Matrix Inversion and Condition Estimation with Triangular Factors

Jurjen Duintjer Tebbens

joint work with

Miroslav Tůma

Institute of Computer Science

Academy of Sciences of the Czech Republic

Joint French-Czech Workshop on Krylov Methods for Inverse Problems,

Prague, July 19, 2010.

Outline

1. Methods of matrix inversion based on triangular factorization
2. Balanced Incomplete Factorization
3. Condition estimation with balanced factorization

1. Matrix Inversion

The main consensus is that explicit computation of the inverse of a matrix should be avoided whenever possible. For example, to solve a linear system

$$Ax = b,$$

it is very inefficient to search for the matrix A^{-1} if only the vector $A^{-1}b$ is needed.

However, in particular applications including image reconstruction or signal processing it may not be possible to circumvent the explicit computation of the inverse.

In the popular LAPACK and Matlab software, matrix inversion is done with the help of a triangular decomposition and in the following we will restrict ourselves to this strategy.

1. Matrix Inversion

Consider the Cholesky decomposition

$$A = LL^T$$

for a symmetric positive definite matrix or the LU decomposition

$$PA = LU$$

for general matrices, where L is unit lower triangular, U is upper triangular and P is a permutation matrix.

A survey of properties of methods for matrix inversion based on triangular decompositions is given in [Du Croz, Higham - 1992]. In the following we summarize the main results on implementation and stability. We start with the inversion of a unit lower triangular matrix L.

1. Matrix Inversion

Like in Gaussian elimination, the computation can be organized in several ways, according to the ordering of the involved loops:

1. Compute L^{-1} one column at a time
2. Compute L^{-1} one row at a time
3. Compute L^{-1} by outer product updates

An example of 1. is the next algorithm („Column-wise Inversion I")

for $j = 1 : n$

$X(j,j) = L(j,j)^{-1}$

$X(j+1:n, j) = -X(j,j) * L(j+1:n, j)$

$L(j+1:n, j+1:n)X(j+1:n, j) = X(j+1:n, j)$ (forward substitution)

end

1. Matrix Inversion

X is the computed inverse of L. Computational costs are dominated by the n forward solves.

The algorithm can be rearranged to avoid the forward solves („Column-wise Inversion II"):

for $j = n : -1 : 1$

$X(j,j) = L(j,j)^{-1}$

$X(j+1:n, j) = X(j+1:n, j+1:n) * L(j+1:n, j)$

$X(j+1:n, j) = -X(j,j) * X(j+1:n, j)$

end

Now there are n matrix-vector multiplications, where the matrix is triangular. This is in general faster than the previous algorithm.

1. Matrix Inversion

Concerning stability, Column-wise Inversion I is derived from the equation

$$LX = I$$

and the computed inverse $\hat{X}$ therefore satisfies a *right* residual componentwise bound

$$|L\hat{X} - I| \leq c_n \epsilon |L||\hat{X}| + \mathcal{O}(\epsilon^2),$$

where ϵ is the machine precision [Du Croz, Higham - 1992].

Column-wise Inversion II, on the other hand, is derived from the equation $XL = I$ and the computed inverse $\hat{X}$ therefore satisfies a *left* residual componentwise bound

$$|\hat{X}L - I| \leq c_n \epsilon |L||\hat{X}| + \mathcal{O}(\epsilon^2),$$

see [Du Croz, Higham - 1992].

1. Matrix Inversion

For all of the three strategies

1. Compute L^{-1} one column at a time
2. Compute L^{-1} one row at a time
3. Compute L^{-1} by outer product updates

there is a variant derived from $XL = I$ and a variant derived from $LX = I$.

The first always satisfy a left residual componentwise bound (but not always a right one) and the second a right residual componentwise bound (but not always a left one) .

Be aware: Block versions of these algorithms are generally faster, but do not necessarily satisfy this rule! In some cases they do not satisfy a left residual componentwise bound and not a right residual componentwise bound either; they are unstable.

1. Matrix Inversion

Now we proceed to inversion of a general matrix A with triangular factorization

$$PA = LU,$$

where L is unit lower triangular, U is upper triangular and P is a permutation matrix. Without loss of generalization we will discard P. Perhaps the most frequently described method is:

for $j = 1 : n$

solve $Ax_j = e_j$ with the given LU decomposition

end

The method has the disadvantage that the factors L, U are needed during the whole computation and can not be overwritten. It is clearly derived from $AX = I$ and, as expected, satisfies a right componentwise bound,

$$|A\hat{X} - I| \leq c_n \epsilon |L||U||\hat{X}| + \mathcal{O}(\epsilon^2).$$

1. Matrix Inversion

A second method, used in LAPACK, is:

compute U^{-1}

solve for X the equation $XL = U^{-1}$

The method is derived from solving $XLU = I$ and thus satisfies a left componentwise bound,

$$|\hat{X}A - I| \leq c_n \epsilon |L||U||\hat{X}| + \mathcal{O}(\epsilon^2),$$

under the assumption, however, that U^{-1} is computed with an implementation that guarantees a left componentwise bound for U^{-1}. Otherwise, we can only obtain weaker bounds.

1. Matrix Inversion

Yet another method follows from solving

$$UXL = I$$

for X.

It enables overwriting L and U with X and thus is efficient with respect to storage costs. Computational costs are dominated by matrix-vector products, making it the fastest of the presented methods.

However, it only satisfies a „mixed" componentwise bound,

$$|U\hat{X}L - I| \leq c_n \epsilon |L||U||\hat{X}| + \mathcal{O}(\epsilon^2).$$

Right or left componentwise bounds for this method are weaker than for the previous methods.

1. Matrix Inversion

Finally, we can do the following straightforward computation:

$$\text{compute } U^{-1} \text{ and } L^{-1}$$
$$\text{form} \quad X = U^{-1}L^{-1}$$

This methods needs no temporary storage at all. However, the best obtainable left componentwise bound is

$$|\hat{X}A - I| \leq c_n\epsilon|L||U||L^{-1}||U^{-1}| + \mathcal{O}(\epsilon^2),$$

under the assumption that both U^{-1} and L^{-1} are computed with an implementation that guarantees a left componentwise bound for U^{-1} resp. L^{-1}. Analogously we have

$$|A\hat{X} - I| \leq c_n\epsilon|L||U||L^{-1}||U^{-1}| + \mathcal{O}(\epsilon^2),$$

under the assumption that both U^{-1} and L^{-1} are computed with an implementation with a right componentwise bound for U^{-1} resp. L^{-1}.

1. Matrix Inversion

Summarizing,

1. There are many implementations of matrix inversion based on triangular decomposition

2. The implementations may differ significantly in computational and storage costs and in stability properties

3. For stability, it is important to choose the correct variants in implementations that exploit inversion of triangular matrices

4. The stability of block versions can be much worse than their point-wise counterparts.

2. Balanced Incomplete Factorization

Bru, Cerdán, Marín and Mas introduced an LU-type factorization called Inverse Sherman-Morrison (ISM) decomposition (SISC 2003), based on the Sherman-Morrison formula

$$(A + XY^T)^{-1} = A^{-1} - A^{-1}X(I_k + Y^T A^{-1} X)^{-1} Y^T A^{-1},$$

valid for $A \in \mathbb{R}^{n \times n}$ and rectangular rank-k matrices $X, Y \in \mathbb{R}^{n \times k}$, $k \leq n$.

Assume a given, general nonsymmetric matrix A can be written as

$$A = A_0 + \sum_{k=1}^{n} x_k y_k^T$$

for a non-singular matrix A_0 and two sets of vectors $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$ in $\mathbb{R}^n$.

2. Balanced Incomplete Factorization

By repeated application of the Sherman-Morrison formula we obtain the identity

$$A^{-1} = A_0^{-1} - A_0^{-1} U_{A_0} D_{A_0}^{-1} V_{A_0}^T {A_0}^{-1},$$

where the columns of U_{A_0} and V_{A_0} are computed through

$$u_k = x_k - \sum_{i=1}^{k-1} \frac{v_i^T A_0^{-1} x_k}{r_i} u_i, \quad v_k = y_k - \sum_{i=1}^{k-1} \frac{y_k^T A_0^{-1} u_i}{r_i} v_i,$$

the denominators r_i are given by

$$r_i = 1 + y_i^T {A_0}^{-1} u_i = 1 + v_i^T {A_0}^{-1} x_i, \quad i = 1, \ldots, n.$$

and

$$D_{A_0} = \mathsf{diag}(r_1, \ldots, r_n).$$

2. Balanced Incomplete Factorization

Consider the special choices of A_0, $(x_k)_{k=1}^n$ and $(y_k)_{k=1}^n$,

$$A_0 = sI_n,\ s > 0, \quad x_k = e_k, \quad y_k = (a^{(k)})^T - se_k, \quad \Rightarrow A = sI_n + \sum_{k=1}^{n} e_k \left((a^{(k)})^T - se_k \right)^T$$

where $a^{(k)}$ denotes the k-th row of A. Then we obtain the identity

$$A^{-1} = s^{-1} I - s^{-2} U_s D_s^{-1} V_s^T,$$

where the columns u_k of U_s and v_k of V_s are computed as

$$u_k = x_k - \sum_{i=1}^{k-1} \frac{e_k^T v_i}{s r_i^{(s)}} u_i, \quad v_k = y_k - \sum_{i=1}^{k-1} \frac{y_k^T u_i}{s r_i^{(s)}} v_i.$$

Clearly, U_s is unit upper triangular.

2. Balanced Incomplete Factorization

For U_s, D_s and V_s there holds [Bru, Cerdán, Marín, Mas - 2003]:

$$U_s = U_1, \qquad D_s = s^{-1} D_1, \qquad V_s = V_1 - (s-1)W,$$

where the columns of the auxiliary matrix W satisfy the recurrence

$$w_k = x_k - \sum_{i=1}^{k-1} \frac{y_k^T u_i}{r_i^{(1)}} w_i, \quad k = 1, \ldots, n.$$

In particular, W is, like U, unit upper triangular.

Now we have

$$\begin{aligned} A^{-1} &= s^{-1} I - s^{-2} U_s D_s^{-1} V_s^T \\ &= s^{-1} I - U_1 (s^{-1} D_s^{-1})(s^{-1} V_s^T) = s^{-1} I - U_1 D_1^{-1} \left(s^{-1} V^T - (1 - s^{-1}) W^T \right). \end{aligned}$$

For $s \to \infty$ this gives

2. Balanced Incomplete Factorization

$$A^{-1} = U_1 D_1^{-1} W^T.$$

This is *the* unique LDU decomposition of A^{-1} and U_1 and W are its triangular factors. Moreover,

$$U_1 D_1^{-1} W^T = s^{-1} I - s^{-1} U_1 D_1^{-1} V_s^T,$$

i.e. by multiplication with s,

$$U_1 D_1^{-1} V_s^T = I - s U_1 D_1^{-1} W^T, \qquad \text{i.e.} \quad V_s^T = D_1 U_1^{-1} - s W^T$$

Pictorially,

$$V_s^T = \begin{pmatrix} \ddots & & D_1 U_1^{-1} \\ & \ddots & \\ -sW^T & & \ddots \end{pmatrix}, \quad \text{diag}(V_s) = D_1 - sI.$$

2. Balanced Incomplete Factorization

From

$$A^{-1} = U_1 D_1^{-1} W^T$$

we have

$$A = W^{-T} D_1 U_1^{-1}$$

We see that the matrix V_s computed during the ISM process contains the factor W of the LDU decomposition of A^{-1} *and* the factor U_1^{-1} of the LDU decomposition of A. For A symmetric positive definite this means that V_s contains both the Cholesky factor $U_1^{-1} = L^T$ of A and its inverse $W^T = L^{-1}$!

In [Bru, Marín, Mas, Tůma - 2008] the presence of the inverse Cholesky factor is exploited for the construction of a robust incomplete factorization called Balanced Incomplete Factorization (BIF). The main idea is to mutually balance the dropping of entries in the Cholesky and inverse Cholesky factor and control their conditioning in this way.

2. Balanced Incomplete Factorization

Denote by L the (exact) Cholesky factor of A and by $\tilde{L}$ an incomplete Cholesky factor of A. Then a robust dropping criterium for $\tilde{L}_{jk}$ is

$$|\tilde{L}_{jk}|\|e_k^T L^{-1}\| \leq \tau$$

for some drop tolerance τ [Bollhöfer, Saad - 2002].

Reversely, if $\tilde{W}$ denotes an incomplete Cholesky factor of A^{-1}, then a robust strategy is to drop an entry $\tilde{W}_{jk}$ when

$$|\tilde{W}_{jk}|\|e_k^T L\| \leq \tau.$$

We will consider in the following balanced *complete* factorization, i.e. we use the ISM process to compute the factors L and W without any dropped entries.

3. Condition estimation

The main question is: How can the presence of the inverse factors in Balanced Factorization be exploited ?

Perhaps the first thing that comes to mind, is to use the inverse triangular factors for improved condition estimation.

We will see that exploiting the inverse factors for better condition estimation is possible, but not as straightforward as it may seem.

3. Condition estimation

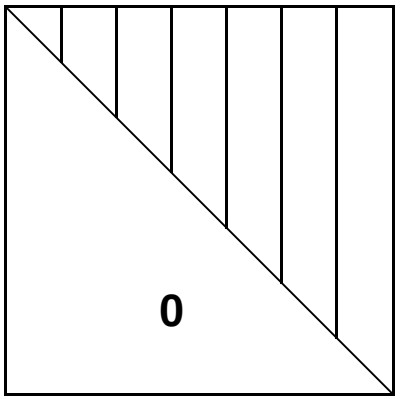
We assume A is real and positive definite symmetric. If

$$A = LL^T$$

is the Cholesky decomposition of A, the condition number of A satisfies

$$\kappa(A) = \kappa(L)^2 = \kappa(L^T)^2.$$

We focus on estimation of the 2-norm condition number of L^T. This can be done cheaply with a technique called *incremental* condition number estimation. Main idea: Subsequent estimation of leading submatrices:

$L^T =$ 

$\Rightarrow$ all columns are accessed only once.

3. Condition estimation

We will call the original incremental technique, introduced by Bischof [Bischof - 1990], simply incremental condition estimation (ICE):

Let R be upper triangular with a given approximate maximal singular value $\sigma_{maxICE}(R)$ and corresponding approximate singular vector y, $\|y\| = 1$,

$$\sigma_{maxICE}(R) = \|y^T R\| \approx \sigma_{max}(R) = \max_{\|x\|=1} \|x^T R\|.$$

ICE approximates the maximal singular value of the extended matrix

$$R' = \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix}$$

by maximizing

$$\left\| \begin{pmatrix} sy, & c \end{pmatrix} \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \right\|, \quad \text{over all } s, c \text{ satisfying} \quad c^2 + s^2 = 1.$$

3. Condition estimation

We have

$$\max_{s,c,c^2+s^2=1} \left\| \begin{pmatrix} sy, & c \end{pmatrix} \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \right\|^2 = \max_{s,c,c^2+s^2=1} \begin{pmatrix} sy, & c \end{pmatrix} \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \begin{pmatrix} R^T & 0 \\ v^T & \gamma \end{pmatrix} \begin{pmatrix} sy \\ c \end{pmatrix}$$

$$= \max_{s,c,c^2+s^2=1} \begin{pmatrix} s, & c \end{pmatrix} \begin{pmatrix} \sigma_{maxICE}(R)^2 + (y^T v)^2 & \gamma(v^T y) \\ \gamma(v^T y) & \gamma^2 \end{pmatrix} \begin{pmatrix} s \\ c \end{pmatrix}.$$

The maximum is obtained with the normalized eigenvector corresponding to the maximum eigenvalue $\lambda_{max}(B_{ICE})$ of

$$B_{ICE} \equiv \begin{pmatrix} \sigma_{maxICE}(R)^2 + (y^T v)^2 & \gamma(v^T y) \\ \gamma(v^T y) & \gamma^2 \end{pmatrix}.$$

We denote the normalized eigenvector by $\begin{pmatrix} \hat{s} \\ \hat{c} \end{pmatrix}$.

3. Condition estimation

Then with $\hat{y}^T = \begin{pmatrix} \hat{s}y, & \hat{c} \end{pmatrix}$ we define the approximate maximal singular value of the extended matrix as

$$\sigma_{maxICE}(R') \equiv \|\hat{y}^T R'\| \approx \sigma_{max}(R').$$

Similarly, if for some y with unit norm,

$$\sigma_{minICE}(R) = \|y^T R\| \approx \sigma_{min}(R) = \min_{\|x\|=1} \|x^T R\|,$$

then ICE uses the *minimal* eigenvalue $\lambda_{min}(B_{ICE})$ of

$$B_{ICE} = \begin{pmatrix} \sigma_{minICE}(R)^2 + (y^T v)^2 & \gamma(v^T y) \\ \gamma(v^T y) & \gamma^2 \end{pmatrix}$$

The corresponding eigenvector of B_{ICE} yields the new vector $\hat{y}^T$ and

$$\sigma_{minICE}(R') \equiv \|\hat{y}^T R'\| \approx \sigma_{min}(R').$$

3. Condition estimation

Experiment:

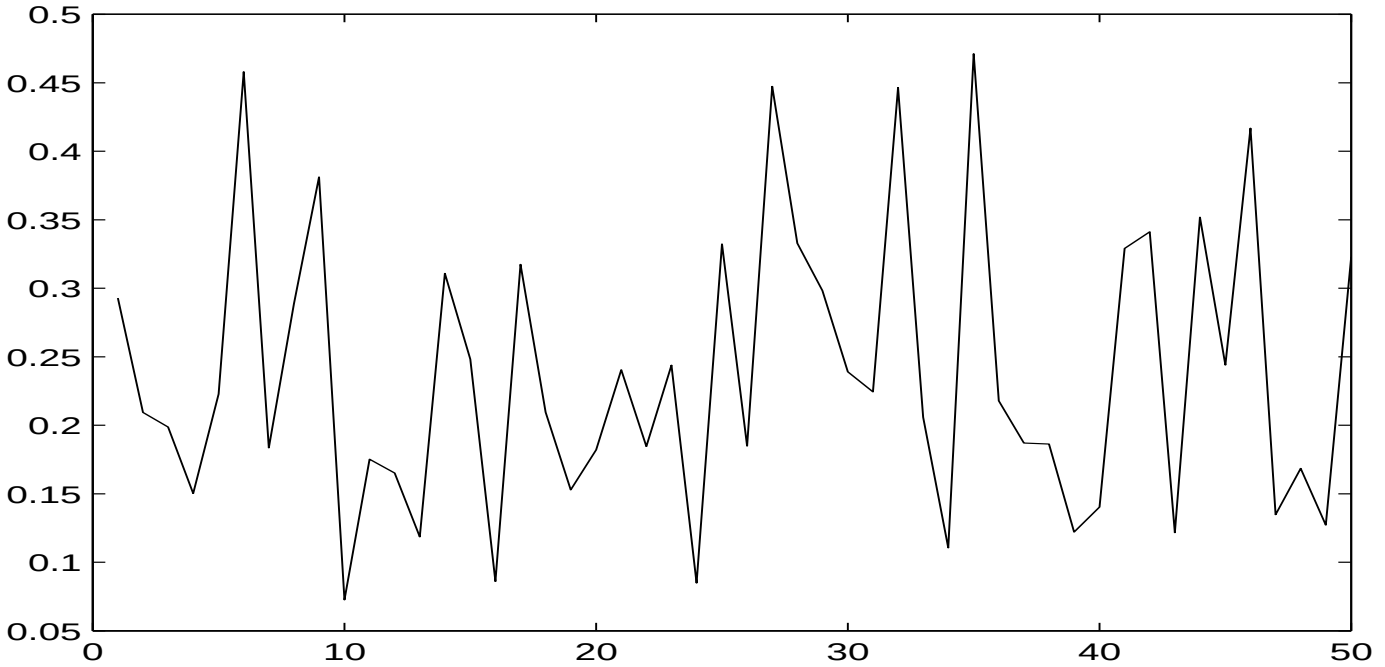
- We generate 50 random matrices B of dimension 100 with the Matlab command B $=$ randn$(100, 100)$
- We compute the Cholesky decompositions LL^T of the 50 symmetric positive definite matrices $A = BB^T$
- We compute the estimations $\sigma_{maxICE}(L^T)$ and $\sigma_{minICE}(L^T)$
- In the following graph we display the quality of the estimations through the number

$$\frac{\left(\frac{\sigma_{maxICE}(L^T)}{\sigma_{minICE}(L^T)}\right)^2}{\kappa(A)},$$

where $\kappa(A)$ is the true condition number. Note that we always have

$$\left(\frac{\sigma_{maxICE}(L^T)}{\sigma_{minICE}(L^T)}\right)^2 \leq \kappa(A).$$

3. Condition estimation



Quality of the estimator ICE for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation

Now assume we have to our disposal not only the Cholesky decomposition of A,

$$A = LL^T$$

but also the inverse Cholesky factors as is the case in balanced factorization, i.e. we have

$$A^{-1} = L^{-T}L^{-1}.$$

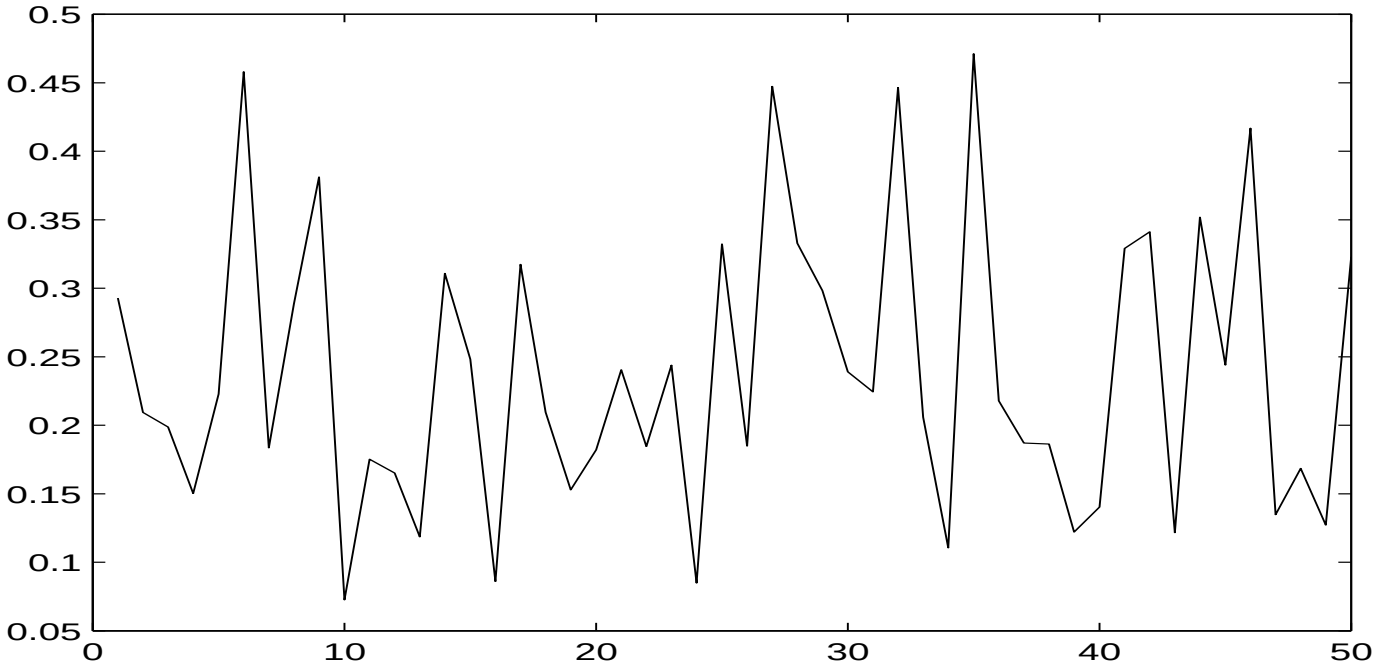
Then we can run ICE on L^{-T} and use the additional estimations

$$\frac{1}{\sigma_{maxICE}(L^{-T})} \approx \sigma_{min}(L^T), \qquad \frac{1}{\sigma_{minICE}(L^{-T})} \approx \sigma_{max}(L^T).$$

In the following graph we take the best of both estimations, we display

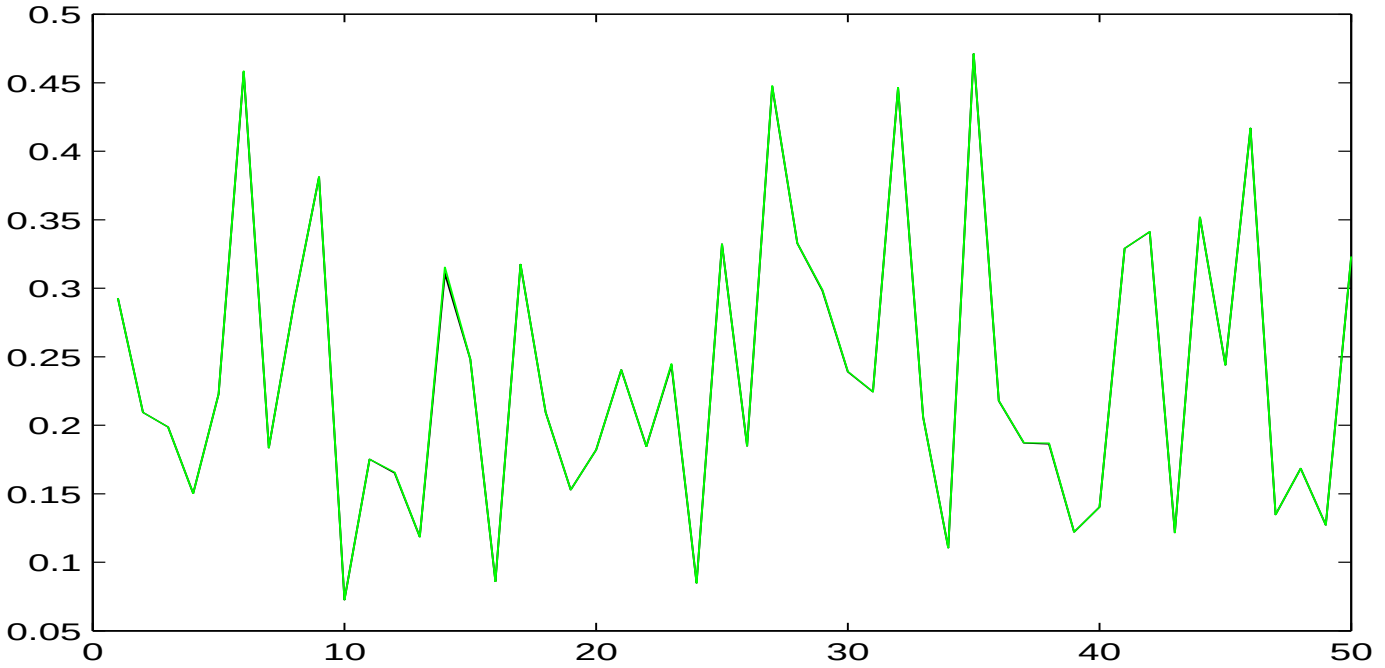
$$\frac{\left(\frac{\max(\sigma_{maxICE}(L^T),\sigma_{minICE}(L^{-T})^{-1})}{\min(\sigma_{minICE}(L^T),\sigma_{maxICE}(L^{-T})^{-1})}\right)^2}{\kappa(A)}.$$

3. Condition estimation



Quality of the estimator ICE for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation



Quality of the estimator ICE without (black) and with exploiting (green) the inverse for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation

An alternative technique called incremental norm estimation (INE) was proposed in [Duff, Vömel - 2002]:

Let R be upper triangular with given approximate maximal singular value $\sigma_{maxINE}(R)$ and corresponding approximate *right* singular vector z, $\|z\| = 1$,

$$\sigma_{maxINE}(R) = \|Rz\| \approx \sigma_{max}(R) = \max_{\|x\|=1} \|Rx\|.$$

INE approximates the maximal singular value of the extended matrix

$$R' = \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix}$$

by maximizing

$$\left\| \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \begin{pmatrix} sz \\ c \end{pmatrix} \right\|, \quad \text{over all } s, c \text{ satisfying } c^2 + s^2 = 1.$$

3. Condition estimation

We have

$$\max_{s,c,c^2+s^2=1} \left\| \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \begin{pmatrix} sz \\ c \end{pmatrix} \right\|^2 = \max_{s,c,c^2+s^2=1} \begin{pmatrix} sz \\ c \end{pmatrix} \begin{pmatrix} R^T & 0 \\ v^T & \gamma \end{pmatrix} \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix} \begin{pmatrix} sz \\ c \end{pmatrix}$$

$$= \max_{s,c,c^2+s^2=1} \begin{pmatrix} s & c \end{pmatrix} \begin{pmatrix} z^T R^T R z & z^T R^T v \\ z^T R^T v & v^T v + \gamma^2 \end{pmatrix} \begin{pmatrix} s \\ c \end{pmatrix}.$$

The maximum is obtained for the normalized eigenvector corresponding to the maximum eigenvalue $\lambda_{max}(B_{INE})$ of

$$B_{INE} \equiv \begin{pmatrix} z^T R^T R z & z^T R^T v \\ z^T R^T v & v^T v + \gamma^2 \end{pmatrix}.$$

We denote the normalized eigenvector by $\begin{pmatrix} \hat{s} \\ \hat{c} \end{pmatrix}$.

3. Condition estimation

Then with $\hat{z} = \begin{pmatrix} \hat{s}z, & \hat{c} \end{pmatrix}^T$ we define the approximate maximal singular value of the extended matrix as

$$\sigma_{maxINE}(R') \equiv \|R'\hat{z}\| \approx \sigma_{max}(R').$$

Similarly, if for a unit vector z,

$$\|Rz\| \approx \sigma_{min}(R) = \min_{\|x\|=1} \|Rx\|,$$

then INE uses the *minimal* eigenvalue $\lambda_{min}(B_{INE})$ of

$$B_{INE} = \begin{pmatrix} z^T R^T Rz & z^T R^T v \\ z^T R^T v & v^T v + \gamma^2 \end{pmatrix}.$$

The corresponding eigenvector of B_{INE} yields the new vector $\hat{z}$ and

$$\sigma_{minINE}(R') \equiv \|R'\hat{z}\| \approx \sigma_{min}(R').$$

3. Condition estimation

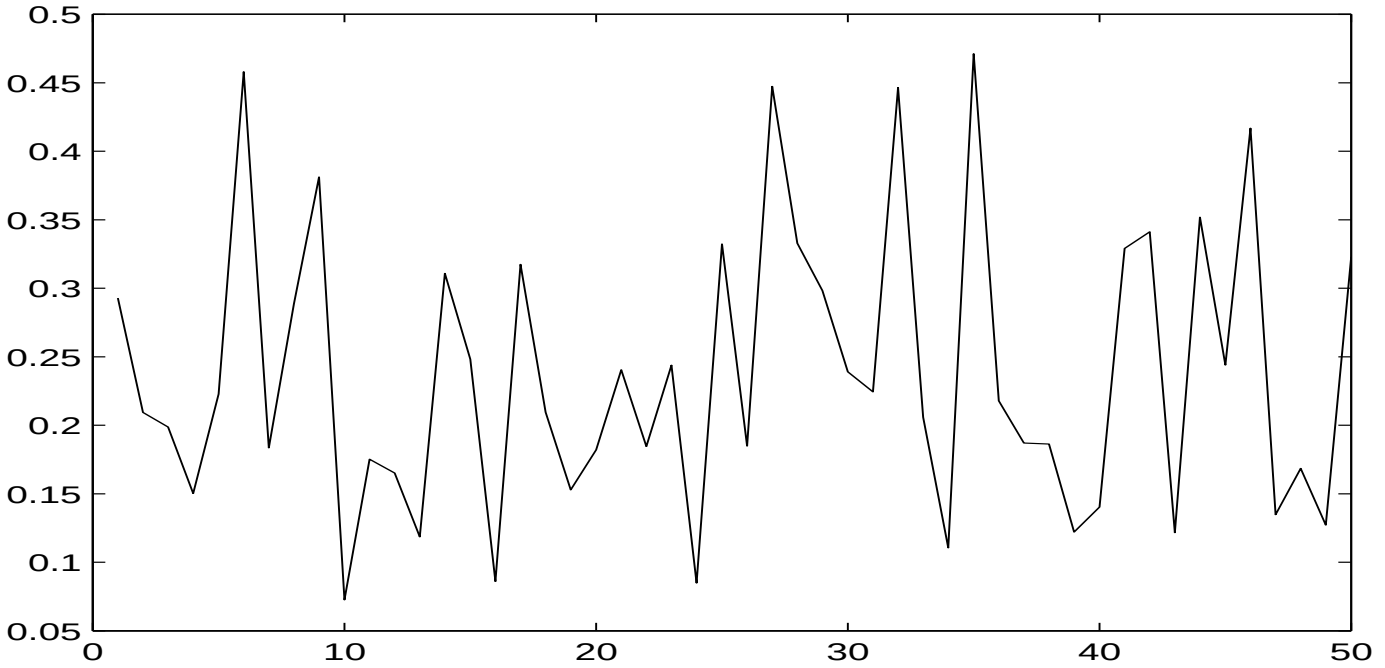
Consider the same experiment as before.

We can combine the estimations from ICE and INE to improve the estimator.

In the following graph we take the best of both estimations and display

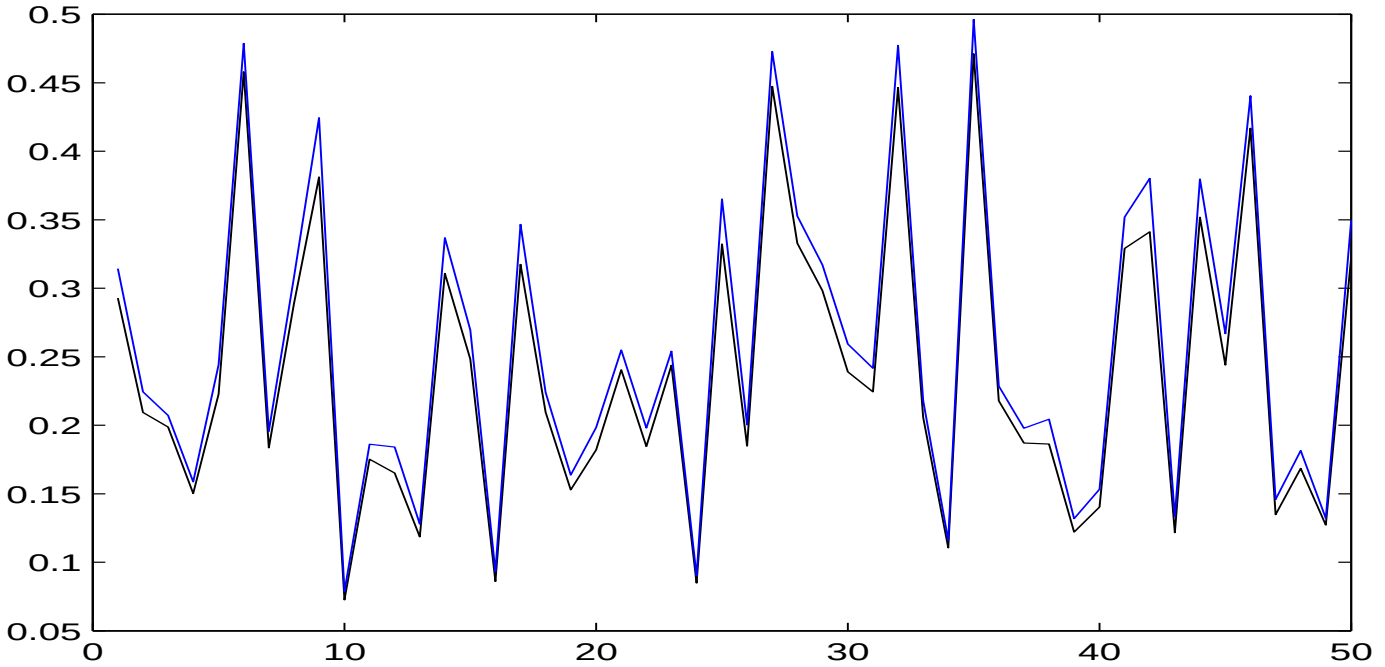
$$\frac{\left(\frac{\max(\sigma_{maxICE}(L^T),\sigma_{maxINE}(L^T))}{\min(\sigma_{minICE}(L^T),\sigma_{minINE}(L^T))}\right)^2}{\kappa(A)}.$$

3. Condition estimation



Quality of the estimator ICE for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation



Quality of the estimator ICE (black) and of ICE combined with INE (blue) for 50 random s.p.d. matrices of dimension 100.

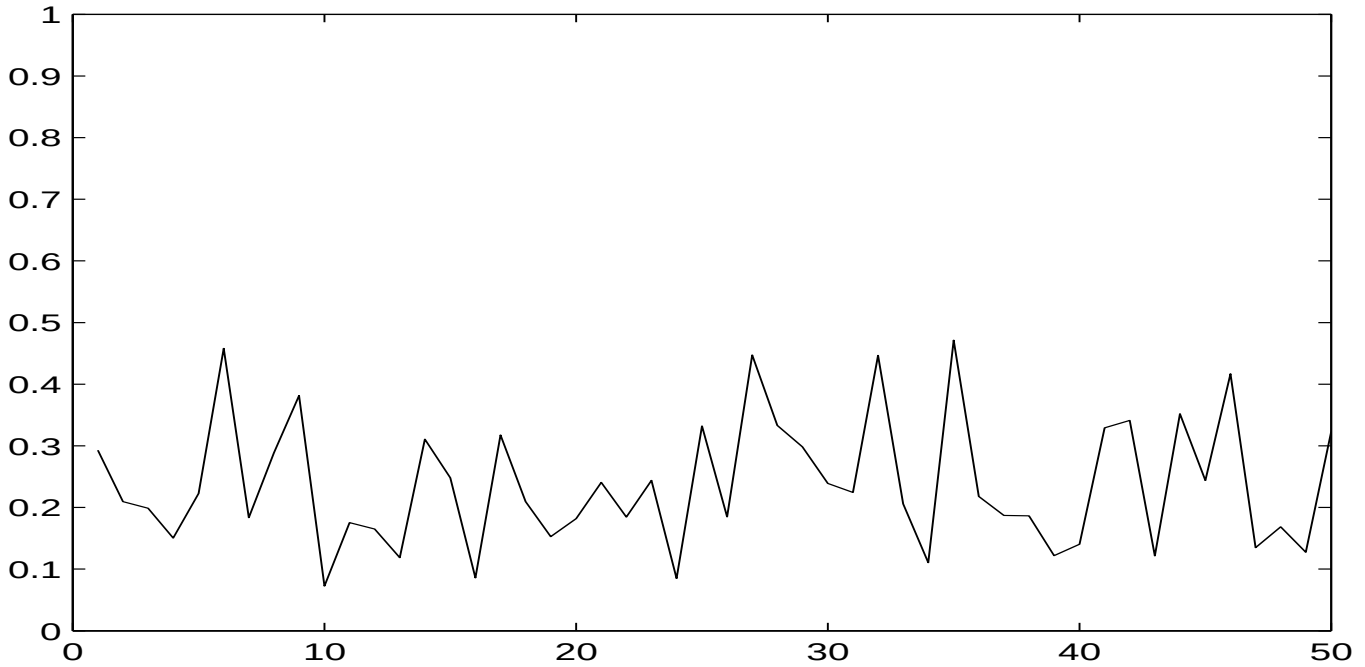
3. Condition estimation

Finally, if we assume we have to our disposal the inverse factors, we can combine ICE with INE for *both* L^T and L^{-T}.

In the following graph we take the best of four estimations and display

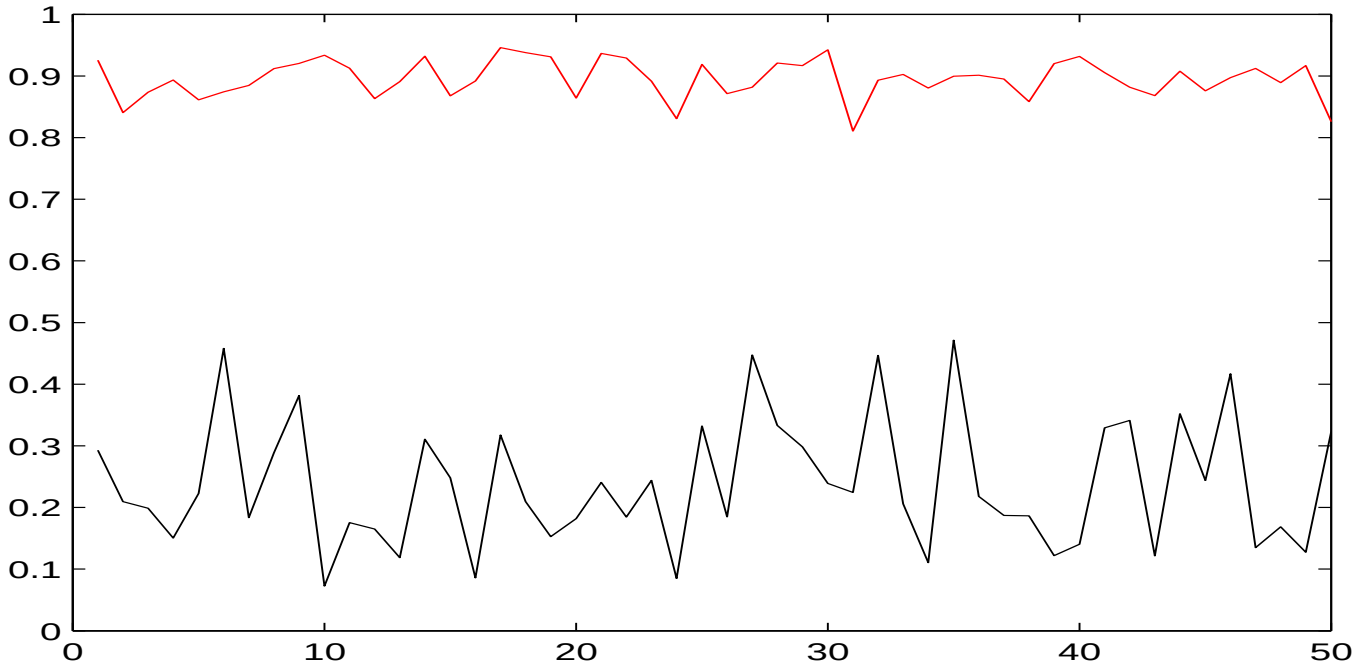
$$\frac{\left(\frac{\max(\sigma_{maxICE}(L^T),\sigma_{maxINE}(L^T),\sigma_{minICE}(L^{-T})^{-1},\sigma_{minINE}(L^{-T})^{-1})}{\min(\sigma_{minICE}(L^T),\sigma_{minINE}(L^T),\sigma_{maxICE}(L^{-T})^{-1},\sigma_{maxINE}(L^{-T})^{-1})}\right)^2}{\kappa(A)}.$$

3. Condition estimation



Quality of the estimator ICE for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation



Quality of the estimator ICE (black) and of ICE combined with INE and exploiting the inverse (red) for 50 random s.p.d. matrices of dimension 100.

3. Condition estimation

Why this improvement ?

- In general, both ICE and INE give a satisfactory approximation of $\sigma_{max}(A)$, though INE tends to be better.
- The problem is to approximate $\sigma_{min}(A)$, for ICE as well as for INE.
- The trick is to translate to the problem of finding the maximal singular value $\sigma_{max}(A^{-1})$ of A^{-1}, which is in general done *better* with INE than with ICE.
- This has an important impact on the estimate because $\sigma_{min}(A)$ is typically small and appears in the denominator of $\frac{\sigma_{max}(A)}{\sigma_{min}(A)}$,

We see that the main reason for the improvement is that INE tends to give a better estimate of maximal singular values. And why is that ?

3. Condition estimation

Note: INE does not *always* give a better estimate of the maximal singular value. But we have the following rather technical result.

Theorem [DT, Tůma - ?]. Consider condition estimation of the matrix

$$R' = \begin{pmatrix} R & v \\ 0 & \gamma \end{pmatrix},$$

where both ICE and INE start with the same approximation of $\sigma_{max}(R)$ denoted by δ. Let y, $\|y\| = 1$ be the approximate singular vector for ICE,

$$\delta = \|y^T R\| \approx \sigma_{max}(R),$$

and let z, $\|z\| = 1$ be the approximate singular vector for INE,

$$\delta = \|Rz\| \approx \sigma_{max}(R).$$

3. Condition estimation

Theorem (continued). Then we have superiority of INE,

$$\sigma_{maxINE}(R') \quad \geq \quad \sigma_{maxICE}(R'),$$

if

$$(v^T R z)^2 \quad \geq \quad \delta^2 (v^T y)^2 + \frac{1}{2}\left(v^T v - (v^T y)^2\right)\left(\alpha - \sqrt{\alpha^2 + 4\delta^2 (v^T y)^2}\right).$$

where $\alpha = \delta^2 - \gamma^2 - (v^T y)^2$.

Hence if $\delta^2 (v^T y)^2 + \frac{1}{2}\left(v^T v - (v^T y)^2\right)\left(\alpha - \sqrt{\alpha^2 + 4\delta^2 (v^T y)^2}\right) \leq 0$, then INE is unconditionally superior to ICE (i.e. regardless of the approximate singular vector z). Let us use the notation

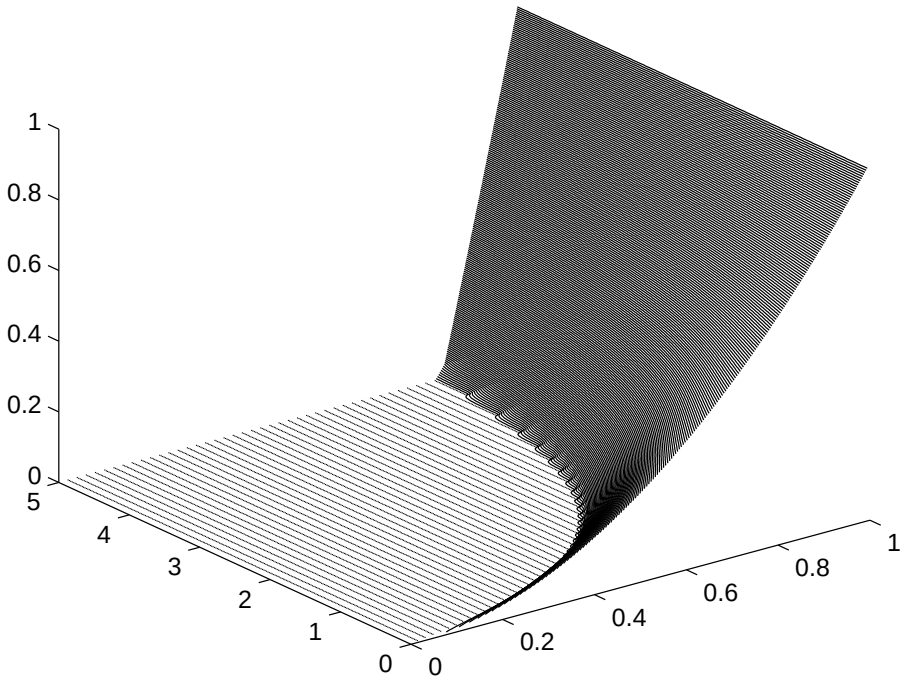
$$\rho \equiv \delta^2 (v^T y)^2 + \frac{1}{2}\left(v^T v - (v^T y)^2\right)\left(\alpha - \sqrt{\alpha^2 + 4\delta^2 (v^T y)^2}\right).$$

3. Condition estimation

To conclude we demonstrate the previous theorem.

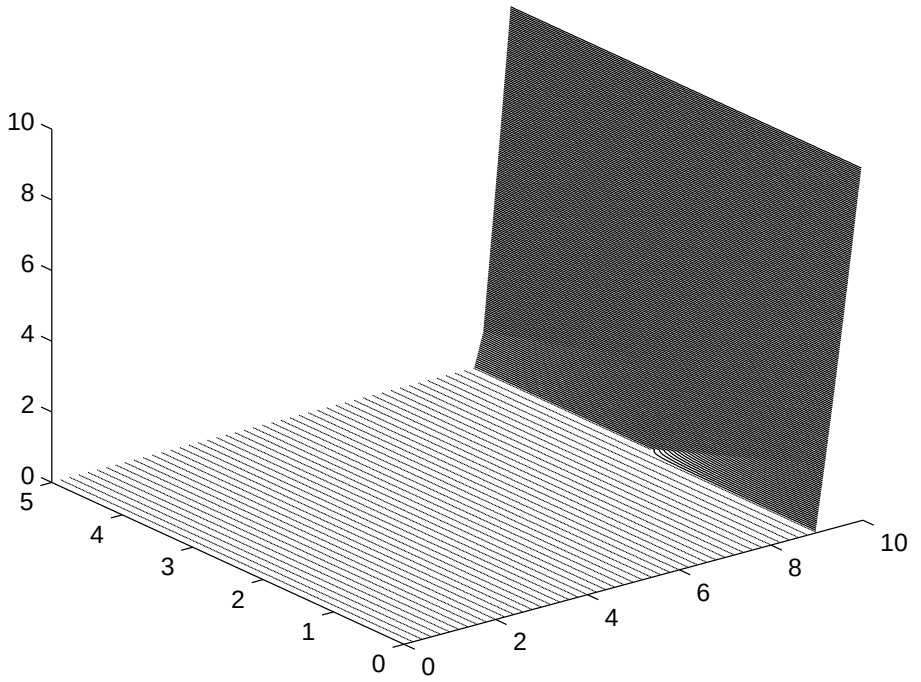
- Assume that at some stage of an incremental condition estimation process we have $\sigma_{maxICE}(R) = \sigma_{maxINE}(R) = 1$.
- Consider possible new columns v of R' that have unit norm, i.e. $v^T v = 1$.
- Then $(v^T y)^2 \leq 1$. The x-axes of the following figures represent the possible values of $(v^T y)^2 < 1$.
- The y-axes represent values of γ^2, i.e. the square of the new diagonal entry.
- The superiority criterion for INE expressed by the value of ρ is given by the z-axes.

3. Condition estimation



Value of ρ in dependence of $(v^T y)^2$ (x-axis) and γ^2 (y-axis) with $\|v\|^2 = 1$.

3. Condition estimation



Value of ρ in dependence of $(v^T y)^2$ (x-axis) and γ^2 (y-axis) with $\|v\|^2 = 10$.

3. Condition estimation

- Exploiting the presence of inverse factors *combined* with INE gives a significant improvement of incremental condition estimation.
- This may be an important advantage of methods where inverse triangular factors are just a by-product of the factorization.
- We did not consider sparse Cholesky factors, which ask for modified ICE [Bischof, Pierce, Lewis - 1990].
- We did not consider exploiting the inverse in estimation of the 1-norm and other non-Euclidean condition number.

Thank you for your attention!

Supported by project number IAA100300802 of the grant agency of the Academy of Sciences of the Czech Republic.