Predictive low-rank decomposition for kernel methods

Francis Bach
Ecole des Mines de Paris

Michael Jordan
UC Berkeley

ICML 2005

Predictive low-rank decomposition for kernel methods

- Kernel algorithms and low-rank decompositions
- Incomplete Cholesky decomposition
- Cholesky with side information
- Simulations – code online

Kernel matrices

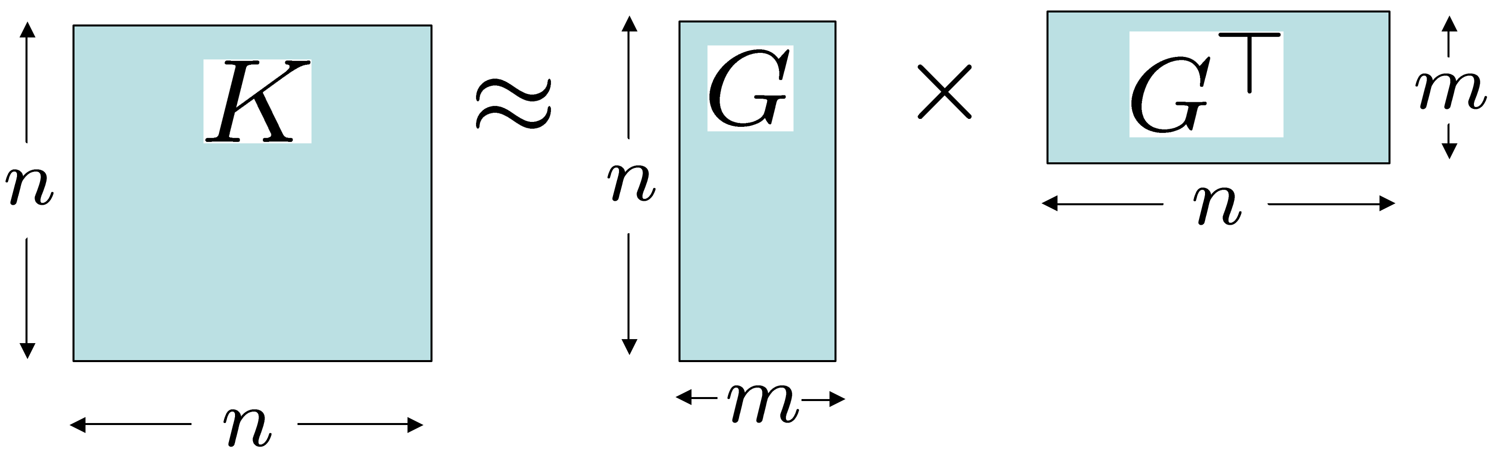
- Given
  - n data points $x_i \in \mathcal{X}$
  - kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
- Kernel methods works with kernel matrix $K \in \mathbb{R}^{n \times n}$
  - defined as a Gram matrix : $K_{ij} = k(x_i, x_j)$
  - symmetric : $K = K^\top$
  - positive semi-definite : $K \succcurlyeq 0$

Kernel algorithms

- Kernel algorithms, usually $O(n^3)$ or worse
  - Eigenvalues: Kernel PCA, CCA, FDA
  - Matrix inversion: LS-SVM
  - Convex optimization problems: SOCP, QP, SDP
- Requires speed-up techniques for medium/large scale problems
- General purpose matrix decomposition algorithms:
  - Linear in n (not even touching all entries!)
    - Nyström method (Williams & Seeger, 2000)
    - Sparse greedy approximations (Smola & Schölkopf, 2000)
    - Incomplete Cholesky decomposition (Fine & Scheinberg, 2001)

Incomplete Cholesky decomposition

$$K \approx GG^\top, \;\; G \in \mathbb{R}^{n\times m}, \;\; m \ll n$$



– m is the rank of G

– Most algorithms become $O(m^3 + m^2 \underline{\underline{n}})$

Kernel matrices and ranks

- Kernel matrices may have full rank, i.e., $m = n$ ...
- ... but eigenvalues decay (at least) exponentially fast for a wide variety of kernels (Williams & Seeger, 2000, Bach & Jordan, 2002)

  $\Longrightarrow$ Good approximation by low rank matrices with small m

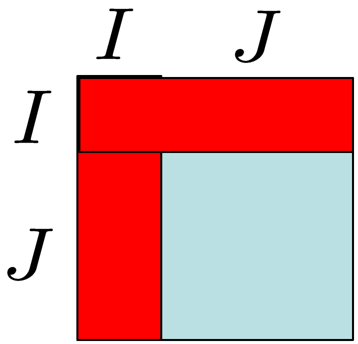
- "Data live near a low-dimensional subspace in feature space"
- In practise, very small m

Incomplete Cholesky decomposition

$$K \approx GG^\top, \;\; G \in \mathbb{R}^{n \times m}, \;\; m \ll n$$

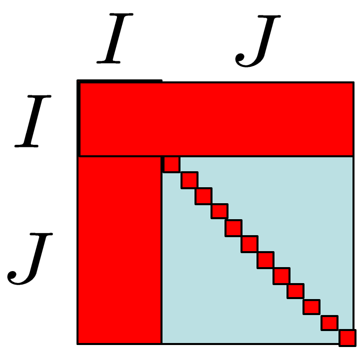
- Approximate full matrix from selected columns:

  ($\Leftrightarrow$ use datapoints in I to approximate all of them)



$$I \cup J = \{1, \ldots, n\}$$

- Use diagonal to characterize behavior of the unknown block



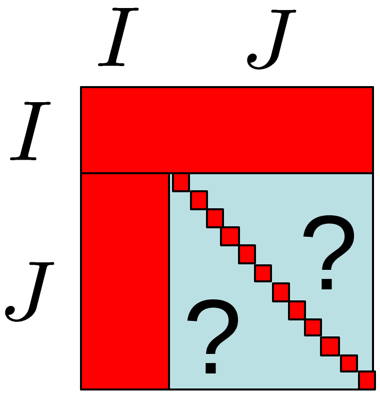
$$I \cup J = \{1, \ldots, n\}$$

Lemma

- Given a positive matrix K and subsets $I \cup J = \{1, \ldots, n\}$
- There exists a unique matrix L such that
  - L is symmetric
  - The column space of L is spanned by $K(:, I)$
  - L agrees with K on columns in I

$$L([I \;\; J], [I \;\; J]) = \begin{pmatrix} K(I,I) & K(J,I)^\top \\ K(J,I) & K(J,I)K(I,I)^{-1}K(J,I)^\top \end{pmatrix}$$

Incomplete Cholesky decomposition



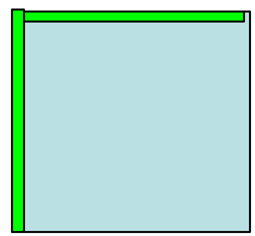
- Two main issues:
  - Selection of columns I (*pivots*)
  - Computation of

$$L(J,J) = K(J,I)K(I,I)^{-1}K(J,I)^\top$$
$$= \sum_{i \in I} G(i,:)G(i,:)^\top \text{ if } L = GG^\top$$

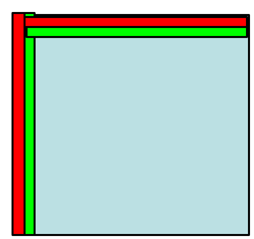
- Incomplete Cholesky decomposition
  - Efficient update of G with linear cost
  - Pivoting: greedy choice of pivot with linear cost

Incomplete Cholesky decomposition (no pivoting)

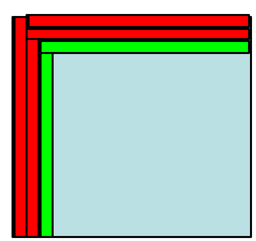
k=1



k=2



k=3



Pivot selection

- $G_k \in \mathbb{R}^{n \times k}$ approximation after k-th iteration
- Error $\|K - G_k G_k^\top\|_1 = \mathrm{tr}(K - G_k G_k^\top)$

$$= \mathrm{tr}(K) - \sum_{j=1}^{k} \|G(:,j)\|_2^2$$

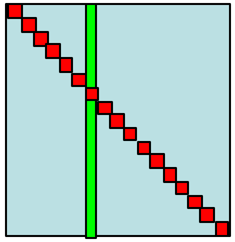
- Gain after between iterations k-1 and k = $\|G(:,k)\|_2^2$
- Exact computation is $O(kn(n-k))$
- Lower bound $\|G(:,k)\|_2^2 \geqslant G(i_k,k)^2 = D(i_k)$

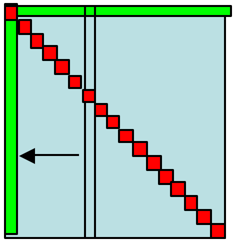
Incomplete Cholesky decomposition with pivoting

Pivot selection

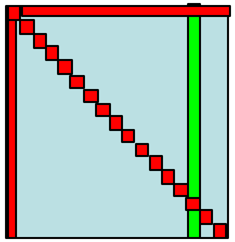
Pivot permutation

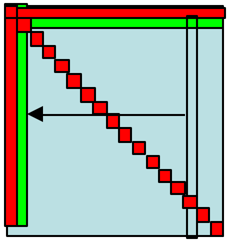
k=1



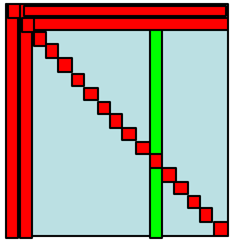


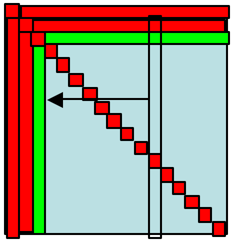
k=2





k=3





Incomplete Cholesky decomposition: what's missing?

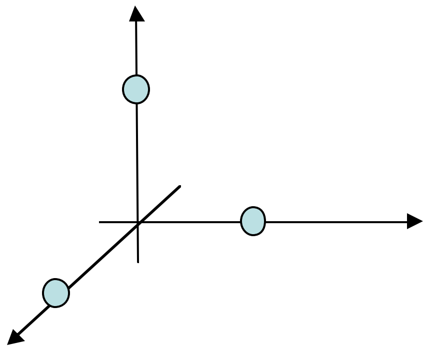
- Complexity after m steps: $O(m^2n)$
- What's wrong with incomplete Cholesky (and other decomposition algorithms)?
  - They don't take into account the classification labels or regression variables
  - cf. PCA vs. LDA

Incomplete Cholesky decomposition: what's missing?

- Two questions:
  - Can we exploit side information to lower the needed rank of the approximation?
  - Can we do it in linear time in n ?

Using side information
(classification labels, regression variables)

- Given
  - kernel matrix $K \in \mathbb{R}^{n \times n}$
  - side information $Y \in \mathbb{R}^{n \times d}$
    - Multiple *regression* with d response variables
    - *Classification* with d classes
      - $Y_{ni} = 1$ if n-th data point belongs to class i
      - 0 otherwise



- Use Y to select pivots

Prediction criterion

- Square loss: $c(y,f) = \frac{1}{2}\|y-f\|_2^2, \;\; y,f \in \mathbb{R}^d$
- Representer theorem: prediction using kernels leads to prediction error for i-th data point $\|y_i - (K\alpha)_i\|_2^2$ where $\alpha \in \mathbb{R}^n$
- Minimum total prediction error

$$\min_{\alpha \in \mathbb{R}^{n\times d}} \frac{1}{2}\|Y - K\alpha\|_F^2$$

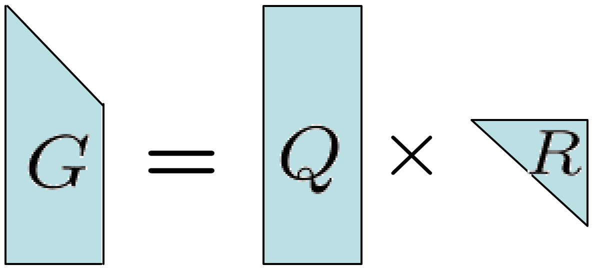
- If $K = GG^\top$, equal to

$$\min_{\beta \in \mathbb{R}^{m\times d}} \frac{1}{2}\|Y - G\beta\|_F^2 = \frac{1}{2}\mathrm{tr}\left\{Y^\top (I - G(G^\top G)^{-1}G^\top)Y\right\}$$

Computing/updating criterion

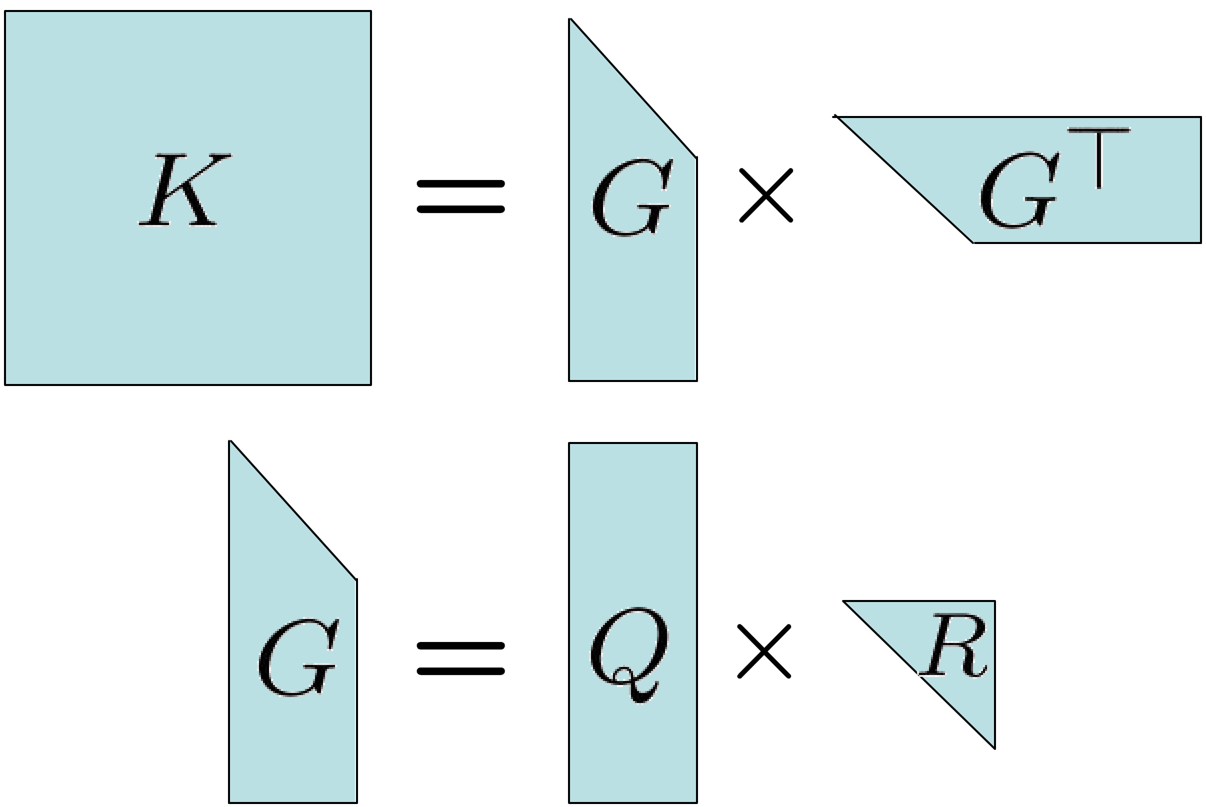
$$\min_{\beta \in \mathbb{R}^{m \times d}} \frac{1}{2} \|Y - G\beta\|_F^2 = \frac{1}{2} \mathrm{tr} \left\{ Y^\top (I - G(G^\top G)^{-1} G^\top) Y \right\}$$

- Requirements: efficient to add one column at a time
  - (cf linear regression setting: add one variable at at time)
- QR decomposition of $G \in \mathbb{R}^{n \times m}$
  - $G = QR$
  - $Q \in \mathbb{R}^{n \times m}$ orthogonal, $R \in \mathbb{R}^{m \times m}$ upper triangular
  - $G(G^\top G)^{-1} G^\top = QQ^\top = \sum_k Q(:,k) Q(:,k)^\top$



Cholesky with side information (CSI)

- Parallel Cholesky and QR decomposition



- Selection of pivots?

Criterion for selection of pivots

- Approximation error + prediction error

$$\lambda \mathrm{tr}(K - GG^\top) + \mu \mathrm{tr}\left\{Y^\top Y - Y^\top G(G^\top G)^{-1} G^\top Y\right\}$$

- Gain in criterion after k-th iteration:

$$\lambda \|G(:,k)\|_2^2 + \mu \|Y^\top Q(:,k)\|_2^2$$

- Cannot compute for each remaining pivot exactly because it requires the entire matrix
- Main idea: compute δ "look-ahead" decomposition steps and use the decomposition to compute gains
  - δ large enough to gain enough information about K
  - δ small enough to incur little additional cost

Incomplete Cholesky decomposition with pivoting and look-ahead

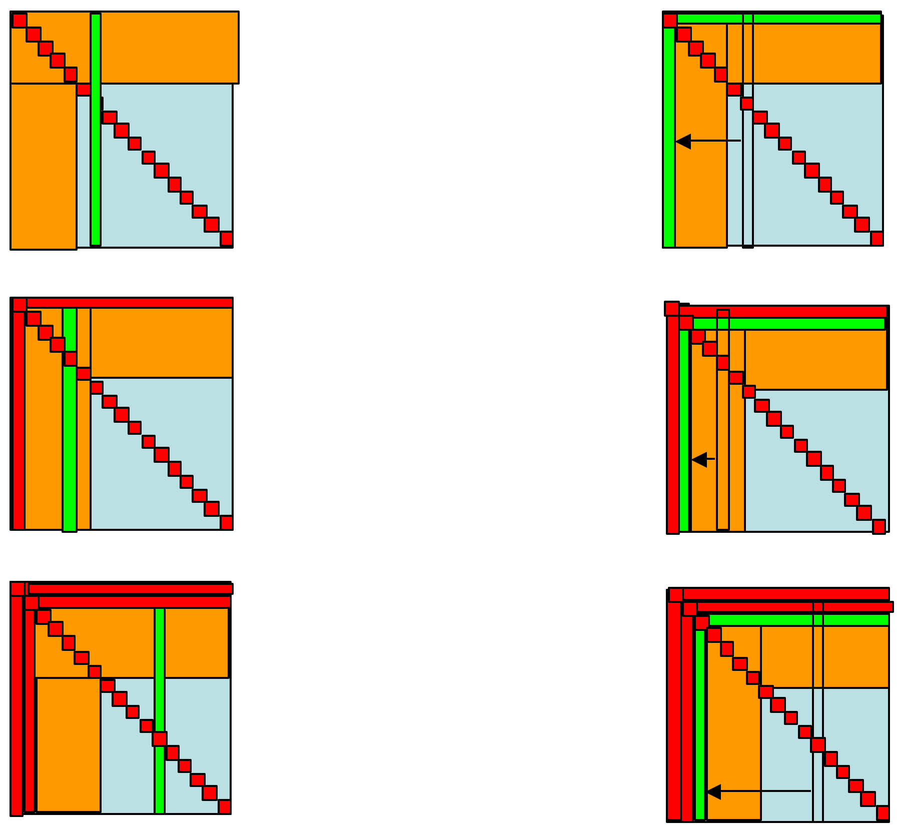
Pivot selection

Pivot permutation

k=1

k=2

k=3

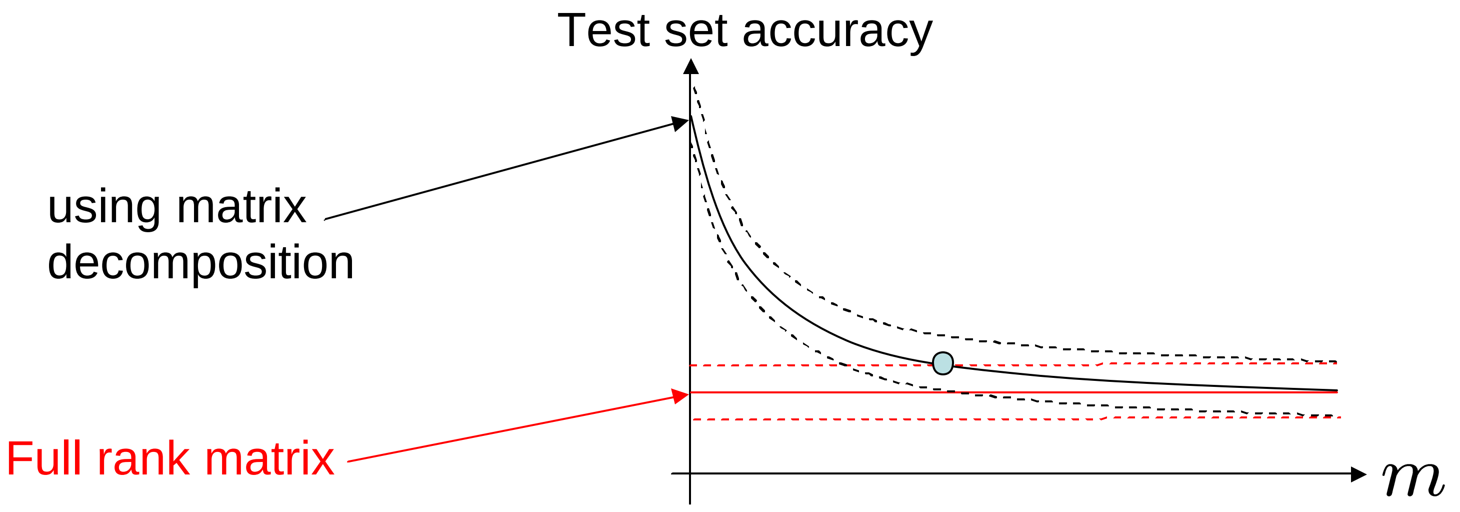


Running time complexity

- "Semi-naïve" computations of look-ahead decompositions (i.e., start again from scratch at each iteration)
  - Decompositions: $O(mn\delta(m+\delta))$
  - Computing criterion gains: $O(ndm\delta)$
- Efficient implementation (see paper/code)
  - $m+\delta$ steps of Cholesky/QR: $O((m+\delta)^2 n)$
  - Computing criterion gains: $O(mdn)$

Simulations

- UCI datasets
- Gaussian-RBF kernels – Least squares SVM
- Width and regularization parameters chosen by cross-validation
- Compare minimal ranks for which the average performance is within a standard deviation from the one with the full kernel matrix



Simulations

dataset	n_f	n_c	n_p	Chol	CSI
ringnorm	20	2	1000	14	3
kin-32fh-c	32	2	2000	25	6
pumadyn-32nm	32	–	4000	93	23
pumadyn-32fh	32	–	4000	30	8
kin-32fh	32	–	4000	34	10
bank-32fh	32	–	4000	221	72
page-blocks	8	2	5473	451	155
spambase	49	2	4000	90	31
isolet	617	8	1798	254	89
twonorm	20	2	4000	8	3
comp-activ	21	–	4000	159	73
abalone	10	–	4000	27	13
kin-32nm-c	32	2	4000	122	68
pendigits	16	4	4485	111	63
kin-32nm	32	–	2000	307	211
add10	10	–	2000	280	204
mushroom	116	2	4000	60	44
bank-32-nm	32	–	4000	413	328
vehicle	18	2	416	31	27
boston	12	–	506	48	61

Conclusion

- Discriminative kernel methods and …

  … discriminative matrix decomposition algorithms
- Same complexity as non discriminative version (linear)
- Matlab/C code available online