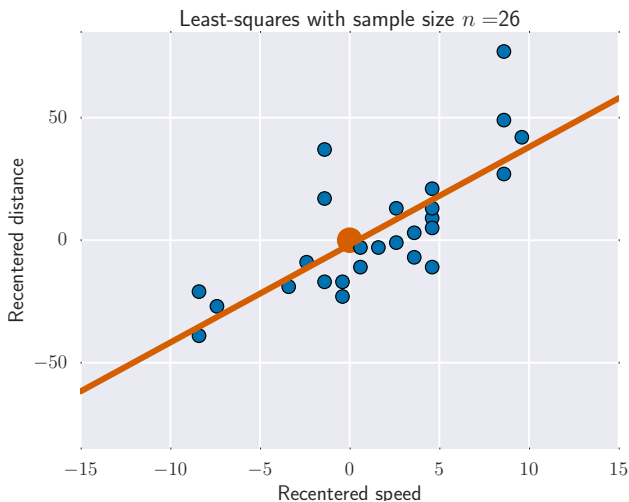
# Illustration de l'influence des points extrêmes



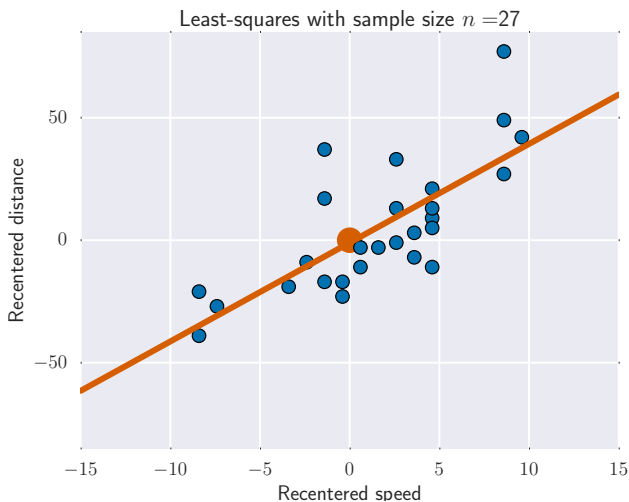
# Illustration de l'influence des points extrêmes



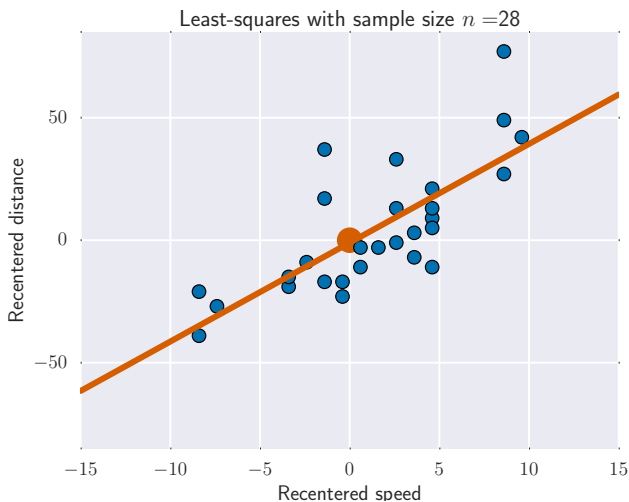
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

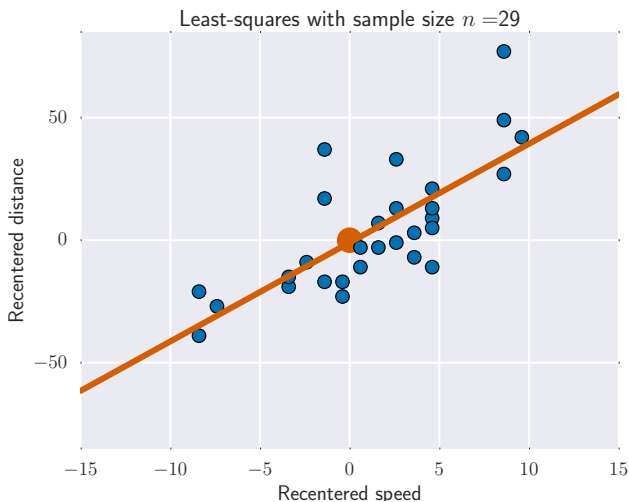


# Illustration de l'influence des points extrêmes

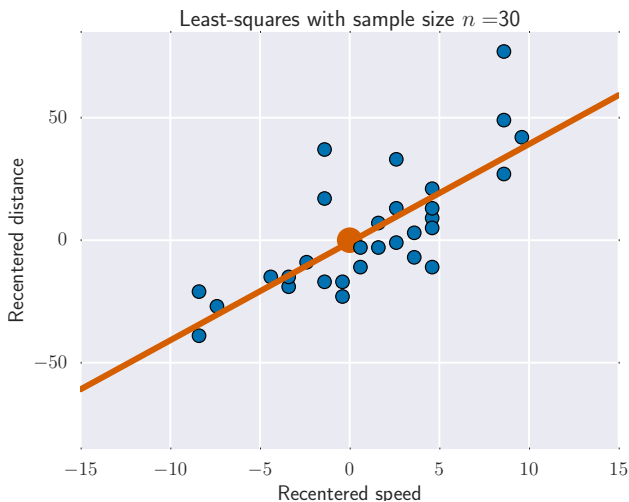




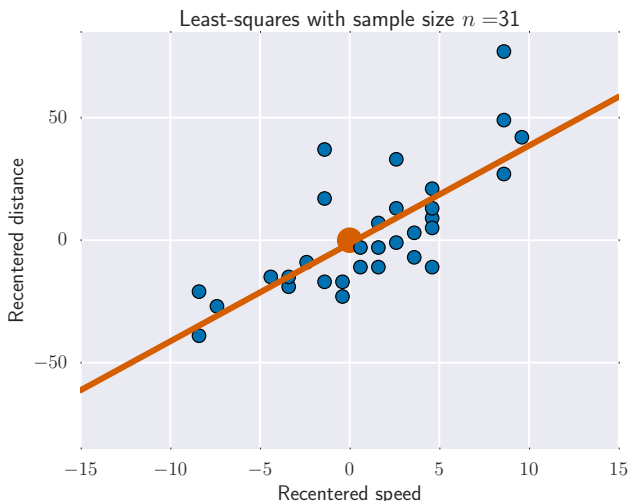
# Illustration de l'influence des points extrêmes



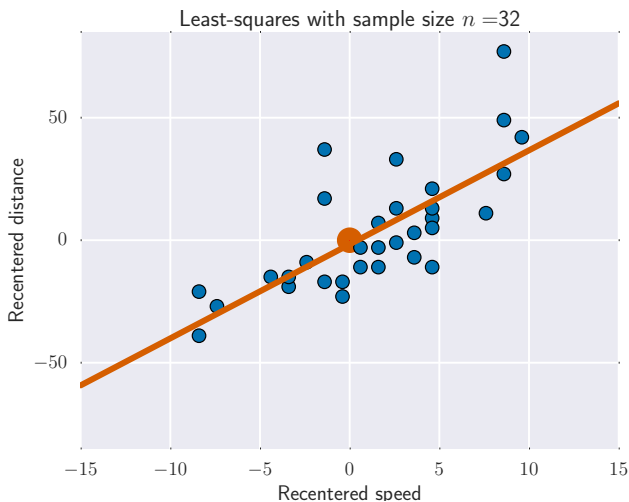
# Illustration de l'influence des points extrêmes



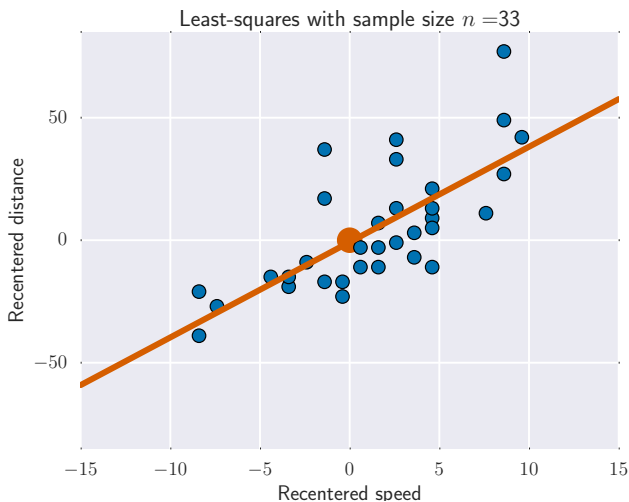
# Illustration de l'influence des points extrêmes



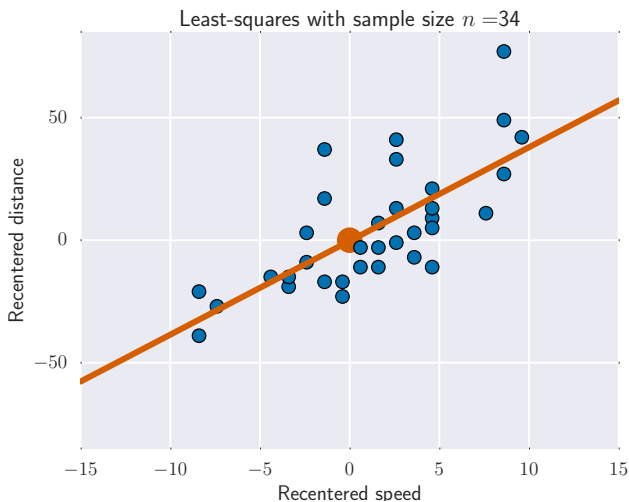
# Illustration de l'influence des points extrêmes



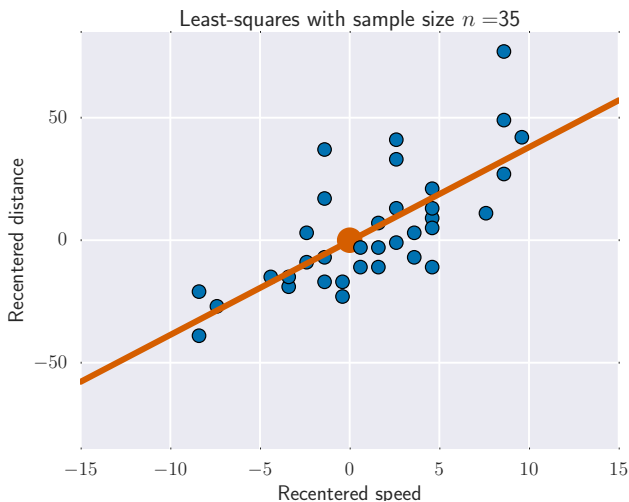
# Illustration de l'influence des points extrêmes



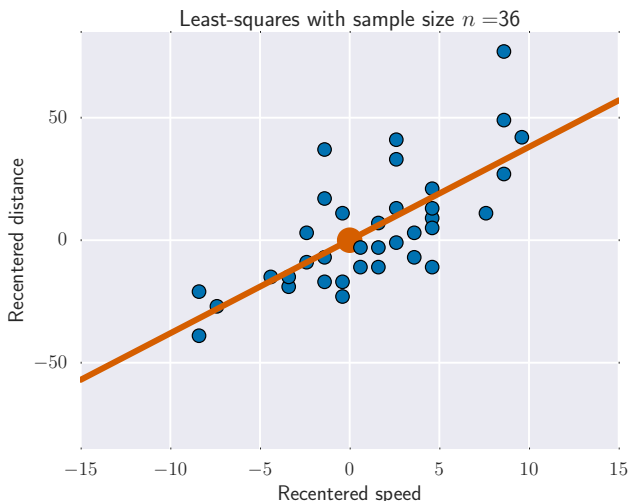
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

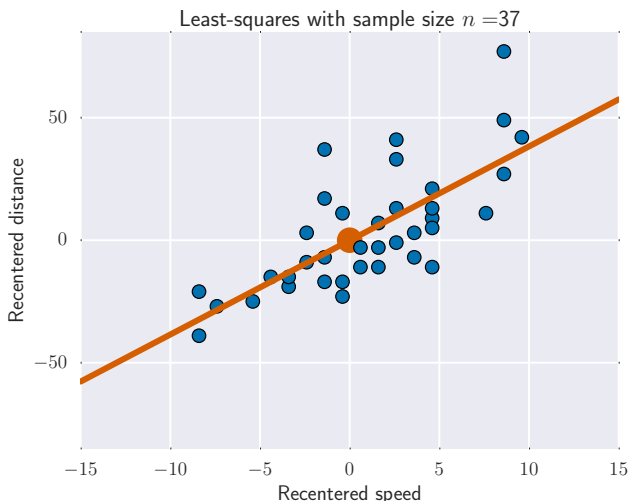


# Illustration de l'influence des points extrêmes

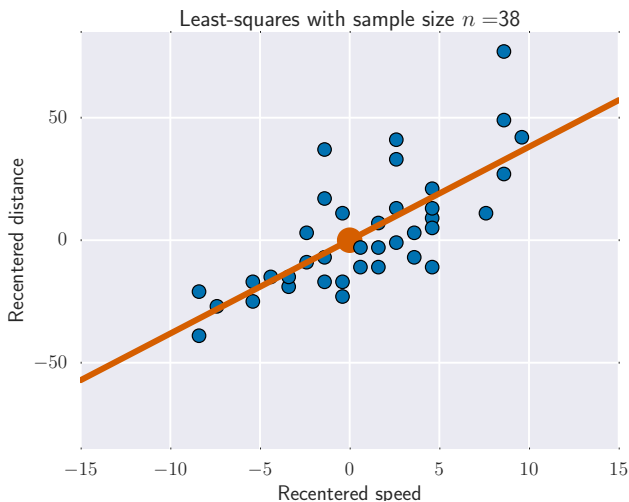




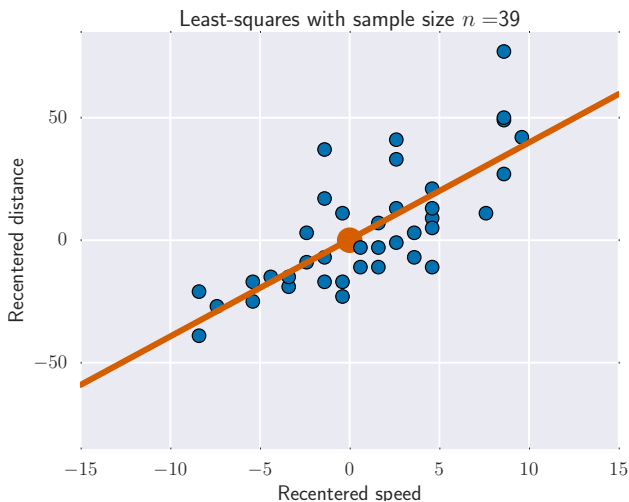
# Illustration de l'influence des points extrêmes



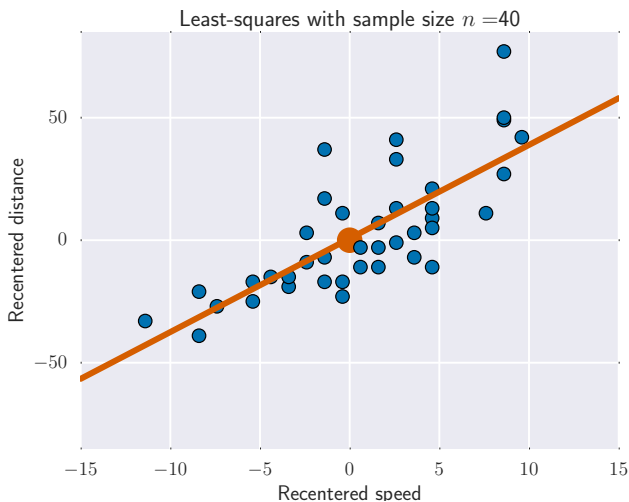
# Illustration de l'influence des points extrêmes



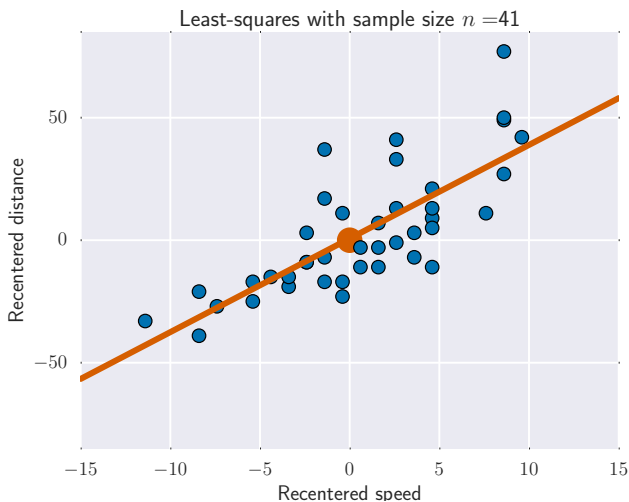
# Illustration de l'influence des points extrêmes



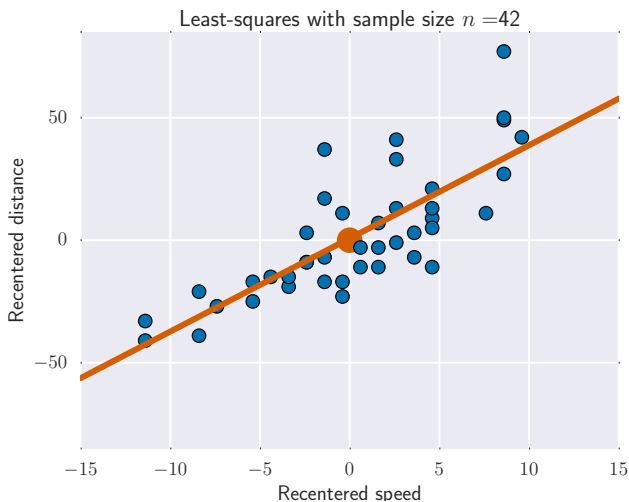
# Illustration de l'influence des points extrêmes



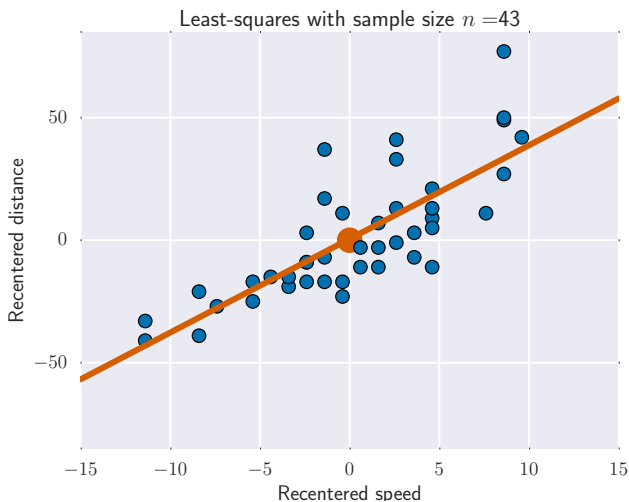
# Illustration de l'influence des points extrêmes



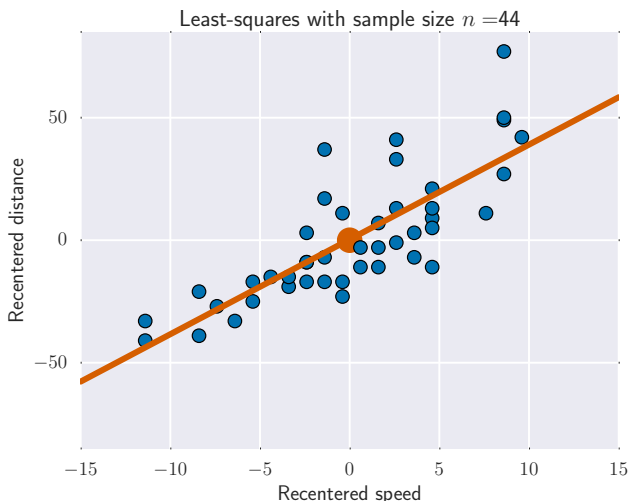
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes

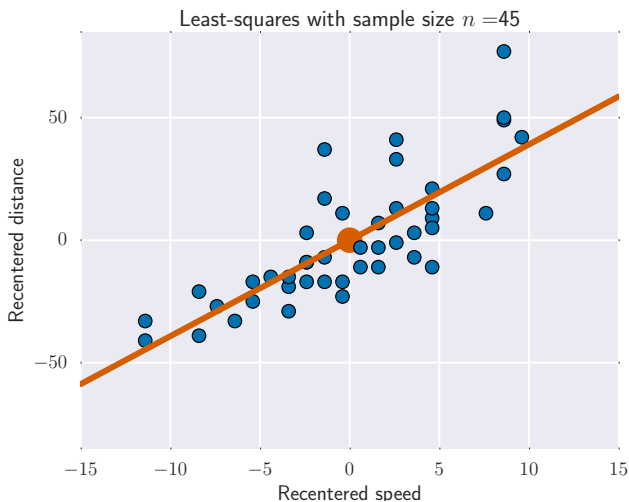


# Illustration de l'influence des points extrêmes

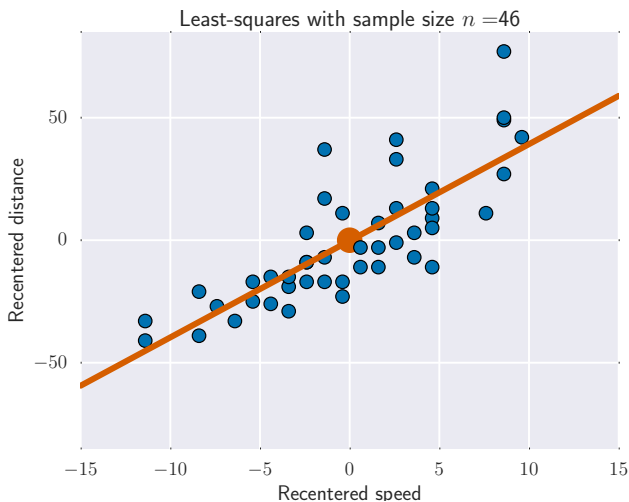




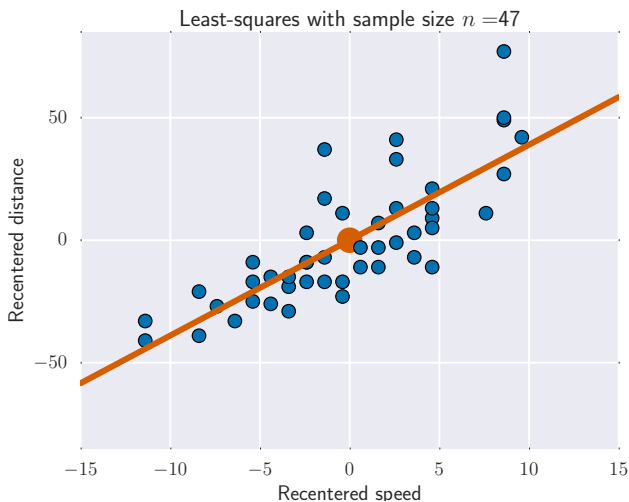
# Illustration de l'influence des points extrêmes



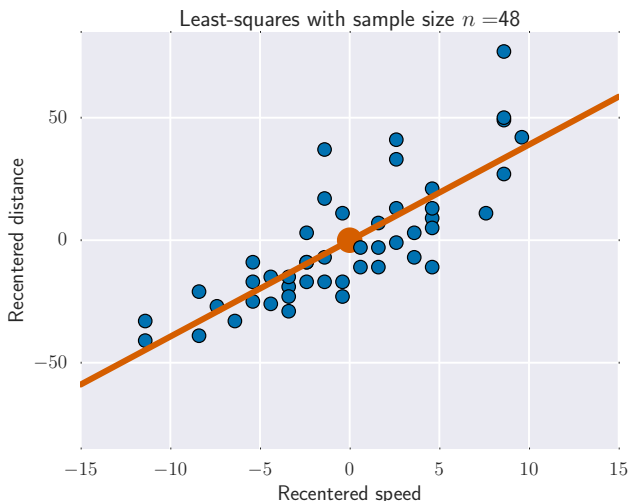
# Illustration de l'influence des points extrêmes



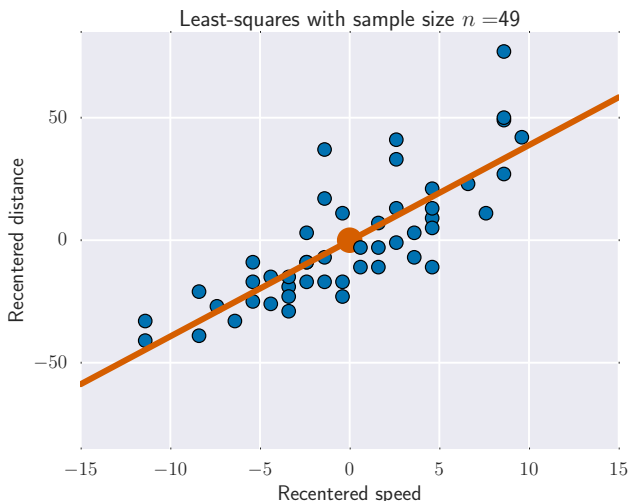
# Illustration de l'influence des points extrêmes



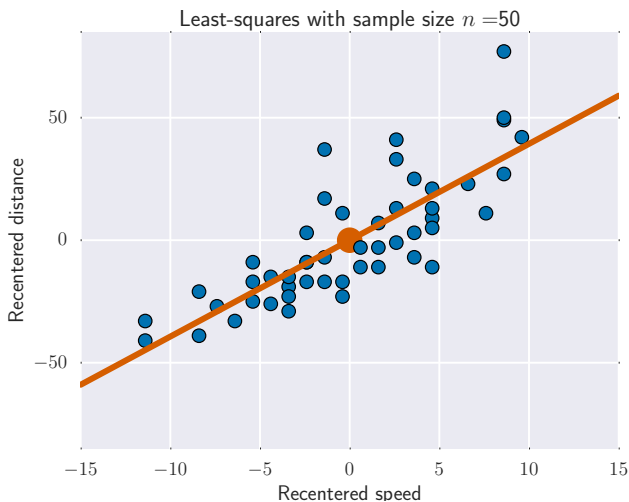
# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes



# Illustration de l'influence des points extrêmes



## Recentrage + mise à l'échelle

Nouveau modèle d'observation, dit aussi **centré-réduit** :

$$\forall i = 1, \dots, n : \begin{cases} x_i'' = (x_i - \bar{x}_n) / \sqrt{\text{var}_n(\mathbf{x})} \\ y_i'' = (y_i - \bar{y}_n) / \sqrt{\text{var}_n(\mathbf{y})} \end{cases} \Leftrightarrow \begin{cases} \mathbf{x}'' = \frac{\mathbf{x} - \bar{x}_n \mathbf{1}_n}{\sqrt{\text{var}_n(\mathbf{x})}} \\ \mathbf{y}'' = \frac{\mathbf{y} - \bar{y}_n \mathbf{1}_n}{\sqrt{\text{var}_n(\mathbf{y})}} \end{cases}$$

En résolvant le programme des moindres carrés pour  $(\mathbf{x}'', \mathbf{y}'')$  alors

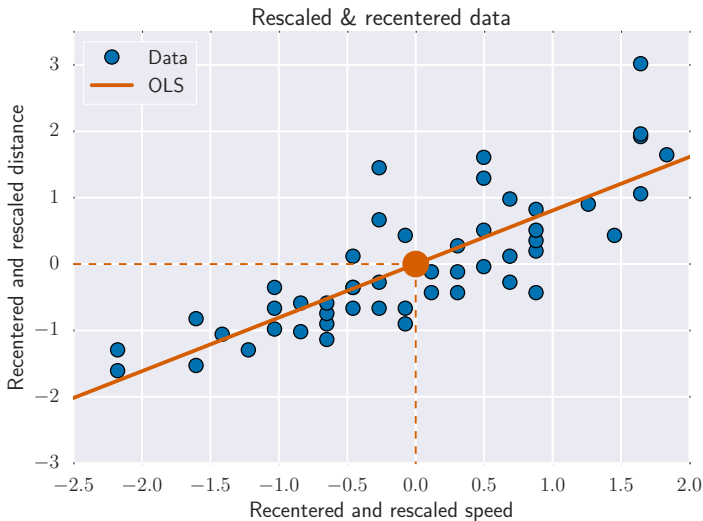
$$\begin{cases} \hat{\theta}_0'' = 0 \\ \hat{\theta}_1'' = \frac{1}{n} \sum_{i=1}^n x_i'' y_i'' \end{cases}$$

C'est équivalent à choisir le centre de gravité du “nuage de points” pour origine et normaliser  $\mathbf{x}$  et  $\mathbf{y}$  pour la **norme empirique**  $\|\cdot\|_n$  :

$$\|\mathbf{x}''\|_n^2 = \frac{1}{n} \sum_{i=1}^n (x_i'')^2 = 1$$

$$\|\mathbf{y}''\|_n^2 = \frac{1}{n} \sum_{i=1}^n (y_i'')^2 = 1$$

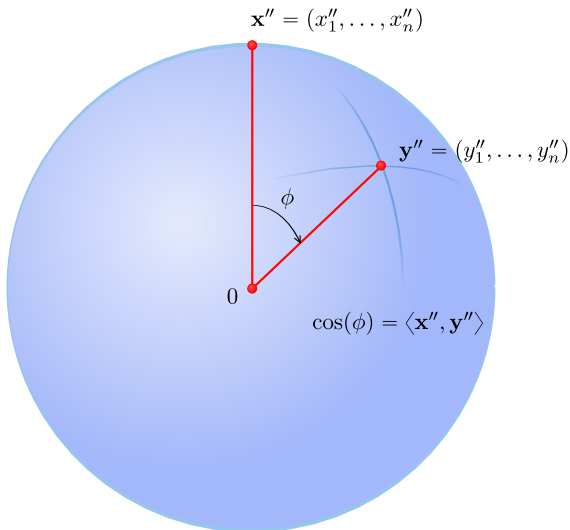
## Recentrage + mise à l'échelle (II)






# Interprétation corrélation (cas centré-réduit)

Exemple : cas  $n = 3$  et  $\|\mathbf{x}''\|_n^2 = \|\mathbf{y}''\|_n^2 = 1$



# Quand/Pourquoi pré-traiter ?

On peut recentrer  $y$  ou bien ajouter une variable constante au modèle, mais recentrer est souvent plus simple

Rem: attention dans les cas creux ( : *sparse*) cela peut être plus difficile à gérer, cf. en régression logistique pour des données textuelles

Pour la/les variables explicatives la mise à l'échelle est importante :

- ▶ si l'on veut interpréter l'ordonnée à l'origine, ou bien interpréter l'amplitude des coefficients de régression (notion de coefficients "petits")
- ▶ si l'on veut pénaliser les coefficients (cf. Lasso, Ridge, etc.)
- ▶ pour des raisons numériques (e.g., accélérer les calculs, améliorer le conditionnement, etc.)

Rem: en anticipant sur la suite, centrer/réduire est plus utile en **estimation** qu'en **prédiction**

# Recentrage en python

Utiliser par exemple le recentrage de sklearn, voir l'aide `skl.preprocessing.StandardScaler()`? si besoin :

```
from sklearn import preprocessing

scaler = preprocessing.StandardScaler().fit(X)
print np.isclose(scaler.mean_, np.mean(X))
print np.array_equal(scaler.std_, np.std(X))

print np.array_equal(scaler.transform(X),
                     (X - np.mean(X)) / np.std(X))

print np.array_equal(scaler.transform([26]),
                     (26 - np.mean(X)) / np.std(X))
```

Plus d'informations, variations, etc. :

<http://scikit-learn.org/stable/modules/preprocessing.html>

# Définitions

## Prédicteur

On appelle **prédicteur** une fonction qui à une nouvelle valeur de la variable explicative  $x_{n+1}$  associe une estimation de la variable à expliquer. Pour les moindres carrées la prédiction est obtenue par :

$$\text{pred}(x_{n+1}) = \hat{\theta}_0 + \hat{\theta}_1 x_{n+1}$$

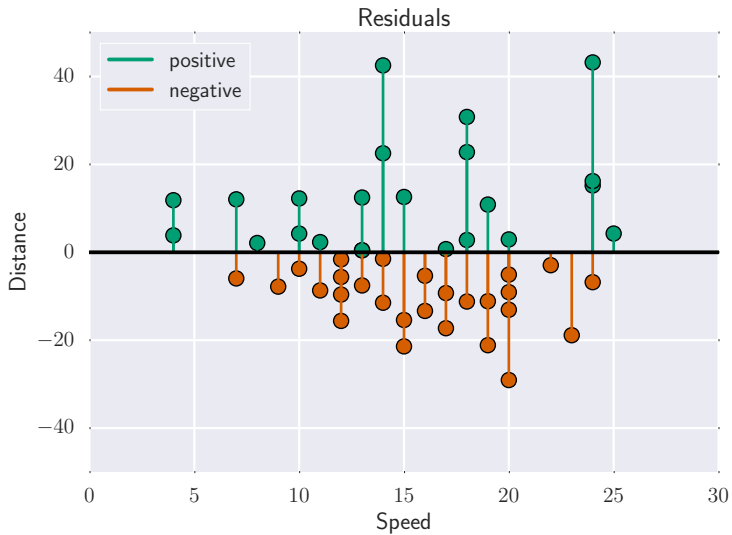
Rem: souvent on note  $\hat{y}_{n+1} = \text{pred}(x_{n+1})$  s'il n'y pas d'ambiguïté

## Résidus

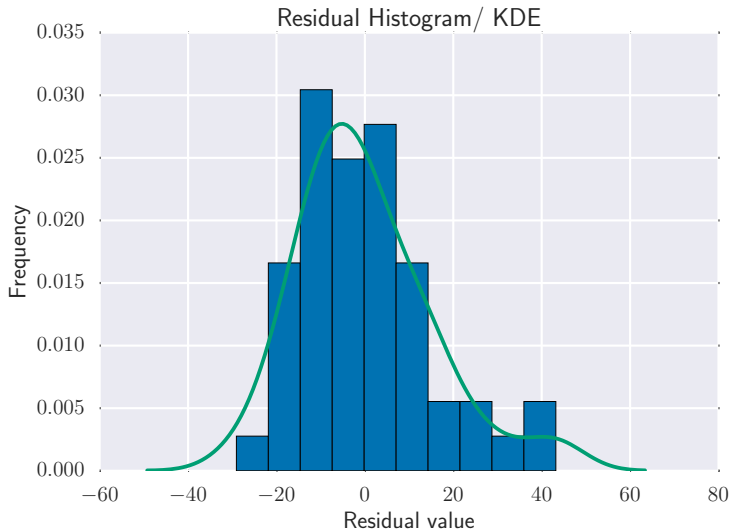
On appelle **résidu** d'un prédicteur la différence entre la valeur observée et la valeur prédite :

$$r_i = y_i - \text{pred}(x_i) = y_i - \hat{y}_i = y_i - (\hat{\theta}_0 + \hat{\theta}_1 x_i)$$

# Résidus



# Histogramme des résidus



## Résidus (suite)

Rappel :  $r_i = y_i - \text{pred}(x_i) = y_i - \hat{y}_i = y_i - (\hat{\theta}_0 + \hat{\theta}_1 x_i)$

### Propriété

Les résidus sont **centrés** :  $\frac{1}{n} \sum_{i=1}^n r_i = 0$

Démonstration :

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n r_i &= \frac{1}{n} \sum_{i=1}^n (y_i - \text{pred}(x_i)) \\ &= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i) \\ &= \frac{1}{n} \sum_{i=1}^n (y_i - (\hat{\theta}_0 + \hat{\theta}_1 x_i)) \\ &= \bar{y}_n - (\hat{\theta}_0 + \hat{\theta}_1 \bar{x}_n) = 0 \end{aligned}$$

# Sommaire

Introduction : visualisation / Python

**Moindres carrés uni-dimensionnels**

Modélisation

Formulation mathématique

Centrer - Réduire

**Vraisemblance**




# Raison du choix des moindres carrés

- ▶ Intérêt calculatoire : historiquement il fallait éviter des méthodes trop gourmandes en calcul (e.g., itératives)
- ▶ Intérêt théorique : il est possible d'analyser en détails l'estimateur sous des hypothèses simples

Exemple : sous l'hypothèse que le bruit suit une loi gaussienne

$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

le maximum de vraisemblance amène à considérer les moindres carrés comme estimateur naturel de  $(\theta_0^*, \theta_1^*)$

Rem: pour un autre modèle de bruit ou pour limiter l'effet de points aberrants ( : *outliers*) on peut alternativement résoudre (e.g., QuantReg dans Statsmodels)

$$\hat{\theta} = (\hat{\theta}_0, \hat{\theta}_1) \in \arg \min_{(\theta_0, \theta_1) \in \mathbb{R}^2} \sum_{i=1}^n |y_i - \theta_0 - \theta_1 x_i|$$

# Vraisemblance gaussienne

## Rappel : la densité d'une gaussienne uni-dimensionnelle

On note  $Y \sim \mathcal{N}(\mu, \sigma^2)$ , une variable dont la densité est

$$\varphi_{\mu, \sigma}(y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - \mu)^2}{2\sigma^2}\right)$$

Supposons :  $y_i \sim \mathcal{N}(\theta_0^* + \theta_1^* x_i, \sigma^2)$ , i.e.,  $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ , alors le couple  $(\theta_0, \theta_1)$  le plus **vraisemblable** au vu des données est celui qui maximise la densité du vecteur  $(y_1, \dots, y_n)$ .

Sous une hypothèse d'indépendance, c'est la solution de :

$$(\hat{\theta}_0, \hat{\theta}_1) \in \arg \max_{(\theta_0, \theta_1) \in \mathbb{R}^2} \prod_{i=1}^n \left( \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \theta_0 - \theta_1 x_i)^2}{2\sigma^2}\right) \right)$$

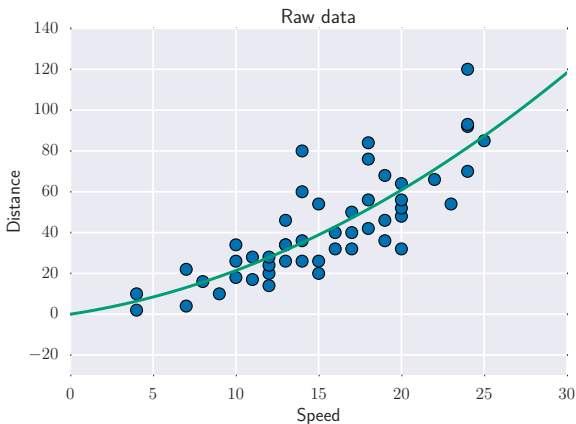
---

**Exo:** retrouver les moindres carrés depuis cette formulation

---

## Discussion : vers le multidimensionnel

D'après les lois physiques (ou vos souvenirs de permis de conduire), la courbe attendue est plutôt une parabole qu'une droite. On verra par la suite comment utiliser la même procédure des moindres carrés pour obtenir l'ajustement suivant :



# Site webs et livres pour aller plus loin

- Quelques [notebooks](#) de moindres carrés avec `statsmodels`
- [McKinney \(2012\)](#) concernant `python` pour les statistiques
- [Lejeune \(2010\)](#) concernant le modèle linéaire (notamment)
- pour aller plus loin (plus technique), lire le cours de régression de [B. Delyon](#)

# Références I

- ▶ B. Delyon.  
Régression, 2015.  
<https://perso.univ-rennes1.fr/bernard.delyon/regression.pdf>.
- ▶ M. Lejeune.  
*Statistiques, la théorie et ses applications*.  
Springer, 2010.
- ▶ W. McKinney.  
*Python for Data Analysis : Data Wrangling with Pandas, NumPy, and IPython*.  
O'Reilly Media, 2012.