# CPSC 340:
# Machine Learning and Data Mining

Sparse Matrix Factorization

# Admin

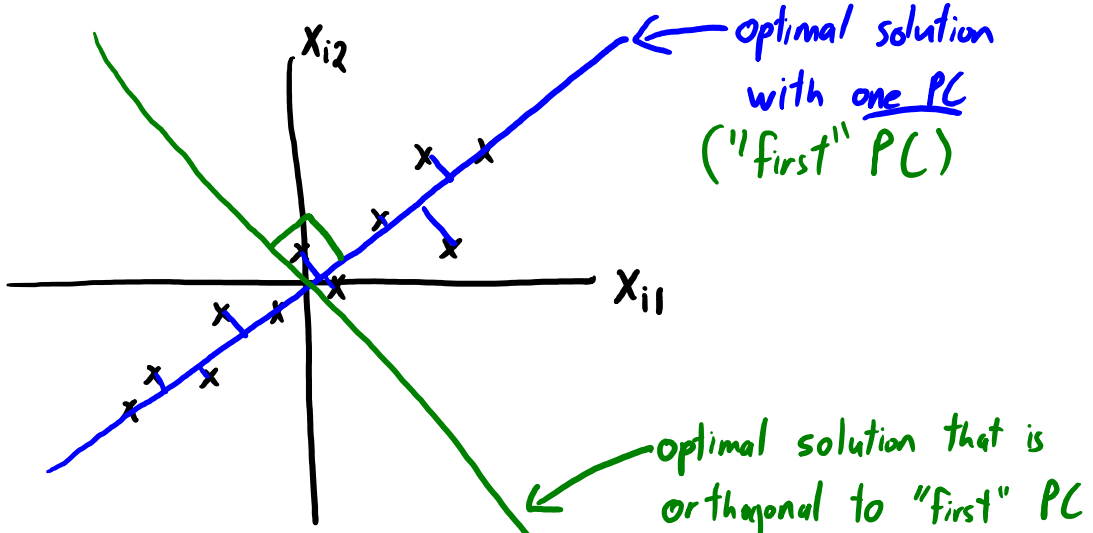
- Assignment 4:
  - Due Sunday night.
- Assignment 5:
  - Out today (hopefully).
- Final exam
  - Tuesday April 25th at 8:30am

# Last Time: PCA with Orthogonal/Sequential Basis

- When k = 1, PCA has a scaling problem.
- When k > 1, have scaling, orthogonality, rotation, label switching.
  - Standard fix: use normalized orthogonal rows $w_c$ of 'W'.

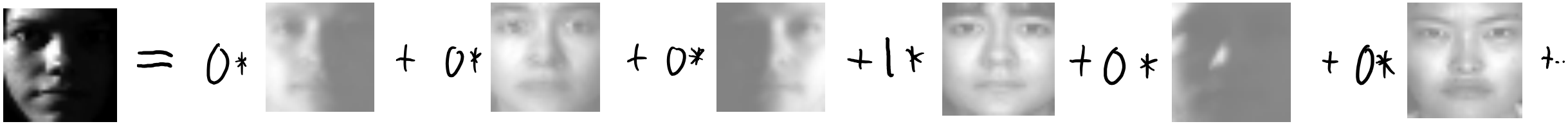
$$w_c^T w_c = 1 \quad \text{and} \quad w_c^T w_{c'} = 0 \quad \text{for} \quad c \neq c'$$

  - And fit the rows in order:
    - First row "explains the most variance" or "reduces error the most".



# Last Time: Learning a Basis for Faces

- We discussed three ways to learn a basis $z_i$ for faces:
  - K-means (vector quantization).
    - Replace face by the average face in a cluster.
    - Can't distinguish between people in the same cluster (only 'k' possible faces).

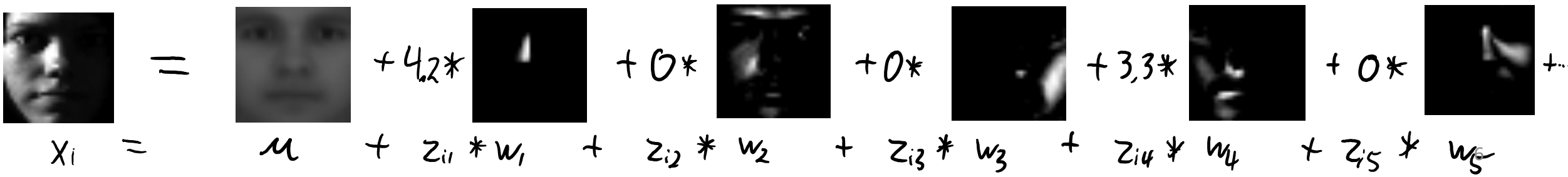


# Last Time: Learning a Basis for Faces

- We discussed three ways to learn a basis $z_i$ for faces:
  - K-means (vector quantization).
  - PCA (orthogonal basis).
    - Global average plus linear combination of "eigenfaces".
    - Can generate an infinite number of faces when changing the $z_i$.
    - But "eigenfaces" are not intuitive ingredients for faces.

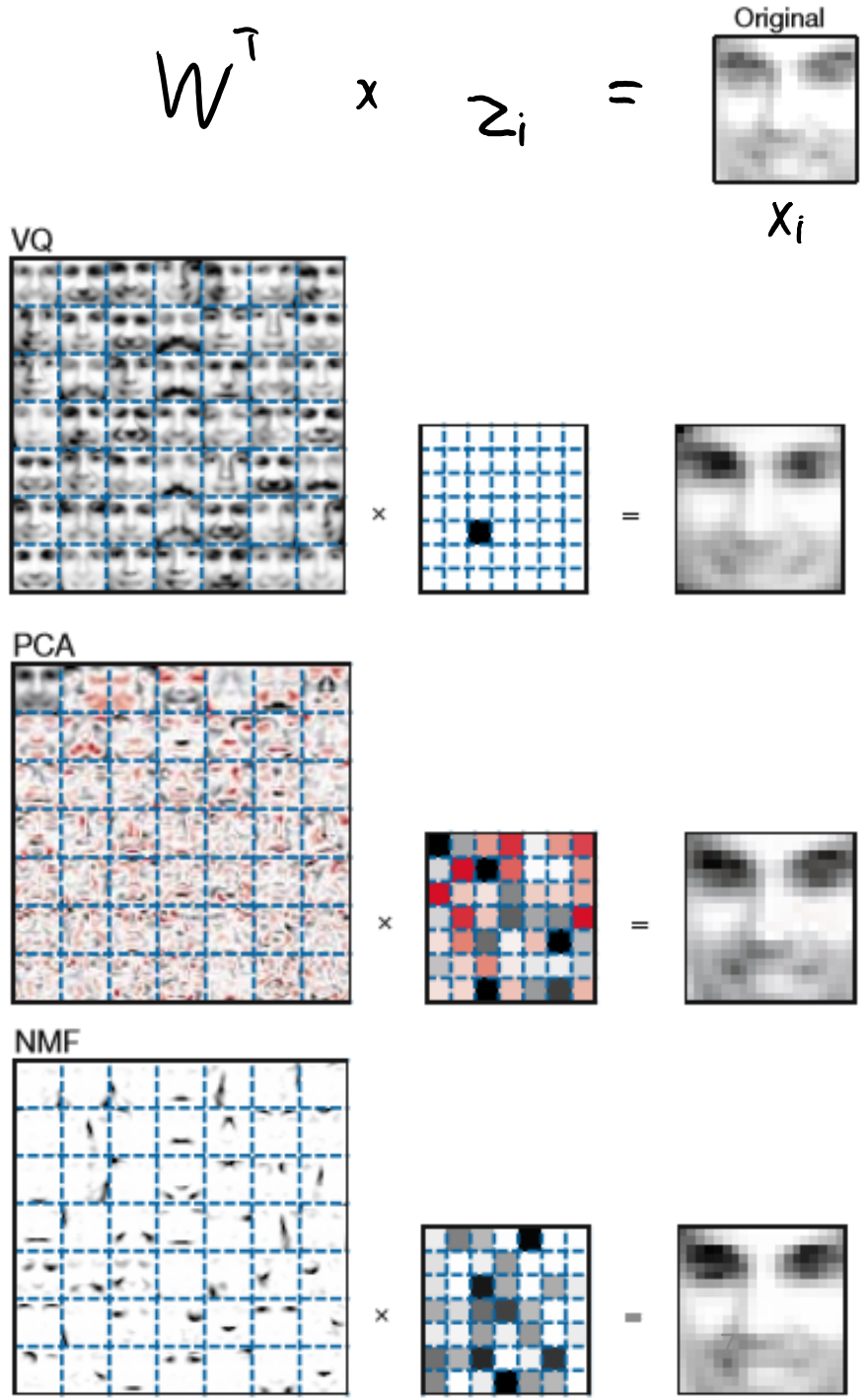
# Learning a Basis for Faces

- We discussed three ways to learn a basis $z_i$ for faces:
  - K-means (vector quantization).
  - PCA (orthogonal basis).
  - NMF (non-negative matrix factorization):
    - Instead of requiring orthogonality requires non-negativity.
      - No "ordering" among parts.
      - The $z_i$ are sparse so each face only uses a subset of the sparse "parts".



# Representing Faces

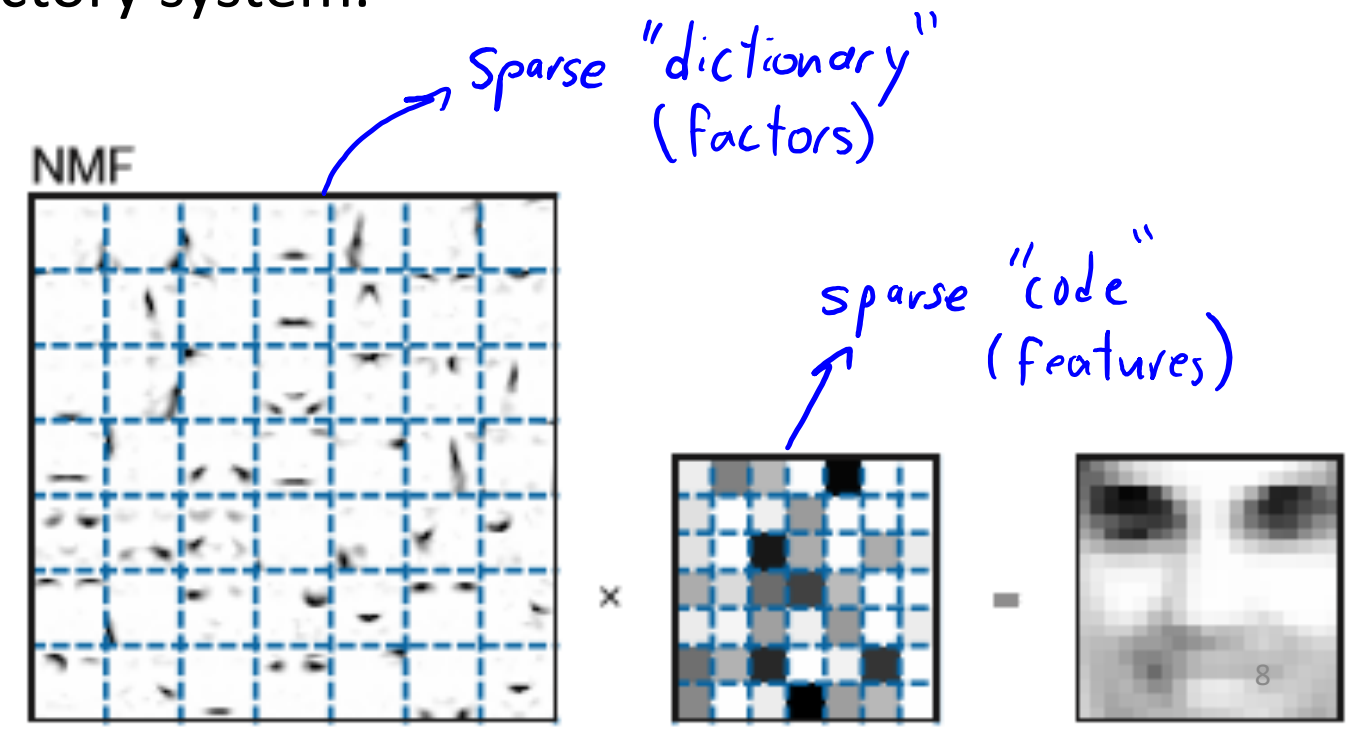
- K-means:
  - ‘Grandmother cell’: one neuron = one face.
  - Almost certainly not true: too few neurons.
- PCA:
  - “Distributed representation”.
    - Coded by pattern of group of neurons.
    - Can represent more concepts.
  - But PCA uses positive/negative cancelling parts.
- Non-negative matrix factorization (NMF):
  - Latent-factor where W and Z are non-negative.
  - Example of “sparse coding”:
    - Coded by small number of neurons in group.
  - NMF makes object out of ‘parts’.




http://www.columbia.edu/~jwp2128/Teaching/W4721/papers/nmf_nature.pdf

# Why sparse coding?

- 'Parts' are intuitive, and brains seem to use sparse representation.
- Energy efficiency if using sparse code.
- Increase number of concepts you can memorize?
  - Some evidence in fruit fly olfactory system.




http://www.columbia.edu/~jwp2128/Teaching/W4721/papers/nmf_nature.pdf

# Warm-up to NMF: Non-Negative Least Squares

- Consider our usual least squares problem:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2$$

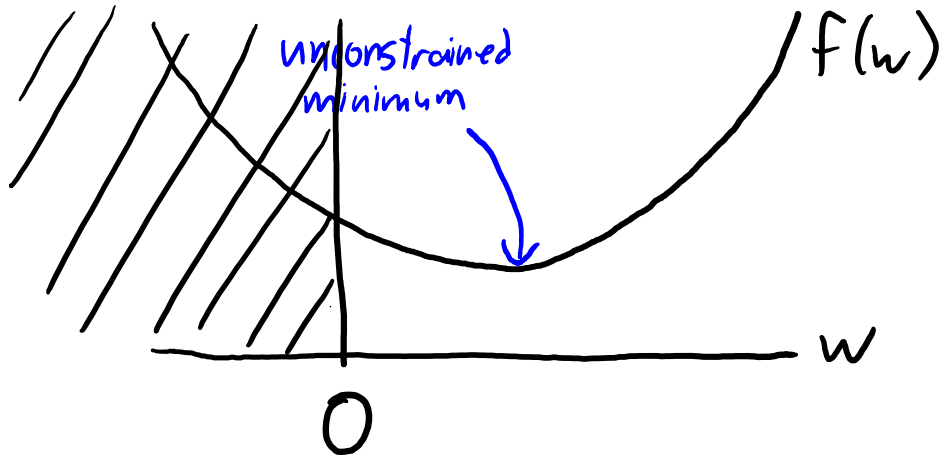
- But assume $y_i$ and elements of $x_i$ are non-negative:
  - Could be sizes ('height', 'milk', 'km') or counts ('vicodin', 'likes', 'retweets').
- Assume we want elements of 'w' to be non-negative, too:
  - No physical interpretation to negative weights.
  - If $x_{ij}$ is amount of product you produce, what does $w_j < 0$ mean?
- Non-negativity tends to generate sparse solutions.

# Sparsity and Non-Negative Least Squares

- Consider 1D non-negative least squares objective:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w\,x_i - y_i)^2 \quad \text{with} \quad w > 0$$

- Plotting the (constrained) objective function:



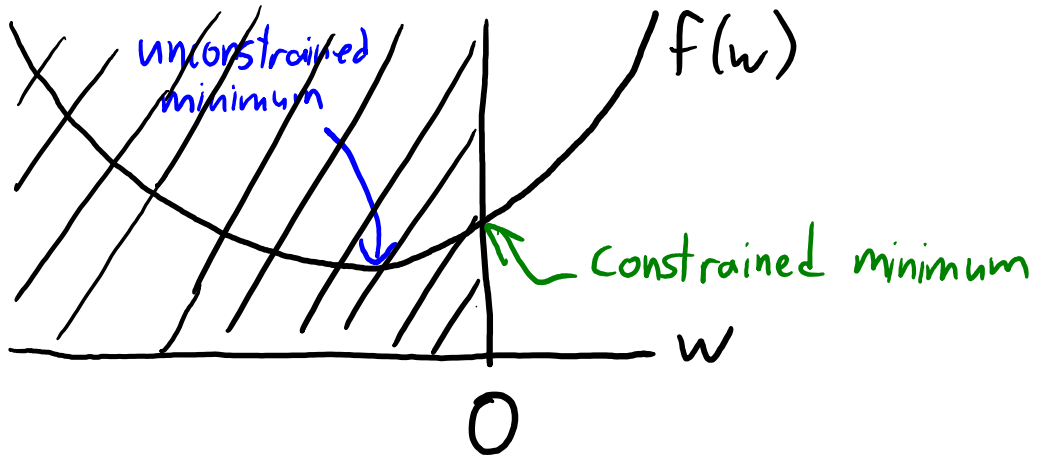

- In this case, non-negative solution is least squares solution.

# Sparsity and Non-Negative Least Squares

- Consider 1D non-negative least squares objective:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w\,x_i - y_i)^2 \quad \text{with} \quad w > 0$$

- Plotting the (constrained) objective function:




- In this case, non-negative solution is w = 0.

# Sparsity and Non-Negativity

- So non-negativity leads to sparsity.
  - Also regularizes: $w_j$ are smaller since can't "cancel" out negative values.
- How can we minimize f(w) with non-negative constraints?
  - Naive approach: solve least squares problem, set negative $w_j$ to 0.

Compute $w = (X^T X) \backslash (X^T y)$

Set $w_j = \max\{0, w_j\}$

  - This is correct when d = 1.
  - Can be worse than setting w = 0 when d ≥ 2.

# Sparsity and Non-Negativity

- So non-negativity leads to sparsity.
  - Also regularizes: $w_j$ are smaller since can't "cancel" out negative values.
- How can we minimize f(w) with non-negative constraints?
  - A correct approach is projected gradient algorithm:
    - Run a gradient descent iteration:

$$w^{t+1/2} = w^t - \alpha^t \nabla f(w^t)$$

    - After each step, set negative values to 0.

$$w_j^{t+1} = \max\{0, w_j^{t+1/2}\}$$

    - Repeat.

# Sparsity and Non-Negativity

- Similar to L1-regularization, non-negativity leads to sparsity.
  - Also regularizes: $w_j$ are smaller since can't "cancel" out negative values.
- How can we minimize f(w) with non-negative constraints?
  - Correct approach is "projected" gradient descent:

$$w^{t+1/2} = w^t - \alpha^t \nabla f(w^t) \qquad w_j^{t+1} = \max\{0, w_j^{t+1/2}\}$$

  - Similar properties to gradient descent:
    - Guaranteed decrease of 'f' if $\alpha_t$ is small enough.
    - Reaches local minimum under weak assumptions (global minimum for convex 'f').
    - Generalizations allow things like L1-regularization instead of non-negativity.

(findMinL1.m)

# Projected-Gradient for NMF

- Back to the non-negative matrix factorization (NMF) objective:

$$f(W,Z) = \sum_{i=1}^{n}\sum_{j=1}^{d}(w_j^T z_i - x_{ij})^2 \quad \text{with } w_{cj} \geq 0 \text{ and } z_{ij} \geq 0$$

  - Different ways to use projected gradient:
    - Alternate between projected gradient steps on 'W' and on 'Z'.
    - Or run projected gradient on both at once.
    - Or sample a random 'i' and 'j' and do stochastic projected gradient.

Set $z_i^{t+1} = z_i^t - \alpha^t \nabla_{z_i} f(W,Z)$ and $w_j^{t+1} = w_j^t - \alpha^t \nabla_{w_j} f(W,Z)$ for selected i and j

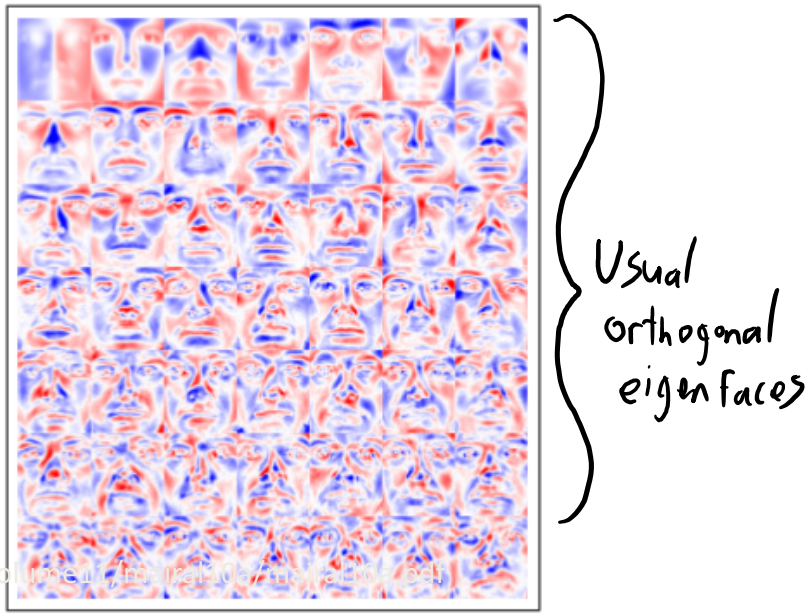
(keep other values of W and Z fixed)

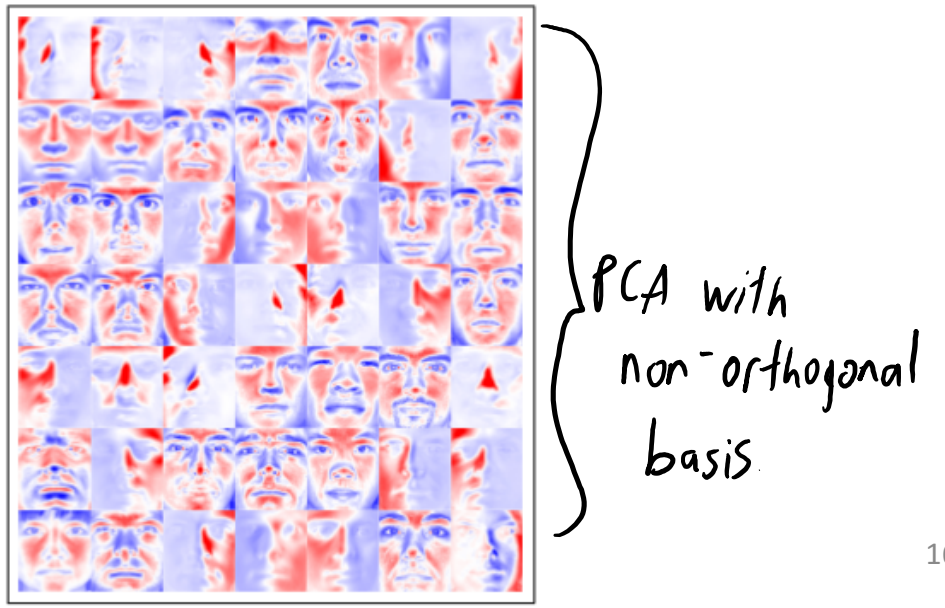
Then set negative values to 0.

  - Non-convex and (unlike PCA) is sensitive to initialization.
    - Hard to find the global optimum.
    - Typically use random initialization.

# Regularized Matrix Factorization

- For many PCA applications, ordering orthogonal PCs makes sense.
  - Latent factors are independent of each other.
  - We definitely want this for visualization.
- In other cases, ordering orthogonal PCs doesn't make sense.
  - We might not expect a natural "ordering".




http://www.jmlr.org/papers/vol[illegible]

# Regularized Matrix Factorization

- More recently people have considered L2-regularized PCA:

$$f(W,Z) = \frac{1}{2}\|ZW - X\|_F^2 + \frac{\lambda_1}{2}\|W\|_F^2 + \frac{\lambda_2}{2}\|Z\|_F^2$$

$$\frac{\lambda_1}{2}\|W\|_F^2 \rightarrow \sum_{c=1}^{k}\sum_{j=1}^{d} w_{cj}^2$$

  - Replaces normalization/orthogonality/sequential-fitting.
    - But requires regularization parameters $\lambda_1$ and $\lambda_2$.
  - Need to regularize W and Z because of scaling problem:
    - Regularizing only 'W' won't work: you could make 'Z' big to compensate.
    - You could alternately constrain one and regularize the other:

$$f(W) = \frac{1}{2}\|ZW - X\|_F^2 + \frac{\lambda}{2}\|Z\|_F^2 \quad \text{with} \quad \|w_c\| \leq 1 \quad \text{for all } 'c'$$

# Sparse Matrix Factorization

- Instead of non-negativity, we could use L1-regularization:

$$f(W,Z) = \frac{1}{2}\|ZW - X\|_F^2 + \frac{\lambda_1}{2}\sum_{i=1}^{n}\|z_i\|_1 + \frac{\lambda_2}{2}\sum_{j=1}^{d}\|w_j\|_1$$

  - Called sparse coding (L1 on 'Z') or sparse dictionary learning (L1 on 'W').
- Disadvantage of using L1-regularization over non-negativity:
  - Sparsity controlled by $\lambda_1$ and $\lambda_2$ so you need to set these.
- Advantage of using L1-regularization:
  - Negative coefficients usually make sense.
  - Sparsity controlled by $\lambda_1$ and $\lambda_2$, so you can control amount of sparsity.

# Sparse Matrix Factorization

- Instead of non-negativity, we could use L1-regularization:

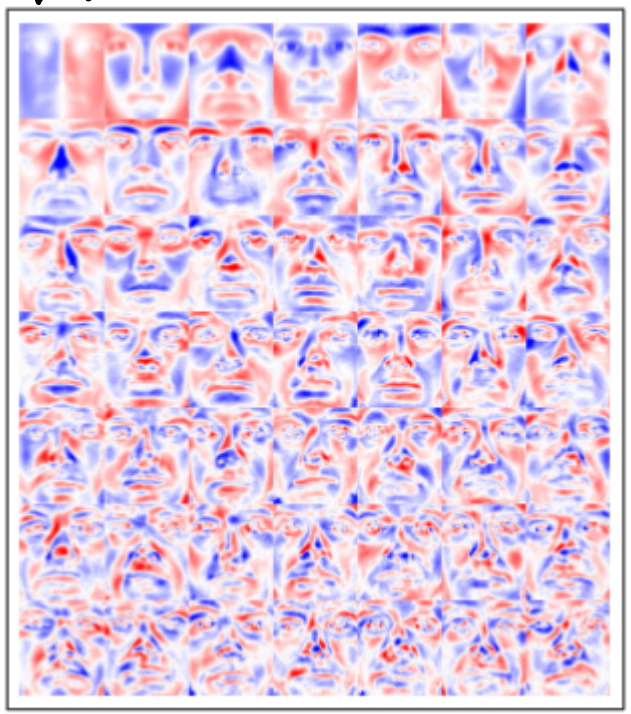
$$f(W,Z) = \frac{1}{2}\|ZW - X\|_F^2 + \frac{\lambda_1}{2}\sum_{i=1}^{n}\|z_i\|_1 + \frac{\lambda_2}{2}\sum_{j=1}^{d}\|w_j\|_1$$

  - Called sparse coding (L1 on 'Z') or sparse dictionary learning (L1 on 'W').
- Many variations exist:
  - Mixing L2-regularization and L1-regularization or making one a constraint.
  - K-SVD constrains each $z_i$ to have at most 'k' non-zeroes:
    - K-means is special case where k = 1.
    - PCA is special case where k = d.

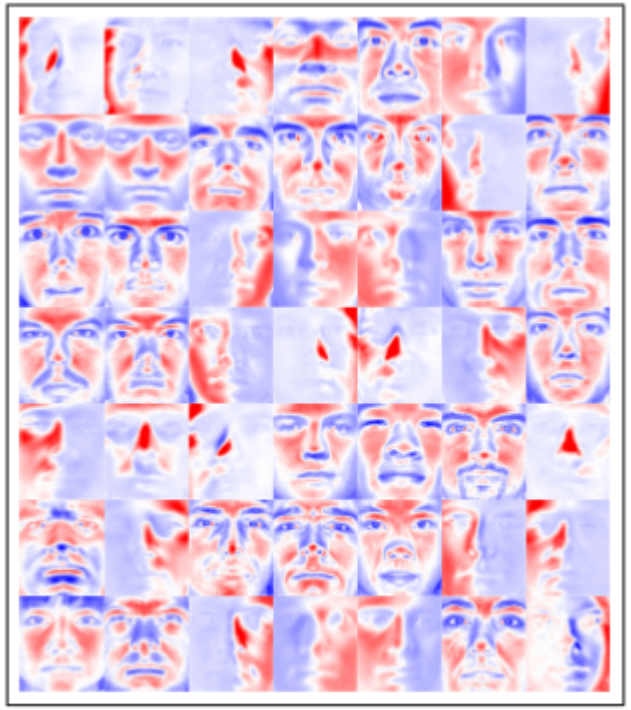
# Matrix Factorization with L1-Regularization

blue: negative

red: positive

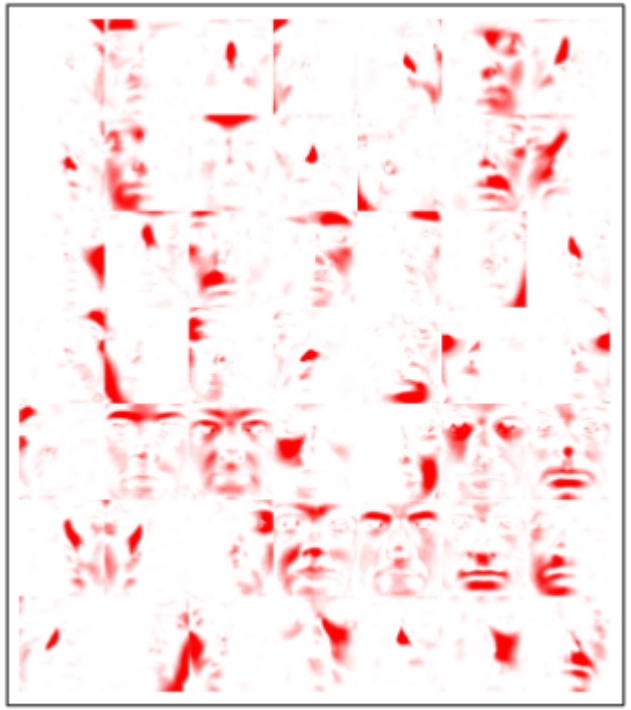


(a) PCA



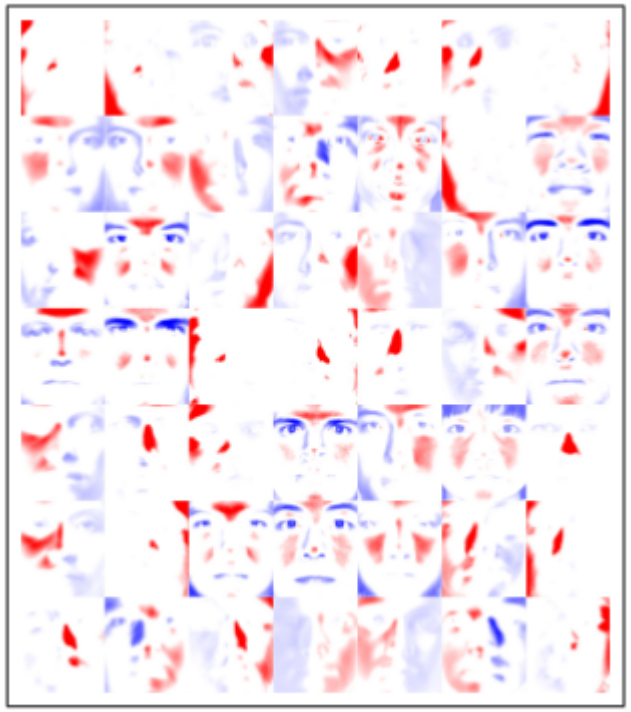
(e) Dictionary Learning

PCA without orthogonality



(c) NMF

sparsity due to non-negativity



(d) SPCA, $\tau = 30\%$

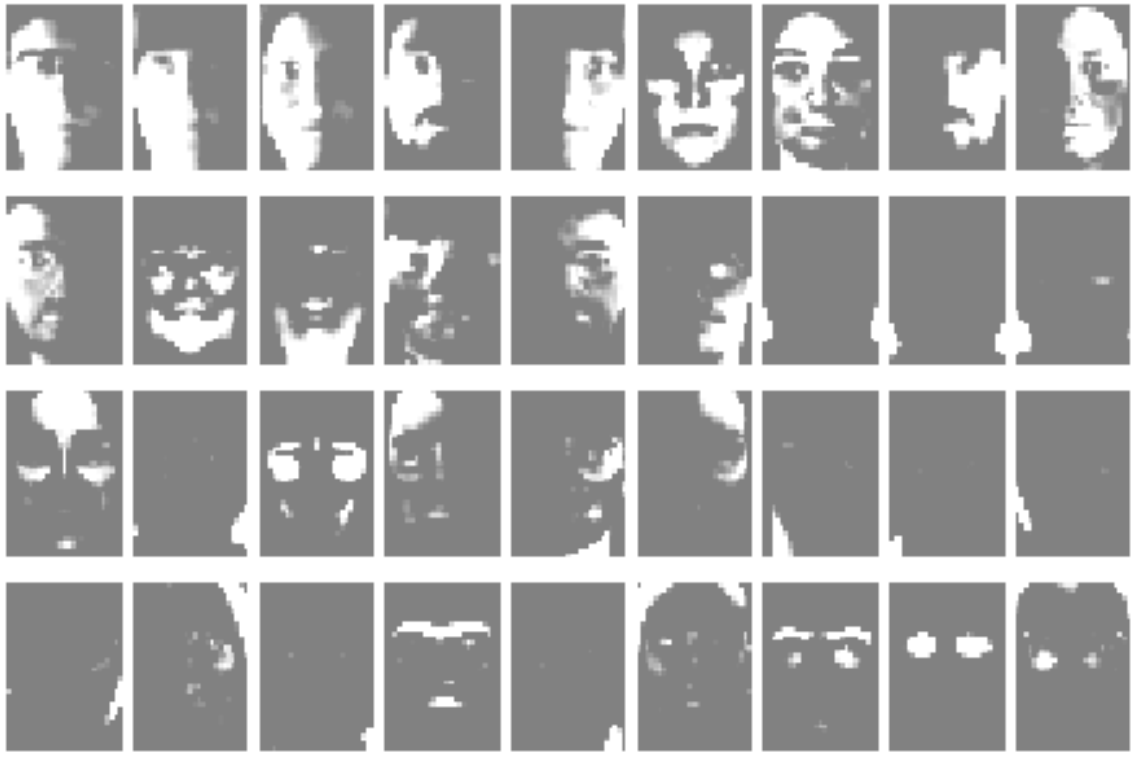
sparsity due to $L_1$-regularization



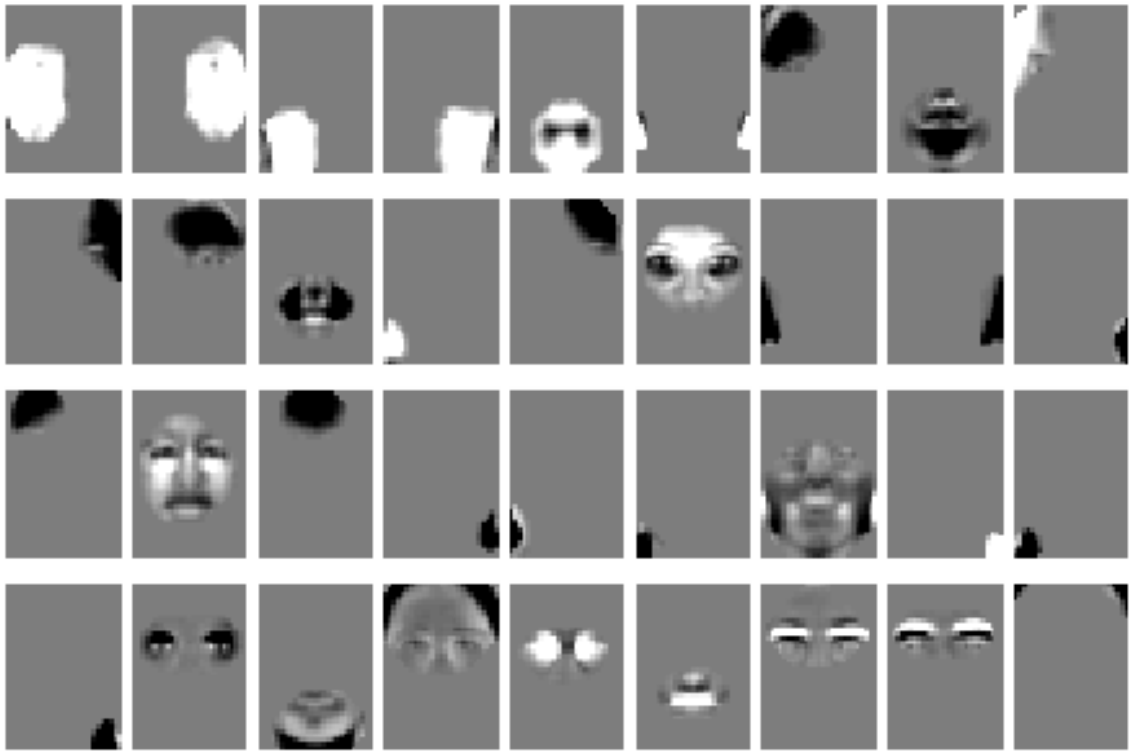
http://www.jmlr.org/papers/volume11/mairal10a/mairal10a.pdf

# Recent Work: Structured Sparsity

- "Structured sparsity" considers dependencies in sparsity patterns.



NMF

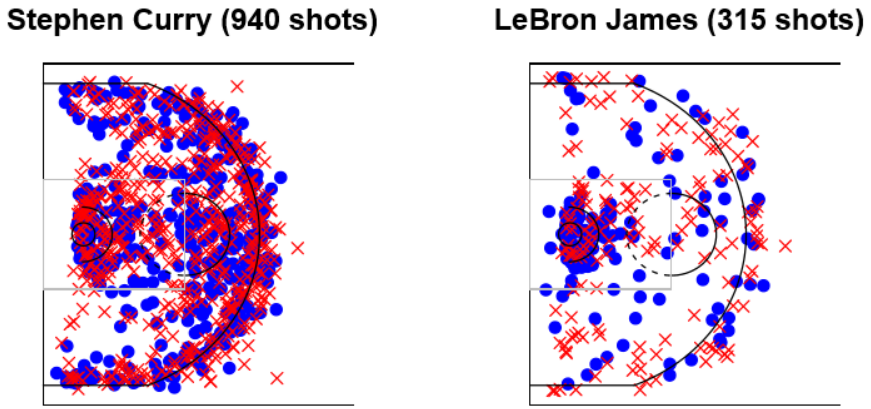


Sparse PCA with "structured" sparsity

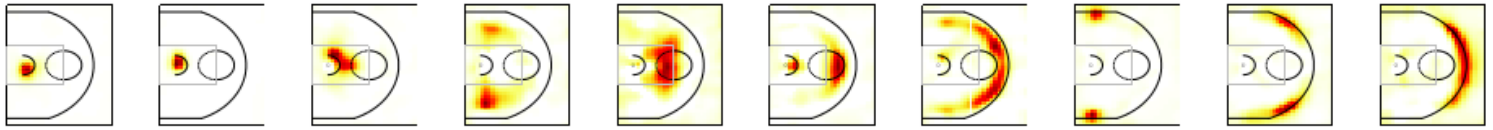
http://jmlr.org/proceedings/papers/v9/jenatton10a/jenatton10a.pdf

# Application: Sports Analytics

- NBA shot charts:




- NMF (using "KL divergence" loss with k=10 and smoothed data).
  - Negative values would not make sense here.



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| LeBron James | 0.21 | 0.16 | 0.12 | 0.09 | 0.04 | 0.07 | 0.00 | 0.07 | 0.08 | 0.17 |
| Brook Lopez | 0.06 | 0.27 | 0.43 | 0.09 | 0.01 | 0.03 | 0.08 | 0.03 | 0.00 | 0.01 |
| Tyson Chandler | 0.26 | 0.65 | 0.03 | 0.00 | 0.01 | 0.02 | 0.01 | 0.01 | 0.02 | 0.01 |
| Marc Gasol | 0.19 | 0.02 | 0.17 | 0.01 | 0.33 | 0.25 | 0.00 | 0.01 | 0.00 | 0.03 |
| Tony Parker | 0.12 | 0.22 | 0.17 | 0.07 | 0.21 | 0.07 | 0.08 | 0.06 | 0.00 | 0.00 |
| Kyrie Irving | 0.13 | 0.10 | 0.09 | 0.13 | 0.16 | 0.02 | 0.13 | 0.00 | 0.10 | 0.14 |
| Stephen Curry | 0.08 | 0.03 | 0.07 | 0.01 | 0.10 | 0.08 | 0.22 | 0.05 | 0.10 | 0.24 |
| James Harden | 0.34 | 0.00 | 0.11 | 0.00 | 0.03 | 0.02 | 0.13 | 0.00 | 0.11 | 0.26 |
| Steve Novak | 0.00 | 0.01 | 0.00 | 0.02 | 0.00 | 0.00 | 0.01 | 0.27 | 0.35 | 0.34 |

http://jmlr.org/proceedings/papers/v32/miller14.pdf

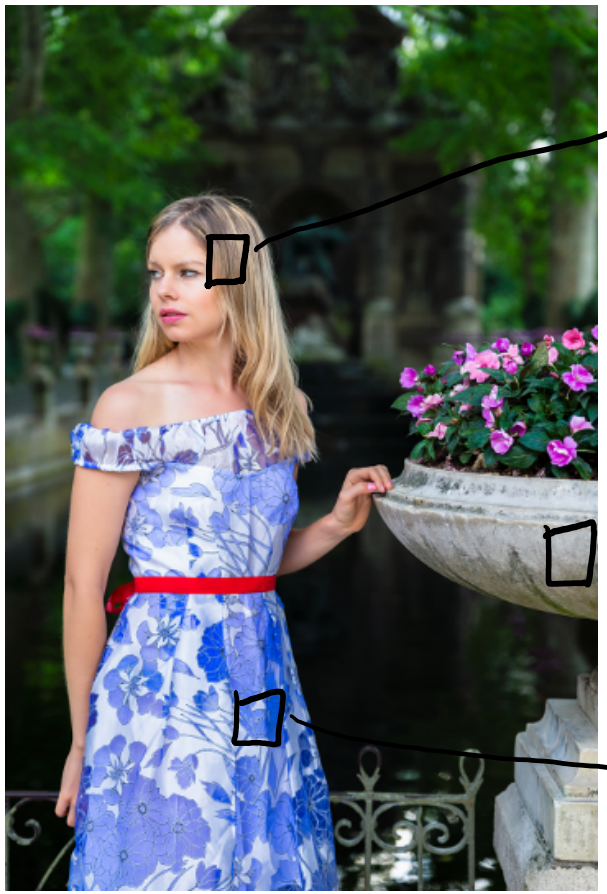
# Application: Image Restoration

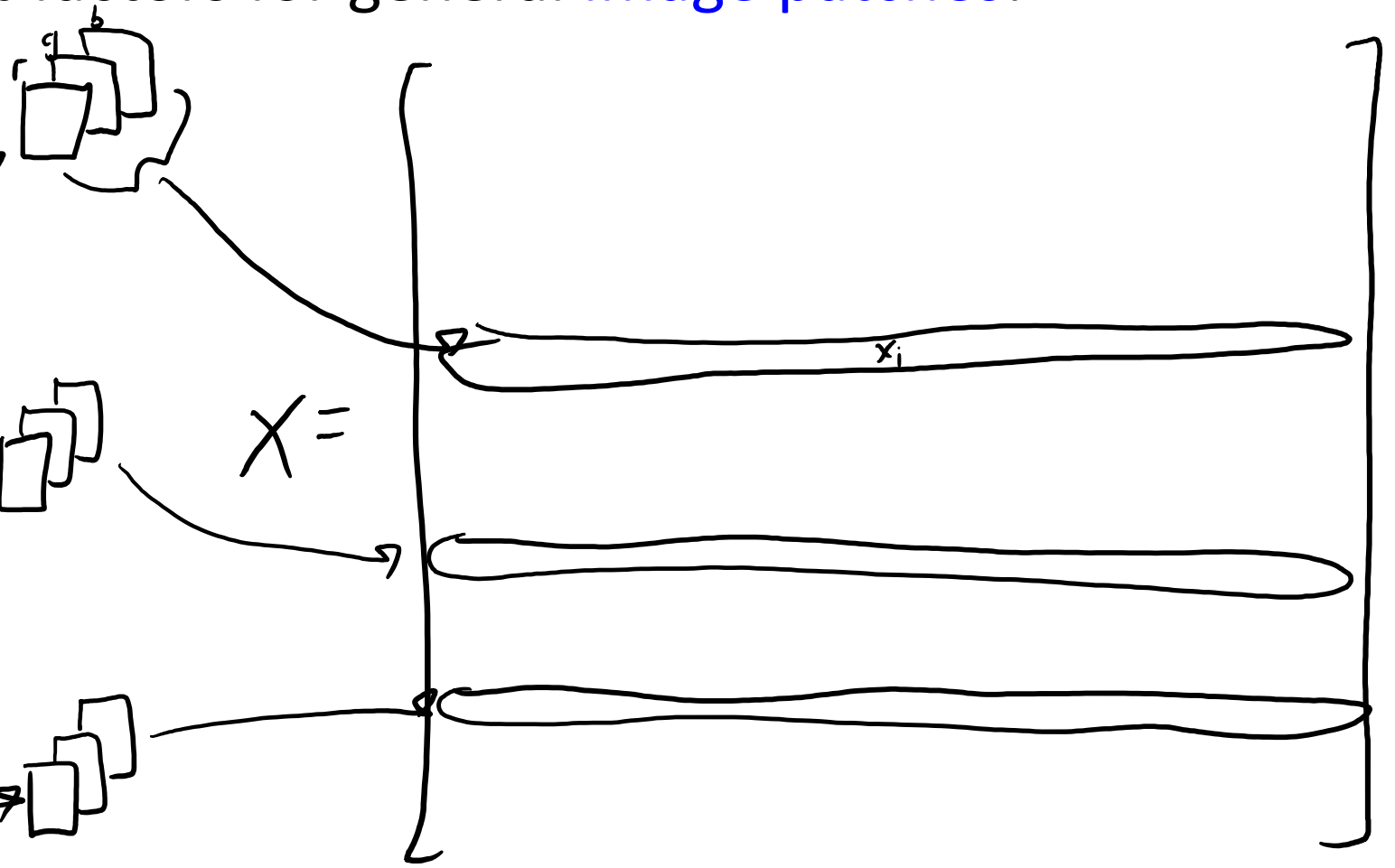



http://www.jmlr.org/papers/volume11/mairal10a/mairal10a.pdf

# Latent-Factor Models for Image Patches

- Consider building latent-factors for general image patches:

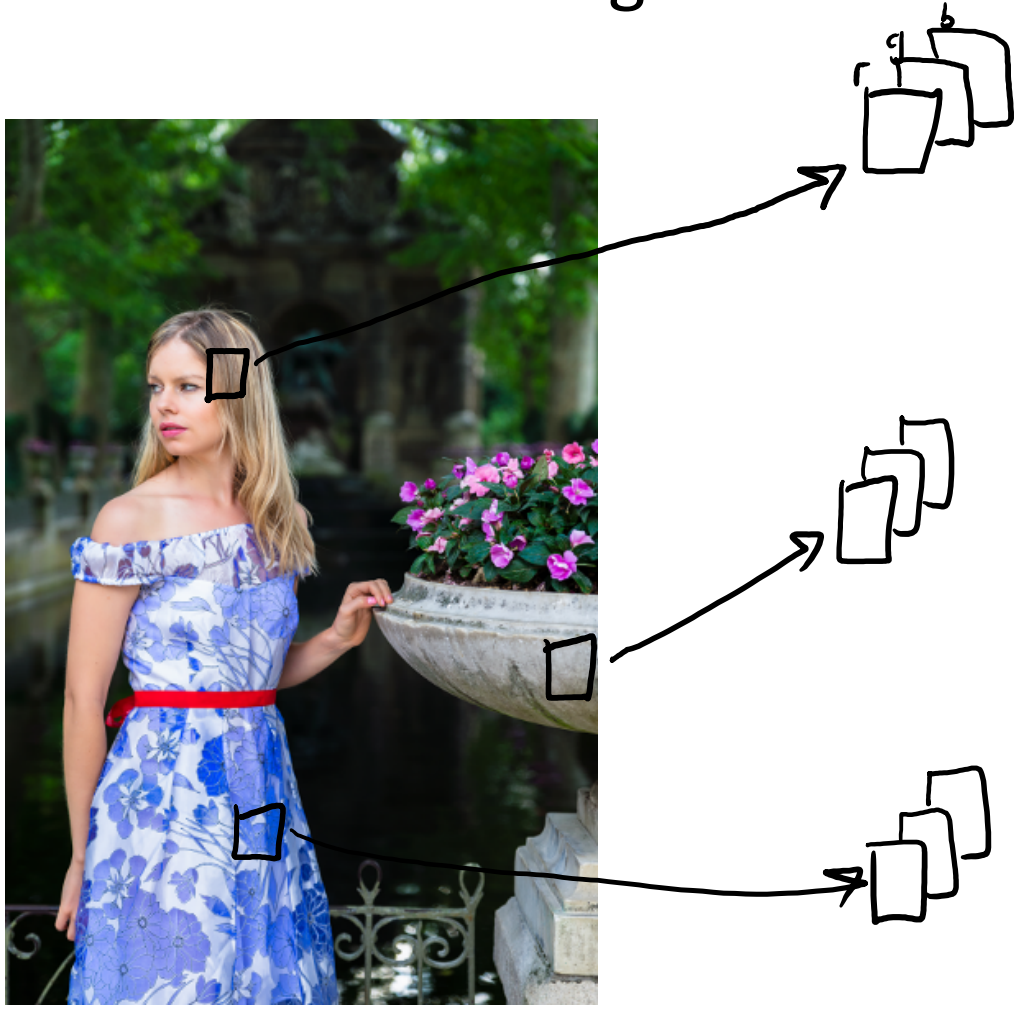





Size of X: ((image height) x (image width) by ((patch height) x (patch width) x 3)

# Latent-Factor Models for Image Patches

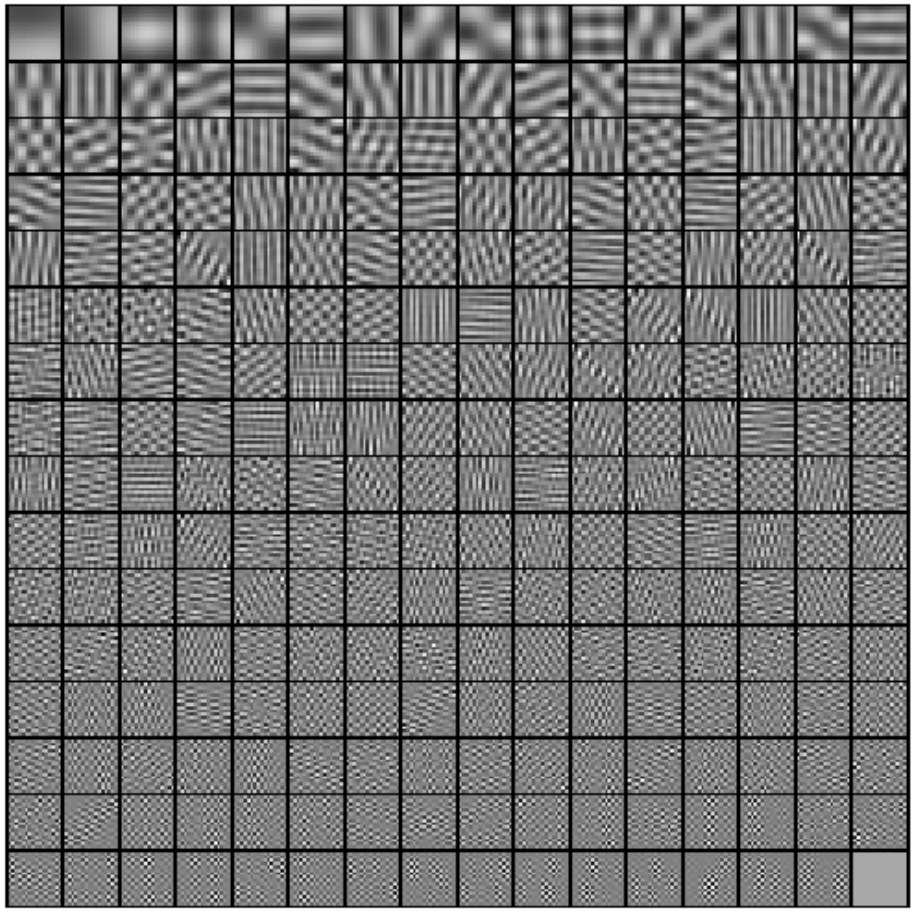
- Consider building latent-factors for general image patches:



Typical pre-processing:

1. Usual variable centering
2. "Whiten" patches. (remove correlations)

# Latent-Factor Models for Image Patches

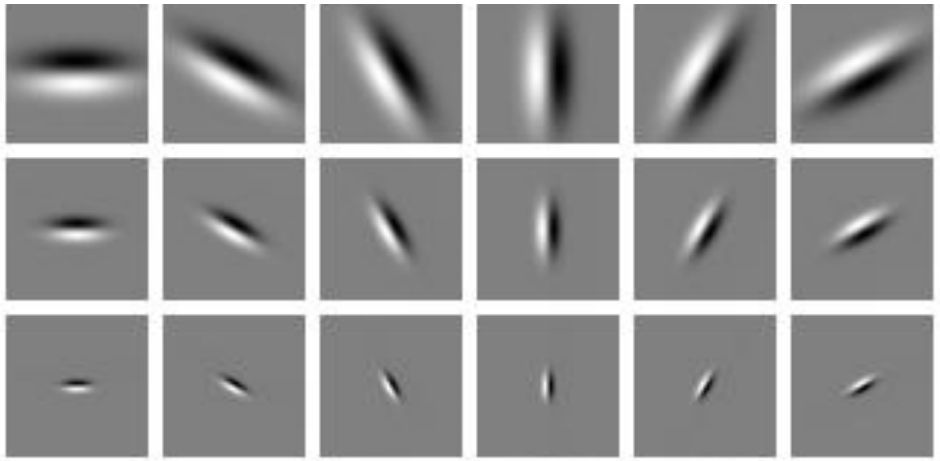


(b) Principal components.

Orthogonal bases don't seem right:

- Few PCs do almost everything.
- Most PCs do almost nothing.

We believe "simple cells" in visual cortex use:

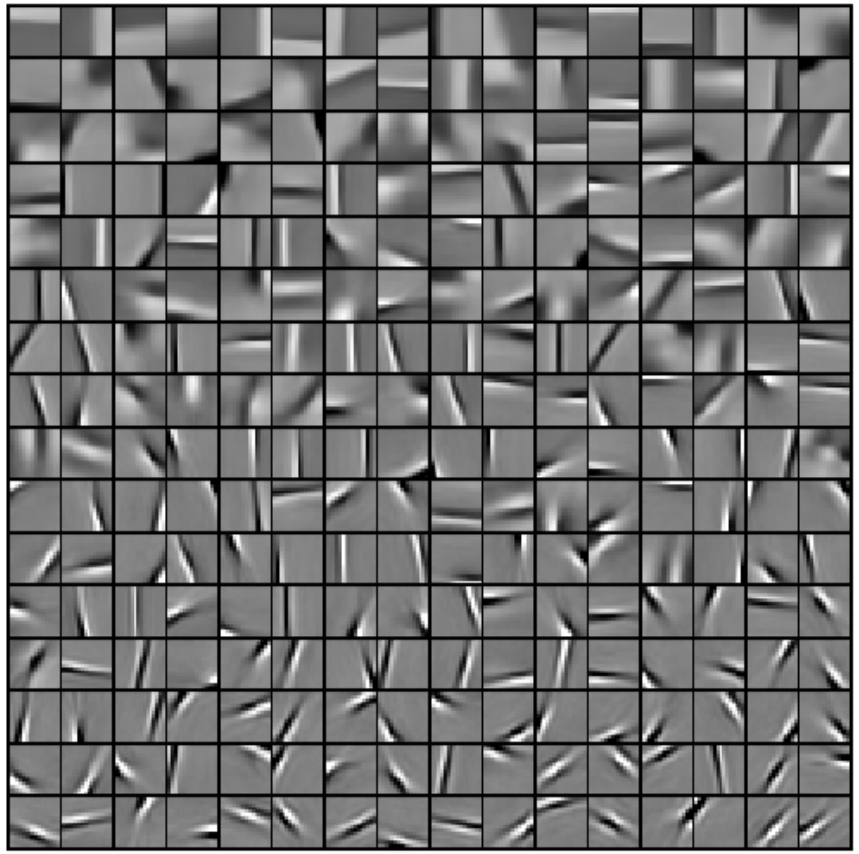


'Gabor' filters

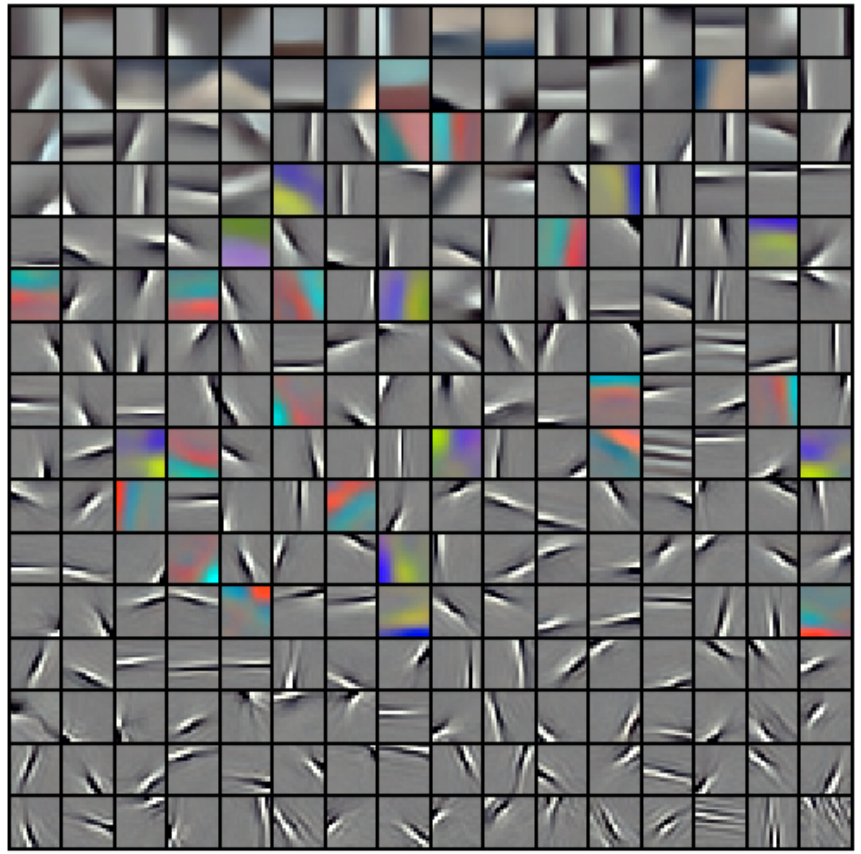
http://lear.inrialpes.fr/people/mairal/resources/pdf/review_sparse_arxiv.pdf
http://stackoverflow.com/questions/16059462/comparing-textures-with-opencv-and-gabor-filters

# Latent-Factor Models for Image Patches

- Results from a sparse (non-orthogonal) latent factor model:



(a) With centering - gray.

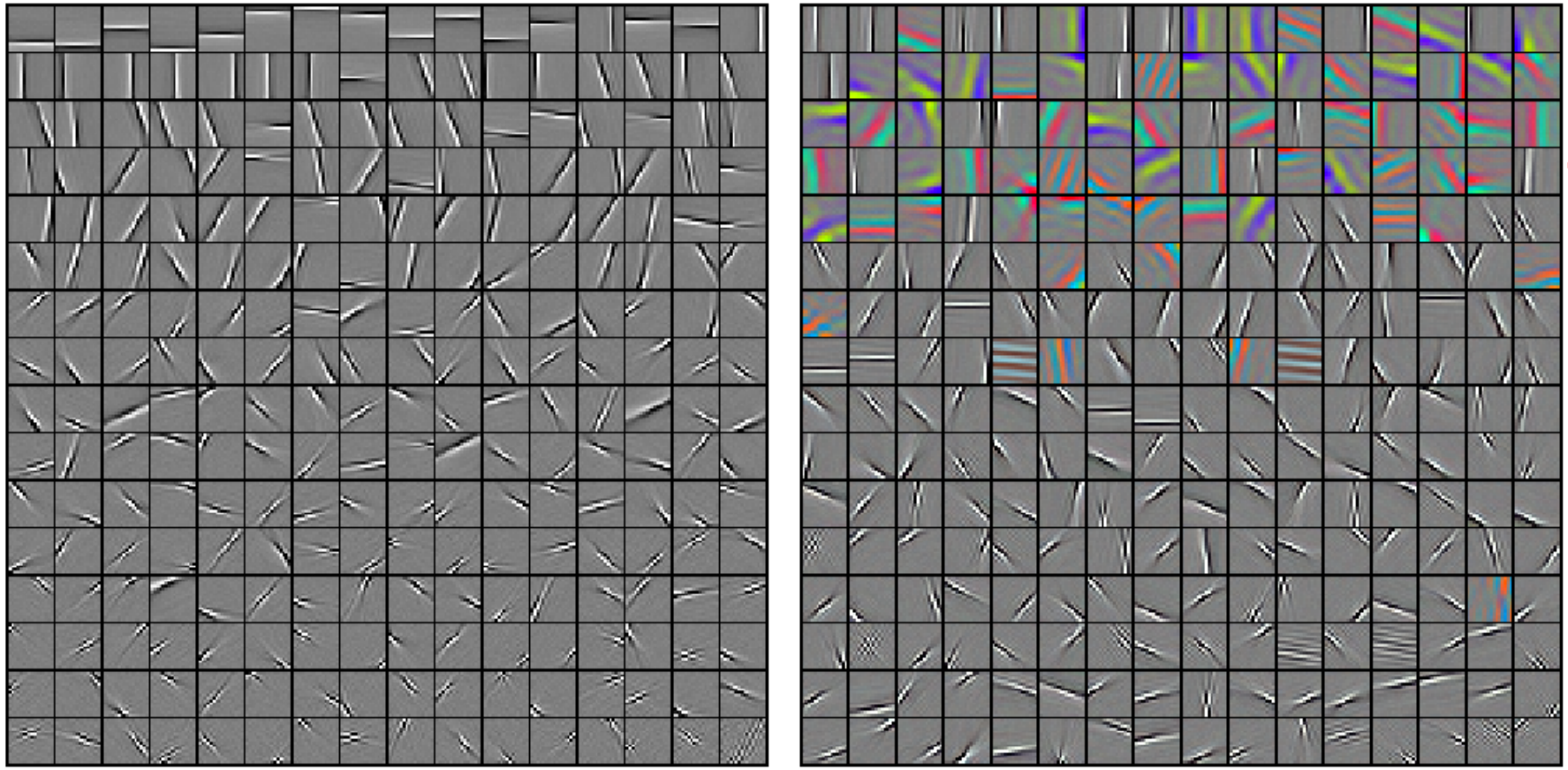


(b) With centering - RGB.

http://lear.inrialpes.fr/people/mairal/resources/pdf/review_sparse_arxiv.pdf

# Latent-Factor Models for Image Patches

- When you add whitening:



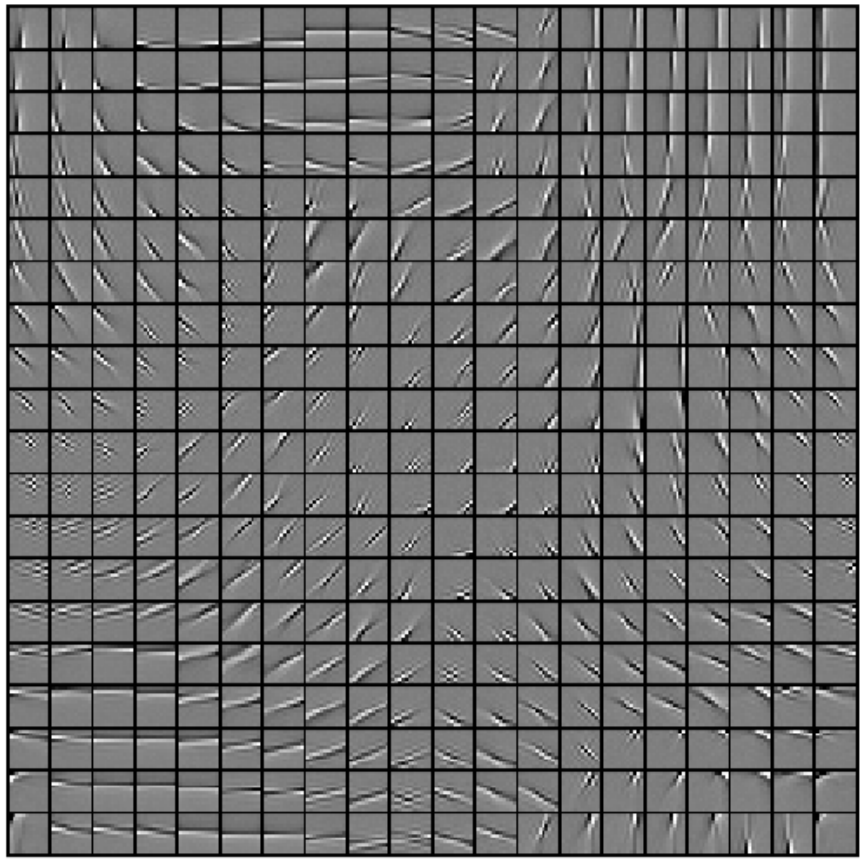

(c) With whitening - gray.

(d) With whitening - RGB.

http://lear.inrialpes.fr/people/mairal/resources/pdf/review_sparse_arxiv.pdf

# Recent Work: Structured Sparsity

- Basis learned with a variant of "structured sparsity":



(b) With $4 \times 4$ neighborhood.

Similar to "cortical columns" theory in visual cortex.

http://lear.inrialpes.fr/people/mairal/resources/pdf/review_sparse_arxiv.pdf

# Beyond Squared Error

- Our (unregularized) objective for latent-factor models (LFM):

$$f(W,Z) = \sum_{i=1}^{n}\sum_{j=1}^{d}(w_j^T z_i - x_{ij})^2$$

- As before, there are squared error alternatives.
- We can get a LFM for binary $x_{ij}$ using the logistic loss:

$$f(W,Z) = \sum_{i=1}^{n}\sum_{j=1}^{d}\log(1+\exp(-x_{ij}w_j^T z_i))$$

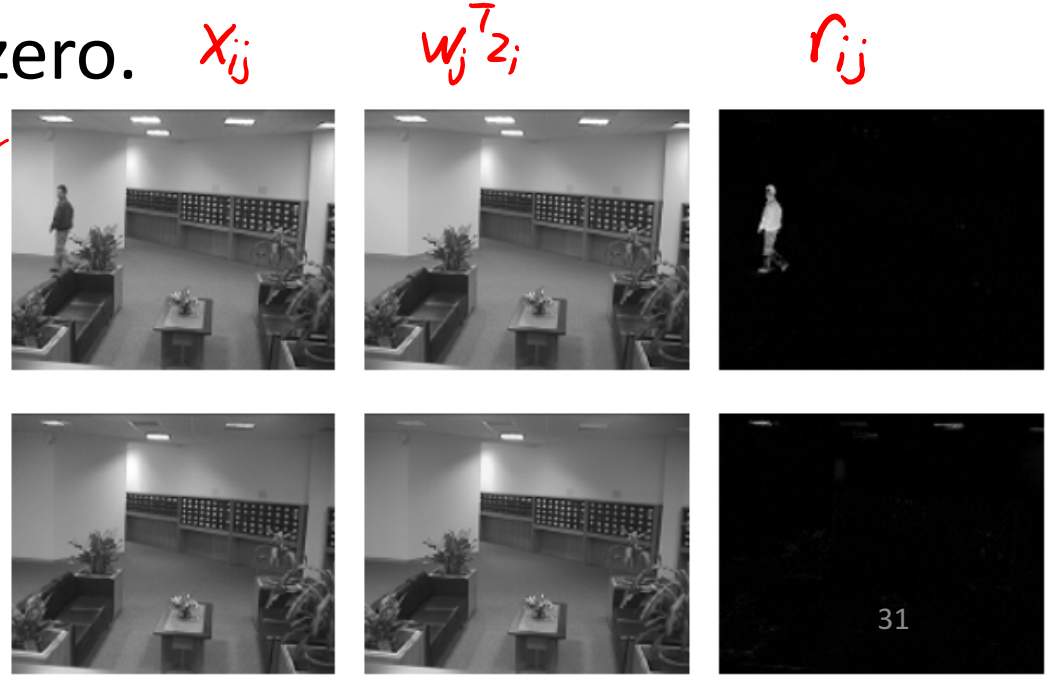
# Robust PCA

- Robust PCA methods use the absolute error:

$$f(W,Z) = \sum_{i=1}^{n}\sum_{j=1}^{d} |w_j^T z_i - x_{ij}|$$

- Will be robust to outliers in the matrix 'X'.
- Encourages "residuals" $r_{ij}$ to be exactly zero.
  - Non-zero $r_{ij}$ are where the "outliers" are.

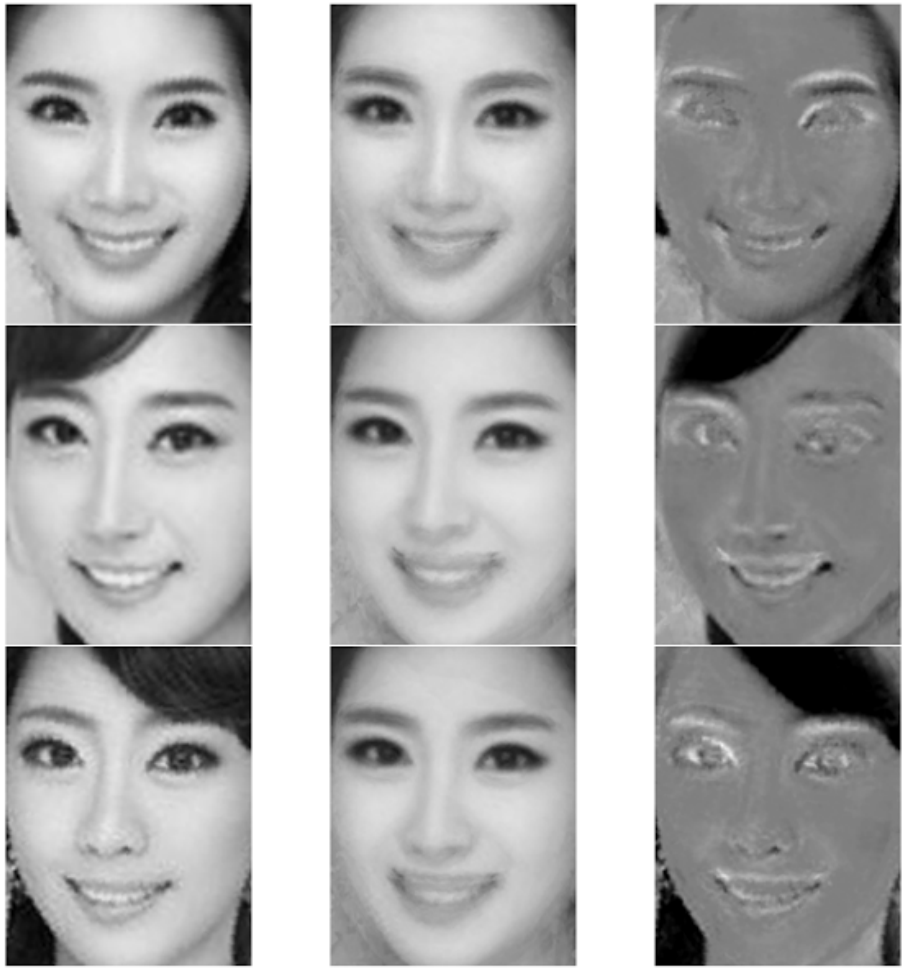
Applying robust PCA to video frames




http://statweb.stanford.edu/~candes/papers/RobustPCA.pdf

# Robust PCA

- Miss Korea contestants and robust PCA:



Original image | Low rank reconstruction | Sparse error

http://jbhuang0604.blogspot.ca/2013/04/miss-korea-2013-contestants-face.html

# Summary

- Non-negativity constraints lead to sparse solution.
- Projected gradient adds constraints to gradient descent.
- Non-orthogonal LFMs make sense in many applications.
- L1-regularization leads to other sparse LFMs.
- Robust PCA allows identifying certain types of outliers.

- Next time: predicting which movies you are going to like.