Linear Algebra Methods for Data Mining

Saara Hyvönen, Saara.Hyvonen@cs.helsinki.fi

Spring 2007

PCA, NMF

Summary: PCA

- PCA is SVD done on **centered** data.
- PCA looks for such a direction that the data projected onto it has maximal variance.
- When found, PCA continues by seeking the next direction, which is orthogonal to all the previously found directions, and which explains as much of the remaining variance in the data as possible.
- Principal components are uncorrelated.

How to compute the PCA:

Data matrix $\mathbf{A}$, rows=data points, columns = variables (attributes, parameters).

1. Center the data by subtracting the mean of each column.

2. Compute the SVD of the centered matrix $\hat{\mathbf{A}}$ (or the k first singular values and vectors):
$$\hat{\mathbf{A}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

3. The principal components are the columns of $\mathbf{V}$, the coordinates of the data in the basis defined by the principal components are $\mathbf{U}\mathbf{\Sigma}$.

Singular values tell about variance

The variance in the direction of the k^{th} principal component is given by the corresponding singular value: σ_k^2.

Singular values can be used to estimate how many principal components to keep.

Rule of thumb: keep enough to explain 85% of the variation:

$$\frac{\sum_{j=1}^{k} \sigma_j^2}{\sum_{j=1}^{n} \sigma_j^2} \approx 0.85.$$

PCA is useful for

- data exploration
- visualizing data
- compressing data
- outlier detection

Example: customer data

2	1	1	1	1	0	0	1	1	1	1	1	0	0
2	3	2	2	2	0	0	2	2	2	2	2	0	0
4	4	3	4	4	0	0	4	4	4	4	4	0	0
3	3	3	4	3	0	0	3	3	3	3	3	0	0
1	1	1	1	1	0	0	1	1	1	1	2	0	0
2	2	2	2	2	0	0	2	2	2	2	1	0	0
4	4	4	4	4	0	0	4	4	4	4	3	0	0
1	1	1	1	1	0	0	1	1	1	1	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	1
0	0	0	0	0	2	2	0	0	0	0	2	2	2
0	0	0	0	0	3	3	0	0	0	0	3	3	3
0	0	0	0	0	4	4	0	0	0	0	4	4	4
0	0	0	0	0	2	2	0	0	0	0	2	2	3
0	0	0	0	0	4	3	0	0	0	0	3	3	3
0	0	3	3	3	3	3	0	0	3	3	3	3	3

Data matrix. Rows=customers, columns=days.

Same data, rows and columns permuted:

1	0	1	0	1	0	0	1	1	0	1	1	1	1
0	3	3	3	3	3	3	0	3	3	0	3	3	0
2	0	2	2	2	0	0	2	2	0	2	2	2	3
0	2	0	2	0	2	2	0	0	2	0	0	0	0
4	0	4	3	4	0	0	4	4	0	4	4	4	4
3	0	3	3	3	0	0	3	4	0	3	3	3	3
4	0	4	4	4	0	0	4	4	0	4	4	3	4
0	3	0	3	0	3	3	0	0	3	0	0	0	0
0	4	0	3	0	3	3	0	0	3	0	0	0	0
2	0	2	1	2	0	0	2	2	0	2	2	2	2
0	4	0	4	0	4	4	0	0	4	0	0	0	0
0	1	0	1	0	1	1	0	0	1	0	0	0	0
1	0	1	2	1	0	0	1	1	0	1	1	1	1
0	2	0	2	0	3	2	0	0	2	0	0	0	0
1	0	1	1	1	0	0	2	1	0	1	1	1	1

Define the data matrix on the previous slide to be $\mathbf{A}$.

The rows of $\mathbf{A}$ correspond to customers.

The columns of $\mathbf{A}$ correspond to days.

Let us compute the principal components of $\mathbf{A}$:

```
pcc=
   -0.2941   -0.0193   -0.3288   -0.2029
   -0.3021   -0.0414   -0.3586   -0.2072
   -0.2592   -0.2009    0.3244   -0.1377
   -0.3012   -0.2379    0.2416    0.1667
   -0.2833   -0.2276    0.2517    0.0071
    0.2603   -0.3879   -0.0717   -0.2790
    0.2437   -0.3683   -0.0229   -0.1669
   -0.2921   -0.0554   -0.3398   -0.2089
   -0.2921   -0.0554   -0.3398   -0.2089
   -0.2833   -0.2276    0.2517    0.0071
   -0.2833   -0.2276    0.2517    0.0071
   -0.0146   -0.4309   -0.4138    0.7722
    0.2437   -0.3683   -0.0229   -0.1669
    0.2573   -0.3633   -0.0349   -0.2277

    columns: principal components
    rows: days
```

```
scc=
   -0.6736    2.8514    0.1816    0.0140
   -3.2867    1.1052   -0.3078    0.0041
   -7.9357   -2.1006   -1.1928    0.3377
   -5.8909   -0.8154   -0.1672    0.3723
   -0.3940    2.4398    0.0966    0.9891
   -2.9700    1.5775    0.4645   -0.5609
   -8.1803   -1.8707   -0.4546   -0.5722
   -0.3649    3.3016    0.9241   -0.5553
    3.2162    2.6761    0.4037    0.1542
    4.2067    0.7574   -0.1624    0.0859
    5.1972   -1.1613   -0.7286    0.0175
    6.1877   -3.0800   -1.2947   -0.0508
    4.4640    0.3941   -0.1973   -0.1419
    5.4575   -1.5492   -0.8002   -0.2615
    0.9667   -4.5260    3.2352    0.1679

  columns: coordinates of customers
                    in pc basis
  rows: customers
```

```
pcc1=
mon    -0.2941
tue    -0.3021
wed    -0.2592
thu    -0.3012
fri    -0.2833
sat     0.2603
sun     0.2437
mon    -0.2921
tue    -0.2921
wed    -0.2833
thu    -0.2833
fri    -0.0146
sat     0.2437
sun     0.2573
```

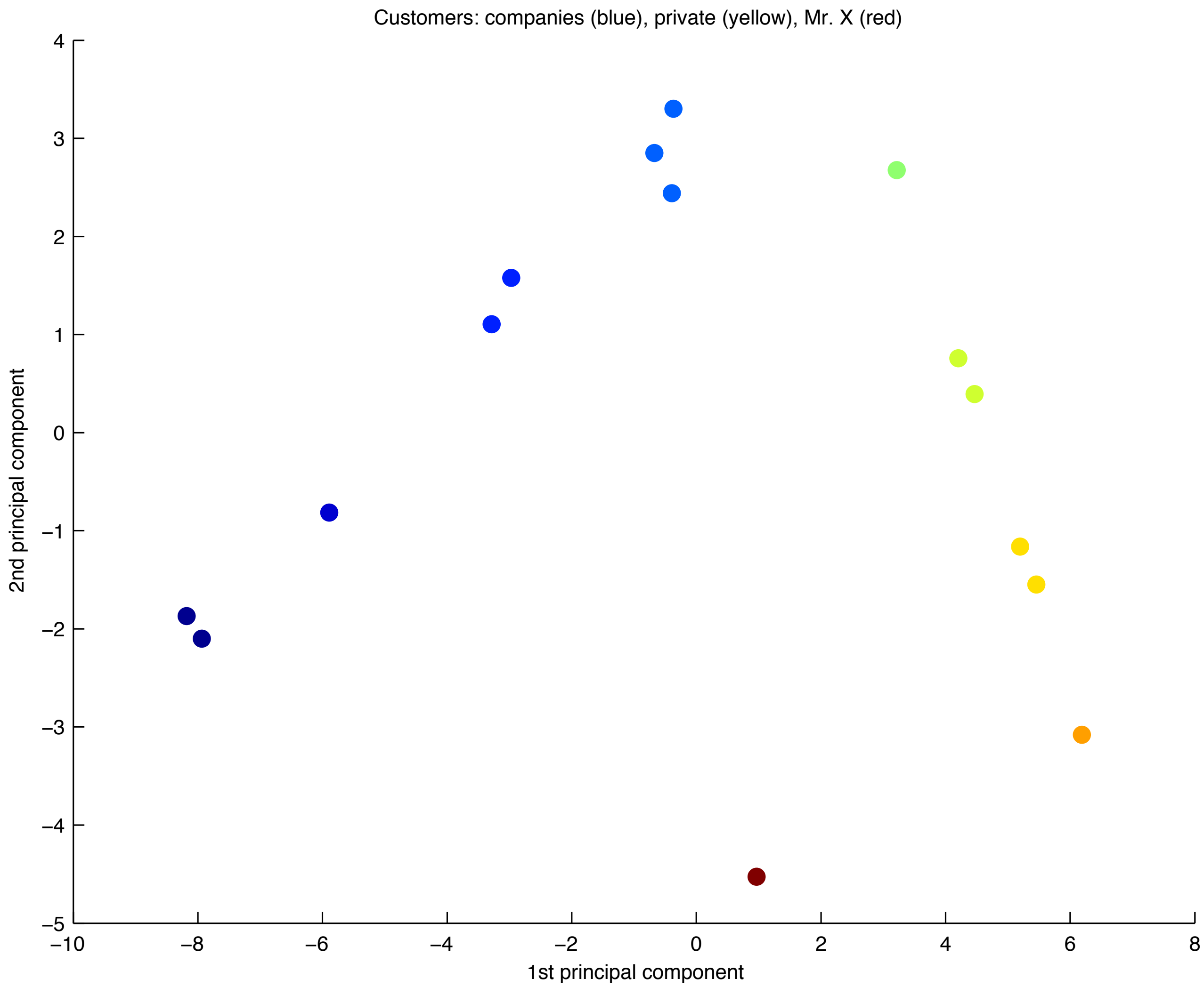
```
scc1=
ABC Ltd   -0.6736
BCD Inc   -3.2867
CDECorp   -7.9357
DEF Ltd   -5.8909
EFG Inc   -0.3940
FGHCorp   -2.9700
GHI Ltd   -8.1803
HIJ Inc   -0.3649
Smith      3.2162
Jones      4.2067
Brown      5.1972
Black      6.1877
Blake      4.4640
Lake       5.4575
Mr. X      0.9667
```

1st pc:
weekdays vs. weekends.
Result: weekday customers (companies) get separated from weekend customers (private citizens).
Big customers end up at exteme ends.

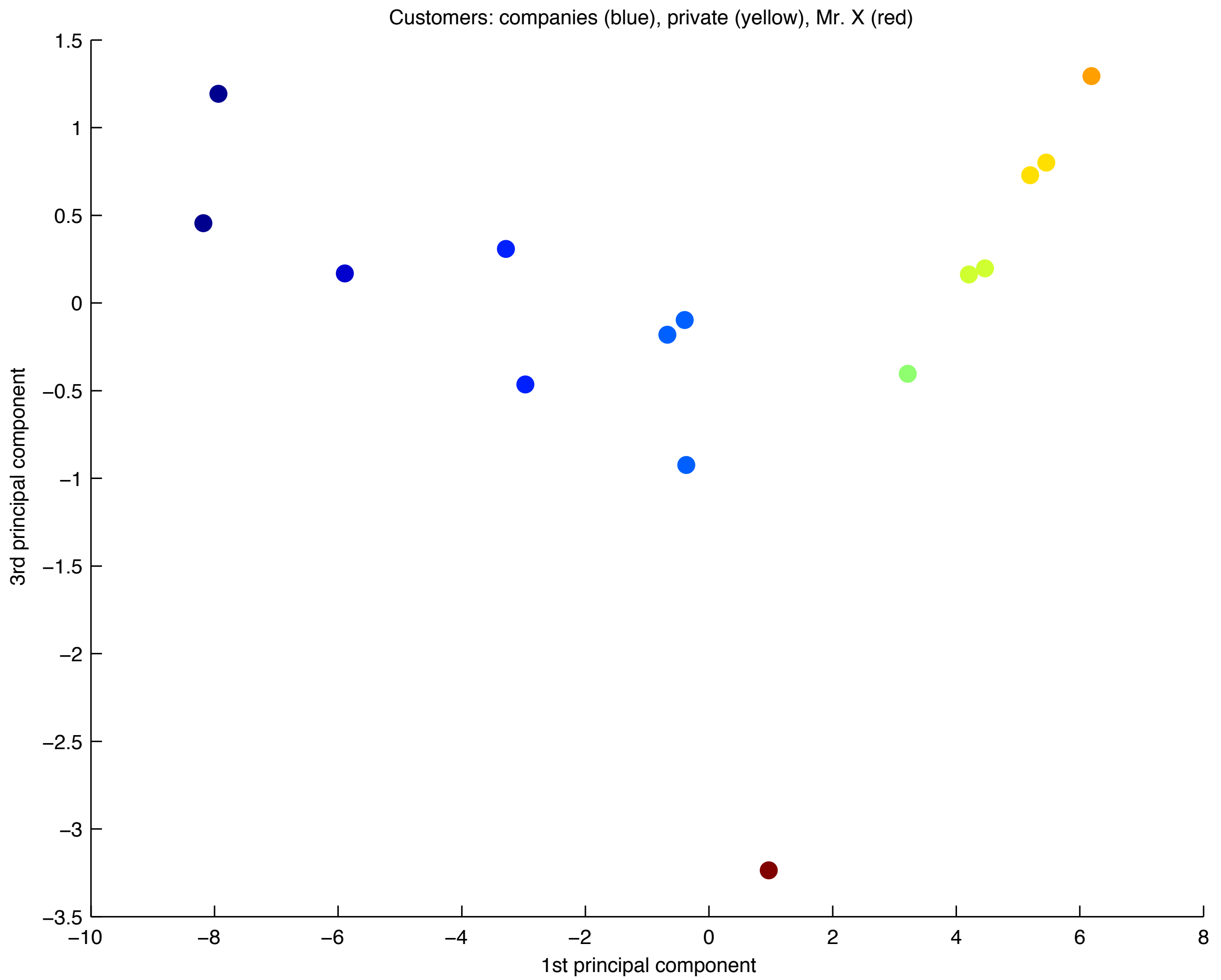
pcc2=	
mon	-0.0193
tue	-0.0414
wed	-0.2009
thu	-0.2379
fri	-0.2276
sat	-0.3879
sun	-0.3683
mon	-0.0554
tue	-0.0554
wed	-0.2276
thu	-0.2276
fri	-0.4309
sat	-0.3683
sun	-0.3633

scc2=	
ABD Ltd	2.8514
BCD Inc	1.1052
CDECorp	-2.1006
DEF Ltd	-0.8154
EFG Inc	2.4398
FGHCorp	1.5775
GHI Ltd	-1.8707
HIJ Inc	3.3016
Smith	2.6761
Jones	0.7574
Brown	-1.1613
Black	-3.0800
Blake	0.3941
Lake	-1.5492
Mr. X	-4.5260

1st pc:
weekends vs week days.
2nd pc:
Weekends and weekdays have about equal total weight. Most weight on exceptional friday.
Result: Separates big customers from small ones. Mr. X gets separated from the other customers.



Customers: companies (blue), private (yellow), Mr. X (red)
4
3
2
1
0
−1
−2
−3
−4
−5
−10
−8
−6
−4
−2
0
2
4
6
8
1st principal component
2nd principal component


Customers: companies (blue), private (yellow), Mr. X (red)
3rd principal component
1st principal component
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−2.5
−3
−3.5
−10
−8
−6
−4
−2
0
2
4
6
8

What if we transpose our problem?

Instead of thinking of customers as our data points, why not think of days as our data points, and customers as the attributes/variables?

The rows of $\mathbf{A}$ correspond to days.

The columns of $\mathbf{A}$ correspond to customers.

Let us compute the principal components of $\mathbf{A}$:

```
pcd=
   -0.1083    0.0278    0.1252   -0.7000
   -0.2043    0.1216    0.1983    0.7000
   -0.3732    0.3353    0.3538   -0.1000
   -0.2987    0.3213    0.1580   -0.1000
   -0.0880    0.2346    0.2239    0.0000
   -0.1989    0.0339   -0.0167   -0.0000
   -0.3902    0.2129    0.1214   -0.0000
   -0.1033   -0.0556   -0.0858   -0.0000
    0.1033    0.0556    0.0858    0.0000
    0.2066    0.1113    0.1716    0.0000
    0.3098    0.1669    0.2574    0.0000
    0.4131    0.2225    0.3432    0.0000
    0.2308    0.0903    0.1574    0.0000
    0.3344    0.1486    0.2483   -0.0000
    0.1506    0.7356   -0.6442    0.0000

    columns: principal components
    rows: customers
```

```
scd=
   -3.5718   -1.3482    1.0128   -0.7000
   -3.6677   -1.2544    1.0859    0.7000
   -2.6385    0.4954   -1.3990    0.1000
   -3.3104    1.1520   -0.8871   -0.1000
   -3.0117    0.8308   -1.0451    0.0000
    6.9418   -0.3998   -0.1646   -0.0000
    6.6073   -0.5484   -0.4129   -0.0000
   -3.4635   -1.3760    0.8876   -0.0000
   -3.4635   -1.3760    0.8876   -0.0000
   -3.0117    0.8308   -1.0451    0.0000
   -3.0117    0.8308   -1.0451    0.0000
    2.1560    3.1695    2.7936    0.0000
    6.6073   -0.5484   -0.4129   -0.0000
    6.8381   -0.4581   -0.2555    0.0000

    columns: coordinates of days in pc basis
    rows: days
```

Lets just look at the coordinates of the new data:

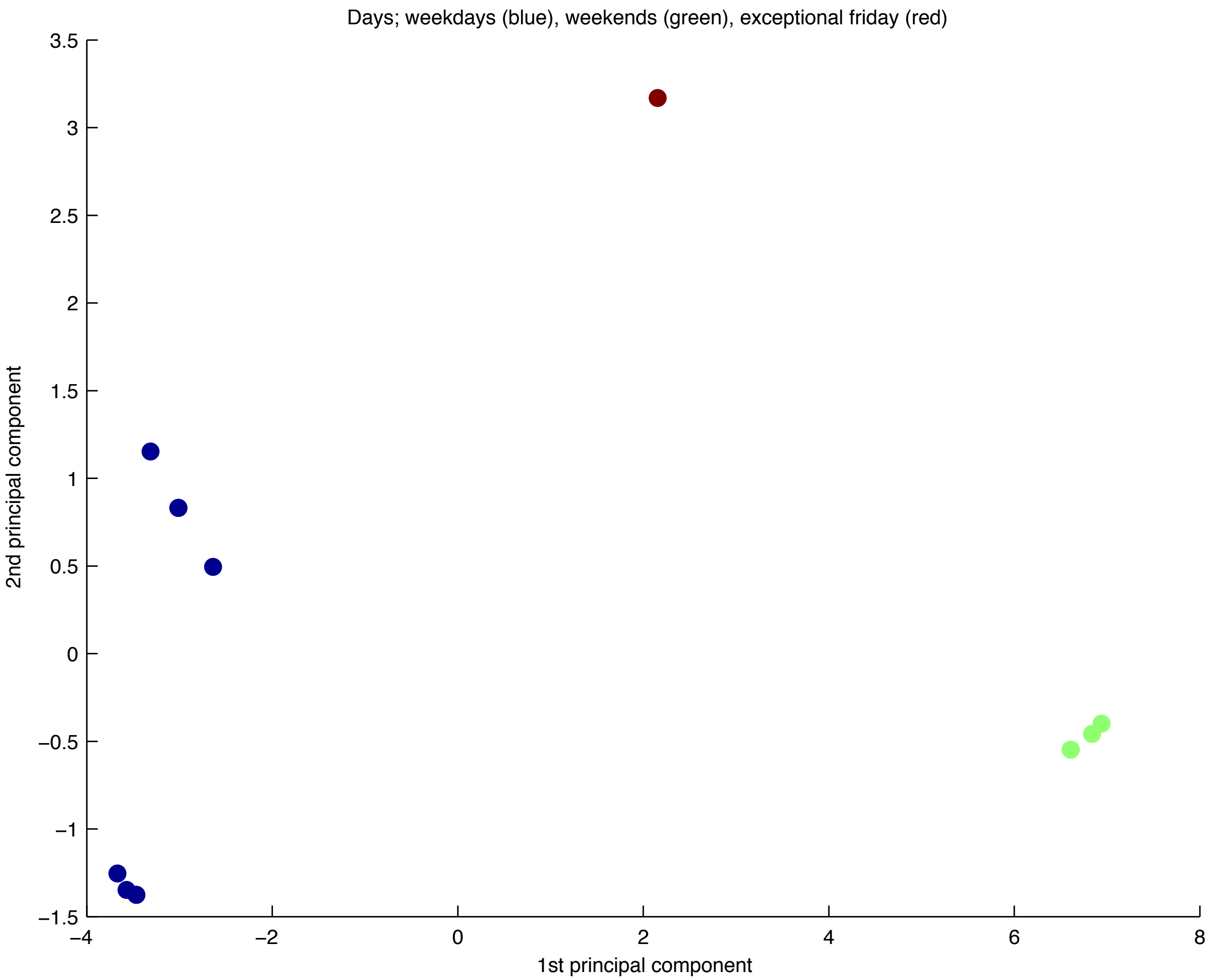
```
scd=
   -3.5718   -1.3482    1.0128   -0.7000
   -3.6677   -1.2544    1.0859    0.7000
   -2.6385    0.4954   -1.3990    0.1000
   -3.3104    1.1520   -0.8871   -0.1000
   -3.0117    0.8308   -1.0451    0.0000
    6.9418   -0.3998   -0.1646   -0.0000
    6.6073   -0.5484   -0.4129   -0.0000
   -3.4635   -1.3760    0.8876   -0.0000
   -3.4635   -1.3760    0.8876   -0.0000
   -3.0117    0.8308   -1.0451    0.0000
   -3.0117    0.8308   -1.0451    0.0000
    2.1560    3.1695    2.7936    0.0000
    6.6073   -0.5484   -0.4129   -0.0000
    6.8381   -0.4581   -0.2555    0.0000
```

Rows=days
1st col = projection along 1st pc: weekdays vs weekends
2nd col = projection along 2nd pc: Mr. X
3rd col = projection along 3rd pc: exceptional friday
4th column: Nothing left to explain, except differences between monday and tuesday...

Look at singular values

The singular values of the centered data matrix $\longrightarrow$
By looking at these you might already conclude that the first three principal components are enough to capture most of the variation in the data.

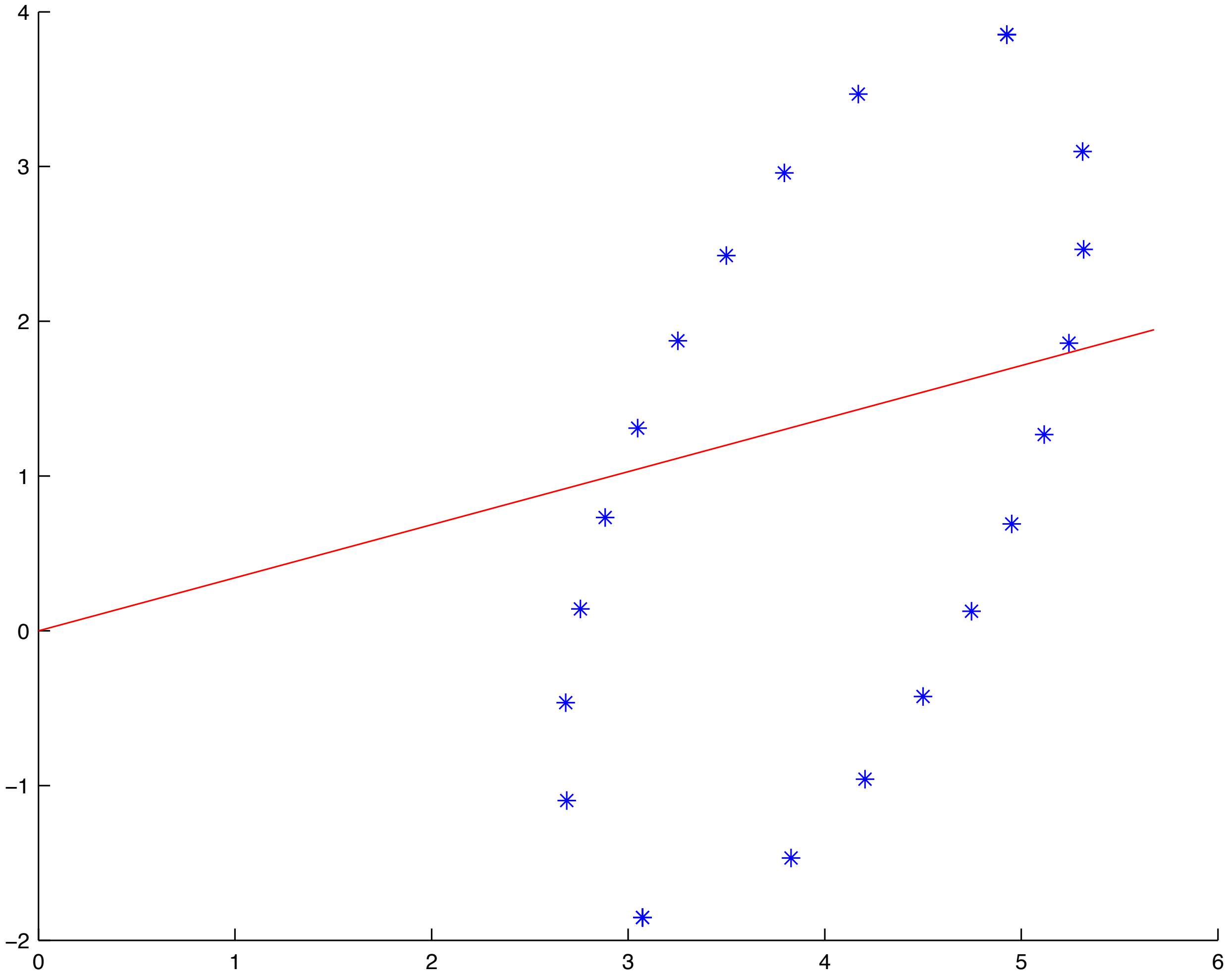
```
16.8001
 4.6731
 4.2472
 1.0000
 0.9957
 0.7907
 0.7590
 0.6026
 0.5025
 0.0000
 0.0000
 0.0000
 0.0000
 0.0000
```


Days; weekdays (blue), weekends (green), exceptional friday (red)
3.5
3
2.5
2
1.5
1
0.5
0
−0.5
−1
−1.5
2nd principal component
−4
−2
0
2
4
6
8
1st principal component

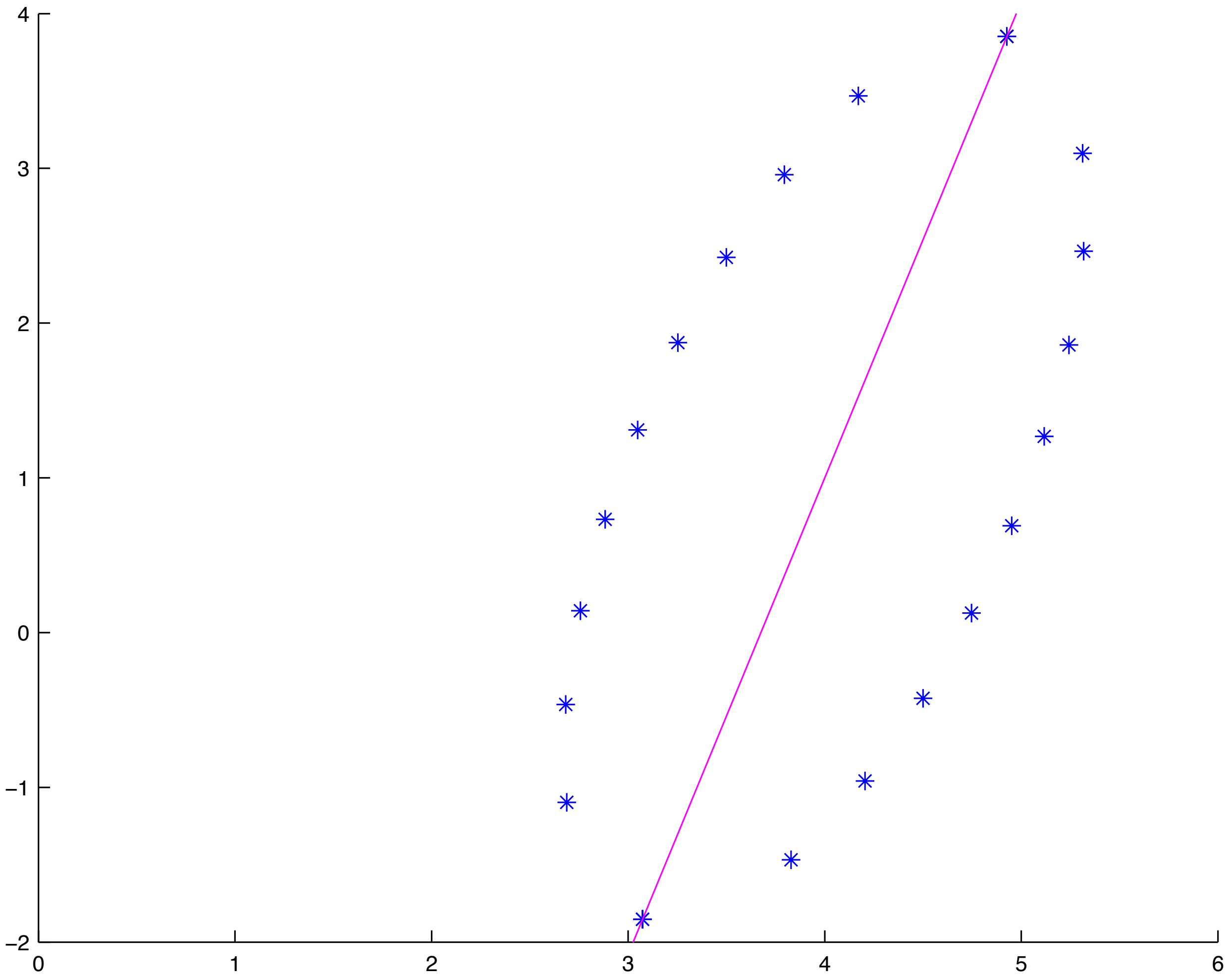
SVD vs PCA: Centering is central

SVD will give vectors that go through the origin.

Centering makes sure that the origin is in the middle of the data set.



4
3
2
1
0
−1
−2
0
1
2
3
4
5
6



4
3
2
1
0
−1
−2
0
1
2
3
4
5
6

Matrix decompositions revisited

- We wish to **decompose** the matrix $\mathbf{A}$ by writing it as a product of two or more matrices:

$$\mathbf{A}_{m\times n} = \mathbf{B}_{m\times k}\mathbf{C}_{k\times n}, \quad \mathbf{A}_{m\times n} = \mathbf{B}_{m\times k}\mathbf{C}_{k\times r}\mathbf{D}_{r\times n}$$

- This is done in such a way that the right side of the equation yields some useful information or insight to the nature of the data matrix $\mathbf{A}$.

- Or is in other ways useful for solving the problem at hand.

- examples: QR, SVD, PCA, NMF, Factor analysis, ICA, CUR, MPCA, AB,....

NMF = Nonnegative Matrix Factorization

- Given a nonnegative matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, we wish to express the matrix as a product of two nonnegative matrices $\mathbf{W} \in \mathbb{R}^{m \times k}$ and $\mathbf{H} \in \mathbb{R}^{k \times n}$:

$$\mathbf{A} \approx \mathbf{W}\mathbf{H}$$

Why require nonnegativity?

- nonnegativity is natural in many applications...
- term-document
- market basket
- etc

Example

3	2	4	0	0	0	0
1	0	1	0	0	0	0
0	1	2	0	0	0	0
0	0	0	1	1	1	1
0	0	0	1	0	1	1
0	0	0	3	2	2	3

Rows: customers, columns: products they buy.

Example continued

```
PC=
   -0.3751    0.4383   -0.8047
   -0.2728    0.2821    0.4166
   -0.5718    0.4372    0.4073
    0.3940    0.4311    0.0413
    0.2537    0.3341    0.0907
    0.2883    0.2322   -0.0378
    0.3940    0.4311    0.0413
```

Principal components. Each column corresponds to a product.

```
W =                                 H' =
         0    2.3515                         0    1.1206
         0    0.5073                         0    0.7992
         0    0.7955                         0    1.7347
    0.8760         0                    1.1918         0
    0.6937         0                    0.7532         0
    2.4179         0                    0.8510         0
                                        1.1918         0
```

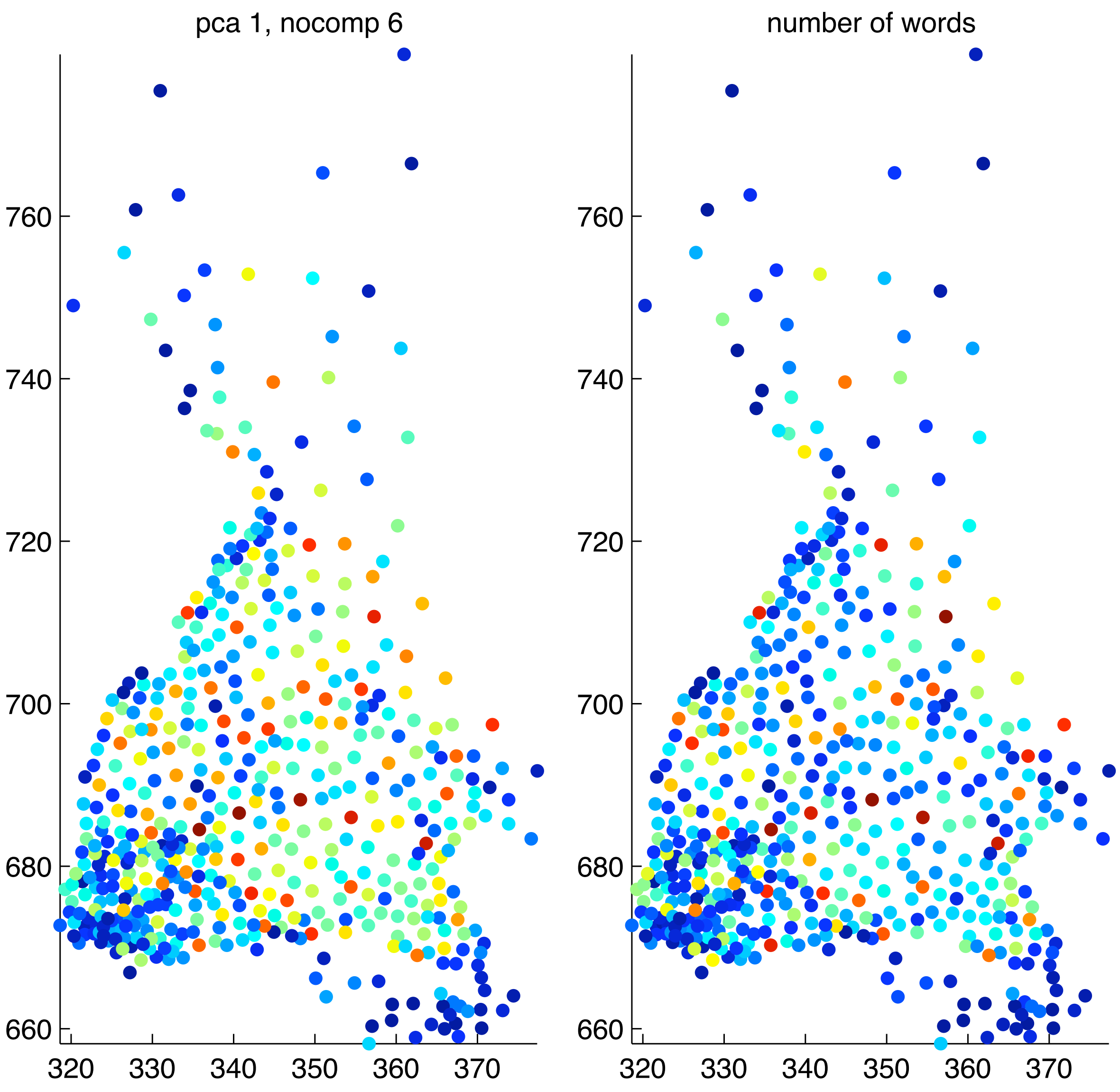
Rows of $\mathbf{W}$ are customers, rows of $\mathbf{H}^T$ are products.

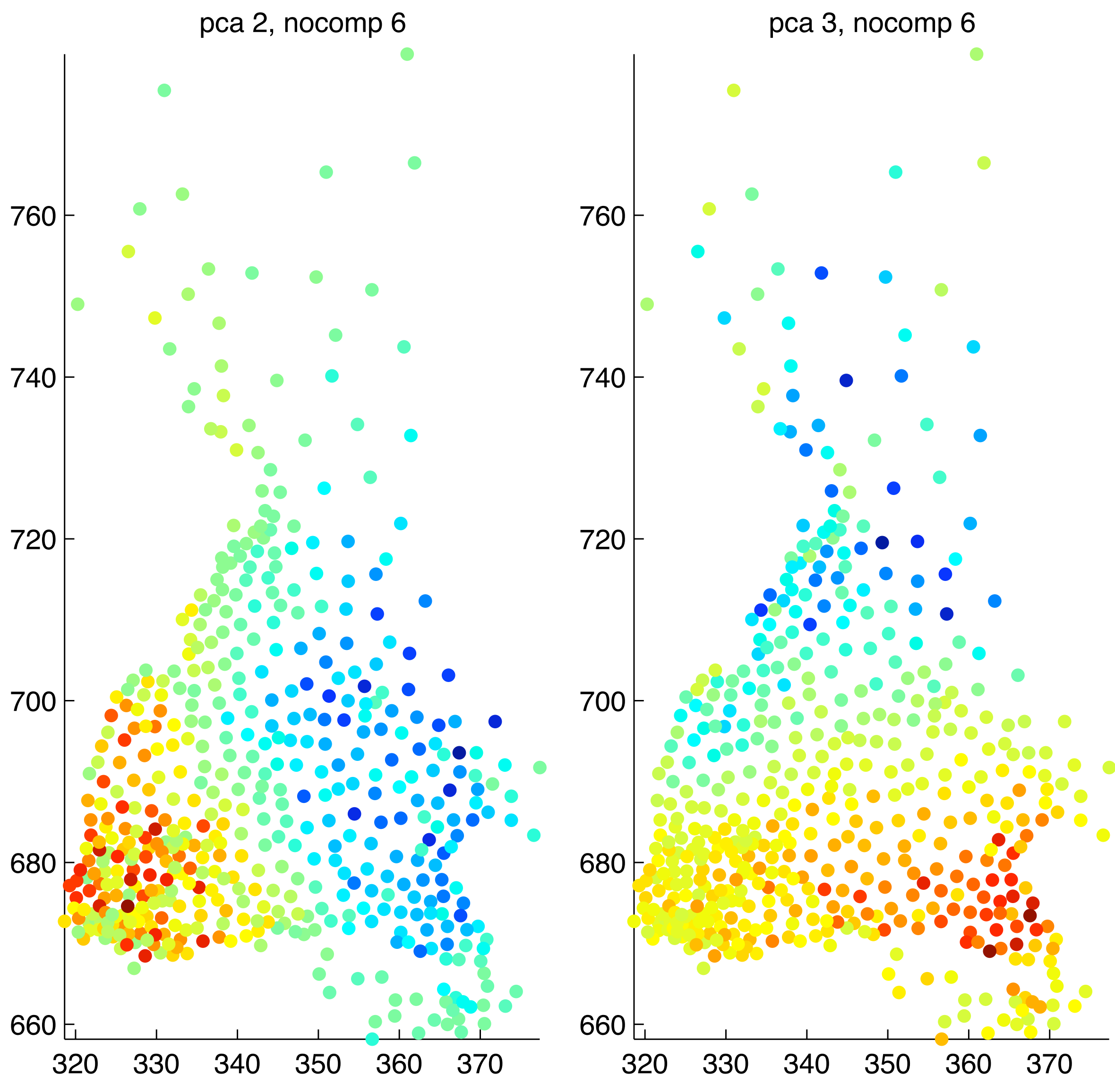
Example: finnish dialects revisited

- Data: 17000 dialect words, 500 counties.
- Word-county matrix $\mathbf{A}$:

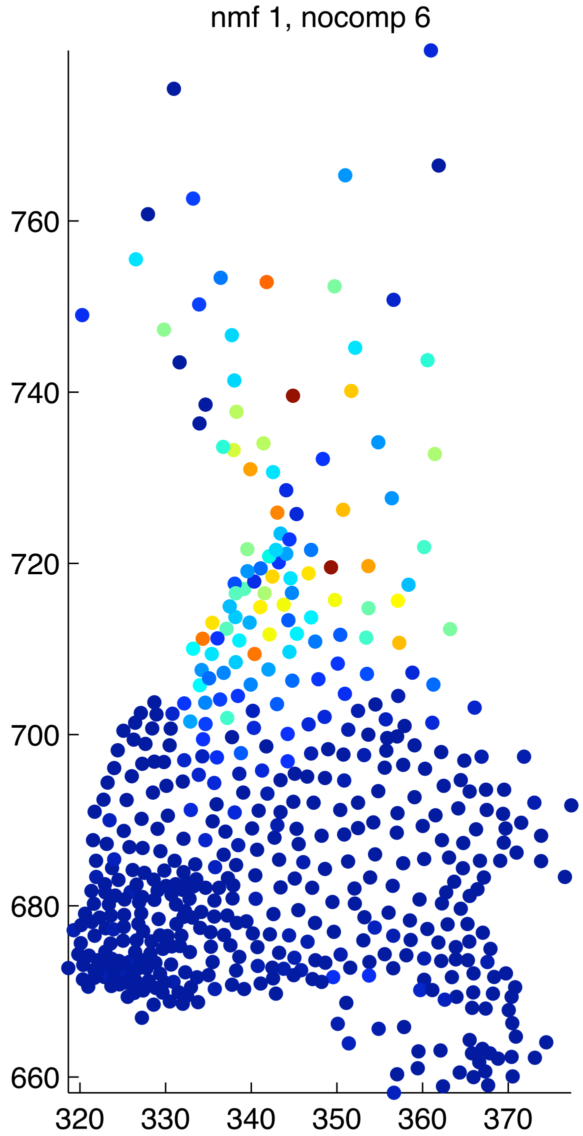
$$\mathbf{A}(i,j) = \begin{cases} 1 & \text{if word i appears in county j} \\ 0 & \text{otherwise.} \end{cases}$$

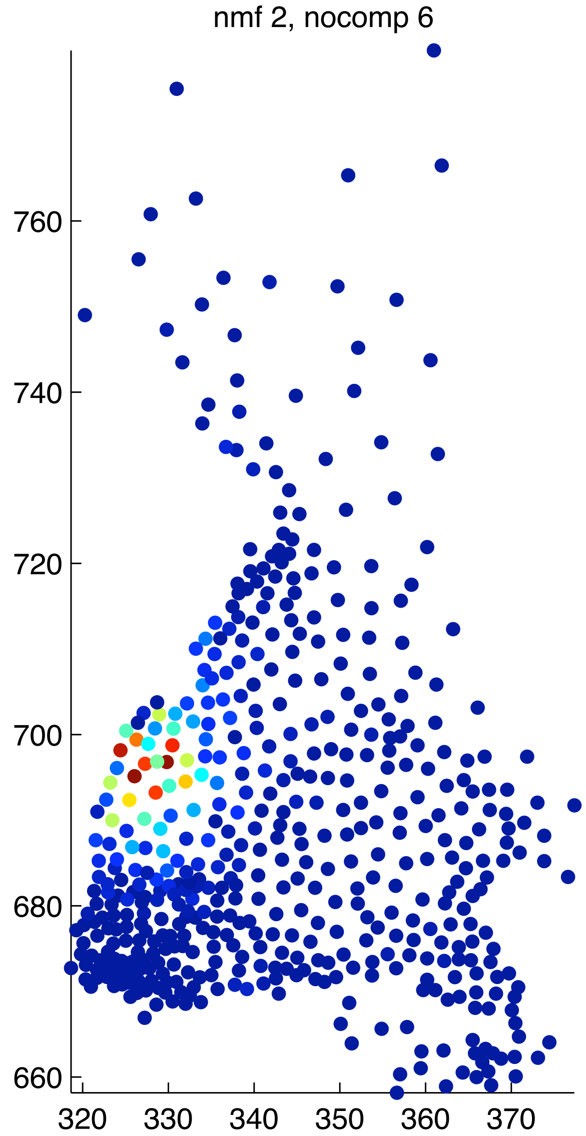
- Apply PCA to this: data points: words, variables: counties
- Each principal component tells which counties explain the most significant part of the variation left in the data.

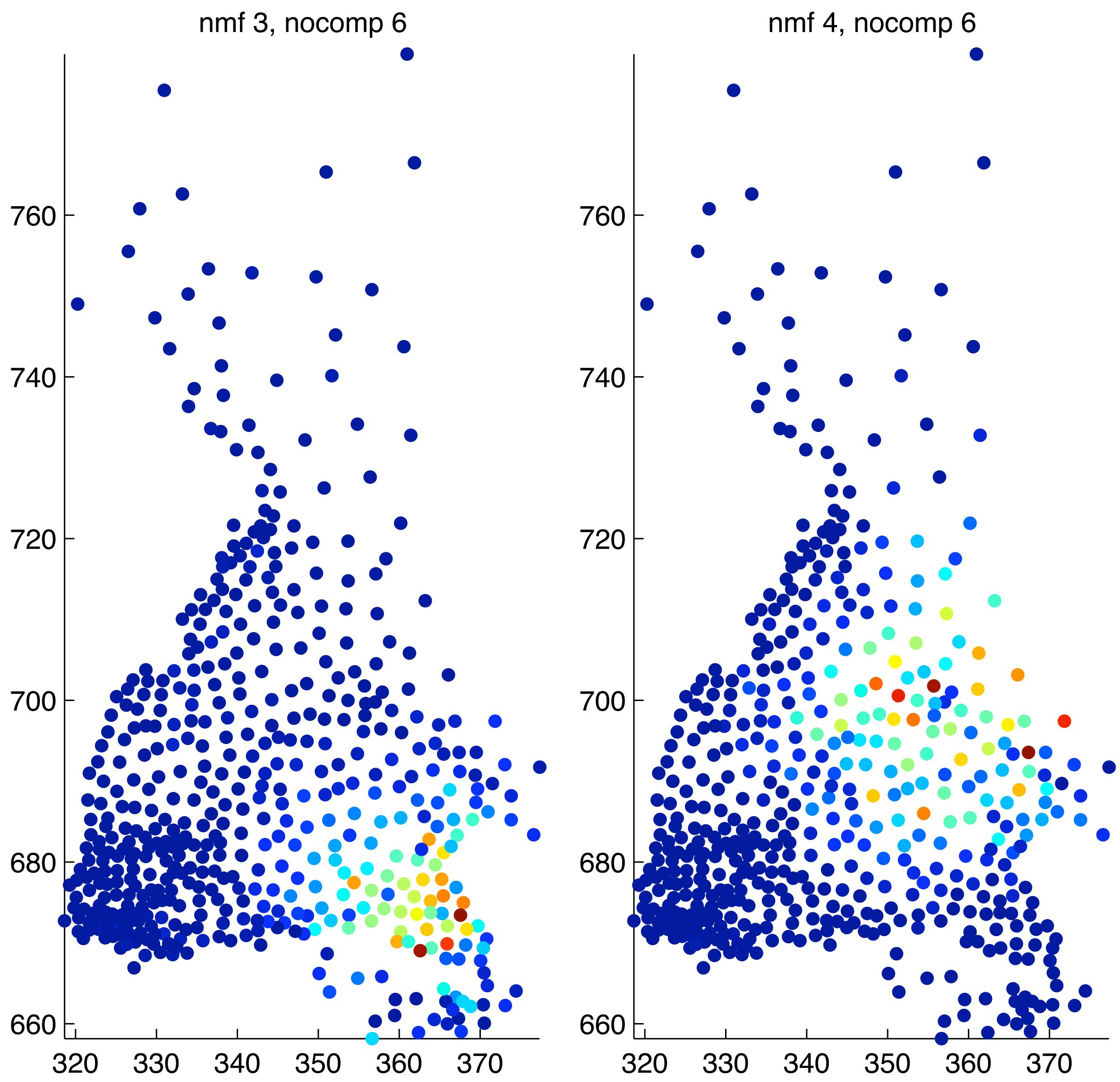

pca 1, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370
number of words
760
740
720
700
680
660
320
330
340
350
360
370

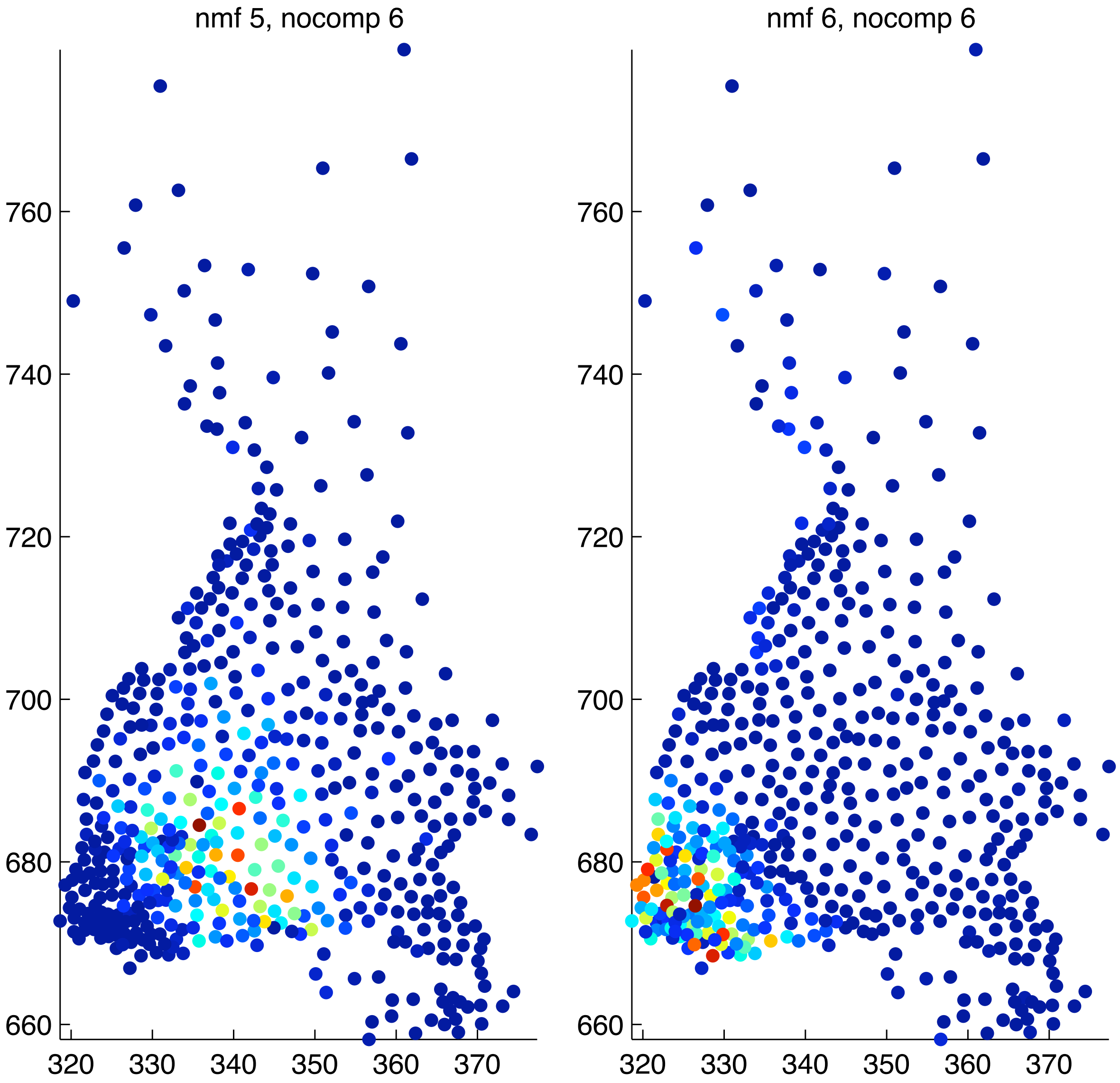

pca 2, nocomp 6
pca 3, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370

- Gives a general idea of how dialects vary...
- But (in general) does not capture local structure very well!
- What we would like would be a decomposition, where the components represent contributions of single dialects.
- NMF?


nmf 1, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370


nmf 2, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370


nmf 3, nocomp 6
nmf 4, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370


nmf 5, nocomp 6
nmf 6, nocomp 6
760
740
720
700
680
660
320
330
340
350
360
370

Results

- More local structure
- Components correspond to dialect regions
- Interpretation:
$$\mathbf{A} = \mathbf{W}\mathbf{H}$$
where $\mathbf{W} \in \mathbb{R}^{m \times k}$ is the "word per dialect region" matrix, and $\mathbf{H} \in \mathbb{R}^{k \times n}$ is the "dialect region per county" matrix.

How to compute NMF: Multiplicative algorithm

```
W=rand(m,k);
H=rand(k,n);
for i=1:maxiter
        H=H.*(W'*A)./(W'*W*H+epsilon);
        W=W.*(A*H')./(W*H*H'+epsilon);
end
```

Comments on the multiplicative algorithm

- Easy to implement.

- Convergence?

- Once an element is zero, it stays zero.

How to compute NMF: ALS

$\mathbf{W} = \text{rand}(m, k)$;

for i=1:maxiter

Solve for $\mathbf{H}$ in equation $\mathbf{W}^T\mathbf{W}\mathbf{H} = \mathbf{W}^T\mathbf{A}$

Set all negative elements in $\mathbf{H}$ to 0.

Solve for $\mathbf{W}$ in equation $\mathbf{H}\mathbf{H}^T\mathbf{W}^T = \mathbf{H}\mathbf{A}^T$.

Set all negative elements of W to zero.

end

Comments on the ALS algorithm

- Can be very fast (depending on implementation)

- Convergence?

- Sparsity

- Improved versions exist

Uniqueness of NMF

- NMF is not unique: let $\mathbf{D}$ be a diagonal matrix with positive diagonal entries. Then

$$\mathbf{WH} = (\mathbf{WD})(\mathbf{D}^{-1}\mathbf{H})$$

Initialization

- Convergence can be slow.
- It can be speeded up using a good initial guess: initialization.
- A good initialization can be found using SVD (see p. 106)

Summary

- Given a nonnegative matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ find nonnegative matrices $\mathbf{W} \in \mathbb{R}^{m \times k}$ and $\mathbf{H} \in \mathbb{R}^{k \times n}$ so that

$$\|\mathbf{A} - \mathbf{W}\mathbf{H}\|$$

  is minimized.

- Algorithms exist, both basic (easy to implement) and more advanced (implementing e.g. sparsity constraints)

- Interpretability

Applications

- text mining
- email surveillance
- music transcription
- bioinformatics
- source separation
- spatial data analysis
- etc

References

[1] Lars Eldén: Matrix Methods in Data Mining and Pattern Recognition, SIAM 2007.

[2] Berry, Browne, Langville, Pauca, Plemmons: Algorithms and Applications for Approximate Nonnegative Matrix Factorization