L16 -- Lasso for Regularized Regression
[Jeff Phillips - Utah - Data Mining]

Input: n x d matrix P = [p_1 p_2 ... p_n]^T
  "n points in d dimensions"

P_i = [p_{i,1} p_{i,2} ... p_{i,d}]
** assume that for all j   sum_{i=1}^n p_{i,j} = 0
P_j = [p_{1,j} p_{2,j} ... p_{n,j}]^T
   + a column with all n points jth coordinate

and:
  Y = [y_1 y_2 ... y_n]^T   y_j scalar
think of f(P_i) = y_i
** assume that sum_{i=1}^n y_i = 0

Let A = [a_1 a_2 ... a_d]^T

Goal: Find g(X) = a_0 + sum_{j=1}^d x_j a_j
  where X = [x_1 x_2 ... x_d]
  and where Loss(g(P)-Y) is minimized
"best linear fit" (can add P_{i'} = P_i^2 or P_i*P_{i'} for non-linear fit)

ignore a_0 by adding dimension where p_{i,0} = 1 for all i.

 Loss Functions

If Loss(g(P)-Y) is ||g(P)-Y||_2 = ||g(P) - Y||_2^2   "least squares"
   A = (P^T P)^{-1} P^T Y
   g(P) = P A = P (P^T P)^{-1} P^T Y

If Loss(g(P)-Y) = ||g(P) - Y||_2 + s||A||_2   "ridge regression"
  (or Loss(g(P)-Y) = ||g(P) - Y||_2   s.t. ||A||_2 < t)
   A = (P^T P + sI)^-1 P^T Y
   g(P) = P A = P (P^T P + sI)^{-1} P^T Y

If Loss(g(P)-Y) = ||g(P) - Y||_2 + s||A||_1   "Lasso" "basis pursuit"
  (or Loss(g(P)-Y) = ||g(P) - Y||_2   s.t. ||A||_1 < t)

 How to solve coming soon...

Note: ridge + Lasso trade off decreased variance for increased (non-zero bias)
      ridge + Lasso are both convex in A (one minimum), so should be easy to solve.

Lasso has "magical" property than many a_j=0.

```
[Draw picture of constraint variant with L_1 or L_2 ball  -- See ESL book]
Want L_0 ball, but then not convex (multiple minimum)

---------------------
Could use "Orthogonal Matching Pursuit" approach
 Init:  set a_j = 0 for all j in [d]
 1: Find j with max_j |<P_j,Y>|      <--- coordinate j
 2: Set a_j = min_a Loss(P_j a - Y)
 3: Calculate residual in P_j a - Y in place of Y (and repeat)

"Forward Subset Selection"
   (also "Backwards Subset Selection": remove P_i with smallest effect)
----------------------

How do we solve Lasso?
  **use constraint variant and start with t = infty
  Set a_j=0 for all j in [d]
  Set t = sum_{j=1}^d |a_j|
  Set r(t) = Y - sum_{j=1}^d P_j a_j(t)

0:  Find j_1 = argmax_j |<P_j,r>|
    Set a_{j_1}(t) = a_j*t

1:  Find t_2 s.t. some j_2 != j_1  has |<P_{j_1},r(t)>| = |<P_{j_2},r(t)>|
    Find correlations (via derivatives) and reset
        a_{j_1}(t) = a_{j_1}(t_2) + (t-t_2)*b_1
        a_{j_2}(t) = (t-t_2)*b_2
        s.t. |b_1| + |b_2| = 1
** cool fact: as t increases, optimal choice of a_j is linear in t with slopes
b_1,b_2...

in general:
1:  Find t_k s.t. some j_t != j_l \in [j_1...j_{t-1}] has |<P_{j_l},r(t)>| = |
<P_{j_k},r(t)>|
    Set a_{j_l}(t) = a_{j_l}(t_k) + (t-t_k) b_l
    s.t. sum_{l=1}^k |b_l| = 1

    "intuitively:"
        Let ~b_l = (d/dt) |<P_{j_l},r(t)>|
         B = sum_{l=1}^k |~b_l|
         b_l = ~b_l/B        <-- normalize

** Sometimes may have slopes b_l as negative, and may snap a_{j_l} = 0
   LAR (least angle regression) does not re-snap a_{j_l} = 0
   This occurs since we initially overfit a_{j_l} and need to adjust,
sometimes remove
```

Cool thing is that we have solved for every value of t (hence every value of s)
  --> can cross-validate to find best value of t
      (leave some data out, and test accuracy on those values)

Low Rank + Sparse

SVD: P = U S V^T = [U_k U_k'] [S_k 0 ; 0 S_k'] [V_k^T ; V_k'^T]
      P_k = U_k S_k V_k^T
            low rank (rank = k)

If P = P_k + N_0 where N_0 is Gaussian Noise, then this is "best" reconstruction

What if P = L + S
       where S is sparse noise (small number << n^2) items are arbitrarily large
       and L is rank k

Solve minimum ||L||_* + ||S||_1 where restrict P = L + S

||M||_* = trace(sqrt(M*M)) = sum (singular values M)

What if P = L_k + S_0 + N_0
       where L_k is rank k
         and S_0 is sparse noise
         and N_0 is Gaussian noise

Solve minimum ||L||_* + ||S||_1 such that ||P - L - S||_F < delta

----

both are convex problem, and can solved using specially designed solvers
  iteratively find PCA, filter out supposed sparse results, and repeat.
  uses time equivalent to about 16 SVD computations.