Proximal methods:

consider

$$f(x) = g(x) + h(x)$$

convex, diff, domain is R^n

awkward: convex but not necessarily diff and domain might be smaller

gradient descent builds a second order model at x_n, and minimizes that.

$w(x) \leftarrow$ conv, diff

$$x_{n+1} = \operatorname*{argmin}_z \left[w(x_n) + (\nabla w)^T (z - x_n) + \frac{1}{2t} \| z - x_n \|^2 \right]$$

$$= x_n - t \nabla w$$

Strategy:

approx g, ignore h

$$x_{n+1} = \underset{z}{\operatorname{argmin}} \left[g(x_n) + (\nabla g)^T (z - x_n) + \frac{1}{2t} \|z - x_n\|_2^2 + h(z) \right]$$

$$= \underset{z}{\operatorname{argmin}} \left[\frac{1}{2t} \| z - (x_n - t\nabla g) \|_2^2 + h(z) \right]$$

[you should check this – expand terms!]

so z – should be close to g.d. pred

– make $h(z)$ smaller.

Define:

$$\mathrm{prox}_{h,t}(x) = \underset{z}{\mathrm{argmin}} \frac{1}{2t}\| x - z\|_2^2 + h(z)$$

proximal gradient Descent:

$$x_{n+1} = \mathrm{prox}_{h,t_n}\left(x_n - t_n \nabla g(x_n)\right)$$

at its most useful when h is such that prox_{h,t_n} is easy

(Quite common — examples follow)

ISTA :

the lasso problem.

$$\underset{\beta}{\text{minimize}} \quad \underbrace{\tfrac{1}{2}\|y - X\beta\|_2^2}_{g\,-\,\text{easy}} + \underbrace{\lambda\|\beta\|_1}_{h\,-\,\text{awkward.}}$$

we solved this by splitting and ADMM.

$$\text{prox}_t(\beta) = \underset{z}{\text{argmin}} \ \frac{1}{2t}\|\beta - z\|_2^2 + \lambda\|z\|_1$$

(which should look familiar).

notice $\quad \text{prox}_t(\beta) = \underset{z}{\text{argmin}} \ \frac{1}{2t}\sum_u (\beta_u - z_u)^2 + \lambda\sum_u \text{abs}(z_u)$

so it splits over the components of z, β.

consider one term

$$\underset{z_u}{\text{argmin}} \ \frac{1}{2t}(\beta_u - z_u)^2 + \lambda\, \text{abs}(z_u)$$

Soln occurs when

$0 \ni \partial_{z_u}$

so $0 \ni -\frac{1}{t}(\beta_u - z_u) + \lambda \partial_{z_u} \mathrm{abs}(z_u).$

<u>cases</u>:

<u>$z_u > 0$</u>

then $-\frac{1}{t}(\beta_u - z_u) + \lambda = 0$

so $z_u = \beta_u - t\lambda.$

and so $\cancel{z_u > \beta_u}$

$\beta_u > t\lambda$

<u>$z_u < 0$</u>:

$z_u = \beta_u + t\lambda$

<u>$z_u = 0$</u>:

$0 \ni -\frac{1}{t}(\beta_u - \underset{0}{\cancel{z_u}}) + \lambda[-1; 1]$

a whole interval!

so $0 \ni \beta_u \; \lambda t [-1; 1]$

so $\lambda \leq \beta_u \leq \lambda t$

This should all look fairly familiar

Notation

$$S_{\lambda t}(\beta) \longleftarrow \text{Shrinkage operator!}$$

$$[S_{\lambda t}(\beta)]_u = \begin{cases} \beta_u - \lambda t, & \beta_u > \lambda t \\ 0, & -\lambda t \leq \beta_u \leq \lambda t \\ \beta_u + \lambda t, & \beta_u < -\lambda t. \end{cases}$$

So proximal gradient descent is

$$\beta_{n+1} = S_{\lambda t}\left[\beta_n + t X^T (y - X\beta)\right]$$

(iterative soft-thresholding algorithm — ISTA)

Convergence:

assume: g convex, diff, $\operatorname{dom}(g) = \mathbb{R}^n$,
∇g is Lipschitz, const L.

h convex, prox can be evaluated

Proximal gradient descent with fixed step size $t \leq \frac{1}{L}$ satisfies

$$f(x_n) - f^* \leq \frac{\| x_0 - x^* \|_2^2}{2tn}$$

notice: as before, this bounds function values, rather than arguments

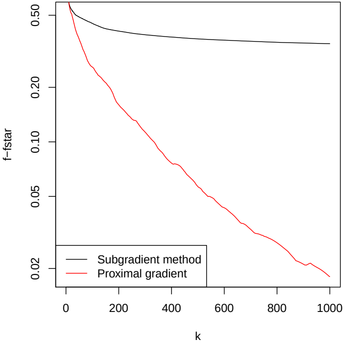
matches gradient descent...

Recall $\nabla g(\beta) = -X^T(y - X\beta)$, hence proximal gradient update is:

$$\beta^+ = S_{\lambda t}\big(\beta + tX^T(y - X\beta)\big)$$

Often called the iterative soft-thresholding algorithm (ISTA).[1] Very simple algorithm

Example of proximal gradient (ISTA) vs. subgradient method convergence curves



[1] Beck and Teboulle (2008), "A fast iterative shrinkage-thresholding algorithm for linear inverse problems"

Matrix completion:

we have $Y \in R^{(m \times n)}$

But observe only some entries.

we want to fill in missing entries

- general case isn't possible
- but assume Y is known to have small nuclear norm

(Ky-Fan norm)

$$\|B\|_{KF} = \sum_i \sigma_i(B)$$

where σ_i are singular values of B AND $\sigma_i \geq 0$

General idea: this norm favors matrices with low rank

Why (applications).

- ~~assume~~ This is a generalization of the case where
$$Y = AB$$ and entries are missing

(in this case, rank of Y is known)

- Classic example of $Y = AB$
  - affine structure from motion with missing observations

General case:

- recommender systems
- smoothing word counts

<u>Reminder</u>:

imagine we <u>know</u> y_{ij} for $i,j \in K$

then

$$\min_{a_{il},\, b_{lj}} \sum_{ij \in K} \left[Y_{ij} - \sum_{l} a_{il} b_{lj} \right]^2$$

(least squares soln for A, B, changing <u>only</u> for known entries of Y_{ij})

is <u>straightforward</u>

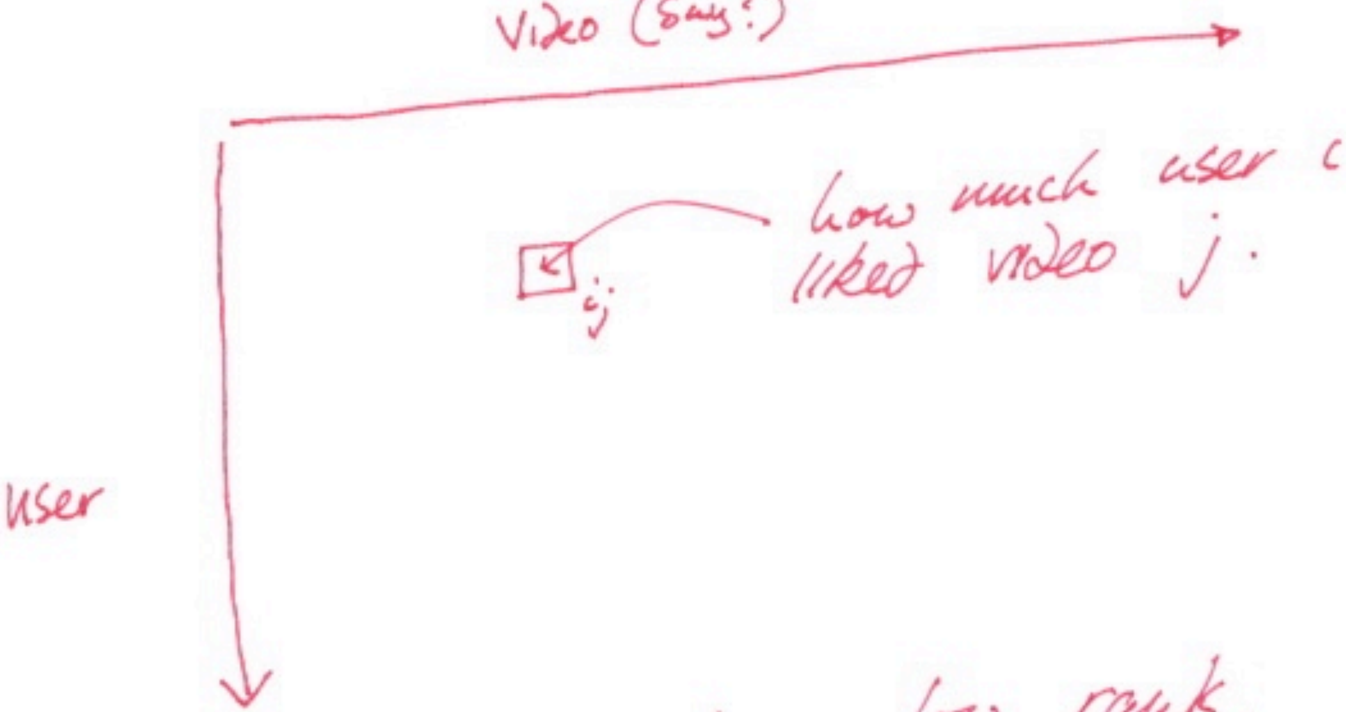
<u>iterate</u>

- fix A, solve for B (linear sys)
- fix B, solve for A "

works-ish, but has annoying features.

- can diverge (uncommon)
- can be slow (common)
- you have to <u>know</u> rank.

Recommender systems



This matrix should have low rank.

- many pairs of users will have about the same tastes
  → many pairs of rows will be similar
- two "similar" [horror; history; etc] videos will be liked by about the same set of people
  → many pairs of cols will be similar.

~~and~~

AND:

- many cells were not observed
- knowing values of those cells is worthwhile.

another example

- PCA with data deleted "at random"

- Have
$$\begin{array}{cc} & \text{feat \# } \rightarrow \\ X = \begin{array}{c} \\ \\ \text{obs \#} \downarrow \end{array} & \begin{bmatrix} & & \cdot & \\ & \cdot & & \cdot \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

- assume $1^T X = 0$
  (i.e. data is zero mean)
  — it is easy to get means w
    missing observations.
    (average the ones you have.)
  - But Covar presents challenges.
    (possible, but nuisance).

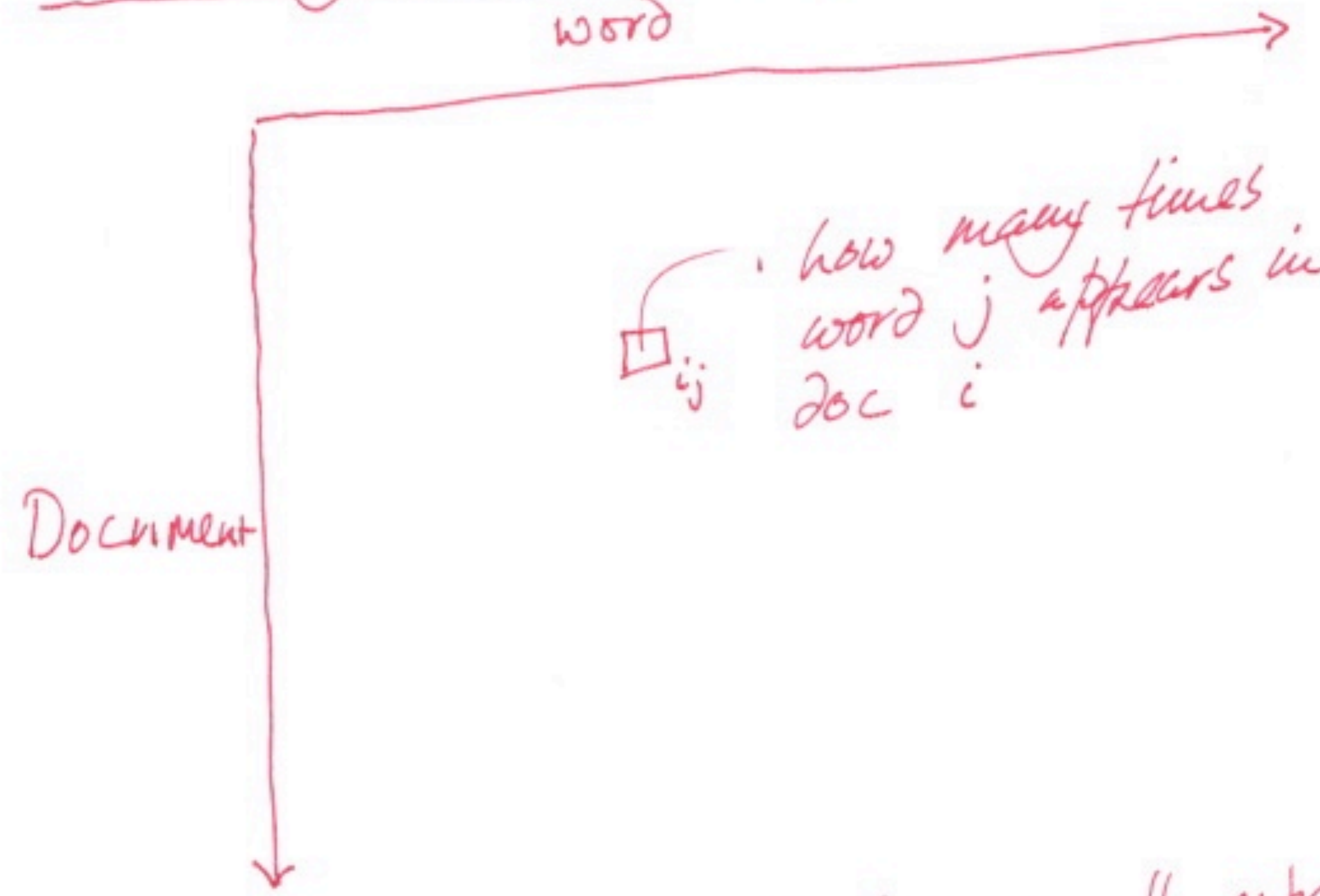
Alternative :

- PCA for X would yield

$$
X = \begin{array}{cc} & \text{weights} \rightarrow \\ \text{obs} \downarrow & \begin{bmatrix} \; \\ \; \\ \; \end{bmatrix} \end{array} \begin{array}{cc} \text{feats} \rightarrow & \\ \begin{bmatrix} \quad\quad\quad\quad \end{bmatrix} & \downarrow \text{P.C.s} \end{array}
$$

- we know for most datasets, rank is small but we don't know what it is.
  - Possibility :
    - choose rank.
    - Factor X w/ SVD (missing data? as above)
  - Alternative:
    - Complete X
    - Then obtain PCA w/ SVD.

Smoothing word counts:



- <u>In principle</u> we could know all entries of this Document-Word matrix.
- <u>But</u> 0 - counts are somewhat unreliable
- many words have similar meaning

  "car", "motorcar", "automobile"

- if "car" appears, there should likely be a <u>small</u> count for "motorcar" etc.
- this means that the matrix should have low rank, too.

- We are now interested in

$$\min_{B} \begin{bmatrix} \text{difference between} \\ \text{observed } Y \text{ and} \\ \text{CSP } B \end{bmatrix}^2 + \lambda \|B\|_{KF}.$$

- Write

$$P_\Omega(B) = \begin{cases} B_{ij} & \text{if } (i,j) \text{ was observed} \\ 0 & \text{otherwise} \end{cases}$$

Then:

$$\min_{B} \quad \frac{1}{2}\|P_\Omega(Y) - P_\Omega(B)\|_F^2 + \lambda \|B\|_{KF}$$

(notice $P_\Omega(B)$ might return every value, if all of Y is observed).

$$\|M\|_F^2 = \sum_{ij} m_{ij}^2 = Tr(M^T M) \leftarrow \text{Frobenius norm.}$$

$$f(B) = \underbrace{\frac{1}{2}\|P_\Omega(Y) - P_\Omega(B)\|_F^2}_{g(B)} + \underbrace{\lambda \|B\|_{KF}}_{h(B)}$$

for proximal, need: gradient, prox

gradient $\quad \nabla_B(g) = -(P_\Omega(Y) - P_\Omega(B))$

Q: what happened to $\frac{\partial P_\Omega}{\partial B}$?

$$\mathrm{prox}_t(B_n) = \underset{Z}{\mathrm{argmin}} \left[\frac{1}{2t} \| B_n - Z \|_F^2 + \lambda \| Z \|_{KF} \right]$$

Claim:

$$\mathrm{prox}_t(B) = S_{\lambda t}(B)$$

$\uparrow$ matrix soft thresholding

$$S_{\lambda t}(B) = U \Sigma_{\lambda t} V^T$$

where $B = U \Sigma V^T$ (SVD)

and. $\Sigma_{\lambda t}$ is diagonal,

$$\left[\Sigma_{\lambda t} \right]_{uu} = \max\left(\Sigma_{uu} - \lambda t, 0 \right)$$

(proof in Tibshirani notes on webpage).

so

$$B_{n+1} = S_{\lambda t}\left(B_n + t\left(P_\Omega(Y) - P_\Omega(B)\right)\right).$$

can choose $t = 1$, get

$$B_{n+1} = S_{\lambda t}\left(P_\Omega(Y) + P_\Omega^{\perp}(B)\right)$$

(where $P_\Omega^{\perp}(B)$ projects onto <u>unobserved set</u>)

This is <u>soft-impute</u> (ref in notes).

<u>Now</u> assume we want

$$\min_{x \in C} \; g(x)$$

where $g(x)$ is diff, convex and defined on $\mathbb{R}^n$

C is convex

<u>write</u>:

$$\min_{x \in \mathbb{R}^n} \; g(x) + I_C(x)$$

indicator

$$I_C(x) = \begin{cases} 0 & x \in C \\ \infty & \text{otherwise} \end{cases}$$

<u>now</u>:

$$\text{prox}_t(x) = \underset{z}{\text{argmin}} \; \frac{1}{2t}\|(x-z)\|_2^2 + I_C(x)$$

$$= \underset{z \in C}{\text{argmin}} \; \|x-z\|_2^2$$

so that

$$\mathrm{prox}_t(x) = P_c(x)$$

- project x onto C
- closest point in C to x.

so

$$x_{n+1} = P_c\left(x_n - t\nabla g(x_n)\right)$$

projected gradient descent

useful if you can compute P_c

- box, sphere, etc.