Convex Optimization and Modeling

Numerical linear algebra

14th (last) lecture, 14.07.2010

Jun.-Prof. Matthias Hein

Oral Exam

Topics:

- convex sets and functions
- convex optimization problems
- optimality conditions and duality theory
- unconstrained minimization (steepest descent, Newton, subgradient)
- interior point method
- constrained first order method (including FISTA)
- EXERCISES !

Convergence proofs:

- key ideas

Program of today

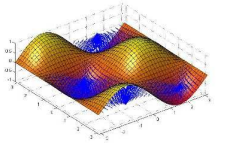
Numerical Linear Algebra:

- Sparse Matrices
- Linear System: direct solution via factorization
- Low rank updates

Semi-definite Programming:

- Globally optimal solution of a non-convex problem
- The best approximation of the sparsest cut

Sparse Matrices



Sparse Matrix: Let $A \in \mathbb{R}^{n \times m}$ be a sparse matrix (most entries are zero). Consider the case where n and m are very large - full matrix would never fit it into memory.

Sparse Matrix Format: N denotes the number of nonzeros

- **List of Coordinates**

  $N \times 3$ array - `[rowIx,colIx,val]`

  Fast for creation of sparse matrix (Matlab).

- **Compressed Column (Row) Storage**

  - array of length N containing value of nonzero elements,
  - array of length N containing row indices of nonzero elements ,
  - array of length $m + 1$ where the s-entries points to the start of the s-th column in the value and row array

  Internal Storage Format of sparse matrices in Matlab.

Compressed Column Storage: $A \in \mathbb{R}^{n \times m}$ with N nonzeros.

$$A = \begin{pmatrix} 0 & 5 & 0 & 0 \\ 4 & 1 & 0 & 3 \\ 1 & 2 & 0 & 7 \\ 0 & 8 & 0 & 0 \end{pmatrix}.$$

has CCS representation (indices start with zero !)

$$\texttt{val} = [4, 1, 5, 1, 2, 8, 3, 7] - \qquad N \text{ entries}$$

$$\texttt{row} = [1, 2, 0, 1, 2, 3, 1, 2] - \qquad N \text{ entries}$$

$$\texttt{col} = [0, 2, 6, 6, 8] - \qquad m + 1 \text{ entries}$$

Number of elements in column j: `col`(j+1)-`col`(j).

Number of non-zero elements in A: `col`(m+1).

Which will be faster ?

`b = A*x` or `bt = xt*At` `with: xt = x'; At = A';`

Which will be faster ?

`b = A*x` or `bt = xt*At with: xt = x'; At = A';`

Matrix Multiplication from the left is much faster !

- the second is faster - quick experiment $10 - 15\%$,
- can be easily parallelized.

Matrix-Vector-Multiplication from the left:

$$b_j = \sum_{i=1}^{n} x_i \, A_{ij} = \sum_{i=\texttt{col}(j)}^{\texttt{col}(j+1)-1} x_{\texttt{row}(i)} \texttt{val}(i).$$

Summary:

- quick access of every column,
- but: adding an element which does not exist yet is expensive !

Solving a linear system

$$Ax = b,$$

where $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$, A has full rank.

Methods:

- direct (non-iterative) methods,
- iterative methods.

$\Rightarrow$ iterative methods are better for **large-scale problems** since they need less memory (fill-in phenomenon) and are also easier to parallelize.

Basic costs of matrix operations

- **inner product in $\mathbb{R}^n$:** $2n-1$ flops,
- **matrix-vector product:** Ax with $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$
  1. general case: $2mn$ flops,
  2. A sparse, N nonzero entries: $2N$ flops,
  3. A factorized, $A = UV$ with $U \in \mathbb{R}^{m\times p}$ and $V \in \mathbb{R}^{p\times n}$: $2p(m+n)$ flops

Solving easy linear systems: $Ax = b$

- **diagonal matrix A:** $x_i = \frac{b_i}{a_{ii}} \Longrightarrow n$ flops,
- **lower triangular matrix A:**

$$\begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix},$$

  flops: $\sum_{k=1}^{n}(2k-1) = n^2$, less for structured matrices.
- **upper triangular matrix A:** same as lower triangular matrix,
- **orthogonal matrix A:** $x = A^{-1}b = A^T b \Longrightarrow n^2$ flops,
- **permutation matrix P:** A permutation matrix P is orthogonal $\Rightarrow$ $x = P^T b$ with 0 flops,

Solving linear systems using factorization

We have a factorization of the matrix A into

$$A = BC.$$

Then we solve the linear system $Ax = b$ using the steps

- $Bz = b$,
- $Cx = z$.

Cost: matrix factorization F + 2 solutions of linear systems S,
$\Longrightarrow B, C$ upper/lower triangular, $S = n^2$.

Multiple right hand sides: $Ax = B$, $B = (b_1, \ldots, b_m)$ $(m \leq n)$.
Cost: $F + 2mS$.

LU Factorization of a non-singular, square matrix A

$$A = PLU,$$

where

- $P \in \mathbb{R}^{n \times n}$ is a permutation matrix,
- $L \in \mathbb{R}^{n \times n}$ is a lower triangular matrix,
- $U \in \mathbb{R}^{n \times n}$ is a upper triangular matrix.

$\Longrightarrow$ **Gaussian elimination (with partial pivoting)** needs $\frac{2n^3}{3}$ flops.

Solving the linear system: $Ax = b$ via

$$z_1 = P^T b, \qquad Lz_2 = z_1, \qquad Ux = z_2,$$

which costs $2n^2$ flops.

Cholesky Factorization of a symmetric, positive-definite matrix A

$$A = LL^T,$$

where $L \in \mathbb{R}^{n \times n}$ is a lower triangular matrix,

Total cost: $\frac{n^3}{3}$ flops.

- **banded:** bandwidth k, nk^2 flops,
- **sparse:** complicated dependency on n, the number of nonzero components and the sparsity pattern.
  Usually reordering necessary for sparse cholesky factor L,

$$A = PLL^T P^T.$$

Factorization of matrices III

Let A_i be the principal submatrix:

$$A_i = \begin{pmatrix} a_{11} & \dots & a_{1i} \\ \vdots & & \vdots \\ a_{i1} & \dots & a_{ii} \end{pmatrix} =: \begin{pmatrix} A_{i-1} & \beta_i \\ \beta_i^T & a_{ii} \end{pmatrix} \text{ with } \beta_i = \begin{pmatrix} a_{1i} \\ \vdots \\ a_{i-1,i} \end{pmatrix}.$$

Iterative Algorithm for the Cholesky factorization:

- $A_1 = L_1 L_1^T$ with $L_1 = \sqrt{a_{11}}$,
- Let $A_{i-1} = L_{i-1} L_{i-1}^T$, then

$$L_i = \begin{pmatrix} L_{i-1} & 0 \\ l_i^T & \lambda_{ii} \end{pmatrix}, \quad \text{where } l_i = L_{i-1}^{-1}\beta_i \text{ and } \lambda_{ii} = \sqrt{a_{ii} - \|l_i\|^2}.$$

We have

$$A_i = L_i L_i^T = \begin{pmatrix} L_{i-1} & 0 \\ l_i^T & \lambda_{ii} \end{pmatrix} \begin{pmatrix} L_{i-1}^T & l_i \\ 0 & \lambda_{ii} \end{pmatrix} = \begin{pmatrix} L_{i-1}L_{i-1}^T & L_{i-1}l_i \\ l_i^T L_{i-1}^T & \lambda_{ii}^2 + l_i^T l_i \end{pmatrix}.$$

LDL^T Factorization of a nonsingular, symmetric matrix A

$$A = PLDL^T P^T,$$

where

- $P \in \mathbb{R}^{n \times n}$ is a permutation matrix,
- $L \in \mathbb{R}^{n \times n}$ is a lower triangular matrix with positive elements,
- $D \in \mathbb{R}^{n \times n}$ is block-diagonal with nonsingular 1×1 and 2×2 diagonal blocks.

Total cost: $\frac{n^3}{3}$ flops.

Solving a linear system with block structure

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where

- $A_{11} \in \mathbb{R}^{k \times k}$, $A_{12} \in \mathbb{R}^{k \times l}$, $A_{21} \in \mathbb{R}^{l \times k}$, $A_{22} \in \mathbb{R}^{l \times l}$,
- $x_1, b_1 \in \mathbb{R}^k$, $x_2, b_2 \in \mathbb{R}^l$,
- $n = k + l$.

Solution:

$$x_1 = A_{11}^{-1}(b_1 - A_{12}x_2), \qquad (A_{22} - A_{21}A_{11}^{-1}A_{12})x_2 = b_2 - A_{21}A_{11}^{-1}b_1,$$

where $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is called the **Schur complement.**

$\Longrightarrow$ advantage over the general case if A_{11}^{-1} easy to compute !

Sherman-Morrison-Woodbury Formula:

Let $A \in \mathbb{R}^{n\times n}$ have full rank, $B \in \mathbb{R}^{n\times p}$ and $C \in \mathbb{R}^{p\times n}$, then

$$\big(A + BC\big)^{-1} = A^{-1} - A^{-1}B(\mathbb{1} + CA^{-1}B)^{-1}CA^{-1}.$$

Proof:

$$(A + BC)x = b \qquad \Longleftrightarrow \qquad Ax + By = b, \quad y = Cx$$

$$\begin{pmatrix} A & B \\ C & -\mathbb{1} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

Block inversion leads to the desired result. □

$\Longrightarrow$ Key simplification: $CA^{-1}B$ has rank p.

Linear system

$$Ax = b.$$

Now we have to solve a new linear system

$$(A + uv^T)x' = b.$$

$\Longrightarrow$ difference has rank one - rank one update of x.

$$x' = (A^{-1} - A^{-1}u(\mathbb{1} + v^T A^{-1}u)^{-1}v^T A^{-1})b$$

$$= x - \frac{\langle v, x\rangle}{1 + \langle v, A^{-1}u\rangle} A^{-1}u.$$

Advantages:

- only factorization of A required !
- uv^T is in general a dense matrix !

What is global optimization ?

Find the globally optimal solution of a (non-convex) problem.

What is global optimization ?

Find the globally optimal solution of a (non-convex) problem.

We consider quadratic optimization with quadratic equality and inequality constraints.

$$\begin{aligned} \min_{f \in \mathbb{R}^n} \quad & \langle f, Af \rangle \\ & \langle f, Bf \rangle = c, \\ & \langle f, Cf \rangle \leq d, \end{aligned}$$

where A, B, C are positive semi-definite.

$\Longrightarrow$ Problem is non-convex due to equality constraint !

Can we find the globally optimal solution ?

Note, that $\langle f, Af \rangle = \text{trace}(Aff^T)$. Thus we get

$$\min_f \ \text{trace}(A\, f f^T)$$

$$\text{trace}(B\, f f^T) = c$$

$$\text{trace}(C\, f f^T) \leq d$$

What would you do ?

Relaxation into an SDP

$$\begin{aligned} \min_{X \in S^n} \quad & \operatorname{trace}(A\,X) \\ & \operatorname{trace}(B\,X) = c \\ & \operatorname{trace}(C\,X) \leq d \\ & X \succeq 0 \end{aligned}$$

Under which conditions can one get the solution of the original problem from the SDP ?

Relaxation into an SDP

$$\begin{aligned} \min_{X \in S^n} \quad & \operatorname{trace}(A\,X) \\ & \operatorname{trace}(B\,X) = c \\ & \operatorname{trace}(C\,X) \leq d \\ & X \succeq 0 \end{aligned}$$

Under which conditions can one get the solution of the original problem from the SDP ?

If the minimizer X^* has rank one !

Suppose we have an SDP of the form:

$$\begin{aligned} \min_{X \in S^n} \quad & \operatorname{trace}(A\,X) \\ & \operatorname{trace}(B_i\,X) = c_i, \quad i = 1, \ldots, m \\ & X \succeq 0 \end{aligned}$$

Theorem 1 (Pataki(1998)). *If X is an extreme point of the above SDP, then* $\operatorname{rank}(X) \leq r_m$, *where*

$$r_m = \max\{r \in \mathbb{N} \,\big|\, r(r+1) \leq m\}.$$

What does that imply for our problem ?

Suppose we have an SDP of the form:

$$\begin{aligned} \min_{X \in S^n} \quad & \operatorname{trace}(A\,X) \\ & \operatorname{trace}(B_i\,X) = c_i, \quad i = 1, \ldots, m \\ & X \succeq 0 \end{aligned}$$

Theorem 2 (Pataki(1998)). *If X is an extreme point of the above SDP, then* $\operatorname{rank}(X) \leq r_m$, *where*

$$r_m = \max\{r \in \mathbb{N} \,\big|\, r(r+1) \leq m\}.$$

What does that imply for our problem ?

- The optimum is attained at an extreme point (linear objective !)
- turn the problem into a problem with two equality constraints
  all extreme points have rank one $\Longrightarrow$ minimizer has rank one.