Programmation et algèbre linéaire

INFO1 - semaines 47 à 48

Guillaume CONNAN

novembre 2015

IUT de Nantes - Dpt d'informatique

Sommaire

Sommaire

IUT Nantes Informatique


DES VECTEURS,
des maladies

Du latin vector, dérivé de veho (« transporter »)
Organisme qui ne provoque pas lui-même une maladie mais qui disperse l'infection en transportant les agents pathogènes d'un hôte à l'autre.

Informatique

Du latin vector, dérivé de veho (« transporter »)
Organisme qui ne provoque pas lui-même une maladie mais qui disperse l'infection en transportant les agents pathogènes d'un hôte à l'autre.

Die Wissenschaft

der

extensiven Grösse

oder

die Ausdehnungslehre,

eine neue mathematische Disciplin

dargestellt und durch Anwendungen erläutert

von

Hermann Grassmann

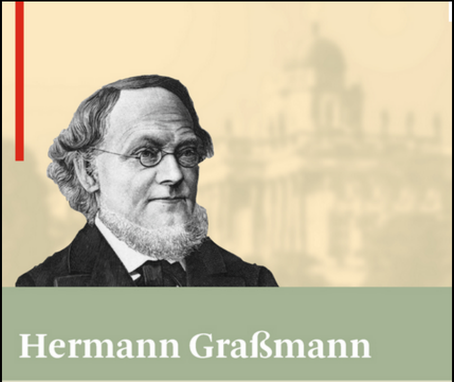
Lehrer an der Friedrich-Wilhelms-Schule zu Stettin.

Erster Theil,

die **lineale Ausdehnungslehre** enthaltend.

Leipzig, 1844.

Verlag von Otto Wigand.


Hermann Graßmann

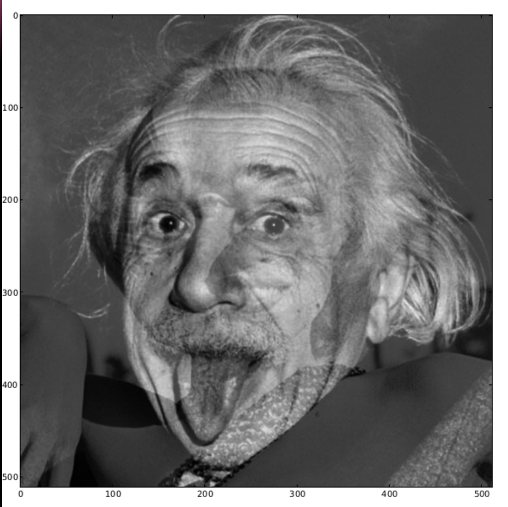

Here as he walked by
on the 16th of October 1843
Sir William Rowan
in a flash of genius discovered
the fundamental formula for
quaternion multiplication
$i^2 = j^2 = k^2 = ijk = -1$
on a stone

3 × Marylin + Albert

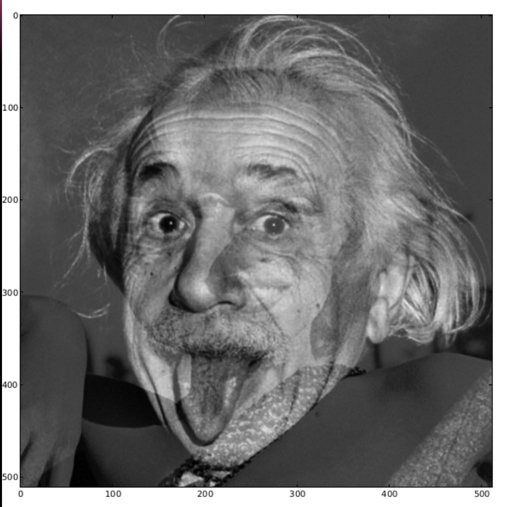
3 × Marylin + Albert

Marylin + Albert

Marylin + Albert



Marylin + 3 × Albert



Marylin + 3 × Albert

Complexe comme la donnée de deux réels

```python
class Complex:
    def __init__(self, reel, imag=0.0):
        self.reel = reel
        self.imag = imag
```

Complexe comme la donnée de deux réels

Il y a une loi de groupe pour les additionner :

```
def __add__(self, other):
        return Complex(self.reel + other.reel, self.imag + other.imag)
```

```
In [28]: Complex(1,2) + Complex(4,10)
Out[28]: Complex(5, 12)
```

Complexe comme la donnée de deux réels

Il y a une loi de groupe pour les additionner :

```
def __add__(self, other):
        return Complex(self.reel + other.reel, self.imag + other.imag)
```

```
In [28]: Complex(1,2) + Complex(4,10)
Out[28]: Complex(5, 12)
```

Complexe comme la donnée de deux réels

Il y a une loi de composition ***externe*** pour multiplier un complexe par un réel :

```
def __mul__(self, k):
        return Complex(self.reel*k, self.imag*k)
```

```
In [29]: Complex(1,2) * 3
Out[29]: Complex(3, 6)
```

```
In [30]: Complex(1,2) * 3 + Complex(5,7)
Out[30]: Complex(8, 13)
```

Complexe comme la donnée de deux réels

Il y a une loi de composition *externe* pour multiplier un complexe par un réel :

```
def __mul__(self, k):
        return Complex(self.reel*k, self.imag*k)
```

```
In [29]: Complex(1,2) * 3
Out[29]: Complex(3, 6)
```

```
In [30]: Complex(1,2) * 3 + Complex(5,7)
Out[30]: Complex(8, 13)
```

IUT NANTES Informatique

Complexe comme la donnée de deux réels

Il y a une loi de composition ***externe*** pour multiplier un complexe par un réel :

```
def __mul__(self, k):
            return Complex(self.reel*k, self.imag*k)
```

```
In [29]: Complex(1,2) * 3
Out[29]: Complex(3, 6)
```

```
In [30]: Complex(1,2) * 3 + Complex(5,7)
Out[30]: Complex(8, 13)
```

```
image = 3*marylin + einstein
imshow(image, cmap = cm.gray)
```

Définition 1 ($\mathbb{K}$-espace vectoriel)

Un $\mathbb{K}$-espace vectoriel V sur un corps commutatif $\langle \mathbb{K}, \oplus, \odot \rangle$ est un groupe abélien $\langle V, \dagger \rangle$ muni d'une loi de composition *externe* $\cdot$ de $\mathbb{K} \otimes V$ dans V vérifiant les quatre axiomes suivant, λ et μ désignant des scalaires quelconques, $\boldsymbol{u}$ et $\boldsymbol{v}$ des vecteurs quelconques et $1_\odot$ l'élément neutre de la loi $\odot$ sur $\mathbb{K}$:

$$(\lambda \oplus \mu) \cdot \boldsymbol{u} = \lambda \cdot \boldsymbol{u} \dagger \mu \cdot \boldsymbol{u} \qquad \text{(distributivité vectorielle)};$$

$$\lambda \cdot (\boldsymbol{u} \dagger \boldsymbol{v}) = \lambda \cdot \boldsymbol{u} \dagger \lambda \cdot \boldsymbol{v} \qquad \text{(distributivité scalaire)};$$

$$(\lambda \odot \mu) \cdot \boldsymbol{u} = \lambda \cdot (\mu \cdot \boldsymbol{u}) \qquad \text{(associativité)};$$

$$1_\odot \cdot \boldsymbol{u} = \boldsymbol{u} \qquad \text{(axiome d'identité)}$$

On dit que le corps $\mathbb{K}$ *opère* sur le groupe $\langle V, \dagger \rangle$.

Sommaire

Une classe Vecteur

```python
class Vecteur:
    def __init__(self, *composantes):
        self.composantes = composantes

    def __str__(self) :
        return str(tuple([c for c in self.composantes]))

    def __repr__(self) :
        return 'Vecteur' + str(self)
```

Une classe Vecteur

```python
def __add__(self, other) :
        c1, c2 = self.composantes, other.composantes
        l1, l2 = len(c1), len(c2)
        assert l1 == l2, "Tailles incompatibles"
        s = ....????
        return Vecteur( *(x for x in s) )
```

```
In [37]: Vecteur(1,2,3,4,12) + Vecteur(2,41,10,-1,0)
Out[37]: Vecteur(3, 43, 13, 3, 12)
```

Une classe Vecteur

```
def __add__(self, other) :
        c1, c2 = self.composantes, other.composantes
        l1, l2 = len(c1), len(c2)
        assert l1 == l2, "Tailles incompatibles"
        s = ....????
        return Vecteur( *(x for x in s) )
```

```
In [37]: Vecteur(1,2,3,4,12) + Vecteur(2,41,10,-1,0)
Out[37]: Vecteur(3, 43, 13, 3, 12)
```

Une classe Vecteur

```
In [36]: Vecteur(1,2,3,4) + Vecteur(2,41,10,-1,0)
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-36-86a61c4cf67e> in <module>()
----> 1 Vecteur(1,2,3,4) + Vecteur(2,41,10,-1,0)

/home/moi/PROG/PYTHON/Matrix/vector.py in __add__(self, other)
     12     #     return Complex(-self.reel, -self.imag)
     13
---> 14
     15     # def __sub__(self, other):
     16     #     return self + (- other)

AssertionError: Tailles incompatibles
```

Une classe Vecteur

```
def __mul__(self, k) :
    return ????

def __neg__(self) :
    return ????

def __sub__(self, other) :
    return ????

def __eq__(self, other):
    ???
```

Cette classe peut s'appliquer à toute classe ayant une structure de corps.
Définissez une classe pour représenter le corps $\mathbb{F}_2$

Cette classe peut s'appliquer à toute classe ayant une structure de corps.
Définissez une classe pour représenter le corps $\mathbb{F}_2$

```python
class F2:
    def __init__(self,val) :
        assert val in {0,1}, "Le représentant doit être dans {0,1}"
        self.val = val

    def __str__(self) :
        return 'O' if self.val == 0 else 'I'

    def __repr__(self) :
        return str(self)

    def __add__(self, other) :
        return ???????????

    def __mul__(self, other) :
        return ??????????

    def __neg__(self) :
        return ???????????

    def __sub__(self, other) :
        return ???????????

    def __truediv__(self, other) :
        ??????????
        return ?????????
```

Alors :

```
In [61]: v1 = Vecteur(F2(0), F2(1), F2(1), F2(1))

In [62]: v1
Out[62]: Vecteur(0, I, I, I)

In [63]: v2 = Vecteur(F2(1), F2(1), F2(0), F2(1))

In [64]: v2
Out[64]: Vecteur(I, I, 0, I)

In [65]: v1 + v2
Out[65]: Vecteur(I, 0, I, 0)
```

Sommaire

Produit scalaire

Soit deux vecteurs $\boldsymbol{u}$ et $\boldsymbol{v}$ de $\mathbb{K}^n$.
Alors le produit scalaire de ces deux vecteurs est défini par :

$$\boldsymbol{u} \cdot_{\mathbb{K}^n} \boldsymbol{v} = \sum_{k=1}^{n} \boldsymbol{u}[k] \times_{\mathbb{K}} \boldsymbol{v}[k]$$

NB

Le produit scalaire de deux vecteurs est un...scalaire et surtout pas un vecteur.

Produit scalaire

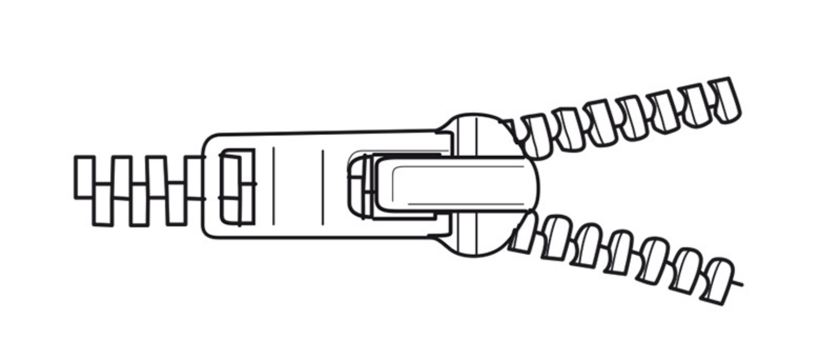
Soit deux vecteurs $\boldsymbol{u}$ et $\boldsymbol{v}$ de $\mathbb{K}^n$.
Alors le produit scalaire de ces deux vecteurs est défini par :

$$\boldsymbol{u} \cdot_{\mathbb{K}^n} \boldsymbol{v} = \sum_{k=1}^{n} \boldsymbol{u}[k] \times_{\mathbb{K}} \boldsymbol{v}[k]$$

NB

Le produit scalaire de deux vecteurs est un...scalaire et surtout pas un vecteur.

Fermeture éclair avec



Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.

Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.

On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.

On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes. On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.

On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

iut NANTES Informatique

Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.
Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.
On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.
On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes. On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.
On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.
Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.
On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.
On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes.On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.
On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.
Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.
On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.
On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes. On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.
On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.
Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.
On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.
On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes.On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.
On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

Un vecteur comme un dictionnaire

Nous ne sommes pas obligés d'indicer les composantes d'un vecteur avec des entiers.
Par exemple, supposons que vous effectuiez votre stage chez une société de fret qui transporte des déchets nucléaires, des bombes atomiques, des missiles chimiques, des mines anti-personnel, des knackys-herta.
On peut travailler dans un ensemble isomorphe à $\mathbb{R}^5$ dont les composantes sont les armes précédentes dans l'ordre.
On peut s'intéresser au vecteur qui associe les prix au kilo des armes précédentes.On va utiliser la structure de dictionnaire qui permet de représenter les fonctions de domaine fini.
On peut aussi s'intéresser au vecteur qui associe les quantités achetées.

```python
prix = {'centrale' : 100, 'bombe' : 200, 'missile' : 150, 'mine' : 1, 'knacky' :
    300}
```

```python
quantités = {'centrale' : 2, 'bombe' : 3, 'missile' : 10, 'mine' : 100, 'knacky'
    : 20}
```

Comment obtenir le prix total des quantités achetées ?

```python
prix = {'centrale' : 100, 'bombe' : 200, 'missile' : 150, 'mine' : 1, 'knacky' :
    ↪ 300}
```

```python
quantités = {'centrale' : 2, 'bombe' : 3, 'missile' : 10, 'mine' : 100, 'knacky'
    ↪ : 20}
```

Comment obtenir le prix total des quantités achetées ?

```
prix = {'centrale' : 100, 'bombe' : 200, 'missile' : 150, 'mine' : 1, 'knacky' :
    ↪ 300}
```

```
quantités = {'centrale' : 2, 'bombe' : 3, 'missile' : 10, 'mine' : 100, 'knacky'
    ↪ : 20}
```

Comment obtenir le prix total des quantités achetées ?

```
In [93]: prix = Vec(domaine, {'centrale' : 100, 'bombe' : 200, 'missile' : 150,
    ↪   'mine' : 1, 'knacky' : 300})

In [94]: quantités = Vec(domaine,  {'centrale' : 2, 'bombe' : 3, 'missile' : 10,
    ↪   'mine' : 100, 'knacky' : 20})

In [95]: prix.dot(quantités)
Out[95]: 8400
```

IUT Nantes Informatique

Le squelette de la classe des vecteurs définis par des dictionnaires :

```
class Vec:
    def __init__(self, domaine, dico):
        self.domaine = domaine
        self.dico = dico

    def __str__(self) :
        return str(self.dico)

    def __repr__(self) :
        return 'Vecteur' + str(self)

    def __add__(self, other) :
        ??????

    def __mul__(self, k) :
        return ?????

    def __neg__(self) :
        return ??????

    def __eq__(self, other):
        return ????

    def dot(self, other) :
        ???
```

Sommaire

IUT Nantes Informatique

Définition 2 (Combinaison linéaire de vecteurs)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ un corps opérant sur le groupe $\langle V, \dagger \rangle$ pour former un espace vectoriel de produit externe $\bullet$.

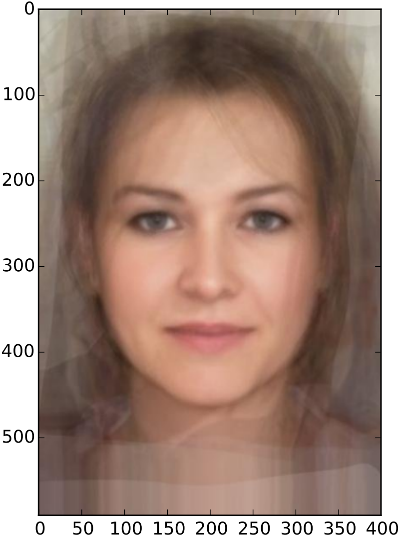
Soit $\langle \boldsymbol{u_0}, \boldsymbol{u_1}, \cdots, \boldsymbol{u_p} \rangle$ une famille de vecteurs de V.

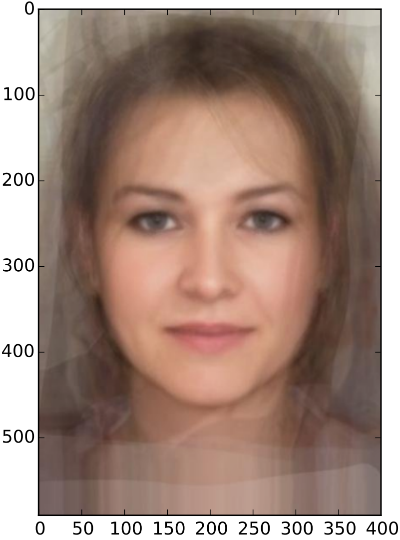
Un vecteur $\boldsymbol{v}$ de V est une combinaison linéaire de la famille des $\langle \boldsymbol{u_i} \rangle_{0 \leqslant i \leqslant p}$ si, et seulement si, il existe une famille $\langle \lambda_0, \lambda_1, \cdots, \lambda_p \rangle$ de scalaires de $\mathbb{K}$ vérifiant :

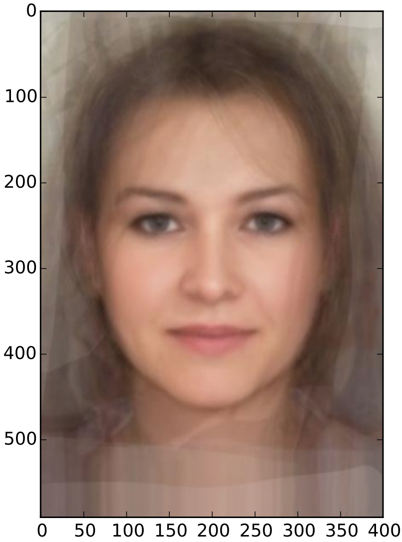
$$\boldsymbol{v} = \mathop{\dagger}_{i=0}^{i=p} \lambda_i \bullet \boldsymbol{u_i} = \lambda_0 \bullet \boldsymbol{u_0} \dagger \lambda_1 \bullet \boldsymbol{u_1} \dagger \cdots \dagger \lambda_p \bullet \boldsymbol{u_p}$$

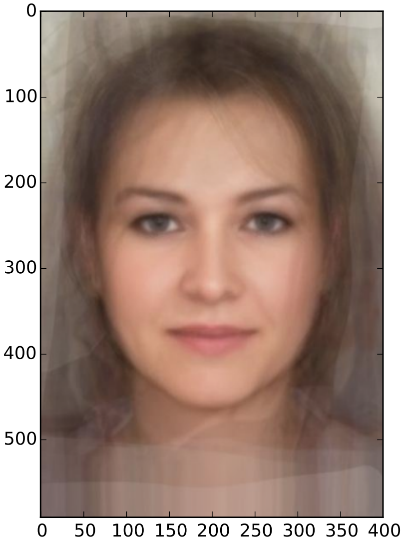
L'ensemble de ces combinaisons linéaires est noté :

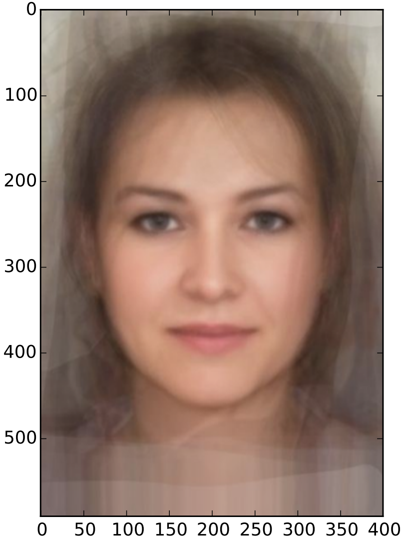
$$Vect\{\boldsymbol{u_0}, \boldsymbol{u_1}, \cdots, \boldsymbol{u_p}\}$$

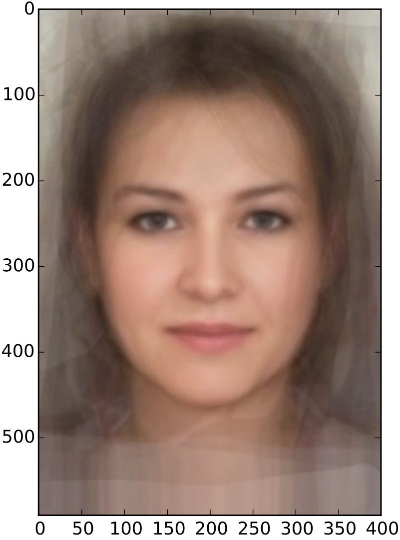

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

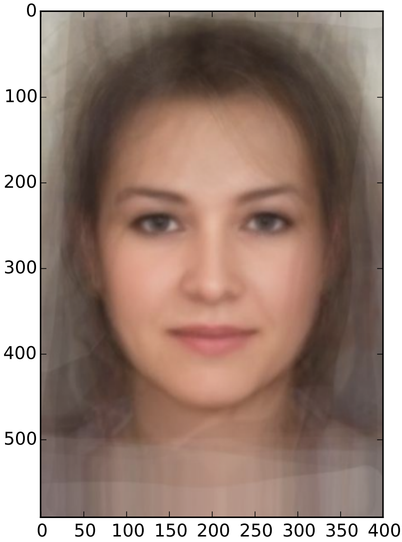

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

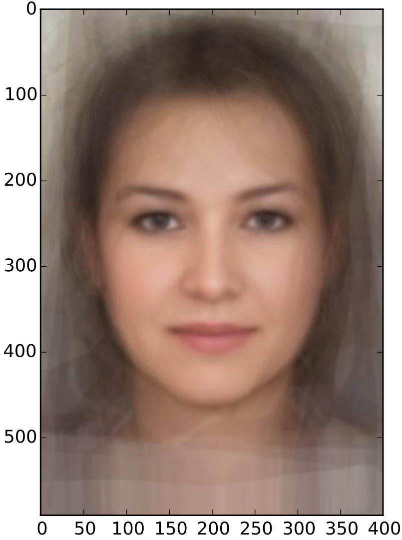

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

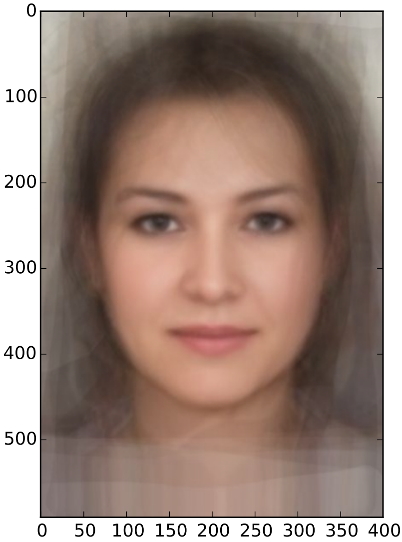

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

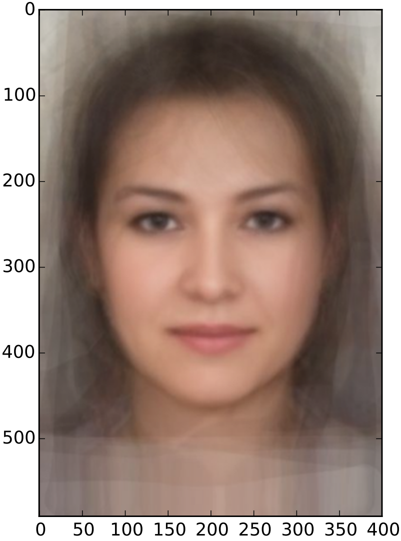

0
100
200
300
400
500
0
50
100
150
200
250
300
350
400

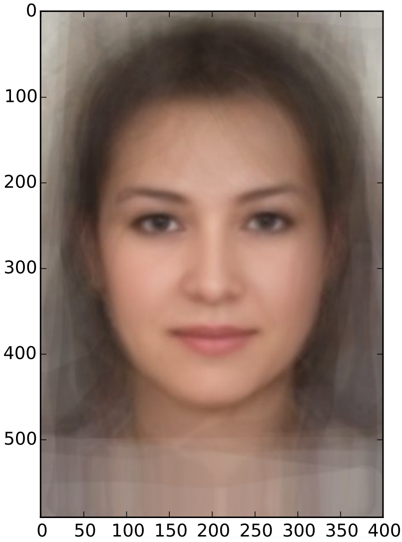

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

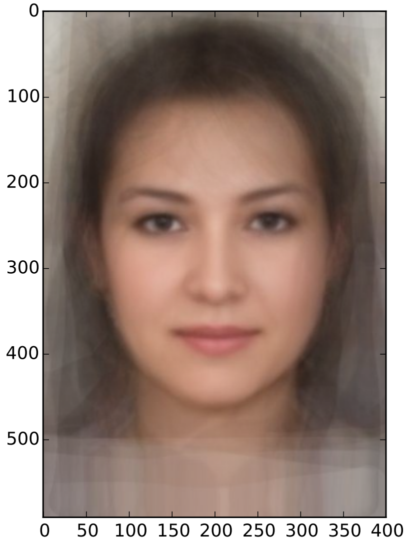

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

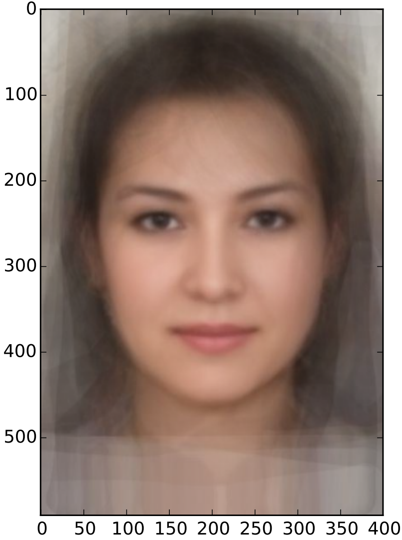

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

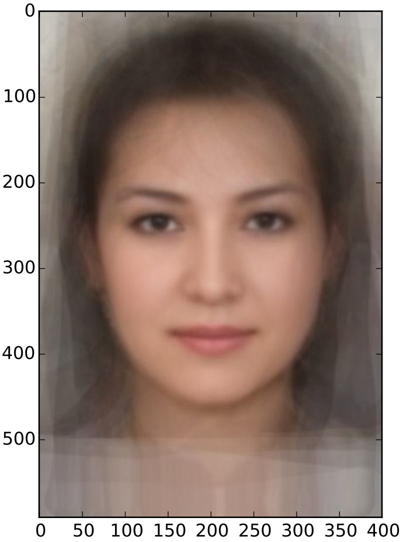

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

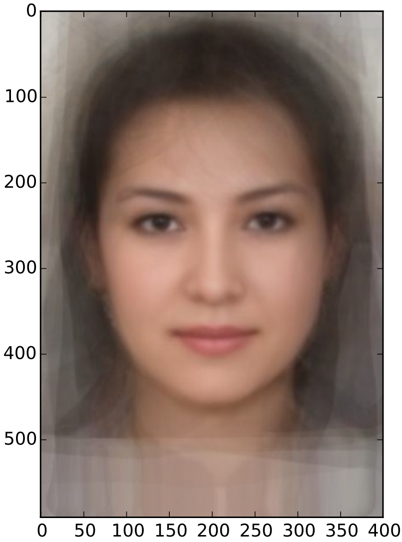

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

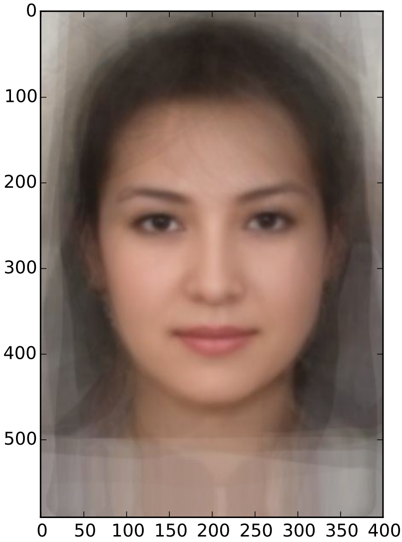

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

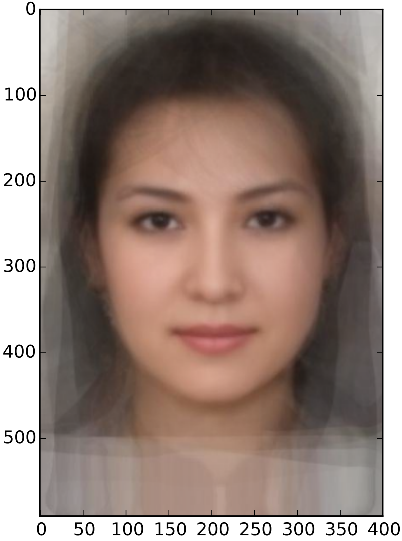

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

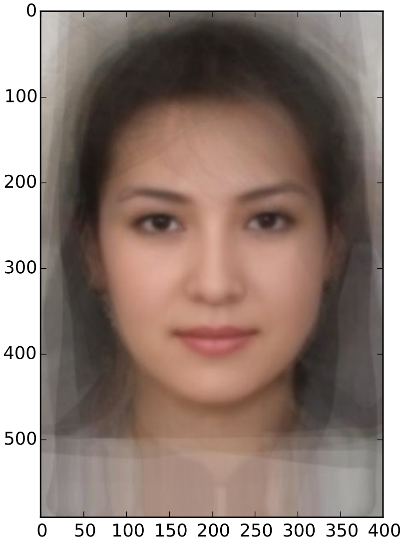

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

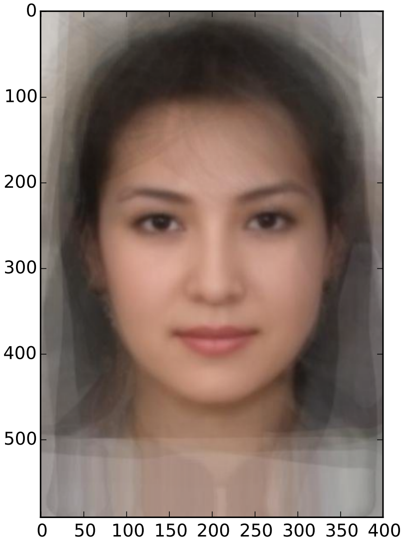

0
100
200
300
400
500
0 50 100 150 200 250 300 350 400

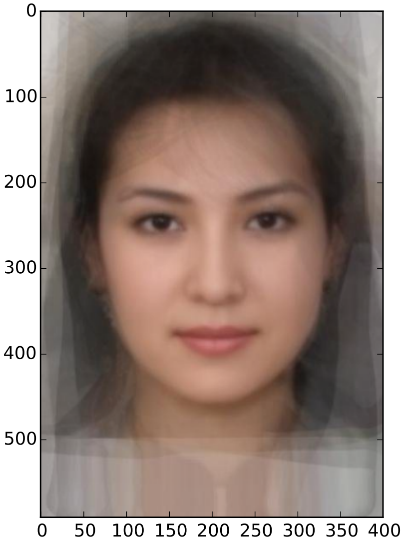

0
100
200
300
400
500
0
50
100
150
200
250
300
350
400

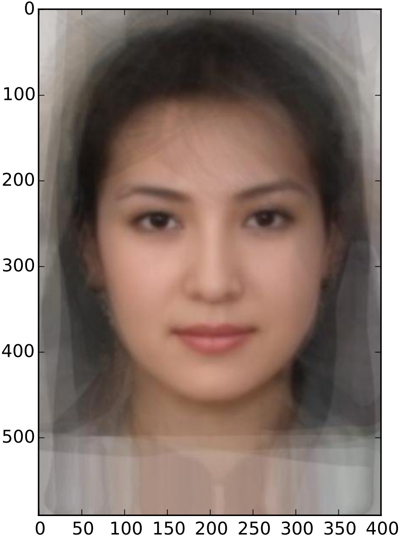

0
100
200
300
400
500
0
50
100
150
200
250
300
350
400


0
100
200
300
400
500
0 50 100 150 200 250 300 350 400


0
100
200
300
400
500
0
50
100
150
200
250
300
350
400


0
100
200
300
400
500
0 50 100 150 200 250 300 350 400


0
100
200
300
400
500
0 50 100 150 200 250 300 350 400


0
100
200
300
400
500
0
50
100
150
200
250
300
350
400

```python
import matplotlib.image as mpimg
import matplotlib.pyplot as plt

w = mpimg.imread('welsh.png')[:591,:,:]
o = mpimg.imread('ouzbek.png')

def fondu(n) :
    for k in range(n) :
        t = k / float(n)
        plt.imshow(o*t + (1 - t)*w)
        plt.savefig('ow' + str(k) + '.pdf')
```

```
\multido{\n=0+1}{21}{%
  \begin{frame}
    \begin{center}
      \includegraphics[height=\textheight]{ow\n}
    \end{center}
  \end{frame}
}
```

Sommaire

Définition 3 (Sev)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ un corps opérant sur le groupe $\langle V, \dagger \rangle$ pour former un espace vectoriel de produit externe $\bullet$.
Soit W un sous-ensemble de V.
Si, muni des mêmes opérations que V, W a une structure de $\mathbb{K}$-espace vectoriel, alors on dit que W est un **sous-espace vectoriel** de V.

Théorème 4 (Caractérisation (1) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ *un corps opérant sur le groupe* $\langle V, \dagger \rangle$ *pour former un espace vectoriel de produit externe* $\bullet$*.*
Soit W *un sous-ensemble de* V*.*
W *est un sous-espace vectoriel (sev) de* V *si, et seulement si :*

- $W \neq \varnothing$
- $(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \boldsymbol{u}, \boldsymbol{v} \rangle \in W^2 \rightarrow \boldsymbol{u} \dagger \boldsymbol{v} \in W)$
- $(\forall \lambda)(\forall \boldsymbol{u})(\langle \lambda, \boldsymbol{u} \rangle \in \mathbb{K} \otimes W \rightarrow \lambda \bullet \boldsymbol{u} \in W)$

Théorème 4 (Caractérisation (1) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ *un corps opérant sur le groupe* $\langle V, \dagger \rangle$ *pour former un espace vectoriel de produit externe* $\bullet$.
Soit W *un sous-ensemble de* V.
W *est un sous-espace vectoriel (sev) de* V *si, et seulement si :*

- $W \neq \varnothing$
- $(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \boldsymbol{u}, \boldsymbol{v} \rangle \in W^2 \rightarrow \boldsymbol{u} \dagger \boldsymbol{v} \in W)$
- $(\forall \lambda)(\forall \boldsymbol{u})(\langle \lambda, \boldsymbol{u} \rangle \in \mathbb{K} \otimes W \rightarrow \lambda \bullet \boldsymbol{u} \in W)$

Théorème 4 (Caractérisation (1) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ un corps opérant sur le groupe $\langle V, \dagger \rangle$ pour former un espace vectoriel de produit externe $\bullet$.
Soit W un sous-ensemble de V.
W est un sous-espace vectoriel (sev) de V si, et seulement si :

- $W \neq \varnothing$
- $(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \boldsymbol{u}, \boldsymbol{v} \rangle \in W^2 \rightarrow \boldsymbol{u} \dagger \boldsymbol{v} \in W)$
- $(\forall \lambda)(\forall \boldsymbol{u})(\langle \lambda, \boldsymbol{u} \rangle \in \mathbb{K} \otimes W \rightarrow \lambda \bullet \boldsymbol{u} \in W)$

Théorème 4 (Caractérisation (1) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ *un corps opérant sur le groupe* $\langle V, \dagger \rangle$ *pour former un espace vectoriel de produit externe* $\bullet$.
Soit W *un sous-ensemble de* V.
W *est un sous-espace vectoriel (sev) de* V *si, et seulement si :*

- $W \neq \varnothing$
- $(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \boldsymbol{u}, \boldsymbol{v} \rangle \in W^2 \to \boldsymbol{u} \dagger \boldsymbol{v} \in W)$
- $(\forall \lambda)(\forall \boldsymbol{u})(\langle \lambda, \boldsymbol{u} \rangle \in \mathbb{K} \otimes W \to \lambda \bullet \boldsymbol{u} \in W)$

Théorème 5 (Caractérisation (2) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ *un corps opérant sur le groupe* $\langle V, \dagger \rangle$ *pour former un espace vectoriel de produit externe* $\bullet$.
Soit W *un sous-ensemble de* V.
W *est un sous-espace vectoriel (sev) de* V *si, et seulement si :*

- $0 \in W$
- $(\forall \lambda)(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \lambda, \boldsymbol{u}, \boldsymbol{v} \rangle \in \mathbb{K} \otimes W \otimes W \to \boldsymbol{u} \dagger \lambda \bullet \boldsymbol{v} \in W)$

Théorème 5 (Caractérisation (2) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ un corps opérant sur le groupe $\langle V, \dagger \rangle$ pour former un espace vectoriel de produit externe $\bullet$.
Soit W un sous-ensemble de V.
W est un sous-espace vectoriel (sev) de V si, et seulement si :

- $0 \in W$
- $(\forall \lambda)(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \lambda, \boldsymbol{u}, \boldsymbol{v} \rangle \in \mathbb{K} \otimes W \otimes W \to \boldsymbol{u} \dagger \lambda \bullet \boldsymbol{v} \in W)$

Informatique

Théorème 5 (Caractérisation (2) des sous-espaces vectoriels)

Soit $\langle \mathbb{K}, \oplus, \odot \rangle$ un corps opérant sur le groupe $\langle V, \dagger \rangle$ pour former un espace vectoriel de produit externe $\bullet$.
Soit W un sous-ensemble de V.
W est un sous-espace vectoriel (sev) de V si, et seulement si :

- $0 \in W$
- $(\forall \lambda)(\forall \boldsymbol{u})(\forall \boldsymbol{v})(\langle \lambda, \boldsymbol{u}, \boldsymbol{v} \rangle \in \mathbb{K} \otimes W \otimes W \rightarrow \boldsymbol{u} \dagger \lambda \bullet \boldsymbol{v} \in W)$

IUT Nantes Informatique

Sommaire

IUT Nantes Informatique