An Introduction to $H$-matrices

Zachary Bookey

March 27, 2017
When dealing with a system of $n$ equations the optimal efficiency is $O(n)$. With dense matrices with $n^2$ entries $O(n^2)$ seems unavoidable.

This paper proposes using a data sparse representation of a matrix.

This representation allows for cheap operations such as:

- Matrix addition
- Matrix-vector multiplication
- Matrix-matrix multiplication
- Matrix inversion
Partitioning a Vector

Definitions

Space of vectors $a = (a_i)_{i \in I}$

$I$ is finite index set. e.g., $I = \{1, \ldots, n\}$

Partition $P = \{I_j : 1 \leq j \leq k\}$

$I = \bigcup_{j=1}^{k} I_j$

Example

$a = \{2, 1, 4, 0\}$

$I = \{1, 2, 3, 4\}$

Partition into equal blocks.

$I_1 = \{1, 2\}, I_2 = \{3, 4\}$

Block 1: $\{2, 1\}$ Block 2: $\{4, 0\}$
Tree Terminology

$T$ is a tree.

- $S(t) := \{s \in T : s \text{ is son of } t\}$ for $t \in T$
- $L(T) := \{t \in T : S(t) = \emptyset\}$
- The root of $T$ is the unique vertex without a parent
- $S^*(t) := \{s \in T : \text{ there is a directed path from } t \text{ to } s\}$
$T$ is an $H$-tree of an index set $I$ if the following hold:

1. All vertices $t \in T$ are subsets of $I$
2. $I \in T$
3. $\forall t \in T, |S(t)| \neq 1$
4. If $t \in T$ and $t \notin L(T)$ then $S(t)$ contains disjoint subsets of $I$ and $t$ is the union of its sons.

$$t = \bigcup_{s \in S(t)} s$$
Important properties of the $H$-tree

1. $s, t \in T$ with $s \neq t$ Then exactly one of the following is true:
   - $s \subset t$. Then $s \in S^*(t) \setminus \{t\}$
   - $t \subset s$. Then $t \in S^*(s) \setminus \{s\}$
   - $s \cap t = \emptyset$. Then there is a unique smallest $r \in T$ with $s, t \in S^*(r)$

2. For any $t \in T$, $S^*(t)$ is a subtree of $T$ satisfying requirements 1, 3, and 4 for being a $H$-tree.

3. For any $t \in T$,

$$t = \bigcup_{s \in L(S^*(t))} s$$
An example of an $H$-tree over an index set $I$
\{1, 2, 3, 4, 5, 6, 7, 8\}
The Matrix Case

With a block partitioning $P$ of $I$, the traditional partitioning is:

$$P_2 := P \times P = \{ I_i \times I_j : I_i, I_j \in P \}$$

Figure of a sample tensor partition

![Sample tensor partition diagram]
A Non-Tensor Partition

Idea

Use an $H$-tree to recursively define a partition $P_2 = P_2(I, T)$ of $I \times I$

Recursion depends on the depth of the $H$-tree.

- Depth $= 0$, $P_2(I, T) := \{I \times I\}$
- Depth $= 1$, $P_2(I, T) := \{I_1 \times I_1, I_1 \times I_2, I_2 \times I_1, I_2 \times I_2\}$
- Depth $> 1$, $P_2(I, T) := P_2(I_1, T_1) \cup \{I_1 \times I_2\} \cup \{I_2 \times I_1\} \cup P_2(I_2, T_2)$
$p = 0: \quad , \quad p = 1: \quad , \quad p = 2: \quad , \quad p = 3: \quad$
Another Recursive Definition

A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \text{ with } \frac{n}{2} \times \frac{n}{2} H\text{-Matrices } A_{ii}

Definition

\[ M_{H,k}(I \times I, P_2) := \{ M : \text{each block } M^b, b \in P_2, \text{satisfies } \text{rank}(M^b) \leq k \} \]

A matrix A is an Rk-matrix if \( \text{rank}(A) \leq k \).
Contents

1 Introduction

2 Partitioning and $H$-Trees

3 Rk-Matrices

4 $H$-Matrices

5 Another Example
Any $n \times m$ matrix $A$ of rank $\leq 1$ can be written of the form:

$$A = a \ast b^H$$ (notation: $A = [a, b]$)

Only need to store the vectors $a$ and $b$. $O(n + m)$ storage
Properties of R1-Matrices

- The amount of work for matrix-vector multiplication $Ac(c \in K^m)$ is $2m - 1$ operations to obtain $\alpha \star a$ and $2m + n - 1$ if $\alpha \star a$ is performed explicitly.
- $A = [a, b]$ then also $A^H = [b, a]$.
- R1-Matrices have left and right ideal properties.
  - $B \star A$ with $A = [a, b]$ requires only $B \star a$ and $B \star A = [B \star a, b]$.
  - If $A$ and $B$ are R1-Matrices $A \star B$ needs only one scalar product.
- The evaluation of any entry $A_{ij}$ requires exactly one operation.
- The evaluation of a complete row or column requires either $m$ or $n$ operations.
In general the sum of two R1-matrices is not an R1-matrix.
To get an approximate sum that is an R1-matrix we use the SVD

\[ A = U \times D \times V \]

Let \( D' \) be a version of \( D \) with only the \( k' \) largest singular values.
\( A' = U \times D' \times V \) has rank \( k' \) and \( \| A - A' \|_F \) is minimized.

Costs \( 9(n + m) + O(1) \) operations.
These properties hold when we extend to Rk-Matrices:

- Storage for $n \times m$ Rk-matrix is $O(n + m)$
- Product $A \star B$ requires $k^2$ scalar products
- $A +_{Rk} B$ requires solving a $k \times k$ eigenvalue problem with an additional $O(1)$
$H$-Partition

Each block is filled with an Rk matrix
Complexity of the $H$-Matrix

**Storage**

$n \times n$ $H$-matrix where $n = 2^p$ requires $(2 \log_2 n + 1)n$

**$+_R 1$**

The $R1$ addition of two $H$-matrices or an $R1$-matrix and $H$-matrix is $18n \log_2 n + O(n)$
Complexity of the $H$-Matrix

**Matrix-Vector Multiplication**

The multiplication of an $H$-matrix and a general vector is $4n \log_2 n - n + 2$

**Matrix-Matrix Multiplication**

- Two $H$-matrices: $9n \log_2^2 n + O(n \log_2 n)$
- $H$-matrix and an R1-matrix: $4n \log_2 n - n + 2$
- Two R1-matrices: $3n - 1$
Complexity of the $H$-Matrix

Approximate Matrix-Inversion

The approximate matrix inversion takes: $\frac{29}{2} n \log_2^2 n + O(n \log_2 n)$

LU-Decomposition

LU-Decomposition takes: $6 n \log_2^2 n + O(n \log_2 n)$
Properties of the $H$-Matrix

Most operations on $H$-matrices are approximations.

- Matrix-Vector multiplication $Ax$ where $A$ is an $H$-matrix is exact.
- If $A$ is an $H$-matrix and $B$ is an $Rk$ matrix, then $AB$ and $BA$ are again $Rk$-matrices.
Properties of the \( H \)-Matrix

The field of \( H \)-matrices with rank \( k \) blocks are invariant with respect to diagonal scaling.

### Banded Matrices

Tridiagonal matrices and their inverses both belong to the field of \( H \)-matrices with the partition described earlier. Similarly band matrices with \( 2k \) off diagonals are belong to the field.
A Second Example

The previous partition may not be dense enough around the diagonal for practical use.

$N_k$-Matrices

Neighborhood matrices

\[
A = \begin{bmatrix}
A_{11} & A_{12} \\
A_{21} & A_{22}
\end{bmatrix} \text{ with } \frac{n}{2} \times \frac{n}{2} \text{ Rk matrices } A_{11}, A_{12}, A_{22} \text{ and } A_{21} \in N_k
\]
A Second Example

\[ A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \text{ with } A_{11}, A_{22} \in H-\text{Matrices}, A_{12} \in N_k, A_{21} \in N_k^T \]
A Second Example

Similar complexity to the original example.

- \( N_{\text{block}} = 9n - 6 \log_2 n - 8 \)
- \( N_{\text{storage}} = 6n \log_2 n + O(n) \)
- Addition: \( O(n \log_2 n) \)
- Matrix-vector: \( 11n \log_2 n + O(n) \)
- Matrix-matrix: \( O(n \log^2 n) \)
- Inversion: \( O(n \log^2 n) \)
Wrap up

- Approximation to dense matrices
- Uses recursive partitioning defined by T-partitions
- Most operations are almost linear time
- Applications:
  - Approximate integral operators through discretization