Weierstraß-Institut
für Angewandte Analysis und Stochastik

im Forschungsverbund Berlin e.V.

Preprint ISSN 0946 – 8633

A generalized non–square Cholesky Decomposition Algorithm with Applications to Finance

Oliver Reiß[1]

submitted: August 7th 2002

[1] Weierstrass-Institute for
Applied Analysis and Stochastics
Mohrenstraße 39
D - 10117 Berlin
Germany
E-Mail: reiss@wias-berlin.de
URL: `http://www.wias-berlin.de/~reiss`

No. 760
Berlin 2002

2000 *Mathematics Subject Classification.* 15-04, 65F30, 91-08.

Key words and phrases. Modified Cholesky decomposition, LDLT decomposition, Semi–positive matrices, Covariance matrix, Correlation matrix, Value at Risk.

Oliver Reiß is affiliated to the bmb+f project "Effiziente Methoden zur Bestimmung von Risikomaßen".

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Mohrenstraße 39
D — 10117 Berlin
Germany

Fax:	+ 49 30 2044975
E-Mail:	preprint@wias-berlin.de
World Wide Web:	http://www.wias-berlin.de/

Abstract

In several applications there is the need to compute a Cholesky decomposition of a given symmetric matrix. The usual Cholesky decomposition algorithm will fail if the given matrix is semi–positive, although such a decomposition exists. To overcome this problem there exists a LDL^T decomposition for semi–positive matrices.

In the case that the given symmetric matrix is not semi–positive, no Cholesky decomposition exists. In such a situation one aims to approximate this matrix by a (semi–)positive one and computes the Cholesky decomposition of the approximation.

From the context of numerical optimization there exist algorithms by Gill, Murray and Wright and a refinement by Eskow and Schnabel. Both methods basicly return a Cholesky decomposition of a positive approximation of an indefinite input matrix.

In this paper we extend the LDL^T algorithm such that it coincides for a semi–definite input with the LDL^T decomposition and for indefinite input it gives the decomposition of a semi–positive approximation. In contrast to the algorithms mentioned before, for indefinite input matrices our algorithm gives a decomposition, which has a lower rank. This gives the important opportunity to introduce a dimension reduction, if possible, and we will show that this algorithm can save computation time in several applications in finance, especially for risk management.

1 Introduction

In this paper we concentrate on the decomposition of real, symmetric matrices. It is a well known fact, that these matrices are diagonalizable with real eigenvalues. A real symmetric matrix is called *positive*, if all eigenvalues are (strictly) greater than 0. A matrix is called *semi–positive*, if and only if all eigenvalues are greater or equal than zero. A matrix is called *indefinite*, if it has as well positive as negative eigenvalues.

The usual Cholesky decomposition is well known and widely used in practice. It is a fast algorithm and is numerical stable, if the given matrix is positive. But if the matrix is not positive, then the usual Cholesky decomposition fails. We will introduce a non–square Cholesky decomposition, which will overcome this problem. Furthermore, if the given matrix is indefinite, our algorithm will give a Cholesky decomposition of a semi–positive matrix, which is an approximation of the given matrix. This approximation will be done such, that the diagonal of the matrix keeps unchanged, if the diagonal elements were non–negative.

The main object of this paper is the presentation of a non–square Cholesky algorithm, which is based on the LDL^T decomposition. For the convenience of the reader we recall in section 2 the LDL^T decomposition for positive matrices. We also give a proof of the existence and uniqueness of this decomposition and make an analysis of the computation time of this algorithm. In fact this algorithm yields to the usual Cholesky decomposition.

In section 3 we cite the LDL^T algorithm for semi–positive matrices. In this situation, pivoting becomes necessary and we use the pivoting strategy suggested in [4]. With this modification, this algorithm (algorithm **1**) is numerically very stable. We also give a criteria which enables us to determine the rank of a matrix semi–positive matrix.

In section 4 we give an example which shows, that in general no LDL^T decomposition for indefinite matrices exists. This indicates in which situation the algorithm **1** fails and brings us to the idea of the non–square Cholesky decomposition. We introduce two new algorithms which approximate an indefinite matrix by a semi–positive one and which give the LDL^T decomposition of this approximation. The algorithm **2** has an additional instruction, which describes the first approximation technique. But the approximation is such, that as well diagonal as off–diagonal elements of an indefinite input matrix will be changed. The next algorithm **3** performs a rescaling which guarantees, that any indefinite matrix with non–negative diagonal elements will be approximated by changing off–diagonal elements only. This algorithm is the main contribution of this paper and brings us to the non–square Cholesky decomposition.

Since we are approximating a matrix by a positive one, we recall in section 5 other algorithms, which also give such approximations and we will compare the algorithm **3** with these standard techniques. We repeat two results for minimal approximations in the Frobenius norm and the spectral norm. For completeness we also recall some approximation techniques, which keep the diagonal elements of the indefinite matrix unchanged, if they are non–negative. This is often a requirement in practical applications, for example a correlation matrix must be unit–diagonal.

In section 6 we compare our algorithms with the two other LDL^T based approximation approaches by Gill,Murray and Wright [3] and the improvement of Schnabel and Eskow [11]. The algorithm **3** is at least as good as the other approaches and has additional two very nice features. It preserves the diagonal of a given matrix, which is very important for several applications and in contrast to the other algorithms our algorithm achieves a dimension reduction.

Furthermore we apply the algorithm **3** on indefinite matrices which can be seen as perturbed correlation matrices and aim to approximate these matrices by semi–positive, unit–diagonal matrices. We compare the approximation error of our algorithm with other standard techniques presented in section 5.2 and we point out, that our algorithm is a right choice, if computation time is critical.

The new algorithm can be used in several applications in finance and we present three examples in section 7. We underline that the dimension reduction feature of the non–square Cholesky decomposition can help to save computing time in the context of Monte Carlo simulations. In the second example we explain, that from a risk managers point of view the diagonal preserving property of the non–square Cholesky decomposition is essential for stress testing. In this context the approximation feature of our algorithm is very important, since due to missing data or to specific modifications of some correlations the given covariance matrix could be indefinite. Finally, for Value at Risk valuations the dimension reduction property of this algorithm can also save a lot of computing time in the context of the so called delta–gamma approximation.

2 The LDL$^{\mathrm{T}}$ -Decomposition for Positive Matrices

We first recall some facts about positive symmetric matrices in the following

Lemma 1 *Let S be a positive symmetric $N \times N$ matrix. Then the following statements hold:*

1. *For all $x \in \mathbb{R}^N$, $x \neq 0$ holds: $x^\top Sx>0$*
2. $S_{ii} > 0$
3. $|S_{ij}| < \sqrt{S_{ii}S_{jj}} \qquad i \neq j$
4. *If*

$$S =: \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \tag{1}$$

then the $(N-1) \times (N-1)$ matrices $\tilde{S}$ and $\tilde{S} - \frac{1}{S_{11}} ss^\top$ are positive.

Proof.

1. Let λ_i are the eigenvalues of S with corresponding eigenvectors s_i. We write $x = \sum\limits_i \beta_i s_i$ with at least one $\beta_i > 0$. Since S is positive, λ_i are also positive:

$$x^\top Sx \;=\; \sum_{i,j} \beta_j \beta_i s_j^\top \lambda_i s_i = \sum_i \beta_i^2 \lambda_i > 0 \tag{2}$$

2. We set $x_k = \delta_{ki}$ and obtain:

$$0 < x^\top Sx = \sum_{k,l} \delta_{ik} S_{kl} \delta_{li} = S_{ii} \tag{3}$$

3. We fix k, l with $k \neq l$ and choose $x_i = \delta_{ik} + \alpha\delta_{il}$. Note that $x \neq 0$ and for all $\alpha \in \mathbb{R}$:

$$0 \;<\; x^\top S x = \sum_{i,j} (\delta_{ki} + \alpha\delta_{li}) S_{ij} (\delta_{jk} + \alpha\delta_{jl}) = S_{kk} + 2\alpha S_{kl} + \alpha^2 S_{ll}$$

   Hence $S_{kl}^2 < S_{kk} S_{ll}$.

4. First set $x = (0, y)^\top$. Then we have

$$0 < x^\top S x = y^\top \tilde{S} y \tag{4}$$

   For the second matrix we have to show, that for all $y \in \mathbb{R}^{N-1}$ holds:

$$y^\top (\tilde{S} - \frac{1}{S_{11}} s s^\top) y = y^\top \tilde{S} y - \frac{1}{S_{11}} (y \cdot s)^2 > 0 \tag{5}$$

   We set $x = (\alpha, y)^\top$ and since S is positive we obtain for all $\alpha \in \mathbb{R}$:

$$0 \;<\; x^\top S x = \alpha^2 S_{11} + 2\alpha (y \cdot s) + y^\top \tilde{S} y \tag{6}$$

   So we have proven (5), since the last inequality yields:

$$(y \cdot s)^2 - S_{11}\, y^\top \tilde{S} y < 0 \tag{7}$$

■

By this lemma, we are able to prove the LDL$^\text{T}$ decomposition theorem for positive symmetric matrices.

Theorem 1 *Let S be a symmetric, positive $N \times N$ matrix. Then there exists a unique decomposition*

$$S = L \cdot D \cdot L^\top \tag{8}$$

where D is a diagonal $N \times N$ matrix and L is a $N \times N$ unit left-triangular matrix, that is $L_{ii} = 1$ for all i and $L_{ij} = 0$ for $j > i$.

Proof. We prove this theorem by complete induction. For $N = 1$ the decomposition reads

$$S_{11} = L_{11} D_{11} L_{11}^\top = D_{11} \tag{9}$$

So the LDL$^\text{T}$ decomposition is given and obviously unique.

Assume that the theorem holds for $N - 1$ with $N > 1$. Set

$$S \;=:\; \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \tag{10}$$

From lemma 1 we know, that the $(N-1) \times (N-1)$ matrix $\tilde{S} - \frac{1}{S_{11}} s s^\top$ is positive and has the unique LDL$^\text{T}$ decomposition $\tilde{L}\tilde{D}\tilde{L}^\top$ by the assumption of the induction. Note that $S_{11} > 0$ and define $l = \frac{1}{S_{11}} s$. Then

$$S = \left(\begin{array}{c|c} 1 & 0 \\ \hline l & \tilde{L} \end{array}\right) \left(\begin{array}{c|c} S_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \left(\begin{array}{c|c} 1 & l^\top \\ \hline 0 & \tilde{L}^\top \end{array}\right) = LDL^\top \tag{11}$$

This solution is obviously unique, because any LDL$^\text{T}$ decomposition yields to the equations $D_{11} = S_{11}$ and $L_{j1} = \frac{1}{S_{11}} S_{1j}$ with $j = 2 \ldots N$. ■

If one knows the LDL$^\text{T}$ decomposition of a positive symmetric matrix, a lot of usual problems arising in the linear algebra are solved. Let $S = LDL^\top$ with the conditions in this section, one has immediately:

- $\det S = \prod_{i=1}^{N} D_{ii}$
- With $U := DL^\top$ this decomposition is a special LU decomposition
- With $C := LD^{\frac{1}{2}}$ one obtains the usual (triangular) Cholesky decomposition $S = CC^\top$.
- $S^{-1} = (L^{-1})^\top D^{-1} L^{-1}$, so instead of inverting a positive symmetric matrix S it is enough to invert the triangular matrix L.

We will now state the algorithm for this decomposition. If we evaluate equation (8) for each component of S, we obtain the following system of equations:

$$\begin{aligned} D_{ii} &= S_{ii} - \sum_{k=1}^{i-1} D_{kk} L_{ik}^2 \\ L_{ji} &= \frac{1}{D_{ii}} \left(S_{ij} - \sum_{k=1}^{i-1} D_{kk} L_{ik} L_{jk} \right) \qquad j > i \end{aligned} \tag{12}$$

One has to compute this equations in the right order and thus obtain L and D. Remark, that in this sequence the right hand sides of the equations (12) are known by the computations before:

1. Set i=1.
2. Compute D_{ii}.
3. Compute L_{ji} for $j = i+1, \ldots, N$
4. If (i<N) set i=i+1 and go to Step 2.

We count the number of basic operations for the $\mathrm{LDL^T}$ decomposition algorithm. A basic operation (op) consists of two multiplications, one subtraction with assign and the expense for the loop management. Obviously we obtain from the algorithm:

$$\begin{aligned}
\mathrm{LDLTcosts}(N) &\approx \sum_{i=1}^{N}\left((i-1)+\sum_{j=i+1}^{N}(i-1)\right)\ \mathrm{op} \\
&= \sum_{i=1}^{N}\left((1+N-i)(i-1)\right)\ \mathrm{op} = \sum_{i=1}^{N}\big((N+2)i-i^2-(N+1)\big)\ \mathrm{op} \\
&= (N+2)\frac{N(N+1)}{2}\ \mathrm{op} - \frac{N(N+1)(2N+1)}{6}\ \mathrm{op} - N(N+1)\ \mathrm{op} \\
&= \frac{1}{6}N^3\ \mathrm{op} + \mathcal{O}(N^2)\ \mathrm{op} \qquad (13)
\end{aligned}$$

So this algorithm is very fast. Recall that the asymptotic operation costs for a usual Cholesky decomposition is also given by $\frac{1}{6}N^3$, the basic operation there is one multiplication, one subtraction and the loop management.

There is no pivoting used within this algorithm. In the situation where the matrix S is strictly positive this algorithm is extremely stable, because the remaining matrices after each step are also strictly positive. This fact is well known from the usual Cholesky decomposition algorithm, which also makes no use of pivoting. In the case of semi–positive matrices pivoting becomes necessary and we will present a pivoting strategy in the next section. Of course, that pivoting can also be applied to strictly positive matrices.

3 $\mathrm{LDL^T}$ for Semi–positive Matrices

From the analysis of the $\mathrm{LDL^T}$ algorithm for positive matrices, we can directly extend this algorithm to semi–positive matrices. In the algorithm above a problem appears, if there occurs one $D_{ii} = 0$ while the algorithm runs. In the next step there is a division by zero then. We will show, how to avoid this problem in the case of semi–positive matrices.

We recall some basics about semi–positive matrices in the following lemma which is a modification of lemma 1 in fact. For the proof of this lemma replace each "$<$" by "$\leq$" and every "$>$" by "$\geq$" in the proof of lemma 1. The last statement of the fourth part is corollary of the third part:

Lemma 2 *Let S be a semi–positive symmetric $N \times N$ matrix. Then the following statements hold:*

1. *For all $x \in \mathbb{R}^N$ holds $x^\top S x \geq 0$*
2. *$S_{ii} \geq 0$*

3. $|S_{ij}| \leq \sqrt{S_{ii}S_{jj}}$

4. *If the* $N \times N$ *matrix*

$$S = \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \tag{14}$$

is semi–positive, then the following statements are true:

- $\tilde{S}$ *is semi–positive*
- *If* $S_{11} > 0$ *then* $\tilde{S} - \frac{1}{S_{11}} s\, s^\top$ *is semi–positive*
- *If* $S_{11} = 0$ *then* $s = 0$.

Using this lemma, we can analyze the LDL$^\mathrm{T}$ decomposition for semi–positive matrices:

Theorem 2 *Let* S *be a symmetric, semi–positive* $N \times N$ *matrix.*

1. *Then there exists a decomposition*

$$S \;=\; L \cdot D \cdot L^\top \tag{15}$$

where D *is a diagonal* $N \times N$ *matrix and* L *is a unit left-triangular* $N \times N$ *matrix.*

2. *We define the head of a column* i *in a matrix* A *by*

$$h_A(i) \;:=\; \min\{j | A_{ji} \neq 0\}. \tag{16}$$

If M *is the rank of* $S \neq \mathbf{0}$, *then there exists a unique decomposition*

$$S \;=\; LDL^\top \tag{17}$$

where D *is positive diagonal* $M \times M$ *matrix and* L *is a* $N \times M$ *matrix with the properties:*

- *The head in each column is* 1, *i.e.* $L_{h_L(i)i} = 1$.
- *The head of the ith column stands below the head of the* $i-1$ *th column, i.e.* $h_L(i) > h_L(i-1)$.

3. *Furthermore, if* M *is the rank of* S *with* $S \neq 0$, *then there exists a permutation* P *such that*

$$PSP^\top \;=:\; U = LDL^\top \tag{18}$$

with a positive diagonal $M \times M$ *matrix* D *and* L *is a* $N \times M$ *matrix with* $L_{ii} = 1$ *and the decomposition of* U *is unique.*

Proof. We prove all statements simultaneously by complete induction. For $N = 1$ a LDL$^{\mathrm{T}}$ decomposition is given by

$$S = (1)(S_{11})(1), \tag{19}$$

hence $D_{11} = S_{11}$. For $S \neq \mathbf{0}$ this decomposition is obviously unique, hence 1,2,3 hold.

Let us assume, that all statements hold for dimension N. Then for the decomposition of a semi–positive $(N+1) \times (N+1)$ matrix

$$S = \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \tag{20}$$

we distinguish three cases:

1. Case: $S_{11} \neq 0$ **and** $\tilde{S} \neq \frac{ss^\top}{S_{11}}$

Statement 1: By the induction assumption there exists a LDL$^{\mathrm{T}}$ decomposition

$$\tilde{S} - \frac{ss^\top}{S_{11}} = \tilde{L}\tilde{D}\tilde{L}^\top \tag{21}$$

and so a LDL$^{\mathrm{T}}$ decomposition is given by

$$\left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) = \left(\begin{array}{c|c} 1 & 0 \\ \hline \frac{s}{S_{11}} & \tilde{L} \end{array}\right) \left(\begin{array}{c|c} S_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \left(\begin{array}{c|c} 1 & \frac{s^\top}{S_{11}} \\ \hline 0 & \tilde{L}^\top \end{array}\right) \tag{22}$$

Statement 2: Since rank$(S) \geq 2$, any decomposition has to satisfy the equation:

$$\left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) = \left(\begin{array}{c|c} L_{11} & 0 \\ \hline l & \tilde{L} \end{array}\right) \left(\begin{array}{c|c} D_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \left(\begin{array}{c|c} 1 & l^\top \\ \hline 0 & \tilde{L}^\top \end{array}\right) \tag{23}$$

$$= \left(\begin{array}{c|c} L_{11}^2 D_{11} & L_{11}D_{11}l^\top \\ \hline L_{11}D_{11}l & D_{11}ll^\top + \tilde{L}\tilde{D}\tilde{L}^\top \end{array}\right) \tag{24}$$

Since L_{11} is either 0 or 1, we find the necessary conditions $L_{11} = 1$, $D_{11} = S_{11}$ and $l = \frac{1}{S_{11}}s$. The remaining equation

$$\tilde{S} - \frac{ss^\top}{S_{11}} = \tilde{L}\tilde{D}\tilde{L}^\top \tag{25}$$

has a unique solution under the structure conditions on L by induction assumption and the matrix

$$L = \left(\begin{array}{c|c} 1 & 0 \\ \hline l & \tilde{L} \end{array}\right) \tag{26}$$

also fulfills the structure conditions.

Statement 3: By induction assumption there exists a permutation $\tilde{P}$ such that

$$\tilde{P}(\tilde{S} - \frac{ss^\top}{S_{11}})\tilde{P}^\top \;=\; \tilde{L}\tilde{D}\tilde{L}^\top \tag{27}$$

We further define $P := \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & \tilde{P} \end{array}\right)$. Then, any decomposition must satisfy

$$PSP^\top \;=\; \left(\begin{array}{c|c} S_{11} & s^\top\tilde{P}^\top \\ \hline \tilde{P}s & \tilde{P}\tilde{S}\tilde{P}^\top \end{array}\right) = \left(\begin{array}{c|c} 1 & 0 \\ \hline l & \tilde{L} \end{array}\right)\left(\begin{array}{c|c} D_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right)\left(\begin{array}{c|c} 1 & l^\top \\ \hline 0 & \tilde{L}^\top \end{array}\right) \tag{28}$$

$$\;=\; \left(\begin{array}{c|c} D_{11} & D_{11}l^\top \\ \hline D_{11}l & D_{11}ll^\top + \tilde{L}\tilde{D}\tilde{L}^\top \end{array}\right) \tag{29}$$

Since we have the necessary conditions $D_{11} = S_{11}$ and $\tilde{P}s = D_{11}l$ the given decomposition is unique for a given $\tilde{P}$.

2. Case: $S_{11} \neq 0$ **and** $\tilde{S} = \frac{ss^\top}{S_{11}}$
Statement 1: The proof of the first case also holds in this case.
Statement 2: The rank of S is 1 and any decomposition has to satisfy the equation

$$\left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \frac{ss^\top}{S_{11}} \end{array}\right) \;=\; \left(\begin{array}{c} L_{11} \\ l \end{array}\right)(D_{11})(L_{11}\; l^\top) \tag{30}$$

which has under the condition $L_{11} \in \{0,1\}$ the unique solution

$$D_{11} = S_{11} \qquad L_{11} = 1 \qquad l = \frac{1}{S_{11}}s \tag{31}$$

and the structure condition is also fulfilled.

Statement 3: Since the rank of S is 1, we choose $P = \mathbf{1}$ and any decomposition has to satisfy

$$\left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \frac{ss^\top}{S_{11}} \end{array}\right) \;=\; \left(\begin{array}{c} 1 \\ l \end{array}\right)(D_{11})(1\; l^\top) = \left(\begin{array}{c|c} D_{11} & D_{11}l^\top \\ \hline D_{11}l & D_{11}ll\top \end{array}\right) \tag{32}$$

with the unique solution $D_{11} = S_{11}$ and $l = s/S_{11}$.

3. Case: $S_{11} = 0$
Statement 1: We decompose the semi–positive $N \times N$ matrix $\tilde{S} = \tilde{L}\tilde{D}\tilde{L}^\top$. By lemma 2 we know, that $s = 0$ and a LDL$^\mathrm{T}$ decomposition is given by

$$\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \tilde{S} \end{array}\right) \;=\; \left(\begin{array}{c|c} 1 & 0 \\ \hline l & \tilde{L} \end{array}\right)\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \tilde{D} \end{array}\right)\left(\begin{array}{c|c} 1 & l^\top \\ \hline 0 & \tilde{L}^\top \end{array}\right) \tag{33}$$

In this situation the decomposition is not unique, since any $l \in \mathbb{R}^N$ fits.

Statement 2: In this situation the rank of S is equal to the rank of $\tilde{S}$ and therefore any decomposition of type 2 must have the structure

$$\left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & \tilde{S} \end{array}\right) = \begin{pmatrix} l^\top \\ \tilde{L} \end{pmatrix} \tilde{D} \begin{pmatrix} l & \tilde{L}^\top \end{pmatrix} \tag{34}$$

The condition of the (1,1) element yields $0 = l\tilde{D}l^\top$. Since $\tilde{D}$ is positive and diagonal we obtain $l = 0$. The remaining condition is $\tilde{S} = \tilde{L}\tilde{D}\tilde{L}^\top$ which has a unique solution under the structure conditions by induction assumption and the matrix L also fulfills these conditions.

Statement 3: Since $S \neq \mathbf{0}$ there is a j such that $S_{jj} \neq 0$. Let P be the permutation matrix which permutes 1 with j. Then there is a unique decomposition of the matrix $U = PSP^\top$, because U belongs to either of the cases studied above.

■

So we are able to give an algorithm for the LDL$^\text{T}$ decomposition of semi–positive matrices. The following recursive algorithm performs the permutations as in the third part of the theorem, but to keep the algorithm more simple, we only deal with quadratic matrices. In order to obtain the decomposition as in part 3 of the theorem, define M as the number of non–zero diagonal elements of D and cut the last $M - N$ columns of L and the last $M - N$ columns and rows of D.

Algorithm 1 (LDL$^\text{T}$ for semi–positive matrices)

1. *If the dimension of S is 1, define $P = (1)$, $L = (1)$ and $D = S$. Stop.*

2. *Choose $\hat{P}$ as the permutation between 1 and j, where j is such, that $S_{jj} \geq S_{ii}$ holds for all $i = 1, \ldots, N$. Define the $N \times N$ matrix U by*

$$\hat{P}S\hat{P}^\top =: U = \left(\begin{array}{c|c} U_{11} & u^\top \\ \hline u & \tilde{U} \end{array}\right) \tag{35}$$

3a. *If $U_{11} > 0$ compute the LDL$^\text{T}$ decomposition of the $(N-1) \times (N-1)$ matrix $\tilde{U} - \frac{1}{U_{11}}uu^\top$:*

$$\tilde{P}(\tilde{U} - \frac{1}{U_{11}}uu^\top)\tilde{P}^\top = \tilde{L}\tilde{D}\tilde{L}^\top \tag{36}$$

Define

$$L := \left(\begin{array}{c|c} 1 & 0 \\ \hline \frac{1}{U_{11}}\tilde{P}u & \tilde{L} \end{array}\right) \qquad D := \left(\begin{array}{c|c} U_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \qquad P := \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & \tilde{P} \end{array}\right)\hat{P} \tag{37}$$

Stop.

3b. If $U_{11} = 0$, then define $P = \hat{P}$, $L = diag(1, \ldots, 1)$ and $D = \mathbf{0}$. Stop.

In step 3 the algorithm makes a case distinction whether U_{11} is zero or not. Under the additional setting $\frac{uu^\top}{U_{11}} = 0$ and $\frac{1}{U_{11}}\tilde{P}u = 0$ in the case $U_{11} = 0$ the part 3b is equivalent to 3a. You will not often find an equation of the form "Division by 0 = 0", but in the context of Singular Value Decomposition (SVD), one also gets such "equality" [7]. This is a remarkable connection between the SVD and the LDL$^\mathrm{T}$ decomposition.

In order to obtain a numerical stable algorithm, one must take care of some points. First of all, it is very critical to decide, whether a value is zero or not. So one defines an $\epsilon > 0$ and every number smaller than ϵ is assumed to be zero. For the choice of ϵ one can take the machine precision for example. If one works with the IEEE representation of floating point numbers, the machine precision is a relative precision; for a 8 byte double the relative precision is about 1e-15 and for a 4 byte float number it is 1e-8. On the other hand we need several (at most N) computations for D_{ii} and therefore one must also consider the roundoffs for this number. So we suggest to take

$$\epsilon = \text{relative precision} \;\cdot\; \max_i S_{ii} \cdot N \tag{38}$$

Unlike in the positive case, the elements D_{ii} are not bounded from below by a positive number and therefore we have to use a pivoting procedure, which will be performed by symmetric permutations on the matrix S. The permutation is chosen such, that the largest diagonal element gets at the leading position. If the pivot is less then ϵ, the corresponding column of L (the arbitrary l) is set to zero.

We will now show, that this algorithm works very well and that it is in fact able to determine the rank of a semi–positive matrix, which is for general matrices an ill–posed problem. Recall that the rank of S is given by the number of non–zero diagonal elements of D.

We take $M < N$ and define a $N \times M$ matrix B which is filled with random numbers in the range of [-10.0,10.0]. Then we define $S = B \cdot B^\top$. By this definition, the rank of S is at most M. We show the determined rank and the relative error of the LDL$^\mathrm{T}$ decomposition for several choices of M, N and we define the relative error by

$$\text{rel. error} \;=\; \frac{||PSP^\top - LDL^\top||_F}{||S||_F} \tag{39}$$

and we recall the Frobenius norm:

$$||A||_F \;:=\; \sqrt{\sum_{ij} A_{ij}^2} \tag{40}$$

The following table shows, that this algorithm is in fact able to determine the rank of a matrix and that the algorithm is backward stable, since the relative error is of the same size as the accuracy of the floating point operations, since we used a 8 byte double.

$M=8, N=15$		$M=40, N=50$		$M=10, N=100$	
detected Rank	rel. Error	det. Rank	rel. Error	det. Rank	rel. Error
8	1.44e-16	40	4.04e-16	10	2.55e-16
8	9.82e-17	40	2.90e-16	10	2.14e-16
8	1.24e-16	40	2.79e-16	10	2.67e-16
8	1.57e-16	40	2.77e-16	10	2.30e-16
8	1.13e-16	40	3.24e-16	10	2.22e-16
8	1.39e-16	40	2.01e-16	10	2.42e-16
8	1.05e-16	40	3.44e-16	10	2.54e-16
8	1.09e-16	40	2.89e-16	10	2.95e-16
8	1.02e-16	40	3.18e-16	10	2.22e-16
8	1.74e-16	40	4.27e-16	10	2.28e-16
8	1.39e-16	40	4.48e-16	10	2.81e-16
8	1.53e-16	40	2.48e-16	10	2.25e-16
8	9.44e-17	40	2.90e-16	10	2.02e-16
8	1.46e-16	40	2.81e-16	10	2.02e-16
8	1.15e-16	40	3.50e-16	10	2.54e-16
8	1.02e-16	40	2.95e-16	10	2.44e-16
8	1.05e-16	40	4.24e-16	10	2.41e-16
8	1.16e-16	40	2.48e-16	10	2.06e-16
8	1.05e-16	40	2.78e-16	10	2.73e-16
8	1.42e-16	40	4.41e-16	10	2.72e-16

The LDL$^{\mathrm{T}}$ decomposition for semi–positive matrices is a very efficient algorithm which can be used for several jobs which occur in the linear algebra. Besides the features of the LDL$^{\mathrm{T}}$ decomposition for positive matrices we also have:

- This algorithm can be used to determine the rank of a semi–positive matrix.
- This algorithm provides a non–squared Cholesky decomposition. We define $C = \hat{L}\hat{D}^{\frac{1}{2}}$ and then holds $S = C \cdot C^{\top}$. We remark that it is as fast as the usual Cholesky decomposition and that it can be used for dimension reduction.
- If S is invertible, one can obtain $S^{-1} = (L^{-1})^{\top} \cdot D^{-1} \cdot L^{-1}$. If S is not invertible one can determine a pseudo inverse S^{+} by $S^{+} = (L^{-1})^{\top} \cdot \tilde{D} \cdot L^{-1}$ where $\tilde{D}$ is diagonal with $\tilde{D}_{ii} = \frac{1}{D_{ii}}$ if $D_{ii} \neq 0$ and $\tilde{D}_{ii} = 0$ if $D_{ii} = 0$.

4 Generalized LDL$^{\mathrm{T}}$ –Decomposition for Indefinite Symmetric Matrices

In the previous sections we showed, that there exists a LDL$^{\mathrm{T}}$ decomposition for positive and semi–positive matrices. But in general, there is no LDL$^{\mathrm{T}}$ decomposition:

Proposition 1 *In general, there is no LDL$^{\mathrm{T}}$ decomposition for symmetric matrices.*

Proof. Consider the following example:

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \overset{?}{=} \begin{pmatrix} 1 & 0 \\ l & 1 \end{pmatrix} \cdot \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix} \cdot \begin{pmatrix} 1 & l \\ 0 & 1 \end{pmatrix} \tag{41}$$

The first column of this matrix equation lead to the following two equations:

$$0 = d_1 \tag{42}$$
$$1 = d_1 l \tag{43}$$

Obviously, this system has no solution for d_1. ■

To understand, what can happen in the case of general symmetric matrices, we again analyze one step of the algorithm for the matrix

$$S = \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \tag{44}$$

If $S_{11} \neq 0$ we can perform the LDL$^{\mathrm{T}}$ algorithm at this step. Even if S_{11} is negative, that does not matter, we have a negative D_{11} then. So the problems arise, if $S_{11} = 0$. Recall that this case can not occur, if S is positive. If S is semi–positive then we know that S_{11} implies $s = 0$ and there is again no problem. In the general case it could happen, that $S_{11} = 0$ and $s \neq 0$. Then the LDL$^{\mathrm{T}}$ algorithm does not work. But of course, this situation may happen for general symmetric matrices and our example is exactly of this kind.

We now extend the LDL$^{\mathrm{T}}$ decomposition algorithm in such a way, that it will give the usual LDL$^{\mathrm{T}}$ decomposition in the case of an semi–positive input. For an indefinite input, we want to obtain the LDL$^{\mathrm{T}}$ decomposition as a semi–positive approximation.

4.1 A first Approximation Algorithm

Let S be a symmetric matrix. The idea is to obtain a semi–positive approximation of S by performing the LDL$^{\mathrm{T}}$ algorithm with some slight modifications. We use the pivoting strategy which we introduced in section 3. So P is a permutation matrix such that S_{11} is the largest diagonal element of the matrix $PSP^\top$.

$$PSP^\top =: \left(\begin{array}{c|c} S_{11} & s^\top \\ \hline s & \tilde{S} \end{array}\right) \approx \left(\begin{array}{c|c} 1 & 0 \\ \hline l & \tilde{L} \end{array}\right) \left(\begin{array}{c|c} D_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \left(\begin{array}{c|c} 1 & l \\ \hline 0 & \tilde{L}^\top \end{array}\right) \tag{45}$$

Algorithm 2 *If $S_{11} > 0$ then one makes a usual LDL$^{\mathrm{T}}$ decomposition step, $D_{11} = S_{11}$ and $l = s/S_{11}$. Then the next LDL$^{\mathrm{T}}$ step will be made on the matrix $\tilde{S} - \frac{ss^\top}{S_{11}}$. If $S_{11} \leq 0$ we set $D_{11} = 0$, $l = 0$ and the next LDL$^{\mathrm{T}}$ step will be performed on $\tilde{S}$.*

If S is semi–positive, algorithm **2** obviously performs the usual LDL$^{\mathrm{T}}$ decomposition on a semi–positive matrix and the result is exact. On the other hand if S is indefinite, $LDL^\top$ is a semi–positive matrix, which can be viewed as an approximation of S.

The disadvantage of this algorithm is, that in general as well the diagonal elements as the off–diagonal elements of the approximation will differ from the original matrix, if some approximation steps have been made. We will therefore introduce an other modification, which yields to a diagonal preserving approximation, if possible, i.e. if all diagonal elements are non–negative.

4.2 A diagonal preserving LDL$^{\mathrm{T}}$ based Approximation Algorithm

In general, the algorithm **2** modifies as well the diagonal elements as the off–diagonal ones. Sometimes however, it may be favorably, to keep the diagonal elements in the semi–positive approximation. Of course this is only possible, if all diagonal elements are non–negative (see lemma 2) and if diagonal elements are negative they will be set to zero. Therefore we now assume, that the diagonal elements are non–negative.

From lemma 2 we know a boundary for the absolute value of an off–diagonal element. Therefore at each step we check the elements of the first column at the cost of $\mathcal{O}(N)$ computations and each element which exceeds the boundary will be set to the nearest value in the range. The total computational effort of this rescaling technique is of order $\mathcal{O}(N^2)$, so that it does not effect the asymptotic computation time.

If the given matrix is semi–positive, one would never detect a violation of the boundary and therefore this diagonal preserving algorithm would yield to the exact LDL$^{\mathrm{T}}$ decomposition. So we summarize the diagonal preserving algorithm of S with non–negative diagonal elements:

Algorithm 3

1. *If the dimension of S is 1, define $P = (1)$, $L = (1)$ and $D = S$. Stop.*
2. *Choose $\hat{P}$ as the permutation between 1 and j, where j is such that $S_{jj} \geq S_{ii}$ holds for all $i = 1, \dots, N$. Define the $N \times N$ matrix U by*

$$\hat{P}S\hat{P}^{\top} \quad =: \quad U = \left(\begin{array}{c|c} U_{11} & u^{\top} \\ \hline u & \tilde{U} \end{array}\right) \tag{46}$$

3. *Rescale the vector u, that is define the vector $\hat{u}$:*

$$\hat{u}_j \quad := \quad sgn(u_j) \min\left(|u_j|, \sqrt{U_{11}U_{jj}}\right) \tag{47}$$

4a. *If $U_{11} > 0$ compute the LDL$^{\mathrm{T}}$ decomposition of the $(N-1) \times (N-1)$ matrix $\tilde{U} - \frac{1}{U_{11}}\hat{u}\hat{u}^{\top}$:*

$$\tilde{P}(\tilde{U} - \frac{1}{U_1 1}\hat{u}\hat{u}^{\top})\tilde{P}^{\top} = \tilde{L}\tilde{D}\tilde{L} \tag{48}$$

Define

$$L := \left(\begin{array}{c|c} 1 & 0 \\ \hline \frac{1}{U_{11}}\tilde{P}\hat{u} & \tilde{L} \end{array}\right) \qquad D := \left(\begin{array}{c|c} U_{11} & 0 \\ \hline 0 & \tilde{D} \end{array}\right) \qquad P := \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & \tilde{P} \end{array}\right)\hat{P} \tag{49}$$

Stop.

4b. If $U_{11} = 0$, then define $P = \hat{P}$, $L = diag(1, \ldots, 1)$ and $D = \mathbf{0}$.

Remark that the rescaling procedure guarantees that the diagonal elements of the remaining matrix are always non–negative. For a number of applications the diagonal preserving feature of algorithm **3** is very important. Of course, this algorithm will give the usual $\mathrm{LDL}^{\mathrm{T}}$ decomposition if the input matrix is semi–positive.

4.3 A generalized non–square Cholesky decomposition

The algorithms **2** and **3** can also be used for dimension reduction in several applications. For a possibly indefinite symmetric $N \times N$ matrix S these algorithms yield to a decomposition

$$PSP^{\top} \cong LDL^{\top} \tag{50}$$

where D is diagonal, L is lower triangular and P is a permutation matrix. In (50) $\cong$ denotes equality if $S \geq 0$ and an approximation if S has negative eigenvalues. Define M as the number of non–zero diagonal elements of D. Cancel the last $N - M$ columns of L and the last $N - M$ rows and columns of D to obtain the matrices $\tilde{L}$ and $\tilde{D}$. Since we have only eliminated zeros we still have:

$$PSP^{\top} \cong \tilde{L}\tilde{D}\tilde{L}^{\top} \tag{51}$$

From this equation we conclude:

$$S \cong \left(P^{\top}\tilde{L}\sqrt{\tilde{D}}\right)\left(P^{\top}\tilde{L}\sqrt{\tilde{D}}\right)^{\top} \tag{52}$$

So we found a Cholesky type decomposition of S with full rank factors. Due to the column elimination $\tilde{L}$ is not square in general and hence we obtain a non–square Cholesky decomposition. If this generalized Cholesky decomposition is an approximation of the matrix S, then the diagonal elements of S will be preserved unless they are non–negative.

Whichever one uses algorithm **2** or **3**, the cost of computation amounts approximately to $N^3/6$ operations consisting of one multiplication, one addition and the loop management. So this algorithm is asymptotical as fast as the usual Cholesky decomposition algorithm.

In fact, this is only a Cholesky type decomposition, since the matrix $C := \tilde{L}\sqrt{\tilde{D}}$ is lower triangular, but due to the row permutations, the matrix PC is not lower triangular in general. Therefore it is useful to store the permutation separately in order to have a control about the zeros, what can help to save time in further processing.

5 Known Approximation of indefinite symmetric Matrices

In practice, one often knows, that a given matrix must be semi–positive by theory, but due to numerical roundoffs or estimation errors one gets an indefinite, symmetric matrix. In this section we recall some known standard techniques for (semi–) positive approximations of symmetric matrices and compare these methods with algorithm **3** in section 6.2.

in order to compare these methods with the algorithm **3** later in section 6.2.

We want to find an approximation S of the matrix S', such that S is positive or semi–positive and that $||S - S'|| = \mathit{small}$. If S' is positive, $S = S'$ must hold. Of course, the condition "$||S - S'|| = \mathit{small}$" is ambiguous, since there are several matrix norms and *small* is also not very clear. It could be perhaps replaced by *minimal*, but we will see, that this cost much more computational time than a relaxed condition, something like "quick and dirty". We first recall some facts about the approximation of indefinite symmetric matrices by semi–positive symmetric matrices.

5.1 Minimal Approximations

5.1.1 Best Approximation in the Frobenius Norm

The most familiar approach to get a semi–definite approximation of an indefinite symmetric matrix S' is given by a spectral decomposition. Let

$$S' \;=\; Q^\top \Lambda' Q \tag{53}$$

with Q orthogonal and Λ' diagonal. Define Λ and S by

$$\begin{aligned} \Lambda &:= \operatorname{diag}\left(\max(0, \Lambda'_{11}), \ldots, \max(0, \Lambda'_{NN})\right) && (54)\\ S &:= Q^\top \Lambda Q && (55) \end{aligned}$$

then the following statement holds:

Theorem 3 *S is the unique best semi–positive approximant of S' with respect to the Frobenius norm, i.e. for all positive symmetric $\tilde{S} \neq S$ holds:*

$$||S' - S||_F < ||S' - \tilde{S}||_F \tag{56}$$

For the proof see N.J. Higham [6], theorem 2.1. The computation of this approximant costs approximately $5N^3$ computations, where one computation consists of one multiplication, one addition and the loop management. Recall that in this respect the multiplication of two square $N \times N$ matrices cost $1 \cdot N^3$ computations.

5.1.2 Best Approximation in the spectral norm

The spectral norm of a matrix A is defined by

$$||A||_2 \;:=\; \sqrt{\text{largest eigenvalue of } A^\top A} \tag{57}$$

Properties of an optimal semi–positive approximation with respect to this norm can be found in Halmos [5], Higham [6].

Theorem 4 *Let S' be a symmetric matrix, $\mathbf{1} = \operatorname{diag}(1, \ldots, 1)$ and define*

$$\delta_2(S') \;:=\; \min\{r \geq 0 \mid S' + r\mathbf{1} \geq 0\} \tag{58}$$

Furthermore, let λ_i denote the eigenvalues of S'.
Then $\delta_2(S') = \max(0, \{-\lambda_i | i = 1, \ldots, N\})$ and the matrix

$$S \;:=\; S' + \delta_2(S')\mathbf{1} \tag{59}$$

is a best semi–positive approximation of S' with respect to the spectral norm with

$$||S' - S||_2 = \delta_2(S') \tag{60}$$

A straight forward algorithm to detect $\delta_2(S')$ is an eigenvalue decomposition. On the other hand, one could also use some bisection algorithm to detect $\delta_2(S')$. For each steps one must decide, whether $S' + r\mathbf{1}$ is semi–positive or not and the LDL^T algorithm can be used for this task. For the details see again the paper of Higham, where also an upper bound for $\delta_2(S')$ is given, which may be used to start the bisection approach.

The optimal approximation S with respect to the spectral norm is not unique, e.g. by the following example. Let

$$S' = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix} \qquad S_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \qquad S_2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \tag{61}$$

then S_1 is the approximant from the theorem. On the other hand we have

$$||S_1 - S'||_2 = ||S_2 - S'||_2 = 2 \tag{62}$$

5.2 Unit–diagonal Approximations

If the symmetric matrix S' is such, that $S'_{ii} = 1$ (e.g. a perturbed correlation matrix) one likes to get a semi–positive approximation S with $S_{ii} = 1$. There are also some algorithms to get such approximations, which we recall for completeness, since we will compare our algorithm **3** with these methods.

5.2.1 Linear Shrinking Method

This method and some generalisations can be found in [2]. For a unit–diagonal symmetric matrix S' one defines

$$\mu \quad = \quad \max\{m \in [0,1] \mid mS' + (1-m)\mathbf{1} \text{ is semi–positive}\} \tag{63}$$

with $\mathbf{1} = \operatorname{diag}(1, \dots, 1)$ and the semi–positive approximation is given by

$$S := \mu S' + (1-\mu)\mathbf{1} \tag{64}$$

There are two techniques to determine μ. One possibility is to perform a bisection and for each bisection step one has to decide, whether the corresponding matrix is semi–positive or not. The latter test can be performed by a $\mathrm{LDL}^{\mathrm{T}}$ algorithm for example.

Another way to determine μ is to determine the eigenvalues. Let λ be the smallest eigenvalue. If λ is less than 0, the matrix S' is indefinite and μ is given by

$$\mu \quad = \quad \frac{1}{1+|\lambda|} \tag{65}$$

5.2.2 Hypersphere Decomposition

Again we assume, that the given $N \times N$ matrix S' is unit diagonal. We know that any unit diagonal, semi–positive approximation S can be decomposed by an lower triangular matrix B, such that $S = BB^{\top}$. Since $S_{ii} = 1$, the rows of B contain unit vectors. An elegant method by Rousseeuw and Molenberghs [10] to describe the matrix B is to use angular coordinates $\Theta_{i,j}$:

$$B_{ij} \quad = \quad \begin{cases} 0 & \text{if } i < j \\ 1 & \text{if } i = j = 1 \\ \cos\Theta_{i,1} & \text{if } i = j > 1 \\ \left(\prod_{l=1}^{i-j} \sin\Theta_{i,l}\right) \cos\Theta_{i,i-j+1} & \text{if } i > j > 1 \\ \prod_{l=1}^{i-j} \sin\Theta_{i,l} & \text{if } i > 1 \text{ and } j = 1 \end{cases} \tag{66}$$

So for the i^{th} row one needs $i-1$ angles to describe this row and the whole matrix B is described by $\frac{(N-1)N}{2}$ angles. Using this parametrization it is convenient to define an error measure, for example by the Frobenius norm:

$$f(\Theta) \quad := \quad \sum_{i,j=1}^{N} (S'_{ij} - (S^{\top})_{ij})^2 \tag{67}$$

In order to find the best approximation S with respect to the error measure (67) one can perform an unconstrained minimum search using the steepest descend method or even more sophisticated methods like conjugate gradients. In our example we suggested to define the error by the Frobenius norm. For this norm we know, that the best approximation is unique and therefore the minimum search will be stable.

The advantage of the hypersphere decomposition approach is that one may use different error measures. However, one has to calibrate $\frac{(N-1)N}{2}$ parameters and for each iteration of the minimizing routine one has to evaluate a matrix product. So one minimizes in a $\mathcal{O}(N^2)$ dimensional space and the evaluation of the function costs about $\mathcal{O}(N^3)$. Therefore the computation time of the hypersphere decomposition algorithm is of $\mathcal{O}(N^5)$ in general, so it is a rather costly procedure for larger dimensions.

5.2.3 Rescaling of a positive Approximation

For a given semi–positive approximation S of S', with positive diagonal elements, e.g. obtained by spectral decomposition (see 5.1.1), we may define:

$$\tilde{S}_{ij} = \frac{S_{ij}}{\sqrt{S_{ii}S_{jj}}} \tag{68}$$

and thus obtain an approximation $\tilde{S}$ with unit diagonal elements.

6 Analysis of the new LDL$^{\mathrm{T}}$ based Algorithms in practice

6.1 Comparison of known LDL$^{\mathrm{T}}$ based Approximation Algorithms

We discuss two known approaches of approximating a symmetric indefinite matrix by a positive symmetric matrix. The first method (**GMW**) is the algorithm by Gill, Murray and Wright (1981), which comes from the context of numerical optimization. The idea is to choose a $\epsilon > 0$ and perform the usual LDL$^{\mathrm{T}}$ decomposition step until a diagonal element is smaller than ϵ. Once a diagonal element less than ϵ occurs, it is set to ϵ and then the LDL$^{\mathrm{T}}$ algorithm proceeds. The value of ϵ has to be chosen carefully to keep the algorithm numerical stable.

In [11] Schnabel and Eskow (**ES**) give an improvement of the **GMW** algorithm. They also gave an error bound for the approximation $||S - S'||$ using the Gerschgorin circle theorem (e.g. see [1]).

Both algorithms have the basic idea to approximate with respect to the spectral norm and therefore to change (increase) the diagonal elements if necessary and they are based on the LDL$^{\mathrm{T}}$ decomposition. But both algorithms also increase the diagonal elements, if the given matrix is only semi–positive but not strict positive, so they are not able to handle rank deficient semi–positive matrices. To compare the different algorithms we consider the same example which has been analyzed in [3] and [11]:

$$S' = \begin{pmatrix} 1 & 1 & 2 \\ 1 & 1 & 3 \\ 2 & 3 & 1 \end{pmatrix} \tag{69}$$

S' is indefinite since the eigenvalues of S' are given by (5.113, 0.089, -2.202). We apply the new algorithms **2** and **3** and compare the results with the results from the algorithms **GMW** and **ES**. One obtains the following approximations:

$$S_{\mathbf{GMW}} = \begin{pmatrix} 3.771 & 1.000 & 2.000 \\ 1.000 & 6.015 & 3.000 \\ 2.000 & 3.000 & 3.242 \end{pmatrix} \qquad S_{\mathbf{ES}} = \begin{pmatrix} 3.000 & 1.000 & 2.000 \\ 1.000 & 3.220 & 3.000 \\ 2.000 & 3.000 & 3.220 \end{pmatrix}$$

$$S_{\mathbf{2}} = \begin{pmatrix} 1.000 & 1.000 & 2.000 \\ 1.000 & 1.000 & 2.000 \\ 2.000 & 2.000 & 4.000 \end{pmatrix} \qquad S_{\mathbf{3}} = \begin{pmatrix} 1.000 & 1.000 & 1.000 \\ 1.000 & 1.000 & 1.000 \\ 1.000 & 1.000 & 1.000 \end{pmatrix}$$

We compare the approximation error with respect to several matrix norms[1]:

Norm	$S_{\mathbf{GMW}}$	$S_{\mathbf{ES}}$	$S_{\mathbf{2}}$	$S_{\mathbf{3}}$
$\|\|S - S'\|\|_2$	5.105	2.220	3.303	2.236
$\|\|S - S'\|\|_{1,\infty}$	5.105	2.220	4.000	3.000
$\|\|S - S'\|\|_F$	6.153	3.722	3.317	3.162

This table shows, that in this example, the algorithm **GMW** is the worst one with respect to all three norms. The algorithm **3** dominates the algorithm **2**, with respect to the different norms. The algorithm **ES** outperforms **2** and **3** with respect to the 1-norm, but one the other hand our new algorithms are better with respect to the Frobenius norm. Note that the algorithm **3** gives a similar spectral norm error like the algorithm **ES**, which is designed to minimize the approximation error with respect to the spectral norm.

From this comparison of the approximation errors, we conclude, that the algorithms **ES** and **2** are comparable and that **3** is the best algorithm in this example, because algorithm **3** minimizes the error with respect to both the Frobenius norm and the spectral norm in the best way.

Another great benefit from the algorithms **2** and **3** is, that the corresponding LDL$^{\mathrm{T}}$ decompositions give a dimension reduction. So from the algorithm **2** respectively **3**

[1]The definition of the $1, \infty$ norm are given by $||S||_1 := \max_j \sum\limits_i |S_{ij}|$ and $||S||_\infty := \max_i \sum\limits_j |S_{ij}|$. Since S is symmetric in our case, we obviously have $||S||_1 = ||S||_\infty$ what explains the notation $||S||_{1,\infty}$ in the table.

one obtains:

$$L_2 = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}, D_2 = (1) \qquad L_3 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, D_3 = (1) \tag{70}$$

Though S_3 in the example is degenerate, it is the best possible approximation with respect to the Frobenius norm under the condition, that the diagonal elements are preserved.

6.2 Comparison of unit–diagonal Approximation Algorithms

Let S be a symmetric, unit–diagonal $N \times N$ matrix. Then we can obtain a semi–positive approximation of S by applying algorithm **3**:

$$PSP^\top \quad \cong \quad LDL^\top \tag{71}$$

where $\cong$ denotes equality if S is semi–positive and if S is indefinite we obtain an approximation. Hence we obtain a semi–positive approximation by algorithm **3** by:

$$S_3 \quad := \quad P^\top LDL^\top P \tag{72}$$

In this section we study the approximation error involved by a semi–positive approximation of perturbed correlation matrices, which in fact are unit–diagonal semi–positive matrices.

We will focus on matrices with rank $M < N$, since only such matrices can become indefinite if small perturbations occur. So we first generate a $M \times N$ matrix B where B_{ij} is iid. U(-1,1) distributed. Then each row of B is rescaled to unit length such that $\sum_j B_{ij}^2 = 1$ and so a correlation matrix S is obtained by $S = BB^\top$. Now, aϵ-perturbation of S is given by $S^\epsilon = S + \epsilon Z$ where Z is a symmetric matrix with diagonal elements 0 and off–diagonal elements ± 1, each with probability $\frac{1}{2}$.

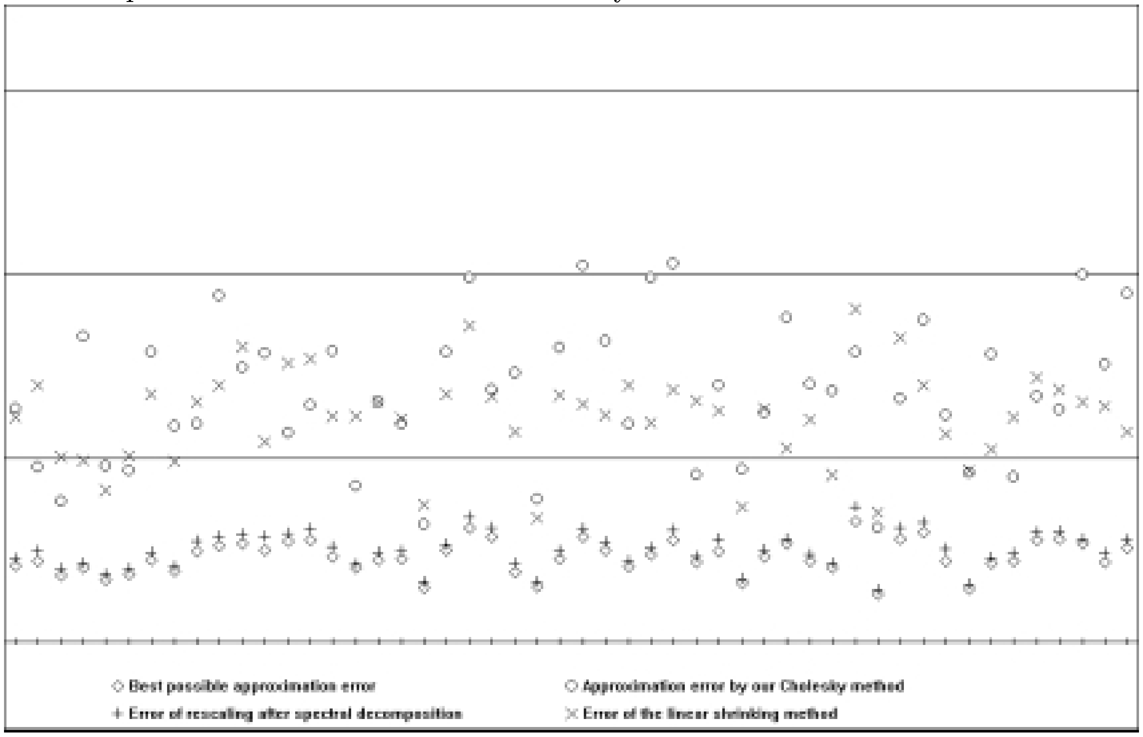
For such a constructed perturbed correlation matrix S^ϵ we want to determine a semi–positive approximant. We observe such an approximant S_{approx} by four algorithms:

- The linear shrinking method described in 5.2.1 yields to the approximation S_{LS}.
- The optimal correlation approximation S_{opt} by using the hypersphere decomposition algorithm described in 5.2.2.
- The scaled correlation approximation S_{scaled} by a rescaling described in (5.2.3) of the best approximation with respect to the Frobenius norm, which one determines by a spectral decomposition (5.1.1).
- The approximation S_3 by using our new algorithm **3**.

In the following picture we have diced 50 perturbed correlation matrices with perturbation $\epsilon = 0.05$, dimension $N = 10$ and the original correlation matrix has rank $M = 5$. Then for each matrix we plotted the distance $||S^\epsilon - S_{approx}||_F$. The distance between S and S^ϵ is given by

$$||S - S^\epsilon||_F \quad = \quad \sqrt{N(N-1)}\;\epsilon \tag{73}$$

and this perturbation distance is indicated by the horizontal lines.



One can see, that the method of rescaling after spectral decomposition attains nearly optimal results. For both methods the approximation error is round about $\frac{1}{2}||S - S^\epsilon||_F$. The approximation errors one gets from the linear shrinking method and from the algorithm **3** lie round about 1–2 times $||S - S^\epsilon||_F$.

Let us consider the complexity of the different methods. The hypersphere decomposition method returns the optimal results, but the computation time of this algorithm is of order $\mathcal{O}(N^5)$, while the other three methods are of order $\mathcal{O}(N^3)$. Clearly, the spectral decomposition methods gives clearly good results but of the costs of round about $5N^3$ operations. The linear shrinking method can be performed by an bisection algorithm where for each iteration one has to decide, whether a given symmetric matrix is positive or not. In order to check a matrix, one could perform a Cholesky based algorithm and the total costs of the linear shrinking method is round about $2N^3$ operations, depending on the desired accuracy of μ in equation (64). The algorithm **3** is with a computation time of $\frac{1}{6}N^3$ operations the fastest algorithm. Depending on the priority of accuracy and computation time one can choose one of these approximation algorithms. If the restriction on the computation time has high priority, our algorithm, which yields comparable approximation errors with the linear shrinking method, seems to be the best choice.

7 Applications to Finance

A typical issue in computational financial mathematics is a Cholesky decomposition of a given usually rank deficient covariance or correlation matrix. In this respect our proposed algorithm **3** provides a useful tool, in particular when the size of the matrix under consideration is large or when the matrix has small negative eigenvalues due to noisy observations. For instance, an estimated covariance or correlation matrix may have negative eigenvalues when the available time series of data is not long enough. Indeed, **3** returns a semi–positive decomposition of a semi–positive matrix and a semi–positive approximation in the case, that the matrix is indefinite. Also the diagonal preserving property of the algorithm is desirable in practice, in particular for the decomposition of correlation matrices and the computation efficiency is comparably with the usual Cholesky decomposition.

7.1 Monte Carlo Evaluations

There are several applications of Monte Carlo methods in finance, such as risk management and option pricing of complex financial products. In risk management one typically needs to simulate a multivariate N dimensional vector X with mean vector m and covariance matrix C. This can be done by a non–square Cholesky decomposition of C

$$P \cdot C \cdot P^\top \quad = \quad B \cdot B^\top \tag{74}$$

and one obtains the permutation P and the $N \times M$ matrix B where M is the rank of C. For one sample of X one needs to generate $M \leq N$ iid N(0,1) random numbers, which are the components of the vector Z. and X is given by:

$$X \quad = \quad m + P^\top B \cdot Z \tag{75}$$

The property of our new non–square Cholesky decomposition to detect possible linear dependencies can improve the efficiency of Monte Carlo Methods, since one does not need to generate N but only $M \leq N$ normal distributed random numbers per sample. Of course, the decomposition has to be made once at the beginning of the Monte Carlo simulation only.

7.2 Stress Testing in Risk Management

An in practice widely-used risk measure is the Value at Risk (VaR), which is equal to the amount of money, such that the loss of the portfolio will not exceed the VaR within a certain time horizon with a given probability. A simple and well known approach to determine the VaR is the delta-normal-approximation:

$$\text{VaR} \quad = \quad k\Delta^\top S \Delta \tag{76}$$

where Δ is the vector of portfolio sensitivities, S is a covariance matrix and k is a constant, which depends on the given probability and time horizon. By rescaling

$$C_{ij} := \frac{S_{ij}}{\sqrt{S_{ii}S_{jj}}} \qquad d_i := \Delta_i \sqrt{S_{ii}} \tag{77}$$

we obtain

$$\text{VaR} = k\, d^\top \cdot C \cdot d \tag{78}$$

where C denotes the correlation matrix of the different risk factors.

In order to evaluate the VaR by this approach, it is not necessary to decompose the correlation matrix and to check, whether it is semi–positive or not. But if such a check will not be made, the risk manager would not have the guaranty, that the VaR is positive. Even if the resulting VaR is positive, there is of course the question, how reliable such a result would be.

In practice, a risk manager will change certain correlations in order to see how the risk will change, if some correlations change (stress testing). Further, it may happen that some entries in the correlation matrix cannot be determined from time series and so they have to be guessed, for example the entries concerning the stock of a firm just after an acquisition or an initial public offering. In such situations the risk manager wants to see, how the VaR depends on a certain correlation. So it is necessary to perform several computations of the VaR, and each computation requires a semi–positive approximation of the perturbed correlation matrix which can be obtained by algorithm **3**. If $BB^\top$ is the non–square Cholesky decomposition of $PCP^\top$, the VaR can be calculated by:

$$\text{VaR} = k(PB^\top d)^\top \cdot (PB^\top d) \tag{79}$$

This VaR is always based on a semi–positive correlation matrix and is therefore much more reliable than a simple calculation based on (78).

7.3 Delta–Gamma–Normal Approach to Value at Risk

Another application of the algorithm **3** is the computation of the VaR by the delta–gamma–approach. In this delta–gamma approximation one assumes, that the value V of the portfolio at some future time T is given by

$$V_T = V_0 + \Delta^\top X + X^\top \Gamma X \tag{80}$$

where X is multivariate normal distributed with mean 0 and covariance C. Again, we decompose $C \cong P^\top BB^\top P$, where P is a permutation matrix and B is a $N \times M$ matrix. Then if Z is a M–dimensional vector of iid N(0,1) random variables, we have $X = P^\top BZ$ and the portfolio value can be written as:

$$\begin{aligned} V_T &= V_0 + \left(B^\top P\Delta\right)^\top Z + Z^\top \left(B^\top P\Gamma P^\top B\right) Z && (81)\\ &=: V_0 + \tilde{\Delta}^\top Z + Z^\top \tilde{\Gamma} Z && (82)\end{aligned}$$

Note that $\tilde{\Delta}$ and $\tilde{\Gamma}$ are M–dimensional, so the dimension of this problem is reduced. In applications a typical value for N is 1000 and the covariances are estimated from a one year time series and so the rank M of the covariance matrix can not exceed the length of this series of typical 250 days.

This aspect is very important in this situation, since for the next step in the calculation one needs to perform an eigenvalue decomposition. If the dimension will be reduced by a non–square Cholesky decomposition, this saves a lot of computation time, since this part is quite expensive for the delta–gamma–normal method to determine the VaR. After the eigenvalue decomposition the characteristic function of the portfolio is known and the profit and loss distribution (hence also the VaR) can be obtained by Fourier inversion. A detailed description of this algorithm goes beyond the scope of this paper and can be found in [9].

We want to emphasize that the dimension reduction feature of the non–square Cholesky decomposition gives a great reduction of the computation time for the delta–gamma VaR–algorithm. Of course, the idea of stress testing can also be applied in this context and demands again a fast decomposition algorithm.

Acknowledgements

Oliver Reiß thanks John Schoenmakers for stimulating discussions and helpful comments.

This paper was developed in cooperation with the bmb+f project "Effiziente Methoden zur Bestimmung von Risikomaßen", which is also supported by Bankgesellschaft Berlin AG.

References

[1] Richard A. Brualdi, Stephen Mellendorf: Regions in the Complex Plane Containing the Eigenvalues of a Matrix, *Amer. Math. Monthly* **101**, pp. 975–985, (1994).

[2] Susan J, Devlin, R. Gnanadesikan, J.R.Kettenring: Robust estimation and outlier detection with correlation coefficients, *Biometrika*, Vol. **62**, No. 3, pp. 531–545, (1975).

[3] Philip E. Gill, Walter Murray, Margaret H. Wright: *Practical Optimization*, Academic Press, (1981).

[4] Gene H. Golub, Charles F. van Loan: *Matrix Computations*, third edition, Johns Hopkins University Press, Baltimore, (1996).

[5] P.R.Halmos : Positive approximants of operators, *Indiana Univ. Math. J.*, **21**, pp.951–960, (1972).

[6] Nicholas J. Higham: Computing a Nearest Symmetric Positive Semidefinit Matrix, *Linear Algebra and Its Applications*, **103**, pp. 103–118, (1988).

[7] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery: *Numerical Recipes in C : The Art of Scientific Computing*, Second Edition, Cambridge University Press, Cambridge, (1992).

[8] Riccardo Rebonato, Peter Jäckel: The most general methodology to create a valid correlation matrix for risk management and option pricing purposes, working paper available at `http://www.rebonato.com`, October (1999).

[9] Oliver Reiß, Johann-Hinrich Zacharias-Langhans: The Fourier Inversion Algorithm for the computation of Value at Risk using the Delta Gamma Normal Approximation, working paper, (2002).

[10] Peter J. Rousseeuw, Geert Molenberghs: Transformation of non positive semidefinite correlation matrices, *Commun. Statist. - Theory Meth.*, Vol. **22**, pp.965–984, (1993).

[11] Robert B. Schnabel, Elizabeth Eskow: A new modified Cholesky Factorization, *SIAM J. Sci. Stat. Comput.*, Vol **11**, No **6**, pp. 1136–1158, (1990).