



Teorie kognitivních systémů

X Redukce dimenzionality

- Motivace ke snižování dimenzionality
- Principal Component Analysis (PCA)
- Princip činnosti PCA
- Algoritmus PCA
- Vlastnosti PCA
- Použití PCA a případy, kdy použití není vhodné





Redukce dimenzionality dat

Učení bez učitele

- **redukce dimenzionality** (*Dimensionality Reduction*) je problém učení bez učitele (*Unsupervised Learning*)
- cílem je **kompresa dat** → snížení paměťové náročnosti algoritmu, snížení výpočetní zátěže, následkem toho **urychljení vykonávání algoritmu** (v případě strojového učení velmi důležité – některé algoritmy běží hodiny, dny i týdny)
- komprese je možná, jsou-li data redundantní (poznatek z TI)
→ redundancy je zřejmě i **korelace dat**: jsou-li data korelovaná, pak není nezbytné uchovávat vše, ale pouze zásadní komponenty, které nesou informaci a z nichž lze ostatní výpočtem odvodit
- korelovaná data také často způsobují problémy související s přeurozením úlohy (např. v neuronových sítích) → žádoucím jevem je tedy kromě komprese i **odstranění korelace**





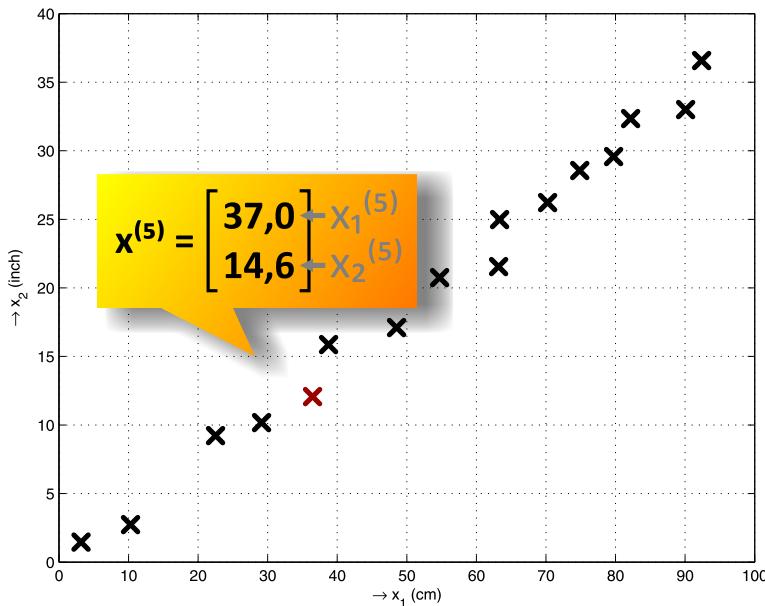
Motivace: Komprese dat

Příklad redukce z 2D do 1D

Mějme množinu údajů $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}$, které vznikly např. zjišťováním délky krku volavek afrických dvěma týmy ornitologů – evropský tým měřil v centimetrech, americký tytéž volavky v palcích (což ovšem nevíme):

- data jsou zřejmě vysoce korelovaná
- lineární funkci ne-tvoří proto, že měření jsou zatížena chybami (v Africe jsou nekvalitní právítka)

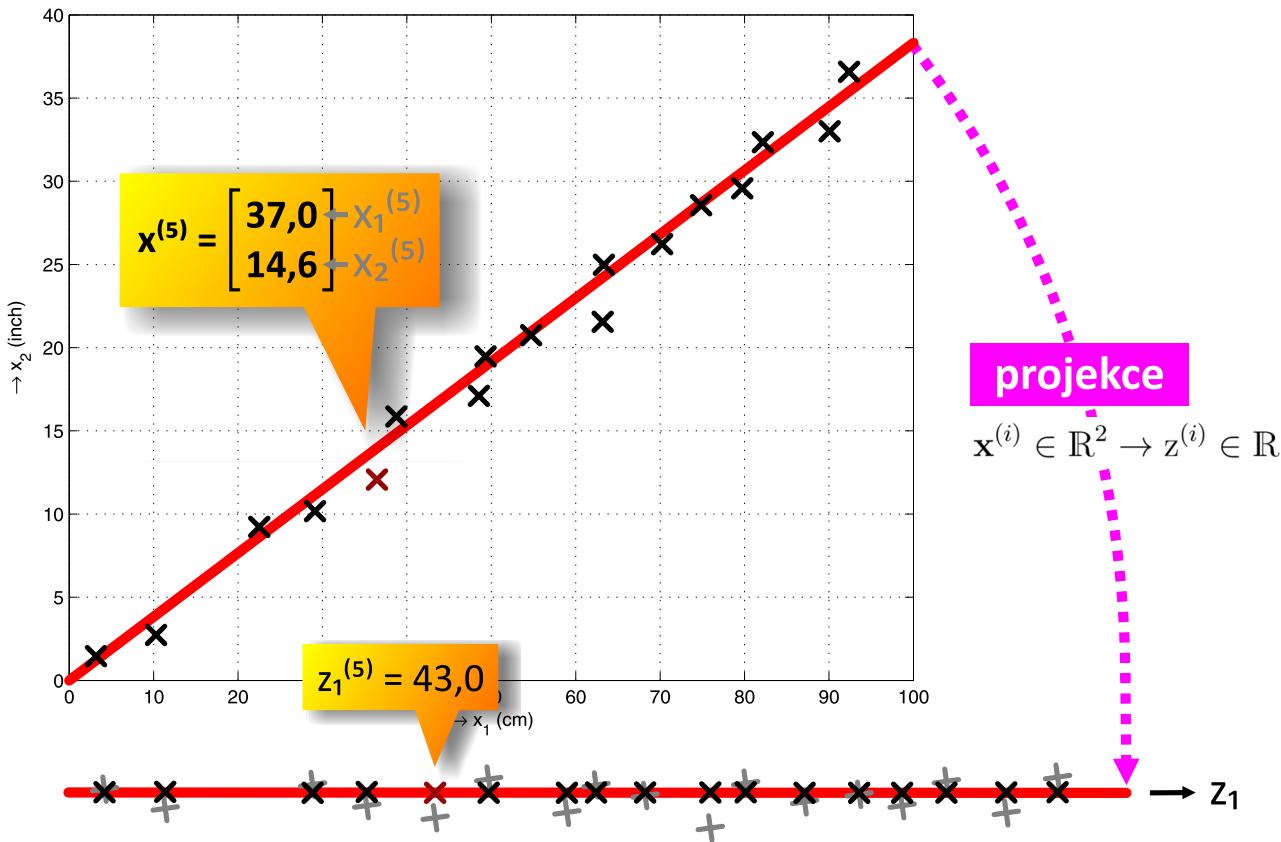
Proč ukládat 2 údaje, když 1 nese dostatečnou informaci?





Motivace: Komprese dat

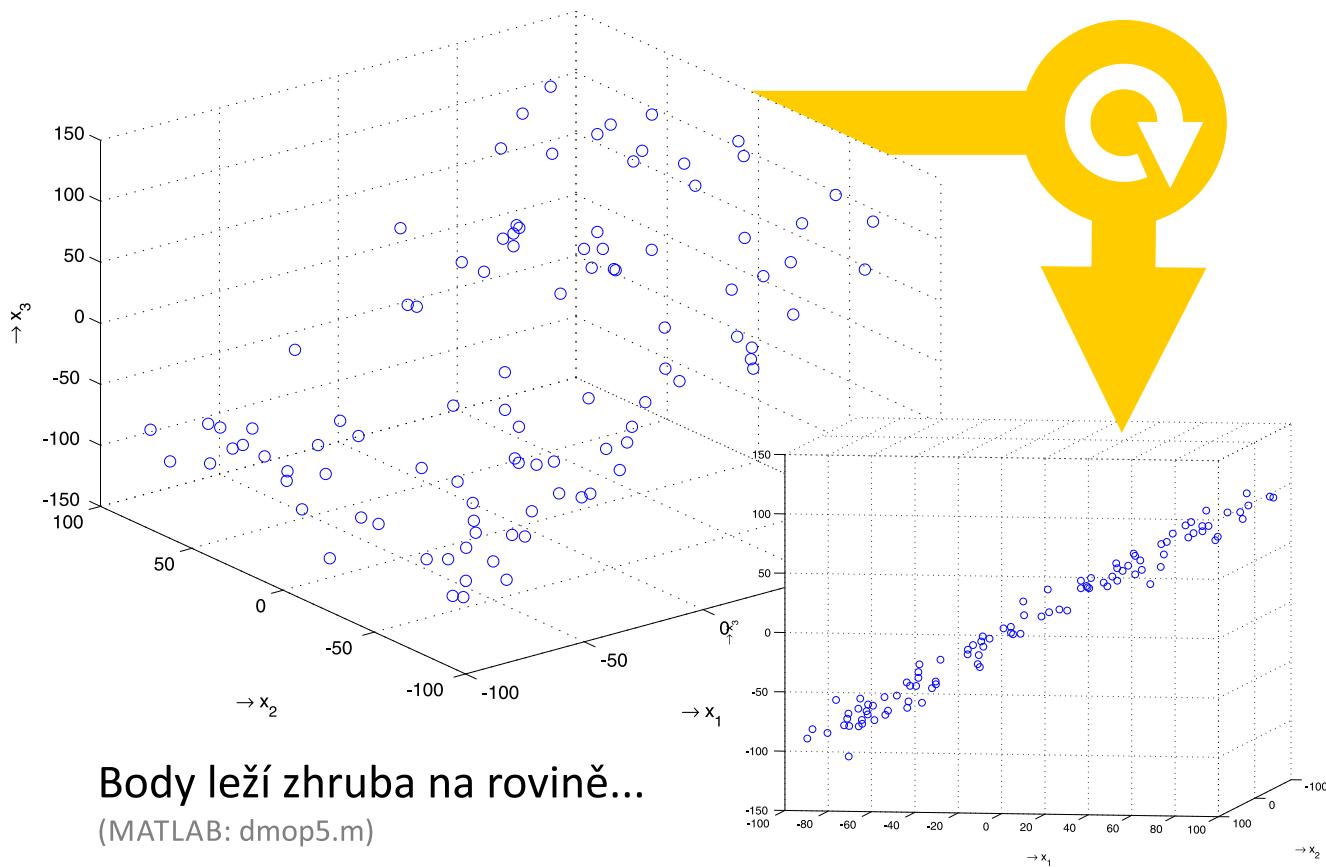
Příklad redukce dat z 2D do 1D





Motivace: Komprese dat

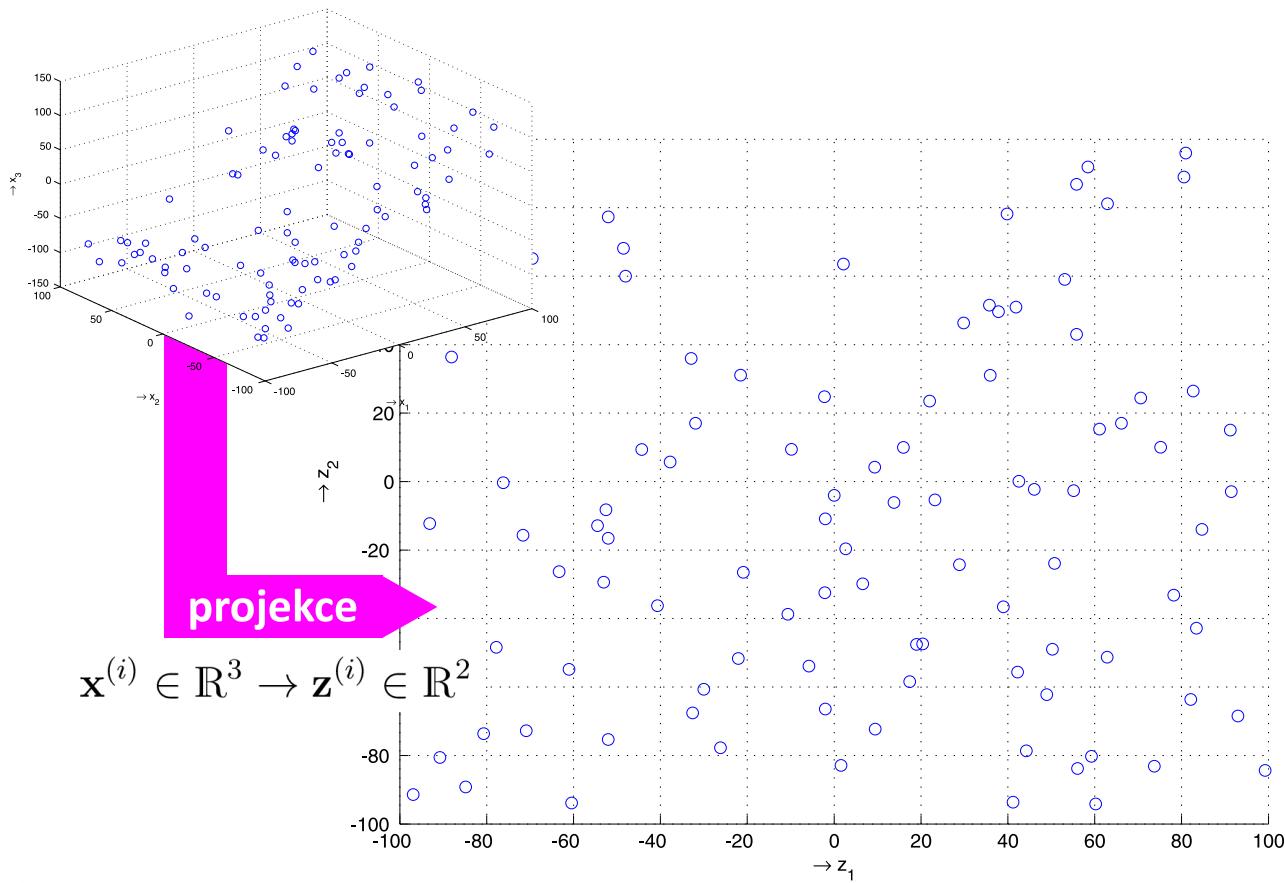
Příklad redukce dat z 3D do 2D





Motivace: Komprese dat

Příklad redukce dat z 3D do 2D





Motivace: Vizualizace či pochopení dat

Příklad použití redukce dimenzionality

- **Poznání charakteru dat** – mnoharozměrná data si nelze představit, nakreslit, zobrazit → vývojář inteligentního systému nemá jasnou představu o charakteru těchto dat, a proto neví, jakou zvolit metodu jejich zpracování (filtrace)
- **Lokalizace extrémů** – v mnoharozměrných datech není na první pohled patrné, které body jsou např. mimo očekávané hranice, nesplňují předpoklady o statistickém rozdělení veličiny, apod.
- **Vizualizace** – často je potřeba data vykreslit na obrazovku, což zřejmě např. u bodů z \mathbb{R}^{100} je problém, typickým příkladem jsou třeba vícekanálová měření EEG – zobrazení ve 2D po jednotlivých kanálech vede nutně ke zkreslené představě o charakteru dat...



Motivace: Vizualizace či pochopení dat

Příklad použití redukce dimenzionality

X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 X_9 X_{10} ...

| Helicopter Model | Max Speed | Cruise Speed | Range | Service Ceiling | Hovering Ceiling | Rate of Climb | Empty Weight | Max Takeoff Weight | Powerplants | Powerplant Output | Crew |
|--------------------------|-----------|--------------|-------|-----------------|------------------|---------------|--------------|--------------------|-------------|-------------------|------|
| (Factory and model name) | [km/h] | [km/h] | [km] | [m] | [m] | [m/s] | [kg] | [kg] | [1] | [kW] | [1] |
| PZL Kania | 215 | 190 | 493 | 4000 | 1375 | 8,75 | 2000 | 3550 | 2 | 313 | |
| MBB/Kawasaki BK117 | 278 | 250 | 541 | 4575 | 3565 | 11,00 | 1727 | 3350 | 2 | 442 | |
| Bell 412 | 259 | 226 | 745 | 6096 | 4250 | 6,86 | 3079 | 5397 | 2 | 671 | |
| Bell 214 | 291 | 260 | 475 | 5000 | 4370 | 7,45 | 3442 | 6805 | 1 | 2185 | |
| Agusta A109 | 285 | 250 | 490 | 6000 | 3785 | 9,80 | 2000 | 2850 | 2 | 423 | |
| Eurocopter EC135 | 287 | 254 | 635 | 6096 | 3525 | 7,62 | 1455 | 2910 | 2 | 473 | |

(data z <http://en.wikipedia.org>)

Redukce dimenzionality dat z 10D do D2:

$$\mathbf{x}^{(i)} \in \mathbb{R}^{10} \rightarrow \mathbf{z}^{(i)} \in \mathbb{R}^2$$

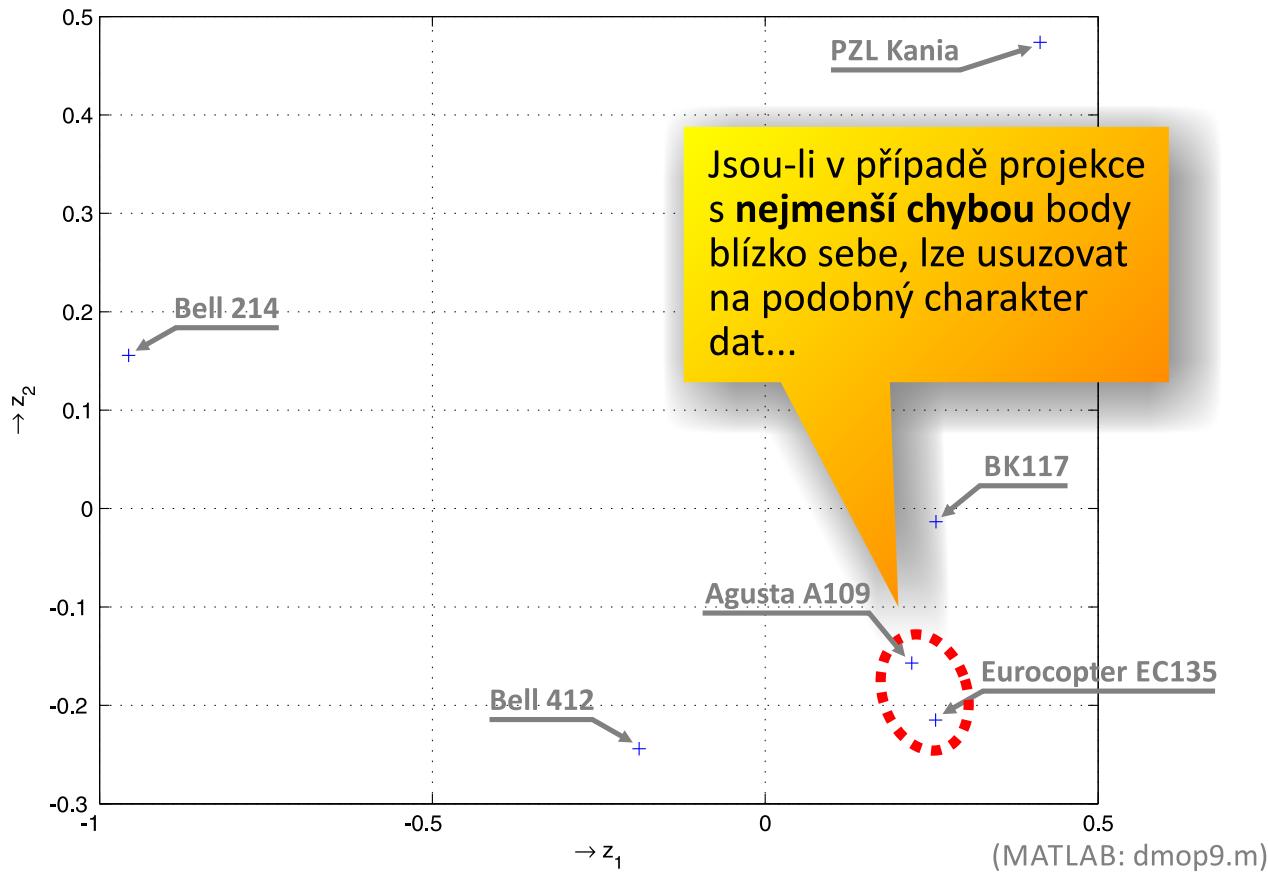
toto lze vykreslit

- výsledné body ve 2D samozřejmě **nelze interpretovat ve smyslu původních charakteristik dat** (v tomto příkladě technických dat vrtulníků)



Motivace: Vizualizace či pochopení dat

Příklad použití redukce dimenziality





Principal Component Analysis (PCA)

Algoritmus pro redukci dimenzionality dat

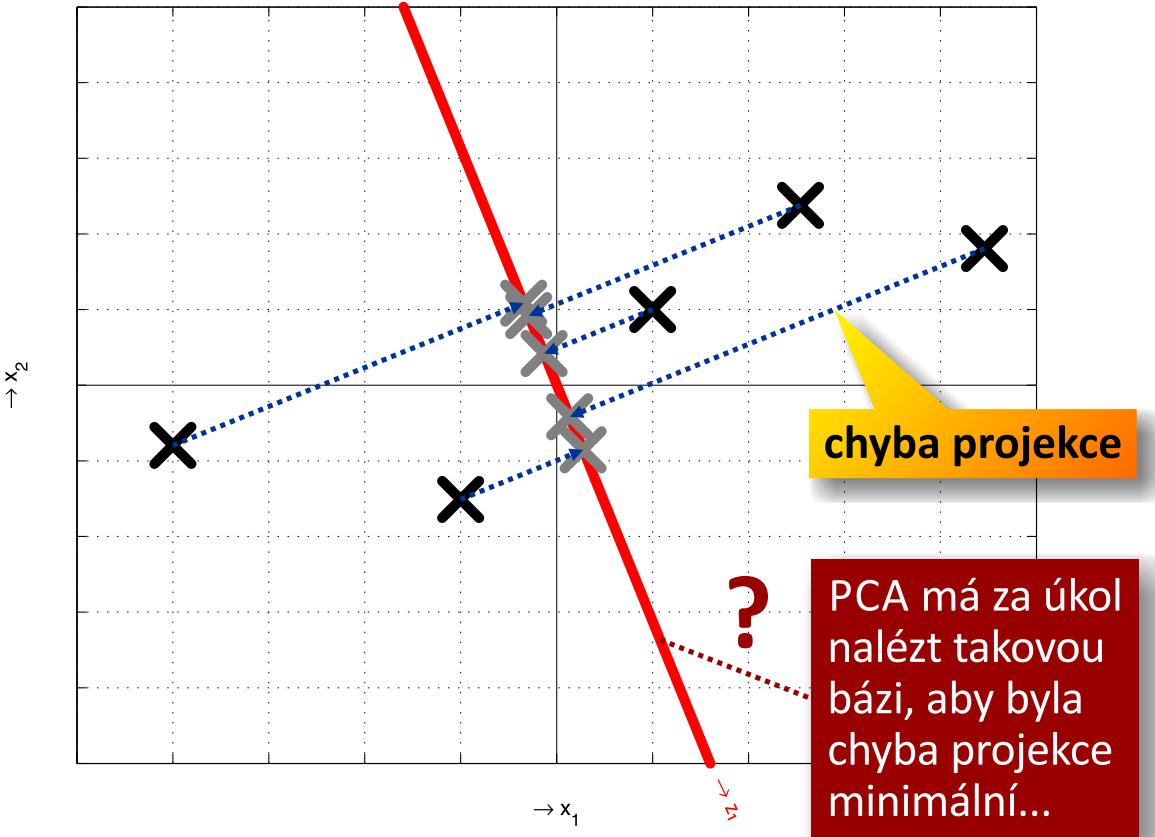
- projekci z prostoru o n rozměrech do podprostoru o k rozměrech ($k < n$) lze provádět mnoha způsoby – jsou-li data původně ryze n -rozměrná (tj. nebyla doplněna např. konstantou či neutrálním prvkem), pak dojde při projekci k jisté ztrátě dat, tzv. **chybě projekce (Projection Error)**
- jedním z algoritmů, který provádí projekci s minimální chybou, je **PCA** – jeden z nejoblíbenějších a nejpoužívanějších algoritmů pro redukci dimenzionality dat
- navrhl ho roku 1901 Karl Pearson; znám také jako diskrétní Karhunen-Loèveova transformace (KLT), Hotellingova transformace, či ryze ortogonální transformace (POD = *Proper Orthogonal Transform*); česky **analýza hlavních komponent**
- je jednoduchý, rychlý, **stabilní**
- **má jistá omezení**, resp. klade určité požadavky na data





Principal Component Analysis (PCA)

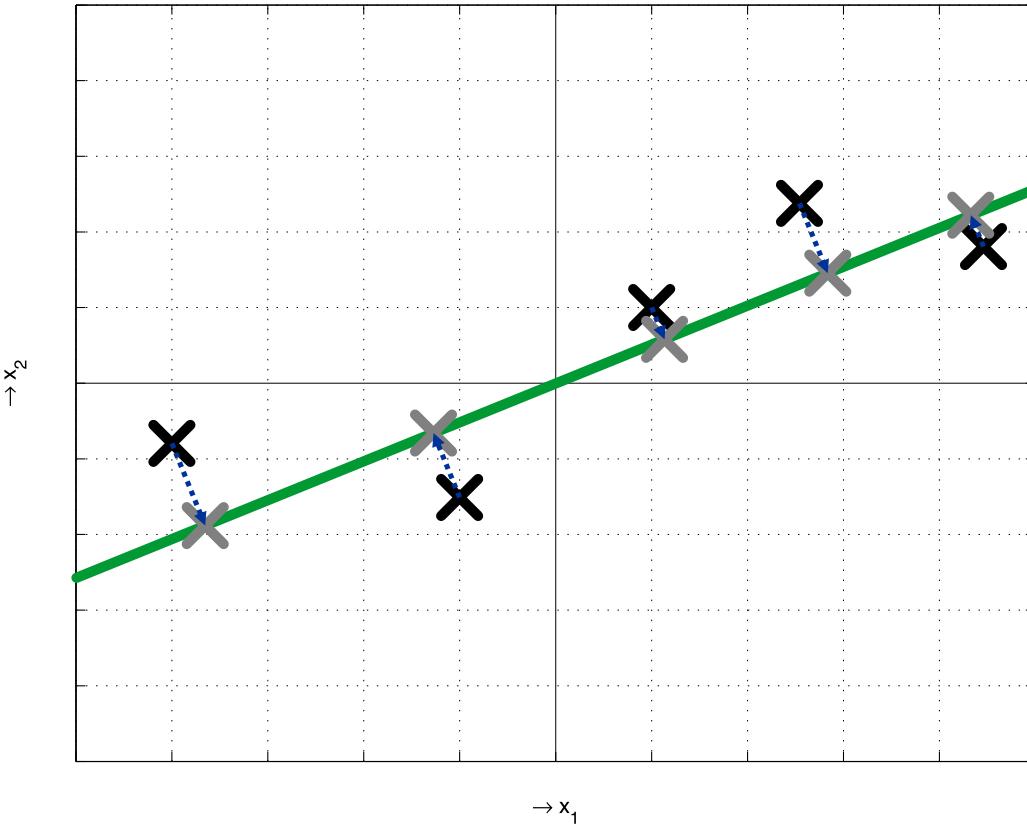
Formulace problému





Principal Component Analysis (PCA)

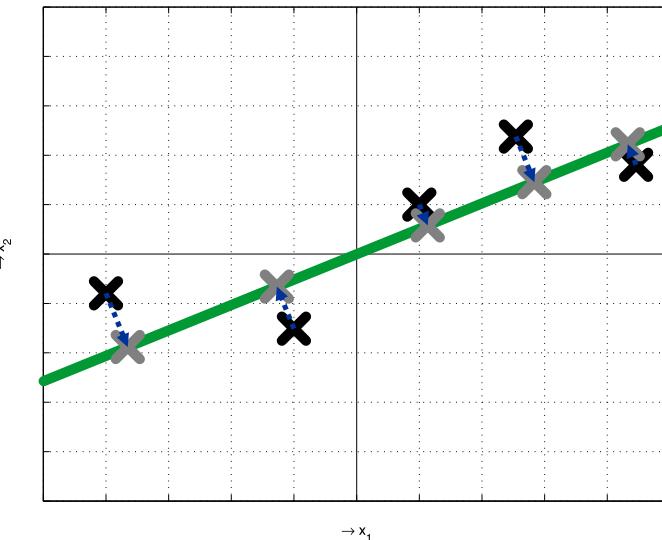
Formulace problému





Principal Component Analysis (PCA)

Formulace problému



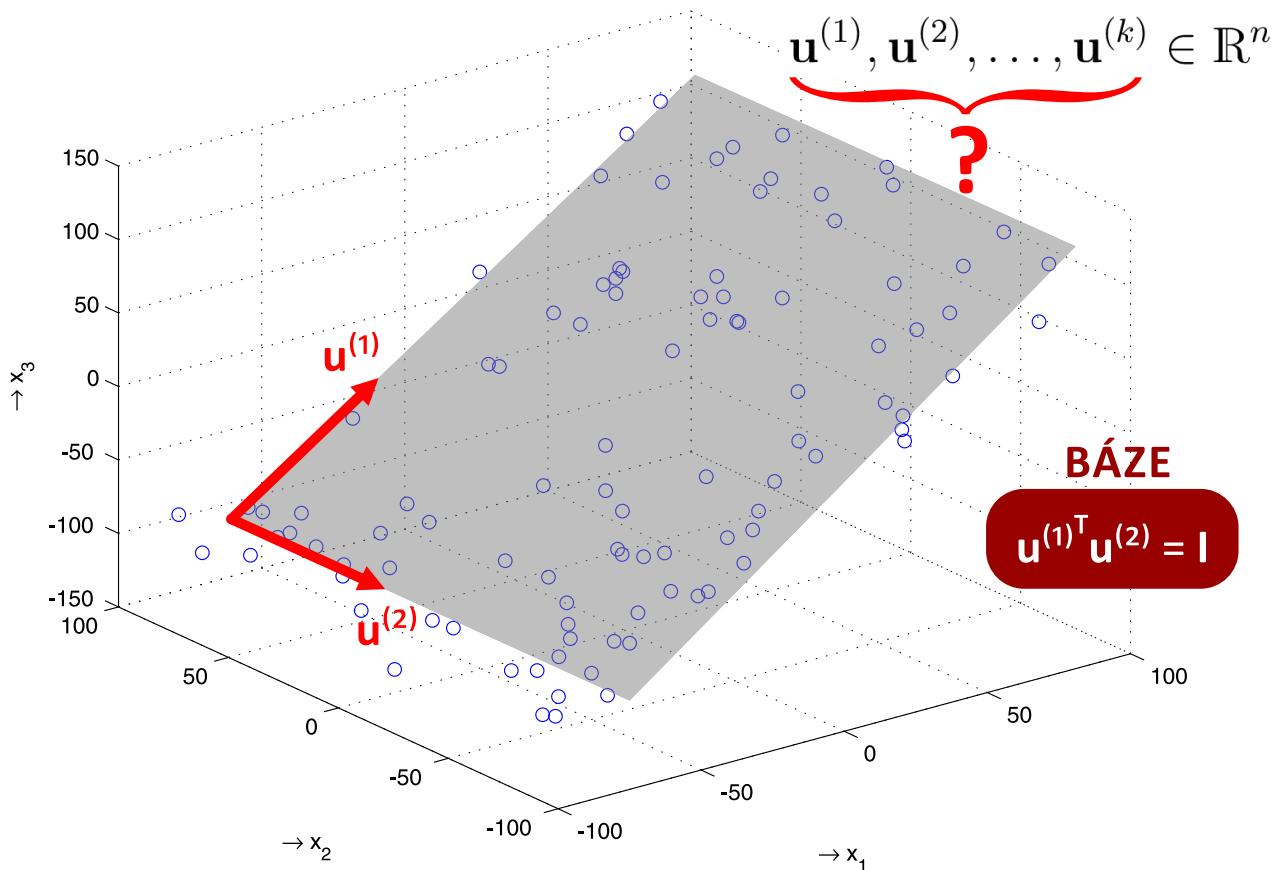
2D → 1D: Nalézt sklon přímky (**bázový vektor** $\mathbf{u}^{(1)} \in \mathbb{R}^2$) tak, aby na ni byla data projikována s nejmenší chybou projekce.

nD → kD: Nalézt **k bázových vektorů** $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \dots, \mathbf{u}^{(k)} \in \mathbb{R}^n$ takových, že projekce dat do jimi generovaného podprostoru proběhne se nejmenší chybou.



Principal Component Analysis (PCA)

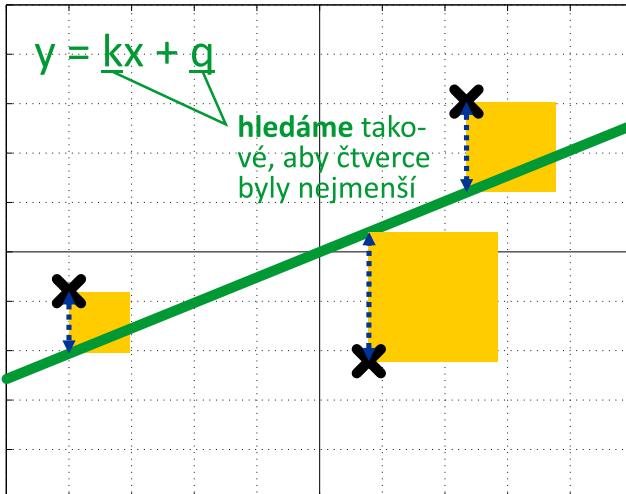
Formulace problému





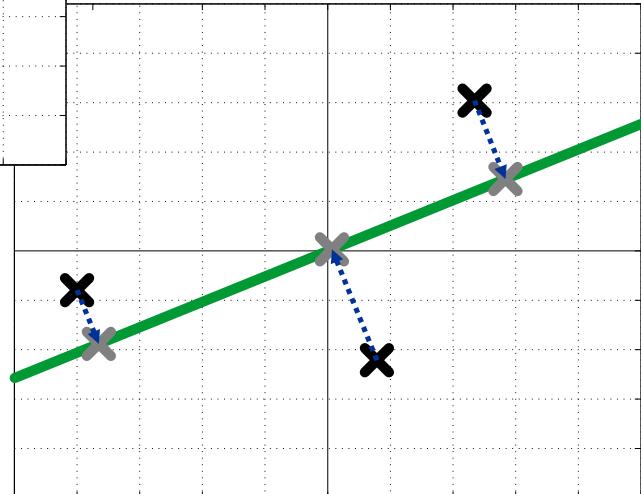
Principal Component Analysis (PCA)

není lineární regrese



Lineární regrese (metodou nejmenších čtverců) je postup k **nalezení lineární funkce**, která s nejmenší chybou approximuje data...

PCA je postup **nalezení báze k-rozměrného podprostoru** takového, že projekce dat do něj z pův. n-rozměrného proběhne s nejmenší chybou...





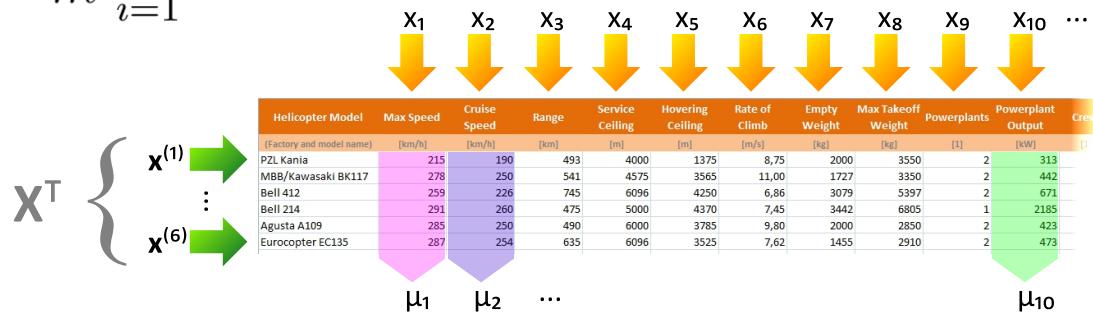
Algoritmus PCA

Příprava dat – tzv. preprocessing

Vstup: Množina m bodů (naměřených hodnot) v n-rozměrném prostoru $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ – tzv. **datová matici** X^T

(1) Normalizace střední hodnoty (Mean Normalization):

$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)} \quad \text{← vypočítat vektor středních hodnot}$$



V datové matici pak nahradit každé $x_j^{(i)}$ rozdílem $x_j^{(i)} - \mu_j$.

Tento krok je **nezbytný** – PCA je citlivá na střední hodnotu dat.



Algoritmus PCA

Příprava dat – tzv. preprocessing

(2) Škálování/normalizace vstupu (*Scaling*):

Pokud mají složky vstupních vektorů výrazně jiné rozsahy hodnot, např. x_9 (počet motorů) zhruba $\langle 1, 8 \rangle$ a x_{10} (výkon motoru) zhruba $\langle 300, 3000 \rangle$, pak je vhodné hodnoty vynásobit nějakým koeficientem, načež budou mít souměřitelné rozsahy.

$$x_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{q_j}$$

např. maximum dané složky
přes všechny vstupní vektory



Algoritmus PCA

Preprocessing (kód v MATLABu)

```
% škálování  
mx = max(X(:, :));  
  
for i = 1:m  
    for j = 1:n  
        if mx(j) ~= 0.0  
            X(i, j) = X(i, j) / mx(j);  
        end  
    end  
end  
  
% normalizace střední hodnoty  
mu = mean(X(:, :));  
  
for i = 1:m  
    for j = 1:n  
        X(i, j) = X(i, j) - mu(j);  
    end  
end
```



pořadí škálování a
normalizace záleží
na charakteru dat

vybereme to, které udrží data
dále od meze strojové přesnosti



Algoritmus PCA

Projekce z n- do k-dimenzionálního prostoru

(3) Výpočet kovarianční matici (*Covariance Matrix*):

$$\Sigma = \frac{1}{m} \mathbf{X}^T \mathbf{X} = \frac{1}{m} \sum_{i=1}^n \mathbf{x}^{(i)T} \mathbf{x}^{(i)}$$

$n \times n$
 $n \times 1$ $1 \times n$

(4) Výpočet vlastních vektorů (*Eigenvectors*) via SVD
(Singular Value Decomposition) matici Σ :

$$\Sigma = \mathbf{U} \mathbf{S} \mathbf{V}^T$$

$n \times n$
SVD

$$\mathbf{U} = \begin{bmatrix} \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{u}^{(1)} & \mathbf{u}^{(2)} & \mathbf{u}^{(3)} & \cdots & \mathbf{u}^{(n)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \end{bmatrix}$$

k
 \mathbf{u}_r



Algoritmus PCA

Projekce z n- do k-dimenzionálního prostoru

(5) Projekce dat:

Z kroku (4) máme $\mathbf{U}_r = \begin{bmatrix} \vdots & \vdots & \dots & \vdots \\ \mathbf{u}^{(1)} & \mathbf{u}^{(2)} & \dots & \mathbf{u}^{(k)} \\ \vdots & \vdots & \dots & \vdots \end{bmatrix} \in \mathbb{R}^{n \times k}$

redukovaná

Výpočet projikovaných dat $\mathbf{x} \in \mathbb{R}^n \rightarrow \mathbf{z} \in \mathbb{R}^k$ se provede vynásobením předzpracovaného (normalizovaného a naškálovaného) pův. vektoru (měření) redukovanou **maticí zobrazení (Projection Matrix)**:

$$\mathbf{z}^{(i)} = \mathbf{U}_r^T \mathbf{x}^{(i)}$$

vektor „skóre“ pův. vektoru (měření) vzhledem k vybraným k hlavním komponentám



Algoritmus PCA

Shrnutí – kód v MATLABu

```
% PCA  
Sigma = (1 / m) * X' * X;  
[U, S, V] = svd(Sigma);  
Ur = U(:, 1:k);  
  
for i = 1:m  
    Z(i, :) = Ur' * X(i, :)';  
end
```

datová matice (s normalizovanými a naškálovanými hodnotami, tj. po průchodu předzpracováním)

matice k-rozměrných redukovaných dat, tj. ortogonální průmět původních dat do kD podprostoru

- při implementaci v nízkoúrovňových jazycích je třeba buď naprogramovat **SVD** nebo využít nějakou **knihovnu algebraických operací**, např. ALGLIB (<http://www.alglib.net/>), LAPACK (<http://www.netlib.org/lapack/>), BLAS, PLASMA, Intel Math Kernel Library, AMD Core Math Library, ...





Potvrzení linearity PCA

Rekonstrukce původních dat z redukovaných

Jelikož projekce z n- do k-dimenzionálního (pod)prostoru se počítá takto:

$$\mathbf{z}^{(i)} = \mathbf{U}_r^T \mathbf{x}^{(i)}$$

Pak zřejmě (důkaz příp. z domácí úlohy):

$$\hat{\mathbf{x}}^{(i)} = \mathbf{U}_r \mathbf{z}^{(i)T}$$

K vektoru odhadu pův. dat je ovšem třeba přičíst odstraněnou střední hodnotu a přeškálovat zpět na pův. rozsah:

$$\hat{\mathbf{x}}^{(i)} = \mathbf{q} \cdot (\mathbf{U}_r \mathbf{z}^{(i)T} + \boldsymbol{\mu}^T)$$

Výsledkem je reprojekce redukovaných dat do původního n-rozměrného prostoru zatížená chybou projekce.

Podívejme se, jak dopadly naše technické údaje vrtulníků...



Potvrzení linearity PCA

Rekonstrukce původních dat z redukovaných

Původní data

| Max Speed | Cruise Speed | Range | Service Ceiling | Hovering Ceiling | Rate of Climb | Empty Weight | Max Takeoff Weight | Powerplants | Powerplant Output | Crew |
|-----------|--------------|-------|-----------------|------------------|---------------|--------------|--------------------|-------------|-------------------|------|
| [km/h] | [km/h] | [km] | [m] | [m] | [m/s] | [kg] | [kg] | [1] | [kW] | [1] |
| 215 | 190 | 493 | 4000 | 1375 | 8,75 | 2000 | 3550 | 2 | 313 | |
| 278 | 250 | 541 | 4575 | 3565 | 11,00 | 1727 | 3350 | 2 | 442 | |
| 259 | 226 | 745 | 6096 | 4250 | 6,86 | 3079 | 5397 | 2 | 671 | |
| 291 | 260 | 475 | 5000 | 4370 | 7,45 | 3442 | 6805 | 1 | 2185 | |
| 285 | 250 | 490 | 6000 | 3785 | 9,80 | 2000 | 2850 | 2 | 423 | |
| 287 | 254 | 635 | 6096 | 3525 | 7,62 | 1455 | 2910 | 2 | 473 | |

Rekonstrukce

| Xa <6x10 double> | | | | | | | | | | |
|------------------|--------|--------|--------|---------|---------|------|---------|---------|------|---------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 226.57 | 201.88 | 450.06 | 3812.64 | 1536.94 | 9.64 | 1816.89 | 3309.39 | 1.94 | 380.66 |
| 2 | 263.98 | 233.43 | 569.98 | 5289.29 | 3169.16 | 8.99 | 1927.28 | 3376.98 | 2.03 | 398.73 |
| 3 | 290.56 | 256.60 | 621.75 | 6053.81 | 4446.87 | 8.07 | 2492.30 | 4504.87 | 1.79 | 910.54 |
| 4 | 281.20 | 250.51 | 511.34 | 4999.76 | 4322.79 | 7.15 | 3635.54 | 7086.39 | 1.07 | 2106.67 |
| 5 | 274.78 | 242.52 | 605.47 | 5722.99 | 3637.19 | 8.81 | 1946.35 | 3368.11 | 2.06 | 390.83 |
| 6 | 277.92 | 245.06 | 620.41 | 5888.51 | 3757.05 | 8.82 | 1884.64 | 3216.26 | 2.11 | 319.56 |

```
Xa = Ur * Z';
Xa = Xa';

for i = 1:m
    Xa(i, :) = Xa(i, :) + mu;
    Xa(i, :) = Xa(i, :) .* mx;
end
```

Chyba projekce se zdá být velká, ale je třeba si uvědomit, že jsme odstranili 80% informace...



Vlastnosti PCA

Matematické detaily

- ortogonální lineární transformace, která transformuje body z n-dimenzionálního prostoru do prostoru k-dimenzionálního tak, že lineární kombinace $\mathbf{X}\mathbf{u}^{(1)}$ má největší rozptyl ze všech lineárních kombinací bodů, $\mathbf{X}\mathbf{u}^{(2)}$ má největší rozptyl z všech těch lineárních kombinací, které vyhovují podmínce kolmosti $\mathbf{u}^{(1)}$ s $\mathbf{u}^{(2)}$, atd.
- generující vektory podprostoru, tj. **hlavní komponenty** (*Principal Components*) jsou seřazeny se snižujícím se rozptylem

$$\mathbf{U} = \begin{bmatrix} \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{u}^{(1)} & \mathbf{u}^{(2)} & \mathbf{u}^{(3)} & \cdots & \mathbf{u}^{(n)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \end{bmatrix}$$

- matematika v pozadí PCA je poměrně komplikovaná



SVD – Matematický základ PCA

Stručný popis

SVD (*Singular Value Decomposition*) je faktorizace reálné či komplexní matice **M** velikosti $m \times n$:

$$\mathbf{M} = \mathbf{U}\mathbf{S}\mathbf{V}^*$$

kde **U** je reálná či komplexní unitární matice (tj. $\mathbf{U}^*\mathbf{U} = \mathbf{U}\mathbf{U}^* = \mathbf{I}$) o velikosti $m \times m$, **S** je obdélníková diagonální matice s nezápornými reálnými čísly na diagonále a **V*** (konjugovaná čili Hermitovsky transponovaná) reálná či komplexní unitární matice o velikosti $n \times n$.

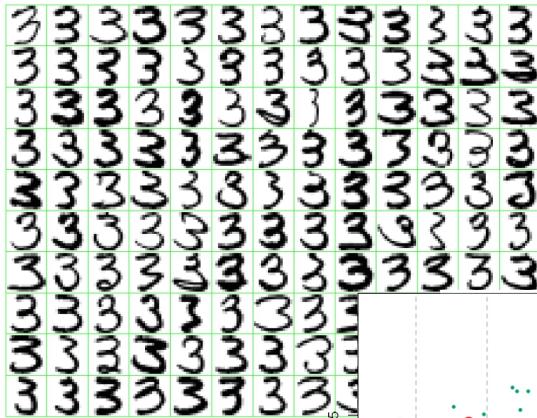
- prvky na diagonále **S** se nazývají singulárni hodnoty **M**
- matice **U** obsahuje m levých singulárnych vektorů **M**
- matice **V** obsahuje n pravých singulárnych vektorů **M**
- detailly, důkazy, odvození, atp. v literatuře



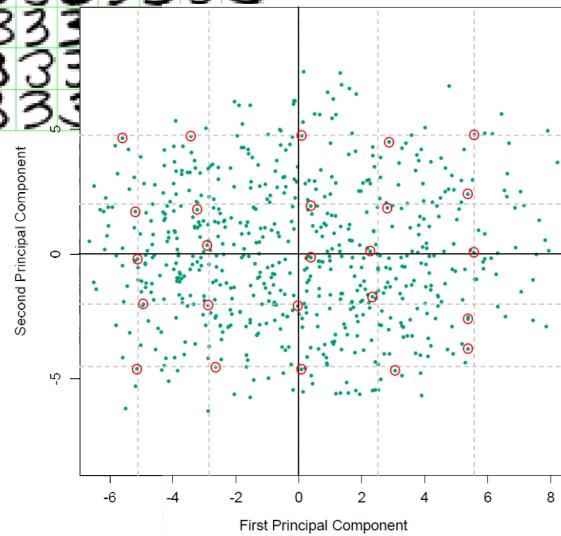


Použití PCA

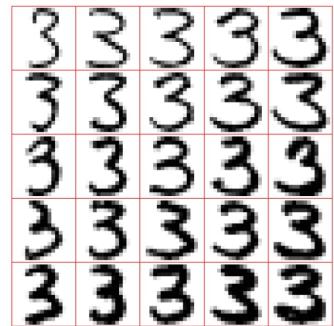
Praktická ukázka



První dvě hlavní komponenty ručně psaných číslic 3



Vstupní data: 130 ručně psaných číslic 3 (vykazují mnohotvárnost jednotlivých rukopisů)





Použití PCA

Praktická ukázka



$64 \times 64 \text{ px}$
grayscale
8 bit/px
 $\rightarrow 4096 \text{ B}$

PCA

projekce z 4096D do 16D

$\rightarrow 16 \text{ B}$
(výsledek
po repro-
jekci zpět
na 64×64)



(obrázky z korpusu LICS Face Recognition Training Data © 2011 Kamil Ekštein)





Použití PCA

Praktická ukázka



} 64 x 64 px
 grayscale
 8 bit/px
 → 4096 B

projekce z 4096D do 2D:
 každý obrázek je repre-
 zentován jen 2 čísly, za-
 chováno je **46,21%** roz-
 ptylu pův. dat

PCA

→ 2 B
 po repro-
 jekci zpět
 na 64 x 64



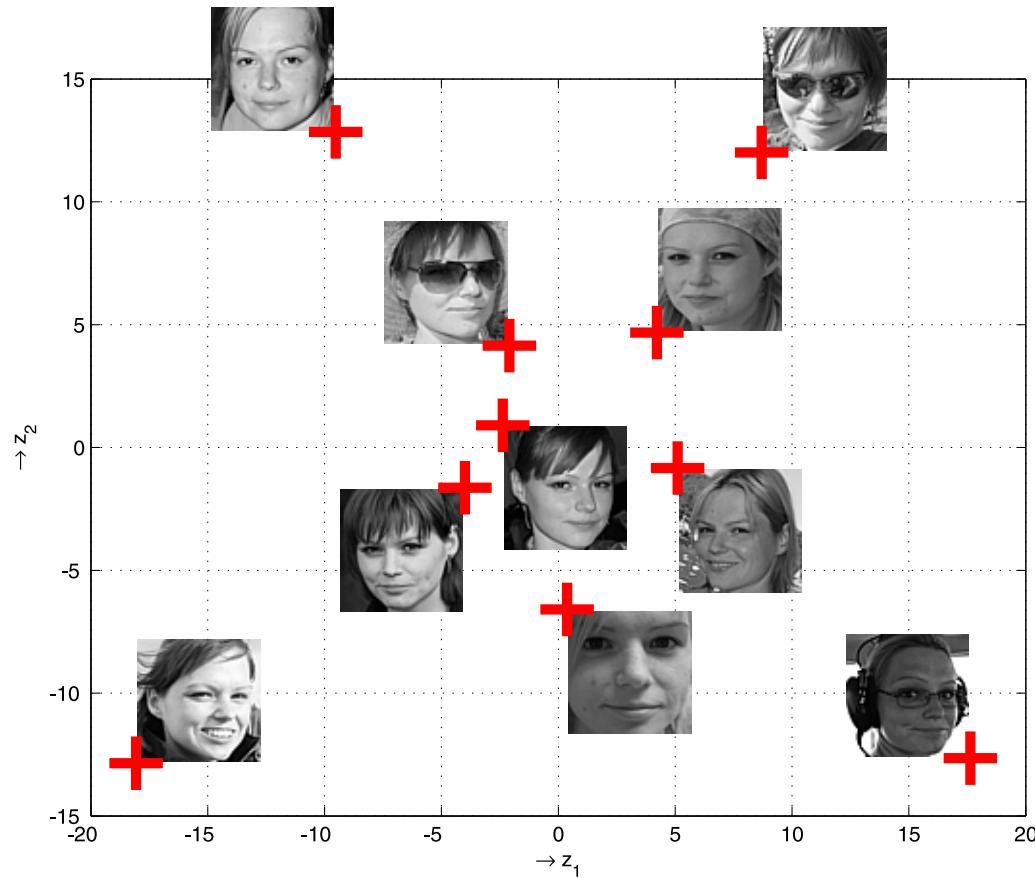
(obrázky z korpusu LICS Face Recognition Training Data © 2011 Kamil Ekštein)

| Z <10x2 double> | |
|-----------------|-------------------|
| | |
| 1 | 2 |
| 1 | 0.3793 -6.5961 |
| 2 | -3.9951 -1.6377 |
| 3 | -9.5229 12.8427 |
| 4 | 5.1074 -0.8382 |
| 5 | 8.7043 12.0087 |
| 6 | -2.3735 0.9017 |
| 7 | -18.0607 -12.8544 |
| 8 | -2.0927 4.1444 |
| 9 | 4.2190 4.6781 |
| 10 | 17.6350 -12.6493 |



Použití PCA

Praktická ukázka





Výběr počtu hlavních komponent

Jak vybrat dimenzi podprostoru?

Vyjdeme ze dvou jednoduchých charakteristik, které jsou k dispozici po výpočtu SVD:

Střední kvadr. chyba projekce: $e_p = \frac{1}{m} \sum_{i=1}^m |\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i)}|^2$

Celkový rozptyl vstupních dat: $\varsigma_d = \frac{1}{m} \sum_{i=1}^m |\mathbf{x}^{(i)}|^2$

Pak **dimenzi podprostoru k vybíráme** tak, aby byla nejmenší možná, přičemž

$$\frac{e_p}{\varsigma_d} \leq 0,01 \quad (= 1\%)$$

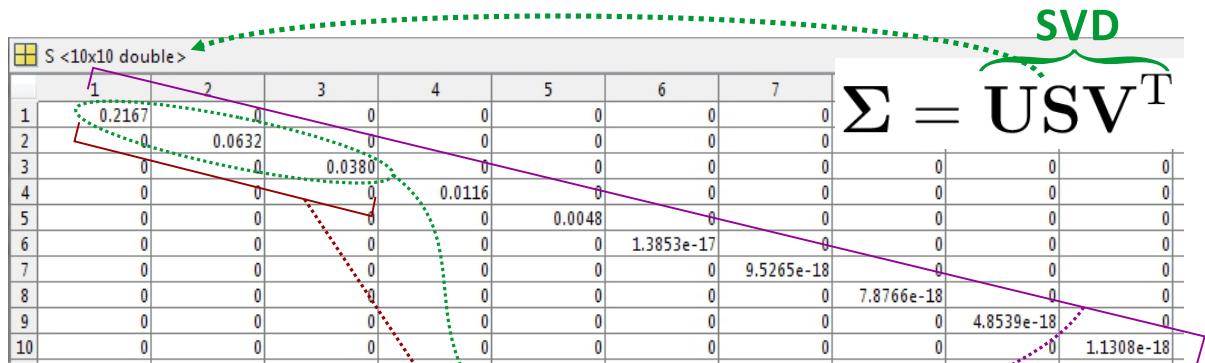
→ 99% rozptylu dat je při projekci zachováno...

Hodnotu zachovávaného rozptylu pův. dat si volíme.



Výběr počtu hlavních komponent

Jak vybrat dimenzi podprostoru?



$$1 - \frac{\sum_{i=1}^k s_{ii}}{\sum_{i=1}^n s_{ii}} \leq 0,01$$

Volíme nejmenší k takové, že platí

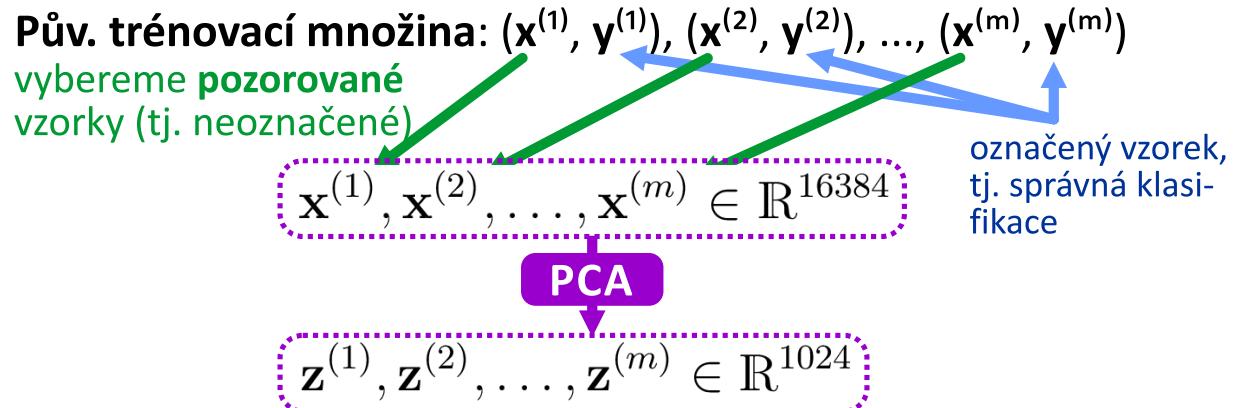
Procento zachovávaného rozptylu původních dat – volí se...

$$\frac{\sum_{i=1}^k s_{ii}}{\sum_{i=1}^n s_{ii}} \geq 0,99$$



Aplikace PCA

Urychljení učení s učitelem



Nová trénovací množina: $(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), \dots, (z^{(m)}, y^{(m)})$

Pozn.: Projekce $x^{(i)} \rightarrow z^{(i)}$ by měla být stanovena aplikací PCA
pouze na vzorky z trénovací množiny. Tuto projekci lze
také použít na vzorky z validační a testovací množiny.



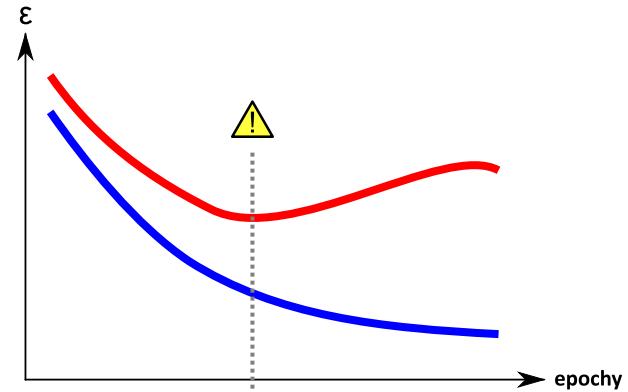
Nevhodná aplikace PCA

Zabránění přeúčení algoritmu učení s učitelem

Dochází-li k přeúčení (*Overtraining/Overfitting*), může se zdát jako dobrý nápad snížit dimenzionalitu vstupních dat via PCA: Použít k učení $z^{(i)}$ místo $x^{(i)}$, tj. k příznaků místo n , $k < n$.

Není to dobrý nápad. Může to sice fungovat, ale PCA obecně není nástroj k potlačení overfittingu. Místo toho lze použít:

- regularizaci (*Regularization*)
- prořezávání (*Pruning*)
- včasné zastavení (*Early Stopping*)
- křížová validace (*Cross-validation*)



(obrázek Overfitting z Wikimedia Commons)



Nevhodná aplikace PCA

Nasazení PCA tam, kde to není nutné...

Návrh učícího se systému:

- pořízení trénovací množiny $(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \dots, (\mathbf{x}^{(m)}, \mathbf{y}^{(m)})$
- aplikace PCA k redukci dimenzionality vzorků $\mathbf{x}^{(i)}$
- natrénování např. logistické regrese na redukované množině $(\mathbf{z}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{z}^{(2)}, \mathbf{y}^{(2)}), \dots, (\mathbf{z}^{(m)}, \mathbf{y}^{(m)})$
- testování na testovací množině (také redukované):
 - projekce via PCA $\mathbf{x}_{\text{test}}^{(i)} \rightarrow \mathbf{z}_{\text{test}}^{(i)}$
 - spuštění $h_{\theta}(\mathbf{z})$ na množině $(\mathbf{z}_{\text{test}}^{(1)}, \mathbf{y}_{\text{test}}^{(1)}), \dots, (\mathbf{z}_{\text{test}}^{(m)}, \mathbf{y}_{\text{test}}^{(m)})$

Před nasazením PCA je třeba **zkusit, zda by to nefungovalo** dobře s původními daty $\mathbf{x}^{(i)}$. Teprve pokud se ukáže, že ne, pak lze uvažovat o redukci dimenzionality dat pomocí PCA.





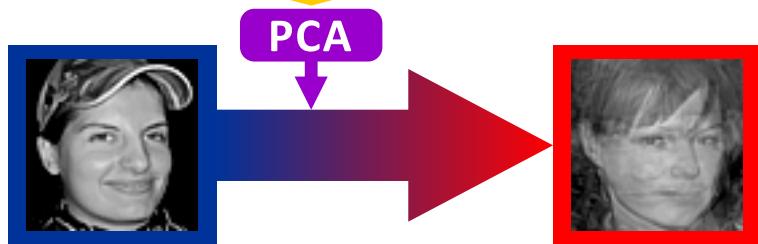
Nevhodná? aplikace PCA

Praktická ukázka



množina vzorků použitá pro výpočet
matice projekce

neznámý
vzorek –
nezúčast-
nil se vý-
počtu PCA



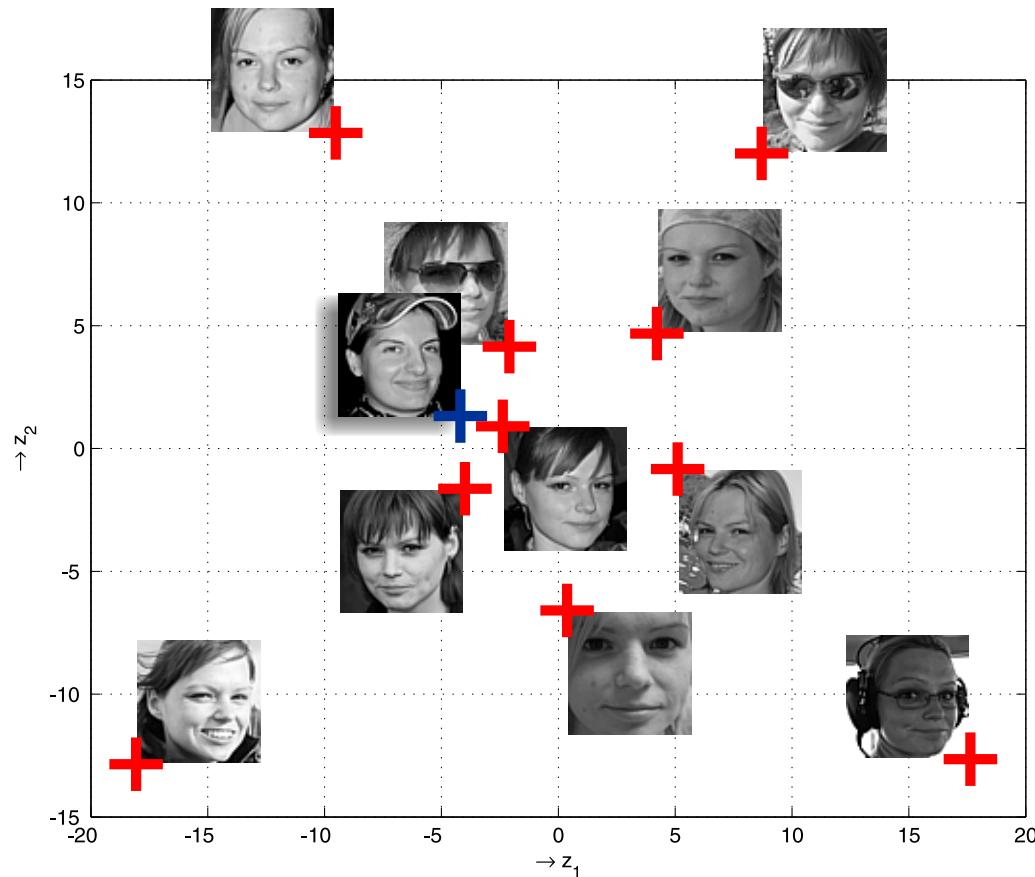
Použijeme získanou matici projekce pro **projekci neznámého vzorku** (tj. situace, kdy chceme snížit dimenzionalitu celé úlohy) → **problém:** Projekce neznámého vzorku do prostoru generovaného bází, která ovšem není optimalizována pro projekci tohoto vzorku...





Nehodná? aplikace PCA

Praktická ukázka





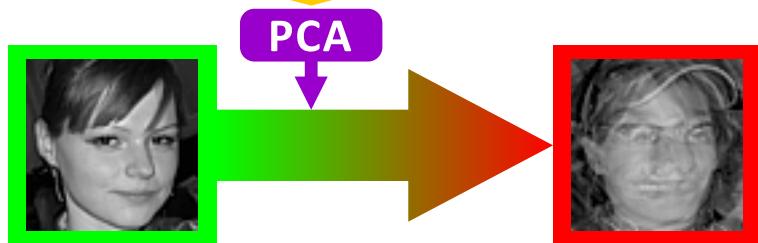
Nevhodná? aplikace PCA

Praktická ukázka – vyhodnocení podobnosti



množina vzorků použitá pro výpočet
matice projekce

klasifikova-
ný vzorek –
komu z tré-
novací mno-
žiny je oso-
ba podobná?



Projekce do podprostoru umožňuje vyhodnocení vzdálenosti vzorku od trénovacích vzorků, tj. zjistit, kterému snímku je klasifikovaný snímek podobný. **Lze tento postup užít např. pro autentikaci uživatelů?**



Nehodná? aplikace PCA

Praktická ukázka – výhodnocení podobnosti

