the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

MCS 471 Lecture 12
Numerical Analysis
Jan Verschelde, 19 September 2022

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

symmetric matrices

Definition

A matrix A is *symmetric* if $A^T = A$.

$\cdot^T$ is the *transpose*, defined by flipping all elements over the diagonal:

If the (i,j) element of A is $a_{i,j}$, then the (i,j) element of A^T is $a_{j,i}$.

Example:

$$A = \begin{bmatrix} 5 & 6 & 0 \\ 2 & 8 & 3 \\ 1 & 7 & 9 \end{bmatrix}, \quad A^T = \begin{bmatrix} 5 & 2 & 1 \\ 6 & 8 & 7 \\ 0 & 3 & 9 \end{bmatrix}.$$

The rows (columns) of A are the columns (rows) of A^T.

If L is the lower triangular part of a symmetric matrix A,
then the upper triangular part of A is L^T.

positive semidefinite matrices

Definition

A matrix A is *positive semidefinite* if for all $\mathbf{x}$: $\mathbf{x}^T A\mathbf{x} \geq 0$.

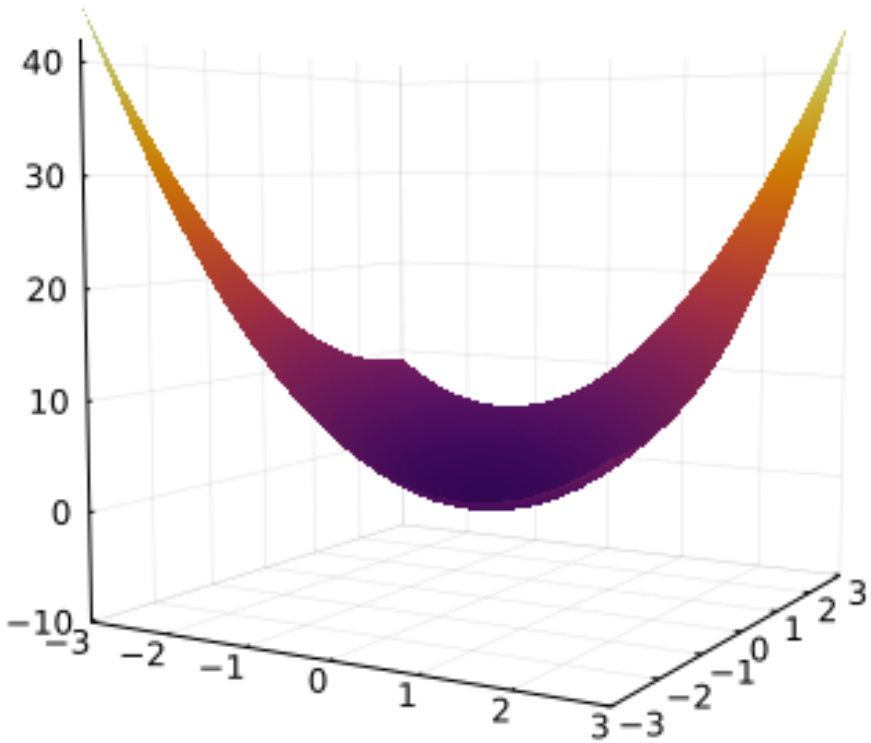
Example: Consider

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

Then

$$\begin{aligned} \mathbf{x}^T A\mathbf{x} &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 2x_1 + x_2 \\ x_1 + 2x_2 \end{bmatrix} \\ &= x_1(2x_1 + x_2) + x_2(x_1 + 2x_2) \\ &= 2x_1^2 + 2x_1x_2 + 2x_2^2 \end{aligned}$$

a geometric interpretation



The plot of $f(x_1, x_2) = 2x_1^2 + 2x_1x_2 + 2x_2^2$
shows that $(0, 0)$ is the minimum of the surface $z = f(x, y)$.

eigenvalues and eigenvectors

Definition

A value λ and a nonzero vector $\mathbf{v}$ are *an eigenvalue, eigenvector pair* $(\lambda, \mathbf{v})$ of a matrix A if $A\mathbf{v} = \lambda\mathbf{v}$.

Example: Consider

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad \lambda = 1, \quad \mathbf{v} = \begin{bmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{bmatrix}.$$

Let us verify this:

$$\begin{aligned} A\mathbf{v} &= \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{bmatrix} \\ &= \begin{bmatrix} -2\sqrt{2}/2 + \sqrt{2}/2 \\ -\sqrt{2}/2 + 2\sqrt{2}/2 \end{bmatrix} = \begin{bmatrix} -\sqrt{2}/2 \\ \sqrt{2}/2 \end{bmatrix} = \lambda\mathbf{v}. \end{aligned}$$

eigenvalues of positive semidefinite matrices

Theorem

If all eigenvalues of A are nonnegative, then A is positive semidefinite.

Consider a 2-by-2 matrix with 2 distinct eigenvalues λ and μ, and corresponding eigenvectors $\mathbf{v}$ and $\mathbf{w}$: $A\mathbf{v} = \lambda\mathbf{v}$ and $A\mathbf{w} = \mu\mathbf{w}$.

The vectors $\mathbf{v}$ and $\mathbf{w}$ form a basis for the vector space, so for any $\mathbf{x}$ we may write $\mathbf{x} = \alpha\mathbf{v} + \beta\mathbf{w}$, for some numbers α and β.

Assuming $\lambda \geq 0$ and $\mu \geq 0$, we check if A is positive semidefinite:

$$
\begin{aligned}
\mathbf{x}^T A\mathbf{x} &= (\alpha\mathbf{v} + \beta\mathbf{w})^T A(\alpha\mathbf{v} + \beta\mathbf{w}) \\
&= (\alpha\mathbf{v} + \beta\mathbf{w})^T (\alpha A\mathbf{v} + \beta A\mathbf{w}) \\
&= (\alpha\mathbf{v} + \beta\mathbf{w})^T (\alpha\lambda\mathbf{v} + \beta\mu\mathbf{w}) \\
&= \lambda\alpha^2 \mathbf{v}^T\mathbf{v} + \mu\beta^2 \mathbf{w}^T\mathbf{w} \\
&= \lambda\alpha^2 (v_1^2 + v_2^2) + \mu\beta^2 (w_1^2 + w_2^2) \geq 0.
\end{aligned}
$$

eigenvalues in a Julia session

```
julia> using LinearAlgebra

julia> A = reshape([2.0, 1.0, 1.0, 2.0], (2, 2))
2×2 Array{Float64,2}:
 2.0  1.0
 1.0  2.0

julia> eigvals(A)
2-element Array{Float64,1}:
 1.0
 3.0
```

The matrix $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ has eigenvalues 1 and 3.

eigenvectors in a Julia session

```
julia> v = eigvecs(A)
2×2 Array{Float64,2}:
 -0.707107  0.707107
  0.707107  0.707107

julia> A*v
2×2 Array{Float64,2}:
 -0.707107  2.12132
  0.707107  2.12132

julia> 3*v[:,2]
2-element Array{Float64,1}:
 2.1213203435596424
 2.1213203435596424
```

factoring symmetric positive definite matrices

If A is symmetric and positive semidefinite,
then we can compute the *Cholesky factorization* $A = LL^T$,
where L is a lower triangular matrix.

Two advantages over LU factorization:

1. No pivoting is needed because A is positive semidefinite.
2. Because $U = L^T$, only half as many operations are needed.

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

a random positive semidefinite matrix

```
julia> using LinearAlgebra

julia> L = LowerTriangular(rand(2,2))
2×2 LowerTriangular{Float64,Array{Float64,2}}:
 0.266687   ·
 0.671371  0.981431

julia> A = L*transpose(L)
2×2 Array{Float64,2}:
 0.0711222  0.179046
 0.179046   1.41395

julia> eigvals(A)
2-element Array{Float64,1}:
 0.047658946405909675
 1.437408604382954
```

We see that the eigenvalues of `A` are all positive.

verifying by random example

The session continues ...

```
julia> x = rand(2,1)
2×1 Array{Float64,2}:
 0.9349208559205571
 0.7658920207613471

julia> x[1] = -x[1];

julia> x
2×1 Array{Float64,2}:
 -0.9349208559205571
  0.7658920207613471

julia> transpose(x)*A*x
1×1 Array{Float64,2}:
 0.6351617921263094
```

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

derivation of the formulas

$$\begin{bmatrix} a_{1,1} & a_{2,1} & a_{3,1} \\ a_{2,1} & a_{2,2} & a_{3,2} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} = \begin{bmatrix} \ell_{1,1} & 0 & 0 \\ \ell_{2,1} & \ell_{2,2} & 0 \\ \ell_{3,1} & \ell_{3,2} & \ell_{3,3} \end{bmatrix} \begin{bmatrix} \ell_{1,1} & \ell_{2,1} & \ell_{3,1} \\ 0 & \ell_{2,2} & \ell_{3,2} \\ 0 & 0 & \ell_{3,3} \end{bmatrix}$$

$$= \begin{bmatrix} \ell_{1,1}^2 & \ell_{1,1}\ell_{2,1} & \ell_{1,1}\ell_{3,1} \\ \ell_{2,1}\ell_{1,1} & \ell_{2,1}^2 + \ell_{2,2}^2 & \ell_{2,1}\ell_{3,1} + \ell_{2,2}\ell_{3,2} \\ \ell_{3,1}\ell_{1,1} & \ell_{3,1}\ell_{2,1} + \ell_{3,2}\ell_{2,2} & \ell_{3,1}^2 + \ell_{3,2}^2 + \ell_{3,3}^2 \end{bmatrix}$$

Observe the symmetry, we can compute the $\ell'_{i,j}s$ columnwise:

$$\begin{array}{rclrcl} \ell_{1,1} & = & \sqrt{a_{1,1}} & \ell_{2,2} & = & \sqrt{a_{2,2} - \ell_{2,1}^2} \\ \ell_{2,1} & = & a_{2,1}/\ell_{1,1} & \ell_{3,2} & = & (a_{3,2} - \ell_{3,1}\ell_{2,1})/\ell_{2,2} \\ \ell_{3,1} & = & a_{3,1}/\ell_{1,1} & \ell_{3,3} & = & \sqrt{a_{3,3} - \ell_{3,1}^2 - \ell_{3,2}^2} \end{array}$$

a numerical example

$$A = \begin{bmatrix} 4 & 2 & 6 \\ 2 & 2 & 5 \\ 6 & 5 & 22 \end{bmatrix}$$

Executing the formulas:

$$\begin{array}{lcllcl} \ell_{1,1} & = & \sqrt{4} = 2 & \ell_{2,2} & = & \sqrt{2 - 1^2} = 1 \\ \ell_{2,1} & = & 2/2 = 1 & \ell_{3,2} & = & 5 - 3 \cdot 1 = 2 \\ \ell_{3,1} & = & 6/2 = 3 & \ell_{3,3} & = & \sqrt{22 - 9 - 4} = \sqrt{9} = 3 \end{array}$$

The result:

$$L = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 3 & 2 & 3 \end{bmatrix}$$

the LL^T formulas

For the elements on the diagonal:

$$\ell_{i,i} = \left(a_{i,i} - \sum_{j=1}^{i-1} \ell_{i,j}^2 \right)^{1/2}, \quad \text{for } i = 1, 2, \ldots, n.$$

For the elements below the diagonal, for $i > j$:

$$\ell_{i,j} = \left(a_{i,j} - \sum_{k=1}^{j-1} \ell_{i,k} \ell_{j,k} \right) / \ell_{j,j}, \quad \text{for } i = 1, 2, \ldots, n, j = 1, 2, \ldots, i-1.$$

the LL^T factorization algorithm

In the algorithm below,
the matrix $A = [a_{i,j}]$ contains the result in its lower half.

for $j = 1, 2, \ldots, n$ do
  for $k = 1, 2, \ldots, j-1$ do
    $a_{j,j} := a_{j,j} - a_{j,k}^2$;
  $a_{j,j} := \sqrt{a_{j,j}}$;
  for $i = j+1, \ldots, n$ do
    for $k = 1, 2, \ldots, j$ do
      $a_{i,j} := a_{i,j} - a_{i,k} a_{j,k}$
    $a_{i,j} := a_{i,j} / a_{j,j}$

our Julia function for the Cholesky factorization

```
"""
    ourCholesky(mat::Array{Float64,2})

Performs an inplace Cholesky factorization on mat.
"""
function ourCholesky(mat::Array{Float64,2})
    nbrows, nbcols = size(mat)
    for col=1:nbcols
        for idx in 1:col-1
            mat[col, col] = mat[col, col] - mat[col, idx]^2;
        end
        mat[col, col] = sqrt(mat[col, col])
        for row in col+1: nbrows
            for idx in 1:col-1
                mat[row, col] = mat[row, col] - mat[row, idx]*mat[col, idx]
            end
            mat[row, col] = mat[row, col]/mat[col, col]
        end
    end
end
```

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

LDL^T factorization

For a symmetric, positive definite matrix

$$A = \begin{bmatrix} a_{1,1} & a_{2,1} & a_{3,1} \\ a_{2,1} & a_{2,2} & a_{3,2} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}$$

consider the decomposition

$$\begin{aligned} A &= LDL^T \\ &= \begin{bmatrix} 1 & 0 & 0 \\ \ell_{2,1} & 1 & 0 \\ \ell_{3,1} & \ell_{3,2} & 1 \end{bmatrix} \begin{bmatrix} d_{1,1} & 0 & 0 \\ 0 & d_{2,2} & 0 \\ 0 & 0 & d_{3,3} \end{bmatrix} \begin{bmatrix} 1 & \ell_{2,1} & \ell_{3,1} \\ 0 & 1 & \ell_{3,2} \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned}$$

Exercise 1: Derive the formulas for the LDL^T factorization for $n = 3$.

two followup exercises

Exercise 2:
Given the formulas from the previous exercise,
generalize the formulas into an algorithm.

Exercise 3:
Write the algorithm for the LDL^T factorization as a Julia program.
Demonstrate the correctness of your program on some runs on
random 4-by-4 matrices.

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

Cholesky factorization in Julia

```
julia> using LinearAlgebra

julia> U = UpperTriangular(rand(3,3))
3×3 UpperTriangular{Float64,Array{Float64,2}}:
 0.885763  0.015683  0.702713
           0.191904  0.315976
                     0.410825

julia> A = transpose(U)*U;

julia> cholesky(A)
Cholesky{Float64,Array{Float64,2}}
U factor:
3×3 UpperTriangular{Float64,Array{Float64,2}}:
 0.885763  0.015683  0.702713
           0.191904  0.315976
                     0.410825
```

comparison with the Julia cholesky

the session continues ...

```
 0.885763   0.015683   0.702713
            0.191904   0.315976
                       0.410825

julia> include("ourcholesky.jl");

julia> ourCholesky(A)

julia> LowerTriangular(A)
3×3 LowerTriangular{Float64,Array{Float64,2}}:
 0.885763
 0.015683   0.191904
 0.702713   0.315976   0.410825
```

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

the cost of the diagonal elements

Lemma (diagonal of Cholesky factorization)

Computing all diagonal elements in the Cholesky factorization of an n-dimensional matrix requires $n^2 + n$ floating-point operations.

Proof. Consider the formula for the i-th diagonal element:

$$\ell_{i,i} = \left(a_{i,i} - \sum_{j=1}^{i-1} \ell_{i,j}^2 \right)^{1/2}, \quad \text{for } i = 1, 2, \ldots, n.$$

Computing $\ell_{i,i}$ requires

1. i subtractions,
2. $i - 1$ multiplications, and
3. one square root,

which adds up to $2i$ operations. $\sum_{i=1}^{n} 2i = 2n(n+1)/2 = n^2 + n$.

Q.E.D.

the cost of the lower diagonal

Lemma (lower diagonal of Cholesky factorization)

Computing all lower diagonal elements in the Cholesky factorization of an n-dimensional matrix requires $n^3/3 - n/3$ floating-point operations.

Proof. Consider the formula for the (i,j)-th element:

$$\ell_{i,j} = \left(a_{i,j} - \sum_{k=1}^{j-1} \ell_{i,k}\ell_{j,k} \right) / \ell_{j,j}, \quad \text{for } i = 1,2,\ldots,n, j = 1,2,\ldots,i-1.$$

Computing $\ell_{i,j}$ requires j subtractions, $j-1$ multiplications, and one division, which adds up to $2j$ operations.

$$\sum_{i=1}^{n}\sum_{j=1}^{i-1} 2j = \sum_{i=1}^{n} 2(i-1)i/2 = \sum_{i=1}^{n} i^2 - \sum_{i=1}^{n} i$$

$$= \frac{n(n+1)(2n+1)}{6} - \frac{n(n+1)}{2} = n^3/3 - n/3.$$

Q.E.D.

the cost of the Cholesky factorization

Theorem (cost of Cholesky factorization)

Computing the Cholesky factorization of an n-dimensional matrix requires $n^3/3 + n^2 + 2n/3$ *floating-point operations.*

The proof follows from the two lemmas.

Computing the LU factorization of an n-dimensional matrix requires $2n^3/3 + n^2/2 - 7n/6$ floating point operations.

The leading terms, respectively $n^3/3$ and $2n^3/3$ dominate the cost of Cholesky and LU factorization.

the cost of the LDL^T factorization

Exercise 4:

Use the formulas for the LDL^T factorization of Exercise 1 to derive the cost of the the LDL^T factorization.

the Cholesky factorization

1. Symmetric Positive Semidefinite Matrices
   - solving structured linear systems
   - positive semidefinite matrices in Julia
2. Cholesky Factorization
   - LL^T factorization
   - LDL^T factorization
   - Cholesky factorization in Julia
3. The Cost of Cholesky Factorization
   - counting the number of floating-point operations
   - timing Julia functions

timing Julia functions

On an Intel X5690 at 3.47GHz CentOS 6.10 computer, 23.4GB memory:

```
julia> using LinearAlgebra

julia> A = rand(1000,1000);

julia> @time L, U, P = lu(A);
  0.333414 seconds (451.11 k allocations: 48.834 MiB,
1.56% gc time, 92.03% compilation time)

julia> B = L*transpose(L);

julia> @time cholesky(B);
  0.118595 seconds (94.44 k allocations: 13.371 MiB,
18.87% gc time, 87.37% compilation time)
```

We see that Cholesky factorization costs less than half of the time of an LU factorization.

timing Julia functions on your computer

Exercise 5:

Repeat the experiment of the previous slide on your computer.

1. Give the main specifications on your computer.
2. Report the timings for both LU and Cholesky.