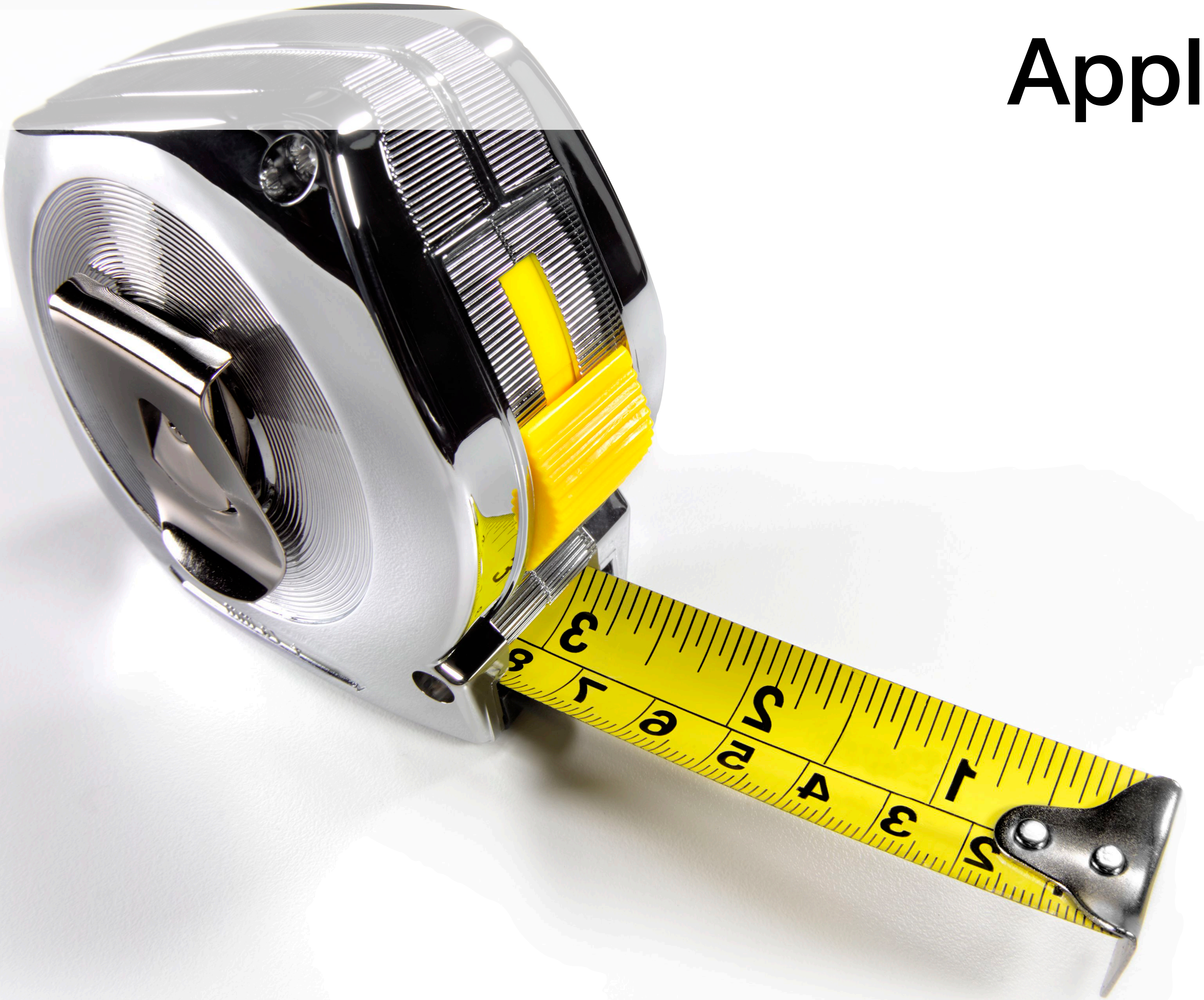
# Distance et approximation

## Applications de la SVD



MAT-2930 Algèbre linéaire appliquée
Jean-François Lalonde

Image

Rappel

# Inversibilité

$$\mathbf{A}^\top \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}^\top \mathbf{b}$$

$$\hat{\mathbf{x}} = (\mathbf{A}^\top \mathbf{A})^{-1}\mathbf{A}^\top \mathbf{b}$$

C’est la matrice pseudo-inverse

Si les colonnes de $\mathbf{A}$ sont linéairement indépendantes, alors $\mathbf{A}^\top\mathbf{A}$ est inversible.

Que faire si les colonnes de $\mathbf{A}$ ne sont pas linéairement indépendantes ?

# Pseudo-inverse

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

$$\mathbf{A}^+ = \mathbf{V}\boldsymbol{\Sigma}^+\mathbf{U}^\top$$

Cela fonctionne peu importe si les colonnes de $\mathbf{A}$ sont indépendantes ou non !

# Moindres carrés $\mathbf{A}^\top \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}^\top \mathbf{b}$

Si les colonnes de $\mathbf{A}$ ne sont pas linéairement indépendantes,
alors $\mathbf{A}^\top\mathbf{A}$ n'est pas inversible et l'équation normale possède une infinité de solutions.

On peut obtenir la solution $\bar{\mathbf{x}}$ ayant la norme minimale grâce à la pseudo-inverse :

$$\bar{\mathbf{x}} = \mathbf{A}^+ \mathbf{b}$$

# Décomposition spectrale

$$\mathbf{A} = \begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_r & \mathbf{u}_{r+1} & \dots & \mathbf{u}_m\end{bmatrix} \begin{bmatrix} \sigma_1 & \dots & 0 & \\ \vdots & \ddots & \vdots & \mathbf{0} \\ 0 & \dots & \sigma_r & \\ & \mathbf{0} & & \mathbf{0} \end{bmatrix} \begin{bmatrix}\mathbf{v}_1 & \dots & \mathbf{v}_r & \mathbf{v}_{r+1} & \dots & \mathbf{v}_n\end{bmatrix}^\top$$

$$= \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \dots + \sigma_r \mathbf{u}_r \mathbf{v}_r^\top$$

# « Contrôler » le rang de A

Par exemple, si je veux obtenir une matrice qui « ressemble » à $\mathbf{A}$ mais qui possède un rang $k < r$ :

$$\mathbf{A} = \begin{bmatrix} \mathbf{u}_1 & \dots & \mathbf{u}_r & \mathbf{u}_{r+1} & \dots & \mathbf{u}_m \end{bmatrix} \begin{bmatrix} \sigma_1 & \dots & 0 & \\ \vdots & \ddots & \vdots & \mathbf{0} \\ 0 & \dots & \sigma_r & \\ & \mathbf{0} & & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 & \dots & \mathbf{v}_r & \mathbf{v}_{r+1} & \dots & \mathbf{v}_n \end{bmatrix}^\top$$

$$\mathbf{A} \approx \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \ldots + \sigma_k \mathbf{u}_k \mathbf{v}_k^\top$$

$$\mathbf{A} \approx \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \ldots + \sigma_k \mathbf{u}_k \mathbf{v}_k^\top$$

# Application : compression

- Une image est une matrice de pixels
- Cette image correspond à une matrice de dimensions $500 \times 423$
- Elle contient 211500 valeurs
- Comment réduire ?

La fonction `scipy.sparse.linalg.svds` permet d'obtenir la SVD partielle

$$\mathbf{A} \approx \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \ldots + \sigma_k \mathbf{u}_k \mathbf{v}_k^\top$$

# Application : compression