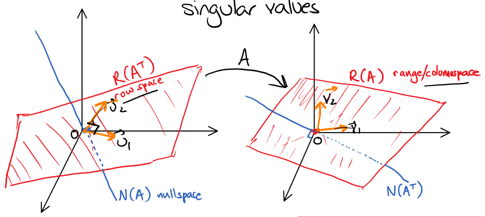
Low-Rank Matrices

$\text{Rank}(A) = \dim R(A) = \dim(\text{span of columns})$
$= \dim R(A^T) = \dim(\text{span of rows})$

SVD: $A = \sum_i \sigma_i \vec{v}_i \vec{u}_i^T$

$\text{Rank}(A) = \#$ of nonzero singular values



$\boxed{\text{Rank}(A) + \dim N(A) = \text{number of columns}}$ $\boxed{\text{Rank}(A) + \dim N(A^T) = \text{number of rows}}$

- $\text{Rank}(A) = 0 \iff A = \begin{pmatrix} \text{all } 0 \\ \text{matrix} \end{pmatrix}$
- $\text{Rank}(A) = 1 \iff A = \vec{v}\vec{u}^T$ with $\vec{u}, \vec{v}$ nonzero
  $\Updownarrow$
  $A = \sigma_1 \vec{v}_1 \vec{u}_1^T$
  $\begin{pmatrix} 1 & 1 & 1 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 3 & 0 \\ 3 & 3 & 3 & 9 & 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 3 \end{pmatrix} (1\ 1\ 1\ 3\ 0)$
- $\text{Rank}(A) = 2$ $\quad A = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T$

  $\text{Rank}(\underbrace{\vec{u}\vec{v}^T}_{=A} + \vec{\omega}\vec{x}^T) \le 2$

  $R(A) = \text{Span}\{\vec{u}, \vec{\omega}\}$
  $R(A^T) = \text{Span}\{\vec{v}, \vec{x}\}$
- $\text{Rank}(A) = 3 \iff A = \vec{u}\vec{v}^T + \vec{\omega}\vec{x}^T + \vec{y}\vec{z}^T$
  for some $\vec{u}, \vec{\omega}, \vec{y}, \ \vec{v}, \vec{x}, \vec{z}$

  $\vdots$
  and so on

Observe: $\text{Rank}(A+B) \le \text{Rank}(A) + \text{Rank}(B)$ $\qquad \text{Rank}(A \cdot B) \le \min\begin{pmatrix} \text{Rank } A, \\ \text{Rank } B \end{pmatrix}$

Observe: Low-rank matrices have fewer parameters.

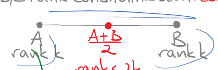
$\text{Rank}(A) = k \Rightarrow \underset{m\times n}{A} = \sum_{i=1}^{k} \vec{u}_i \vec{v}_i^T$ for some vectors

$= \underset{m \times k}{\begin{pmatrix} \vec{u}_1 & \cdots & \vec{u}_k \end{pmatrix}} \cdot \begin{pmatrix} \text{—— } \vdots \text{ ——} \\ \text{—— } \vec{v}_k^T \text{ ——} \end{pmatrix}$

$\Rightarrow k \cdot (m+n)$ parameters $\ll m \cdot n$ for small k

Generic matrices, eg., random matrices, will have full rank
$\text{rank}(A_{m\times n}) = \min\{m, n\}$.
but we would prefer to use low-rank matrices when possible.

Optimizing over low-rank matrices ← b/c rank constraints aren't convex
~ a common problem, often NP-hard
(eg., https://en.wikipedia.org/wiki/Matrix_completion
https://en.wikipedia.org/wiki/Low-rank_approximation)



$\text{Rank}(A) \le k \Rightarrow A = \vec{u}_1 \vec{v}_1^T + \cdots + \vec{u}_k \vec{v}_k^T$
rowspace $= \text{Span}\{\vec{v}_1, \ldots, \vec{v}_k\}$.
each row is a combination, eg.,
row 1 $= e_1^T A = (u_1)_1 \vec{v}_1^T + \cdots + (u_k)_1 \vec{v}_k^T$

$\begin{pmatrix} 1 & 2 & 3 \\ 4 & * & 0 \end{pmatrix}$

The Singular-Value Decomposition and Distance to Lower-Rank Matrices

Example: $A = \begin{pmatrix} 1 & & & 0 \\ & 1/2 & & \\ & & 1/3 & \\ 0 & & & 1/4 \end{pmatrix}_{4\times 4}$ $\quad \text{rank}(A) = 4$

Intuitively
What is the rank-one matrix closest to A? $\quad \underset{B:\ \text{rank} B = 1}{\text{argmin}} \|A - B\| = \begin{pmatrix} 1 & & & 0 \\ & 0 & & \\ & & 0 & \\ 0 & & & 0 \end{pmatrix}$

Closest rank-two matrix? → $\begin{pmatrix} 1 & & & \\ & 1/2 & & \\ & & 0 & \\ & & & 0 \end{pmatrix}$

e approximations

Closest rank-two matrix? $\rightarrow \begin{pmatrix} 1 & & & \\ & 1/2 & & \\ & & 0 & \\ & & & 0 \end{pmatrix}$

Closest rank-3 matrix? $\rightarrow \begin{pmatrix} 1 & & & \\ & 1/2 & & \\ & & 1/3 & \\ & & & 0 \end{pmatrix}$

successive approxim

Theorem 1: Let A have SVD

$$A = \sigma_1 v_1 u_1^T + \cdots + \sigma_n v_n u_n^T$$

with sorted singular values

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$

Then for any $k \leq r$, the closest rank k matrix to A is

$$B = \sum_{i=1}^{k} \sigma_i v_i u_i^T$$

Observe: B truncates the smaller singular values of A.

$\|A - B\| = \sigma_{k+1}$

Example: $\begin{pmatrix} 1 & & & 0 \\ & 1/1000 & & \\ & & 1/1000 & \\ 0 & & & 1/1000 \end{pmatrix}$ has full rank, but is $1/1000$-close to a rank-one (singular) matrix.

Proof: Let C be a rank k matrix.

We want to show that $\|A - C\| \geq \|A - B\| = \underbrace{\left\| \sum_{i=k+1}^{n} \sigma_i v_i u_i^T \right\|}_{= \sigma_{k+1}}$

How? It is enough to find $\vec{x}$ such that

$\|(A - C)\vec{x}\| \geq \sigma_{k+1} \|\vec{x}\|$. ✓

Let $S = \mathrm{Span}\{u_1, u_2, \ldots, u_k, u_{k+1}\}$ $\quad \dim S = k+1$

$\dim N(C) = n - k$

$\Rightarrow \dim(S \cap N(C)) \geq (\dim S) + (\dim N(C)) - n$

$= (k+1) + (n-k) - n$

$= 1$

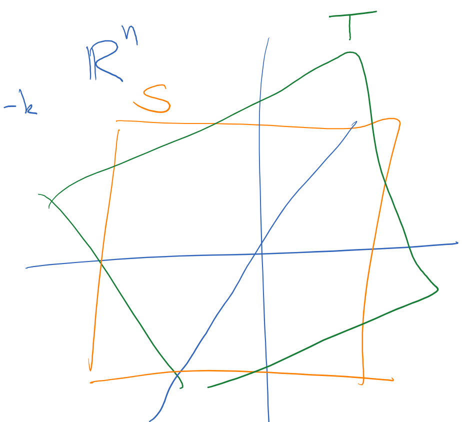
$\Rightarrow \exists\, \vec{x} \neq \vec{0}$ s.t. $\vec{x} \in S$ and $x \in N(C)$

$\Downarrow \qquad\qquad \Downarrow$

$\|Ax\| \geq \sigma_{k+1}\|x\| \qquad C\vec{x} = \vec{0}$

$\Rightarrow \|(A - C)\vec{x}\| = \|Ax\| \geq \sigma_{k+1}\|x\|$. ✓ ◻

If $A = \sum_{i=1}^{n} \sigma_i \vec{v}_i \vec{u}_i^T$ $\quad \mathrm{Rank}(A) = \#\{i \mid \sigma_i > 0\}$



$\dim S \cap T \geq \dim S + \dim T - n$

Theorem 2: The same holds for Frobenius norm:

$B = \sum_{i=1}^{k} \sigma_i v_i u_i^T$ achieves $\min_{B : \mathrm{rank}(B) = k} \|A - B\|_F$

Frobenius norm

$$\|A\|_F = \sqrt{\sum_{i,j} |a_{ij}|^2} = \sqrt{\mathrm{Tr}(A^\dagger A)}$$

$\max_{ij} |A_{ij}| \leq \|A\| \leq \|A\|_F$

easy to compute!

matrix inner product $\langle A, B \rangle := \mathrm{Tr}(A^\dagger B)$

Exercise: This satisfies

- $\|cA\|_F = |c| \cdot \|A\|_F$
- $\|A + B\|_F \leq \|A\|_F + \|B\|_F$
- $\|A \cdot B\|_F \leq \|A\|_F \cdot \|B\|_F$ (like operator norm)

Claim: The Frobenius norm is "basis-independent", i.e.,

$$\|A\|_F = \|\mathcal{U} A V\|_F$$

for any unitary matrices $\mathcal{U}, V$. $\quad A^\dagger = A^{T*}$

Proof: $\|A\|_F^2 = \sum_{ij} |A_{ij}|^2 = \mathrm{Tr}(A^\dagger A)$

$= \sum_j (A^\dagger A)_{jj} = \sum_{jk} (A^\dagger)_{jk} A_{kj}$

$= \sum_{jk} A^*_{kj} A_{kj} = \sum_{jk} |A_{kj}|^2$

$\|UAV\|_F^2 = \mathrm{Tr}((UAV)^\dagger (UAV))$

$= \mathrm{Tr}(V^\dagger A^\dagger \underbrace{U^\dagger U}_{I} A V)$ $\quad U^\dagger = U^{-1},\ V^\dagger = V^{-1}$

$= \mathrm{Tr}(V^\dagger A^\dagger A V)$

Fact: Trace is cyclic:

$A = \begin{pmatrix} \xrightarrow{a_1} \\ \\ \\ \end{pmatrix}$ n, m

data point

data set

$\sum_{j=1}^{m} \|a_j - b_j\|^2$

$B = \begin{pmatrix} \xrightarrow{b_1} \\ \\ \\ \end{pmatrix}$ rank 2

projected to a 2D space

$$\|UAV\|_F = Tr((UAV)^\dagger (UAV))$$
$$= Tr(V^\dagger A^\dagger \underbrace{U^\dagger U}_{I} A V) \qquad U^\dagger = U^{-1},\ V^\dagger = V^{-1}$$
$$= Tr(V^\dagger A^\dagger A V)$$
$$= Tr(V V^\dagger A^\dagger A)$$
$$= Tr(A^\dagger A) = \|A\|_F^2 \qquad \square$$

Fact: Trace is cyclic:

$$Tr(AB) = Tr(BA)$$

Proof: $Tr(AB) = \sum_j (AB)_{jj}$
$$= \sum_{jk} A_{jk} B_{kj} = \sum_{jk} B_{kj} A_{jk}$$
$$= \sum_k (BA)_{kk} = Tr(BA). \quad \square$$

$$Tr(ABC) = Tr(CAB) = Tr(BCA)$$

$\Rightarrow$ Trace is basis-independent

Let U be any invertible matrix.

$$Tr(UAU^{-1}) = Tr(\underbrace{U^{-1}U}_{I} A) = Tr(A)$$

$$Tr(ABC) = Tr(CAB) = Tr(BCA)$$

Proof: $Tr(AB) = \sum_i (AB)_{i,i} = \sum_{i,j} a_{ij} b_{ji} = \sum_{i,j} b_{ji} a_{ij} = Tr(BA)$ $\square$

Hence

$$\|AV\|_F^2 = Tr[(AV)^\dagger (AV)]$$
$$= Tr(V^\dagger A^\dagger A V)$$
$$= Tr(A^\dagger A V V^\dagger) \quad \text{by cyclic trace}$$
$$= Tr(A^\dagger A) \quad V \text{ unitary means } V^\dagger = V^{-1}.$$

Similarly $\|UA\|_F = \|A\|_F$. $\square$

(The Frobenius norm is the same in all orthonormal bases.)

Theorem 2: The same holds for Frobenius norm:

$B = \sum_{i=1}^{k} \sigma_i \vec{v}_i \vec{u}_i^T$ achieves $\min_{B:\,rank(B)=k} \|A-B\|_F = \sqrt{\sum_{j\geq k+1} \sigma_j^2}$

Interpretation:

$\|A-B\|_F^2 = \sum \|\text{row vectors}\|^2$

$Rank(B) = k \Rightarrow$ all its rows lie in k-dim. subspace

If $A = \begin{pmatrix} — \vec{a}_1^T — \\ \vdots \\ — \vec{a}_m^T — \end{pmatrix}$, $B = \sum_i e_i b_i^T$

$$\|A-B\|_F^2 = \sum_i \|\vec{a}_i - \vec{b}_i\|^2$$

The optimization is equivalent to

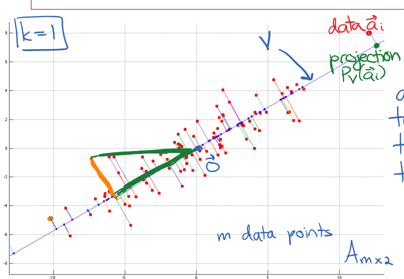
1. Choose a k-dim subspace $V \subseteq \mathbb{R}^n$
2. Choose $\vec{b}_1, \dots, \vec{b}_m \in V$

$\vec{b}_i$ should minimize $\|\vec{a}_i - \vec{b}_i\|$

$\Rightarrow \vec{b}_i = Proj_V(\vec{a}_i)$!

Hence:

$$\boxed{\min_{B:\,rank(B)=k} \|A-B\|_F^2 = \min_{\substack{V\subseteq \mathbb{R}^n \\ dim(V)=k}} \sum_{i=1}^{m} \|\vec{a}_i - Proj_V(\vec{a}_i)\|^2} = \sum_{j\geq k+1} \sigma_j^2$$



If the rows of A $\vec{a}_1, \dots, \vec{a}_m$ are data points, we want to find the dim-k subspace that minimizes the sum of the squared errors.

$A_{m\times 2}$ $\quad B_{m\times 2}$

Claim: The best subspace is

$V = Span\{\text{first } k \text{ right sing. vectors of } A\}$.

$\vec{v}_1, \vec{v}_2, \dots, \vec{v}_k$

$$\min_{B:\,rank(B)=k} \|A-B\|_F^2$$
$$= \min_{V\subseteq \mathbb{R}^n} \sum_{i=1}^{m} \|\vec{a}_i - Proj_V(\vec{a}_i)\|^2$$

Proof:

$\|A-B\|_F = \|U(A-B)V\|_F$

for any unitaries U, V

$$\min_{B:\ \mathrm{rank}(B)=k} \|A-B\|_F$$

$$= \min_{\substack{V\subseteq\mathbb{R}^n\\ \dim(V)=k}} \sum_{i=1}^{m} \|\vec{a}_i - \mathrm{Proj}_V(\vec{a}_i)\|^2$$

Proof:

$\|A-B\|_F = \|U(A-B)V\|_F$ for any unitaries U, V

In particular, we can work in the bases of left & right sing. vectors

in those bases

$$[A] = \begin{array}{c|ccc} & u_1 & u_2 & u_3 \\ \hline v_1 & \sigma_1 & 0 & 0 \\ v_2 & 0 & \sigma_2 & 0 \\ v_3 & 0 & 0 & \sigma_3 \ \ddots \end{array}$$

Now proof is by induction.

$\boxed{k=1}$: Goal: $B = \begin{pmatrix} \sigma_1 & & 0 \\ & 0 & \\ 0 & & \ddots \end{pmatrix}$ minimizes $\|A-B\|_F$ where B has rank 1.

Proof: B has rank 1 $\quad R(B^T) = \mathrm{Span}(\hat{u})$

$\Rightarrow B = \vec{v}\vec{u}^T$

We can assume that $\|u\| = 1$

Then $\|A-B\|_F^2 = \sum_{j=1}^{n} \|\vec{a}_j - P_{\vec{u}}\vec{a}_j\|^2 \qquad \|a\| = a^\dagger a$

$\underbrace{\frac{uu^T a_j}{\|u\|^2}}_{} = (u\cdot a_j)u$

$= \sum_j \|(\underbrace{I - uu^T}_{P_{u^\perp}})a_j\|^2 = \sum_j a_j^\dagger \underbrace{P_{u^\perp}^\dagger P_{u^\perp}}_{P_{u^\perp}P_{u^\perp} = P_{u^\perp}} a_j$

bases

$= \sum_j a_j^\dagger P_{u^\perp} a_j = \sum_j a_j^\dagger (I - P_u) a_j$

$= \sum_j (a_j^\dagger a_j - a_j^\dagger P_u a_j) = \|A\|_F^2 - \sum_j |u \cdot a_j|^2 \qquad a_j = \sigma_j \vec{e}_j$

$= \|A\|_F^2 - \sum_j \sigma_j^2 u_j^2$

Now we have to

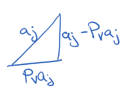
$\max \sum_j \sigma_j^2 u_j^2 \qquad \sigma_1 \geq \sigma_2 \geq \cdots > 0$

s.t. $\sum_j u_j^2 = 1$

$\Rightarrow$ let $\vec{u} = \vec{e}_1$ achieves the max

$\|A-B\|_F^2 = \|A\|_F^2 - \sigma_1^2 = \sum_{j\geq 2} \sigma_j^2$

$\|A\|_F^2 = \sum_j \sigma_j^2$



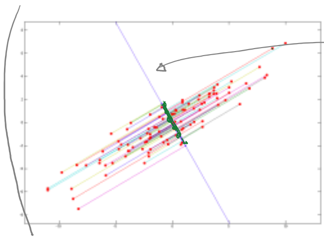
Induction step:

Observe: $\overset{\min}{\sum_j} \|a_j - P_V a_j\|^2 = \sum_j (\|a_j\|^2 - \|P_V a_j\|^2)$

$= \|A\|_F^2 - \sum_j \|P_V a_j\|^2$ ← max

$\therefore$ minimizing sum of squared distances to V $\iff$ maximizing $\sum_j \|P_V(\vec{a}_j)\|^2$

i.e., maximizing the projected data's variance

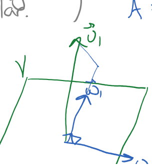


In contrast, projecting to this line minimizes the projections' variance

Let us show the induction for $k=2$. (General argument is similar.)

$$A = \begin{array}{c|ccc} & u_1 & u_2 & u_3 \\ \hline v_1 & \sigma_1 & & \\ v_2 & & \sigma_2 & \\ & & & \sigma_3 \ \ddots \end{array}$$

Let V achieve $\max_{\dim(V)=2} \sum_j \|P_V(\vec{a}_j)\|^2$.

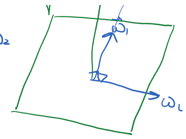


We can choose an orthonormal basis $\vec{\omega}_1, \vec{\omega}_2$

s.t. $\vec{\omega}_2 \perp \vec{u}_1$

$\sum \|P_V(\vec{a})\|^2 = \sum |\ \ \ |^2 + \sum |\ \ \ |^2$

dim(V)=2



We can choose an orthonormal basis $\vec{\omega}_1, \vec{\omega}_2$ s.t. $\vec{\omega}_2 \perp \vec{v}_1$

$$\sum_j \|P_V(\vec{a}_j)\|^2 = \sum_j |a_j \cdot \omega_1|^2 + \sum_j |a_j \cdot \omega_2|^2$$

$$\sum_j |a_j \cdot v_1|^2 \quad \text{by the } k=1 \text{ case}$$

For this to be maximal, it must be that $\vec{\omega}_1 = \pm \vec{v}_1$.
Then we just need to maximize over vectors $\vec{\omega}_2$ perpendicular to $\vec{v}_1$.
But that's the $k=1$ problem! So $\vec{\omega}_2 = \vec{v}_2$ $\square$

Observations:

① The optimal low-rank approximation B is diagonal with respect to the same singular vector bases as A. (Off-diagonal terms don't help.) $\|A-B\|_F^2 = \sum_{j=k+1}^{n} \sigma_j^2$

② You don't need to compute the full SVD! Compute the sing vectors one at a time, largest sing. value to smallest, until you are happy with the approximation. (The algorithm is greedy.) $\|A\|_F^2 = \sum_{j=1}^{n} \sigma_j^2$

$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$\|A\| = \sigma_1$

$\sigma_2 = \sqrt{\|A\|_F^2 - \|A\|^2}$

$\frac{1}{\|A^{-1}\|}$

$|a|^2+|b|^2+|c|^2+|d|^2$

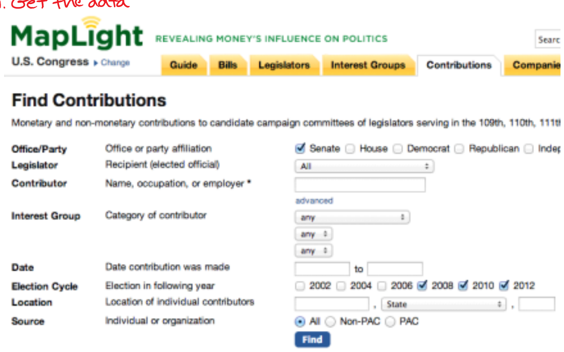
Remark:
Dimension reduction is a major theme in computational linear algebra and most approaches for data analysis.

- Linear regression, as we have presented it, effectively reduces the dimension of the data by 1, eg., fitting a 1D line to a 2D cloud of data points.
  The Johnson-Lindenstrauss Lemma projects n data points to $\log n$ (random) dimensions, approximately preserving angles & lengths (after rescaling).
- Matrix rank reduction via the SVD is another example. Moreover, it is used in Principal Component Analysis for reducing the dimensionality of data sets. We'll see this next.

PRINCIPAL COMPONENT ANALYSIS (PCA)

Example: Campaign contributions to US senators

1. Get the data



2. Clean it up

Contributor categories

Senators

Senator	Agribusiness	Communic/Electronics	Construction	Defense	Education
{Alexander, Lamar, R}	223 171	208 781	287 750	75 400	145 700
{Ayotte, Kelly, R}	98 000	273 969	124 938	67 150	30 600
{Baldwin, Tammy, D}	85 509	440 506	74 609	4450	287 934
{Barrasso, John, R}	166 150	262 850	186 933	86 500	37 050
{Baucus, Max, D}	627 036	517 595	302 437	101 600	50 538
{Begich, Mark, D}	16 781	310 583	95 819	30 000	40 175
{Bennet, Michael, D}	214 341	826 273	161 425	53 150	274 355
{Blumenthal, Richard, D}	30 050	309 600	76 400	21 750	155 400
{Blunt, Roy, R}	805 677	728 737	481 335	155 350	70 150
{Boozman, John, R}	172 829	70 296	228 786	3000	26 700
{Boxer, Barbara, D}	207 350	1 634 559	388 955	24 150	338 404
{Brown, Sherrod, D}	180 532	508 863	239 702	102 450	461 577
{Burr, Richard, R}	603 168	264 838	277 904	156 947	55 365
{Cantwell, Maria, D}	90 883	490 380	89 673	40 975	94 278
{Cardin, Benjamin, D}	50 750	350 837	157 822	80 550	121 665
{Carper, Thomas, D}	91 885	276 990	97 810	60 400	25 466
{Casey, Robert, D}	251 250	537 416	244 976	125 075	257 211
{Chambliss, Saxby, R}	1 758 731	380 684	483 351	286 150	44 368
{Coburn, Thomas, R}	100 625	115 630	57 800	36 650	5400
{Cochran, Thad, R}	373 294	98 500	119 250	167 050	21 350
{Collins, Susan, R}	207 838	459 982	269 546	291 801	56 350
{Coons, Chris, D}	32 550	200 450	34 150	19 500	55 077
{Corker, Bob, R}	316 050	436 582	600 050	70 850	81 250
{Cornyn, John, R}	686 106	486 367	559 927	243 076	52 725
{Crapo, Michael, R}	288 499	145 962	132 420	39 000	3930
{Cruz, Ted, R}	247 200	234 178	340 903	38 700	69 500
{Donnelly, Joe, D}	194 546	144 158	181 548	120 283	140 709
{Durbin, Richard, D}	249 014	486 625	191 600	127 350	149 375
{Enzi, Michael, R}	90 444	73 050	50 500	25 500	13 000
{Feinstein, Dianne, D}	522 217	591 836	184 400	198 500	89 386
{Fischer, Deb, R}	291 502	97 400	131 028	10 000	12 500
{Flake, Jeff, R}	399 577	212 700	317 850	26 900	79 000

3. "Standardize" the data
   a) Shift the numbers in each column so the mean $\sum_{i=1}^{97} x_j^{(i)} = 0$.
   b) Scale each column so the variance $\sigma_j^2 = \sum_{i=1}^{97} (x_j^{(i)})^2 = 1$.

Note: The scaling may be omitted if the different attributes are known to lie on the same scale.
Here it has the effect of making all categories equally significant, even though some give more than others!

```
{sectors, Plus @@ puredata} // Transpose // Sort[#, #1[[2]] > #2[[2]] &] & // MatrixForm
```

Finance/Insur/RealEst	168 151 725
Unknown	154 842 094
Lawyers & Lobbyists	112 167 979
Other	103 253 137
Ideology/Single-issue	100 385 417
Misc Business	92 373 646
Health	71 844 560
Communic/Electronics	45 991 954
Energy/Nat Resource	40 100 000
Agribusiness	30 637 498
Construction	28 244 716
Transportation	23 257 205
Labor	20 546 123
Education	12 200 463
Defense	10 484 140
Party Cmte	1 690 362
Joint Candidate Cmtes	514 864

```
{height, width} = Dimensions[puredata]
puredatastandardized = puredata // N;
```

$$\texttt{means} = \frac{1}{\texttt{height}} \sum_{j=1}^{\texttt{height}} \texttt{puredatastandardized[[j]]};$$

```
For[j = 1, j ≤ height, j ++,
  puredatastandardized[[j]] -= means;
 ];
```

$$\texttt{variances} = \frac{1}{\texttt{height}} \sum_{j=1}^{\texttt{height}} \texttt{puredatastandardized[[j]]}^2;$$

```
For[j = 1, j ≤ height, j ++,
  puredatastandardized[[j]] /= √variances ;
 ];
```

```
{97, 17}
```

4. Take the SVD:

```
{U, M, V} = SingularValueDecomposition[puredatastandardized];
Plus @@ Plus @@ Abs /@ (U.M.Transpose[V] - puredatastandardized)
{Dimensions[U], Dimensions[M], Dimensions[V]}
```

5.96535×10^{-12} ← $\mathcal{U} M V^T$ = data matrix ✓

dim $\mathcal{U}$ dim M   dim V

```
{{97, 97}, {97, 17}, {17, 17}}
```

The singular values, sorted from largest on down:

```
M // MatrixForm
```

33.265	0.	0.	0.	0.	0.	0.	0.
0.	12.4751	0.	0.	0.	0.	0.	0.
0.	0.	10.0409	0.	0.	0.	0.	0.
0.	0.	0.	9.3013	0.	0.	0.	0.
0.	0.	0.	0.	7.3885	0.	0.	0.
0.	0.	0.	0.	0.	5.60893	0.	0.
0.	0.	0.	0.	0.	0.	4.79764	0.
0.	0.	0.	0.	0.	0.	0.	4.4487
0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.

Note: The columns of $\mathcal{U}$ are the left singular vectors, forming a basis for columnspace R(data).
The columns of V are the right singular vectors, forming a basis for rowspace R(data^T).

5. Now what?
   a) We could approximate the data matrix with a lower-rank matrix, e.g., keeping

$$94\left(\begin{array}{c}\text{first two}\\ \text{columns of } \mathcal{U}\end{array}\right)_{2} \begin{pmatrix} 33.265 & 0 \\ 0 & 12.4751 \end{pmatrix} {}_{2}\!\left(\begin{array}{c}\text{— first two columns —}\\ \text{— of } V\text{, transposed —}\end{array}\right)^{17}$$

gives a rank-2 matrix.

Each data point (row) now lies in a 2D subspace of $\mathbb{R}^{17}$.

```
"These are the two most important directions:";
principaldirections = V[All, {1, 2}];

principaldirectionslabeled = Append[principaldirections // Transpose, sectors] // Transpose;
"Now let's sort these by the second component (we'll see why in a moment).";
Sort[principaldirectionslabeled,
  #1[2] < #2[2] &
 ] // MatrixForm
```

$\vec{v}_1$	$\vec{v}_2$	
-0.00387268	-0.707777	Labor
-0.238924	-0.329765	Education
-0.261419	-0.268357	Lawyers & Lobbyists
-0.260683	-0.216817	Ideology/Single-issue
-0.266349	-0.211043	Communic/Electronics
-0.282418	-0.00418024	Other
-0.288545	0.00920218	Unknown
-0.286914	0.0197931	Finance/Insur/RealEst
-0.291324	0.0353523	Misc Business
-0.273799	0.0478608	Health
-0.2148	0.05117	Defense
-0.0122403	0.0995367	Joint Candidate Cmtes
-0.285753	0.110642	Construction
-0.273507	0.174331	Transportation
-0.0855998	0.186206	Party Cmte
-0.241713	0.244521	Agribusiness
-0.248974	0.270343	Energy/Nat Resource

$$\min_{B:\,\mathrm{rank}(B)=k} \|A-B\|_F^2$$

b) Or, if you care only about the data points' positions in the 2D subspace — and not about the subspace's position in $\mathbb{R}^{17}$ — then we can use the first two columns of V to project down to $\mathbb{R}^2$:

```
puredatastandardized // Dimensions
principaldirections // Dimensions
dataprojectedmanually = puredatastandardized.principaldirections;
```

{97, 17}

{17, 2}

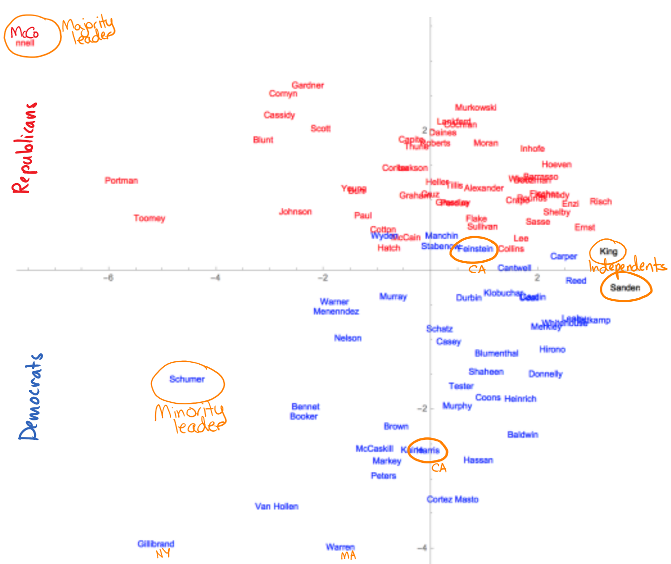
Since $\text{data} = UMV^T = \sum_{i=1}^{17} \sigma_i \vec{u}_i \vec{v}_i^T$, $\text{data} * \begin{pmatrix} \text{first two} \\ \text{columns of } V \end{pmatrix} = \sum_{i=1}^{2} \sigma_i \vec{u}_i \vec{e}_i^T$

$$\begin{pmatrix} \text{first two} \\ \text{columns of } V \end{pmatrix} = \vec{v}_1\vec{e}_1^T + \vec{v}_2\vec{e}_2^T$$

$$= \begin{pmatrix} | & | \\ \sigma_1\vec{u}_1 & \sigma_2\vec{u}_2 \\ | & | \end{pmatrix}$$

Of course, we could also just have picked out the first two left singular vectors directly.

Let's plot it!



Interpretation: 2^{nd} principal component $\to$ political party (maybe liberal vs. conservative?)

1^{st} principal component $\to$???

Since all entries of the first principal direction are <0, the x-direction above says who raised more money (with the most to the left). The y-direction (second principal direction) says how that money is distributed away from average. Negative means more Labor. Positive means more Energy/Nat. Resources.

Notice that there is a slight funnel effect; candidates who raise more money (on the left) also get it from a more biased distribution of donors.

Find correlations, clusters, voids, study more principal components, etc.

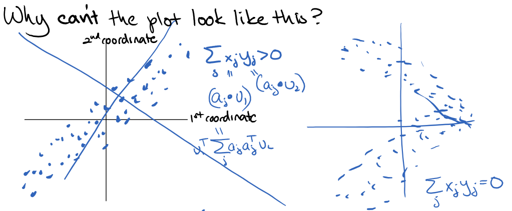
Note: Matlab/Mathematica/R all have built-in PCA functions:

$|u \cdot a_j|^2$

```
principalcomponents = PrincipalComponents[puredatastandardized];

"To keep just the first two principal components (first two columns), use:";
principalcomponents[[All, {1, 2}]]
```

Question: Why can't the plot look like this?



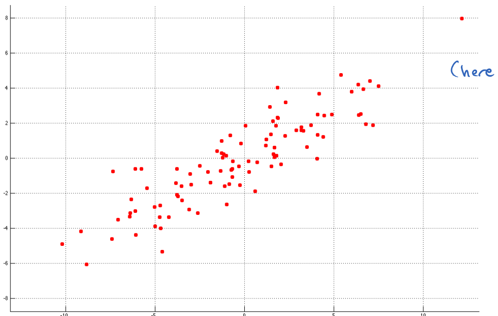
Answer: We are plotting $(x_1, y_1), \ldots, (x_m, y_m)$, where $(x_1, \ldots, x_m)$ is the first left singular vector and $(y_1, \ldots, y_m)$ is the second left singular vector. These vectors are orthogonal: $\sum_{j=1}^{m} x_j y_j = 0$.

Informal motivation for PCA:

The first k principal directions are the k factors that "explain the data best."

Formal mathematical motivation:

Given data points $\{(x_1^{(1)}, \ldots, x_k^{(1)}), (x_1^{(2)}, \ldots, x_k^{(2)}), \ldots, (x_1^{(m)}, \ldots, x_k^{(m)})\}$,



(here $k=2$)

The "method of least squares" lets us predict the last component/attribute in terms of the first k-1 components, minimizing the squared error

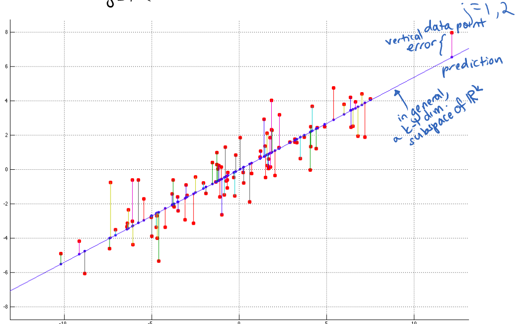
$$\left\| \begin{pmatrix} 1 & x_1^{(1)} & \cdots & x_{k-1}^{(1)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(m)} & \cdots & x_{k-1}^{(m)} \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{k-1} \end{pmatrix} - \begin{pmatrix} x_k^{(1)} \\ \vdots \\ x_k^{(m)} \end{pmatrix} \right\|^2$$

$$= \sum_{j=1}^{m} \left(\alpha_0 + \alpha_1 x_1^{(j)} + \cdots + \alpha_{k-1} x_{k-1}^{(j)} - x_k^{(j)} \right)^2$$

$$\underset{m \times n}{A = \sum_j \sigma_j v_j u_j^T}$$

$$\downarrow$$

$$\underset{m \times 2}{\sum_{j=1,2} \sigma_j v_j e_j^T} = A \begin{pmatrix} | & | \\ u_1 & u_L \\ | & | \end{pmatrix}$$



This is most useful if the first k-1 attributes are exact, and the last one imprecise or noisy.

Least-squares regression is not useful for analyzing the campaign finance data, because all the components are imprecise/noisy (equally noisy after scaling the columns), and we aren't just trying to predict one of them. Basically, the data are less structured.

In contrast, principal component analysis (PCA) finds the subspace (with a dimension you can choose) that minimizes the total squared distances from the subspace.