ETH Lecture 401-0663-00L Numerical Methods for CSE

# Mid-term Exam

Autumn Term 2020

Oct 31, 2020, 10:00, HG E1.2 / HG E7 / HG F1



Family Name		%
First Name		
Department		
Legi Nr.		
Date	Oct 31, 2020	

### Points:

	1	2	3	Total
max	10	9	8	
achvd				

- This is a closed-book exam.
- Keep only writing material and your ETH ID card on the table.
- Keep mobile phones, tablets, smartwatches, etc. turned off in your bag.
- Fill in this cover sheet first.
- Turn the cover sheet only when instructed to do so.
- Then write your name and ETH ID number on every page.
- Write your answers in the appropriate fields on these problem sheets.
- Wrong ticks in multiple-choice boxes will lead to points being subtracted. Hence, mere guessing is really dangerous! If you have no clue, leave all tickboxes empty.
- If you change your mind about an answer to an MC-question, write a clear NO next to the old answer, draw fresh tickboxes and fill them.
- Anything written outside the answer boxes will not be taken into account.
- Do not write with red/green color or with pencil.
- Make sure to hand in every sheet.
- Two blank pages handed out with the exam: space for notes
- Duration: 30 minutes.

Throughout the exam use the notations introduced in class, in particular [Lecture  $\rightarrow$  Section 1.1.1]:

•  $(\mathbf{A})_{i,j}$  to refer to the entry of the matrix  $\mathbf{A} \in \mathbb{K}^{m,n}$  at position (i,j).

- $(\mathbf{A})_{:,i}$  to designate the *i*th-column of the matrix  $\mathbf{A}$ ,
- $(\mathbf{A})_{i,:}$  to denote the *i*-th row of the matrix  $\mathbf{A}$ ,
- $(\mathbf{A})_{i:j,k:\ell}$  to single out the sub-matrix  $\left[ (\mathbf{A})_{r,s} \right]_{\substack{i \leq r \leq j \\ k < s < \ell}}$  of the matrix  $\mathbf{A}$ ,
- $(\mathbf{x})_k$  to reference the k-th entry of the vector  $\mathbf{x}$ ,
- ullet  $\mathbf{e}_{j} \in \mathbb{R}^{n}$  to write the j-th Cartesian coordinate vector,
- I to denote the identity matrix,
- O to write a zero matrix,
- $\mathcal{P}_n$  for the space of (univariate polynomials of degree  $\leq n$ ),
- ullet and superscript indices in brackets to denote iterates:  ${f x}^{(k)},$  etc.

By default, vectors are regarded as column vectors.

## **Problem 0-1: Economical QR-decomposition**

The economical QR-decomposition is a fundamental matrix factorization in numerical linear algebra. This problem reviews some of its properties.

This problems is based on the contents of [Lecture  $\rightarrow$  Section 3.3.3].

Given a matrix  $A \in \mathbb{R}^{m,n}$ , m > n,  $A \neq O$ , we write decomposition.

 ${f A}={f Q}{f R}$  for its economical (thin)  ${f Q}{f R}$ 

**(0-1.a)** • (2 pts.) Fill

Fill in the correct matrix sizes



SOLUTION of (0-1.a):

$$\mathbf{Q} \in \mathbb{R}^{m,n}$$
 ,  $\mathbf{R} \in \mathbb{R}^{n,n}$  .

**(0-1.b)**  $\odot$  (4 pts.) Decide, whether the following statements are true for any  $\mathbf{A} \in \mathbb{R}^{m,n} \setminus \{\mathbf{O}\}$ , m > n.

1. The matrix **Q** is an orthogonal matrix.





2. The columns of Q provide an orthonormal basis of  $\mathcal{R}(A)$ .





3. rank(**R**) = n.





4. rank(**Q**) = n.



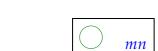
SOLUTION of (0-1.b):

1. NO: because **Q** is not even square in general.

2. NO: Consider a rank-1 matrix A with  $n \ge 2$ . Then Q has more columns than  $\dim \mathcal{R}(A)$ . As a consequence, the columns of Q cannot be a basis of  $\mathcal{R}(A)$ .

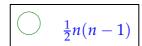
- 3. NO: because  $rank(\mathbf{R}) = rank(\mathbf{A})$ , but  $rank(\mathbf{A}) < n$  is not excluded.
- 4. YES: because the *n* orthonormal columns of  $Q \in \mathbb{R}^{m,n}$  are linearly independent.

How many left-multiplications by Householder matrices does it take to compute



R?





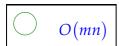


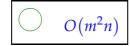
SOLUTION of (0-1.c):

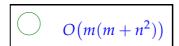
We need n left-multiplications with Householder matrices in order to annihilate the bottom entries of the n columns of A.

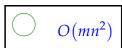
Note that m > n was assumed. In the case  $m \le n$  the computation of  $\mathbf{R}$  requires only m-1 left-multiplications with Householder matrices.

**(0-1.d)** • (2 pts.) What is the asymptotic complexity of EIGEN's HouseholderQR() methods applied to dense  $m \times n$  matrices, m > n, for  $m, n \to \infty$ ?









SOLUTION of (0-1.d):

 $O(mn^2)$  according to [Lecture  $\rightarrow$  § 3.3.3.37].

End Problem 0-1, 10 pts.

#### Problem 0-2: Evaluation of a quadratic form in COO format

This problem examines the algorithmic realization of the evaluation of a quadratic form described by a sparse matrix in triplet/COO storage format.

Familiarity with C++/EIGEN and [Lecture  $\rightarrow$  Section 2.7.1] is required.

In an EIGEN-based numerical code a sparse matrix in COO format is represented by the data type

```
using COOMat = std::vector < Eigen::Triplet < double >>;
```

This function converts a matrix in triplet/COO format into a dense matrix format:

```
Eigen::MatrixXd cooToDense(const COOMat &A) {
    Eigen::Index m = 0;
    Eigen::Index n = 0;
    for (const Eigen::Triplet<double> &t : A) {
        m = (m > t.row()) ? m : t.row() + 1;
        n = (n > t.row()) ? n : t.col() + 1;
    }
    Eigen::MatrixXd M = Eigen::MatrixXd::Zero(m, n);
    for (const Eigen::Triplet<double> &t : A) {
        M(t.row(), t.col()) += t.value();
    }
    return M;
}
```

Supplement the missing parts of C++ code in the boxes.

SOLUTION of (0-2.a):

```
C++ code 0.2.1: Implementation of evalQuadFormCOO()
```

```
double evalQuadFormCOO(const COOMat &A, const Eigen::VectorXd &x) {
    double s = 0.0;
    for (const Eigen::Triplet < double > &t : A) {
        assert((t.row() < x.size()) && (t.col() < x.size()));
        s += (1.0 - x[t.row()]) * x[t.col()] * t.value();
    }
    return s;</pre>
```

9 }

End Problem 0-2, 9 pts.

#### Problem 0-3: Analysis of an EIGEN-based C++ code

In this problem you are asked to analyze a complex code and determine the computational cost of certain operations and the size of matrices stored in certain variables.

This problem assumes familiarity with EIGEN.

In a project you are expected to understand the following undocumented C++ function, which takes two arguments, a matrix  $\mathbf{A} \in \mathbb{R}^{m,n}$ ,  $m \ge n$ , and an integer d < n.

## C++ code 0.3.1: A complex EIGEN-based C++ code

```
std::pair < Eigen:: VectorXd, Eigen:: VectorXd > clsq2(const Eigen:: MatrixXd &A,
                                                        const unsigned d) {
3
     unsigned int n = A.cols(), m = A.rows();
     assert(m >= n);
5
     const Eigen::MatrixXd R =
6
       A. householderQr().matrixQR().template triangularView < Eigen::Upper > (); //
     const auto V = R.block(d, d, m-d, n-d) //
8
                         .jacobiSvd (Eigen :: ComputeFullV) //
9
                         . matrixV(); //
10
     const auto y = V.col(n-d-1); //
11
     const auto b = R.block(0, d, d, n-d) * y; //
12
     const auto S = R.topLeftCorner(d, d); //
13
     const Eigen::VectorXd D = S.diagonal().cwiseAbs(); //
14
     if (D.minCoeff() < (numeric_limits < double > :: epsilon()) * D.maxCoeff()) //
15
       throw runtime error("Upper left block of R not regular"); //
16
     const auto z = -(S.template triangularView < Eigen :: Upper > ()) . solve(b); //
17
     return {z, y};
18
19
```

(0-3.a) (4 pts.) Find out the type of matrix/vector stored in various variables

(i)	R defined in Line 7	stores matrix/vector	$\in \mathbb{R}$
(ii)	∨ defined in Line 8	stores matrix/vector	$\in \mathbb{R}$ ,
(iii)	b defined in Line 12	stores matrix/vector	$\in \mathbb{R}$
(iv)	z defined in Line 17	stores matrix/vector	$\in \mathbb{R}$

HINT 1 for (0-3.a): The method block (int i, int j, int p, int q) of a matrix type in EIGEN returns a reference to the submatrix with upper left corner at position (i, j) and size  $p \times q$ .

SOLUTION of (0-3.a):

(i) R is the upper triangular part of the **R**-factor of the QR-decomposition of the matrix  $\mathbf{A} \in \mathbb{R}^{m,n}$ . It has the same size as **R** and stores an  $m \times n$ -matrix.

(ii) v is the V-factor of the SVD of a  $(m-d) \times (n-d)$ -matrix, hence of size  $(n-d) \times (n-d)$ .

- (iii) b stores a column vector of length d obtained by the multiplication of a  $d \times (n-d)$ -matrix (in V) with and n-d-vector (in y).
- (iv) S contains a  $d \times d$ -matrix  $\mathbf{S} \in \mathbb{R}^{d,d}$ . Hence z holds the vector  $\mathbf{S}^{-1}\mathbf{b} \in \mathbb{R}^d$ .

▲

**(0-3.b)**  $\odot$  (4 pts.) Assuming that d is small and fixed, determine the asymptotic complexity in terms of  $m, n \to \infty$  of the statements in different lines of Code 0.3.1

- (i) Line 7 Cost = O(
- (ii) Line 8–Line 10 Cost = O(
- (iii) Line 14 Cost = O(
- (iv) Line 17 Cost = O(

SOLUTION of (0-3.b):

- (i) Line 7: Computation of the QR-decomposition of  $A \in \mathbb{R}^{m,n}$ : cost =  $O(mn^2)$ , see [Lecture  $\rightarrow$  § 3.3.3.37].
- (ii) Line 8-Line 10: Computation of the SVD of an  $(m-d)\times(n-d)$ -matrix,  $m\geq n$ , which is an economical SVD, because the full Q-factor is not requested: Cost =  $O(mn^2)$ , see [Lecture  $\rightarrow$  § 3.4.2.2].
- (iii) Line 14: Extraction of a d-vector by taking the absolute values of the entries of the diagonal of a  $d \times d$ -matrix. Cost = O(1), because d was supposed to be fixed and small.
- (iv) Line 17: Solution of a small  $d \times d$  linear system of equations. Cost = O(1), because d was supposed to be fixed and small.

•

End Problem 0-3, 8 pts.