Convex Structure Learning in Log-Linear Models: Beyond Pairwise Potentials

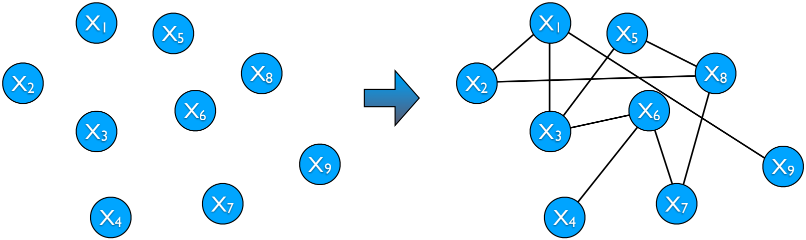
Mark Schmidt and Kevin Murphy

Department of Computer Science
University of British Columbia

May 15, 2010

Outline

Structure Learning with ℓ_1-Regularization



- Several authors have recently examined parameter estimation in graphical models with ℓ_1-regularization.
- Regularization and structure learning in a convex framework.
- First works looked at Gaussian graphical models.
- Recent works consider log-linear models of discrete data.

Structure Learning with ℓ_1-Regularization

For example, assume we have a pairwise undirected graphical model,

$$p(\mathbf{x}) \triangleq \frac{1}{Z} \prod_i \phi_i(x_i) \prod_{j>i} \phi_{ij}(x_i, x_j),$$

with node parameters $\mathbf{w}_i$ and edge parameters $\mathbf{w}_{ij}$.

Assume that $\mathbf{w}_{ij} = \mathbf{0}$ is equivalent to removing the edge (i,j).

We can use group ℓ_1-regularization for simultaneous parameter estimation and structure learning:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \lambda \sum_i \sum_{j>i} ||\mathbf{w}_{ij}||_2,$$

Structure Learning with ℓ_1-Regularization

For example, assume we have a pairwise undirected graphical model,

$$p(\mathbf{x}) \triangleq \frac{1}{Z}\prod_i \phi_i(x_i)\prod_{j>i}\phi_{ij}(x_i, x_j),$$

with node parameters $\mathbf{w}_i$ and edge parameters $\mathbf{w}_{ij}$.

Assume that $\mathbf{w}_{ij} = \mathbf{0}$ is equivalent to removing the edge (i, j).

We can use group ℓ_1-regularization for simultaneous parameter estimation and structure learning:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n}\log p(\mathbf{x}^i|\mathbf{w}) + \lambda\sum_i\sum_{j>i}||\mathbf{w}_{ij}||_2,$$

Structure Learning with ℓ_1-Regularization

For example, assume we have a pairwise undirected graphical model,

$$p(\mathbf{x}) \triangleq \frac{1}{Z}\prod_i \phi_i(x_i)\prod_{j>i}\phi_{ij}(x_i, x_j),$$

with node parameters $\mathbf{w}_i$ and edge parameters $\mathbf{w}_{ij}$.

Assume that $\mathbf{w}_{ij} = \mathbf{0}$ is equivalent to removing the edge (i, j).

We can use group ℓ_1-regularization for simultaneous parameter estimation and structure learning:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n}\log p(\mathbf{x}^i|\mathbf{w}) + \lambda\sum_i\sum_{j>i}||\mathbf{w}_{ij}||_2,$$

Structure Learning with ℓ_1-Regularization

A list of papers on this topic (incomplete):

[Li & Yang, 2004], [Li & Yang, 2005], [Banerjee et al., 2006], [Huang et al., 2006], [Lee et al., 2006], [Meinshausen & Bühlmann, 2006], [Wainwright et al., 2006], [Dahinden et al., 2007], [Schmidt et al., 2007], [Shimamura et al., 2007], [Yuan & Lin, 2007], [d' Aspremont et al., 2008], [Banerjee et al., 2008], [Dahl et al., 2008], [Duchi et al., 2008], [Friedman et al., 2008], [Kolar & Xing, 2008], [Levina et al., 2008], [Schmidt et al., 2008], [Fan & Feng, 2009], [Höling & Tibshirani, 2009], [Krishnamurphy & d'Aspremont, 2009], [Lu, 2009a], [Lu, 2009b], [Marlin et al., 2009a], [Marlin et al., 2009b], [Schmidt et al., 2009], [Schmidt & Murphy, 2009], [Schnitzspan et al., 2009], [Yuan, 2009], [Vidaurre et al., 2010].

Structure Learning with ℓ_1-Regularization

Many of these papers have made the pairwise assumption:

[Li & Yang, 2004], [Li & Yang, 2005], [Banerjee et al., 2006], [Huang et al., 2006], [Lee et al., 2006], [Meinshausen & Bühlmann, 2006], [Wainwright et al., 2006], [Dahinden et al., 2007], [Schmidt et al., 2007], [Shimamura et al., 2007], [Yuan & Lin, 2007], [d' Aspremont et al., 2008], [Banerjee et al., 2008], [Dahl et al., 2008], [Duchi et al., 2008], [Friedman et al., 2008], [Kolar & Xing, 2008], [Levina et al., 2008], [Schmidt et al., 2008], [Fan & Feng, 2009], [Höling & Tibshirani, 2009], [Krishnamurphy & d'Aspremont, 2009], [Lu, 2009a], [Lu, 2009b], [Marlin et al., 2009a], [Marlin et al., 2009b], [Schmidt et al., 2009], [Schmidt & Murphy, 2009], [Schnitzspan et al., 2009], [Yuan, 2009], [Vidaurre et al., 2010].

Structure Learning with ℓ_1-Regularization

Many of these papers have made the pairwise assumption:

[Li & Yang, 2004], [Li & Yang, 2005], [Banerjee et al., 2006], [Huang et al., 2006], [Lee et al., 2006], [Meinshausen & Bühlmann, 2006], [Wainwright et al., 2006], [Dahinden et al., 2007], [Schmidt et al., 2007], [Shimamura et al., 2007], [Yuan & Lin, 2007], [d' Aspremont et al., 2008], [Banerjee et al., 2008], [Dahl et al., 2008], [Duchi et al., 2008], [Friedman et al., 2008], [Kolar & Xing, 2008], [Levina et al., 2008], [Schmidt et al., 2008], [Fan & Feng, 2009], [Höling & Tibshirani, 2009], [Krishnamurphy & d'Aspremont, 2009], [Lu, 2009a], [Lu, 2009b], [Marlin et al., 2009a], [Marlin et al., 2009b], [Schmidt et al., 2009], [Schmidt & Murphy, 2009], [Schnitzspan et al., 2009], [Yuan, 2009], [Vidaurre et al., 2010].

Structure Learning with ℓ_1-Regularization

Many of these papers have made the pairwise assumption:

[Li & Yang, 2004], [Li & Yang, 2005], [Banerjee et al., 2006], [Huang et al., 2006], [Lee et al., 2006], [Meinshausen & Bühlmann, 2006], [Wainwright et al., 2006], [Dahinden et al., 2007], [Schmidt et al., 2007], [Shimamura et al., 2007], [Yuan & Lin, 2007], [d' Aspremont et al., 2008], [Banerjee et al., 2008], [Dahl et al., 2008], [Duchi et al., 2008], [Friedman et al., 2008], [Kolar & Xing, 2008], [Levina et al., 2008], [Schmidt et al., 2008], [Fan & Feng, 2009], [Höling & Tibshirani, 2009], [Krishnamurphy & d'Aspremont, 2009], [Lu, 2009a], [Lu, 2009b], [Marlin et al., 2009a], [Marlin et al., 2009b], [Schmidt et al., 2009], [Schmidt & Murphy, 2009], [Schnitzspan et al., 2009], [Yuan, 2009], [Vidaurre et al., 2010].

Structure Learning with ℓ_1-Regularization

Many of these papers have made the pairwise assumption:

[Li & Yang, 2004], [Li & Yang, 2005], [Banerjee et al., 2006], [Huang et al., 2006], [Lee et al., 2006], [Meinshausen & Bühlmann, 2006], [Wainwright et al., 2006], [Dahinden et al., 2007], [Schmidt et al., 2007], [Shimamura et al., 2007], [Yuan & Lin, 2007], [d' Aspremont et al., 2008], [Banerjee et al., 2008], [Dahl et al., 2008], [Duchi et al., 2008], [Friedman et al., 2008], [Kolar & Xing, 2008], [Levina et al., 2008], [Schmidt et al., 2008], [Fan & Feng, 2009], [Höling & Tibshirani, 2009], [Krishnamurphy & d'Aspremont, 2009], [Lu, 2009a], [Lu, 2009b], [Marlin et al., 2009a], [Marlin et al., 2009b], [Schmidt et al., 2009], [Schmidt & Murphy, 2009], [Schnitzspan et al., 2009], [Yuan, 2009], [Vidaurre et al., 2010].

Our Contribution

- The pairwise assumption is inherent to Gaussian models.
- The pairwise assumption has not traditionally been associated with log-linear models [Goodman, 1971], [Bishop et al., 1975].
- The assumption is restrictive if higher-order statistics matter.
- Eg. Mutations in both gene A and gene B lead to cancer.
- This work gives give one way to go beyond pairwise potentials.

Our Contribution

- The pairwise assumption is inherent to Gaussian models.
- The pairwise assumption has not traditionally been associated with log-linear models [Goodman, 1971], [Bishop et al., 1975].
- The assumption is restrictive if higher-order statistics matter.
- Eg. Mutations in both gene A and gene B lead to cancer.
- This work gives give one way to go beyond pairwise potentials.

Our Contribution

- The pairwise assumption is inherent to Gaussian models.
- The pairwise assumption has not traditionally been associated with log-linear models [Goodman, 1971], [Bishop et al., 1975].
- The assumption is restrictive if higher-order statistics matter.
- Eg. Mutations in both gene A and gene B lead to cancer.
- This work gives give one way to go beyond pairwise potentials.

Our Contribution

- The pairwise assumption is inherent to Gaussian models.
- The pairwise assumption has not traditionally been associated with log-linear models [Goodman, 1971], [Bishop et al., 1975].
- The assumption is restrictive if higher-order statistics matter.
- Eg. Mutations in both gene A and gene B lead to cancer.
- This work gives give one way to go beyond pairwise potentials.

Our Contribution

The challenge in going beyond pairwise potentials is the exponential number of possible higher-order potentials:

- We consider the special case of hierarchical log-linear models.
- We give a convex formulation that utilizes overlapping group ℓ_1-regularization to enforce the hierarchy.
- We give an active set method that rules out non-hierarchical higher-order potentials.
- We use projected gradient methods and Dykstra's cyclic projection algorithm to optimize with respect to the active set.

Our Contribution

The challenge in going beyond pairwise potentials is the exponential number of possible higher-order potentials:

- We consider the special case of hierarchical log-linear models.
- We give a convex formulation that utilizes overlapping group ℓ_1-regularization to enforce the hierarchy.
- We give an active set method that rules out non-hierarchical higher-order potentials.
- We use projected gradient methods and Dykstra's cyclic projection algorithm to optimize with respect to the active set.

Our Contribution

The challenge in going beyond pairwise potentials is the exponential number of possible higher-order potentials:

- We consider the special case of hierarchical log-linear models.
- We give a convex formulation that utilizes overlapping group ℓ_1-regularization to enforce the hierarchy.
- We give an active set method that rules out non-hierarchical higher-order potentials.
- We use projected gradient methods and Dykstra's cyclic projection algorithm to optimize with respect to the active set.

Our Contribution

The challenge in going beyond pairwise potentials is the exponential number of possible higher-order potentials:

- We consider the special case of hierarchical log-linear models.
- We give a convex formulation that utilizes overlapping group ℓ_1-regularization to enforce the hierarchy.
- We give an active set method that rules out non-hierarchical higher-order potentials.
- We use projected gradient methods and Dykstra's cyclic projection algorithm to optimize with respect to the active set.

Our Contribution

The challenge in going beyond pairwise potentials is the exponential number of possible higher-order potentials:

- We consider the special case of hierarchical log-linear models.
- We give a convex formulation that utilizes overlapping group ℓ_1-regularization to enforce the hierarchy.
- We give an active set method that rules out non-hierarchical higher-order potentials.
- We use projected gradient methods and Dykstra's cyclic projection algorithm to optimize with respect to the active set.

Outline

General Log-Linear Models

In log-linear models [Bishop et al., 1975] we write the probability of a vector $\mathbf{x} \in \{1, 2, \ldots, k\}^p$ as a normalized product

$$p(\mathbf{x}) \triangleq \frac{1}{Z} \prod_{A \subseteq S} \phi_A(\mathbf{x}_A),$$

over each subset A of $S \triangleq \{1, 2, \ldots, p\}$,

We consider a full parameterization of these potential functions, and a more parsimonious weighted Ising parameterization.

General Log-Linear Models

In log-linear models [Bishop et al., 1975] we write the probability of a vector $\mathbf{x} \in \{1, 2, \dots, k\}^p$ as a normalized product

$$p(\mathbf{x}) \triangleq \frac{1}{Z} \prod_{A \subseteq S} \phi_A(\mathbf{x}_A),$$

over each subset A of $S \triangleq \{1, 2, \dots, p\}$,

We consider a full parameterization of these potential functions, and a more parsimonious weighted Ising parameterization.

General Log-Linear Models

The full parameterization for a threeway potential on binary nodes,

$$\begin{aligned}\log \phi_{ijk}(\mathbf{x}_{ijk}) = &\mathbb{I}(x_i = 1, x_j = 1, x_k = 1)w_{ijk111} + \mathbb{I}(x_i = 1, x_j = 1, x_k = 2)w_{ijk112} \\ &+ \mathbb{I}(x_i = 1, x_j = 2, x_k = 1)w_{ijk121} + \mathbb{I}(x_i = 1, x_j = 2, x_k = 2)w_{ijk122} \\ &+ \mathbb{I}(x_i = 2, x_j = 1, x_k = 1)w_{ijk211} + \mathbb{I}(x_i = 2, x_j = 1, x_k = 2)w_{ijk212} \\ &+ \mathbb{I}(x_i = 2, x_j = 2, x_k = 1)w_{ijk221} + \mathbb{I}(x_i = 2, x_j = 2, x_k = 2)w_{ijk222}.\end{aligned}$$

$\phi_A(\mathbf{x}_A)$ has $k^{|A|}$ parameters $\mathbf{w}_A$.

Setting $\mathbf{w}_A = \mathbf{0}$ is equivalent to removing the potential.

In pairwise models we assume $\mathbf{w}_A = \mathbf{0}$ if $|A| > 2$.

General Log-Linear Models

The full parameterization for a threeway potential on binary nodes,

$$\begin{aligned}\log \phi_{ijk}(\mathbf{x}_{ijk}) &= \mathbb{I}(x_i = 1, x_j = 1, x_k = 1)w_{ijk111} + \mathbb{I}(x_i = 1, x_j = 1, x_k = 2)w_{ijk112} \\ &+ \mathbb{I}(x_i = 1, x_j = 2, x_k = 1)w_{ijk121} + \mathbb{I}(x_i = 1, x_j = 2, x_k = 2)w_{ijk122} \\ &+ \mathbb{I}(x_i = 2, x_j = 1, x_k = 1)w_{ijk211} + \mathbb{I}(x_i = 2, x_j = 1, x_k = 2)w_{ijk212} \\ &+ \mathbb{I}(x_i = 2, x_j = 2, x_k = 1)w_{ijk221} + \mathbb{I}(x_i = 2, x_j = 2, x_k = 2)w_{ijk222}.\end{aligned}$$

$\phi_A(\mathbf{x}_A)$ has $k^{|A|}$ parameters $\mathbf{w}_A$.

Setting $\mathbf{w}_A = \mathbf{0}$ is equivalent to removing the potential.

In pairwise models we assume $\mathbf{w}_A = \mathbf{0}$ if $|A| > 2$.

General Log-Linear Models

The full parameterization for a threeway potential on binary nodes,

$$\begin{aligned}\log \phi_{ijk}(\mathbf{x}_{ijk}) &= \mathbb{I}(x_i = 1, x_j = 1, x_k = 1)w_{ijk111} + \mathbb{I}(x_i = 1, x_j = 1, x_k = 2)w_{ijk112} \\ &+ \mathbb{I}(x_i = 1, x_j = 2, x_k = 1)w_{ijk121} + \mathbb{I}(x_i = 1, x_j = 2, x_k = 2)w_{ijk122} \\ &+ \mathbb{I}(x_i = 2, x_j = 1, x_k = 1)w_{ijk211} + \mathbb{I}(x_i = 2, x_j = 1, x_k = 2)w_{ijk212} \\ &+ \mathbb{I}(x_i = 2, x_j = 2, x_k = 1)w_{ijk221} + \mathbb{I}(x_i = 2, x_j = 2, x_k = 2)w_{ijk222}.\end{aligned}$$

$\phi_A(\mathbf{x}_A)$ has $k^{|A|}$ parameters $\mathbf{w}_A$.

Setting $\mathbf{w}_A = \mathbf{0}$ is equivalent to removing the potential.

In pairwise models we assume $\mathbf{w}_A = \mathbf{0}$ if $|A| > 2$.

Group ℓ_1-Regularization for General Log-Linear Models

We can extend the work on pairwise models to the general case by solving [Dahinden et al., 2007]:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A||_2,$$

However,

- Sparsity in the groups A does not correspond to conditional independence.
- Without a cardinality restriction, we have an exponential number of variables.

Group ℓ_1-Regularization for General Log-Linear Models

We can extend the work on pairwise models to the general case by solving [Dahinden et al., 2007]:

$$\min_{\mathbf{w}} - \sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A||_2,$$

However,

- Sparsity in the groups A does not correspond to conditional independence.
- Without a cardinality restriction, we have an exponential number of variables.

Hierarchical Log-Linear Models

Instead of using a cardinality restriction, we use:

Hierarchical Inclusion Restriction:
If $\mathbf{w}_A = \mathbf{0}$ and $A \subset B$, then $\mathbf{w}_B = \mathbf{0}$.

We can only have $(1,2,3)$ if we also have $(1,2)$, $(1,3)$, and $(2,3)$.

Hierarchical Log-Linear Models

Instead of using a cardinality restriction, we use:

Hierarchical Inclusion Restriction:
If $\mathbf{w}_A = \mathbf{0}$ and $A \subset B$, then $\mathbf{w}_B = \mathbf{0}$.

We can only have $(1, 2, 3)$ if we also have $(1, 2)$, $(1, 3)$, and $(2, 3)$.

Hierarchical Log-Linear Models

- This is the well-known class of hierarchical log-linear models [Bishop et al., 1975].
- Much larger than the set of pairwise models
- Group-sparsity corresponds to conditional independence.
- However, we can't enforce the hierarchical constraint with (disjoint) group ℓ_1-regularization.

Hierarchical Log-Linear Models

- This is the well-known class of hierarchical log-linear models [Bishop et al., 1975].
- Much larger than the set of pairwise models
- Group-sparsity corresponds to conditional independence.
- However, we can't enforce the hierarchical constraint with (disjoint) group ℓ_1-regularization.

Hierarchical Log-Linear Models

- This is the well-known class of hierarchical log-linear models [Bishop et al., 1975].
- Much larger than the set of pairwise models
- Group-sparsity corresponds to conditional independence.
- However, we can't enforce the hierarchical constraint with (disjoint) group ℓ_1-regularization.

Hierarchical Log-Linear Models

- This is the well-known class of hierarchical log-linear models [Bishop et al., 1975].
- Much larger than the set of pairwise models
- Group-sparsity corresponds to conditional independence.
- However, we can't enforce the hierarchical constraint with (disjoint) group ℓ_1-regularization.

Overlapping Group ℓ_1-Regularization for Hierarchical Constraints

Bach [2008], Zhao et al. [2009] enforce hierarchical inclusion restrictions with overlapping group ℓ_1-regularization.

Example:

- We can enforce that B is zero whenever A is zero by using two groups: $\{B\}$ and $\{A, B\}$.
- The resulting regularizer is $\lambda_B||\mathbf{w}_B||_2 + \lambda_{A,B}||\mathbf{w}_{A,B}||_2$

Overlapping Group ℓ_1-Regularization for Hierarchical Constraints

Bach [2008], Zhao et al. [2009] enforce hierarchical inclusion restrictions with overlapping group ℓ_1-regularization.

Example:

- We can enforce that B is zero whenever A is zero by using two groups: $\{B\}$ and $\{A, B\}$.
- The resulting regularizer is $\lambda_B ||\mathbf{w}_B||_2 + \lambda_{A,B} ||\mathbf{w}_{A,B}||_2$

Overlapping Group ℓ_1-Regularization for Hierarchical Log-Linear Models

We can learn hierarchical log-linear models by solving

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A \Big(\sum_{\{B | A \subseteq B\}} ||\mathbf{w}_B||_2^2 \Big)^{1/2}.$$

Under reasonable assumptions a minimizer of this convex optimization problem will satisfy hierarchical inclusion.

A nicer way to write this:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A^*||_2.$$

Overlapping Group ℓ_1-Regularization for Hierarchical Log-Linear Models

We can learn hierarchical log-linear models by solving

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A \Big(\sum_{\{B|A\subseteq B\}} ||\mathbf{w}_B||_2^2 \Big)^{1/2}.$$

Under reasonable assumptions a minimizer of this convex optimization problem will satisfy hierarchical inclusion.

A nicer way to write this:

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A^*||_2.$$

Outline

Active Set Method

- We want to avoid considering the exponential number of possible higher-order potentials.
- We know the solution will be hierarchical, so we propose to only consider groups that satisfy hierarchical inclusion.
- The resulting method guarantees a weak form of global optimality.

Active Set Method

- We want to avoid considering the exponential number of possible higher-order potentials.
- We know the solution will be hierarchical, so we propose to only consider groups that satisfy hierarchical inclusion.
- The resulting method guarantees a weak form of global optimality.

Active Set Method

- We want to avoid considering the exponential number of possible higher-order potentials.
- We know the solution will be hierarchical, so we propose to only consider groups that satisfy hierarchical inclusion.
- The resulting method guarantees a weak form of global optimality.

Active, Inactive, Boundary Groups

- We call A an active group if A or some superset of A is non-zero.
- If A is not active, and some subset of A is zero, we call A an inactive group.
- The remaining groups are called boundary group.
- Boundary groups can be made non-zero without violating hierarchical inclusion.

Active, Inactive, Boundary Groups

- We call A an active group if A or some superset of A is non-zero.
- If A is not active, and some subset of A is zero, we call A an inactive group.
- The remaining groups are called boundary group.
- Boundary groups can be made non-zero without violating hierarchical inclusion.

Active, Inactive, Boundary Groups

- We call A an active group if A or some superset of A is non-zero.
- If A is not active, and some subset of A is zero, we call A an inactive group.
- The remaining groups are called boundary group.
- Boundary groups can be made non-zero without violating hierarchical inclusion.

Active, Inactive, Boundary Groups

- We call A an active group if A or some superset of A is non-zero.
- If A is not active, and some subset of A is zero, we call A an inactive group.
- The remaining groups are called boundary group.
- Boundary groups can be made non-zero without violating hierarchical inclusion.

Optimality of Boundary Groups

With inactive groups fixed, the optimality conditions with respect to a boundary group A are

$$||\nabla_{\mathbf{w}_A} \sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w})||_2 \leq \lambda_A.$$

If the gradient is 0 for active groups:

- These are necessary and sufficient optimality conditions if inactive groups are fixed.
- They are necessary conditions of global optimality.

Optimality of Boundary Groups

With inactive groups fixed, the optimality conditions with respect to a boundary group A are

$$||\nabla_{\mathbf{w}_A} \sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w})||_2 \leq \lambda_A.$$

If the gradient is 0 for active groups:

- These are necessary and sufficient optimality conditions if inactive groups are fixed.
- They are necessary conditions of global optimality.

Active Set Method

Similar to Bach [2008], we use an active set method:

- Find the set of active groups, and the boundary groups violating the necessary conditions.
- Solve the problem with respect to these variables.

This adds groups that satisfy hierarchical inclusion, and where the model poorly estimates the higher-moment in the data.

(analogous to the greedy method of [Gevarter, 1987] for fitting maximum entropy distributions subject to marginal constraints [Cheeseman, 1983]).

Active Set Method

Similar to Bach [2008], we use an active set method:

- Find the set of active groups, and the boundary groups violating the necessary conditions.
- Solve the problem with respect to these variables.

This adds groups that satisfy hierarchical inclusion, and where the model poorly estimates the higher-moment in the data.

(analogous to the greedy method of [Gevarter, 1987] for fitting maximum entropy distributions subject to marginal constraints [Cheeseman, 1983]).

Active Set Method

Similar to Bach [2008], we use an active set method:

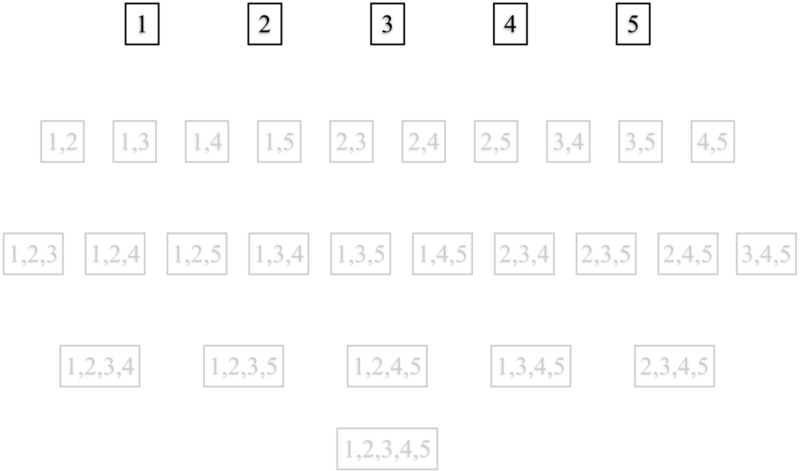
- Find the set of active groups, and the boundary groups violating the necessary conditions.
- Solve the problem with respect to these variables.

This adds groups that satisfy hierarchical inclusion, and where the model poorly estimates the higher-moment in the data.

(analogous to the greedy method of [Gevarter, 1987] for fitting maximum entropy distributions subject to marginal constraints [Cheeseman, 1983]).

Example of Active Set Method

Initial boundary groups.



Example of Active Set Method

Optimize initial boundary groups.

1 2 3 4 5

1,2 1,3 1,4 1,5 2,3 2,4 2,5 3,4 3,5 4,5

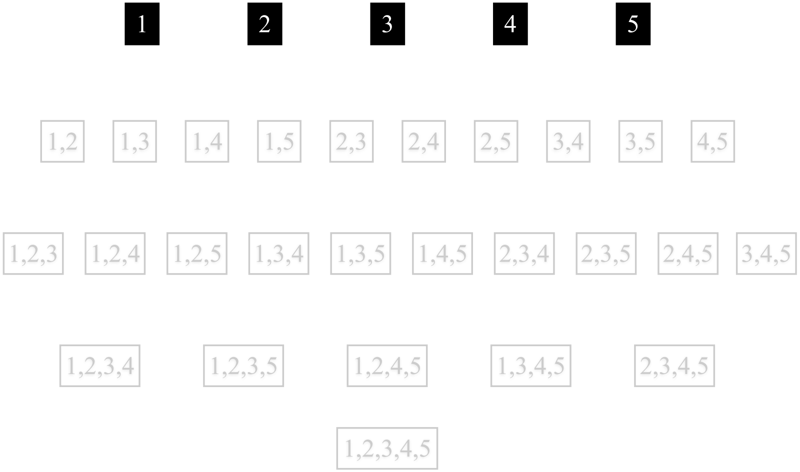
1,2,3 1,2,4 1,2,5 1,3,4 1,3,5 1,4,5 2,3,4 2,3,5 2,4,5 3,4,5

1,2,3,4 1,2,3,5 1,2,4,5 1,3,4,5 2,3,4,5

1,2,3,4,5

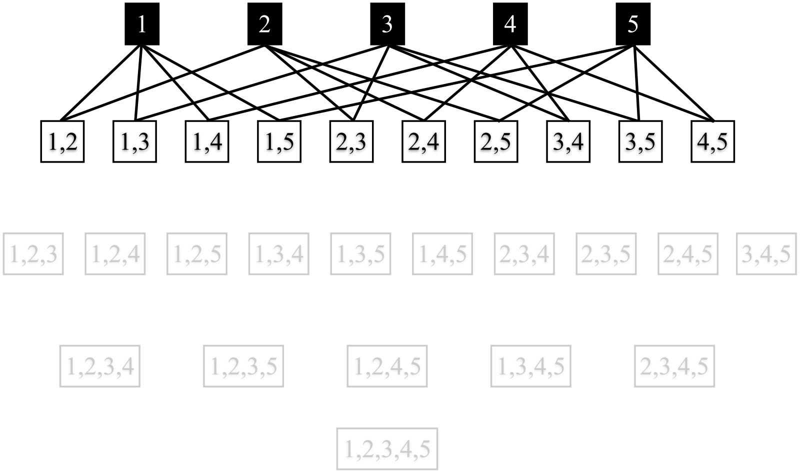
Example of Active Set Method

Find new **active groups**.



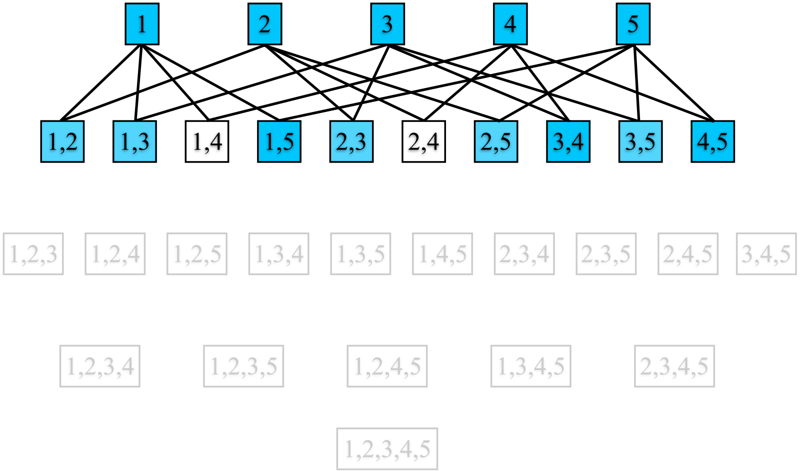
Example of Active Set Method

Find new boundary groups.



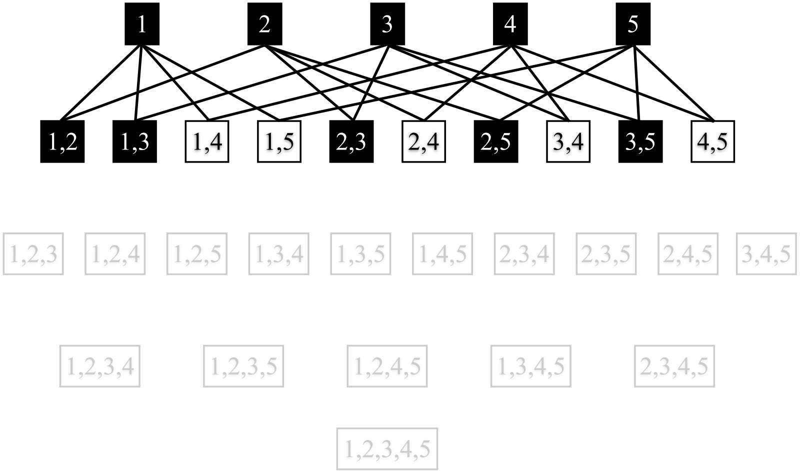
Example of Active Set Method

Optimize active groups and sub-optimal boundary groups.



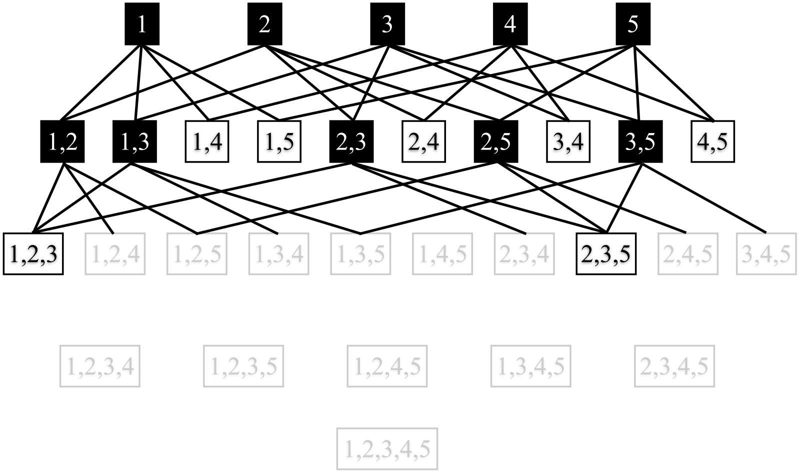
Example of Active Set Method

Find new **active groups**.



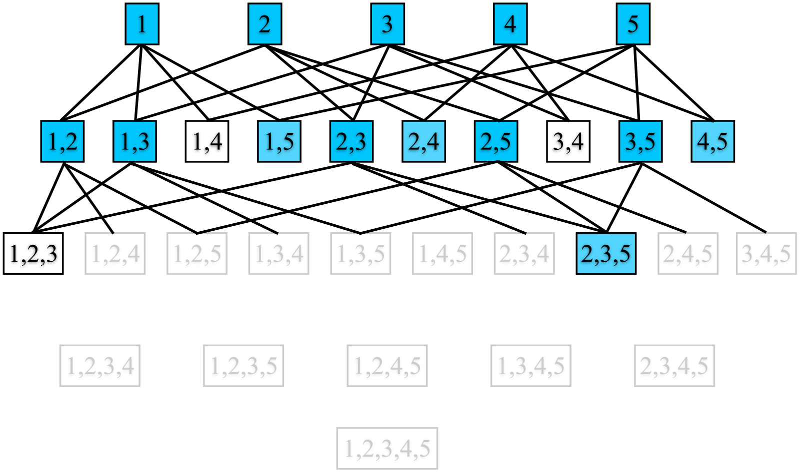
Example of Active Set Method

Find new boundary groups.



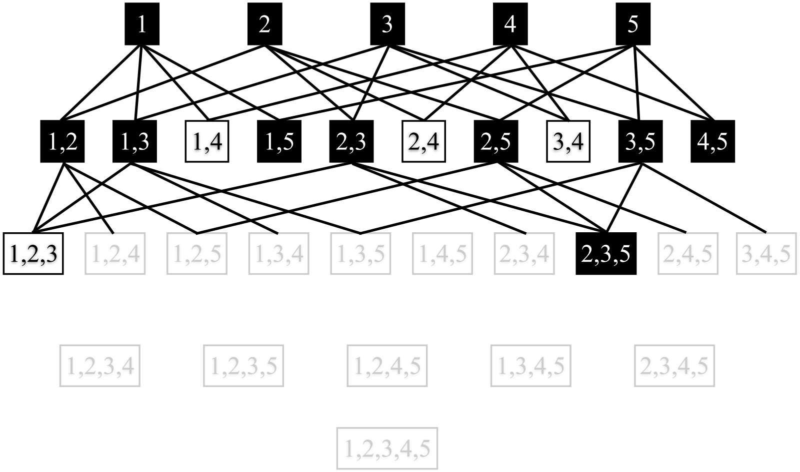
Example of Active Set Method

Optimize active groups and sub-optimal boundary groups.



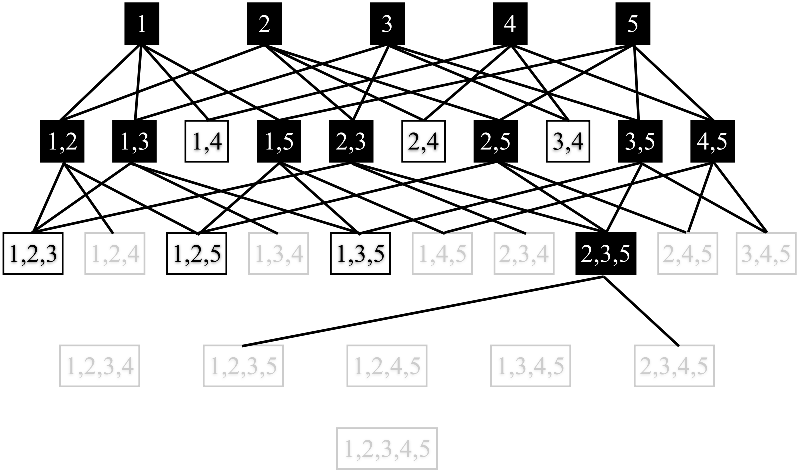
Example of Active Set Method

Find new **active groups**.



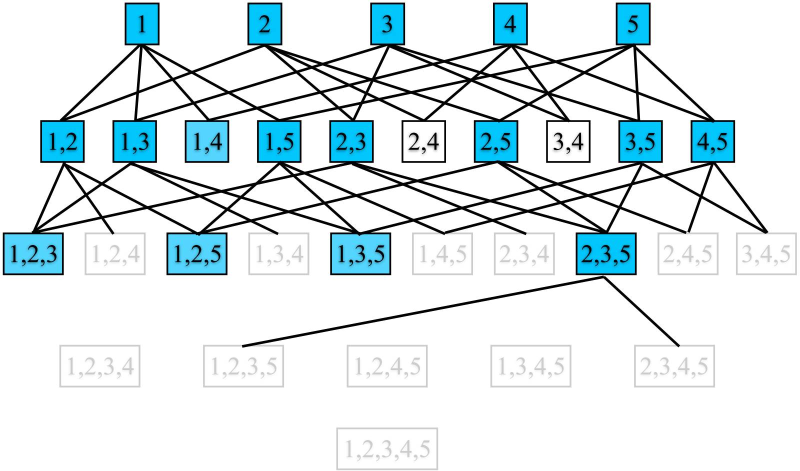
Example of Active Set Method

Find new boundary groups.



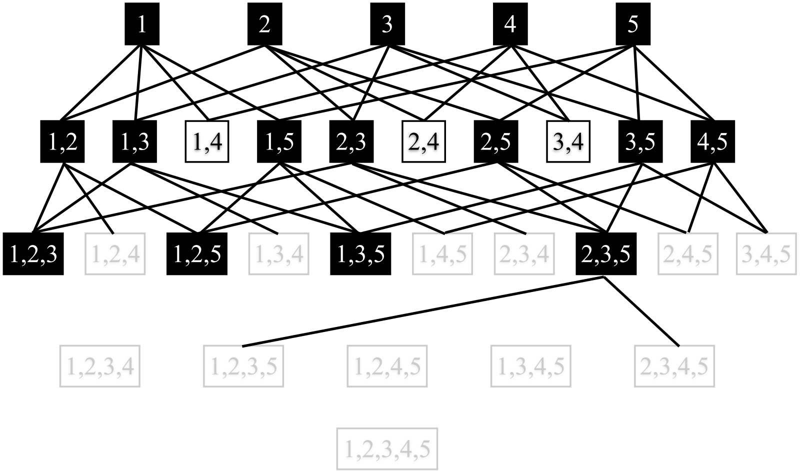
Example of Active Set Method

Optimize active groups and sub-optimal boundary groups.



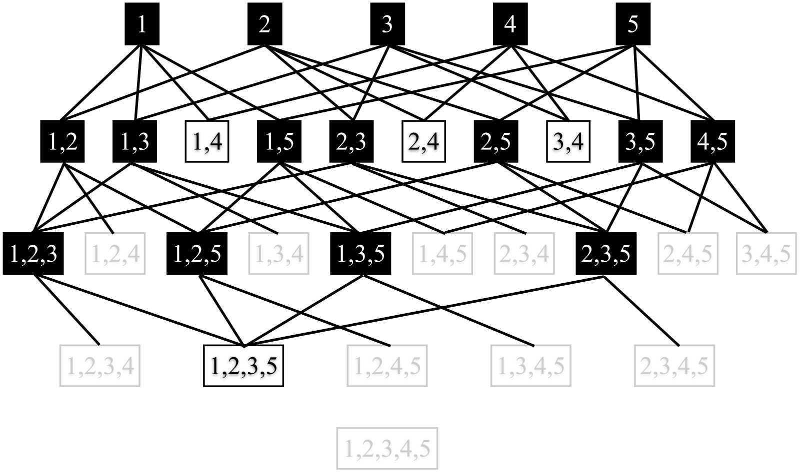
Example of Active Set Method

Find new **active groups**.



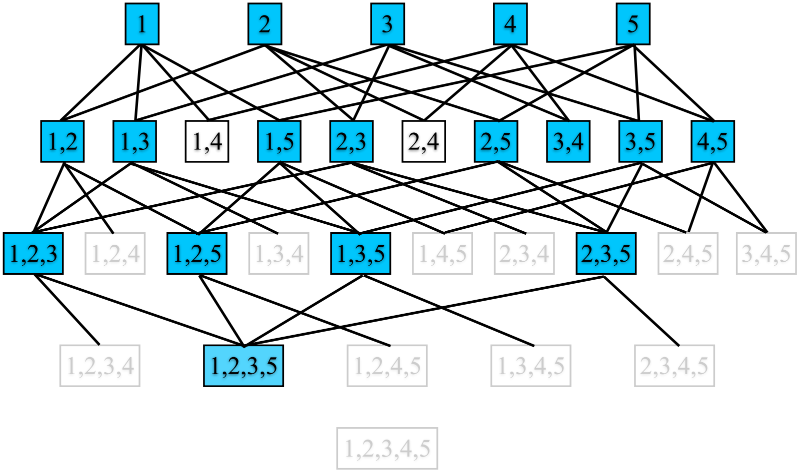
Example of Active Set Method

Find new boundary groups.



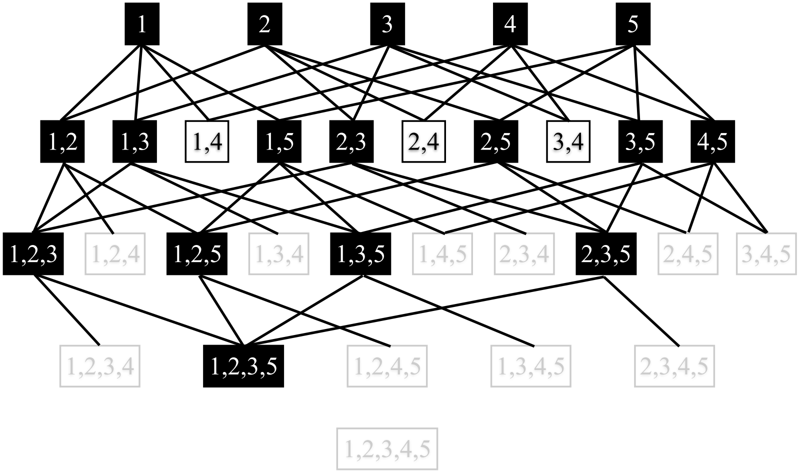
Example of Active Set Method

Optimize active groups and sub-optimal boundary groups.



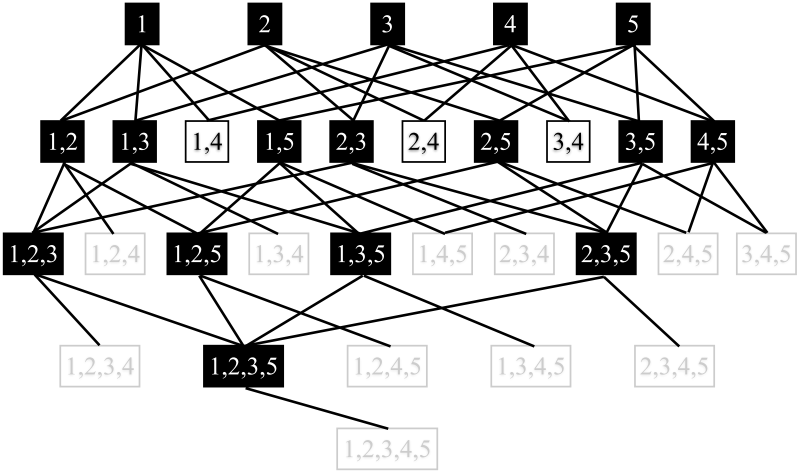
Example of Active Set Method

Find new **active groups**.



Example of Active Set Method

No new boundary groups, so we are done.



Example of Active Set Method

- In this example, we only needed to consider 4 of 10 possible threeway interactions, 1 of 5 fourway interactions, and no fiveway interactions.
- The active set method can save us from looking at an exponential number of higher-order factors.
- We still need to efficiently optimize the active groups and sub-optimal boundary groups...

Example of Active Set Method

- In this example, we only needed to consider 4 of 10 possible threeway interactions, 1 of 5 fourway interactions, and no fiveway interactions.
- The active set method can save us from looking at an exponential number of higher-order factors.
- We still need to efficiently optimize the active groups and sub-optimal boundary groups...

Example of Active Set Method

- In this example, we only needed to consider 4 of 10 possible threeway interactions, 1 of 5 fourway interactions, and no fiveway interactions.
- The active set method can save us from looking at an exponential number of higher-order factors.
- We still need to efficiently optimize the active groups and sub-optimal boundary groups...

Optimizing the Active Set

- Solving with the current active set is a group ℓ_1-regularization problem with overlapping groups,

$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A^*||_2.$$

- We write this non-smooth problem as an equivalent smooth problem with simple Euclidean norm cone constraints,

$$\min_{\mathbf{w},\mathbf{g}} -\log p(\mathbf{x}|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A g_A,$$

$$s.t.\ \forall_A,\ \ g_A \geq ||\mathbf{w}_A^*||_2.$$

Optimizing the Active Set

- Solving with the current active set is a group ℓ_1-regularization problem with overlapping groups,

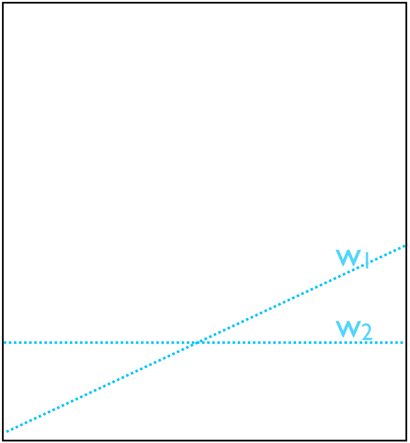
$$\min_{\mathbf{w}} -\sum_{i=1}^{n} \log p(\mathbf{x}^i|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A ||\mathbf{w}_A^*||_2.$$

- We write this non-smooth problem as an equivalent smooth problem with simple Euclidean norm cone constraints,

$$\min_{\mathbf{w},\mathbf{g}} -\log p(\mathbf{x}|\mathbf{w}) + \sum_{A \subseteq S} \lambda_A g_A,$$

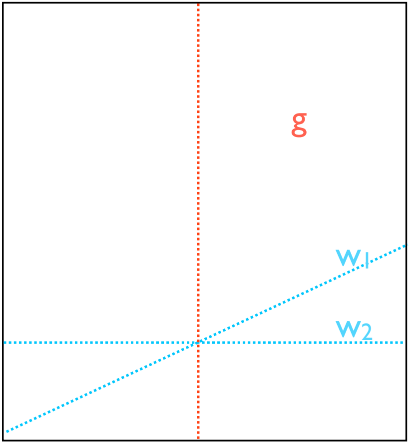
$$s.t.\ \forall_A,\ \ g_A \geq ||\mathbf{w}_A^*||_2.$$

Euclidean Norm Cone



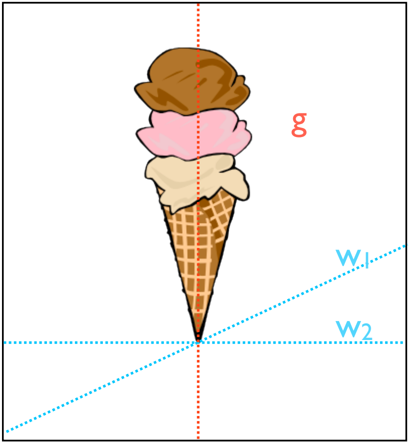
$$\{\{\mathbf{w}, g\} | g \geq ||\mathbf{w}||_2\}$$

Euclidean Norm Cone



$$\{\{\mathbf{w}, g\} | g \geq ||\mathbf{w}||_2\}$$

Euclidean Norm Cone



$$\{\{\mathbf{w}, g\} | g \geq ||\mathbf{w}||_2\}$$

Projected Gradient

- Projected gradient methods [Goldstein, 1964, Levitin and Poljak, 1965] are widely used for optimization with simple constraints.
- These methods use iterations of the form

$$\mathbf{w}_{k+1} = \mathcal{P}_{\mathcal{C}}(\mathbf{w}_k - \alpha \nabla f(\mathbf{w}_k)).$$

- The function $\mathcal{P}_{\mathcal{C}}(\mathbf{w})$ computes the Euclidean projection of a point $\mathbf{w}$ onto the convex set $\mathcal{C}$,

$$\mathcal{P}_{\mathcal{C}}(\mathbf{w}) = \arg\min_{\mathbf{x} \in \mathcal{C}} ||\mathbf{x} - \mathbf{w}||_2.$$

Projected Gradient

- Projected gradient methods [Goldstein, 1964, Levitin and Poljak, 1965] are widely used for optimization with simple constraints.
- These methods use iterations of the form

$$\mathbf{w}_{k+1} = \mathcal{P}_{\mathcal{C}}(\mathbf{w}_k - \alpha \nabla f(\mathbf{w}_k)).$$

- The function $\mathcal{P}_{\mathcal{C}}(\mathbf{w})$ computes the Euclidean projection of a point $\mathbf{w}$ onto the convex set $\mathcal{C}$,

$$\mathcal{P}_{\mathcal{C}}(\mathbf{w}) = \arg\min_{\mathbf{x} \in \mathcal{C}} ||\mathbf{x} - \mathbf{w}||_2.$$

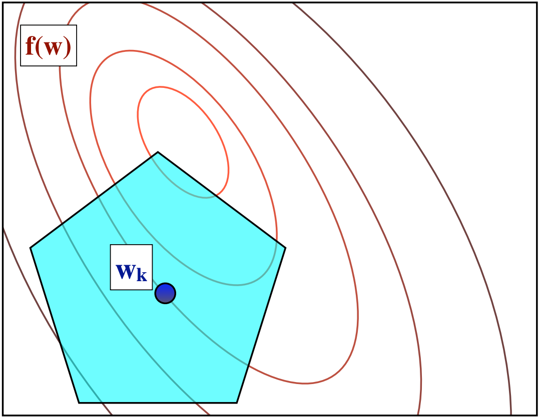
Projection onto Euclidean Norm Cone

It is easy to project onto the Euclidean norm cone [Boyd and Vandenberghe, 2004, Exercise 7.3(c)]:

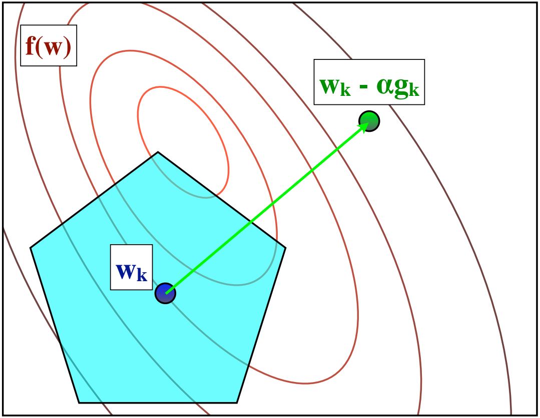
$$\mathcal{P}_{\mathcal{C}}(\mathbf{w}_A^*, g_A) = \begin{cases} (\mathbf{0}, 0), & \text{if } ||\mathbf{w}_A^*||_2 \leq -g_A, \\ (\mathbf{w}_A^*, g_A), & \text{if } ||\mathbf{w}_A^*||_2 \leq g_A, \\ \frac{1+g_A/||\mathbf{w}_A^*||_2}{2}(\mathbf{w}_A^*, ||\mathbf{w}_A^*||_2), & \text{if } ||\mathbf{w}_A^*||_2 > |g_A|. \end{cases}$$

Thus, it is simple to analytically compute the projection onto a single constraint.

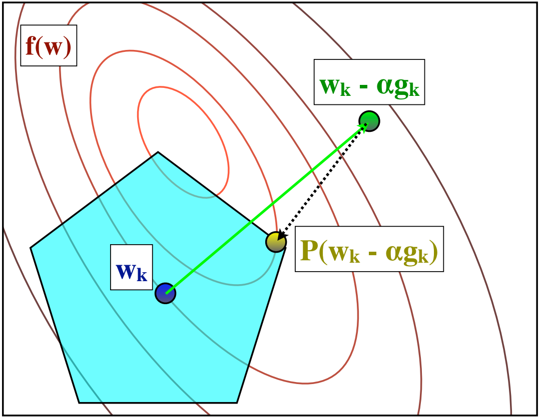
Projected Gradient Algorithm



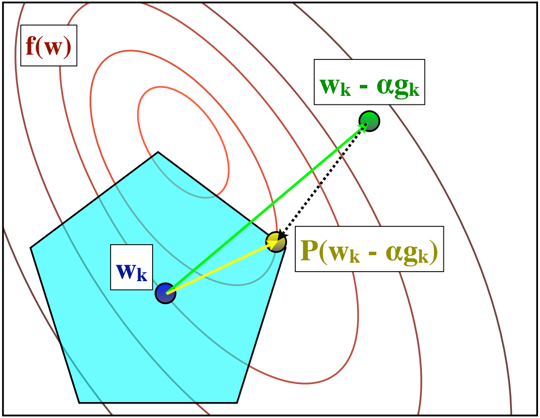
Projected Gradient Algorithm



Projected Gradient Algorithm



Projected Gradient Algorithm



Enhanced Projected Gradient Methods

The basic projected gradient method converges slowly, but several enhancements are possible:

- Spectral projected gradient: Barzilai-Borwein step length and non-monotomic line search [Birgin et al., 2000].
- Accelerated projected gradient: Extrapolation step to achieve a better worst-case convergence rate [Nesterov, 2004].
- Inexact projected quasi-Newton: L-BFGS approximation to Hessian matrix [Schmidt et al., 2009].

Projection onto the Intersection of Simple Sets

- We can easily compute the projection onto each norm cone.
- But since the groups overlap we can't do this independently.
- We want the projection onto the intersection of simple sets.

Projection onto the Intersection of Simple Sets

- We can easily compute the projection onto each norm cone.
- But since the groups overlap we can't do this independently.
- We want the projection onto the intersection of simple sets.

Projection onto the Intersection of Simple Sets

- We can easily compute the projection onto each norm cone.
- But since the groups overlap we can't do this independently.
- We want the projection onto the intersection of simple sets.

Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

von Neumann's Result

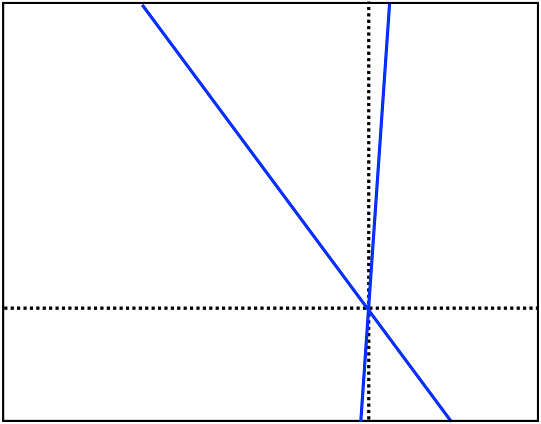
Definition 13.7: If $\phi_1, \phi_2, \ldots$ is a sequence Σ of s.v. operators, if f is an element of $\prod_{n=1}^{\infty} D(\phi_n)$ such that $\lim_{n\to\infty} \phi_n f$ exists, and if D is the set of all such elements f, then Σ is said to have a limit ϕ over D, and, for f ε D = = $D(\phi)$, $\phi f = \lim_{n\to\infty} \phi_n f$.

THEOREM 13.7. If $E = P_M$ and $F = P_N$, then the sequence Σ_1 of operators E, FE, EFE, FEFE, ... has a limit G, the sequence Σ_2: F, EF, FEF, ... has the same limit G, and $G = P_{MN}$. (The condition EF = FE need not hold.)

Proof: Let A_n be the n^{th} operator of the sequence Σ_1. Then $(A_m f, A_n g) = (A_{m+n-\varepsilon} f, g)$, where $\varepsilon = 1$ if m and n have the same parity and $\varepsilon = 0$ if m and n have opposite parity. It must be shown that if f is any element of S, then $\lim_{n\to\infty} A_n f$ exists. But $\| A_m f - A_n f \|^2 = (A_m f - A_n f, A_m f - A_n f) =$

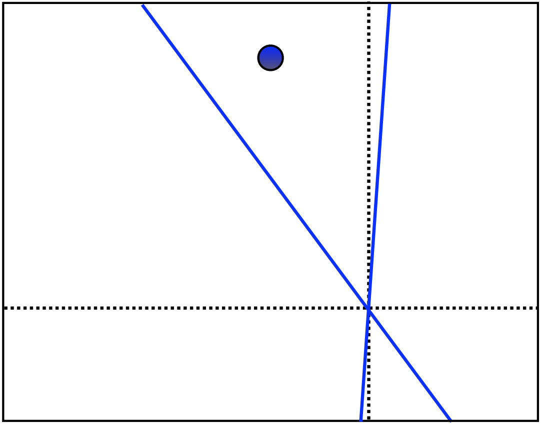
von Neumann's Result

Take two intersecting subspaces.



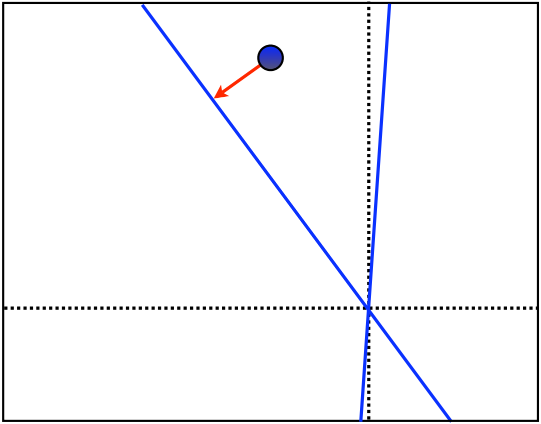
von Neumann's Result

We want to project a point onto their intersection.



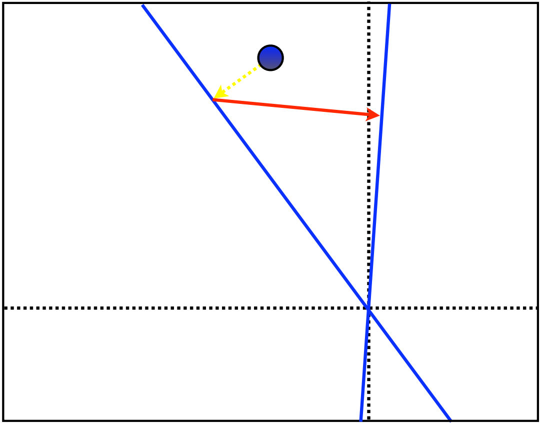
von Neumann's Result

Project onto subspace 1.



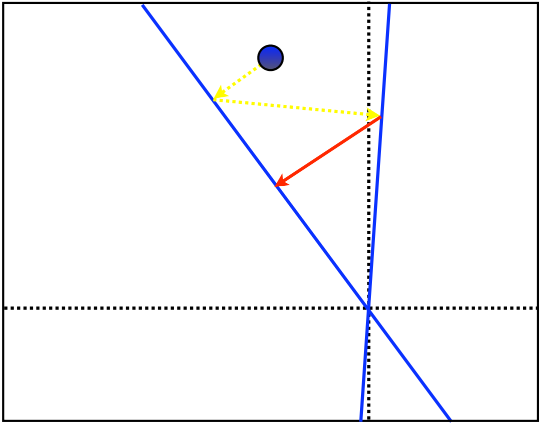
von Neumann's Result

Project onto subspace 2.



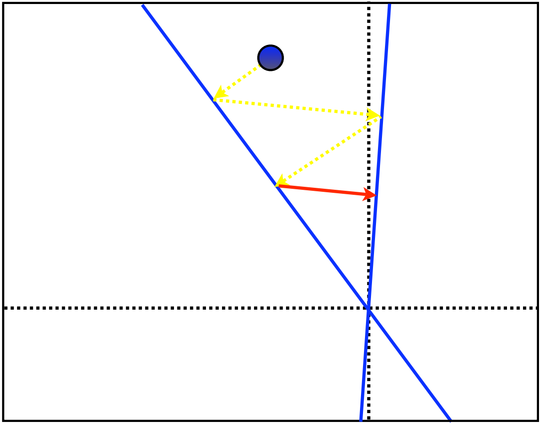
von Neumann's Result

Project onto subspace 1.



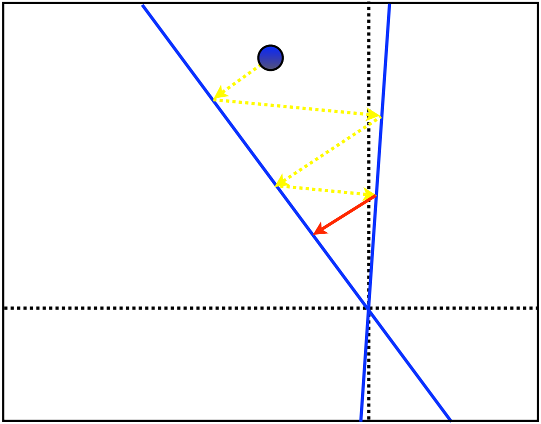
von Neumann's Result

Project onto subspace 2.



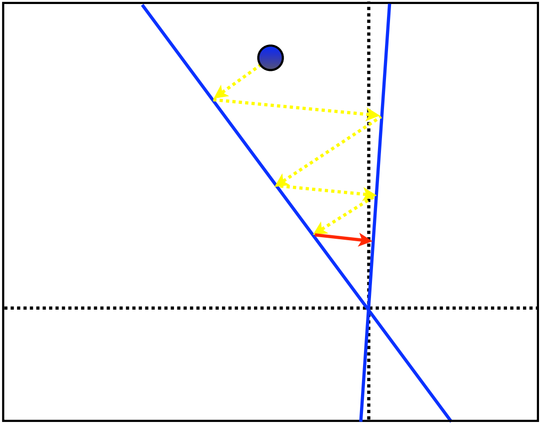
von Neumann's Result

Project onto subspace 1.



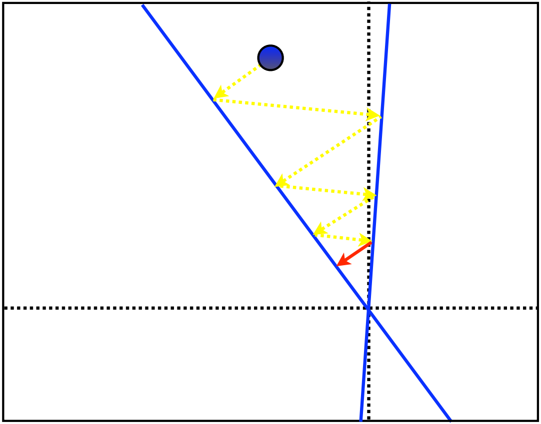
von Neumann's Result

Project onto subspace 2.



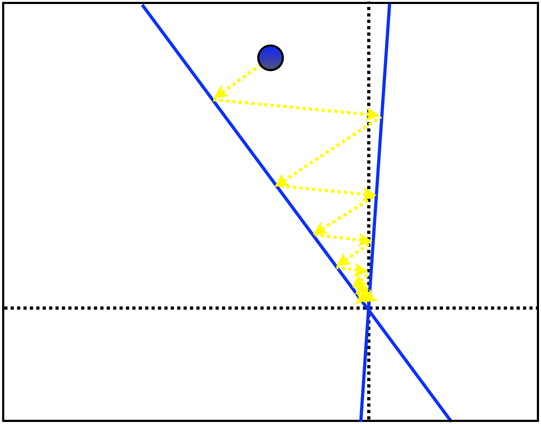
von Neumann's Result

Project onto subspace 1.



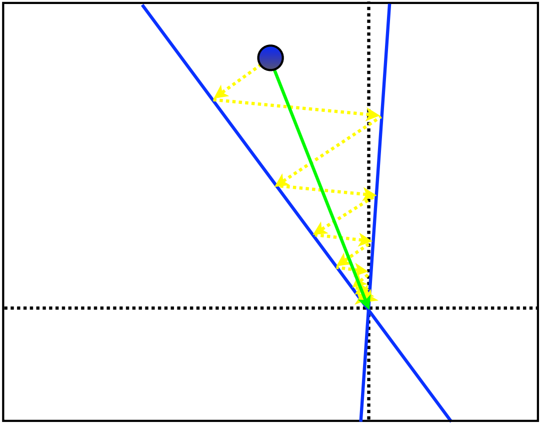
von Neumann's Result

And keep going...



von Neumann's Result

The limit is the projection onto the intersection.



Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

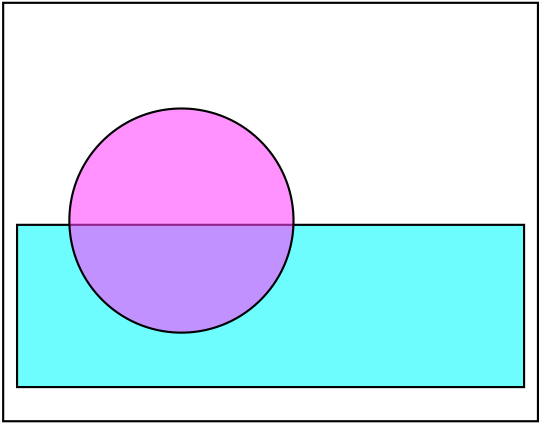
Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

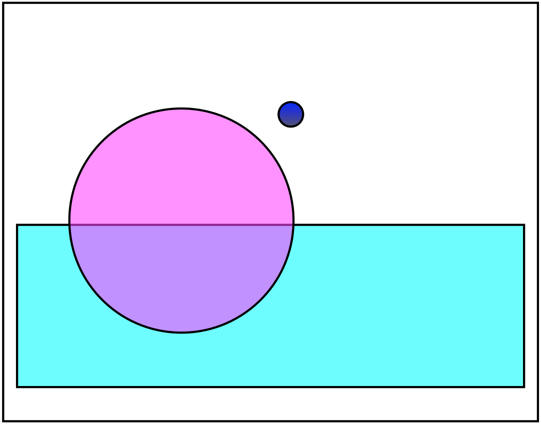
Bregman's Algorithm

We have an arbitrary number of convex sets.



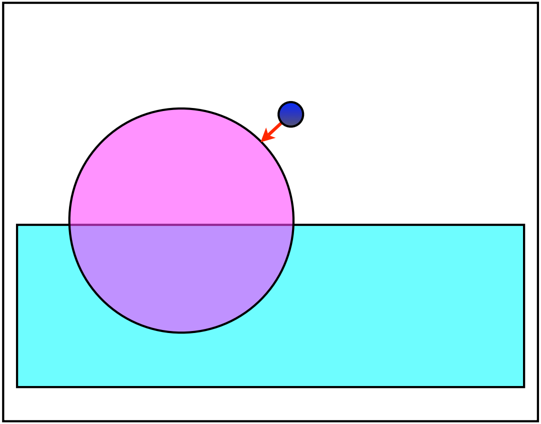
Bregman's Algorithm

Start with some initial point.



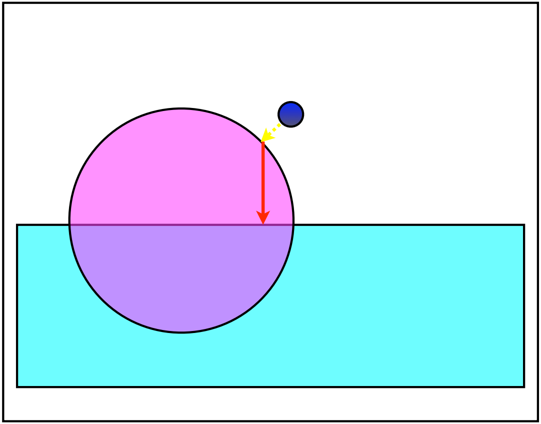
Bregman's Algorithm

Project onto convex set 1.



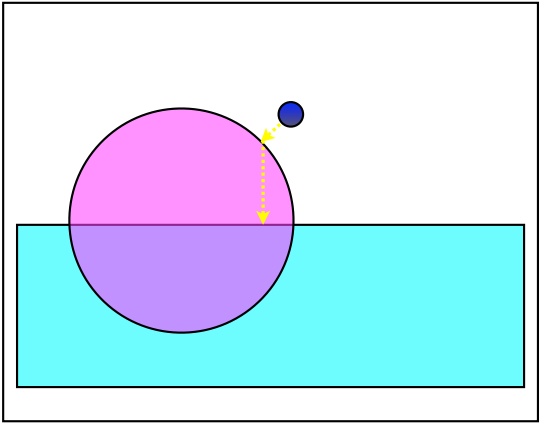
Bregman's Algorithm

Project onto convex set 2.



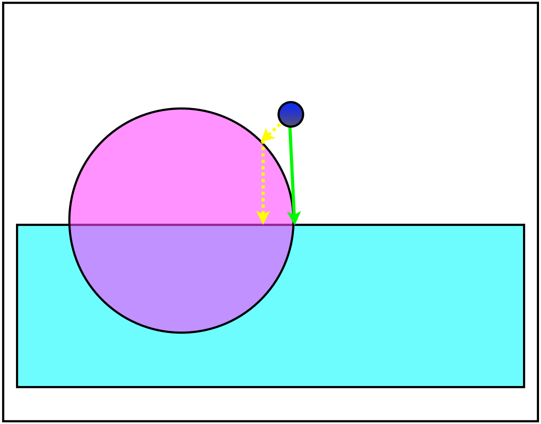
Bregman's Algorithm

The limit is a point in the intersection.



Bregman's Algorithm

In general, the limit is not the projection.



Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

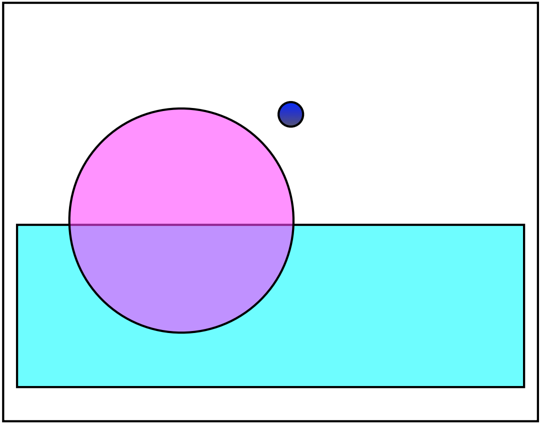
Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

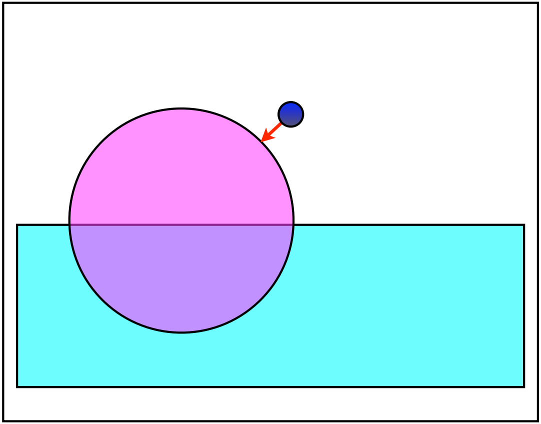
Dykstra's Algorithm

We want to project a point onto the intersection of convex sets.



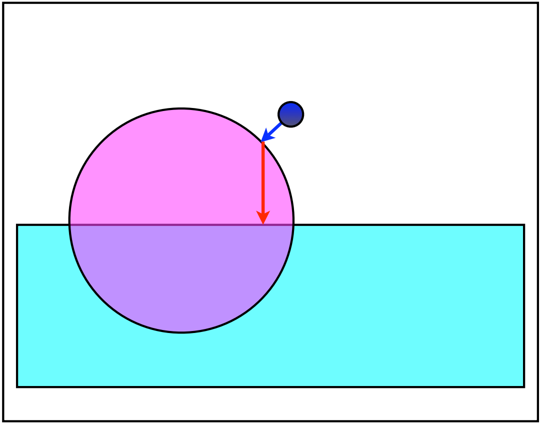
Dykstra's Algorithm

Project onto convex set 1, and store the difference.



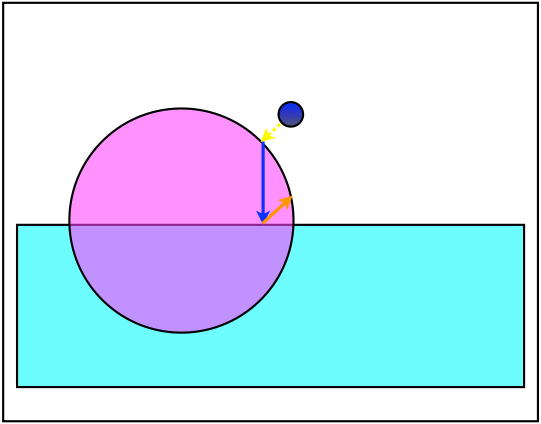
Dykstra's Algorithm

Project onto convex set 2, and store the difference.



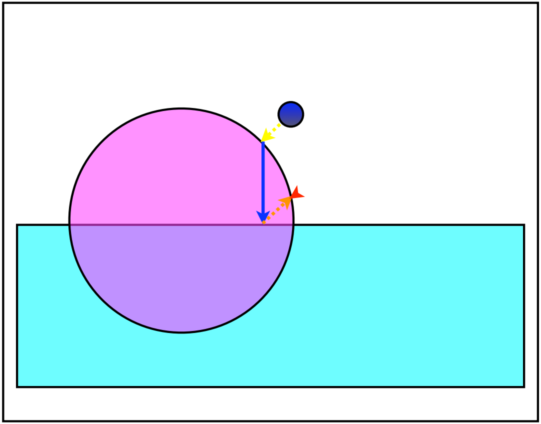
Dykstra's Algorithm

Remove the difference from projecting on convex set 1.



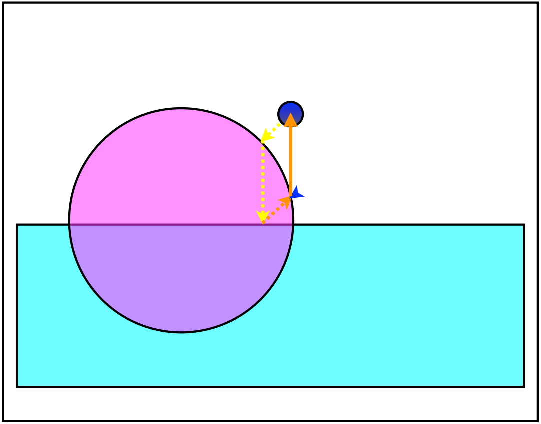
Dykstra's Algorithm

Project onto convex set 1, and store the difference.



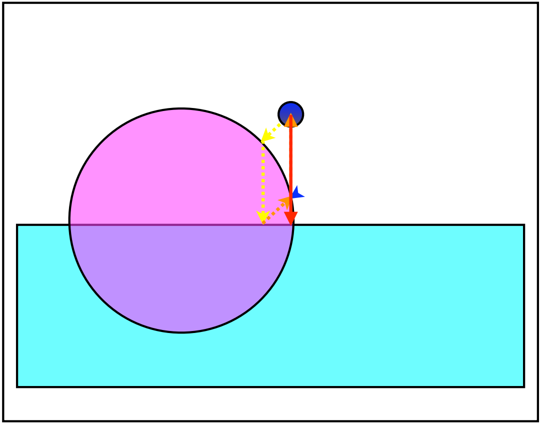
Dykstra's Algorithm

Remove the difference from projecting on convex set 2.



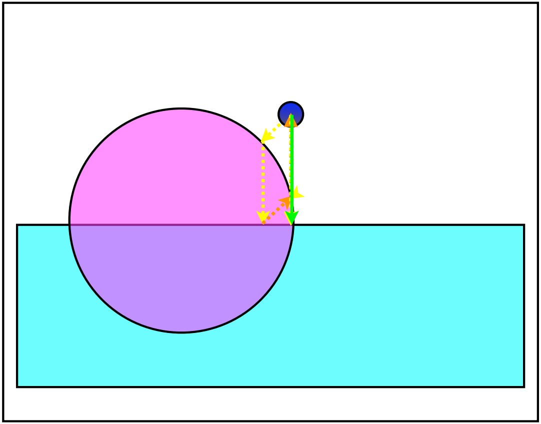
Dykstra's Algorithm

Project onto convex set 2, and store the difference.



Dykstra's Algorithm

The limit is the projection onto the intersection.



Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

Cyclic Projection Algorithms

Projecting onto the intersection of simple sets is a classic problem:

- In his 1933-34 lecture notes, von Neumann showed that cyclically projecting a point onto two subspaces converges to the projection onto their intersection.
- Bregman [1965] showed that cyclically projecting onto general convex sets converges to a point in their intersection. (but not necessarily the projection)
- Dykstra [1983] showed that a simple modification makes the method converge to the projection for general convex sets.
- Deutsch and Hundal [1994] showed that Dykstra's algorithm converges at a geometric rate for polyhedral sets.

Outline

Multivariate Flow Cytometry Experiments

Does it empirically help to have higher-order potentials?

We first consider a small data set where we can tractably compute the normalizing constant:

- Multivariate flow cytometry [Sachs et al., 2005].

We compared:

- Pairwise with ℓ_2-regularization and group ℓ_1-regularization.
- Threeway with ℓ_2-regularization and group ℓ_1-regularization.
- Hierarchical with overlapping group ℓ_1-regularization.

We trained on $1/3$, used $1/3$ to select λ, and used $1/3$ as a test set (for 10 random splits).

Multivariate Flow Cytometry Experiments

Does it empirically help to have higher-order potentials?

We first consider a small data set where we can tractably compute the normalizing constant:

- Multivariate flow cytometry [Sachs et al., 2005].

We compared:

- Pairwise with ℓ_2-regularization and group ℓ_1-regularization.
- Threeway with ℓ_2-regularization and group ℓ_1-regularization.
- Hierarchical with overlapping group ℓ_1-regularization.

We trained on 1/3, used 1/3 to select λ, and used 1/3 as a test set (for 10 random splits).

Multivariate Flow Cytometry Experiments

Does it empirically help to have higher-order potentials?

We first consider a small data set where we can tractably compute the normalizing constant:

- Multivariate flow cytometry [Sachs et al., 2005].

We compared:

- Pairwise with ℓ_2-regularization and group ℓ_1-regularization.
- Threeway with ℓ_2-regularization and group ℓ_1-regularization.
- Hierarchical with overlapping group ℓ_1-regularization.

We trained on 1/3, used 1/3 to select λ, and used 1/3 as a test set (for 10 random splits).

Multivariate Flow Cytometry Experiments

Does it empirically help to have higher-order potentials?

We first consider a small data set where we can tractably compute the normalizing constant:

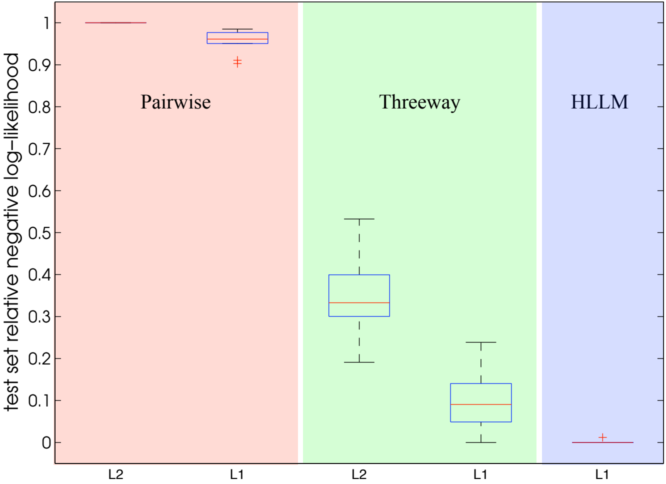
- Multivariate flow cytometry [Sachs et al., 2005].

We compared:

- Pairwise with ℓ_2-regularization and group ℓ_1-regularization.
- Threeway with ℓ_2-regularization and group ℓ_1-regularization.
- Hierarchical with overlapping group ℓ_1-regularization.

We trained on $1/3$, used $1/3$ to select λ, and used $1/3$ as a test set (for 10 random splits).

Flow Cytometry Data



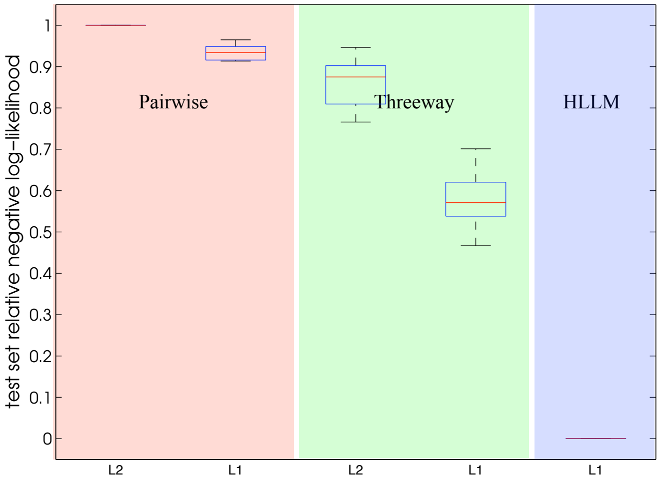
Traffic and USPS Experiments

We next consider two larger data sets:

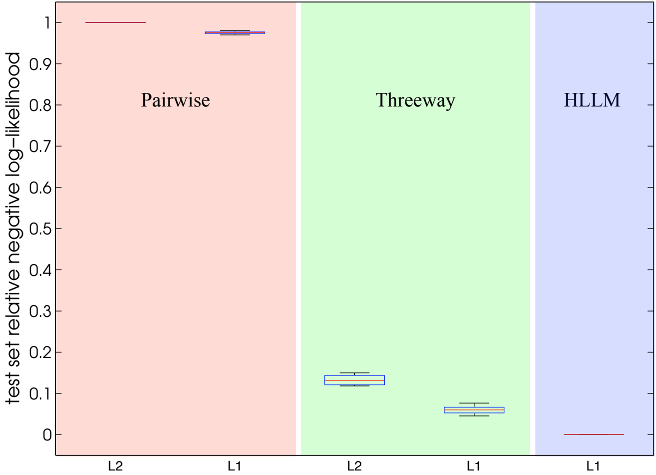
- Traffic flow level [Shahaf et al., 2009].
- USPS digits data discretized into four states.

On these experiments we used weighted Ising potentials, and used a pseudo-likelihood for training/test.

Traffic Flow Data



USPS Data


test set relative negative log−likelihood
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Pairwise
Threeway
HLLM
L2
L1
L2
L1
L1

Structure Estimation

- We sought to test whether the HLLM model could recover a true structure.
- We generated samples from a 10-node data set with potentials $(2,3)(4,5,6)(7,8,9,10)$ and parameters from $\mathcal{N}(0,1)$.
- We recorded the number of false positives of different orders for the first model along the regularization path that includes the true model.
- Eg., with 20000 samples the order was (8,10)(7,9)(9,10)(7,10)(4,5)(8,9)(2,3)(4,6)(8,9,10)(7,8)(7,8,9)(7,8,10)(5,6)(1,8)(5,9)(3,8)(3,7)(4,5,6)(1,7)(7,9,10)(7,8,9,10)

Structure Estimation

- We sought to test whether the HLLM model could recover a true structure.
- We generated samples from a 10-node data set with potentials $(2,3)(4,5,6)(7,8,9,10)$ and parameters from $\mathcal{N}(0,1)$.
- We recorded the number of false positives of different orders for the first model along the regularization path that includes the true model.
- Eg., with 20000 samples the order was (8,10)(7,9)(9,10)(7,10)(4,5)(8,9)(2,3)(4,6)(8,9,10)(7,8) (7,8,9)(7,8,10)(5,6)(1,8)(5,9)(3,8)(3,7)(4,5,6)(1,7)(7,9,10) (7,8,9,10)

Structure Estimation

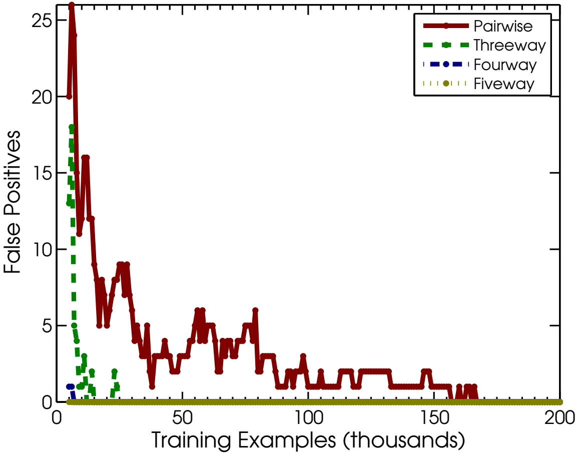
- We sought to test whether the HLLM model could recover a true structure.
- We generated samples from a 10-node data set with potentials $(2,3)(4,5,6)(7,8,9,10)$ and parameters from $\mathcal{N}(0,1)$.
- We recorded the number of false positives of different orders for the first model along the regularization path that includes the true model.
- Eg., with 20000 samples the order was (8,10)(7,9)(9,10)(7,10)(4,5)(8,9)(2,3)(4,6)(8,9,10)(7,8) (7,8,9)(7,8,10)(5,6)(1,8)(5,9)(3,8)(3,7)(4,5,6)(1,7)(7,9,10) (7,8,9,10)

Structure Estimation

- We sought to test whether the HLLM model could recover a true structure.
- We generated samples from a 10-node data set with potentials $(2,3)(4,5,6)(7,8,9,10)$ and parameters from $\mathcal{N}(0,1)$.
- We recorded the number of false positives of different orders for the first model along the regularization path that includes the true model.
- Eg., with 20000 samples the order was (8,10)(7,9)(9,10)(7,10)(4,5)(8,9)(2,3)(4,6)(8,9,10)(7,8) (7,8,9)(7,8,10)(5,6)(1,8)(5,9)(3,8)(3,7)(4,5,6)(1,7)(7,9,10) (7,8,9,10)

Synthetic Data: Types of Errors

Types of errors made by HLLM:



Outline

Extensions

- Dykstra's algorithm may be useful for other overlapping group ℓ_1-regularization problems.
- The model can be applied to learn hierarchical conditional random fields.
- The main remaining issue is finding inactive groups that do not satisfy sufficient optimality conditions. A simple heuristic is to look at an extended boundary.

Summary

- We give a convex formulation of structure learning in hierarchical log-linear models.
- We proposed methods to deal with the exponential number of variables.
- We found that going beyond pairwise potentials gives similar or better results on every data set we tried.

(thanks to the reviewers, and code will be online soon...)

Summary

- We give a convex formulation of structure learning in hierarchical log-linear models.
- We proposed methods to deal with the exponential number of variables.
- We found that going beyond pairwise potentials gives similar or better results on every data set we tried.

(thanks to the reviewers, and code will be online soon...)

Summary

- We give a convex formulation of structure learning in hierarchical log-linear models.
- We proposed methods to deal with the exponential number of variables.
- We found that going beyond pairwise potentials gives similar or better results on every data set we tried.

(thanks to the reviewers, and code will be online soon...)

Summary

- We give a convex formulation of structure learning in hierarchical log-linear models.
- We proposed methods to deal with the exponential number of variables.
- We found that going beyond pairwise potentials gives similar or better results on every data set we tried.

(thanks to the reviewers, and code will be online soon...)