The convex geometry of inverse problems

Benjamin Recht
Department of Computer Sciences
University of Wisconsin-Madison

Joint work with
Venkat Chandrasekaran
Pablo Parrilo
Alan Willsky

Linear Inverse Problems

- Find me a solution of

$$y = \Phi x$$

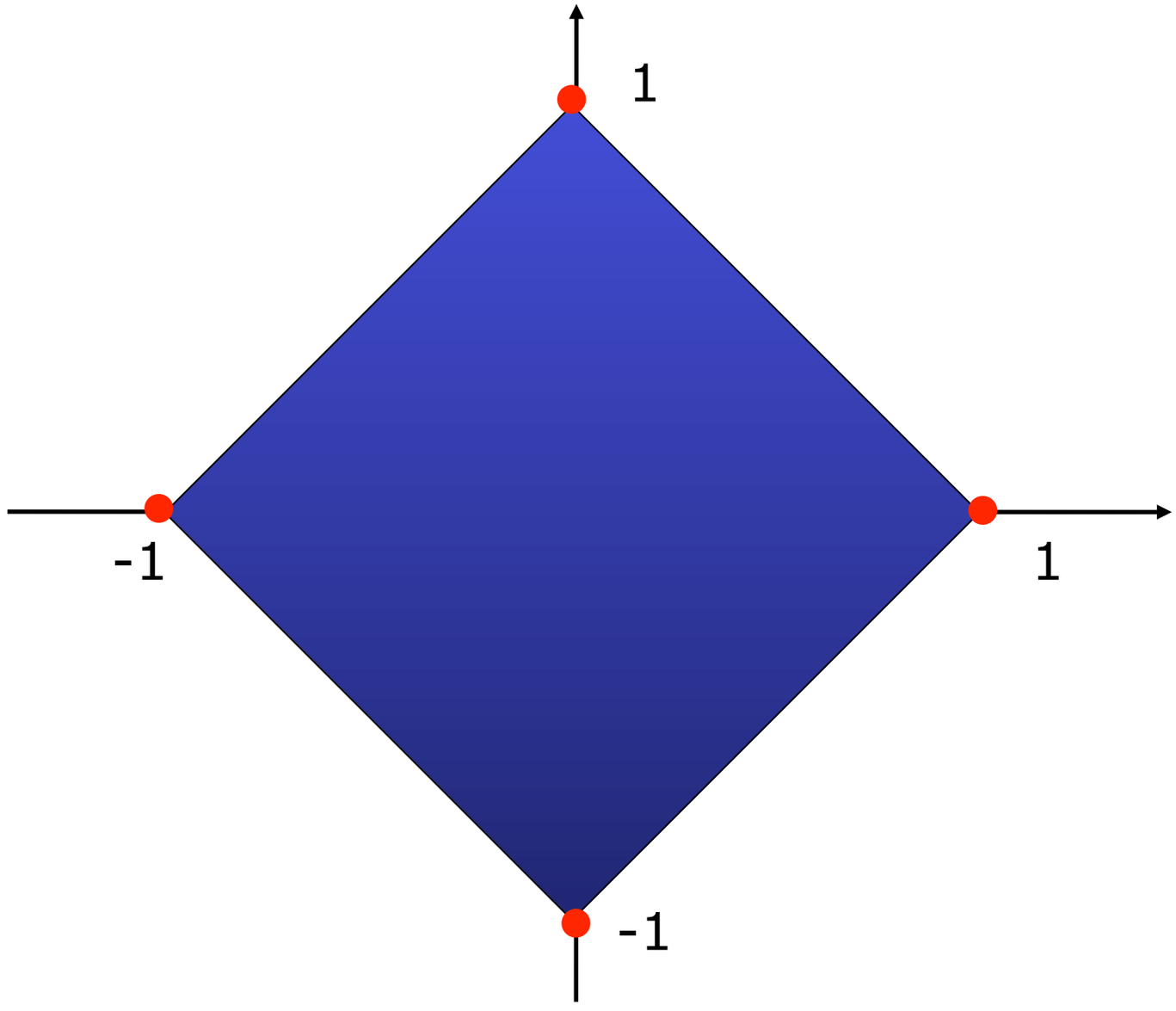
- Φ m x n, m<n
- Of the infinite collection of solutions, which one should we pick?
- Leverage structure:

Sparsity | Rank | Smoothness | Symmetry

- How do we design algorithms to solve underdetermined systems problems with priors?

Sparsity

- 1-sparse vectors of Euclidean norm 1

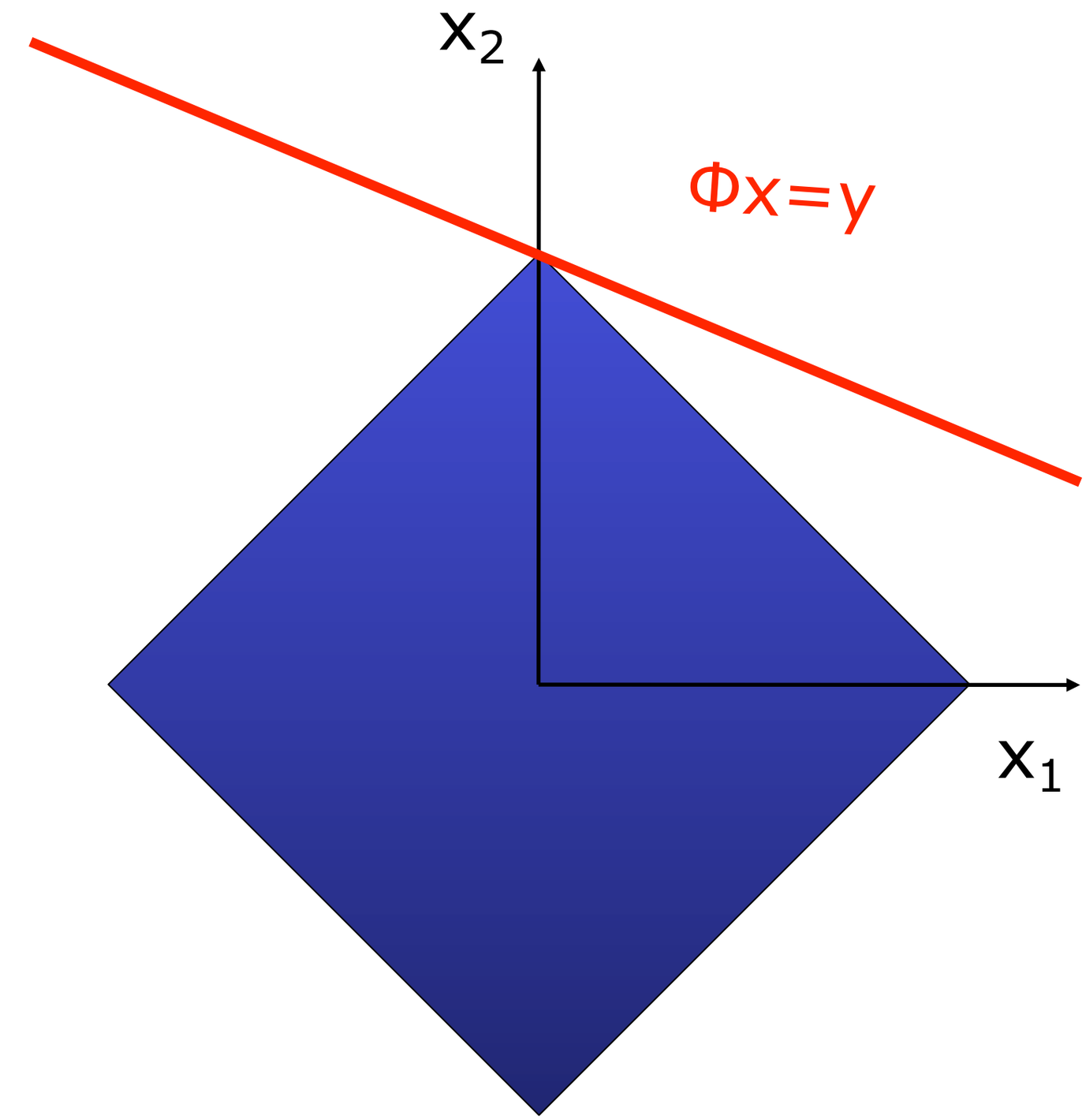


- Convex hull is the unit ball of the l_1 norm

$$\{x \ : \ \|x\|_1 \leq 1\}$$

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

$$\begin{aligned} \text{minimize} \quad & \|x\|_1 \\ \text{subject to} \quad & \Phi x = y \end{aligned}$$

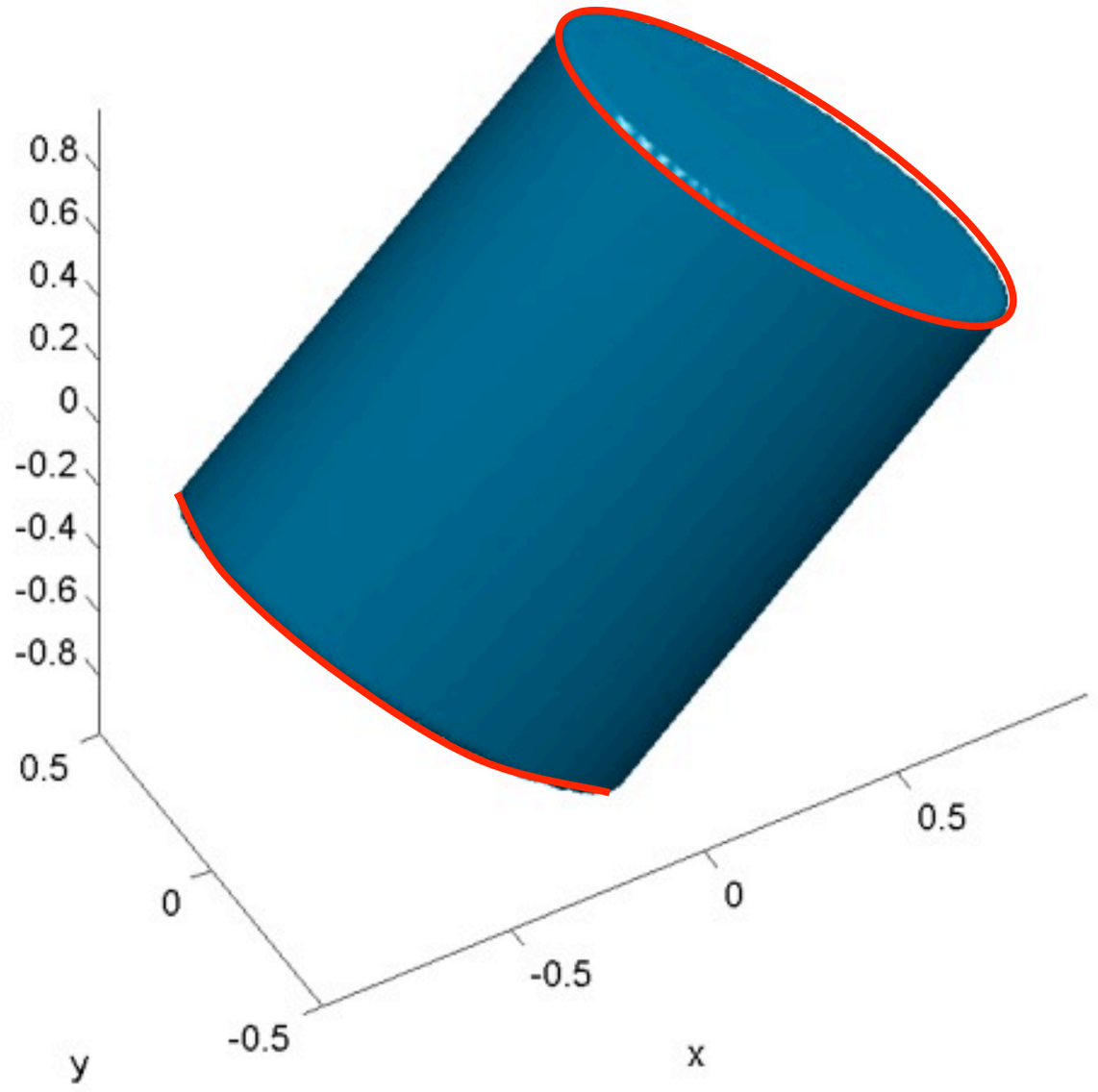


Compressed Sensing: Candes, Romberg, Tao, Donoho, Tanner, Etc...

Rank

- 2x2 matrices
- plotted in 3d

$$\begin{bmatrix} x & y \\ y & z \end{bmatrix}$$



— rank 1

$x^2 + z^2 + 2y^2 = 1$

Convex hull:

$$\{X \ : \ \|X\|_* \leq 1\}$$

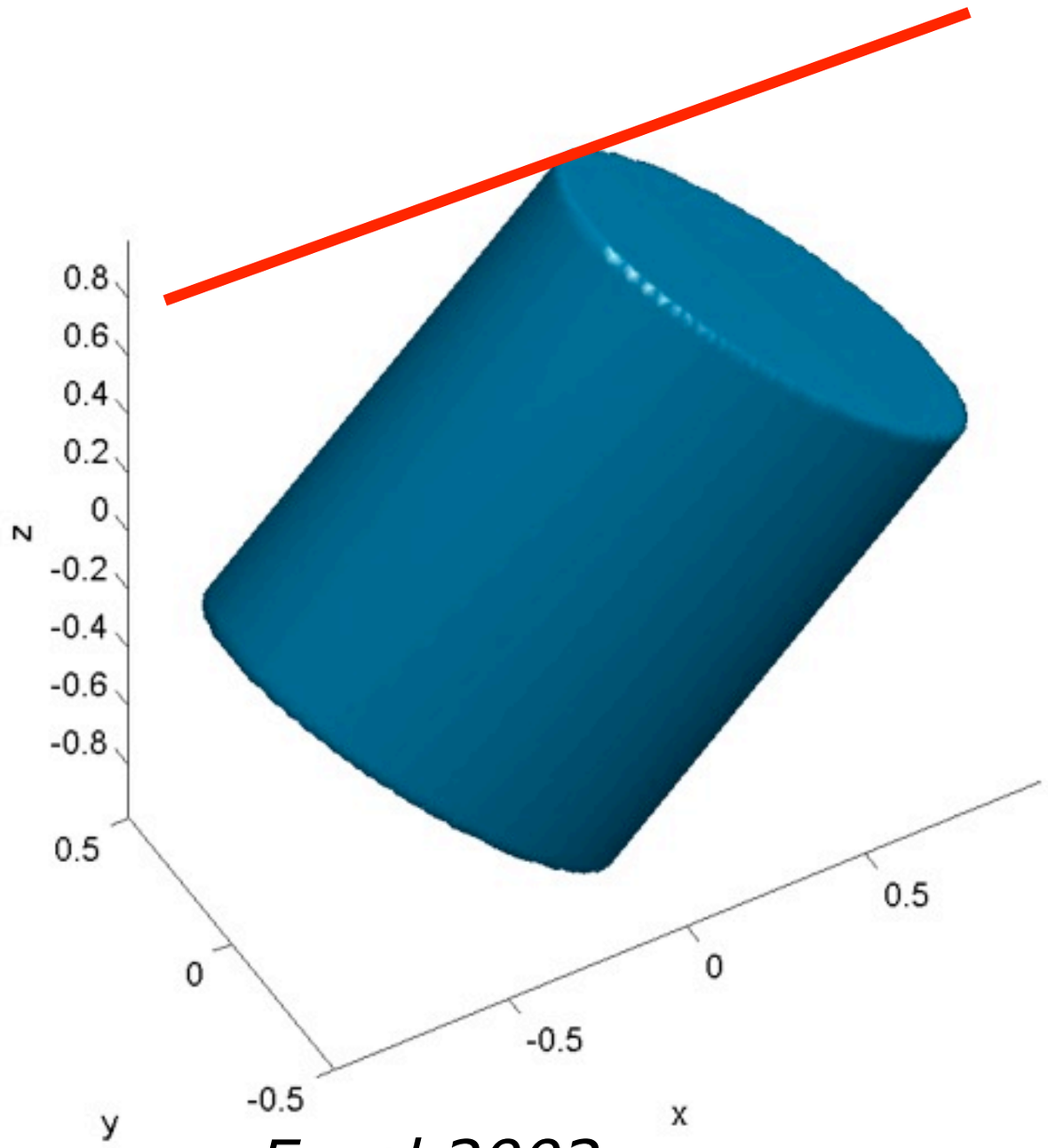
$$\|X\|_* = \sum_i \sigma_i(X)$$

- 2x2 matrices
- plotted in 3d

$$\left\| \begin{bmatrix} x & y \\ y & z \end{bmatrix} \right\|_* \leq 1$$

$$\|X\|_* = \sum_i \sigma_i(X)$$

Nuclear Norm Heuristic

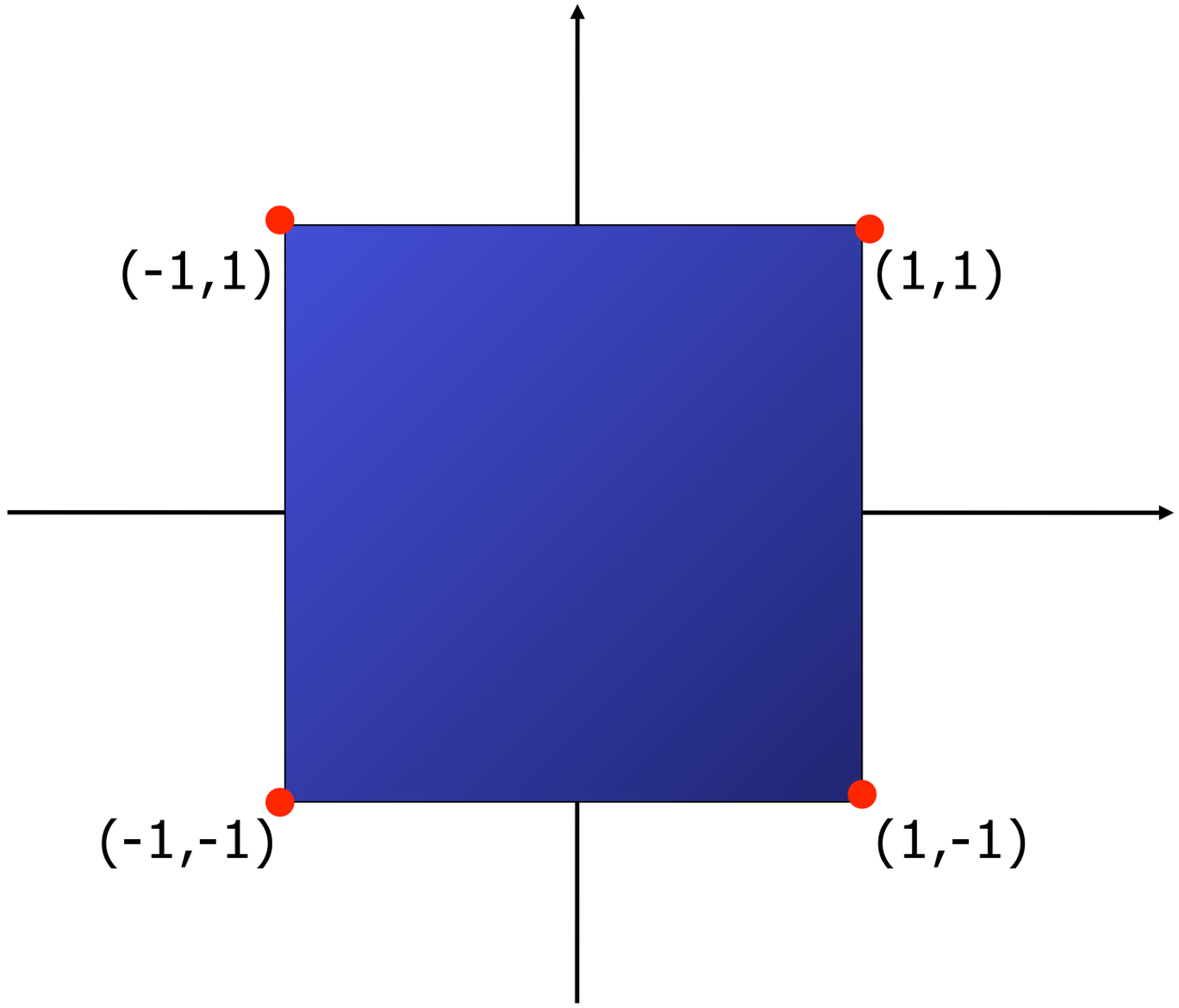


Fazel 2002.
R, Fazel, and Parillo 2007
Rank Minimization/Matrix Completion

Integer Programming

- Integer solutions:

  all components of x are ±1



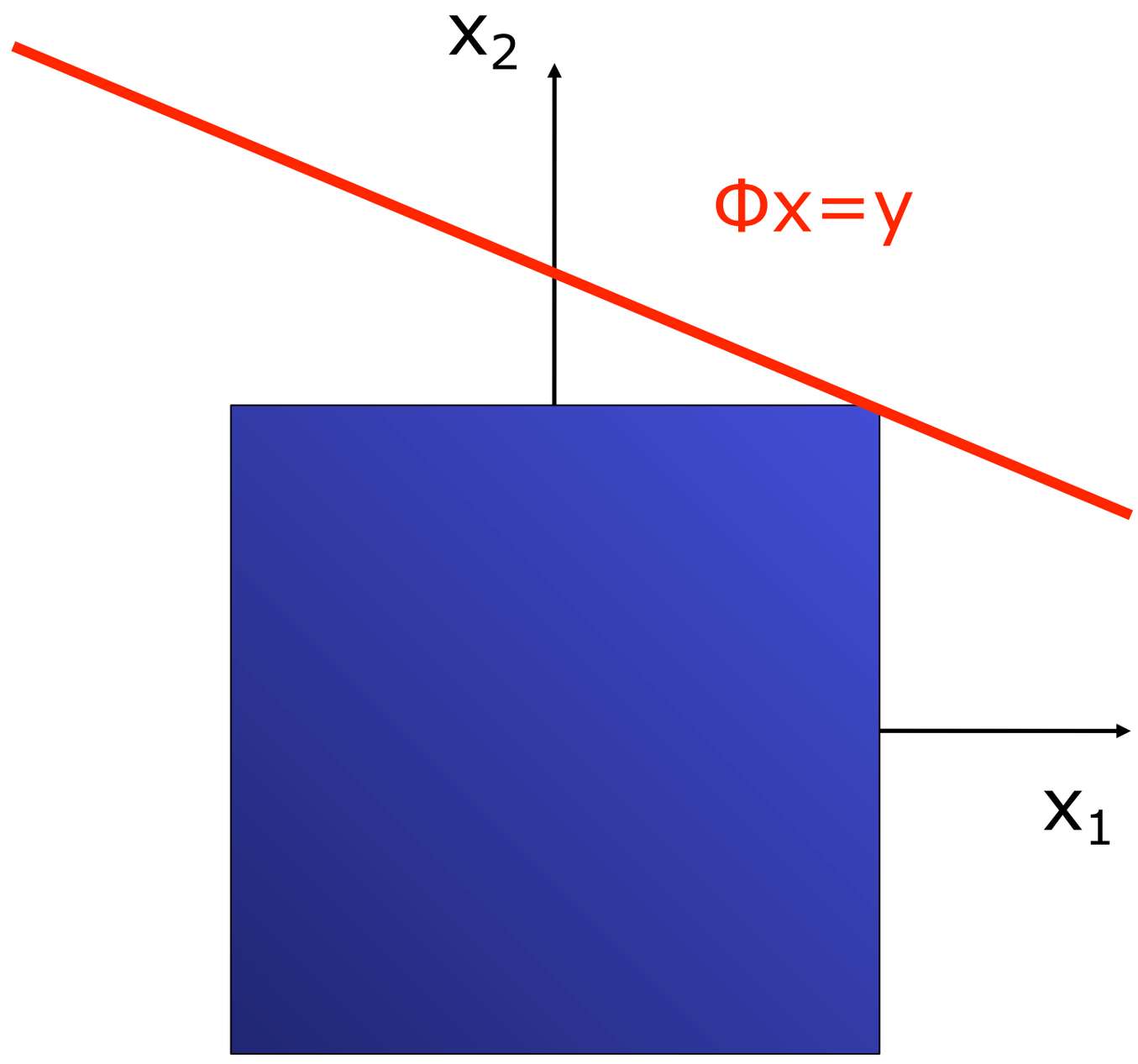
- Convex hull is the

  unit ball of the l_1 norm

$$\{x \ : \ \|x\|_\infty \leq 1\}$$

$$\|x\|_\infty = \max_i |x_i|$$

$$\begin{aligned} \text{minimize} \quad & \|x\|_\infty \\ \text{subject to} \quad & \Phi x = y \end{aligned}$$



Donoho and Tanner 2008
Mangasarian and Recht. 2009.

Parsimonious Models

$$x = \sum_{k=1}^{r} w_k \alpha_k$$

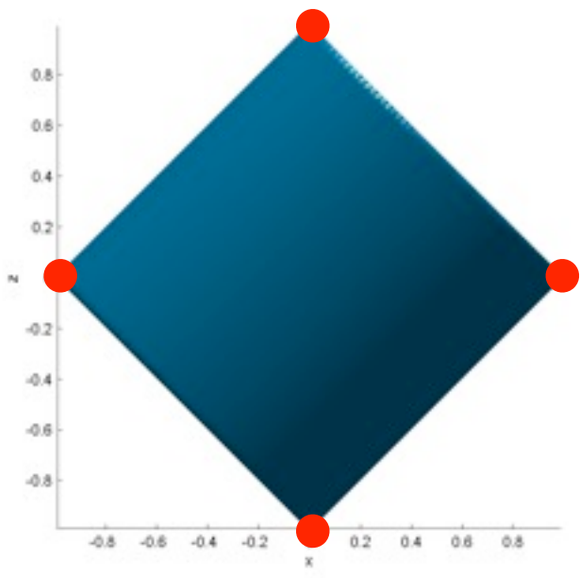
rank

model

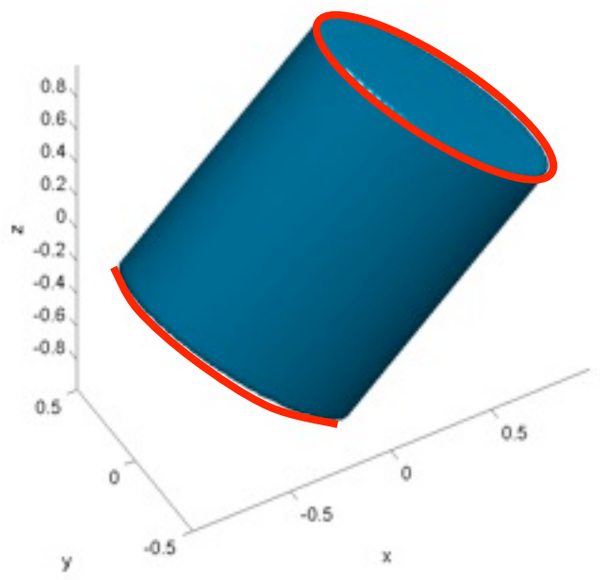
weights

atoms

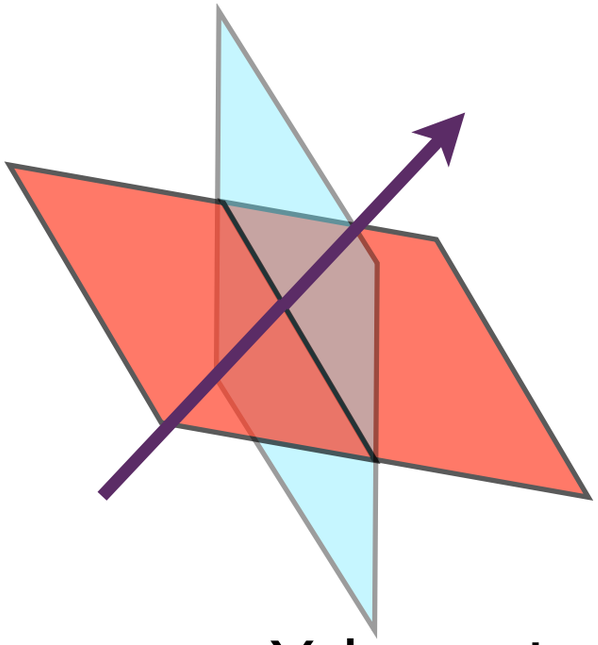
- Search for best linear combination of fewest atoms
- “rank” = fewest atoms needed to describe the model

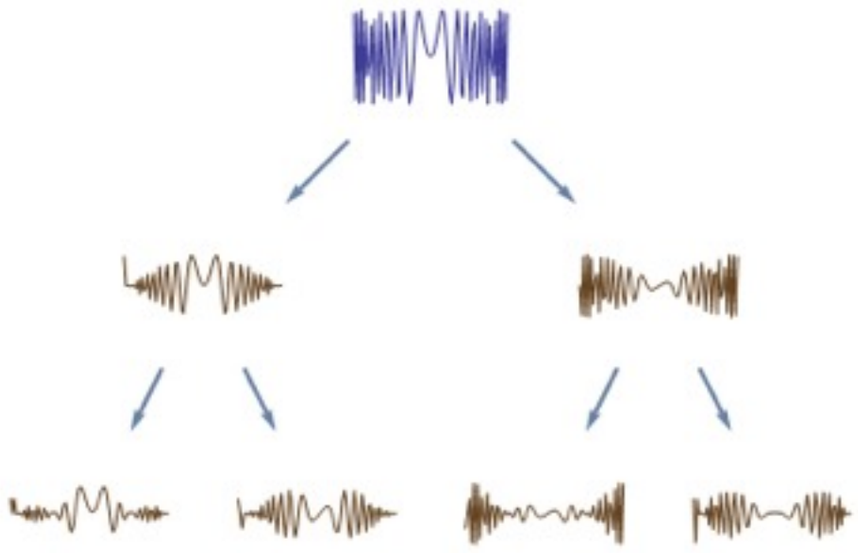


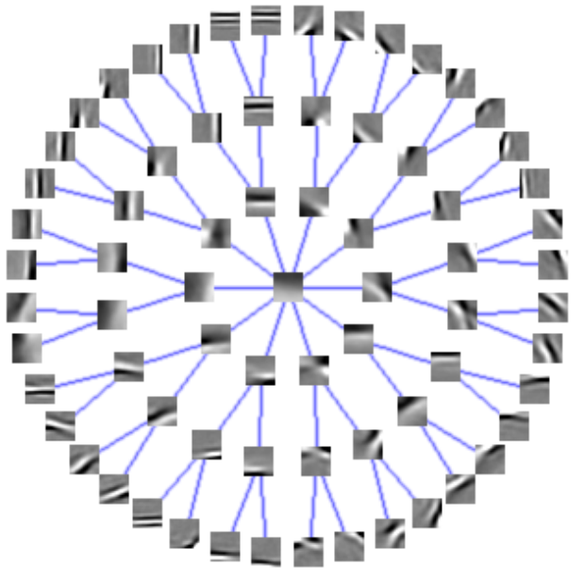
$$\|x\|_{\mathcal{A}} \equiv \inf_{(w,\alpha)} \sum_{k=1}^{r} |w_k|$$



Union of Subspaces







- X has structured sparsity: linear combination of elements from a set of subspaces $\{U_g\}$.
- Atomic set: unit norm vectors living in one of the U_g

$$\|x\|_{\mathcal{G}} = \inf \left\{ \sum_{g \in G} \|w_g\| \ : \ x = \sum_{g \in G} w_g, \ \ w_g \in U_g \right\}$$

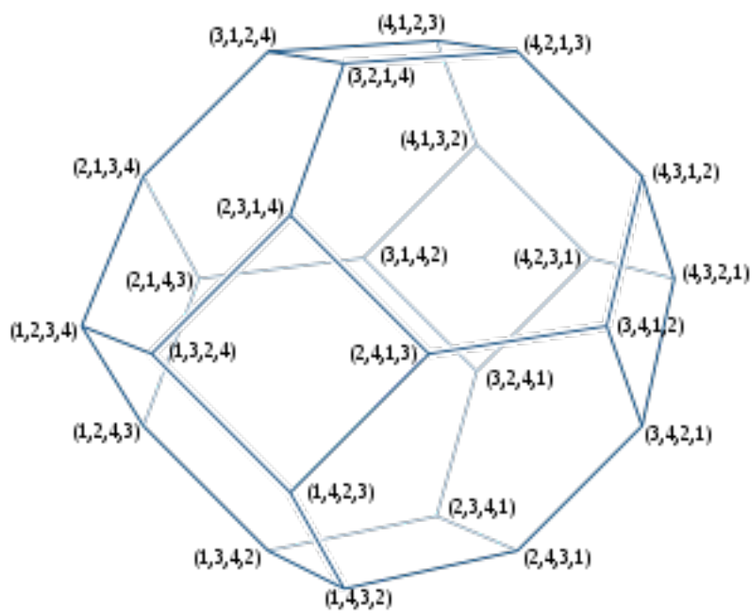
- Proposed by Jacob, Obozinski and Vert (2009).

Permutation Matrices

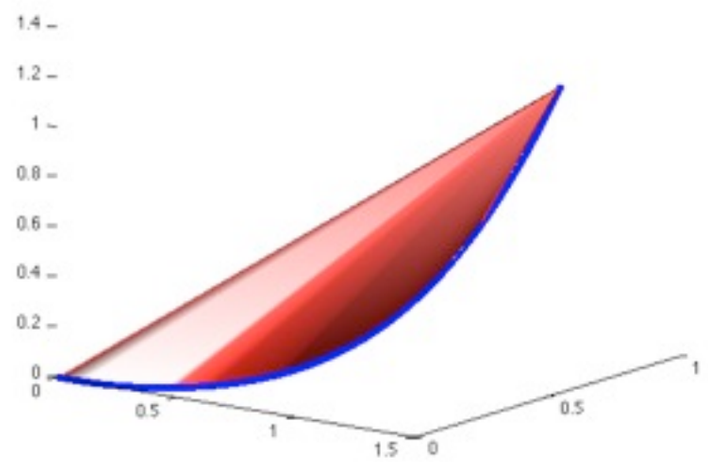
- X a sum of a few permutation matrices
- Examples: Multiobject Tracking (Huang et al), Ranked elections (Jagabathula, Shah)

- Convex hull of the permutation matrices: Birkhoff Polytope of doubly stochastic matrices
- *Permutahedra*: convex hull of permutations of a fixed vector.

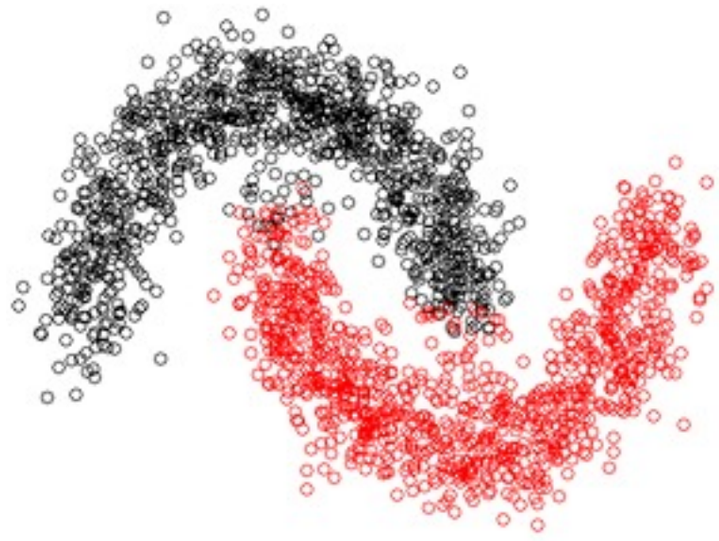
[1,2,3,4] →



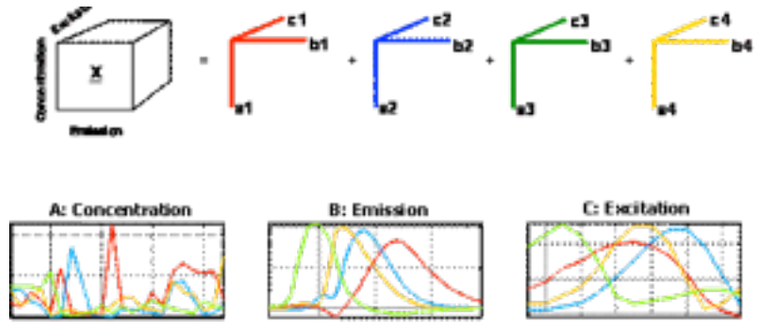
- Moments: convex hull of of $[1,t,t^2,t^3,t^4,...]$, $t \in T$, some basic set.
- System Identification, Image Processing, Numerical Integration, Statistical Inference
- **Solve with semidefinite programming**



- Cut-matrices: sums of rank-one sign matrices.
- Collaborative Filtering, Clustering in Genetic Networks, Combinatorial Approximation Algorithms
- **Approximate with semidefinite programming**



- Low-rank Tensors: sums of rank-one tensors
- Computer Vision, Image Processing, Hyperspectral Imaging, Neuroscience
- **Approximate with alternating least-squares**



Atomic Norms

- Given a basic set of *atoms,* $\mathcal{A}$, define the function

$$\|x\|_{\mathcal{A}} = \inf\{t > 0 \ : \ x \in t\mathrm{conv}(\mathcal{A})\}$$

- When $\mathcal{A}$ is centrosymmetric, we get a norm

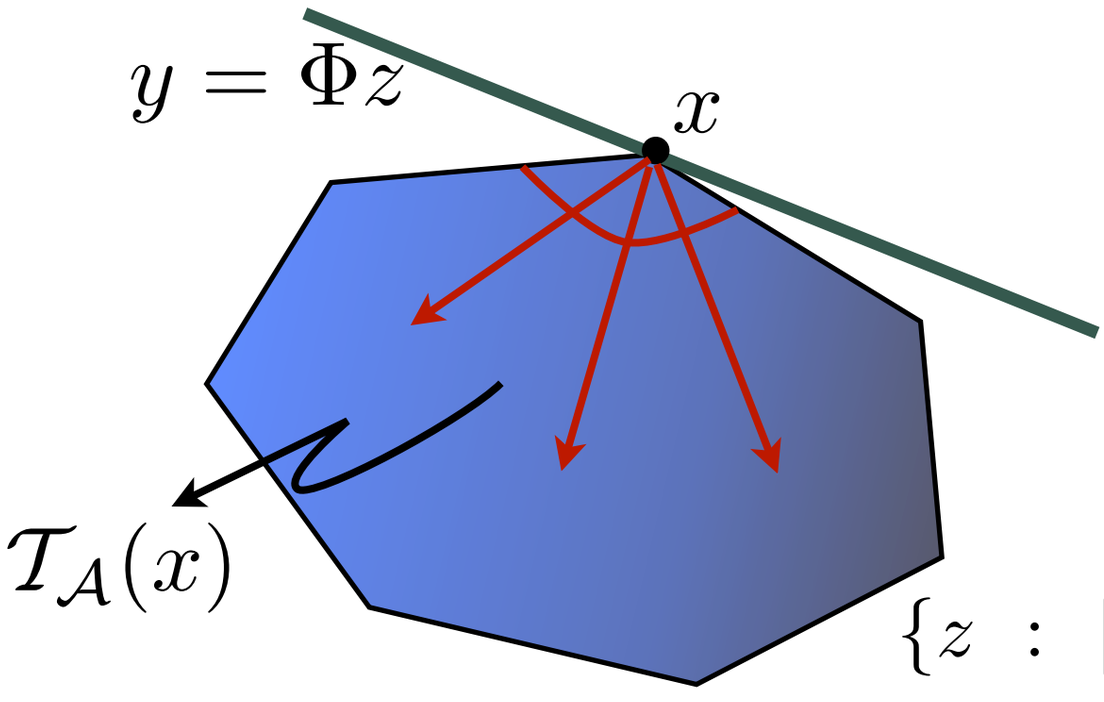
$$\|x\|_{\mathcal{A}} = \inf\{\sum_{a \in \mathcal{A}} |c_a| \ : \ x = \sum_{a \in \mathcal{A}} c_a a\}$$

IDEA:
$$\begin{array}{ll} \text{minimize} & \|z\|_{\mathcal{A}} \\ \text{subject to} & \Phi z = y \end{array}$$

- When does this work?
- How do we solve the optimization problem?

Tangent Cones

- Set of directions that decrease the norm from x form a cone:
  $\mathcal{T}_{\mathcal{A}}(x) = \{d \ : \ \|x + \alpha d\|_{\mathcal{A}} \leq \|x\|_{\mathcal{A}} \text{ for some } \alpha > 0\}$



$$\begin{aligned} \text{minimize} \quad & \|z\|_{\mathcal{A}} \\ \text{subject to} \quad & \Phi z = y \end{aligned}$$

$\{z \ : \ \|z\|_{\mathcal{A}} \leq \|x\|_{\mathcal{A}}\}$

- x is the unique minimizer if the intersection of this cone with the null space of Φ equals {0}

Gaussian Widths

- When does a random subspace, *U*, intersect a convex cone *C* at the origin?

- **Gordon 88:** with high probability if

$$\operatorname{codim}(U) \geq w(C)^2$$

- Where $w(C) = \mathbb{E}\left[\max_{x \in C \cap \mathbb{S}^{n-1}} \langle x, g \rangle\right]$ is the *Gaussian width.*

$$\boxed{g \sim \mathcal{N}(0, I_n)}$$

- **Corollary:** For inverse problems: if Φ is a random Gaussian matrix with m rows, need $m \geq w(\mathcal{T}_{\mathcal{A}}(x))^2$ for recovery of *x*.

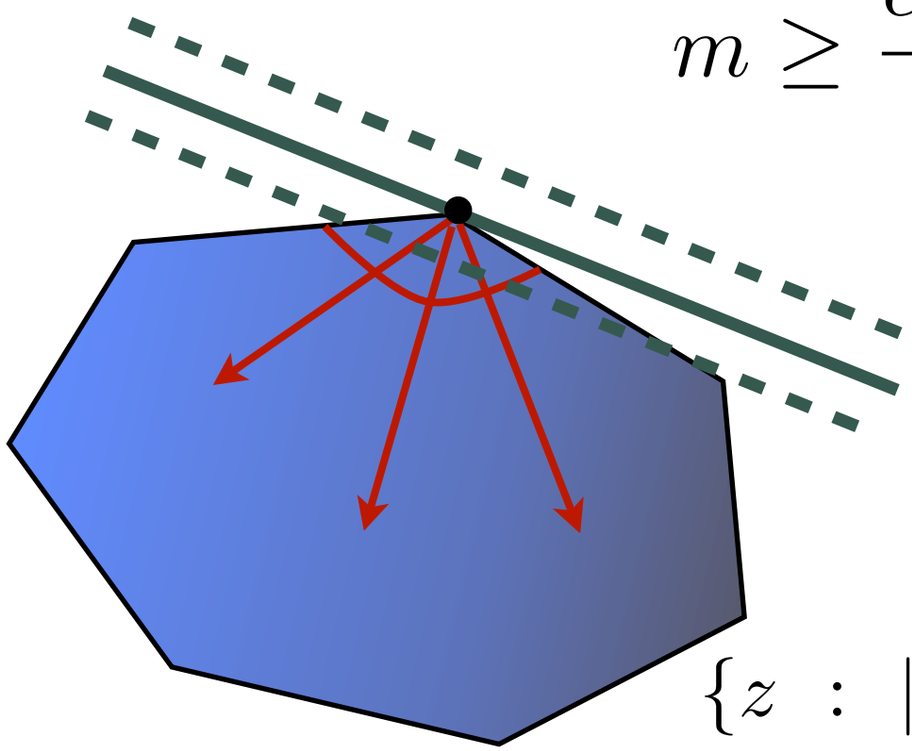
Robust Recovery

- Suppose we observe $y = \Phi x + w \qquad \|w\|_2 \leq \delta$

$$\begin{array}{ll} \text{minimize} & \|z\|_{\mathcal{A}} \\ \text{subject to} & \|\Phi z - y\| \leq \delta \end{array}$$

- If $\hat{x}$ is an optimal solution, then $\|x - \hat{x}\| \leq \dfrac{2\delta}{\epsilon}$ provided that

$$m \geq \frac{c_0 w(T_{\mathcal{A}}(x))^2}{(1-\epsilon)^2}$$

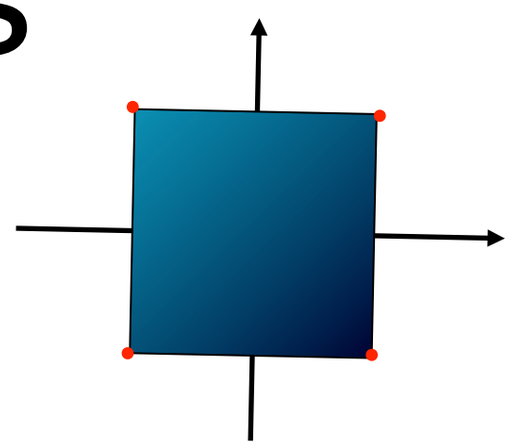


$\|\Phi z - y\| \leq \delta$

$\{z \ : \ \|z\|_{\mathcal{A}} \leq \|x\|_{\mathcal{A}}\}$

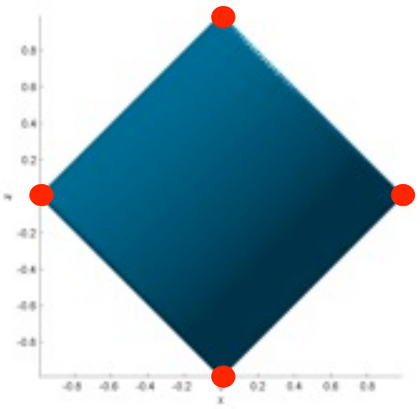
Calculating Widths

- Hypercube: $m \geq n/2$



- Sparse Vectors, n vector, sparsity s<0.25n

$$m \geq 2s\left(\log\left(\frac{n-s}{s}\right)+1\right)$$

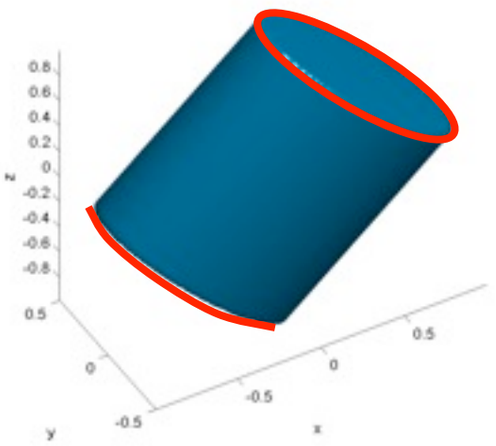


- Block sparse, M groups (possibly overlapping), maximum group size B, k active groups

$$m \geq 2k\left(\log\left(M-k\right)+B\right)+k$$

- Low-rank matrices: n_1 x n_2, (n_1<n_2), rank r

$$m \geq 3r(n_1+n_2-r)$$



General Cones

- **Theorem:** Let *C* be a nonempty cone with polar cone *C**. Suppose C* subtends normalized solid angle μ. Then

$$w(C) \leq 3\sqrt{\log\left(\frac{4}{\mu}\right)}$$

- **Corollary:** For a vertex-transitive (i.e., "symmetric") polytope with p vertices, O(log p) Gaussian measurements are sufficient to recover a vertex via convex optimization.

- For n x n permutation matrix: m = O(n log n)
- For n x n cut matrix: m = O(n)

Algorithms

$$\text{minimize}_z \quad \|\Phi z - y\|_2^2 + \mu \|z\|_{\mathcal{A}}$$

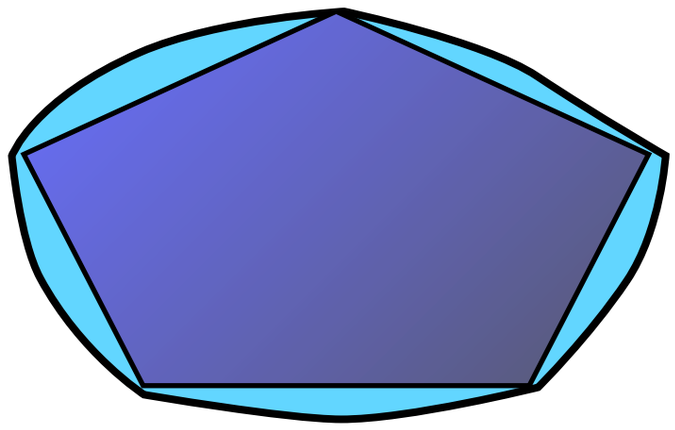
- Naturally amenable to projected gradient algorithm:

$$z_{k+1} = \Pi_{\eta\mu}(z_k - \eta \Phi^* r_k)$$

"shrinkage"

$$\Pi_\tau(z) = \arg\min_u \tfrac{1}{2}\|z - u\|^2 + \tau \|u\|_{\mathcal{A}}$$

- Relaxations: $\mathcal{A}_1 \subset \mathcal{A}_2 \implies \|x\|_{\mathcal{A}_2} \leq \|x\|_{\mathcal{A}_1}$



NB! tangent cone gets wider

- Hierarchy of relaxations based on *θ-Bodies* yield progressively tighter bounds on the atomic norm

Atomic Norm Decompositions

- Propose a natural convex heuristic for enforcing prior information in inverse problems

- Bounds for the linear case: heuristic succeeds for most sufficiently large sets of measurements

- Stability without restricted isometries

- Standard program for computing these bounds: distance to normal cones

- Algorithms and approximation schemes for computationally difficult priors

Extensions…

- Width Calculations for more general structures
- Recovery bounds for structured measurement matrices (application specific)
- Incorporating stochastic noise models
- Understanding of the loss due to convex relaxation and norm approximation
- Scaling generalized shrinkage algorithms to massive data sets

JELLYFISH

- SGD for Matrix Factorizations.

with Christopher Ré

Example: $\text{minimize} \quad \sum_{(u,v)\in E}(X_{uv} - M_{uv})^2 + \mu\|\mathbf{X}\|_*$

- Idea: approximate $\mathbf{X} \approx \mathbf{L}\mathbf{R}^T$

$$\text{minimize}_{(\mathbf{L},\mathbf{R})} \sum_{(u,v)\in E} \left\{(\mathbf{L}_u\mathbf{R}_v^T - Z_{uv})^2 + \mu_u\|\mathbf{L}_u\|_F^2 + \mu_v\|\mathbf{R}_v\|_F^2\right\}$$

- Step 1: Pick (*u*,*v*) and compute residual:

$$e = (\mathbf{L}_u\mathbf{R}_v^T - Z_{uv})$$

- Step 2: Take a gradient step:

$$\begin{bmatrix} \mathbf{L}_u \\ \mathbf{R}_v \end{bmatrix} \leftarrow \begin{bmatrix} (1-\gamma\mu_u)\mathbf{L}_u - \gamma e\mathbf{R}_v \\ (1-\gamma\mu_v)\mathbf{R}_v - \gamma e\mathbf{L}_u \end{bmatrix}$$

JELLYFISH

Observation: With replacement sample=poor locality
Idea: Bias sample to improve locality.

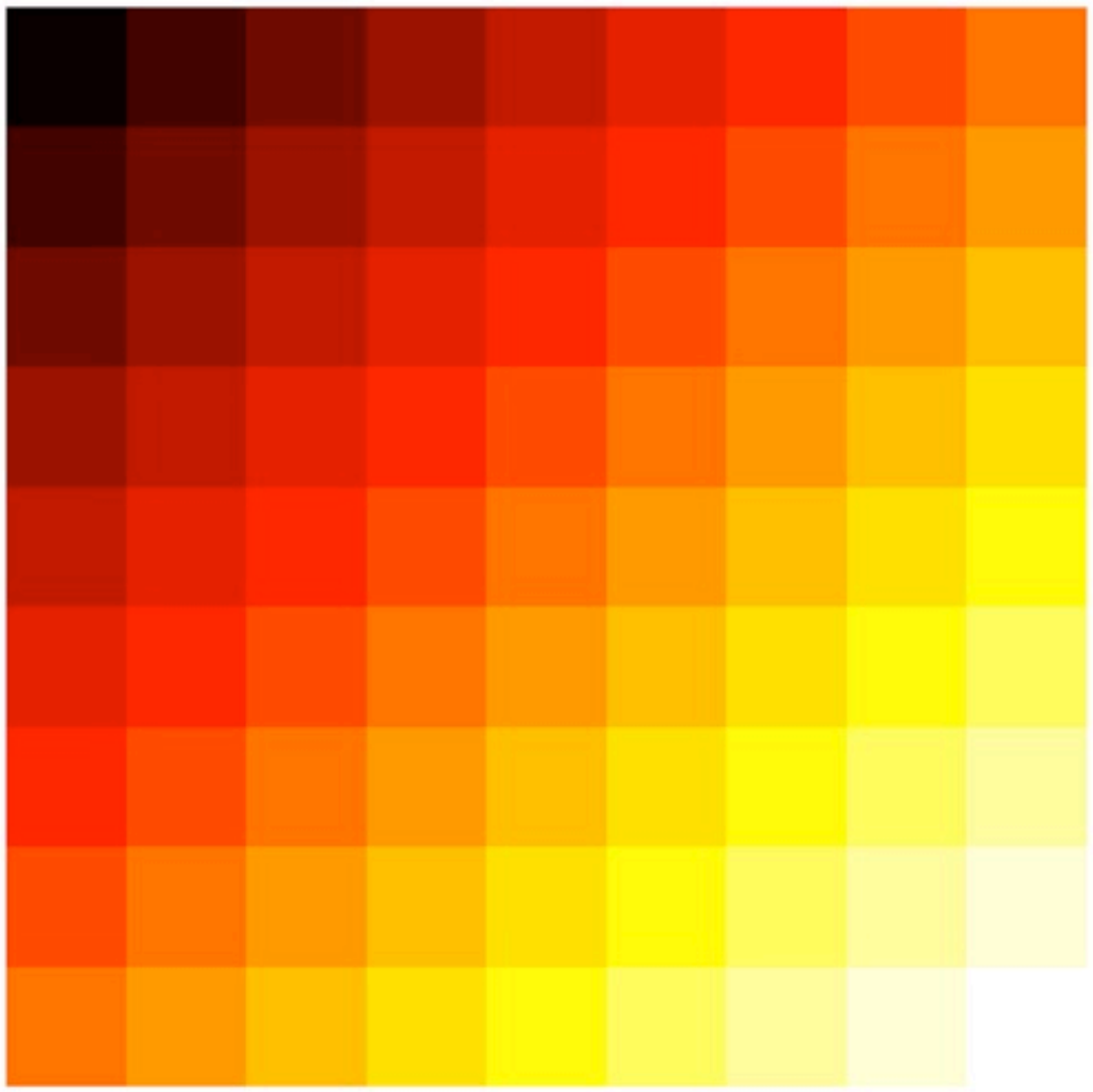
$$\begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} \begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} = \begin{bmatrix} L_1R_1 & L_1R_2 & L_1R_3 \\ L_2R_1 & L_2R_2 & L_2R_3 \\ L_3R_1 & L_3R_2 & L_3R_3 \end{bmatrix}$$

Algorithm: Shuffle the data.

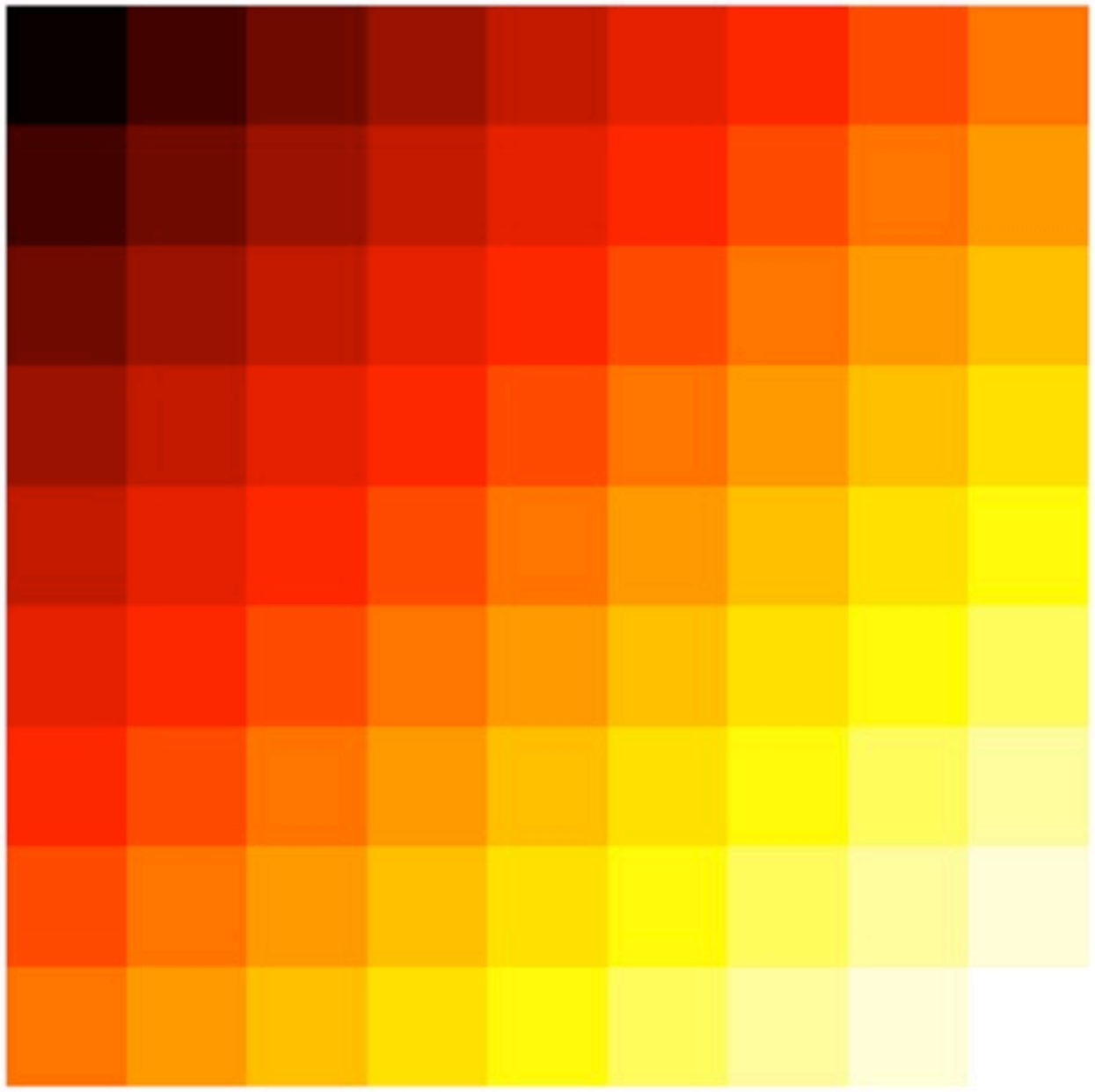
1. Process $\{L_1R_1, L_2R_2, L_3R_3\}$ in parallel
2. Process $\{L_1R_2, L_2R_3, L_3R_1\}$ in parallel
3. Process $\{L_1R_3, L_2R_1, L_3R_2\}$ in parallel

Big win: No locks!
(model access)

Example Optimization

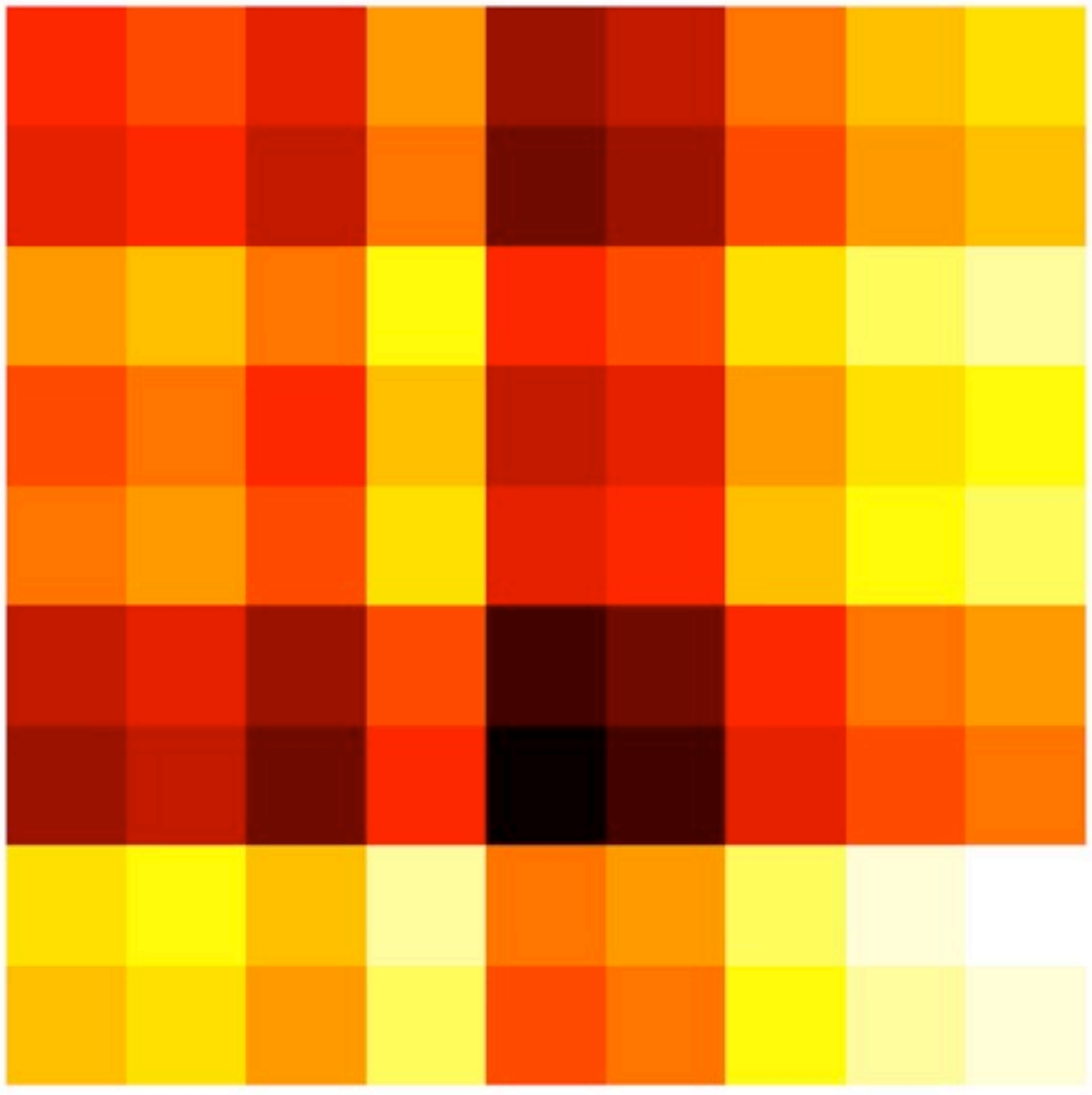


Example Optimization



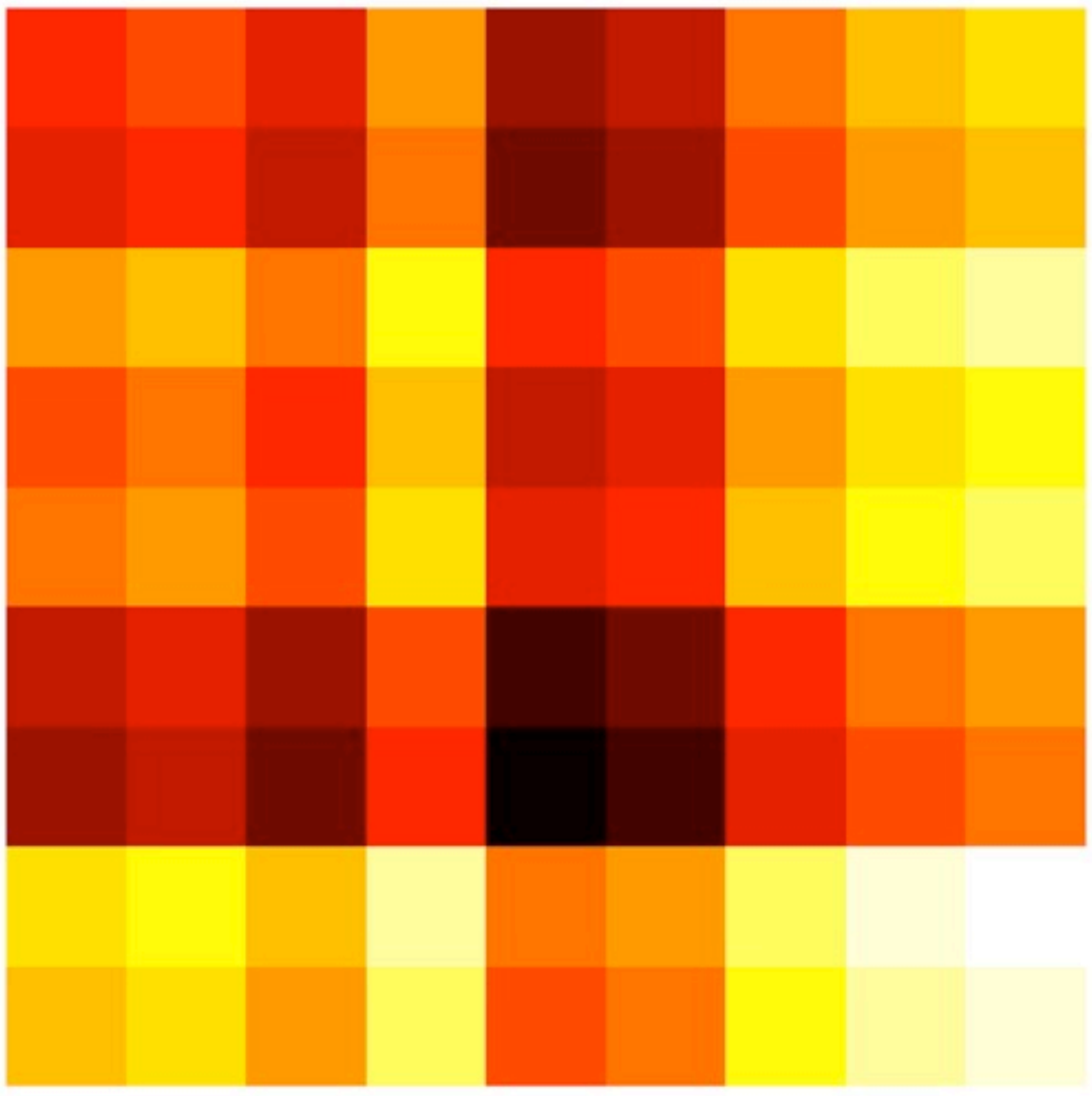
- Shuffle all the rows and columns

Example Optimization



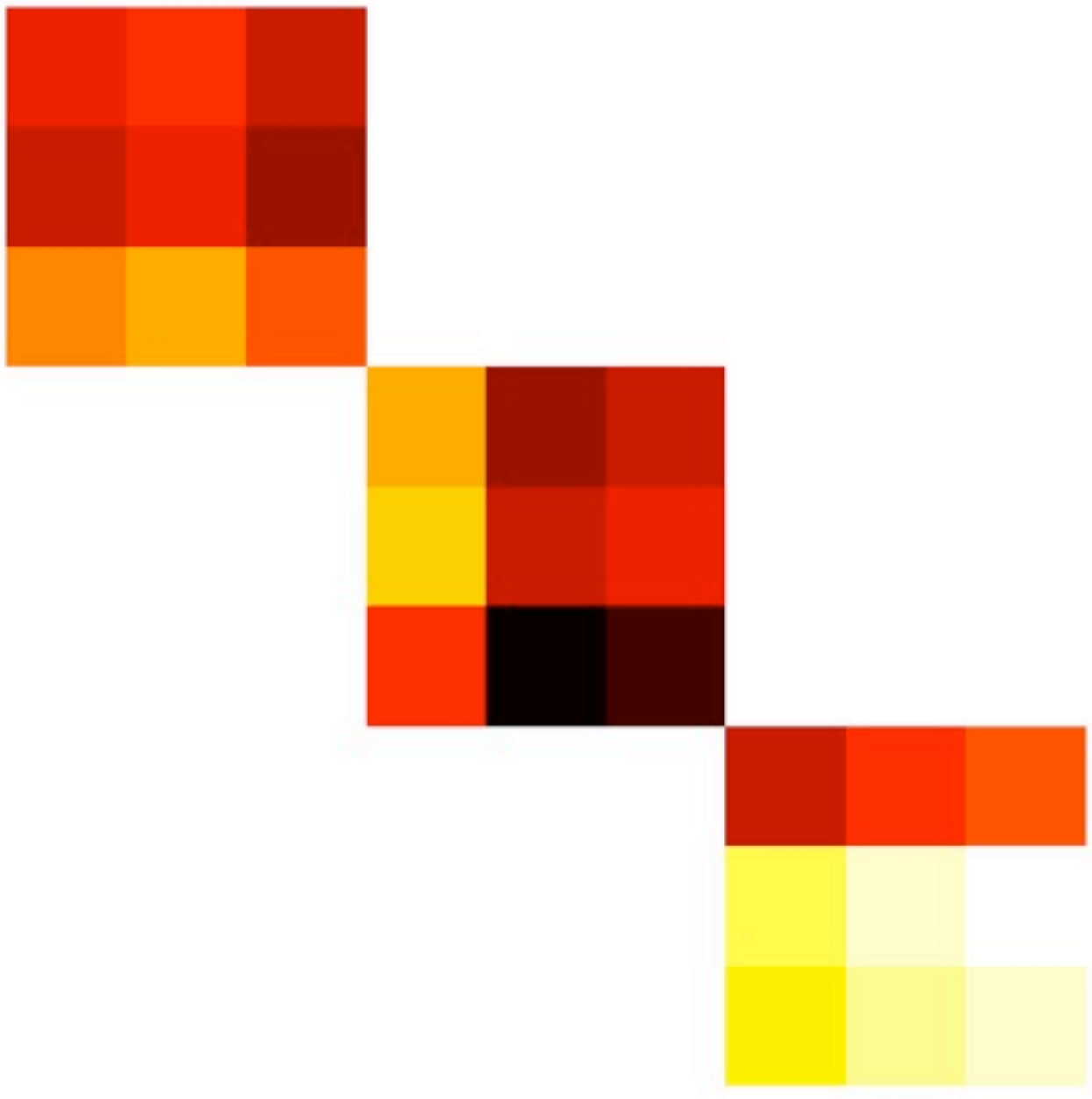
- Shuffle all the rows and columns

Example Optimization



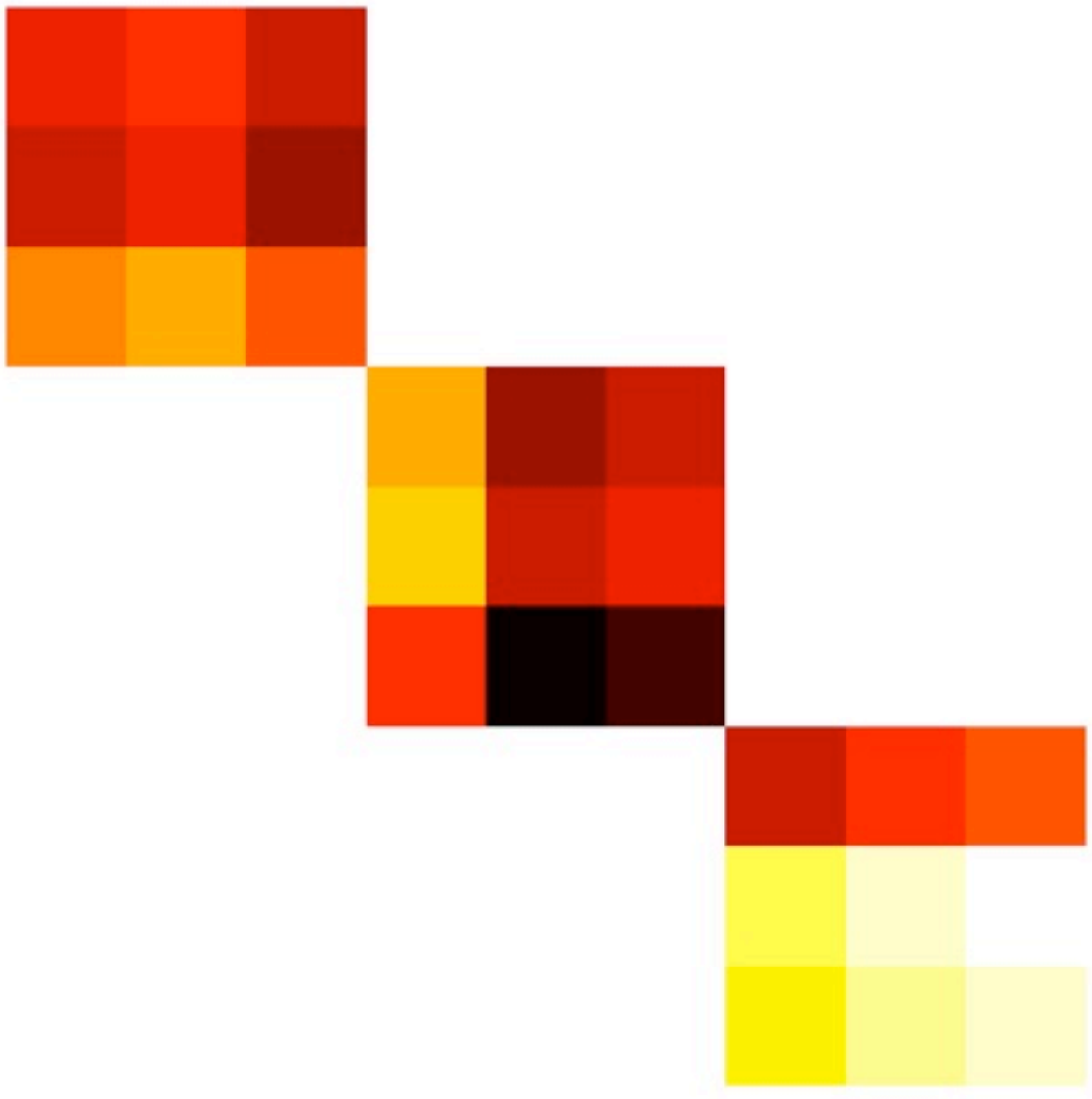
- Shuffle all the rows and columns
- Apply block partitioning

Example Optimization



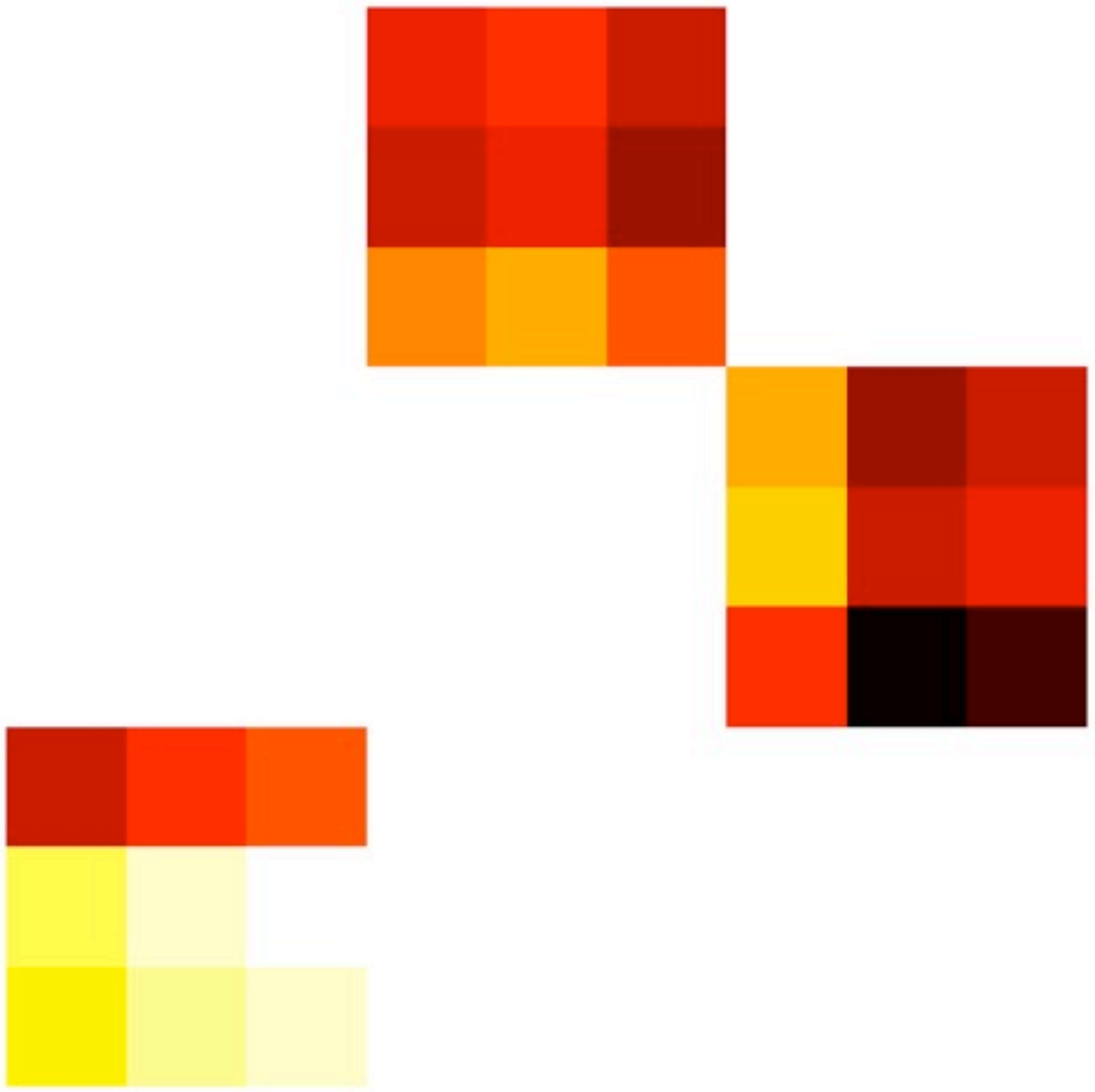
- Shuffle all the rows and columns
- Apply block partitioning

Example Optimization



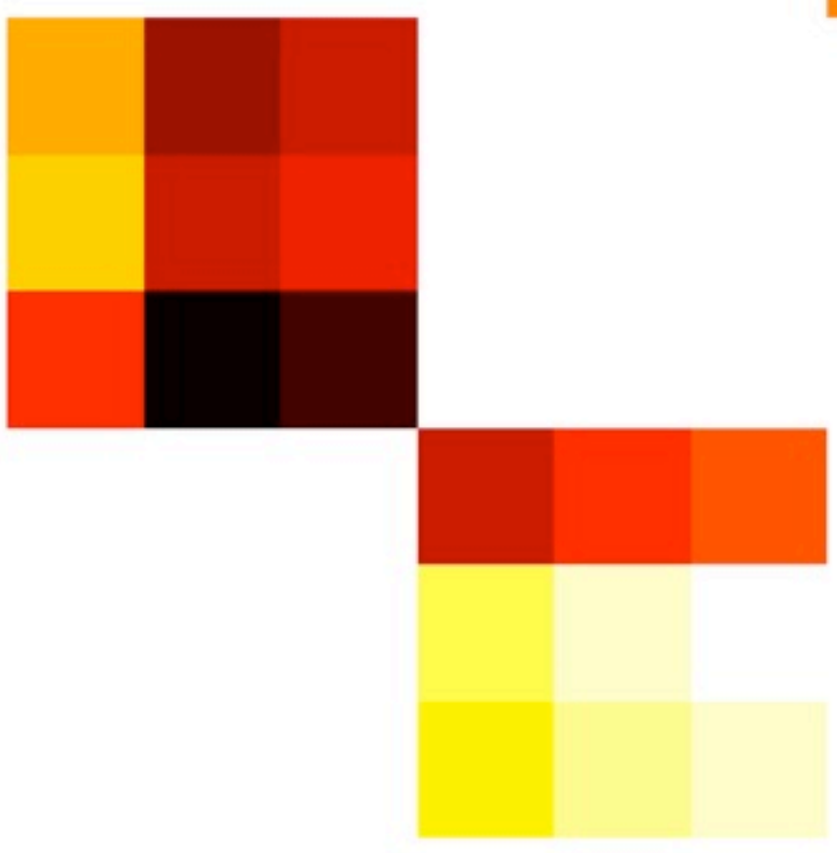
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently

Example Optimization



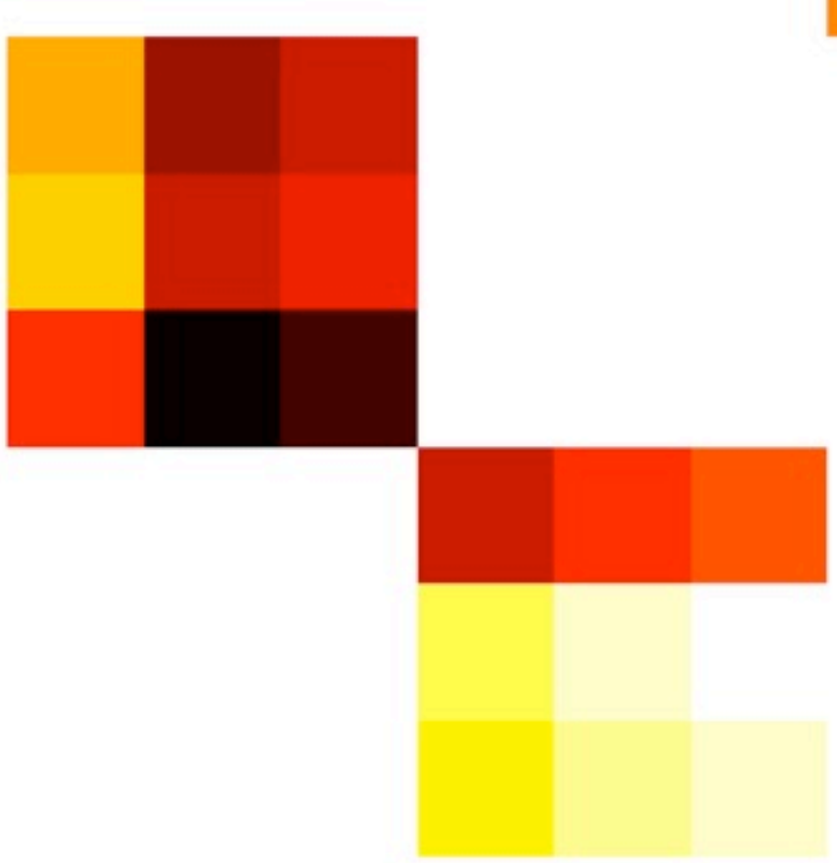
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently

Example Optimization



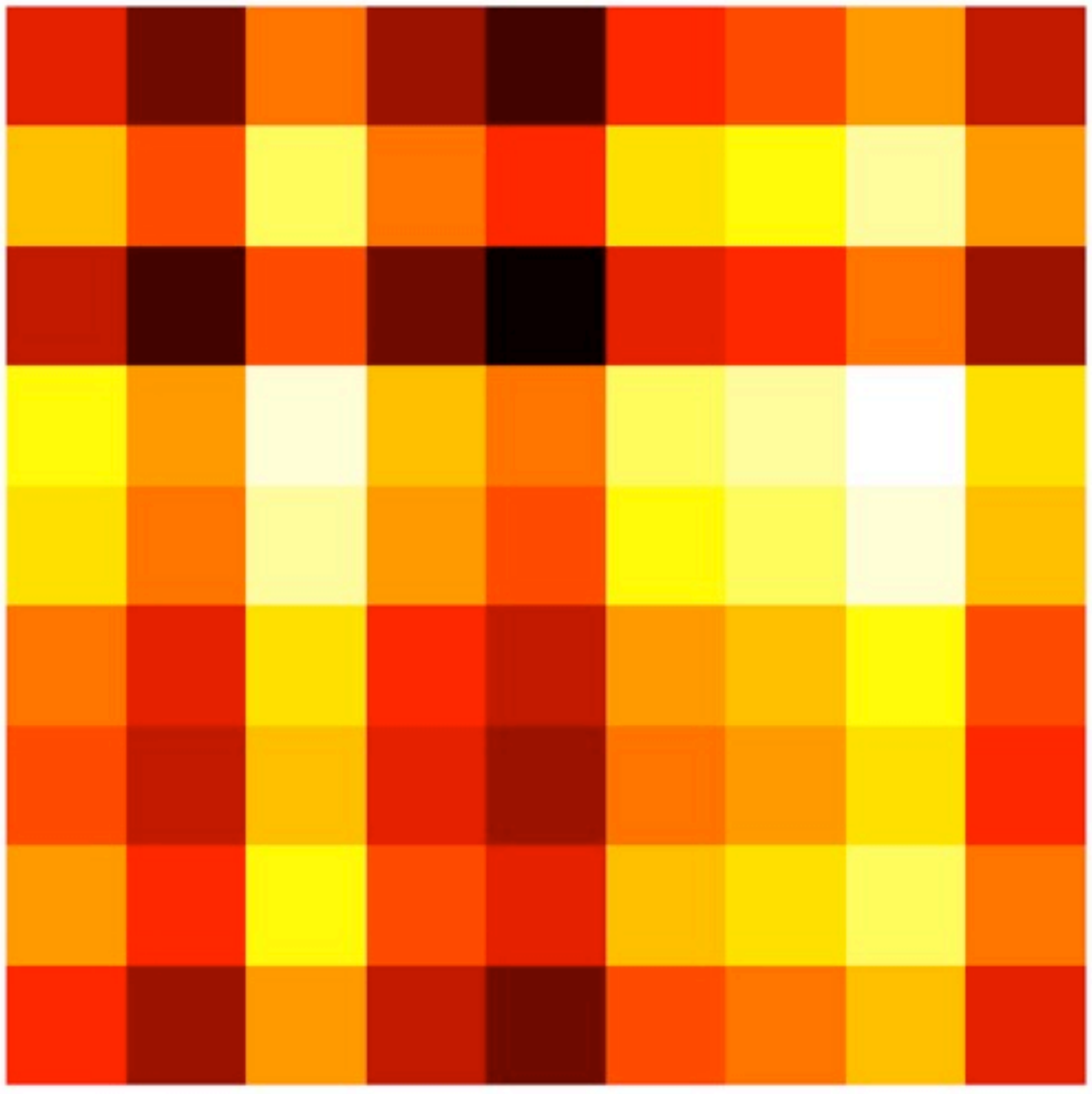
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently

Example Optimization



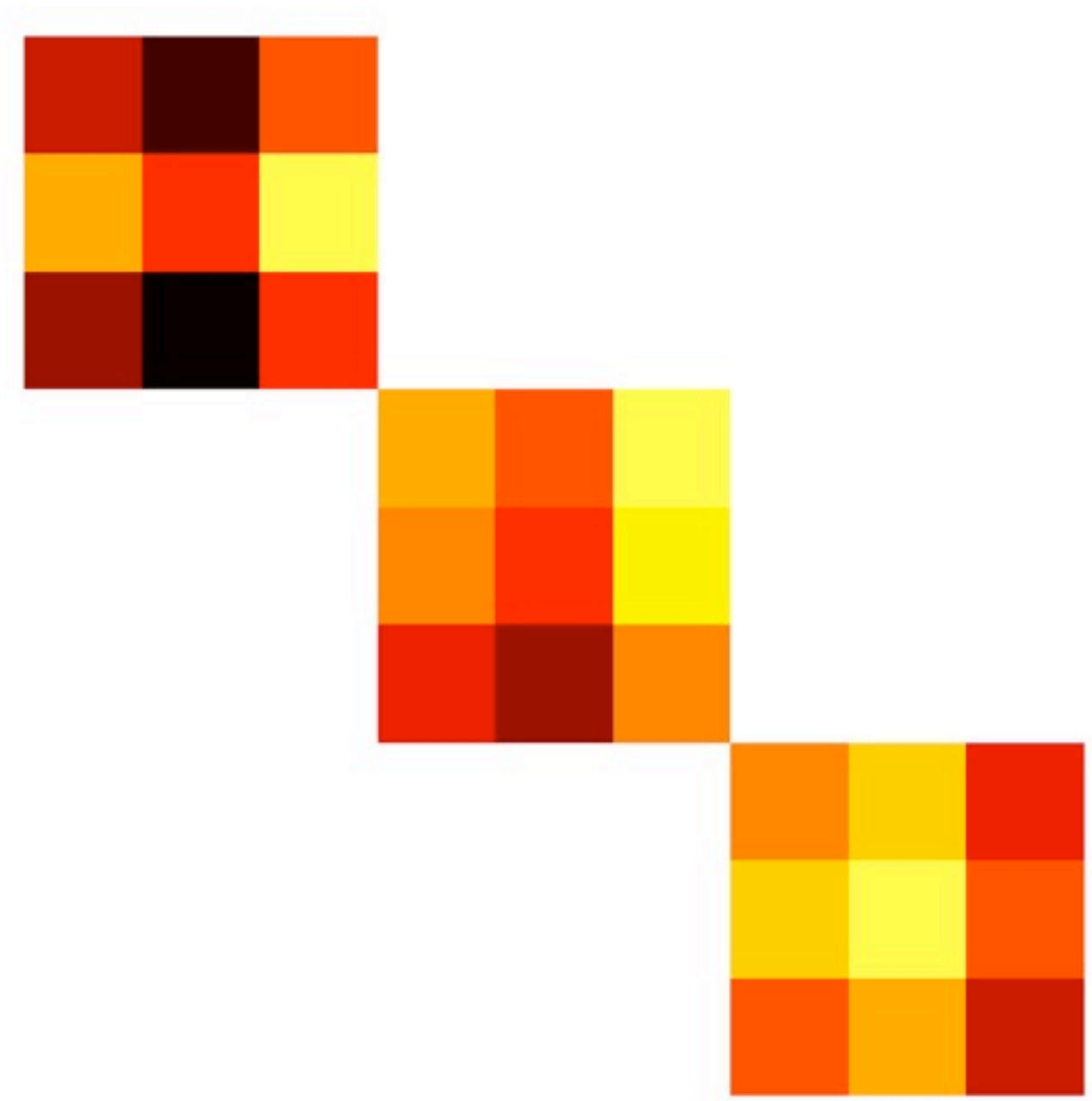
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...

Example Optimization



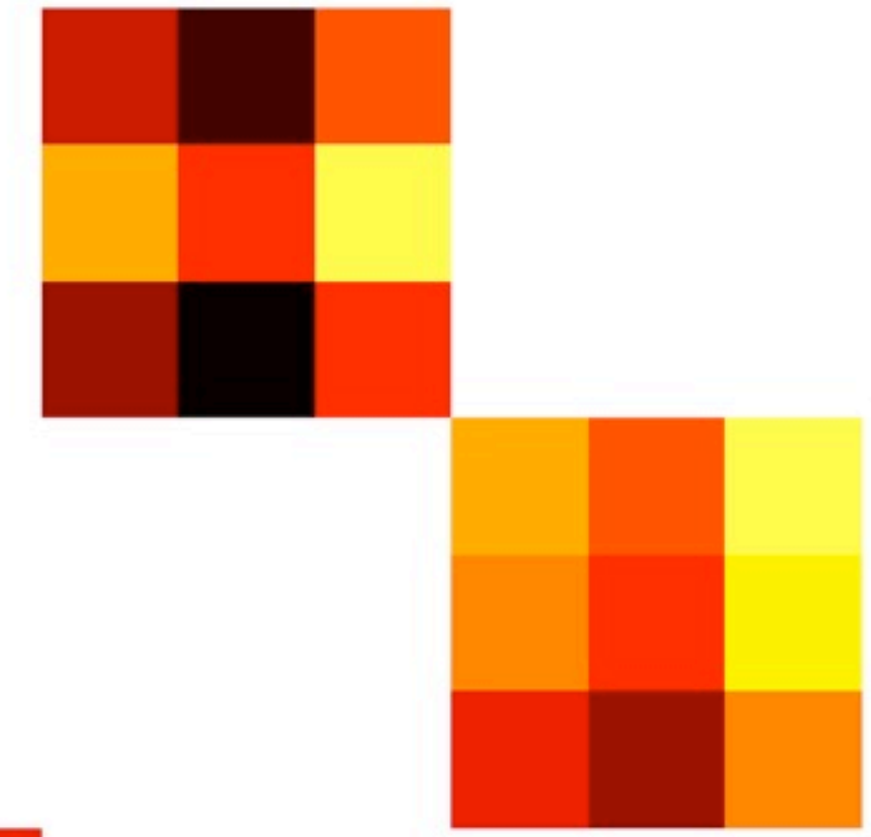
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...

Example Optimization



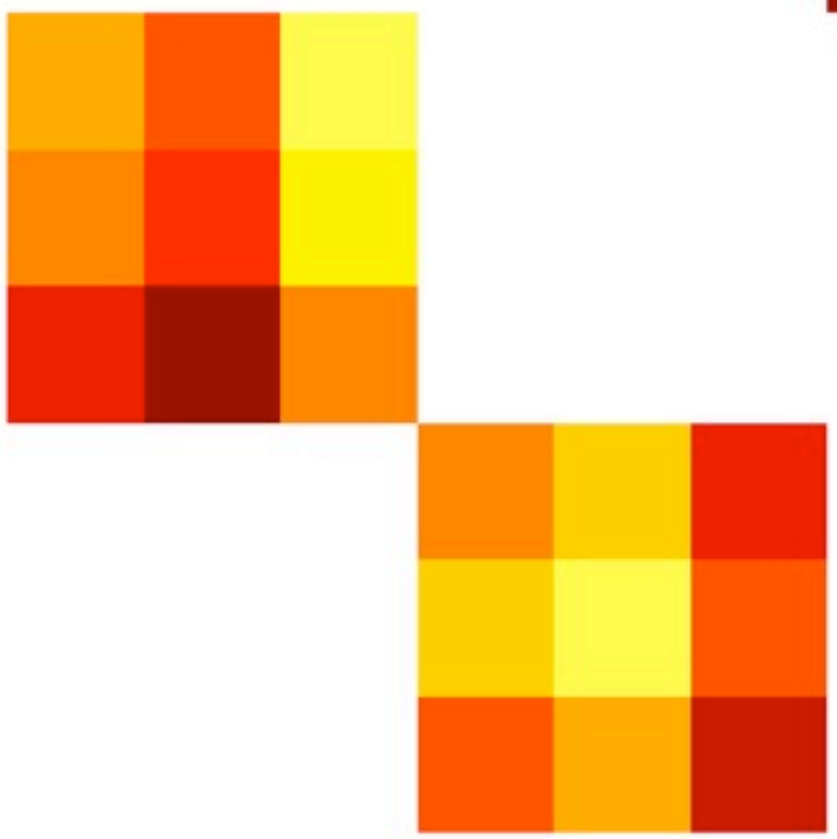
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...

Example Optimization



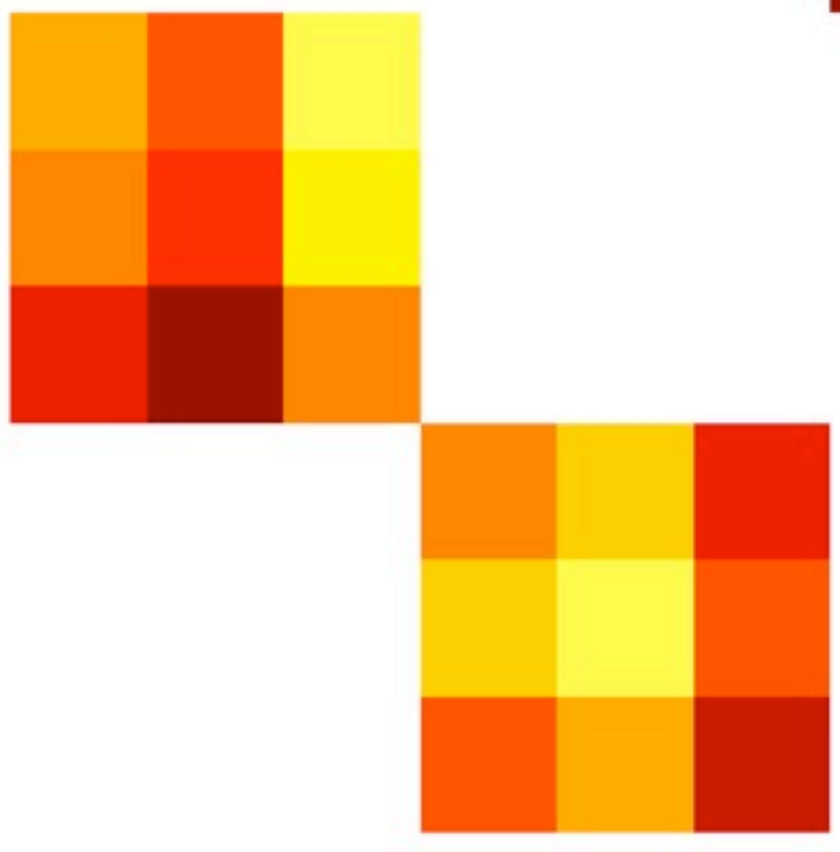
- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...

Example Optimization



- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...

Example Optimization



- Shuffle all the rows and columns
- Apply block partitioning
- Train on each block independently
- Repeat...
- Solves Netflix prize in under 1 minute on a 40 core machine
- Over 100x faster than standard solvers