Use of Structure

• The use of structure is a major theme in numerical analysis.

• Example: non-singular square linear systems

\[ Ax = b \]

• In principle we can solve any such system:
  1. Factor

\[ A = PLU \]

  where \( P \) is a permutation matrix, \( L \) is unit lower triangular, and \( U \) is upper triangular.

  2. Solve

\[ Ly = P^Tb \]

  3. Solve

\[ Ux = y \]

• In fact, there is a school of thought that the purpose of numerical analysis is to make itself obsolete, by inventing methods that can solve any problem.

• It does not work like that, scientists are constantly coming up with new and bigger problems.

• Much work in algorithm design relates to exploiting special structure.

• This can make it possible to solve linear systems
  – faster
  – more accurately
  – using less memory
  – or, quite possibly, at all
• The principle applies to all kinds of numerical problems, not just $Ax = b$.

• Example of some pretty contrived special structure, Mike Hohn MS thesis:

\[
A = \begin{bmatrix}
S & C^T \\
C & 0
\end{bmatrix}
\]

where

$S$ is sparse (few non-zero entries), symmetric, positive semi-definite, and rank deficient.

$C$ is sparse

$A$ is large and non-singular.
Positive Definite Matrices

- A matrix $A$ is **positive definite** if it is symmetric\(^{-1}\) and $x^T Ax > 0$ for all vectors $x \neq 0$.

- Here are some facts about positive definite matrices:
  - The diagonal entries of a p.d. matrix are positive.
  - The eigenvalues of a p.d. matrix are real and positive.
  - A symmetric matrix whose eigenvalues are all positive is p.d.
  - The determinant of a p.d. matrix is positive.
  - A symmetric matrix may have a positive determinant but not be p.d. Example:
    - Some entries of a p.d. matrix may be negative. Example:
    - A symmetric matrix with entries that are all positive may not be p.d. Example:

\(^{-1}\) Golub/van Loan drop the requirement of symmetry.
• A matrix $A$ is **positive semi-definite** if it is symmetric and $x^T Ax \geq 0$ for all vectors $x$.

• similar definitions for **negative definite** and **negative semi-definite**.

**Exercise:** modify the above statements of fact for positive definite, and negative definite or semi-definite matrices.

• We saw at the beginning of the semester that positive definite matrices occur naturally when solving minimization problems, and that the positive definiteness of matrices is the natural generalization of the positivity of numbers.

• Idea: Factor

$$A = LL^T$$

(1)

where $L$ is lower triangular (not necessarily unit lower triangular).

• The factorization (1) is called the **Cholesky decomposition** (or **factorization**) of $A$.

• Any product $A = LL^T$, for non-singular $L$, is symmetric and positive definite since

$$A^T = (LL^T)^T = (L^T)^T L^T = LL^T = A$$

and, for $x \neq 0$

$$y = L^T x \neq 0 \quad \text{and} \quad x^T Ax = x^T LL^T x = y^T y > 0$$

• So we have seen that if $L$ is a lower triangular matrix then $LL^T$ is positive definite.

• Can we go the other way? If $A$ is positive definite, does the Cholesky decomposition always exist?

• Yes!

---

**André-Louis Cholesky, 1875–1918, was a French military officer and mathematician.**
• I’ll give two proofs, one by induction, and one algorithmically.

**Existence by induction**

• The induction proof illustrates another major technique in linear algebra, the use of block matrices. A block matrix is a matrix whose entries are matrices.

• Loosely speaking the main fact about block matrices is that if the dimensions of their entries line up you can treat them essentially like ordinary matrices. **Exercise:** Make this statement precise and convince yourself that it is true.

• Consider first the case $n = 1$. Then

$$A = [a]$$

and $A$ is positive definite if and only if $a > 0$. In that case we can write

$$A = LL^T \quad \text{where} \quad L = [\sqrt{a}].$$

• Now suppose we have established the result for all (symmetric positive definite) $(n - 1) \times (n - 1)$ matrices and suppose $A = n \times n$. Then we write $A$ as the block matrix

$$A = \begin{bmatrix} A_{n-1} & a \\ a^T & c \end{bmatrix}$$

where $A_{n-1}$ is $(n - 1) \times (n - 1)$, $a \in \mathbb{R}^{n-1}$, and $c$ is a constant.

• It’s easy to see that $A_{n-1}$ is positive definite and $c > 0$.

• By the induction hypothesis we can write

$$A_{n-1} = L_{n-1}L_{n-1}^T$$

where $L_{n-1}$ is lower triangular.
• So let us see if we can find a vector $r \in \mathbb{R}^{n-1}$ and a constant $\xi$ such that

$$A = LL^T$$

where

$$L = \begin{bmatrix} L_{n-1} & 0 \\ r^T & \xi \end{bmatrix}$$

• Computing $LL^T$ as usual we get

$$L = \begin{bmatrix} L_{n-1} & 0 \\ r^T & \xi \end{bmatrix} \begin{bmatrix} L_{n-1}^T & r \\ 0^T & \xi \end{bmatrix} = \begin{bmatrix} A_{n-1} & a \\ a^T & c \end{bmatrix} = A$$

• We get the requirements
  - $L_{n-1}L_{n-1}^T = A_{n-1}$ true, good
  - $L_{n-1}r = a$ We can solve this equation for $r$ since $L_{n-1}$ is non-singular by the induction hypothesis.
  - $\xi^2 = c - r^Tr$.
  - We need to show that $c - r^Tr$ is positive so we can set
    $$\xi = \sqrt{c - r^Tr}.$$

To see this note that even if $c - r^Tr$ was negative we’d get the factorization we want, except that $\xi$ would be imaginary. In either case we can compute the following determinants (indicated by vertical bars). We get

$$|L_n| = |L_{n-1}|\xi$$

and hence

$$|L_nL_n^T| = |A| = |L_n|^2 = |L_{n-1}|^2\xi^2 > 0$$

since $A$ is positive definite. This implies that $\xi^2 = c - r^Tr > 0$ as required.

• This proof can actually be turned into an algorithm for computing the Cholesky Decomposition, but there is an easier way that will provide an additional important insight.
Existence by Construction

- We want
\[ LL^T = A \]  \hspace{1cm} (2)

where \( L = [l_{ij}] \) is lower triangular. This means that
\[ k > i \implies l_{ik} = 0. \]

- Now consider the \((i, j)\) entry \(a_{ij}\) of \(A\) obtained by (2):
\[ a_{ij} = \sum_{k=1}^{n} l_{ik} l_{jk} = \sum_{k=1}^{\min\{i,j\}} l_{ik} l_{jk}. \]

- Hence
\[ a_{11} = l_{11}^2 \implies l_{11} = \sqrt{a_{11}} \]
and
\[ a_{i1} = l_{i1} l_{11} \implies l_{i1} = \frac{a_{i1}}{l_{11}} \]

- This gives us the first column of \(L\). To get the second column we proceed as follows, noting that we can use the entries in the first column:
\[ a_{22} = l_{21}^2 + l_{22}^2 \implies l_{22} = \sqrt{a_{22} - l_{21}^2} \]
and
\[ a_{i2} = l_{i1} l_{21} + l_{i2} l_{22} \implies l_{i2} = \frac{a_{i2} - l_{i1} l_{21}}{l_{22}} \]
• Continuing in this fashion we obtain the general algorithm:

**Cholesky:**

For \( k = 1, \ldots, n \)

\[
l_{kk} = \sqrt{a_{kk} - \sum_{i=1}^{k-1} l_{ki}^2}
\]

For \( i = k + 1, \ldots, n \)

\[
l_{ik} = \frac{a_{ik} - \sum_{j=1}^{k-1} l_{ij} l_{kj}}{l_{kk}}
\]

• What numerical effort do you expect for this algorithm?
Numerical Stability

• Recall the equation

\[ a_{ii} = \sum_{j=1}^{i} l_{ij}^2 \]

• This implies that for all \( a \) and \( j \)

\[ |l_{ij}| \leq \sqrt{a_{ii}} \]

• The \( l_{ij} \) are bounded!

• This means that pivoting is not required when computing the Cholesky Decomposition!

• That computation is numerically stable!

• Of course, if we did pivot we would use simultaneous row and column interchanges to maintain symmetry.
Pivoting to reduce fill-in

- This is a very brief introduction to a fascinating idea: Use pivoting to minimize or reduce the fill-in when applying direct methods to large sparse positive definite linear systems.


- (Recently George announced a forthcoming 2nd edition of this book).

- If we don’t need to pivot for stability, can we pivot for something else?

- The problem with Gaussian Elimination for large sparse systems is fill-in, the introduction of new non-zero matrix elements.

- For a positive definite system we can pivot to reduce fill-in.
• Here is an extreme example. Compare

\[
A = \begin{bmatrix}
x & x & x & x & x & x \\
x & x & 0 & 0 & 0 & 0 \\
x & 0 & x & 0 & 0 & 0 \\
x & 0 & 0 & x & 0 & 0 \\
x & 0 & 0 & 0 & x & 0 \\
x & 0 & 0 & 0 & 0 & x \\
\end{bmatrix}
\]

with

\[
B = \begin{bmatrix}
x & 0 & 0 & 0 & 0 & x \\
0 & x & 0 & 0 & 0 & x \\
0 & 0 & x & 0 & 0 & x \\
0 & 0 & 0 & x & 0 & x \\
0 & 0 & 0 & 0 & x & x \\
x & x & x & x & x & x \\
\end{bmatrix}
\]

• The matrix \( B \) can be obtained from the matrix \( A \) by interchanging the first and last rows and columns.

• Thinking for simplicity in terms of straight Gaussian Elimination (the observation carries over to the Cholesky Decomposition) it is clear that for matrix \( A \) we get complete fill-in, and for matrix \( B \) we get zero fill-in.

• This is the tip of a large ice-berg, but we need to move on.