Neurovõrgud. Praktikum 12.

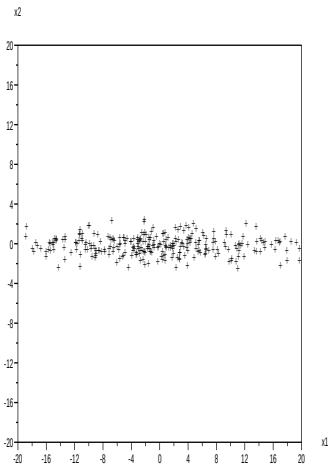
6. mai 2005. a.

1 Peakomponentide analüüs (PCA)

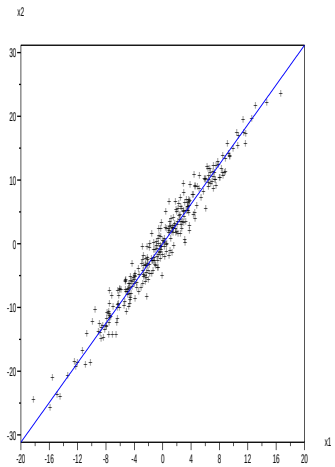
Andmed, mille peal praktikas rakendatakse masinõppimise algoritme on sageli väga kõrgemõõtmelised. On tavaline, et andmetes on sadu või tuhandeid tunnusi. Nii suur tunnuste arv võib tihti olla probleemiks. Esiteks tavaliselt neid on ,,liiga palju" ses mõttes et sama informatsioon võib olla sama hästi ja isegi mürakindlam esitatud ka väiksema arvu tunnustega. Teiseks lühemate vektoritega töötamine on lihtsalt kiirem. Selle pärast on masinõppimises olulisel kohal *mõõtmelisuse vähendamise probleem (dimensionality reduction)*. On olemas palju erinevaid meetodeid tunnuste arvu vähendamiseks: PCA *(peakomponentide analüüs)* ja SVD *(singular value decomposition)*, MDS *(multidimensional scaling)*, klasterdamine, otsustuspuud ja vastastikune informatsioon. Nendest üks populaarsemaid on *peakomponentide analüüs (principal component analysis, PCA)* millega me siin tegelemegi.

1.1 Mitmemõõtmeline normaaljaotus (jälle)

Vaatleme punkte kahemõõtmelisest normaaljaotusest.



Joonis 1



Joonis 2

Need punktid olgu andmed, mille mõõtmelisust me soovime vähendada. Iga punkt $\mathbf{x}$ on esitatav kahe koordinaatiga (x_1, x_2) kuid me tahaksime esitada teda kasutades ainult ühte koordinaati: (x'). Esimene idee mis tuleb pähe on lihtsalt ära kaotada andmetest ühte koordinaati, pannes näiteks $x' := x_2$. Selline idee on selgelt mitteoptimaalne. Miks peaksime me valima just teist koordinaati? Kui punktid on jaotatud nii nagu näidatud joonisel 1 tundub kasulikum valida esimest koordinaati (miks?). Enamgi, kui jaotus on nagu joonisel 2, ükski koordinaat ei anna parimat tulemust. Hoopis parem oleks projitseerida punkte joonisel näidatud sirgele, kuna just siis on projektsioonide dispersioon maksimaalne. Selline mõttekäik viib meid PCA põhimõttele: üritame leida sellist sirget (*andmete esimest peakomponenti*), millele andmeid projitseerides on projektsioonidel maksimaalne dispersioon. Paneme tähele, et informatsioon mida me kaotame kui projitseerime punkte leitud sirgele on sisuliselt esitatav nende punktide projektsioonidega *sirgega ortogonaalsele alamruumile*. Näiteks vaadeldud näite puhul on peakomponendiga ortogonaalne alamruum sellega perpendikulaarne sirge, kuid kui andmed oleksid kolmemõõtmelised oleks see juba tasand. Nende ,,ülejäänute" andmete peal (mille mõõtmelisus on nüüd ühe võrra väiksem kui algsete andmete oma) võiks protseduuri korrata ning leida niimoodi *teist peakomponenti*, pärast kolmandat, jne.
Kui tähistada joonisel 2 näidatud andmete projektsiooni esimesele peakomponendile x_1'-ga, ning projektsiooni temaga perpendikulaarsele sirgele (teisele peakomponendile) x_2'-ga, siis x_1' vs x_2' graafikuks on joonis 1. Niiet peakomponentide leidmine sisuliselt vastab andmehulga pööramisele nii, et koordinaatid muutuksid korreleerimatuteks. Kui andmed on juba korreleerimatud (joonis 1), siis esimene peakomponent on see koordinaat, mille dispersioon on suurim, viimane peakomponent — see, mille dispersioon on väiksem. Mõõtmelisuse vähendamiseks piisab valida sobiva hulga peakomponente. Näiteks joonisel 1 on esimese koordinaati dispersioon 99, ning teise — 1. Seega kui me jätame ainult esimese peakomponendi, jääb alles ,,99% algsest dispersioonist", mis on ,,suhteliselt palju".

Ülesanne 1 (1p): Formaalselt ei pruugi informatsioon alati kaotsi minema kui kahemõõtmelisi punkte ühe arvuga esitada. On teada, näiteks, et leidub bijektsioon ühikruudu $[0,1]^2$ ja lõiku $[0,1]$ vahel. Esita seda bijektsiooni. Miks ei saa seda kasutada mõõtmelisuse vähendamiseks?

Teatavasti on mitmemõõtmeline normaaljaotus määratud keskväärtusega $\boldsymbol{\mu}$ ja kovariatsioonimaatriksiga $\boldsymbol{\Sigma}$ (vt. **8.** praktikumi materjale). Edaspidises oletame et andmed on tsentreeritud, s.t. $\boldsymbol{\mu} = \mathbf{0}$. Uurime kovariatsioonimaatriksit. Kui koordinaatide omavahelised korrelatsioonid on nullid (joonis 1), siis on kovariatsioonimaatriks diagonaalne, kusjuures $(\boldsymbol{\Sigma})_{ii}$ näitab i-nda koordinaati dispersiooni. Näiteks joonisel 1 näidatud jaotusele vastab

kovariatsioonimaatriks $\begin{pmatrix} 99 & 0 \\ 0 & 1 \end{pmatrix}$. Seega selleks et leida sel juhul esimest peakomponenti lihtsalt leiame kus asub kovariatsioonimaatriksi diagonaalis suurim element.
Olgu nüüd kovariatsioonimaatriks mitte diagonaalne (jaotus joonisel 2). Ta ikkagi peab olema sümmeetriline ja positiivselt määratud, ning teda saab esitada kujul $\mathbf{\Sigma} = \mathbf{V}\mathbf{D}\mathbf{V}^T$, kus $\mathbf{D}$ on positiivsete elementidega diagonaalmaatriks ning $\mathbf{V}$ on ortogonaalne maatriks (ehk ,,pööre maatriks"). Sisuliselt tähendab see seda, et kui me pöörame andmeid lineaarteisendusega $\mathbf{V}^{-1}$, muutub nende kovariatsioonimaatriks diagonaalseks. Seega maatriksi $\mathbf{V}$ veergudes ongi peakomponentide vektorid.

Ülesanne 2 (1p): Näita, et kui me kujutame iga andmepunkti $\mathbf{x}$ kujutusega $\mathbf{V}^{-1}$, siis niimoodi kujutatud andmete kovariatsioonimaatriks on $\mathbf{D}$. Miks saab sellest järeldada et $\mathbf{V}$ veergudes on peakomponentide vektorid?

1.2 Standartne PCA tuletus

Vaadeldud näide oli seotud normaaljaotusega, ning oli eelkõige toodud näitlikustamiseks. Olgu nüüd andmed suvalisest jaotusest, oletame ainult et nad on tsentreeritud. Otsime esimest peakomponenti kui ühikvektorit $\mathbf{w}$, millele andmeid projitseerides on dispersioon maksimaalne. Kui $\mathbf{X}$ on andmete maatriks, siis $\mathbf{X}\mathbf{w}$ on punktide projektsioonide maatriks, ning $\frac{1}{n}(\mathbf{X}\mathbf{w})^T(\mathbf{X}\mathbf{w}) = \frac{1}{n}\mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w}$ on projektsioonide dispersioon. Paneme tähele, et $\frac{1}{n}\mathbf{X}^T\mathbf{X}$ on andmete kovariatsioonimaatriks, tähistame seda $\tilde{\mathbf{\Sigma}}$-ga. Ülesandeks on seega leida ühikvektorit $\mathbf{w}$, mis maksimiseerib avaldist $\mathbf{w}^T\tilde{\mathbf{\Sigma}}\mathbf{w}$.

Ülesanne 3 (1p): Näita, et otsitav $\mathbf{w}$ on maatriksi $\tilde{\mathbf{\Sigma}}$ suurimale omaväärtusele vastav omavektor[1].

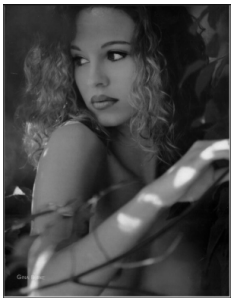
Saab näidata, et kõik peakomponendid on maatriksi $\tilde{\mathbf{\Sigma}}$ omavektorid, kusjuures esimene peakomponent vastab suurimale omaväärtusele, ning viimane — väiksemale. Lisaks osutub, et andmete tsentreerimist võib ära jätta, ning võtta peakomponentideks maatriksi $\mathbf{X}^T\mathbf{X}$ omaväärtusi. See maatriks pole küll enam andmete kovariatsioonimaatriks, kuid edaspidises mõistame kovariatsioonimaatriksi all just teda.
Kokkuvõttes on seega PCA algoritm järgmine:

1. Leia maatriksi $\tilde{\mathbf{\Sigma}} = \mathbf{X}^T\mathbf{X}$ omavektorid. Vali nendest k suurimale omaväärtusele vastavad omavektorid, olgu need $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$.

2. Projitseeri (tsentreeritud) andmed leitud omavektoritele:
   $\mathbf{x}_i \to (\mathbf{v}_1^T\mathbf{x}_i, \mathbf{v}_2^T\mathbf{x}_i, \ldots \mathbf{v}_k^T\mathbf{x}_i)$.

[1] Vektor $\mathbf{v}$ on maatriksi $\mathbf{A}$ *omavektor*, kui leidub $\lambda \in \mathbb{R}$ selline et $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. Arvu λ nimetatakse siis maatriksi $\mathbf{A}$ (omavektorile $\mathbf{v}$ vastavaks) *omaväärtuseks*.

2 Piltlik näide



Mõõtmelisuse vähendamist saab loomulikult kasutada andmete kompressimiseks (kadudega). Vaatleme siin pildi pakkimist PCA abil.
Failis `image.data` on salvestatud 250×323 pilt. Pildi saab seega Scilabi sisse laadida käsuga

```
img = read("image.data", -1, 250);
```

ning vaadata järgmise funktsiooni abil:

```
function show(img)
    xbasc();
    cm = [0:1/255:1]';
    xset("colormap", [cm cm cm]);
    xset("wdim", 250, 323);
    Matplot(256*img, strf="020");
endfunction
```

Vaatleme pildi kui andmepunktide hulga (iga pildi rida olgu üks 250-mõõtmeline punkt). Pildi pakkimiseks rakendame punktidel PCA-d ning jätame ainult niipalju peakomponente kui palju on vaja selleks et pildi kvaliteet väga halvaks ei läheks. Kokkuvõtteks on pakkimisalgoritm järgmine:

1. Leiame punktidest k peakomponenti $(\mathbf{v}_1, \ldots, \mathbf{v}_k)$.
2. Projitseerime iga punkti $\mathbf{x}_i$ peakomponentidele. Tähistame saadud projektsioonide vektorit $\mathbf{x}'_i$.
3. Salvestame vektorid $\mathbf{v}_i$, $\mathbf{x}'_i$.

Salvestatud andmetest saab pildi lihtsalt taastada: meil on olemas kasutatud peakomponentide vektorid $\mathbf{v}_i$, ning iga pildi rea i kohta me teame teda lahutust peakomponentideks $\mathbf{x}'_i$: kui x'_{ij} tähistab vektori $\mathbf{x}'_i$ j-nda komponendi, siis taastatud pildi i-s rida $\mathbf{x}''_i$ avaldub peakomponentide kaudu kui $\mathbf{x}''_i = \sum_{j=1}^{k} x'_{ij}\mathbf{v}_j$.
Olgu $\mathbf{V}$ maatriks, mille veergudes on peakomponendid $\mathbf{v}_i$. Siis pakkimine avaldub kui $\mathbf{X}' = \mathbf{XV}$, ning taastamine kui $\mathbf{X}'' = \mathbf{X}'\mathbf{V}^T$ (veendu selles!).

Ülesanne 4 (2p): Realiseeri pildi pakkimist PCA abil. Olgu $k = 20$, s.t. jäta ainult 20 peakomponendi 250-st. Võrdle pakitud kujul salvestatud pildi kvaliteedi originaalse pildi kvaliteediga. Selleks et salvestada originaalse pildi on vaja $250 \times 323 = 80750$ reaalarvu. Kui palju reaalarvu on vaja pakitud kujul pildi hoidmiseks? Esita kood.
Vihjed:

- Maatriksi `C` omavektoreid ja omaväärtusi saab kätte käsuga `spec`:

```
[V, D] = spec(C);
```

Tulemusena tagastatakse eelnevalt mainitud maatriksid **V** ja **D**. Kui `C` on punktide kovariatsioonimaatriks `X'*X`, siis maatriksi **V** veergudes on kõik peakomponendid, ning maatriksi **D** diagonaalelemendid — vastavad dispersioonid (omaväärtused).

- Järgmine kood valib siis k suurimat peakomponenti:

```
[d, eigenv_sorted_idx] = sort(diag(D));
V_base = V(:, eigenv_sorted_idx(1:k));
```

Ülesanne 5 (1p): Selleks et otsustada kui palju peakomponente tuleb tegelikult jätta on kasuks vaadata *peakomponentide spektrit*. Selleks sorteerime peakomponentide dispersioonid ning vaatleme neid graafikul pulkadena. Koosta selline graafik pakitava pildi peakomponentide jaoks. Graafikust on näha et 40-st peakomponendist peaks piisama sellest et salvestada pildi suhteliselt hästi. Kas tõesti 40-ga on kvaliteet piisavalt hea? Kui hea on sel juhul pakkimise määr (s.t. kui vähem ruumi võtavad pakitud andmed võrreldes originaaliga)? Esita kood, pilt ja vastused küsimustele.
Vihje:

```
[d, eigenv_sorted_idx] = sort(diag(D));
plot2d3([1:length(d)], d);
```

3 Andmete mürast taastamine

PCA-d saab tihti kasutada andmetes müra vähendamiseks. Vaatleme järgmist näidet: olgu meil 400 signaali, kusjuures pool nendest on mürased sinusoidid, ning pool — mürased eksponendid. Genereerime neid ja vaatame graafikul järgmiselt:

```
rand("normal");
t = [0:0.1:3]; err = 0.5;
// Sinusoidid
S1 = [];
for i = 1:200, S1 = [S1; sin(t) + err*rand(t)];,end
// Eksponendid
S2 = [];
for i = 1:200, S2 = [S2; exp(-t) + err*rand(t)];,end
// Andmete maatriks
X = [S1; S2];
plot2d(t, X');
```

Graafikust on näha et müra on suhteliselt tugev ning originaalsinusoidi ja eksponenti eristada on võimatu. Osutub aga et PCA abil saab neid taastada.

Ülesanne 6 (1p): Leia andmete peakomponente. Uuri peakomponentide spektrit. Pane graafikule kolm suurimat peakomponenti. Esita kood ja pildid.

Peakomponentide spektrist on näha et teistest olulisemad on kaks esimest peakomponenti. Kui vaatleme esimest kolme peakomponenti graafikul, siis näeme, et tõepoolest, kaks esimest on teatud ,,mõistlikud" jooned, kuid kolmas on juba puhas müra. Esimesed peakomponendid pole küll võrdsed algsete signaalidega (s.t. nad pole sinusoid ja eksponent), kuid me teame et algseid signaale saab esitada nende lineaarkombinatsioonina.

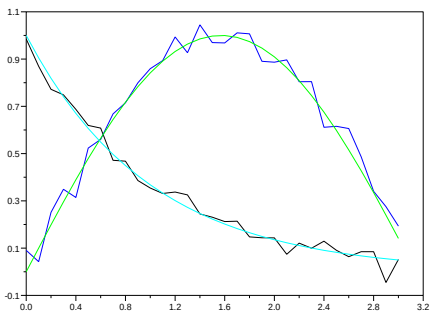
Ülesanne 7 (1p): Projitseeri andmeid esimese kahe peakomponenti peale ning pane punktidena graafikule. On näha et punktid modustavad kaks klastrit. Nende klastrite keskpunktid ongi algsed signaalid. Leida neid keskpunkte ning taasta nende abil algseid signaale. Pane saadud signaalid graafikule koos funktsioonidega $\sin(x)$ ja $\exp(-x)$.
Vihjed:

- Klastrite keskpunktide leidmiseks võib kasutada funktsiooni *k-means*, mis on antud failis `k_means.sci`. Funktsioon võtab parameetritena andmete maatriksi ja otsitavate klastrite arvu ning tagastab maatriksi, mille ridades on klastrite keskpunktid.

```
means = k_means(X_p, 2);
```

- Saadud keskpunktid annavad otsitavate signaalide lahutust peakomponentideks.

Ülesanne 8 (1p): Tsentreeri andmeid kohe pärast genereerimist ning tee läbi kogu eelneva protseduuri tsentreeritud andmete peal. Mis muutus? Tegelikult kogu ,,mürast taastamise" protseduuri võiksime siin sama edukalt läbi viia ilma PCA abita, kasutades ainult klasterdamist (proovi kohe kutsuda välja `k_means(X, 2)`). Milles siis seisnes PCA roll?



Joonis 3: Taastatud signaalid