

Numerical Analysis Takehome Exam 1

Magnus Erik Hvass Pedersen (971055)
October 2004, Daimi, University of Aarhus

Introduction

Consider n -dimensional tridiagonal matrices of the following symmetric form:

$$A_n(\beta) = \begin{bmatrix} 1 & \beta & & \\ \beta & 1 & \beta & \\ & & \ddots & \beta \\ & & & \beta & 1 \end{bmatrix}$$

and of the form:

$$B_n(\beta) = \begin{bmatrix} 1 & \beta & & \\ -\beta & 1 & \beta & \\ & & \ddots & \beta \\ & & & -\beta & 1 \end{bmatrix}$$

which is notably not skew-symmetric, as the diagonal is non-zero. Let a , b , and c denote the diagonal vectors for these matrices, that is:

$$c^T = [\beta \cdots \beta]$$

with $a = c$ for $A_n(\beta)$, and for $B_n(\beta)$:

$$a^T = [-\beta \cdots -\beta]$$

where both a and c are $(n - 1)$ -dimensional. The diagonal vector b is n -dimensional and filled with 1's:

$$b^T = [1 \cdots 1]$$

The following sections develop theory and methods for these particular kinds of matrices, enabling us to rapidly compute various aspects, including LU -factorization, the inverse, infinity norms, and condition numbers. These methods are then applied by hand to answer most of the questions posed in the exam, but the results are also compared to those obtained with the computer program *MatLab*. Finally, a small computer program implements the LU -factorization and calculation of infinity-norms for such matrices.

LU -Factorization

To begin with, we wish to factor $A_n(\beta)$ into the product of a unit lower triangular matrix $L_n(\beta)$, and an upper triangular matrix $U_n(\beta)$. This will be our foundation in the development of properties and methods for these matrices.

If we initially ignore whether the elements of $L_n(\beta)$ and $U_n(\beta)$ are mutually consistent, in that their product is a tridiagonal matrix of the desired form, then it is clear that the lower triangular matrix can be of the form:

$$L_n(\beta) = \begin{bmatrix} 1 & & & & \\ l_1 & 1 & & & \\ & & \ddots & & \\ & & & l_{n-1} & 1 \end{bmatrix} \quad (1)$$

and the upper triangular matrix being:

$$U_n(\beta) = \begin{bmatrix} d_1 & u_1 & & & \\ & d_2 & u_2 & & \\ & & \ddots & & \\ & & & u_{n-1} & \\ & & & & d_n \end{bmatrix} \quad (2)$$

since multiplying such matrices will produce values on the diagonal and to its left and right. This is confirmed by performing the matrix multiplication, where we find $L_n(\beta)U_n(\beta)$ to be:

$$\begin{bmatrix} d_1 & u_1 & & & \\ l_1 d_1 & l_1 u_1 + d_2 & u_2 & & \\ & l_2 d_2 & l_2 u_2 + d_3 & u_3 & \\ & & & \ddots & u_{n-1} \\ & & & l_{n-1} d_{n-1} & l_{n-1} u_{n-1} + d_n \end{bmatrix}$$

Then for this $L_n(\beta)U_n(\beta)$ to equal $A_n(\beta)$, we clearly have $u = c$, meaning that:

$$u_i = \beta$$

And for the diagonal b of the tridiagonal matrix, the following must be satisfied:

$$b_i = l_{i-1}u_{i-1} + d_i$$

and since all b_i 's equal 1, the first d_i is easily found:

$$1 = b_1 = d_1$$

And rewriting the iterative case, we get:

$$1 = b_i = l_{i-1}u_{i-1} + d_i = l_{i-1}\beta + d_i \Leftrightarrow d_i = 1 - l_{i-1}\beta$$

And as we have the following for a :

$$a_i = l_i d_i$$

we can use the iterative case for d_i as follows:

$$a_i = l_i(1 - l_{i-1}\beta) \Leftrightarrow l_i = \frac{a_i}{1 - l_{i-1}\beta}$$

where $a_i = \beta$ for $A_n(\beta)$, and $-\beta$ for $B_n(\beta)$. Since $d_1 = 1$ we have the following for the base-case:

$$a_1 = l_1 d_1 = l_1$$

again meaning that $l_1 = a_1 = \beta$ for $A_n(\beta)$, and $l_1 = -\beta$ for $B_n(\beta)$.

So not only is the above form of the LU -factorization valid, we have also found a recursive relationship between their elements. Because of the simple dependence of l_i on l_{i-1} , it is plausible that there exist explicit (i.e. non-recursive) expressions for l_i and d_i .

Algorithm

Let $\alpha = \beta$ if we are to LU -factorize $A_n(\beta)$, and $\alpha = -\beta$ for $B_n(\beta)$. From the above, we have that the algorithm for LU -factorization is as follows:

- Set $l_1 = \alpha$.
- For each l_i from $i = 2$ up to $n - 1$, set:

$$l_i = \frac{\alpha}{1 - l_{i-1}\beta} \tag{3}$$

and note that $t \neq 0 \Leftrightarrow l_{i-1}\beta \neq 1$ is a required condition.

- This completes the $L_n(\beta)$ matrix, now compute $U_n(\beta)$:
- Set $d_1 = 1$ and $u_i = \beta$ for all $i \in \{1, \dots, n - 1\}$.
- For each d_i from $i = 2$ up to n , set:

$$d_i = 1 - l_{i-1}\beta \tag{4}$$

using l from the $L_n(\beta)$ matrix. Note that we could reuse parts of the loop calculating the l -vector, but we require one more iteration for calculating the d -vector.

- This completes the upper triangular matrix $U_n(\beta)$.

Since there are no nested loops in the algorithm, it can directly be seen to be of time-complexity $O(n)$, with n being the dimensionality of the tridiagonal matrix in question. This is a significant improvement over Doolittle's factorization [1, p. 155].

Inverse Matrix

If we can find an LU -factorization of a matrix A , then its inverse is also given by:

$$A^{-1} = (LU)^{-1} = U^{-1}L^{-1} \tag{5}$$

Finding L^{-1} can be done using Gaussian elimination:

$$[L | I] \sim \dots \sim [I | L^{-1}]$$

and similarly for U^{-1} . Nevertheless, some shortcuts can be taken, from assuming the special form of $L_n(\beta)$ and $U_n(\beta)$.

Inverse of $L_n(\beta)$

Beginning with the unit lower triangular $L_n(\beta)$ from Eq.(1), we have the following sequence of eliminations:

$$\begin{aligned}
[L_n(\beta) | I] &\sim \left[\begin{array}{cccc|cccc} 1 & & & & 1 & & & \\ l_1 & 1 & & & & 1 & & \\ & l_2 & 1 & & & & 1 & \\ & & l_3 & 1 & & & & 1 \\ & & & \ddots & & & & \ddots \\ & & & & l_{n-1} & 1 & & & 1 \end{array} \right] \\
&\sim \left[\begin{array}{cccc|cccc} 1 & & & & 1 & & & \\ 0 & 1 & & & -l_1 & & & \\ & l_2 & 1 & & & 1 & & \\ & & l_3 & 1 & & & 1 & \\ & & & \ddots & & & & \ddots \\ & & & & l_{n-1} & 1 & & & 1 \end{array} \right] \\
&\sim \left[\begin{array}{cccc|cccc} 1 & & & & 1 & & & \\ 0 & 1 & & & -l_1 & 1 & & \\ & 0 & 1 & & l_1 l_2 & -l_2 & 1 & \\ & & l_3 & 1 & & & & 1 \\ & & & \ddots & & & & \ddots \\ & & & & l_{n-1} & 1 & & & 1 \end{array} \right] \\
&\sim \left[\begin{array}{cccc|cccc} 1 & & & & 1 & & & \\ 0 & 1 & & & -l_1 & 1 & & \\ & 0 & 1 & & l_1 l_2 & -l_2 & 1 & \\ & & 0 & 1 & -l_1 l_2 l_3 & l_2 l_3 & -l_3 & 1 \\ & & & \ddots & & & & \ddots \\ & & & & l_{n-1} & 1 & & & 1 \end{array} \right] \\
&\sim \dots
\end{aligned}$$

From this, the following structure is evident¹ for $L_n^{-1}(\beta)$ (or L^{-1} for short):

$$L_{ij}^{-1} = \begin{cases} 0 & , \text{ if } i < j \\ 1 & , \text{ if } i = j \\ (-1)^{i+j} \prod_{k=j}^{i-1} l_k & , \text{ otherwise, i.e. if } i > j \end{cases} \quad (6)$$

where L_{ij}^{-1} is of course taken to mean the element at the i 'th row and j 'th column of matrix L^{-1} . Note that the above structure is independent of whether

¹But could be proven more formally with induction.

L is the lower triangular matrix for $A_n(\beta)$ or $B_n(\beta)$,² and also note that $L_n^{-1}(\beta)$ is itself a unit lower triangular matrix.

The algorithm for creating $L_n^{-1}(\beta)$ is given by:

- Assign 1's to the diagonal, i.e. for $i = 1$ to n do the following:

$$L_{ii}^{-1} = 1$$

and assign 0's in the upper triangle, that is whenever $i < j$:

$$L_{ij}^{-1} = 0$$

- For each row $i = 2$ to n do the following:
 - Compute the i 'th row from the $(i - 1)$ 'th row, by assigning each column element before the diagonal, i.e. from $j = 1$ up to $i - 1$, as follows:

$$L_{ij}^{-1} = -L_{(i-1,j)}^{-1} \cdot l_{i-1}$$

Which clearly works in quadratic time and space as all the elements of the $n \times n$ matrix are either assigned or calculated with a fixed number of operations.

²That is, L^{-1} will be different for $A_n(\beta)$ and $B_n(\beta)$, but the structure of $L_n^{-1}(\beta)$ is the same, as is the formula for calculating it.

Inverse of $U_n(\beta)$

The inverse of the upper triangular matrix $U_n(\beta)$ from Eq.(2), is also reduced by Gaussian elimination:

$$\begin{aligned}
& [U_n(\beta) | I] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & d_{n-2} & \beta & & \\ & & & d_{n-1} & \beta & \\ & & & & d_n & \end{array} \right] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & d_{n-2} & \beta & & \\ & & & d_{n-1} & \beta & \\ & & & & 1 & \\ & & & & & d_n^{-1} \end{array} \right] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & d_{n-2} & \beta & & \\ & & & d_{n-1} & 0 & \\ & & & & 1 & \\ & & & & & -d_n^{-1}\beta \\ & & & & & d_n^{-1} \end{array} \right] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & d_{n-2} & \beta & & \\ & & & 1 & 0 & \\ & & & & 1 & \\ & & & & & d_{n-1}^{-1} \\ & & & & & -d_{n-1}^{-1}d_n^{-1}\beta \\ & & & & & d_n^{-1} \end{array} \right] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & d_{n-2} & 0 & & \\ & & & 1 & 0 & \\ & & & & 1 & \\ & & & & & -d_{n-1}^{-1}\beta \\ & & & & & d_{n-1}^{-1} \\ & & & & & d_{n-1}^{-1}d_n^{-1}\beta^2 \\ & & & & & -d_{n-1}^{-1}d_n^{-1}\beta \\ & & & & & d_n^{-1} \end{array} \right] \\
& \sim \left[\begin{array}{ccc|ccc} d_1 & \beta & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & 0 & & \\ & & & 1 & 0 & \\ & & & & 1 & \\ & & & & & d_{n-2}^{-1} \\ & & & & & -d_{n-2}^{-1}d_{n-1}^{-1}\beta \\ & & & & & d_{n-1}^{-1} \\ & & & & & d_{n-2}^{-1}d_{n-1}^{-1}d_n^{-1}\beta^2 \\ & & & & & -d_{n-1}^{-1}d_n^{-1}\beta \\ & & & & & d_n^{-1} \end{array} \right] \\
& \sim \dots
\end{aligned}$$

Although a bit more involved than for $L_n^{-1}(\beta)$, the structure for $U_n^{-1}(\beta)$ (or U^{-1} for short) is as follows:³

$$U_{ij}^{-1} = \begin{cases} 0 & , \text{ if } i > j \\ (-1)^{i+j} \beta^{j-i} \prod_{k=i}^j d_k^{-1} & , \text{ otherwise, i.e. if } i \leq j \end{cases} \quad (7)$$

³Again, this could be proven more formally using induction.

Which is again an upper triangular matrix.

The algorithm for creating $U_n^{-1}(\beta)$ is similar to that for creating $L_n^{-1}(\beta)$, but its main loop iterates backwards through the rows of the matrix instead:

- Assign the diagonal elements, i.e. for $i = 1$ to n do the following:

$$U_{ii}^{-1} = d_i^{-1}$$

and assign 0's in the lower triangle, that is whenever $i > j$, we have:

$$U_{ij}^{-1} = 0$$

- For each row from $i = n - 1$ down to 1 do the following:
 - Compute the i 'th row from the $(i + 1)$ 'th row, by assigning each column element after the diagonal, i.e. from $j = i + 1$ up to n , as follows:

$$U_{ij}^{-1} = -U_{(i+1,j)}^{-1} \cdot d_i^{-1} \cdot \beta$$

Which also works in quadratic time and space.

Inverse of $A_n(\beta)$

Now having obtained the lower triangular matrix L^{-1} and the upper triangular matrix U^{-1} , we may calculate the inverse of $A_n(\beta)$ using Eq.(5) as follows:

$$A^{-1} = U^{-1}L^{-1}$$

meaning the elements of A^{-1} are given by:

$$A_{ij}^{-1} = \sum_{s=1}^n U_{is}^{-1}L_{sj}^{-1} = \sum_{s=\max(i,j)}^n U_{is}^{-1}L_{sj}^{-1}$$

since the upper triangle of L^{-1} and the lower triangle of U^{-1} only contain zeros. But as A is also symmetric, we have that:

$$I = (AA^{-1})^T = (A^{-1})^T A^T = (A^{-1})^T A$$

and because the inverse is unique, this means that the inverse is also symmetric:

$$A^{-1} = (A^{-1})^T$$

and we can therefore limit our attention to one half of the inverse matrix, or more specifically, the elements A_{ij}^{-1} for which $i \leq j$, so that we have:

$$A_{ij}^{-1} = \sum_{s=\max(i,j)}^n U_{is}^{-1}L_{sj}^{-1} = \sum_{s=j}^n U_{is}^{-1}L_{sj}^{-1}$$

But since L_{ij}^{-1} is defined differently for the case $i = j$, we must rewrite as follows:

$$A_{ij}^{-1} = U_{ij}^{-1} L_{jj}^{-1} + \sum_{s=j+1}^n U_{is}^{-1} L_{sj}^{-1} = U_{ij}^{-1} + \sum_{s=j+1}^n U_{is}^{-1} L_{sj}^{-1}$$

From Eqs.(6) and (7) above, we then have:

$$A_{ij}^{-1} = (-1)^{i+j} \beta^{j-i} \prod_{k=i}^j d_k^{-1} + \sum_{s=j+1}^n \left((-1)^{i+s} \beta^{s-i} \prod_{k=i}^s d_k^{-1} \right) \cdot \left((-1)^{s+j} \prod_{k=j}^{s-1} l_k \right)$$

which can be simplified somewhat to:

$$A_{ij}^{-1} = (-1)^{i+j} \cdot \beta^{-i} \left(\beta^j \prod_{k=i}^j d_k^{-1} + \sum_{s=j+1}^n \beta^s \left(\prod_{k=i}^s d_k^{-1} \right) \cdot \left(\prod_{k=j}^{s-1} l_k \right) \right) \quad (8)$$

again with $i \leq j$, and with the rest of the matrix being assigned as follows:

$$A_{ij}^{-1} = A_{ji}^{-1}, \quad i > j$$

So unless we explicitly need the inverse of $L_n(\beta)$ and $U_n(\beta)$ for some other purpose, it is not necessary to compute them in order to find the inverse of $A_n(\beta)$, all we need is to compute the vectors l and d , and then use Eq.(8).

Computing $A_n^{-1}(\beta)$ in Quadratic Time

To create an efficient algorithm we seek a way to generate the elements of the inverse matrix from their neighbours. If we write out Eq.(8) for $i - 1$, we find that $A_{(i-1,j)}^{-1}$ equals:

$$(-1)^{(i-1)+j} \cdot \beta^{-(i-1)} \left(\beta^j \prod_{k=i-1}^j d_k^{-1} + \sum_{s=j+1}^n \beta^s \left(\prod_{k=i-1}^s d_k^{-1} \right) \cdot \left(\prod_{k=j}^{s-1} l_k \right) \right)$$

which can be rewritten to:

$$(-1)^{(i-1)+j} \cdot \beta^{-(i-1)} \cdot d_{i-1}^{-1} \left(\beta^j \prod_{k=i}^j d_k^{-1} + \sum_{s=j+1}^n \beta^s \left(\prod_{k=i}^s d_k^{-1} \right) \cdot \left(\prod_{k=j}^{s-1} l_k \right) \right)$$

and noting that $(-1)^{(i-1)+j} = (-1)^{(i+1)+j} = (-1)(-1)^{i+j}$, we then find:

$$A_{(i-1,j)}^{-1} = -\beta \cdot d_{i-1}^{-1} \cdot A_{ij}^{-1}$$

for $i \leq j$, and the rest of the matrix generated symmetrically. We may also view this from the perspective of row i , and equivalently have:

$$A_{ij}^{-1} = -\beta \cdot d_i^{-1} \cdot A_{(i+1,j)}^{-1} \quad (9)$$

Now we just need to find the diagonal of A^{-1} , which can be computed in linear time, beginning with the last element:

$$A_{nn}^{-1} = d_n^{-1}$$

and then computing the rest of the diagonal by traversing it backwards. That is, for $i = n - 1$ down to 1, compute the following:

$$A_{ii}^{-1} = \left(1 + \beta \cdot l_i \cdot A_{(i+1,i+1)}^{-1}\right) d_i^{-1} \quad (10)$$

which can be shown by induction, or one might be informally convinced by studying the form of the L^{-1} and U^{-1} matrices above.

The diagonal is computed in linear time and each column-element above the diagonal uses a fixed amount of computational time. We therefore have that the algorithm for computing $A_n^{-1}(\beta)$ using Eqs.(9) and (10) uses constant time for each element, and is hence of $O(n^2)$ time complexity. Once more, note that no time and space is required to construct the L^{-1} and U^{-1} matrices, because only the l - and d -vectors are being used.

Symmetries of $A_n^{-1}(\beta)$

In the algorithm for computing the lower triangle of $A_n^{-1}(\beta)$, we use that $A_n^{-1}(\beta)$ is a symmetric matrix. Symmetry can be thought of, as the matrix mirroring itself around the diagonal axis from the element at position $(1, 1)$ and down to (n, n) . However, there is another symmetry present in both $A_n^{-1}(\beta)$ and $B_n^{-1}(\beta)$, which will be exploited in the computation of their matrix norms below. In lack of a better name, let us refer to it as *cross*-symmetry, because it mirrors the matrix onto itself around the axis going from position $(1, n)$ down to $(n, 1)$. To prove that this symmetry exists, first define the *cross*-transpose as:

$$(A^{\mathbb{T}})_{ij} = A_{(n-j+1, n-i+1)} \quad (11)$$

with a cross-symmetric matrix thus satisfying:

$$A^{\mathbb{T}} = A \quad (12)$$

The proof that the inverse of a cross-symmetric matrix is itself cross-symmetric, resembles the above proof concerning the regular matrix-transposition. To this end, we will first need to prove the following property of the cross-transpose of a product:

$$(AB)^{\mathbb{T}} = B^{\mathbb{T}}A^{\mathbb{T}}$$

Remembering that an element in the result of a matrix-product is given by:

$$(AB)_{ij} = \sum_{s=1}^n A_{is}B_{sj}$$

we have from the definition in Eq.(11) that:

$$(AB)_{ij}^{\mathbb{T}} = (AB)_{(n-j+1, n-i+1)} = \sum_{s=1}^n A_{(n-j+1, s)} B_{(s, n-i+1)}$$

And similarly we have the product of two cross-transposed matrices:

$$(B^{\mathbb{T}} A^{\mathbb{T}})_{ij} = \sum_{s=1}^n B_{is}^{\mathbb{T}} A_{sj}^{\mathbb{T}} = \sum_{s=1}^n B_{(n-s+1, n-i+1)} A_{(n-j+1, n-s+1)}$$

But the difference between these two expressions lies merely in the direction of the summation. To see this, let $s' = n - s + 1$ be a new summation-index. we then have:

$$(B^{\mathbb{T}} A^{\mathbb{T}})_{ij} = \sum_{s'=1}^n B_{(s', n-i+1)} A_{(n-j+1, s')} = (AB)_{ij}^{\mathbb{T}}$$

which proves the property. Using this, we can now show that if some matrix A is cross-symmetric and invertible, then its inverse is also cross-symmetric. We have that:

$$I = I^{\mathbb{T}} = (AA^{-1})^{\mathbb{T}} = (A^{-1})^{\mathbb{T}} A^{\mathbb{T}} = (A^{-1})^{\mathbb{T}} A$$

where the last equality stems from the assumption on cross-symmetry. Since the inverse matrix is unique and therefore the only matrix having the property $AA^{-1} = A^{-1}A = I$, we have that:

$$A^{-1} = (A^{-1})^{\mathbb{T}} \tag{13}$$

so the inverse is indeed also cross-symmetric.

This means that $A_n^{-1}(\beta)$ can be generated by producing just a fourth of the matrix using Eqs.(10) and (9), the other fourth by cross-symmetric duplication of elements, and the last half by symmetric duplication of elements. The same goes for $B_n^{-1}(\beta)$, with the only difference being its alternation of sign in its symmetric duplication, as proved next.

Inverse of $B_n(\beta)$

As long as the l - and d -vectors are found from $B_n(\beta)$, Eqs.(9) and (8) can also be used to calculate the upper triangle of $B_n^{-1}(\beta)$, but the lower triangle is unfortunately not simply its transpose, as was the case with $A_n^{-1}(\beta)$.

Relationship Between Elements in $B_n^{-1}(\beta)$

To deduce the lower triangular part of $B_n^{-1}(\beta)$ from its upper triangular part, first note that:

$$BB^{-1} = B^{-1}B = I$$

and therefore that I_{ij} equals row i of B multiplied with column j of B^{-1} , and this also equals row i of B^{-1} multiplied with column j of B :

$$I_{ij} = \sum_{s=1}^n B_{is} B_{sj}^{-1} = \sum_{s=1}^n B_{is}^{-1} B_{sj}$$

Since B is of the particular tridiagonal form with $B_{ij} = 0$ for $j \notin \{i-1, i, i+1\}$, we have that:

$$I_{ij} = B_{(i,i-1)} B_{(i-1,j)}^{-1} + B_{ii} B_{ij}^{-1} + B_{(i,i+1)} B_{(i+1,j)}^{-1}$$

but also:

$$I_{ij} = B_{(i,j-1)}^{-1} B_{(j-1,j)} + B_{ij}^{-1} B_{jj} + B_{(i,j+1)}^{-1} B_{(j+1,j)}$$

Putting these two identities together, and using that $B_{ii} = 1$, $B_{(i+1,i)} = B_{(i,i-1)} = -\beta$, and $B_{(i,i+1)} = B_{(i-1,i)} = \beta$, we have:

$$-\beta B_{(i-1,j)}^{-1} + B_{ij}^{-1} + \beta B_{(i+1,j)}^{-1} = \beta B_{(i,j-1)}^{-1} + B_{ij}^{-1} - \beta B_{(i,j+1)}^{-1}$$

which is equivalent to:

$$B_{(i+1,j)}^{-1} = B_{(i-1,j)}^{-1} + B_{(i,j-1)}^{-1} - B_{(i,j+1)}^{-1}$$

where we let $B_{ij}^{-1} = 0$ for $i \leq 0 \vee i > n$ or $j \leq 0 \vee j > n$. From this, the local relationship between elements in $B_n^{-1}(\beta)$, can be likened to a geometric rhombus, and through rearranging and changes of indices, we get:

$$B_{(i,j)}^{-1} = B_{(i-2,j)}^{-1} + B_{(i-1,j-1)}^{-1} - B_{(i-1,j+1)}^{-1} \quad (14)$$

$$B_{(i,j)}^{-1} = B_{(i+2,j)}^{-1} - B_{(i+1,j-1)}^{-1} + B_{(i+1,j+1)}^{-1} \quad (15)$$

$$B_{(i,j)}^{-1} = B_{(i+1,j+1)}^{-1} - B_{(i-1,j+1)}^{-1} + B_{(i,j+2)}^{-1} \quad (16)$$

$$B_{(i,j)}^{-1} = B_{(i-1,j-1)}^{-1} + B_{(i,j-2)}^{-1} - B_{(i+1,j-1)}^{-1} \quad (17)$$

which are all equivalent, but suitable for different situations. For example Eq.(14) relates the element at position (i, j) to elements above it in the matrix, whereas Eq.(15) relates it to elements below in the matrix. Keeping this rhombus-structure in mind, will ease the understanding of the sections to follow, and provide insight into why particular assumptions were made, and the inductive proofs made the way they are.

Various Symmetries of $B_n^{-1}(\beta)$

If we try setting $i = 2$ and $j = 1$ in Eq.(14), we have:

$$B_{(2,1)}^{-1} = B_{(0,1)}^{-1} + B_{(1,0)}^{-1} - B_{(1,2)}^{-1} = -B_{(1,2)}^{-1}$$

Now assume that $B_{(i-1,i-2)}^{-1} = -B_{(i-2,i-i)}^{-1}$ holds. From Eq.(14) we find that the element at position $(i, i-1)$ is:

$$B_{(i,i-1)}^{-1} = B_{(i-2,i-1)}^{-1} + B_{(i-1,i-2)}^{-1} - B_{(i-1,i)}^{-1}$$

and using the assumption to cancel out the $(i-2, i-1)$ elements, we get:

$$B_{(i,i-1)}^{-1} = -B_{(i-1,i)}^{-1}$$

So it was shown inductively, that the elements directly below the diagonal in B^{-1} , are equal to the negated elements directly above the diagonal. Now set $(i, j) = (3, 1)$ in Eq.(14):

$$B_{(3,1)}^{-1} = B_{(1,1)}^{-1} + B_{(2,0)}^{-1} - B_{(2,2)}^{-1} = B_{(1,1)}^{-1} - B_{(2,2)}^{-1}$$

and then using Eq.(17) for the position $(i, j) = (1, 3)$:

$$B_{(1,3)}^{-1} = B_{(0,2)}^{-1} + B_{(1,1)}^{-1} - B_{(2,2)}^{-1} = B_{(1,1)}^{-1} - B_{(2,2)}^{-1}$$

so we then have:

$$B_{(3,1)}^{-1} = B_{(1,3)}^{-1}$$

Then assuming the following holds for $i' = i - 1$:

$$B_{(i',i'-2)}^{-1} = B_{(i'-2,i')}^{-1}$$

we wish to show that it also holds for $i' = i$. Proceed by inserting $(i, j) = (i, i-2)$ in Eq.(16):

$$B_{(i,i-2)}^{-1} = B_{(i-2,i-2)}^{-1} + B_{(i-1,i-3)}^{-1} - B_{(i-1,i-1)}^{-1}$$

and setting $(i, j) = (i-2, i)$ in Eq.(14) gives us:

$$B_{(i-2,i)}^{-1} = B_{(i-3,i-1)}^{-1} + B_{(i-2,i-2)}^{-1} - B_{(i-1,i-1)}^{-1}$$

and holding these two together, we have:

$$B_{(i,i-2)}^{-1} - B_{(i-1,i-3)}^{-1} = B_{(i-2,i)}^{-1} - B_{(i-3,i-1)}^{-1}$$

but from the assumption we have that $B_{(i-1,i-3)}^{-1} = B_{(i-3,i-1)}^{-1}$, and therefore:

$$B_{(i,i-2)}^{-1} = B_{(i-2,i)}^{-1}$$

as we wanted to show.

General Symmetry of $B_n^{-1}(\beta)$

The results of the previous section suggest that the general relationship between elements in the lower and upper triangular parts of B^{-1} , are as follows:

$$B_{ij}^{-1} = (-1)^{i+j} \cdot B_{ji}^{-1} \tag{18}$$

which obviously holds for $i = j$, and was shown above, to also hold for $j = i - 1$ and $j = i - 2$. To prove it in general, we consider elements at a row- or column-wise distance of k from the diagonal, that is, we wish to prove Eq.(18) holds for $j = i - k$ or $i = j + k$, with $k \geq 0$:

$$B_{(i,i-k)}^{-1} = (-1)^k \cdot B_{(i-k,i)}^{-1} \tag{19}$$

Again, the three first cases were proved above, so we now assume that Eq.(18) holds for $j = i - k'$ for all $k' < k$, and then need to show that it also holds for $k' = k$. First, by using Eq.(17) we have that the element at position $(1 + k, 1)$ is:

$$B_{(1+k,1)}^{-1} = B_{(k-1,1)}^{-1} + B_{(k,0)}^{-1} - B_{(k,2)}^{-1} = B_{(k-1,1)}^{-1} - B_{(k,2)}^{-1}$$

and by setting $(i, j) = (1, 1 + k)$ in Eq.(17) also, we get:

$$B_{(1,1+k)}^{-1} = B_{(0,k)}^{-1} + B_{(1,k-1)}^{-1} - B_{(2,k)}^{-1} = B_{(1,k-1)}^{-1} - B_{(2,k)}^{-1}$$

From the assumption we have that:

$$B_{(k-1,1)}^{-1} = (-1)^{k-1+1} \cdot B_{(1,k-1)}^{-1} = (-1)^k \cdot B_{(1,k-1)}^{-1}$$

And note for the remaining part with $(i, j) = (k, 2)$, that $j = i - k + 2 = i - (k - 2)$, and hence using the assumption again we have:

$$B_{(k,2)}^{-1} = (-1)^{k+2} \cdot B_{(2,k)}^{-1} = (-1)^k \cdot B_{(2,k)}^{-1}$$

Using all of this, we have:

$$B_{(1+k,1)}^{-1} = (-1)^k \cdot B_{(1,1+k)}^{-1}$$

Which means that the element at the $(1 + k, 1)$ 'th position in the inverse matrix, also satisfies Eq.(18). To show this also holds for all the other elements at a row- or column-wise distance of k to the diagonal, we use *induction-within-induction*, which is a tad more intricate. That is, assume the following holds:

$$B_{(i'+k',i')}^{-1} = (-1)^{k'} \cdot B_{(i',i'+k')}^{-1}$$

for all i' when $k' < k$ (this is the *outer*-induction), and for $i' < i$ when $k' = k$ (this is the *inner*-induction, that we are dealing with now), and then show that it also holds for $i' = i$ when $k' = k$. To do this, insert $(i, j) = (i + k, i)$ into Eq.(14):

$$B_{(i+k,i)}^{-1} = B_{(i+k-2,i)}^{-1} + B_{(i-1+k,i-1)}^{-1} - B_{(i+k-1,i+1)}^{-1} \quad (20)$$

and then set $(i, j) = (i, i + k)$ also in Eq.(14):

$$B_{(i,i+k)}^{-1} = B_{(i-1,i-1+k)}^{-1} + B_{(i,i+k-2)}^{-1} - B_{(i+1,i+k-1)}^{-1} \quad (21)$$

From the assumption on the outer-induction we have:

$$B_{(i+k-2,i)}^{-1} = B_{(i+(k-2),i)}^{-1} = (-1)^{k-2} \cdot B_{(i,i+(k-2))}^{-1} = (-1)^k \cdot B_{(i,i+k-2)}^{-1}$$

And from the assumption on the inner-induction we have:

$$B_{(i-1+k,i-1)}^{-1} = (-1)^k \cdot B_{(i-1,i-1+k)}^{-1}$$

By noticing that $(i + k - 1, i + 1) = (i + 1 + (k - 2), i + 1)$, and this position is covered by the assumption for the outer-induction with $k' = k - 2$, then we have:

$$B_{(i+k-1, i+1)}^{-1} = (-1)^{(i+k-1)+(i+1)} \cdot B_{(i+1, i+k-1)}^{-1} = (-1)^k \cdot B_{(i+1, i+k-1)}^{-1}$$

So replacing these three identities back into Eq.(20), we have:

$$B_{(i+k, i)}^{-1} = (-1)^k \cdot \left(B_{(i, i+k-2)}^{-1} + B_{(i-1, i-1+k)}^{-1} - B_{(i-1, i-1+k)}^{-1} \right)$$

which can be used with Eq.(21) to yield:

$$B_{(i+k, i)}^{-1} = (-1)^k \cdot B_{(i, i+k)}^{-1}$$

as we wanted to prove. To sum up the entire proof, first Eq.(18) was shown for the diagonal and its two *sub*-diagonals. Then it was shown, that given any row-distance k to the diagonal, the element at position $(1 + k, 1)$ of the matrix $B_n^{-1}(\beta)$, also satisfies Eq.(18). And finally it was shown that for any row-number i , the element at position $(i + k, i)$ (and hence $(i - k, i)$) satisfies Eq.(18) also, and the conclusion is therefore, that all elements of $B_n^{-1}(\beta)$ satisfy Eq.(18).

Condition Number

The condition-number for a matrix A is defined as:

$$\kappa(A) = \|A\| \|A^{-1}\|$$

Since different matrix-norms are equivalent (but not equal) in regards to their magnitude, we may just as well choose the norm $\|A_n(\beta)\|$ that has the simplest form. For an n -dimensional vector x , use the l_∞ vector-norm:

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

which means the subordinate matrix-norm can be proven to be the largest sum of the absolute row-elements:

$$\|A\|_\infty = \sup \{ \|Au\| : u \in \mathbb{R}^n, \|u\|_\infty = 1 \} = \dots = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|$$

So for these tridiagonal matrix-forms, we have the simple expression:

$$\|A_n(\beta)\|_\infty = \|B_n(\beta)\|_\infty = 1 + 2|\beta|$$

regardless of the dimensionality n .

Ill- or Well-Conditioned Matrix?

If we were only interested in whether A was ill-conditioned or not, that is, whether κ was *high* or *low*, then we could use the following inequality for matrix-norms to give us an upper boundary or approximation called $\bar{\kappa}(A)$:

$$\kappa(A) = \|A\| \|A^{-1}\| = \|A\| \|U^{-1}L^{-1}\| \leq \|A\| \|U^{-1}\| \|L^{-1}\| = \bar{\kappa}(A)$$

and similarly for $B_n(\beta)$.

Computing $\|L_n^{-1}(\beta)\|_\infty$ in Linear Time & Space

The naive algorithm for calculating $\|L^{-1}\|_\infty$ is as follows:

- Set $max = 1$.
- For $i = 2$ to n , do the following:
 - Let the temporary value sum , be defined as the sum of the absolute elements in a row:

$$sum = 1 + \sum_{j=1}^{i-1} |L_{ij}^{-1}|$$

where L_{ij}^{-1} is as defined in Eq.(6).

- If $sum > max$ then $max \leftarrow sum$.

- Now we have: $\|L^{-1}\|_\infty = max$.

But looking at the form of the $L_n^{-1}(\beta)$ matrix above, one might think that its last row is the one with the highest sum, and hence the one used as the matrix-norm. Let $s_L(i)$ be the sum of the absolute elements of row i in $L_n^{-1}(\beta)$, defined as:

$$s_L(i) = 1 + |l_{i-1}| + |l_{i-1}| \cdot |l_{i-2}| + |l_{i-1}| \cdot |l_{i-2}| \cdot |l_{i-3}| + \cdots + |l_{i-1}| \cdot |l_{i-2}| \cdot |l_{i-3}| \cdots |l_1|$$

where we have used $|a \cdot b| = |a| \cdot |b|$. The expression can be simplified to:

$$s_L(i) = 1 + |l_{i-1}|(1 + |l_{i-2}|(1 + |l_{i-3}| \cdots (1 + l_1)))$$

which can be seen to be of the recursive form:

$$s_L(i) = 1 + |l_{i-1}|s_L(i-1) \tag{22}$$

with the sum for the first row in $L_n^{-1}(\beta)$ being:

$$s_L(1) = 1$$

Now we wish to show inductively, that:

$$\forall k < i : s_L(k) < s_L(i)$$

The base case $i = 2$ is easily found to be true:

$$s_L(2) = 1 + |l_1|s_L(1) = 1 + |l_1| > 1 = s_L(1)$$

Now assume the hypothesis is true for $i - 1$, and show that it also holds for i . First note that $s_L(i)$ is positive (and hence also non-zero) whenever $s_L(i - 1)$ is, we then have:

$$s_L(i) = 1 + |l_{i-1}|s_L(i-1) > s_L(i-1) \Leftrightarrow |l_{i-1}| > \frac{s_L(i-1) - 1}{s_L(i-1)} = 1 - \frac{1}{s_L(i-1)}$$

So without further knowledge about $|l_{i-1}|$ – which is complicated to deduce due to the recursive form of Eq.(3) – we can not generally assume that this inequality holds, and are therefore forced to compute and compare all $s_L(i)$'s to find the largest one, which is then $\|L_n^{-1}(\beta)\|$.

However, Eq.(22) does give us an easy formula for accumulating the temporary row-sum, and thus computing $\|L_n^{-1}(\beta)\|_\infty$ in linear time, with the algorithm being:

- Set $max = sum = 1$, where sum is the accumulated sum.
- For $i = 2$ up to n , do the following:

– Update the accumulated sum:

$$sum \leftarrow 1 + |l_{i-1}|sum$$

– If $sum > max$ then $max \leftarrow sum$.

- Now we have: $\|L_n^{-1}(\beta)\|_\infty = max$.

Note that we only require the vector l for this algorithm, and therefore do not need to construct the actual $L_n^{-1}(\beta)$ matrix at all, so the matrix norm $\|L_n^{-1}(\beta)\|_\infty$ is computable in linear space as well as linear time. In fact, we could compute $s_L(i)$ after having computed the i 'th element of the l -vector, that is l_i , and thereby save the storage for l to obtain an algorithm having linear time but constant space usage.

Computing $\|U_n^{-1}(\beta)\|_\infty$ in Linear Time & Space

From Eq.(7) we have that the sum $s_U(i)$ of the absolute elements of the i 'th row in $U_n^{-1}(\beta)$ is:

$$s_U(i) = \sum_{k=i}^n \left| \beta^{k-i} \prod_{m=i}^k d_m^{-1} \right|$$

Which can be shown inductively, to have the recursive form:

$$s_U(i) = |d_i^{-1}| \cdot (1 + |\beta|) \cdot s_U(i + 1)$$

where it should be noted that $s_U(i)$ is defined from $s_U(i + 1)$ instead of from $s_U(i - 1)$, with the base case then being:

$$s_U(n) = |d_n^{-1}|$$

So the algorithm for calculating $\|U_n^{-1}(\beta)\|_\infty$ is:

- Set $max = sum = |d_n^{-1}|$, where sum is the accumulated sum.
- For $i = n - 1$ down to 1, do the following:
 - Update the accumulated sum:
$$sum \leftarrow |d_i^{-1}| \cdot (1 + |\beta|) \cdot sum$$
 - If $sum > max$ then $max \leftarrow sum$.
- Now we have: $\|U_n^{-1}(\beta)\|_\infty = max$.

Obviously this algorithm uses $O(n)$ time, and since we only require data storage for the vector d , the algorithm also only consumes linear space. It is not possible to interleave the computation of each $s_U(i)$ after each of l_i and d_i have been computed, because the above summation algorithm traverses d backwards from element n down to 1, as opposed to the previous summation algorithm for computing $\|L_n^{-1}(\beta)\|_\infty$ from $s_L(i)$. We can however limit the storage to that of the d -vector.

Computing $\|A_n^{-1}(\beta)\|_\infty$ in Linear Time & Space

The upper boundary $\bar{\kappa}$ on the condition number, along with the above algorithms for computing $\|L_n^{-1}(\beta)\|_\infty$ and $\|U_n^{-1}(\beta)\|_\infty$, were developed before the below algorithm for computing $\|A_n^{-1}(\beta)\|_\infty$ in linear time and space was discovered.

First define $t(i)$ as the sum of the absolute elements from the diagonal and to its right, that is:

$$t(i) = \sum_{k=i}^n |A_{ik}^{-1}|$$

The sequence of $t(i)$'s for $i \in \{1, \dots, n\}$, can be computed by first having:

$$t(n) = |A_{nn}^{-1}|$$

and then computing the rest of the $t(i)$'s starting with $i = n - 1$ as follows:

$$t(i) = |A_{ii}^{-1}| + t(i + 1) \cdot \left| \frac{\beta}{d_i} \right|$$

where Eq.(9) was used to incorporate $t(i+1)$ into the calculation of $t(i)$. Now using that $A_n^{-1}(\beta)$ is both symmetric and cross-symmetric as described by Eq.(13), we have that the actual sum $s(i)$ of row i in $A_n^{-1}(\beta)$ is given by:

$$s(i) = t(i) + t(n - i + 1) - |A_{ii}^{-1}| \tag{23}$$

which means that we may limit our attention to finding the maximum $s(i)$ for half of the rows:

$$\|A_n^{-1}(\beta)\|_\infty = \max_{i \in \{1, \dots, \lceil n/2 \rceil\}} s(i)$$

So the algorithm is first to compute the l - and d -vectors, then the diagonal of $A_n^{-1}(\beta)$, and from these the temporary sum $t(i)$ for each row i , and finally the sums $s(i)$ for half of the rows, and taking the maximum one of them. Hence, the algorithm uses both linear time and space.

Computing $\|B_n^{-1}(\beta)\|_\infty$ in Linear Time & Space

Since $B_n^{-1}(\beta)$ is cross-symmetric, but also – with the exception of the sign of the elements – symmetric in regards to regular transposition, then its infinity norm can be found in the same way as for $A_n^{-1}(\beta)$.

Non-Iterative Computation

If non-recursive expressions were known for Eqs.(3) and (4), then it is plausible that there would be explicit expressions for $\|L_n^{-1}(\beta)\|$ and $\|U_n^{-1}(\beta)\|$, so that iterative algorithms would not be needed in the computation of the upper boundary $\bar{\kappa}(A_n(\beta))$.

Based on non-recursive expressions for Eqs.(3) and (4), there could even exist simple expressions for calculating $\|A_n^{-1}(\beta)\|_\infty$ directly, without using any algorithmic iterations either.

Question 1

Now compute the LU -factorization of $A_5(1/4)$. First we have $l_1 = 1/4$, and then Eq.(3) yields:

$$l_2 = \frac{\alpha}{1 - l_1\beta} = \frac{1/4}{1 - 1/4 \cdot 1/4} = \dots = \frac{1/4}{15/16} = \frac{4}{15}$$

And then:

$$l_3 = \frac{1/4}{1 - 4/15 \cdot 1/4} = \dots = \frac{1/4}{14/15} = \frac{15}{56}$$

And finally:

$$l_4 = \frac{1/4}{1 - 15/56 \cdot 1/4} = \dots = \frac{1/4}{209/224} = \frac{56}{209}$$

For the upper triangular matrix, we first have $d_1 = 1$, and from Eq.(4), we find that: $d_2 = 15/16$, $d_3 = 14/15$, $d_4 = 209/224$, and finally:

$$d_5 = 1 - l_4 \cdot 1/4 = 1 - 56/209 \cdot 1/4 = 1 - 14/209 = 195/209$$

We therefore have:

$$L_5(1/4) = \begin{bmatrix} 1 & & & & \\ 1/4 & 1 & & & \\ & 4/15 & 1 & & \\ & & 15/56 & 1 & \\ & & & 56/209 & 1 \end{bmatrix}$$

and:

$$U_5(1/4) = \begin{bmatrix} 1 & & & & \\ & 1/4 & & & \\ & 15/16 & 1/4 & & \\ & & 14/15 & 1/4 & \\ & & & 209/224 & 1/4 \\ & & & & 195/209 \end{bmatrix}$$

Which has been verified by hand, to result in $L_5(1/4)U_5(1/4) = A_5(1/4)$.

Question 2

The last diagonal element of the inverse is given by:

$$A_{55}^{-1} = d_5^{-1} = 209/195$$

and using Eq.(10), we find the rest of the diagonal:

$$A_{44}^{-1} = (1 + \beta \cdot l_4 \cdot A_{55}^{-1}) d_4^{-1} = (1 + 1/4 \cdot 56/209 \cdot 209/195) 224/209 = 224/195$$

$$A_{33}^{-1} = (1 + 1/4 \cdot 15/56 \cdot 224/195) 15/14 = 15/13$$

$$A_{22}^{-1} = (1 + 1/4 \cdot 4/15 \cdot 15/13) 16/15 = 224/195$$

$$A_{11}^{-1} = (1 + 1/4 \cdot 1/4 \cdot 224/195) = 209/195$$

Where it should be noted that we could just as well have used the cross-symmetric property, immediately resulting in $A_{11}^{-1} = A_{55}^{-1}$ and $A_{22}^{-1} = A_{44}^{-1}$. Now, from Eq.(9) we have the elements $i \in \{1, \dots, 4\}$ above the diagonal for column $j = 5$:

$$A_{45}^{-1} = -\beta \cdot d_4^{-1} \cdot A_{55}^{-1} = -\frac{1}{4} \cdot \frac{224}{209} \cdot \frac{209}{195} = -\frac{56}{195}$$

$$A_{35}^{-1} = -\frac{1}{4} \cdot \frac{15}{14} \cdot \left(-\frac{56}{195}\right) = \frac{1}{13}$$

$$A_{25}^{-1} = -\frac{1}{4} \cdot \frac{16}{15} \cdot \frac{1}{13} = -\frac{4}{195}$$

$$A_{15}^{-1} = -\frac{1}{4} \cdot 1 \cdot \left(-\frac{4}{195}\right) = \frac{1}{195}$$

Again we could use the cross-symmetry of the inverse matrix, but instead we calculate the entire upper triangular matrix. So for column $j = 4$ we have:

$$A_{34}^{-1} = -\beta \cdot d_3^{-1} \cdot A_{44}^{-1} = -\frac{1}{4} \cdot \frac{15}{14} \cdot \frac{224}{195} = -\frac{4}{13}$$

$$A_{24}^{-1} = -\frac{1}{4} \cdot \frac{16}{15} \cdot \left(-\frac{4}{13}\right) = \frac{16}{195}$$

$$A_{14}^{-1} = -\frac{1}{4} \cdot 1 \cdot \frac{16}{195} = -\frac{4}{195}$$

And for column $j = 3$ we have:

$$A_{23}^{-1} = -\beta \cdot d_2^{-1} \cdot A_{33}^{-1} = -\frac{1}{4} \cdot \frac{16}{15} \cdot \frac{15}{13} = -\frac{4}{13}$$

$$A_{13}^{-1} = -\frac{1}{4} \cdot 1 \cdot \left(-\frac{4}{13}\right) = \frac{1}{13}$$

And for column $j = 2$ we have:

$$A_{12}^{-1} = -\beta \cdot d_1^{-1} \cdot A_{22}^{-1} = -\frac{1}{4} \cdot 1 \cdot \frac{224}{195} = -\frac{56}{195}$$

And then transposing all of this to the lower triangle of A^{-1} also, we find that:

$$A_5^{-1}(1/4) = \begin{bmatrix} \frac{209}{195} & -\frac{56}{195} & \frac{1}{13} & -\frac{4}{195} & \frac{1}{195} \\ -\frac{56}{195} & \frac{224}{195} & -\frac{13}{15} & \frac{16}{195} & -\frac{1}{195} \\ \frac{1}{195} & -\frac{13}{195} & \frac{13}{15} & -\frac{4}{195} & \frac{1}{195} \\ -\frac{13}{195} & \frac{16}{195} & -\frac{13}{15} & \frac{13}{224} & -\frac{13}{56} \\ -\frac{1}{195} & -\frac{4}{195} & \frac{1}{13} & -\frac{16}{195} & \frac{209}{195} \end{bmatrix}$$

In preparation of using Eq.(23) to calculate $\|A_5^{-1}(1/4)\|_\infty$, we have that the temporary sums of the absolute row-elements (i.e. the diagonal and the elements to their right), are as follows:

$$t(5) = |A_{55}^{-1}| = \frac{209}{195}$$

$$t(4) = |A_{44}^{-1}| + t(5) \cdot \left|\frac{\beta}{d_4}\right| = \frac{224}{195} + \frac{209}{195} \cdot \left|\frac{1/4}{209/224}\right| = \frac{56}{39}$$

$$t(3) = \frac{15}{13} + \frac{56}{39} \cdot \left|\frac{1/4}{14/15}\right| = \frac{20}{13}$$

$$t(2) = \frac{224}{195} + \frac{56}{39} \cdot \left|\frac{1/4}{15/16}\right| = \frac{304}{195}$$

$$t(1) = \frac{209}{195} + \frac{304}{195} \cdot \left|\frac{1/4}{1}\right| = \frac{285}{195}$$

And then using Eq.(23) we have that the actual row-sums are:

$$s(1) = t(1) = \frac{285}{195}$$

$$s(2) = t(2) + t(4) - |A_{22}^{-1}| = \frac{360}{195}$$

$$s(3) = t(3) + t(3) - |A_{33}^{-1}| = \frac{25}{13} = \frac{375}{195}$$

So we have:

$$\|A_n^{-1}(\beta)\|_\infty = \max_{i \in \{1,2,3\}} s(i) = \frac{375}{195}$$

And therefore:

$$\kappa(A_5(1/4)) = \|A_n(\beta)\|_\infty \cdot \|A_n^{-1}(\beta)\|_\infty = (1 + 2 \cdot |\beta|) \cdot \frac{375}{195} = \frac{3}{2} \cdot \frac{375}{195} = \frac{75}{26}$$

which approximately equals 2.8846.

Question 3.1

Now compute the LU -factorization of $B_5(1/4)$. First note that although the formulae are very similar to those of factorizing $A_5(1/4)$, the numerator in computing l_i from Eq.(3) is now negative, which propagates through the rest of the l_k 's for $k > i$. We therefore have:

$$l_1 = -1/4$$

$$l_2 = -\frac{1/4}{1 + 1/4 \cdot 1/4} = \dots = -\frac{1/4}{17/16} = -\frac{4}{17}$$

$$l_3 = -\frac{1/4}{1 + 4/17 \cdot 1/4} = \dots = -\frac{1/4}{18/17} = -\frac{17}{72}$$

$$l_4 = -\frac{1/4}{1 + 17/72 \cdot 1/4} = \dots = -\frac{1/4}{305/288} = -\frac{72}{305}$$

This gives us the following diagonal of the upper triangular matrix:

$$d_1 = 1$$

$$d_2 = 17/16$$

$$d_3 = 18/17$$

$$d_4 = 305/288$$

And finally we have:

$$d_5 = 1 + 72/305 \cdot 1/4 = 1 + 18/305 = 323/305$$

This means that for $B_5(1/4)$ we have:

$$L_5(1/4) = \begin{bmatrix} 1 & & & & \\ -1/4 & 1 & & & \\ & -4/17 & 1 & & \\ & & -17/72 & 1 & \\ & & & -72/305 & 1 \end{bmatrix}$$

and:

$$U_5(1/4) = \begin{bmatrix} 1 & 1/4 & & & \\ & 17/16 & 1/4 & & \\ & & 18/17 & 1/4 & \\ & & & 305/288 & 1/4 \\ & & & & 323/305 \end{bmatrix}$$

And it has once more been checked by hand, that the above factoring indeed yields $L_5(1/4)U_5(1/4) = B_5(1/4)$.

Question 3.2

The upper triangle of $B_5^{-1}(1/4)$ is found similarly to that of $A_5^{-1}(1/4)$, starting with the last diagonal element of the inverse:

$$B_{55}^{-1} = d_5^{-1} = 305/323$$

and then using Eq.(10), we find the rest of the diagonal:

$$B_{44}^{-1} = (1 + \beta \cdot l_4 \cdot B_{55}^{-1}) d_4^{-1} = (1 + 1/4 \cdot (-72/305) \cdot 305/323) 288/305 = 288/323$$

$$B_{33}^{-1} = (1 + 1/4 \cdot (-17/72) \cdot 288/323) 17/18 = 289/323$$

Because of cross-symmetry we stop calculating the diagonal here, and use Eq.(9) to find the elements above the diagonal for row $i \in \{1, \dots, 4\}$ and column $j = 5$:

$$B_{45}^{-1} = -\beta \cdot d_4^{-1} \cdot B_{55}^{-1} = -\frac{1}{4} \cdot \frac{288}{305} \cdot \frac{305}{323} = -\frac{72}{323}$$

$$B_{35}^{-1} = -\frac{1}{4} \cdot \frac{17}{18} \cdot \left(-\frac{72}{323}\right) = \frac{1}{19}$$

$$B_{25}^{-1} = -\frac{1}{4} \cdot \frac{16}{17} \cdot \frac{1}{19} = -\frac{4}{323}$$

$$B_{15}^{-1} = -\frac{1}{4} \cdot 1 \cdot \left(-\frac{4}{323}\right) = \frac{1}{323}$$

Again because of cross-symmetry we only need to calculate B_{i4}^{-1} for $i \in \{2, 3\}$ as $B_{14}^{-1} = B_{25}^{-1}$, so we have:

$$B_{34}^{-1} = -\beta \cdot d_3^{-1} \cdot B_{44}^{-1} = -\frac{1}{4} \cdot \frac{17}{18} \cdot \frac{288}{323} = -\frac{4}{19}$$

$$B_{24}^{-1} = -\frac{1}{4} \cdot \frac{16}{17} \cdot \left(-\frac{4}{19}\right) = \frac{16}{323}$$

The following part of the inverse matrix has now been calculated:

$$\begin{bmatrix} & & \frac{1}{323} \\ & \frac{16}{323} & -\frac{323}{4} \\ \frac{289}{323} & -\frac{19}{288} & \frac{19}{72} \\ & \frac{323}{323} & -\frac{323}{305} \\ & & \frac{305}{323} \end{bmatrix}$$

Copying elements according to Eq.(13), we have:

$$\begin{bmatrix} \frac{305}{323} & -\frac{72}{323} & \frac{1}{323} & -\frac{4}{323} & \frac{1}{323} \\ & \frac{323}{288} & -\frac{19}{4} & \frac{323}{16} & -\frac{323}{4} \\ & \frac{323}{289} & \frac{19}{323} & -\frac{323}{4} & \frac{323}{1} \\ & & \frac{19}{288} & -\frac{19}{72} & -\frac{323}{305} \\ & & & \frac{323}{323} & \frac{305}{323} \end{bmatrix}$$

And finally from Eq.(18) we get:

$$B_5^{-1}(1/4) = \begin{bmatrix} \frac{305}{323} & -\frac{72}{323} & \frac{1}{323} & -\frac{4}{323} & \frac{1}{323} \\ \frac{323}{72} & \frac{323}{288} & -\frac{19}{4} & \frac{323}{16} & -\frac{323}{4} \\ \frac{323}{1} & \frac{323}{4} & \frac{289}{323} & -\frac{323}{4} & \frac{323}{1} \\ \frac{19}{4} & \frac{19}{16} & \frac{323}{4} & -\frac{19}{288} & -\frac{19}{72} \\ \frac{323}{1} & \frac{323}{4} & \frac{19}{323} & \frac{323}{72} & \frac{323}{305} \\ \frac{323}{323} & \frac{323}{323} & \frac{19}{323} & \frac{323}{323} & \frac{323}{323} \end{bmatrix}$$

Again in preparation of using Eq.(23) to calculate $\|B_5^{-1}(1/4)\|_\infty$, we have:

$$\begin{aligned} t(5) &= |B_{55}^{-1}| = \frac{305}{323} \\ t(4) &= |B_{44}^{-1}| + t(5) \cdot \left| \frac{\beta}{d_4} \right| = \frac{288}{323} + \frac{305}{323} \cdot \left| \frac{1/4}{305/288} \right| = \frac{360}{323} \\ t(3) &= \frac{289}{323} + \frac{360}{323} \cdot \left| \frac{1/4}{18/17} \right| = \frac{374}{323} \\ t(2) &= \frac{288}{323} + \frac{374}{323} \cdot \left| \frac{1/4}{17/16} \right| = \frac{376}{323} \\ t(1) &= \frac{305}{323} + \frac{376}{323} \cdot \left| \frac{1/4}{1} \right| = \frac{399}{323} \end{aligned}$$

And then using Eq.(23) we have that the actual row-sums are:

$$s(1) = t(1) = \frac{399}{323}$$

$$s(2) = t(2) + t(4) - |B_{22}^{-1}| = \frac{348}{323}$$

$$s(3) = t(3) + t(3) - |B_{33}^{-1}| = \frac{459}{323}$$

So we have:

$$\|B_n^{-1}(\beta)\|_\infty = \max_{i \in \{1,2,3\}} s(i) = \frac{459}{323}$$

And therefore:

$$\kappa(B_5(1/4)) = \|B_n(\beta)\|_\infty \cdot \|B_n^{-1}(\beta)\|_\infty = (1 + 2 \cdot |\beta|) \cdot \frac{459}{323} = \frac{3}{2} \cdot \frac{459}{323} = \frac{81}{38}$$

which approximately equals 2.1316.

Question 4

The task is to compute the LU -factorization of $A_{10}(1.2) = A_{10}(6/5)$. We first have that $l_1 = 6/5$, and then:

$$l_2 = \frac{6/5}{1 - 6/5 \cdot 6/5} = \dots = \frac{6/5}{-11/25} = -\frac{30}{11}$$

$$l_3 = \frac{6/5}{1 + 30/11 \cdot 6/5} = \dots = \frac{6/5}{47/11} = \frac{66}{235}$$

$$l_4 = \frac{6/5}{1 - 66/235 \cdot 6/5} = \dots = \frac{6/5}{779/1175} = \frac{1410}{779}$$

$$l_5 = \frac{6/5}{1 - 1410/779 \cdot 6/5} = \dots = \frac{6/5}{-913/779} = -\frac{4674}{4565}$$

$$l_6 = \frac{6/5}{1 + 4674/4565 \cdot 6/5} = \dots = \frac{6/5}{50869/22825} = \frac{27390}{50869}$$

$$l_7 = \frac{6/5}{1 - 27390/50869 \cdot 6/5} = \dots = \frac{6/5}{18001/50869} = \frac{305214}{90005}$$

$$l_8 = \frac{6/5}{1 - 305214/90005 \cdot 6/5} = \dots = \frac{6/5}{-1381259/450025} = -\frac{540030}{1381259}$$

$$l_9 = \frac{6/5}{1 + 540030/1381259 \cdot 6/5} = \dots = \frac{6/5}{2029295/1381259} = \frac{8287554}{10146475}$$

And for the upper triangular matrix, we first have $d_1 = 1$, and then: $d_2 = -11/25$, $d_3 = 47/11$, $d_4 = 779/1175$, $d_5 = -913/779$, $d_6 = 50869/22825$, $d_7 = 18001/50869$, $d_8 = -1381259/450025$, $d_9 = 2029295/1381259$, and finally:

$$d_{10} = 1 - 8287554/10146475 \cdot 6/5 = 1007051/50732375$$

and the following for U :

$$\begin{bmatrix} 1.2 & 1 & 1.2 & & & & & & & & \\ & 1.2 & 1 & 1.2 & & & & & & & \\ & & -1.3056 & -0.3667 & & & & & & & \\ & & & 1.2 & 1 & 1.2 & & & & & \\ & & & & 1.2 & 1 & 1.2 & & & & \\ & & & & & -1.2026 & -0.6475 & & & & \\ & & & & & & 1.2 & 1 & 1.2 & & \\ & & & & & & & 1.2 & 1 & 1.2 & \\ & & & & & & & & 1.2 & 1 & \\ & & & & & & & & & & 0.0183 \end{bmatrix}$$

and finally the permutation matrix:

$$P = \begin{bmatrix} 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \end{bmatrix}$$

with:

$$PA = LU$$

where the matrices are computed with a partial pivoting threshold of 1 by default.

Counting on P , it clearly takes 7 row-interchange operations to transform P into the identity matrix I .

Question 6

If we instead use a pivoting threshold of 0, then there is diagonal (i.e. no) pivoting. First we need to convert the matrix A in MatLab, to be of the sparse-matrix datatype:

```
S = sparse(A)
(...)
[L,U,P] = lu(S,0)
```


$$\begin{aligned}
l_7 &= -\frac{6/5}{1 + 243390/369181 \cdot 6/5} = \dots = -\frac{6/5}{661249/369181} = -\frac{2215086}{3306245} \\
l_8 &= -\frac{6/5}{1 + 2215086/3306245 \cdot 6/5} = \dots = -\frac{6/5}{29821741/16531225} = -\frac{19837470}{29821741} \\
l_9 &= -\frac{6/5}{1 + 19837470/29821741 \cdot 6/5} = \dots = -\frac{6/5}{53626705/29821741} = -\frac{178930446}{268133525}
\end{aligned}$$

So $d_1 = 1$, and $d_2 = 61/25$, $d_3 = 97/61$, $d_4 = 4621/2425$, $d_5 = 8113/4621$, $d_6 = 369181/202825$, $d_7 = 661249/369181$, $d_8 = 29821741/16531225$, $d_9 = 53626705/29821741$, and finally:

$$d_{10} = 1 + 178930446/268133525 \cdot 6/5 = 2414250301/1340667625$$

Again, these fractions reduce to moderate numbers, e.g. $d_{10} \simeq 1.8001$. Using the above numbers, we find that the lower triangular matrix $L_{10}(6/5)$ for $B_{10}(6/5)$ is:

$$\begin{bmatrix}
1 & & & & & & & & & & \\
-\frac{6}{5} & 1 & & & & & & & & & \\
& -\frac{30}{61} & 1 & & & & & & & & \\
& & -\frac{366}{485} & 1 & & & & & & & \\
& & & -\frac{2910}{4621} & 1 & & & & & & \\
& & & & -\frac{27726}{40565} & 1 & & & & & \\
& & & & & -\frac{243390}{369181} & 1 & & & & \\
& & & & & & -\frac{2215086}{3306245} & 1 & & & \\
& & & & & & & -\frac{19837470}{29821741} & 1 & & \\
& & & & & & & & -\frac{178930446}{268133525} & 1 & \\
& & & & & & & & & & 1
\end{bmatrix}$$

And the upper triangular matrix $U_{10}(6/5)$ for $B_{10}(6/5)$ is:

$$\begin{bmatrix}
1 & & & & & & & & & & \\
& \frac{61}{25} & & & & & & & & & \\
& & \frac{97}{61} & & & & & & & & \\
& & & \frac{4621}{2425} & & & & & & & \\
& & & & \frac{8113}{4621} & & & & & & \\
& & & & & \frac{369181}{202825} & & & & & \\
& & & & & & \frac{661249}{369181} & & & & \\
& & & & & & & \frac{29821741}{16531225} & & & \\
& & & & & & & & \frac{53626705}{29821741} & & \\
& & & & & & & & & \frac{2414250301}{1340667625} &
\end{bmatrix}$$

Again, using Matlab it has been checked that $L_{10}(6/5)U_{10}(6/5) = B_{10}(6/5)$.

Question 7.3

Trying diagonal pivoting for $B_{10}(6/5)$, results in the following for L :

$$\begin{bmatrix} 1 & & & & & & & & & & \\ -1.2 & 1 & & & & & & & & & \\ & -0.4918 & 1 & & & & & & & & \\ & & -0.7546 & 1 & & & & & & & \\ & & & -0.6297 & 1 & & & & & & \\ & & & & -0.6835 & 1 & & & & & \\ & & & & & -0.6593 & 1 & & & & \\ & & & & & & -0.6700 & 1 & & & \\ & & & & & & & -0.6652 & 1 & & \\ & & & & & & & & -0.6673 & 1 & \\ & & & & & & & & & & 1 \end{bmatrix}$$

and the following for U :

$$\begin{bmatrix} 1 & 1.2 & & & & & & & & & \\ & 2.4400 & 1.2 & & & & & & & & \\ & & 1.5902 & 1.2 & & & & & & & \\ & & & 1.9056 & 1.2 & & & & & & \\ & & & & 1.7557 & 1.2 & & & & & \\ & & & & & 1.8202 & 1.2 & & & & \\ & & & & & & 1.7911 & 1.2 & & & \\ & & & & & & & 1.8040 & 1.2 & & \\ & & & & & & & & 1.7982 & 1.2 & \\ & & & & & & & & & 1.8008 & \\ & & & & & & & & & & 1 \end{bmatrix}$$

both of which also equal the factorization performed by hand, with the specialized algorithms.

Question 8

A diagram has to be made, that shows the reciprocal condition number for $A_{10}(\beta)$ and $B_{10}(\beta)$ for β going from 0.01 up to 2.0, in incrementing steps of 0.01.

Algorithm Implementation

The above $O(n)$ algorithm for computing the condition number κ , has been implemented in *Microsoft Visual .NET C++* and compiles to an *MS Windows* executable. The algorithm is implemented in a so-called *template class* which allows us to change the datatype for the matrix-elements, without changing the entire implementation. It is declared as follows:

```
template <class T> class LMatrixTD { ... }
```

with T being the abstract datatype used for the matrix-elements. The member-fields of the class are then:

```

const T    kBeta;           // Value of upper-diagonal elements.
const T    kAlpha;         // Value of lower-diagonal elements.
const int  kN;             // Dimensionality of the matrix.
const bool kNegativeLower; // Whether kAlpha = -kBeta or kBeta.

T         *mL;             // The l-vector.
T         *mD;             // The d-vector.
T         *mInvDiagonal;   // The diagonal of the inverse matrix.
T         *mRowSumAbs;     // The temporary row-sums.

```

The function for computing the LU -factorization is:

```

void LU()
{
    mL[0] = kAlpha;
    mD[0] = 1;

    for (int i=1; i<kN-1; i++)
    {
        mD[i] = 1 - mL[i-1] * kBeta;
        mL[i] = kAlpha / mD[i];
    }

    mD[kN-1] = 1 - mL[kN-2] * kBeta;
}

```

And the function for computing the diagonal of the inverse matrix is:

```

void InverseDiagonal()
{
    // Create the diagonal
    mInvDiagonal[kN-1] = 1/mD[kN-1];

    for (int i=kN-2; i>=0; i--)
    {
        mInvDiagonal[i] = (1 + kBeta * mL[i] * mInvDiagonal[i+1])
                          / mD[i];
    }
}

```

And these are then used in the function for computing the infinity-norm of the inverse matrix:

```

T InverseNorm()
{
    // First compute the LU-factorization.
    LU();
}

```

```

// Then compute the inverse's diagonal.
InverseDiagonal();

int i;

mRowSumAbs[kN-1] = std::fabs(mInvDiagonal[kN-1]);

// Calculate the row sums from the diagonal and elements
// to the right.
for (i=kN-2; i>=0; i--)
{
    mRowSumAbs[i] = std::fabs(mInvDiagonal[i])
        + mRowSumAbs[i+1]
        * std::fabs(kBeta / mD[i]);
}

T max = mRowSumAbs[0];

for (i=1; i<std::ceil((double)kN/2); i++)
{
    // Note the index-change because C++ indexes from 0 to n-1
    int otherIndex = kN-i-1;
    T totalRowSum = mRowSumAbs[i] + mRowSumAbs[otherIndex]
        - std::fabs(mInvDiagonal[i]);

    if (totalRowSum>max)
        max = totalRowSum;
}

return max;
}

```

The function for computing the infinity-norm of the matrix itself, is of course:

```
T Norm() { return 2*std::fabs(kBeta) + 1; }
```

Main Program

In the file `na_exam1.cpp` that holds the entry-point for the program, the following is also defined:

```
typedef double FTYPE;
```

which means that we use double-precision floating point numbers in the object-instantiations below. First however, we need to know whether to use a matrix of type $A_n(\beta)$ or $B_n(\beta)$, that is, whether the lower-diagonal should be negative or positive, this is done in the following:


```

int input;

std::cout << "Use negative lower part (0 for positive,
             1 for negative): ";
std::cin >> input;

bool negativeLower = (input != 0);

The loop for calculating the condition number with incrementing  $\beta$  is then:

int i;
for (i=0; i<2.0/0.01; i++)
{
    FTYPE beta = (i+1)*0.01;

    if (beta != 1)
    {
        NA::LMatrixTD<FTYPE> matrix(10, beta, negativeLower);

        std::cout << beta << " " << 1/matrix.Cond() << std::endl;
    }
}

```

where it should be noted that the `LMatrixTD`-class could easily have been implemented in a manner where only β was changed, thus alleviating the need to construct and destruct the object on each iteration of the loop. Furthermore, this way of incrementing an integer counter and then calculating β from that, ensures that the floating-point number for β does not drift due to rounding errors, as it would if it was accumulated as follows:

```

for (FTYPE beta=0.01; beta<=2.0; beta+=0.01)
{
    ...
}

```

in which case the condition `(beta != 1)` would always be true, and the loop would not execute for $\beta \simeq 2$, again because rounding errors would cause the variable `beta` to go from approximately 1.9 to slightly more than 2.0.

The results are shown in figures 1 and 2, from which it can be seen that $1/\kappa(A_{10}(\beta))$ is more erratic and also lower than it is for $B_{10}(\beta)$, which means that $A_{10}(\beta)$ has a greater tendency to be ill-conditioned, whereas the condition of $B_{10}(\beta)$ degrades more gracefully with increasing β .

Question 9

Another diagram is to be made, this time of the reciprocal condition number for $A_n(1.2)$ and $B_n(1.2)$ with n going from 5 up to 20 in incrementing steps of 1. The solution is similar to that for question 8, only the loop is now:

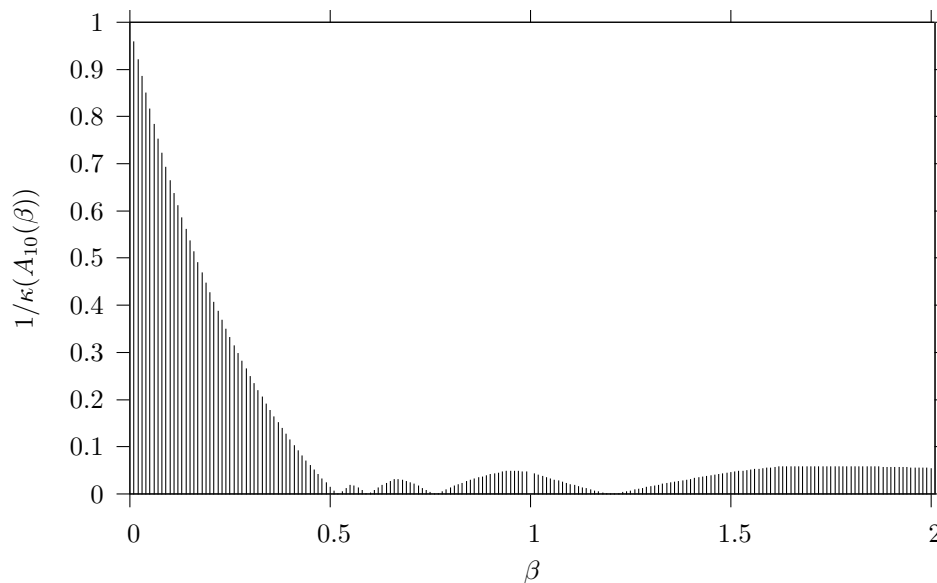


Figure 1: The reciprocal condition number for $A_{10}(\beta)$.

```

for (int i=5; i<=20; i++)
{
    NA::LMatrixTD<FTYPE> matrix(i, 1.2, negativeLower);

    std::cout << i << " " << 1/matrix.Cond() << std::endl;
}

```

It would be possible to develop specialized formulae for mathematically adjusting the results of one matrix to those of the same matrix with one more dimension. But since the algorithms are so efficient, there is no present need to develop such identities.

The results are shown in figures 3 and 4, which clearly have the same tendencies as figures 1 and 2, namely that $B_n(1.2)$ is better conditioned and more predictable than $A_n(1.2)$ with varying n .

Conclusion

It was shown that significant reductions were possible in the time- and space-complexities of certain matrix operations for these specific kinds of matrices. Although the algorithms were developed from scratch, they are believed to have been previously discovered elsewhere, as the computational field of linear algebra is widely studied.

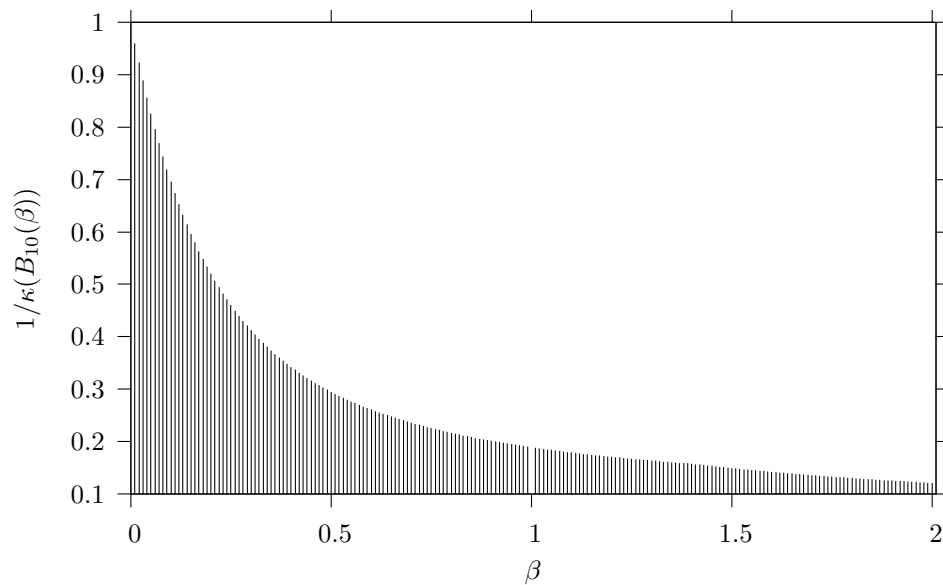


Figure 2: The reciprocal condition number for $B_{10}(\beta)$.

However, it is also believed that if explicit expressions could be found instead of the recursive Eq.(3) (and hence also Eq.(4)), then further reductions of the time complexity could be achieved, possibly even obtaining $O(1)$ algorithms (i.e. mathematical expressions) for computing the matrix-norm, condition number κ , elements of the inverse, as well as the LU -factorization.

References

- [1] David Kincaid and Ward Cheney: Numerical Analysis, Third Edition, Brooks/Cole, 2002, ISBN 0-534-38905-8

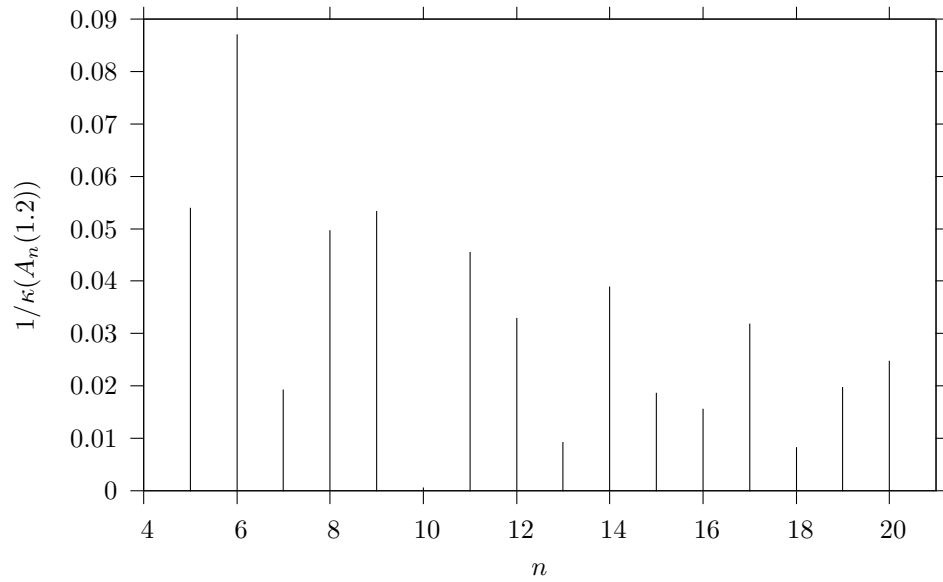


Figure 3: The reciprocal condition number for $A_n(1.2)$.

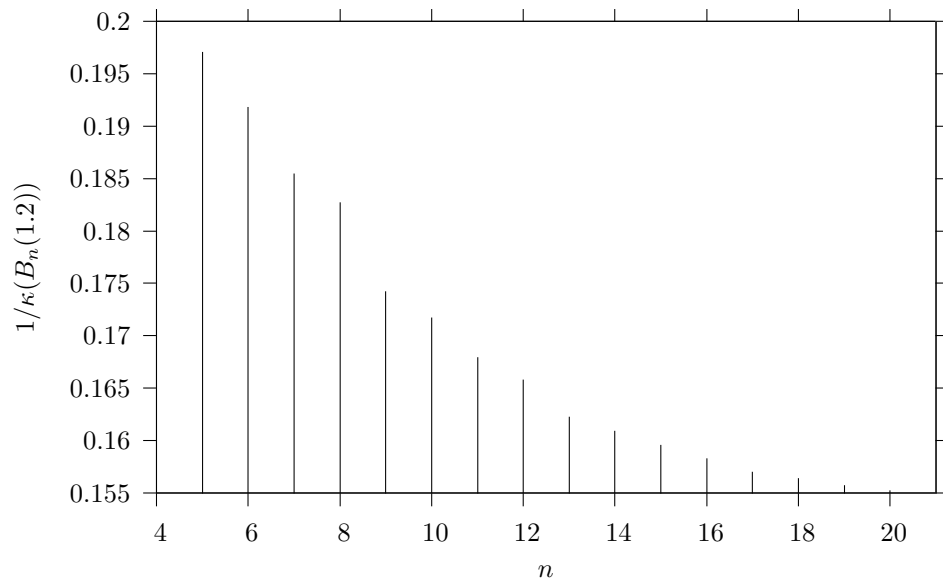


Figure 4: The reciprocal condition number for $B_n(1.2)$.