I'm not robot

reCAPTCHA

Continue

Square root of 147 simplified

A square root of a number is a number that, when it is multiplied by itself (squared), gives the first number again. For example, 2 is the square root of 4, because 2x2=4. Only numbers bigger than or equal to zero have real square roots. A number bigger than zero has two square roots: one is positive (bigger than zero) and the other is negative (smaller than zero). For example, 4 has two square roots: 2 and -2. The only square root of zero is zero. A whole number with a square root that is also a whole number is called a perfect square. The square root radical is simplified or in its simplest form only when the radicand has no square factors left. A radical is also in simplest form when the radicand is not a fraction. In linear algebra, the Cholesky decomposition or Cholesky factorization (pronounced /ʃəˈlɛski/ shə-LES-kee) is a decomposition of a Hermitian, positive-definite matrix into the product of a lower triangular matrix and its conjugate transpose, which is useful for efficient numerical solutions, e.g., Monte Carlo simulations. It was discovered by André-Louis Cholesky for real matrices. When it is applicable, the Cholesky decomposition is roughly twice as efficient as the LU decomposition for solving systems of linear equations.[1] Statement The Cholesky decomposition of a Hermitian positive-definite matrix A, is a decomposition of the form A = L L ∗ , {\displaystyle \mathbf {A} =\mathbf {LL} ^{*},} where L is a lower triangular matrix with real and positive diagonal entries, and L* denotes the conjugate transpose of L. Every Hermitian positive-definite matrix (and thus also every real-valued symmetric positive-definite matrix) has a unique Cholesky decomposition.[2] The converse holds trivially: if A can be written as LL* for some invertible L, lower triangular or otherwise, then A is Hermitian and positive definite. When A is a real matrix (hence symmetric positive-definite), the factorization may be written A = L L T , {\displaystyle \mathbf {A} =\mathbf {LL} ^{\mathsf {T}},} where L is a real lower triangular matrix with positive diagonal entries.[3][4][5] Positive semidefinite matrices If a Hermitian matrix A is only positive semidefinite, instead of positive definite, then it still has a decomposition of the form A = LL* where the diagonal entries of L are allowed to be zero.[6] The decomposition need not be unique, for example: [0 0 0 1] = L L ∗ , L = [0 0 cos ⁡ θ sin ⁡ θ] . {\displaystyle {\begin{bmatrix}0&0\\0&1\end{bmatrix}}=\mathbf {L} \mathbf {L} ^{*},\quad \quad \mathbf {L} ={\begin{bmatrix}0&0\\\cos \theta &\sin \theta \end{bmatrix}}.} However, if the rank of A is r, then there is a unique lower triangular L with exactly r positive diagonal elements and n−r columns containing all zeroes.[7] Alternatively, the decomposition can be made unique when a pivoting choice is fixed. Formally, if A is an n × n positive semidefinite matrix of rank r, then there is at least one permutation matrix P such that P A PT has a unique decomposition of the form P A PT = L L* with L = [L 1 0 L 2 0] {\displaystyle \mathbf {L} ={\begin{bmatrix}\mathbf {L} _{1}&0\\\mathbf {L} _{2}&0\end{bmatrix}}} , where L1 is an r × r lower triangular matrix with positive diagonal.[8] LDL decomposition A closely related variant of the classical Cholesky decomposition is the LDL decomposition, A = L D L ∗ , {\displaystyle \mathbf {A} =\mathbf {LDL} ^{*},} where L is a lower unit triangular (unitriangular) matrix, and D is a diagonal matrix. That is, the diagonal elements of L are required to be 1 at the cost of introducing an additional diagonal matrix D in the decomposition. The main advantage is that the LDL decomposition can be computed and used with essentially the same algorithms, but avoids extracting square roots.[9] For this reason, the LDL decomposition is often called the square-root-free Cholesky decomposition. For real matrices, the factorization has the form A = LDLT and is often referred to as LDLT decomposition (or LDLT decomposition, or LDL′). It is closely related to the eigendecomposition of real symmetric matrices, A = QΛQT. The LDL decomposition is related to the classical Cholesky decomposition of the form LL* as follows: A = L D L ∗ = L D 1 / 2 (D 1 / 2) ∗ L ∗ = L D 1 / 2 (L D 1 / 2) ∗ . {\displaystyle \mathbf {A} =\mathbf {LDL} ^{*}=\mathbf {L} \mathbf {D} ^{1/2}\left(\mathbf {D} ^{1/2}\right)^{*}\mathbf {L} ^{*}=\mathbf {L} \mathbf {D} ^{1/2}\left(\mathbf {L} \mathbf {D} ^{1/2}\right)^{*}.} Conversely, given the classical Cholesky decomposition A = C C ∗ {\displaystyle \mathbf {A} =\mathbf {C} \mathbf {C} ^{*}} of a positive definite matrix, if S is a diagonal matrix that contains the main diagonal of C {\displaystyle \mathbf {C} } , then a A can be decomposed as L D L ∗ {\displaystyle \mathbf {L} \mathbf {D} \mathbf {L} ^{*}} where L = C S − 1 {\displaystyle \mathbf {L} =\mathbf {C} \mathbf {S} ^{-1}} (this rescales each column to makes diagonal elements 1), D = S 2 . {\displaystyle \mathbf {D} =\mathbf {S} ^{2}.} If A is positive definite then the diagonal elements of D are all positive. For positive semidefinite A, an L D L ∗ {\displaystyle \mathbf {L} \mathbf {D} \mathbf {L} ^{*}} decomposition exists where the number of non-zero elements on the diagonal D is exactly the rank of A.[10] Some indefinite matrices for which no Cholesky decomposition exists have an LDL decomposition with negative entries in D: it suffices that the first n−1 leading principal minors of A are non-singular.[11] Example Here is the Cholesky decomposition of a symmetric real matrix: (4 12 − 16 12 37 − 43 − 16 − 43 98) = (2 0 0 6 1 0 − 8 5 3) (2 6 − 8 0 1 5 0 0 3) . {\displaystyle {\begin{aligned}\left({\begin{array}{*{3}{r}}4&12&-16\\12&37&-43\\-16&-43&98\\\end{array}}\right)=\left({\begin{array}{*{3}{r}}2&0&0\\6&1&0\\-8&5&3\\\end{array}}\right)\left({\begin{array}{*{3}{r}}2&6&-8\\0&1&5\\0&0&3\\\end{array}}\right).\end{aligned}}} And here is its LDLT decomposition: (4 12 − 16 12 37 − 43 − 16 − 43 98) = (1 0 0 3 1 0 − 4 5 1) (4 0 0 0 1 0 0 0 9) (1 3 − 4 0 1 5 0 0 1) . {\displaystyle {\begin{aligned}\left({\begin{array}{*{3}{r}}4&12&-16\\12&37&-43\\-16&-43&98\\\end{array}}\right)&=\left({\begin{array}{*{3}{r}}1&0&0\\3&1&0\\-4&5&1\\\end{array}}\right)\left({\begin{array}{*{3}{r}}4&0&0\\0&1&0\\0&0&9\\\end{array}}\right)\left({\begin{array}{*{3}{r}}1&3&-4\\0&1&5\\0&0&1\\\end{array}}\right).\end{aligned}}} Applications The Cholesky decomposition is mainly used for the numerical solution of linear equations A x = b {\displaystyle \mathbf {Ax} =\mathbf {b} } . If A is symmetric and positive definite, then we can solve A x = b {\displaystyle \mathbf {Ax} =\mathbf {b} } by first computing the Cholesky decomposition A = L L ∗ {\displaystyle \mathbf {A} =\mathbf {LL} ^{\mathrm {*} }} , then solving L y = b {\displaystyle \mathbf {Ly} =\mathbf {b} } for y by forward substitution, and finally solving L ∗ x = y {\displaystyle \mathbf {L^{*}x} =\mathbf {y} } for x by back substitution. An alternative way to eliminate taking square roots in the L L ∗ {\displaystyle \mathbf {LL} ^{\mathrm {*} }} decomposition is to compute the Cholesky decomposition A = L D L ∗ {\displaystyle \mathbf {A} =\mathbf {LDL} ^{\mathrm {*} }} , then solving L y = b {\displaystyle \mathbf {Ly} =\mathbf {b} } for y, and finally solving D L ∗ x = y {\displaystyle \mathbf {DL} ^{\mathrm {*} }\mathbf {x} =\mathbf {y} } . For linear systems that can be put into symmetric form, the Cholesky decomposition (or its LDL variant) is the method of choice, for superior efficiency and numerical stability. Compared to the LU decomposition, it is roughly twice as efficient.[1] Linear least squares Systems of the form Ax = b with A symmetric and positive definite arise quite often in applications. For instance, the normal equations in linear least squares problems are of this form. It may also happen that matrix A comes from an energy functional, which must be positive from physical considerations; this happens frequently in the numerical solution of partial differential equations. Non-linear optimization Non-linear multi-variate functions may be minimized over their parameters using variants of Newton's method called quasi-Newton methods. At iteration k, the search steps in a direction p k {\displaystyle p_{k}} defined by solving B k p k = − g k {\displaystyle B_{k}p_{k}=-g_{k}} for p k {\displaystyle p_{k}} , where p k {\displaystyle p_{k}} is the step direction, g k {\displaystyle g_{k}} is the gradient, and B k {\displaystyle B_{k}} is an approximation to the Hessian matrix formed by repeating rank-1 updates at each iteration. Two well-known update formulas are called Davidon–Fletcher–Powell (DFP) and Broyden–Fletcher–Goldfarb–Shanno (BFGS). Loss of the positive-definite condition through round-off error is avoided if rather than updating an approximation to the inverse of the Hessian, one updates the Cholesky decomposition of an approximation of the Hessian matrix itself .[12] Monte Carlo simulation The Cholesky decomposition is commonly used in the Monte Carlo method for simulating systems with multiple correlated variables. The covariance matrix is decomposed to give the lower-triangular L. Applying this to a vector of uncorrelated samples u produces a sample vector Lu with the covariance properties of the system being modeled.[13] The following simplified example shows the economy one gets from the Cholesky decomposition: suppose the goal is to generate two correlated normal variables x 1 {\displaystyle x_{1}} and x 2 {\displaystyle x_{2}} with given correlation coefficient ρ {\displaystyle \rho } . To accomplish that, it is necessary to first generate two uncorrelated Gaussian random variables z 1 {\displaystyle z_{1}} and z 2 {\displaystyle z_{2}} , which can be done using a Box–Muller transform. Given the required correlation coefficient ρ {\displaystyle \rho } , the correlated normal variables can be obtained via the transformations x 1 = z 1 {\displaystyle x_{1}=z_{1}} and x 2 = ρ z 1 + 1 − ρ 2 z 2 {\textstyle x_{2}=\rho z_{1}+{\sqrt {1-\rho ^{2}}}z_{2}} . Kalman filters Unscented Kalman filters commonly use the Cholesky decomposition to choose a set of so-called sigma points. The Kalman filter tracks the average state of a system as a vector x of length N and covariance as an N × N matrix P. The matrix P is always positive semi-definite and can be decomposed into LLT. The columns of L can be added and subtracted from the mean x to form a set of 2N vectors called sigma points. These sigma points completely capture the mean and covariance of the system state. Matrix inversion The explicit inverse of a Hermitian matrix can be computed by Cholesky decomposition, in a manner similar to solving linear systems, using n 3 {\displaystyle n^{3}} operations (1 2 n 3 {\displaystyle {\tfrac {1}{2}}n^{3}} multiplications).[9] The entire inversion can even be efficiently performed in-place. A non-Hermitian matrix B can also be inverted using the following identity, where BB* will always be Hermitian: B − 1 = B ∗ (B B ∗) − 1 . {\displaystyle \mathbf {B} ^{-1}=\mathbf {B} ^{*}(\mathbf {BB} ^{*})^{-1}.} Computation There are various methods for calculating the Cholesky decomposition. The computational complexity of commonly used algorithms is O(n3) in general.[citation needed] The algorithms described below all involve about (1/3)n3 FLOPs (n3/6 multiplications and the same number of additions) for real flavors and (4/3)n3 FLOPs for complex flavors,[14] where n is the size of the matrix A. Hence, they have half the cost of the LU decomposition, which uses 2n3/3 FLOPs (see Trefethen and Bau 1997). Which of the algorithms below is faster depends on the details of the implementation. Generally, the first algorithm will be slightly slower because it accesses the data in a less regular manner. The Cholesky algorithm The Cholesky algorithm, used to calculate the decomposition matrix L, is a modified version of Gaussian elimination. The recursive algorithm starts with i := 1 and A(1) := A. At step i, the matrix A(i) has the following form: A (i) = (I i − 1 0 0 0 a i , i b i ∗ 0 b i B (i)) , {\displaystyle \mathbf {A} ^{(i)}={\begin{pmatrix}\mathbf {I} _{i-1}&0&0\\0&a_{i,i}&\mathbf {b} _{i}^{*}\\0&\mathbf {b} _{i}&\mathbf {B} ^{(i)}\end{pmatrix}},} where Ii−1 denotes the identity matrix of dimension i − 1. If we now define the matrix Li by L i := (I i − 1 0 0 0 a i , i 0 0 1 a i , i b i I n − i) , {\displaystyle \mathbf {L} _{i}:={\begin{pmatrix}\mathbf {I} _{i-1}&0&0\\0&{\sqrt {a_{i,i}}}&0\\0&{\frac {1}{\sqrt {a_{i,i}}}}\mathbf {b} _{i}&\mathbf {I} _{n-i}\end{pmatrix}},} then we can write A(i) as A (i) = L i A (i + 1) L i ∗ {\displaystyle \mathbf {A} ^{(i)}=\mathbf {L} _{i}\mathbf {A} ^{(i+1)}\mathbf {L} _{i}^{*}} where A (i + 1) = (I i − 1 0 0 0 1 0 0 0 B (i) − 1 a i , i b i b i ∗) . {\displaystyle \mathbf {A} ^{(i+1)}={\begin{pmatrix}\mathbf {I} _{i-1}&0&0\\0&1&0\\0&0&\mathbf {B} ^{(i)}-{\frac {1}{a_{i,i}}}\mathbf {b} _{i}\mathbf {b} _{i}^{*}\end{pmatrix}}.} Note that bi b*i is an outer product, therefore this algorithm is called the outer-product version in (Golub & Van Loan). We repeat this for i from 1 to n. After n steps, we get A(n+1) = I. Hence, the lower triangular matrix L we are looking for is calculated as L := L 1 L 2 … L n . {\displaystyle \mathbf {L} :=\mathbf {L} _{1}\mathbf {L} _{2}\dots \mathbf {L} _{n}.} The Cholesky–Banachiewicz and Cholesky–Crout algorithms Access pattern (white) and writing pattern (yellow) for the in-place Cholesky—Banachiewicz algorithm on a 5×5 matrix If we write out the equation A = L L T = (L 11 0 0 L 21 L 22 0 L 31 L 32 L 33) (L 11 L 21 L 31 0 L 22 L 32 0 0 L 33) = (L 11 2 (symmetric) L 21 L 11 L 21 2 + L 22 2 L 31 L 11 L 31 L 21 + L 32 L 22 L 31 2 + L 32 2 + L 33 2) , {\displaystyle {\begin{aligned}\mathbf {A} =\mathbf {LL} ^{T}&={\begin{pmatrix}L_{11}&0&0\\L_{21}&L_{22}&0\\L_{31}&L_{32}&L_{33}\\\end{pmatrix}}{\begin{pmatrix}L_{11}&L_{21}&L_{31}\\0&L_{22}&L_{32}\\0&0&L_{33}\end{pmatrix}}\\[8pt]&={\begin{pmatrix}L_{11}^{2}&&({\text{symmetric}})\\L_{21}L_{11}&L_{21}^{2}+L_{22}^{2}&\\L_{31}L_{11}&L_{31}L_{21}+L_{32}L_{22}&L_{31}^{2}+L_{32}^{2}+L_{33}^{2}\end{pmatrix}},\end{aligned}}} we obtain the following: L = (A 11 0 0 A 21 / L 11 A 22 − L 21 2 0 A 31 / L 11 (A 32 − L 31 L 21) / L 22 A 33 − L 31 2 − L 32 2) {\displaystyle {\begin{aligned}\mathbf {L} ={\begin{pmatrix}{\sqrt {A_{11}}}&0&0\\A_{21}/L_{11}&{\sqrt {A_{22}-L_{21}^{2}}}&0\\A_{31}/L_{11}&\left(A_{32}-L_{31}L_{21}\right)/L_{22}&{\sqrt {A_{33}-L_{31}^{2}-L_{32}^{2}}}\end{pmatrix}}\end{aligned}}} and therefore the following formulas for the entries of L: L j , j = (±) A j , j − ∑ k = 1 j − 1 L j , k 2 , {\displaystyle L_{j,j}=(\pm){\sqrt {A_{j,j}-\sum _{k=1}^{j-1}L_{j,k}^{2}}},} L i , j = 1 L j , j (A i , j − ∑ k = 1 j − 1 L i , k L j , k) for i > j . {\displaystyle L_{i,j}={\frac {1}{L_{j,j}}}\left(A_{i,j}-\sum _{k=1}^{j-1}L_{i,k}L_{j,k}\right)\quad {\text{for }}i>j.} For complex and real matrices, inconsequential arbitrary sign changes of diagonal and associated off-diagonal elements are allowed. The expression under the square root is always positive if A is real and positive-definite. For complex Hermitian matrix, the following formula applies: L j , j = A j , j − ∑ k = 1 j − 1 L j , k L j , k ∗ , {\displaystyle L_{j,j}={\sqrt {A_{j,j}-\sum _{k=1}^{j-1}L_{j,k}L_{j,k}^{*}}},} L i , j = 1 L j , j (A i , j − ∑ k = 1 j − 1 L i , k L j , k ∗) for i > j . {\displaystyle L_{i,j}={\frac {1}{L_{j,j}}}\left(A_{i,j}-\sum _{k=1}^{j-1}L_{i,k}L_{j,k}^{*}\right)\quad {\text{for }}i>j.} So we can compute the (i, j) entry if we know the entries to the left and above. The computation is usually arranged in either of the following orders: The Cholesky–Banachiewicz algorithm starts from the upper left corner of the matrix L and proceeds to calculate the matrix row by row. for (i = 0; i < dimensionSize; i++) { for (j = 0; j